**Android Part**

1. Set capture number as 51.

2. Get range of sensitivities by using ‘CameraCharacteristics.SENSOR\_INFO\_SENSITIVITY\_RANGE’. Get lower and higher sensitivity by using ‘Integer lower\_sstt = sstt.getLower();’ and ‘Integer higher\_sstt = sstt.getUpper();’. Change the capture request's sensitivity as ‘lower\_sstt’ and ‘higher\_sstt’.

3. Then start to take pictures.

**Matlab Part**

1. I load the raw images into MATLAB. The image is too large, so I sample a part of them.

For example, two of the sampled raw images of minimum and maximum sensitivity setting are like this:

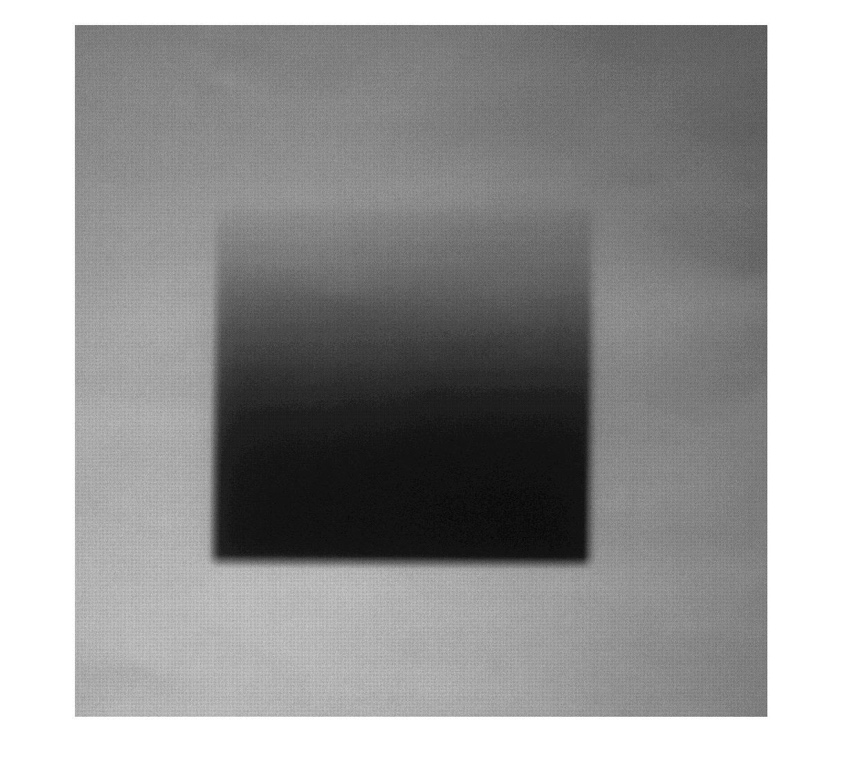


Figure 1 sample a part of raw image of minimum sensitivity setting

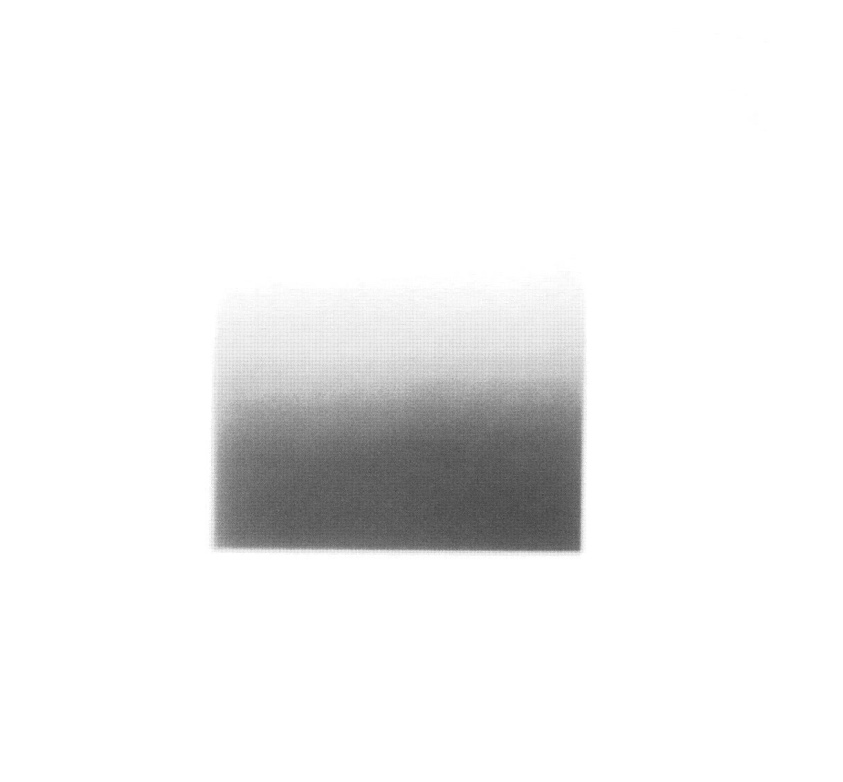


Figure 2 sample a part of raw image of maximum sensitivity setting

**Problems I encountered:**

At first, I don’t know how to implement ‘t = Tiff(filename,'r') im = read(t) function’, after discussing with my teammates, I know that I need to find the path of my images and let filename = sprintf('path/%d.dng',i), and use looping to read all images.

2. I plot a histogram of these pixel values for a given pixel. A few histograms are as follows.

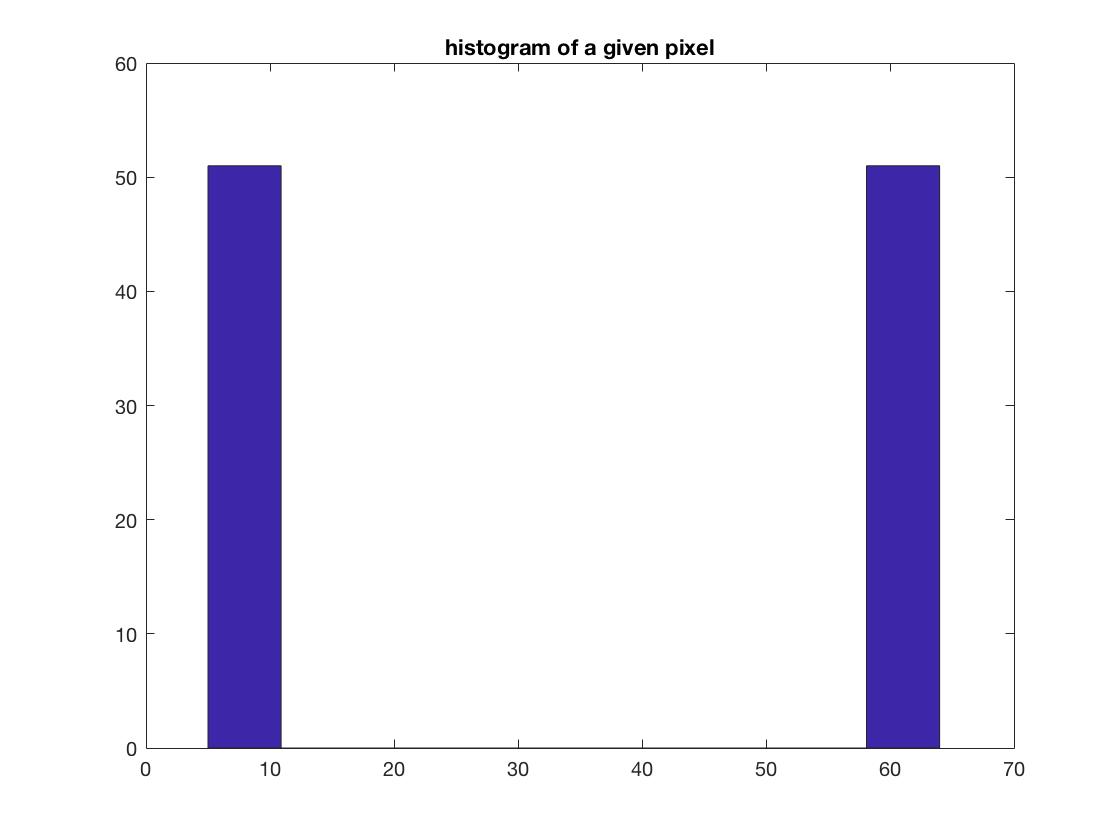


Figure 3 histogram of pixel(500,700)

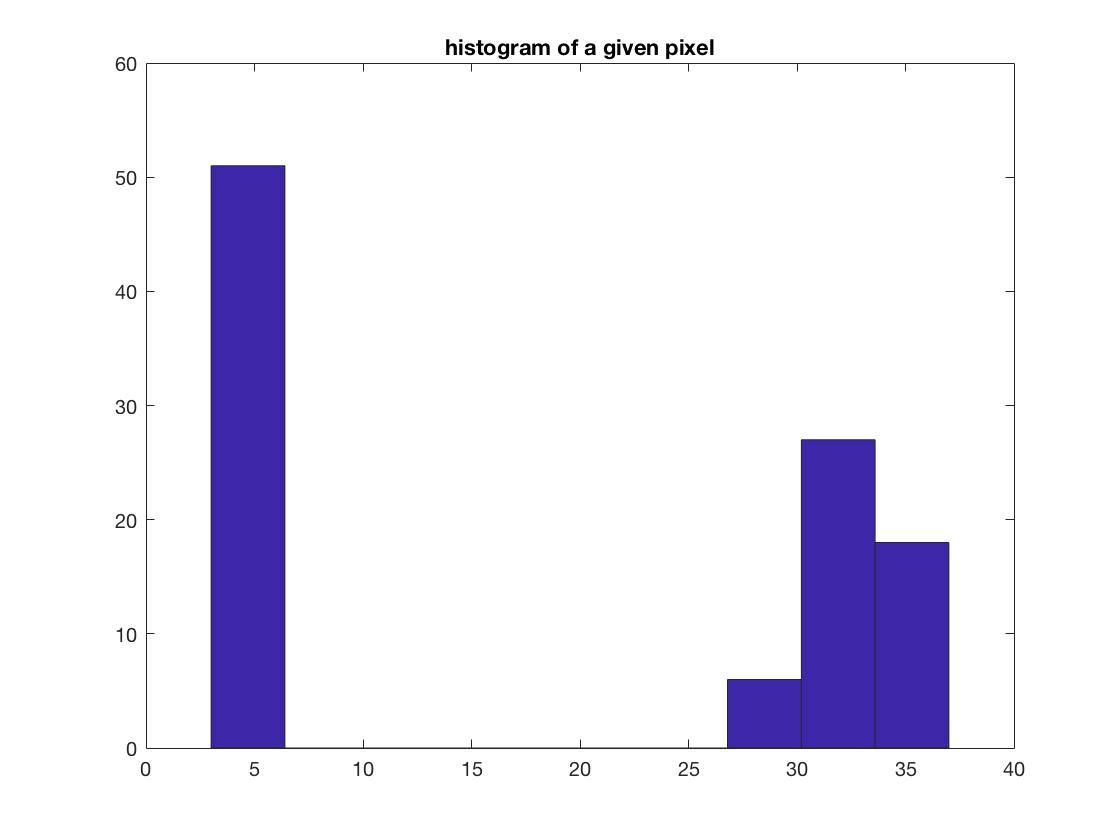


Figure 4 histogram of pixel(600,500)

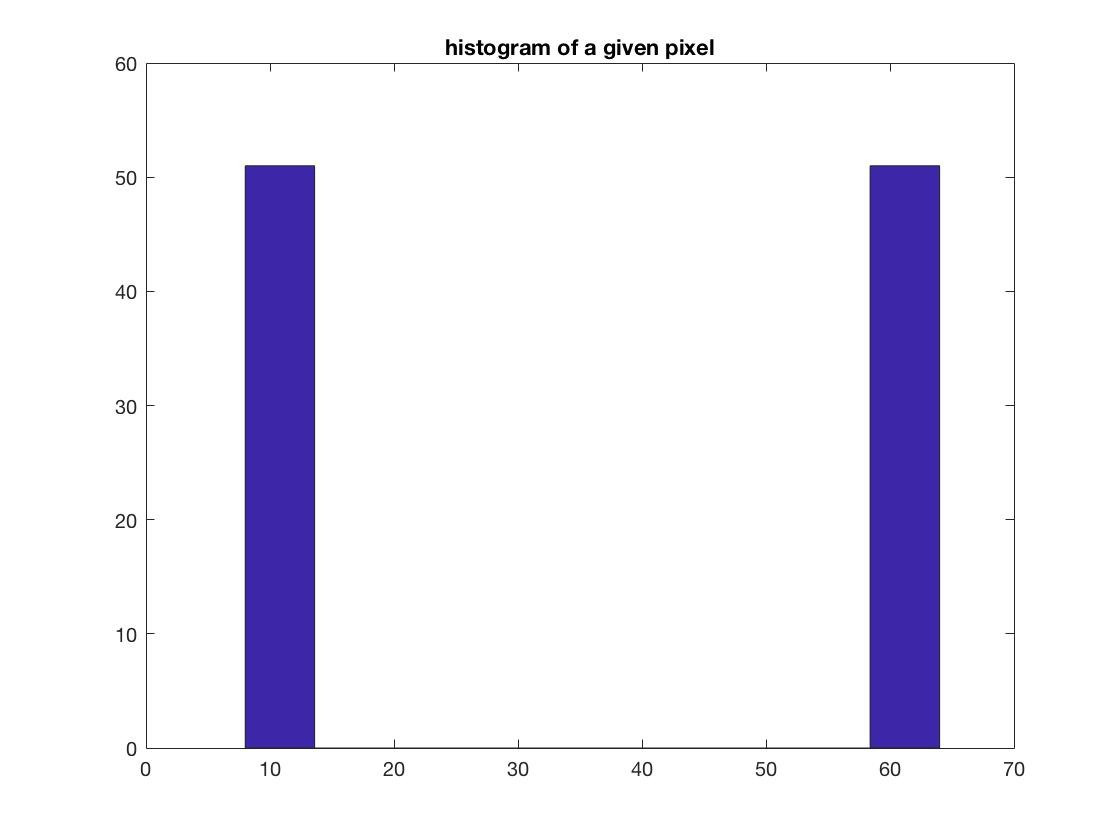


Figure 5 histogram of pixel(425,725)

This plot serves as an estimate of the Probability Distribution Function (PDF) for the noise.

What do you observe about the PDF? What is its approximate shape?

These histograms have two parts, one part is at low values and the other part is at high values, which means that minimum and maximum sensitivity settings lead to low and high values. The heights of the bar are the number of these values.

Some pixel values’ histograms are just two bars locate at low and high pixel value. But sometimes at high pixel value, these pixel values’ histograms are like normal distribution.

The PDF for the noise is normal distribution.

3. I calculate the mean and variance for each pixel in the image of minimum and maximum sensitivity settings. Figures that show both the mean and variance images are as follows.

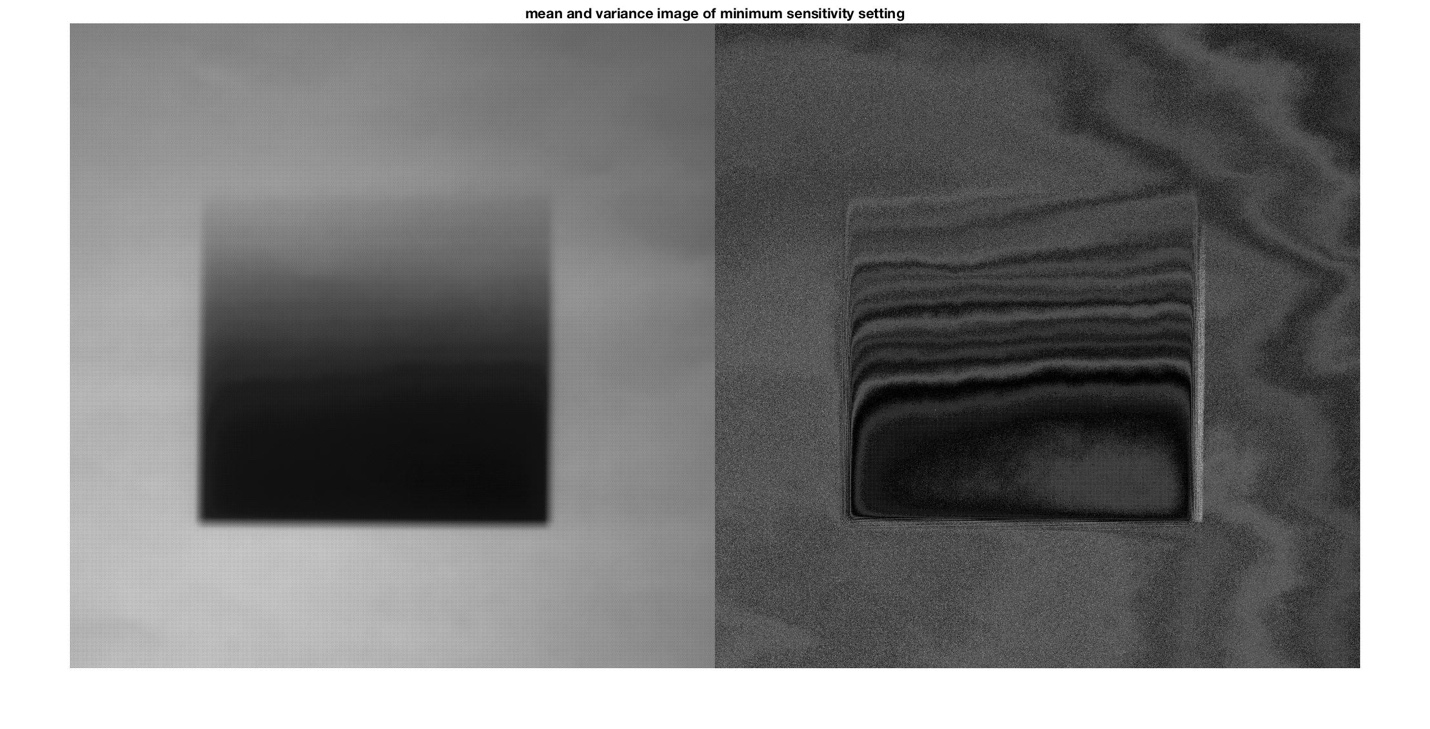


Figure 6 mean and variance image of minimum sensitivity setting

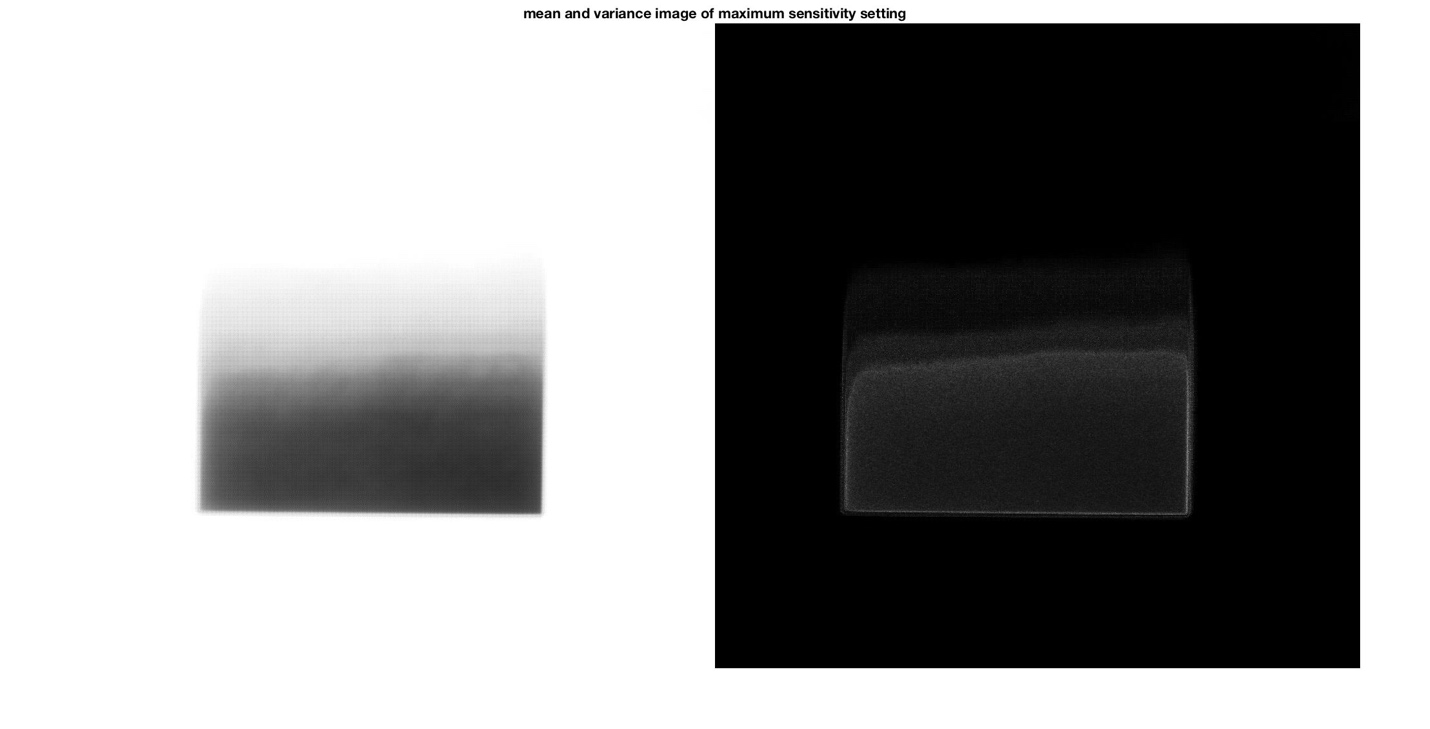


Figure 7 mean and variance image of maximum sensitivity setting

A few of the original images I captured are as follows.

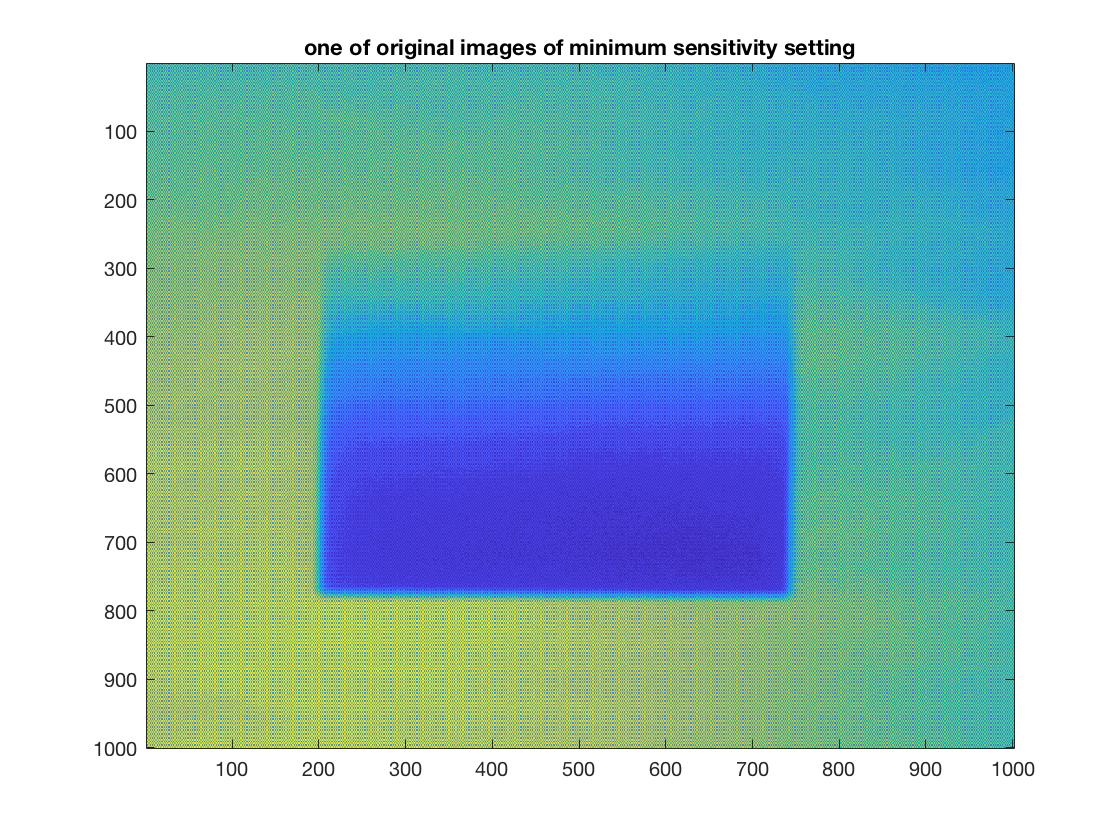


Figure 8

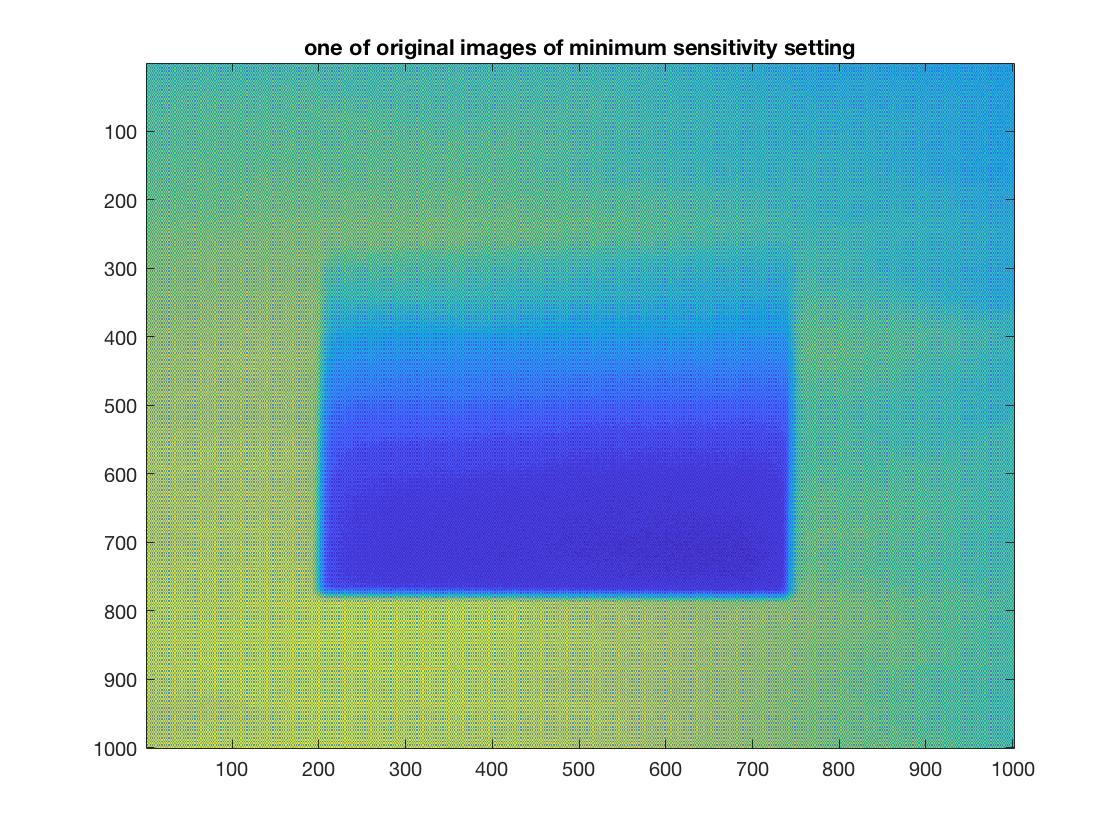


Figure 9

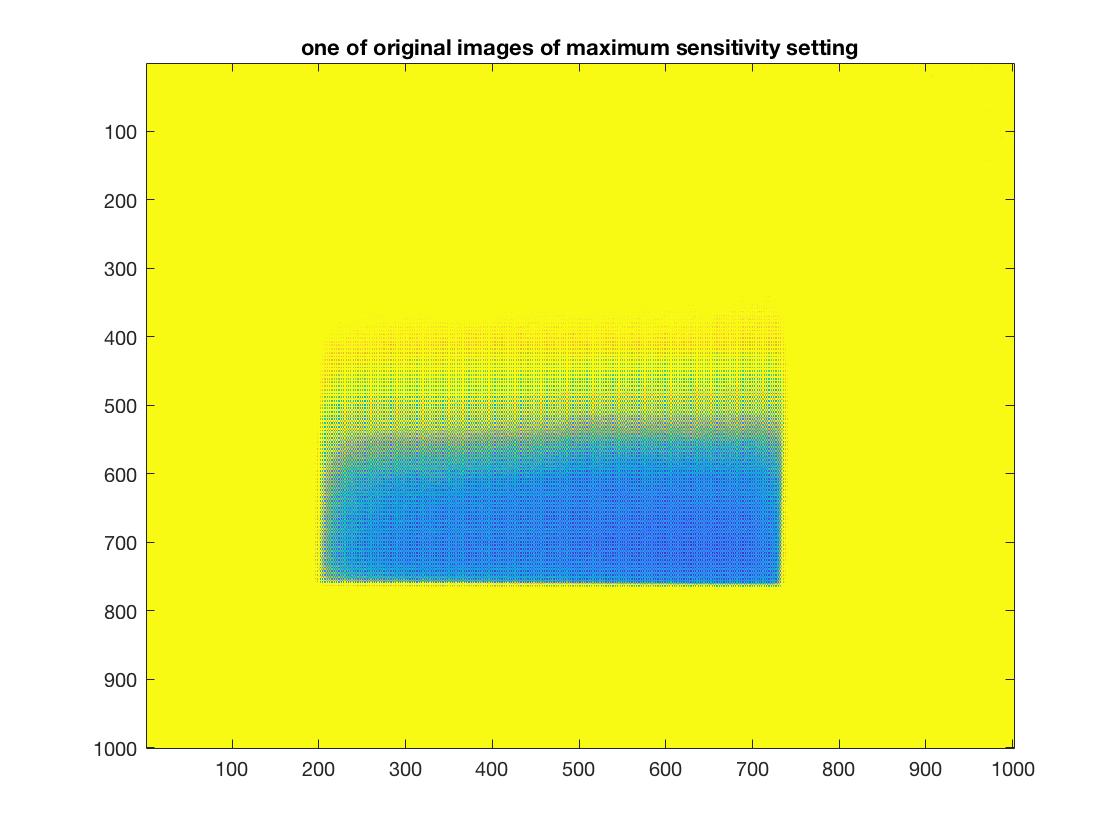


Figure 10

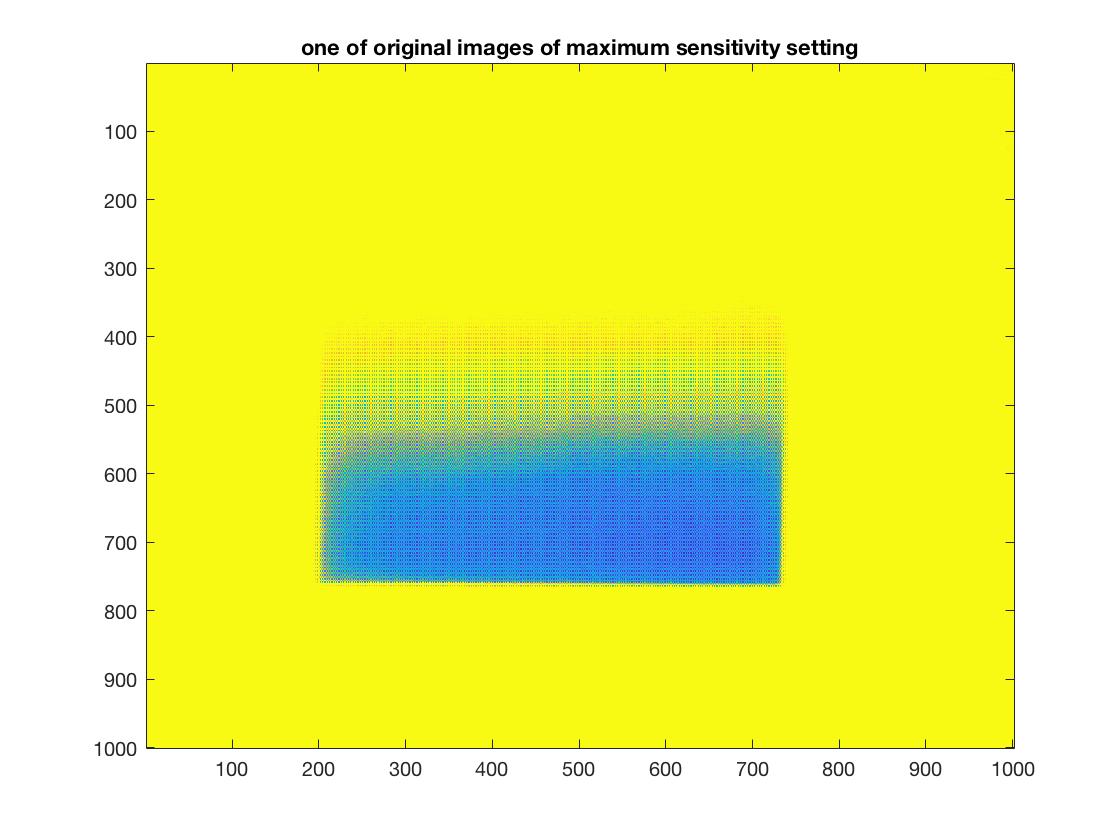


Figure 11

**Problems I encountered:**

(1) I have trouble showing figures that show both the mean and variance images. The results are just not right according to values in matrixes. I search on the Internet and find out a function ‘figure,imshowpair(mean1,var1,'montage')’, ‘montage’ is used to splice two pictures together.

(2) To show some original images by using ‘imagesc’, I do wrong to directly implement ‘imagesc’ the 3-demision matrix. Afterwards, I set a new 2-demision matrix to store each original image’s value, then I can use ‘imagesc’ to show original image.

4.&5. I plot the variance as a function of the mean for each experiment. For each experiment, I fit a line to the plotted data.

Before I plot the mean vs. the variance of maximum sensitivity, I delete the value which are larger than 60 in the mean matrix and the corresponding variance values. Because when the mean value is larger than 60, it means that the light intensity is really high and thus the variance would be very low. And I cannot fit a right line to the plotted data. If I don’t do this step, the result is as follow.

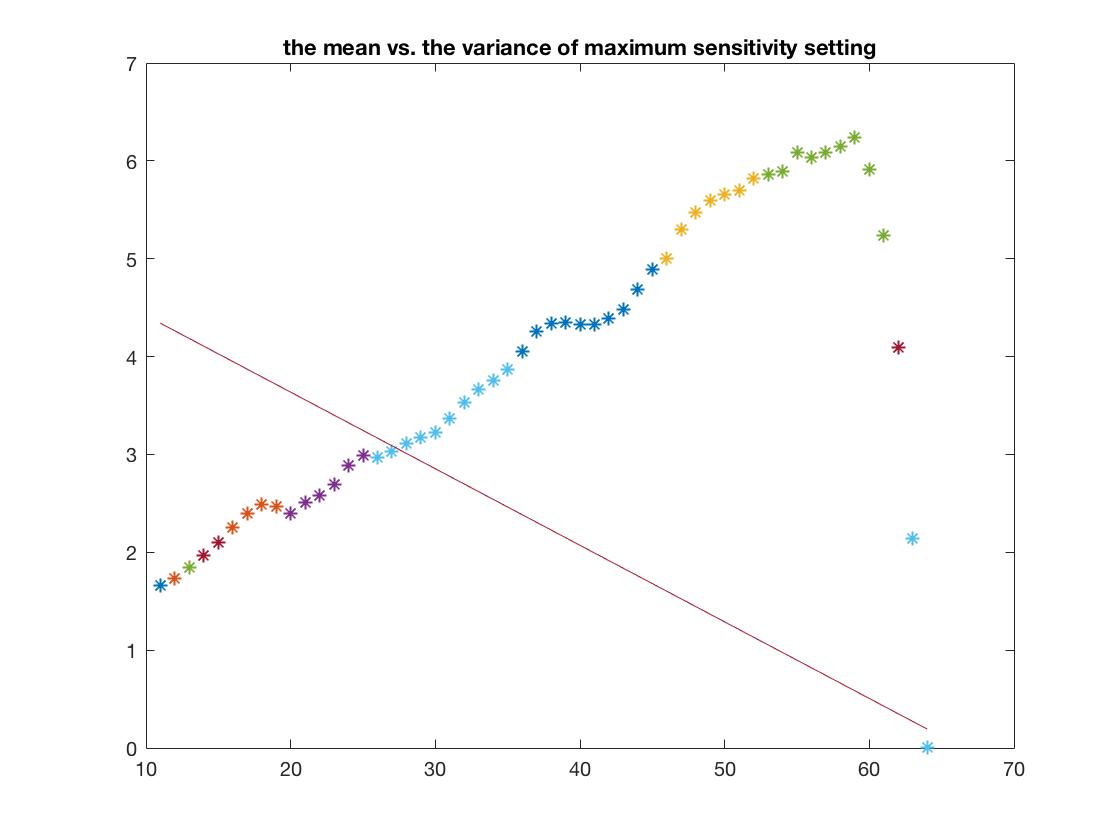


Figure 12

So, the results are as follows.

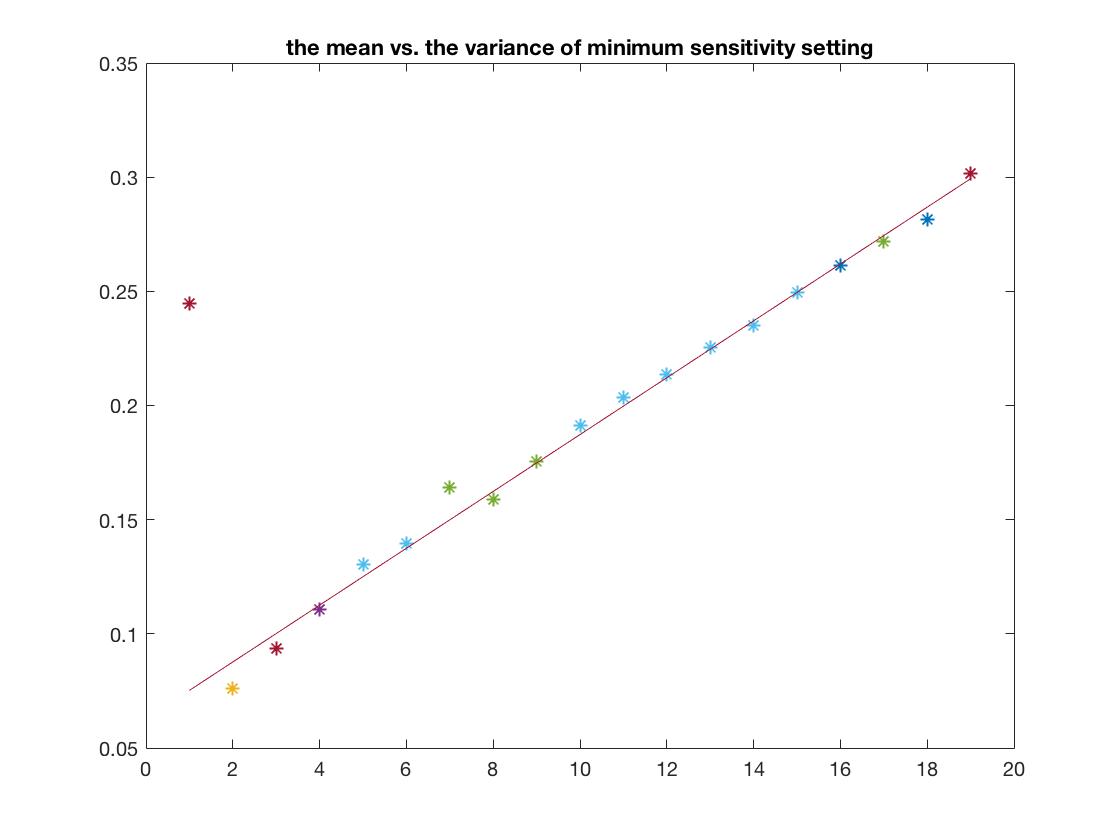


Figure 13 the mean vs. the variance of minimum sensitivity setting & fit a line to the plotted data

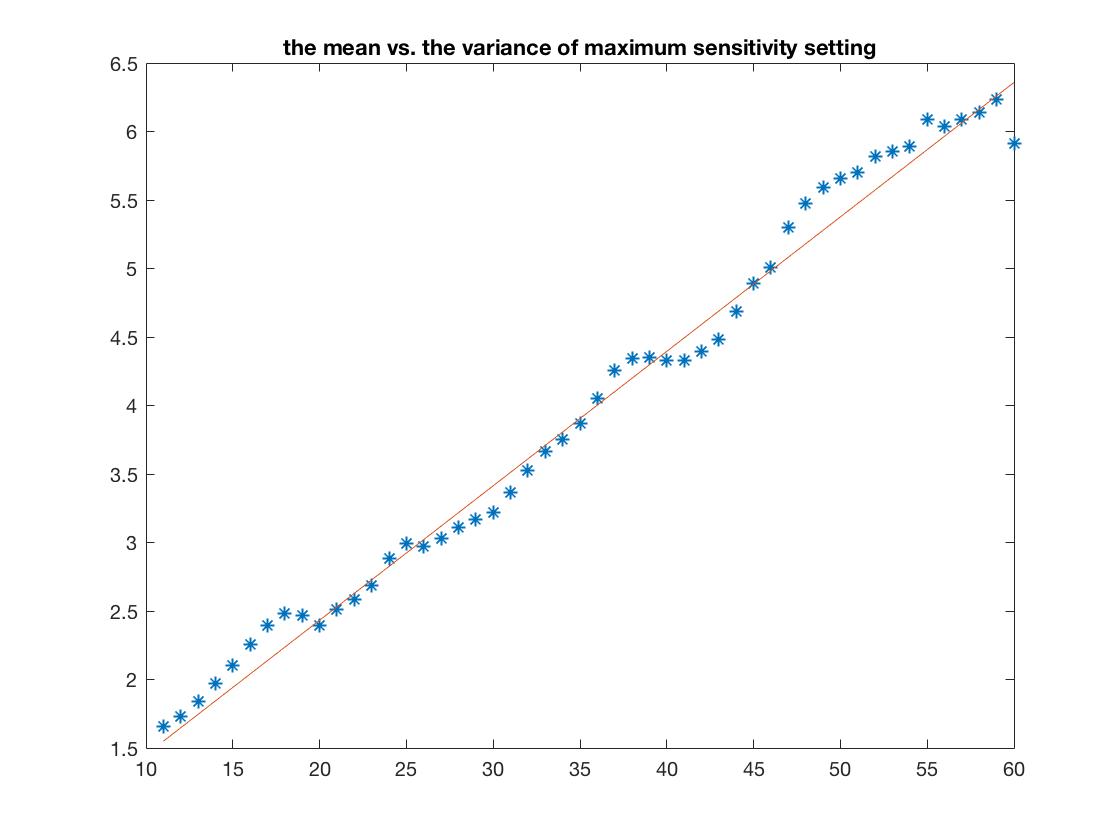


Figure 14 the mean vs. the variance of maximum sensitivity setting & fit a line to the plotted data

**Problems I encountered:**

(1) To round the mean values to the nearest integer, at first, I just round the entire matrix, obviously it is not right. Afterwards, I tried to use looping to round values one by one, and in this case, the results are perfect.

(2) To calculate the average variance for the same mean value, I need to find the same mean value in mean matrix first. I considered using looping to solve the problem, but it was complicated to be implemented. I find a function of ‘find’ to search the value that fit a given condition. So, I search locations of values from 0 to 255 in the mean matrix to find the corresponding locations in the variance matrix. Thus, I can calculate the average variance for the same mean value.

(3) To fit a line to the plotted data, in the case of maximum sensitivity, the fitted line was not quite ‘fitted’ at first. When the value is larger than 60, the variance would be very low. I cannot fit a right line to the plotted data. So I try to delete the mean values which are larger than 60 in the mean matrix and the corresponding variance values. Then I get a proper result.

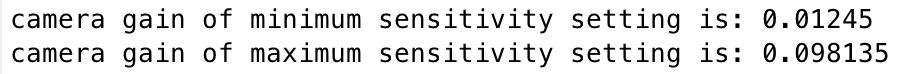
(4) I am not very familiar with the ‘polyfit’ function and I don’t know how to draw the fitted line. So I search it on the Internet how to use it, and find out that I need to use ‘polyval’ function to draw the line.

6. I use the fitted line data to calculate the camera gain g, the read noise variance (measured in photo-electrons), and the ADC noise variance (measured in gray levels).

Since the slope of the fitted line is the camera gain g, and according to definition of p = polyfit(x,y,1), I can get g=p(1). Then I find a point of each fitted line to calculate the read noise variance and the ADC noise variance.

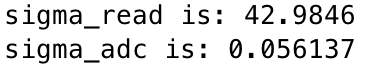
I choose (2,0.0877) on the line of minimum sensitivity setting and (20,2.4328) on the line of maximum sensitivity setting.

Results are as follows:



The g of minimum sensitivity setting is 0.01245.

The g of maximum sensitivity setting is 0.098135.



The read noise variance is about 42.9846.

The ADC noise variance is about 0.056137.

7. I calculate Signal-to-Noise Ratio (SNR) and plot the SNR as a function of mean pixel value. Results are as follows:

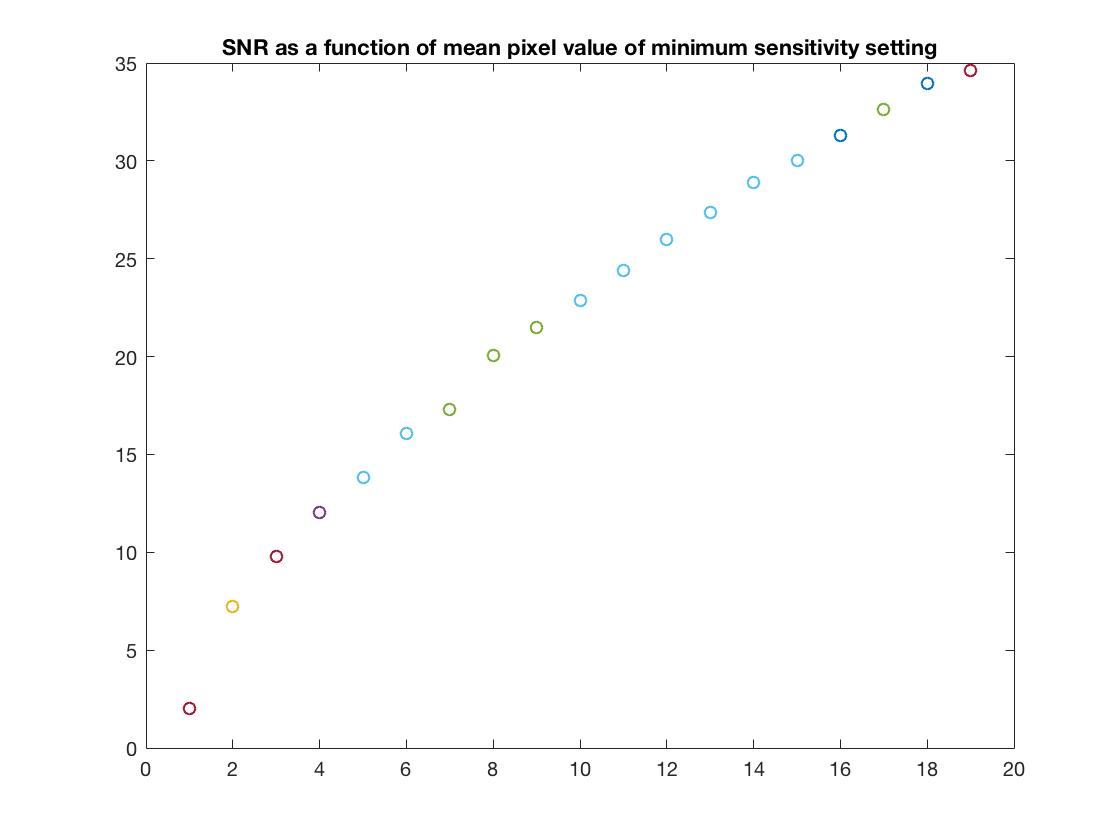


Figure 15

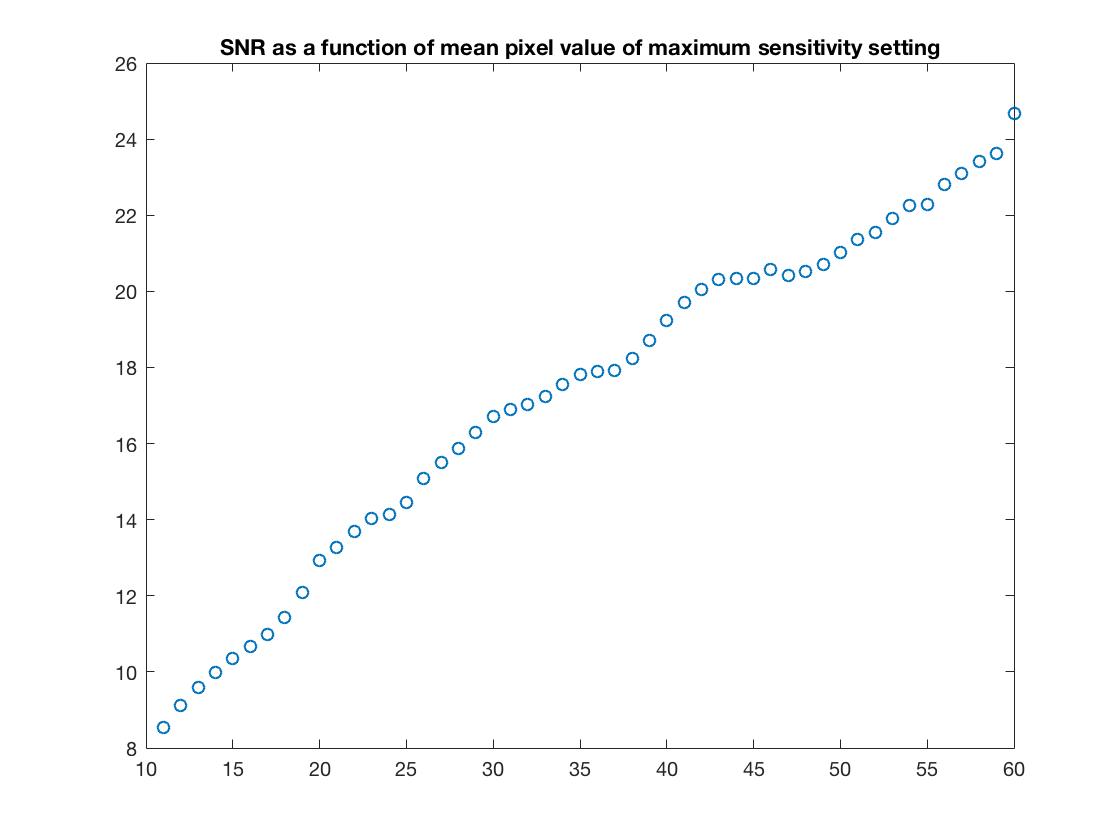


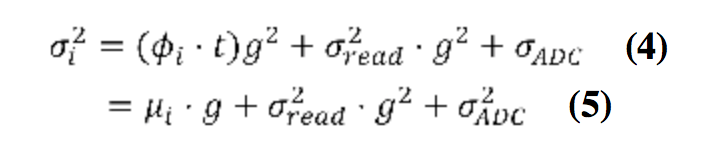
Figure 16

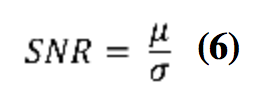
a. What is your interpretation of this plot?

The SNR is proportional to the square root of mean value μ.

If the mean value turns larger, the SNR will be larger.

b. How does it relate to the three types of noise from equations (4) and (5)?





According to equations above, the relation is as follows:

When gets larger, the SNR will be smaller. When and get larger, the SNR will be smaller.

c. What is the maximum SNR that can be achieved by the Tegra camera?

The maximum SNR can be obtained by inputting max(snr1) in command window, which is 34.6035.