Readout Noise and Dark Current

ASTR 222/522

Introduction

Charge-coupled devices (CCDs) are extremely sensitive and accurate photon detectors, but there are limitations on their performance. In this lab, you will learn about some of those limitations and measure the performance of the Santa Barbara Instrumentation Group (SBIG) CCD camera currently mounted on the 24" telescope.

Equipment

- An SBIG STL-6303E camera, mounted on the 24" telescope
- A computer running the KStars and DS9 for image analysis

Readout Noise and Dark Current

CCDs can make accurate, but not perfect, measurements of the charge accumulated in each CCD pixel. One important limitation is the noise associated with the electronics that amplifies and digitizes the charge signal in the CCD readout. Even if the exact same charge is placed in a given pixel in two different images, this noise will produce fluctuations in the number of analog to digital units (ADU) recorded. We can measure the noise by taking repeated images with the same amount of charge in each pixel. Since the amount of charge in a pixel depends on the amount of light entering that pixel, the easiest way to get the same amount of charge is to have no light enter the pixel. Thus, for these images, we block any light from entering the camera by simply not opening the shutter.

Because the analog to digital converter (ADC) in the CCD reports only positive values, while the noise fluctuations can be positive or negative, a constant offset, called a 'bias', is added to the ADC value. Thus, images obtained with no light entering the camera and with no exposure length are called 'bias' frames. The CCD noise is the fluctuation in the ADU counts around the constant bias value.

Even in the absence of light, charge will accumulate in each pixel. This is because the CCD is at a temperature above absolute zero and thermal fluctuations in the silicon can release electrons. This accumulation of charge is called 'dark current'. Dark current depends on the CCD temperature (as you will see in the analysis section).

Setting up the Camera

In this lab, we will be taking dark and bias frames – images in which no light enters the camera. In this lab, you will vary the exposure time for the images as well as the CCD temperature. The thermoelectric cooler (TEC) in the camera can only produce a temperature differential of about 20º C. Watch the status display in KStars to see the temperature decrease. The percentile near the temperature is the TEC duty cycle: 100% means that the TEC is on all the time. To get a stable temperature, the duty cycle must drop below 100%. That way the TEC can compensate for small temperature fluctuations. If the TEC duty cycle does not drop below 100% after a few minutes, increase the setpoint

temperature gradually until it does drop below 100%. Record the CCD temperature and the TEC value. Use the lowest temperature that you can achieve in place of 5º C below. Since it's a chilly day, you may be able to get to -5º or -10º C.

Bias and Dark Frames

All electronics are temperature sensitive. When using a CCD for astronomical imaging, it is important to maintain the CCD at a constant temperature and to take all calibration frames (bias, dark, and flat) at the same temperature as the astronomical images. To illustrate this point, let's look at some bias frames.

Grab a bias frame and have a look at it. The image should be mostly dark shades of gray with a few white dots. The white dots are 'hot' pixels. Notice the patterns in the image (ignoring the white dots). Does there seem to be a preferred orientation? Now set the temperature to 25 C and repeatedly grab images as the CCD heats up. Does the pattern change with temperature? What you are seeing is the effect of temperature on the electronics in the CCD and its readout. Write down your observations and explain how the patterns that you see related to the structure of the CCD. Feel free to repeat the temperature cycling.

To measure the readout noise, it is best to minimize the contribution of dark current, so, let's take some bias frames to measure the readout noise of the STL-6303E.

- Set the camera temperature to the ambient temperature and record this temperature.
- Grab a set of 5 bias frames, and be sure to include 'bias' in the filenames.

Now we will take frames to measure the dark current versus exposure time.

- Without adjusting the camera temperature, take dark frames with exposure times of 0, 1, 5, 20, 60, 180, and 600 seconds. (For the zero second exposure, take a bias frame.) Save each frame in FITS format with 'dark' and the exposure time in the filename.

Now we will take frames to measure the dark current versus temperature.

- Take a bias frame and a dark frame of exposure time 180 seconds with the CCD at temperatures in 5 degree intervals from 20º C below ambient up to the ambient temperature. Save each of these frames (two frames per temperature). Be sure to use the words 'bias' and 'dark' in the names of the appropriate files. Give the CCD a few minutes to come to the set temperature after changing the temperature and remember to record your procedure, conditions and file name for each frame, etc.

The gain for the STL-6303E is 2.3e-/ADU. Please note this number. When you're done, shut down the camera.

Measuring the Read Noise in ADU

The read noise of the CCD causes the ADU recorded for each pixel to differ from the ADU value that should be produced given the charge deposited in the pixel. Using the bias frames, we can measure the read noise by calculating the mean and standard deviation of the pixel values (after removing hot and dead pixels). The standard deviation is often described as the RMS (or rms) since it is the square root of the mean of the squared deviations.

We can use Python to calculate these statistics and plot a histogram of the pixel values in our bias frames. Bring up a copy of histimage.py code from Moodle and edit it to read in one of your bias frames. In calculating the statistics of a data set, one often prefers to discard outlier, e.g. pixels that have high values because they are saturated or 'hot' or pixels that have low values because they are damaged. Run histimage.py and examine the histogram of your pixel values. Are there outliers? In lines 41-44, we set an allowed range for 'good' pixels values (between plow and phi) and then make a new array, imghcut, keeping only the values in that range. Adjust the values for plow and phi according to your histogram and record them. Then, run histimage.py again and record the statistics of the pixel values of your image when keeping only the 'good' pixels. The mean value reported is the bias value of the CCD. Record those. Examine the same image in ds9. Is the standard deviation dominated by systematic deviations between groups of pixels (say along columns or rows of the CCD) or do the fluctuations look random?

For a better measurement of the noise of the CCD, we should remove the systematic variations between pixels and examine only the fluctuations within individual pixels. We can do this by looking at the difference between two bias frames. Open the code diffimage.py to load two of your bias frames and calculated their difference. Add in code from histimage.py to make a histogram of the difference values and calculate their statistics. Note that you will still want to remove outliers. With a bit of thought, you should be able to do this automatically without needing to plot and inspect a histogram.

Do any of your histograms contain bins with many more or many fewer counts than the adjacent bins? What is the cause of this? How can you fix it?

Edit the program to produce a histogram that covers only the difference values of interest in the bias frame. You histogram should extend out to +/- about 3 to 5 times the standard deviation. Save the histogram plot. Record the mean and standard deviation of the differences. The standard deviation is a good estimate of the read noise.

Using the value of gain above, convert your standard deviation values to electrons. Record this value. Find the read noise value for the CCD quoted in the STL-6303E manual on Moodle and compare with the value that you just measured. Are you impressed about how low the read noise is? Record your findings.

Dark Current versus Time

We wish to measure how the average accumulated dark current in the CCD varies with exposure time. Load the python program darktime.py into a text editor and have a look. Edit the file names to correspond to the names that you used in saving the bias and dark frames. The array 'darkfile' contains a list of the dark frames with exposures of 1 second or longer and the array time should contain the corresponding exposure times.
In the main loop, the stuff after '# process the files', the program reads in a dark frame, subtracts off the bias frame, and then does some manipulation to get the pixel value differences into a properly shaped array so that we can calculate the statistics. The stuff after 'choose selection region' drops the pixels with the lowest and highest readings in order to weed out bad pixels. You can adjust the value of 'f' which sets the fraction of pixels dropped on the low and high ends.

The program then plots an image of the differences, plots a histogram, and calculates the statistics. We have set the plot mode to be non-interactive, so each time you go through the loop and read a new file, you will need to close the plot window to proceed. Note that you can stretch out the plot windows to get better views of the plots (and make the labels not overlap). The program plots the mean pixel value as a vertical solid line and the median pixel value as a vertical dashed line. Are these values different? Why?

As you get to longer exposure times, you might see a second peak and/or a large tail to high values develop. We already know that some pixels look bad with very short exposures. In addition, there are pixels with unusually high leakage currents. These pixels become apparent when you use long exposure times. As you go to longer exposures, which provides a better estimate of the behavior of a typical pixel, the mean or the median? You might wish to save one or more plots, particularly one with a long exposure time.

After processing all the data files, the program then makes a plot of the statistic (mean or median) versus exposure time and does a linear fit. The program can be edited (uncomment either the line 'm = c_mean' or the line 'm = c_median' and also uncomment the appropriate plt.ylabel line) to use either the mean or the median. Which should you use? Record your choice and reasoning, and write down the fit parameters. What do you expect for the intercept? Is your value reasonably close? (What is the uncertainty in measuring the median?). A correlation coefficient of r = 1 indicates a perfect linear relation. Do your data present a good linear relation? What are the units of the slope? Convert the slope to electrons/pixel/second and record the value.

At first glance, dark current should not be a problem because we can always take a dark frame with the same exposure time as our image frames and subtract off the dark current. However, the generation of dark current is an inherently random process. Thus, if we take several dark frames with equal exposure, the accumulated charge in each pixel will

fluctuate. The standard deviation or rms of these fluctuations is equal to the square root of the number of accumulated dark current electrons. Using your value from the previous paragraph for the dark current electrons/pixel/second, calculate the rms fluctuations in the dark current versus exposure time. Then make a plot of the rms fluctations versus exposure time. Overplot a line showing the read noise (this will be a horizontal line). At what exposure times does read noise dominant? At what exposure time is the read noise equal to the fluctuations in the dark current? At what exposure times do the dark current fluctuations dominate?

Dark Current versus Temperature

Now we wish to do the same sort of analysis, but looking at the dependence of dark current on temperature rather than time. Save darktime.py as darktemp.py and edit the program to read the set of frames that you took to measure the dark current versus temperature. Again look at using the mean versus the median. Make a plot of counts (mean or median) versus temperature. Then program in the conversions to make a plot of dark current in terms of electrons/pixel/second (on the y-axis with a log scale) versus temperature (on the x-axis with a linear scale). Now we want to fit an exponential function to the data, i.e. dark current = A*exp(B*temperature). You can do this using the scipy.stats.linregress function. Plot your fit on your data, then save the plot. From your fit parameters, calculate by what factor the dark current decreases if the temperature decreases by 20° C. Measure the same ratio from your data and see how they compare.