SLAC NATIONAL ACCELERATOR LABORATORY

Optimizing Pulnix TM4200CL camera timing for 120Hz operation, and Developing a PyDM camera viewer application

Ram Hari Dahal[1], Bruce Hill[2]

[1]Electrical Engineering and Computer Science, Howard University, 2300 Sixth St NW, Washington, DC, 20059, USA

[2]Linac Coherent Light Source, SLAC National Accelerator Laboratory, 2575 Sand Hill Road, Menlo Park, CA 94025, USA.

+Contact: dahal01@slac.stanford.edu, bhill@slac.stanford.edu

Introduction

The Pulnix TM-4200CL camera (ptm4200) is used in many places from the accelerator to the photon experimental hutches. Typical use is for alignment and analysis of beam position and footprint. The resolution is 2048x2048 with 12 bit pixels resulting in an 8.4MB image size. The readout is over a parallel interface called Camera Link, which provides a predictable transmission time per image based on the number of pixels in the image and the Camera Link clock speed.

Max frame rate for the ptm4200 in full resolution is 15 Hz, and since LCLS rates jump from 10 Hz to 30 Hz, it is typically operated in full resolution at 10 Hz, and the synchronization of image to beam pulseID has been easy. Operation of the ptm4200 at 120Hz with a reduced region of interest (ROI) of 256x2048 should be possible in order to capture pulse by pulse beam footprints, but has proven unreliable due to inaccurate estimates of CamLink transmission time for smaller ROI.

The goal for this part of the project was to conduct timing tests and modify the original formula for calculating CamLink transmission time derived from the vendor documentation in order to achieve +- 2ms accuracy from 8 lines to the full 2048 lines. Only the vertical ROI was varied as the ptm4200 does not support horizontal ROI. The conclusion of this part of the project includes a presentation of the results, committing the revised formula to version control and building a new release of the ptm4200 device support module.

PyDM (Python Display Manager) is a PyQt-based framework for building user interfaces for control systems. The goal is to provide a no-code, drag-and-drop system to make simple screens, as well as a straightforward python framework to build complex applications.

PyDM Camviewer is an application/tool built using pydm which allows the user to pass a configuration file (with a list of Camera names, Image address, and Camera Description) to view the camera image in real time and see the real time stats.

Keywords: pulnix, timing, camlink transmission, pydm, camviewer

Research

For the Pulnix TM-4200CL camera, the CamLink XmitTime was previously calculated using the formula: XmitTime =

$$(\lceil\frac{MinY_{RBV}+16}{8}\rceil+\lceil\frac{2056-MinY_{RBV}-SizeY_{RBV}}{8}+SizeY_{RBV}\rceil)*32.5us$$

Here, MinY_RBV is the minimum pixel line value where the image starts from and SizeY_RBV is the size of the image starting from MinY_RBV.

So, for example, if MinY_RBV=0, and SizeY_RBV=2048, the expected delay could be calculated using the following formula:

XmitTime = (CEIL((0 + 16)/8) + CEIL((2056-0-2048)/8) + 2048)) * 32.5us

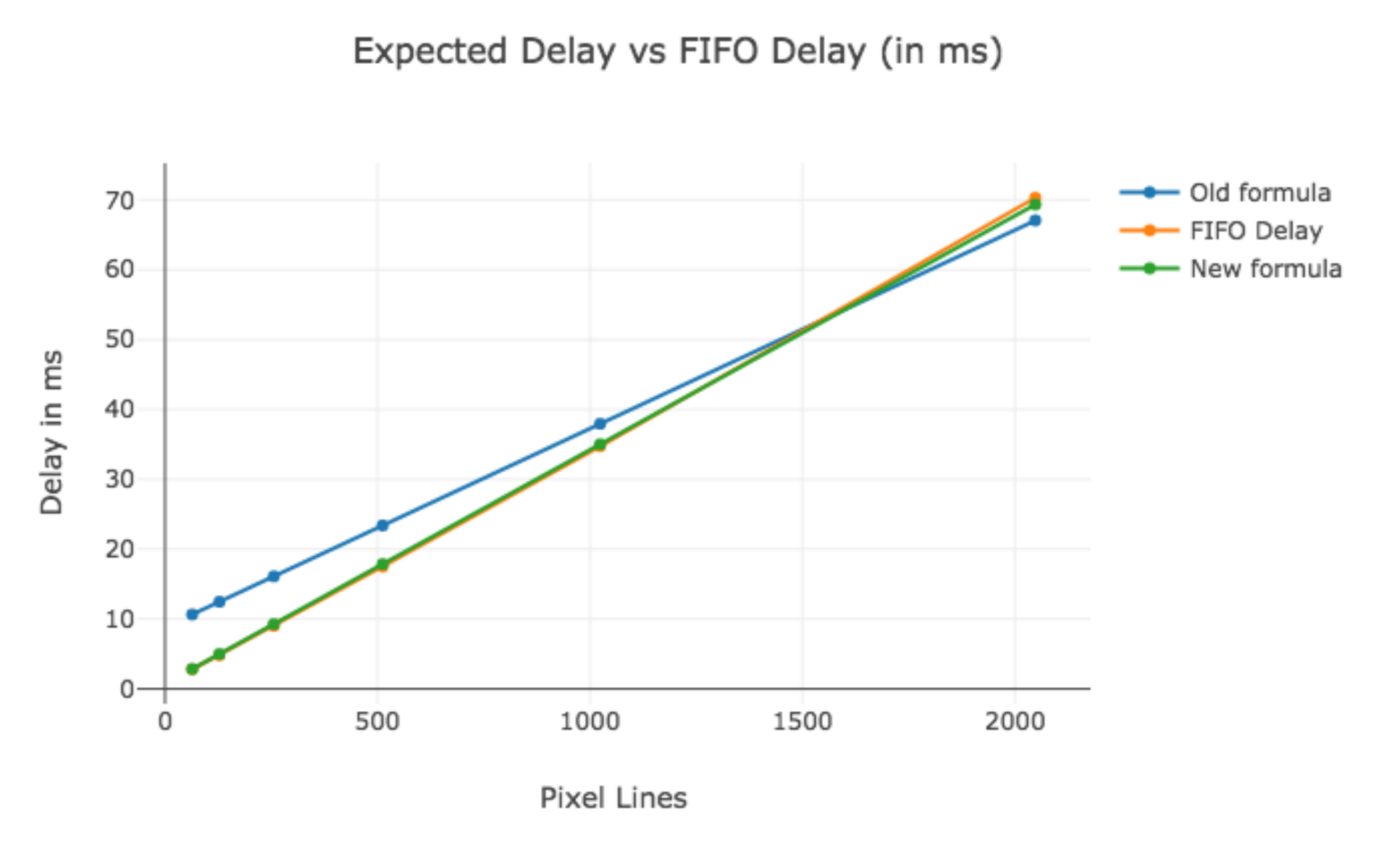
From this, we get the expected delay to be = 67.07ms. However, from the tests we could see that the average delay (of sample size 15) was around 70.31ms.

Using the dataset for different data sizes, ranging from 8 pixel lines to full 2048, we adjusted the formula to calculate XmitTime as below: XmitTime =

$$(\lceil\frac{MinY_{RBV}+16}{8}\rceil-250+10*2^{\lfloor\frac{\log(SizeY_{RBV}+1)}{\log(2)}-6\rfloor}+\lceil\frac{2056-MinY_{RBV}-SizeY_{RBV}}{8}+SizeY_{RBV}\rceil)*32.5us$$

From this, for MinY_RBV=0 and SizeY_RBV=2048, we could get the expected delay to be = 69.34. This, as we can see is closer to what the real delay is.

The following graph shows the expected delay of old formula vs expected delay of new formula compared with the real expected delay that we got from experiments.



We went from having error rates of +-10ms for sizes greater than 512 pixels to +-0.7-0.9ms (average) and error rates of +- 12ms for sizes less than 512 pixels to +- 0.5-0.7ms (average).

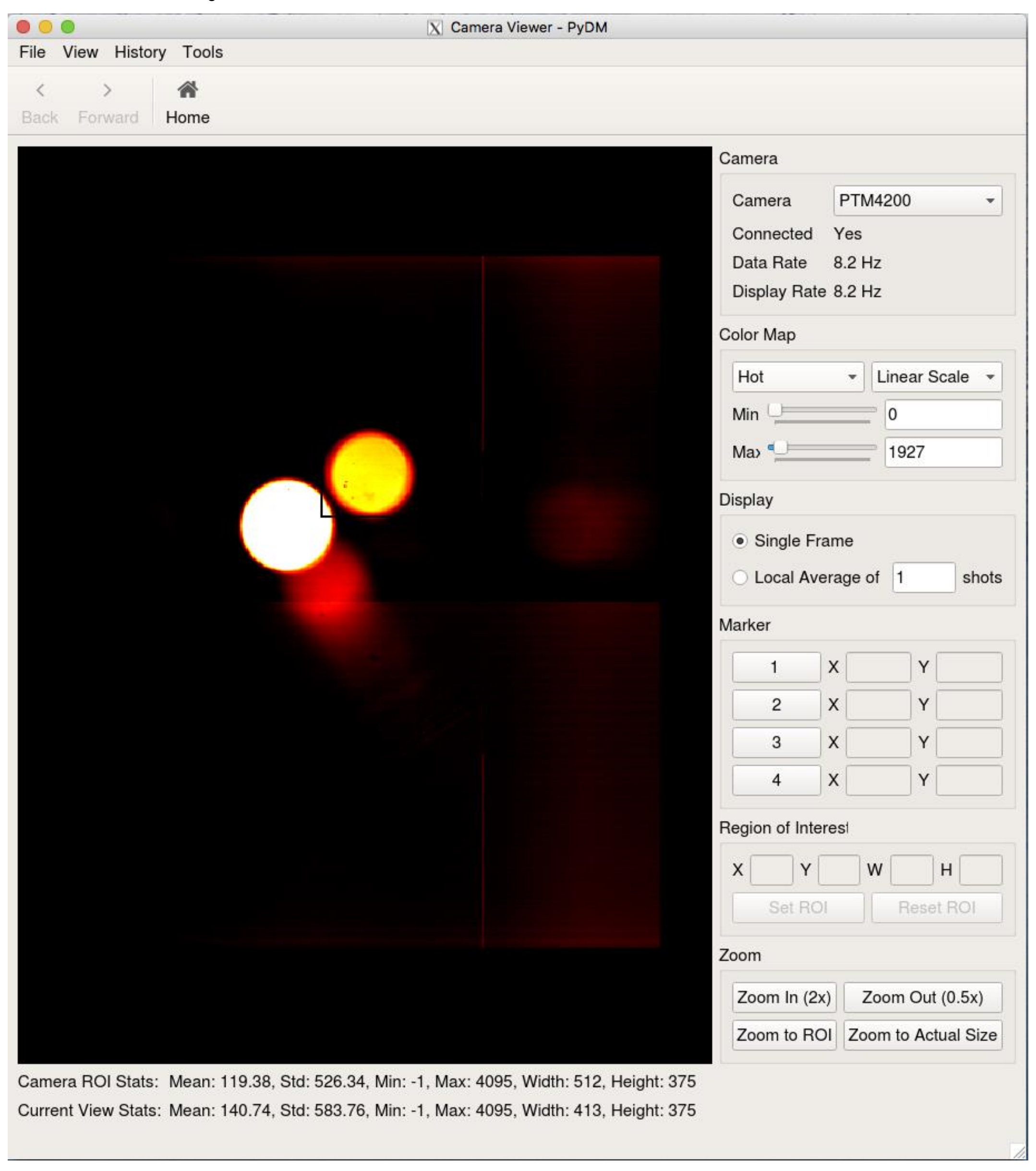
For the second project we created a PyDM camera viewer application. You can feed a cfg file and open a pydm camviewer application. The command to run the pydm camviewer application is as follows:

pydm camviewer.py <your .cfg file>

where, the .cfg file should have three properties (Camera Name, Image Address, and Camera Description) separated by a space.

Now, when the application is running, from within the application you can control and see the camera details (ROI stats, view stats, etc), check the data and display rate, view the camera images in different color map (monochrome, magma, etc). You can also see if the camera you are using is connected or not, and also zoom into the camera image.

There is also a dropdown menu to choose from the list of cameras provided from the .cfg file. You can just click on the dropdown menu and choose the camera that you want to choose and view the real time stats of that camera. You can also choose between viewing the single frame of the camera image or the average of as many shots as you like.



Camera
Camera PTM4200
Connected Yes
Data Rate 8.2 Hz
Display Rate 8.2 Hz

Color Map
Hot | Linear Scale
Min 0
Max 1927

Display
● Single Frame
○ Local Average of 1 shots

Marker
1 X Y
2 X Y
3 X Y
4 X Y

Region of Interest
X Y W H

Zoom

Camera ROI Stats: Mean: 119.38, Std: 526.34, Min: -1, Max: 4095, Width: 512, Height: 375

Current View Stats: Mean: 140.74, Std: 583.76, Min: -1, Max: 4095, Width: 413, Height: 375

Fig. The User Interface of the PyDM camera viewer application that shows the image viewer, camera chooser, camera ROI and camera view stats, display options, zoom options, etc.

Conclusions

While there were several roadblocks while doing the projects including the steep learning curve to understand the technology used and the codebase, and several network issues that made it harder to perform the timing tests, I was able to come up with a better algorithm to predict the expected camlink transmit time for Pulnix TM4200CL camera, and also work on camviewer application that can do amazing stuff.

For the future work, I would want to make the PyDM camviewer application have more features and make it bug-free. Currently, the application has some issues with automatically resizing into any display size. There is also a problem that causes the application to crash when the .cfg file has an invalid image address.

Acknowledgments

I want to thank my mentor Bruce Hill, manager Ernest Williams, and co-mentor Hugo Slepicka for guiding me throughout the project.

Use of the Linac Coherent Light Source (LCLS), SLAC National Accelerator Laboratory, is supported by the U.S. Department of Energy, Office of Science, Office of Basic Energy Sciences under Contract No. DE-AC02-76SF00515.

Date: 08/03/2018