EPICS, MATLAB, GigE CCD CAMERA BASED BEAM IMAGING SYSTEM FOR THE IAC-RADIABEAM THz PROJECT

Chris Eckman*, Y. Kim, A. Andrews, P. Buaphad, and T. Downer
Idaho State University, Pocatello, ID 83201, USA

Abstract

At the Idaho Accelerator Center (IAC) of Idaho State University, we have been operating an L-band RF linear accelerator running at low energies (5 - 44 MeV) for the IAC-RadiaBeam THz project [1]. We have designed and implemented an image acquisition and analysis system that can be used for real time observation of the electron beam, tuning of THz radiation production, and measurement of the transverse beam emittance. The imaging system contains an Yttrium Aluminium Garnet (YAG) screen on an actuator, a Prosilica GC1290 GigE CCD camera with an adjustable lens, a screen illuminator, an optical alignment structure, and a lead tube for the camera shielding. The real time continuous beam images can be acquired by SampleViewer, while the single shot beam image can be acquired by the Experiential Physics and Industrial Control System (EPICS) and areaDetector. In this paper, we describe components of the imaging system, the real time beam image acquisition with SampleViewer, the single shot beam image acquisition with areaDetector, and a remote controllable beam image acquisition via MATLAB Channel Access (MCA), MATLAB, and EPICS.

INTRODUCTION

For the IAC-RadiaBeam THz experiments, due to the 1.2 mm wide opening gap of the THz radiator, good beam shape control is required at the THz radiator to get good beam transmission. To optimize the beam shape freely, to measure beam size, and transverse beam emittance, ISU Advanced Accelerator and Ultrafast beam Lab (AAUL) group has developed an advanced beam imaging system, which can supply the continuous beam image acquisition in real time by using SampleViewer program as well as the single shot beam imaging acquisition with remote controlling function by using areaDetector, EPICS, MCA, and MATLAB. The imaging system consists of a Prosilica GC1290 GigE CCD camera, a lens to focus the camera, Thorlabs cage that holds the camera in place, and two Thorlabs supporters to mount the cage on an optical table [2].

THE IMAGING SYSTEM

The full imaging system can be seen in Fig. 1. All of the components of the optics cage were purchased through Thorlabs and are compatible with each other, thus the optics cage was easily assembled. The main Thorlab components are listed in Table 1. The camera is situated to look into the crystal window to the YAG screen. The camera uses Gigabit Ethernet to output data onto a local area network where it is used by a Linux operating system running EPICS. This is where the data can be captured, processed, and controlled.

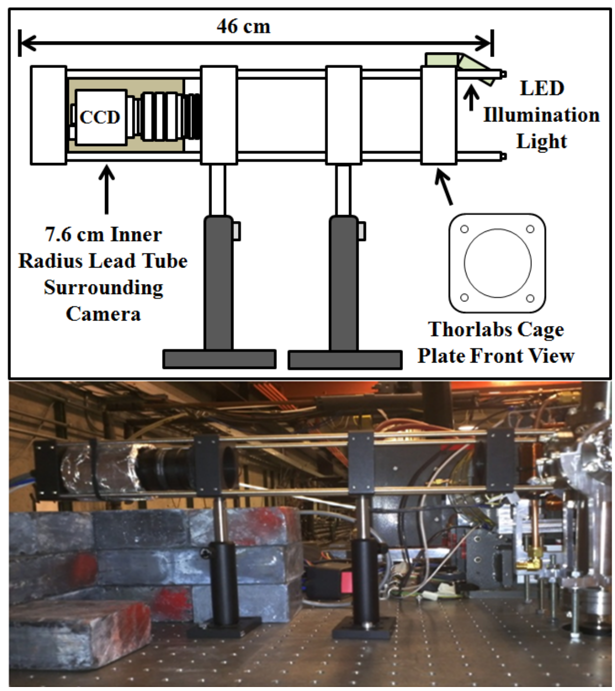

Figure 1: This is the imaging system set-up at the IAC.

Stray light adds to background noise of the image, so a black covering is placed around the optics cage to prevent any unwanted light from entering. In addition, to protect the camera sensor from damaging radiation, a lead tube is placed inside the cage to surround the camera. For extra protection, the optics cage is almost completely surrounded by lead blocks when ready to use. There is an adjustable LED light on the end of the optics cage that is used to illuminate the YAG screen for alignment and calibration. The overall construction, testing and cost makes this set-up attractive for an effective imaging system.

PROSILICA CAMERA

The GigE CCD cameras are essential in beam diagnostics where high quality beam images are used for measuring various parameters and diagnosing beam quality. This particular camera was chosen because it has a good resolution and pixel size (see Table 2). The parameters of the

* cryptoscientia@gmail.com

Table 1: Main Components Purchased Through Thorlabs for the Imaging Cage Set-up

Image	Description	Part Number	QTY
	Threaded Cage Plate	LCP01T	4
	Cage Assembly Rod	ER18	4
	Steel Optical Post	TR4	2
	Post Holder	PH4	2
	Mounting Base	BA2T2	2

beam calculation rely heavily on the image quality, resolution, distance from object being viewed, electronic noise or broken pixels, and image exposer.

Table 2: Main Specifications on the Prosilica GigE CCD Camera and the Fujinon HF50SA-1 Lens

Parameter	Value or Type
Interface	GigE
Resolution	1280 x 960
Sensor size	Type 1/3
Cell Size	3.75 μm
Max frame rate at full resolution	32 fps
Lens Focal Length	50 mm
Lens Minimum Field of View	38 mm x 28 mm

For optimum viewing and imaging, a Fujinon HF50SA-1 lens is directly attached to the CCD camera on a C mount and focused onto the YAG screen (see camera and lens in bottom right of Fig. 2). Using the Fujinon lens makes focusing, cleaning and adjustment of the equipment quick and easy. The Prosilica camera has an external trigger input port that can be used to receive a trigger signal from the timing system. The power cable for the camera had to be physically modified to add this triggering port. Triggering the camera allows it to be synchronized to the beam so proper images can be taken.

SAMPLEVIEWER

Sampleviewer is the GigE CCD camera software from the Prosilica company and is provided for free. This program provides a user friendly tool that works over an internet connection to test a CCD camera quickly, check image quality, camera alignment, camera triggering, image gain, image exposure time, and for real time imaging (see Fig. 2). In order to put the CCD camera on the network, the computer's Ethernet adapter needs to be configured to improve the system performance when using the GigE CCD camera [3]. These changes include the packet size (MTU of 8228), interrupt moderate rate, transmit buffers, and receive buffers. The camera also uses a companion program to SampleViewer, named IPConfig, to change the IP address of the camera to a static address.

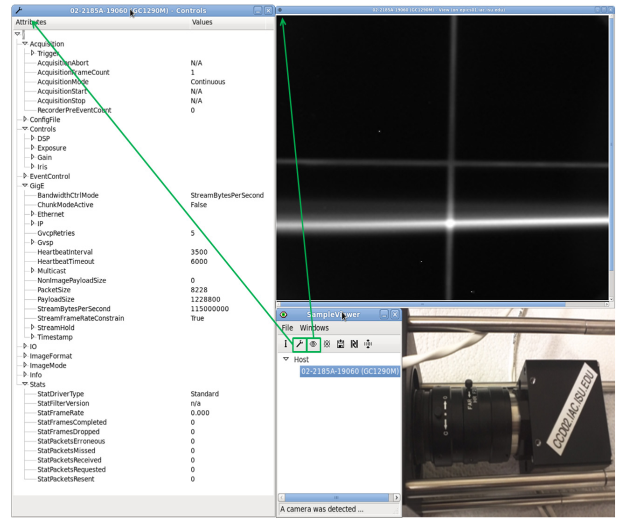

Figure 2: This is SampleViewer and the Prosilica GC1290 GigE CCD camera with the attached lens.

EPICS AND AREADETECTOR

To more precisely control these CCD cameras and use images for data extraction, we need a precise and programmable method that can be accessed by the accelerator control Graphic User Interface (GUI). EPICS provides a platform that will allow camera control in an accelerator system [4]. EPICS provides GUI for user interaction when needed, such as the main areaDetector GUI that is shown in Fig. 3. Each button and input box on this GUI screen has an associated Process Variable (PV). Images are taken by the CCD camera using areaDetector's "Start" button, where the PV associated with that button is named "13PS1:cam1:Acquire". All PV's in areaDetector can be controlled in MATLAB through MCA.

MATLAB CHANNEL ACCESS

MCA will allow a pathway from MATLAB to the EPICS PVs, this link allows talk between the two different programs [5]. PVs are a named piece of data associated with the machine, things like status, readback and physical parameters. With MCA, we are able to send any information (in the form of PVs) between EPICS and MATLAB by means of the channel access, and the accelerator can be controlled with MATLAB codes directly. MCA is one of the extensions of EPICS that can be downloaded and installed in both EPICS and MATLAB. When it is installed properly, commands can be entered into MATLAB and executed to control equipment in EPICS. The main commands of MCA to open the desired channels between

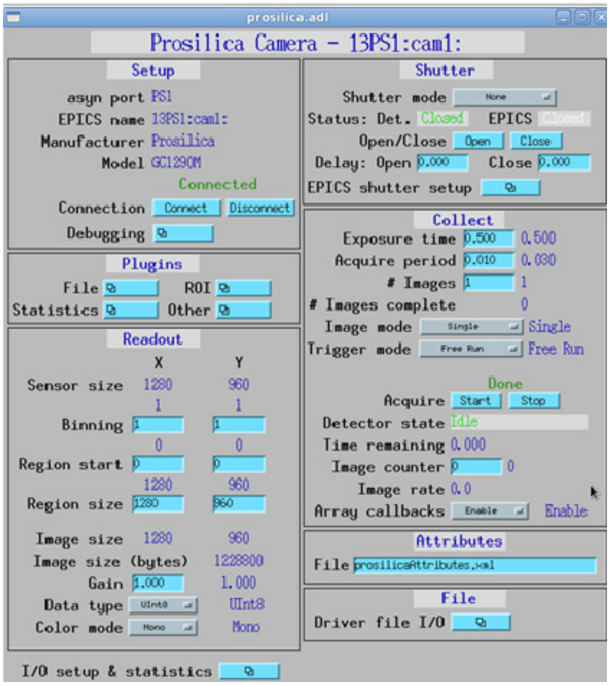

Figure 3: This is the main GUI screen of AreaDetector in EPICS.

the equipment and MATLAB, to read information, and to write to PVs are MCAOPEN, MCAGET, and MCAPUT. An example of the MCA and MATLAB code is following:

```
CameraAcquire=MCAOPEN('13PS1:cam1:Acquire');
AcquireData=MCAGET(CameraAcquire);
MCAPUT(CameraAcquire,1);
MCACLOSE(CameraAcquire);
```

The MCAOPEN will open the channel to the PVs. The MCAGET will retrieve the status on the PV associated with the camera shutter. If the CameraAcquire is 0, then the camera shutter is closed. However if the CameraAcquire is 1, then the shutter is open, and the camera starts taking images. The MCAPUT will write values into the PV, such as 1 or 0, and the MCACLOSE will close the channel of the PVs. These images are then transferred to the hard drive using MCA to be saved in the designated file path (this path is also a PV and can be changed). These MCA commands are used in MATLAB to control all aspects of the camera as well as other equipment such as power supplies [4]. Once the images are in MATLAB they can be processed to remove background, plot projections, filter the image, and extract data for parameter measurement. In Fig. 4, the left image is the same one as in Fig. 2, however the background is subtracted and is color enhanced. The image to the right is used by an automatic emittance measurement system running in MATLAB and shows the x and y projections on the image directly [6].

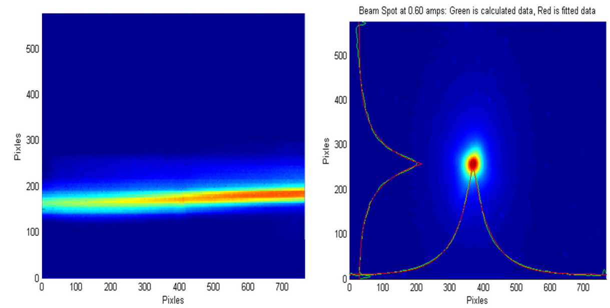

Figure 4: This is MATLAB processed beam images.

SUMMARY

EPICS, areaDetector and MCA are successfully installed and working. The single shot beam images were taken by the Prosilica GigE CCD camera using MATLAB code in conjunction with areaDetector. By using MATLAB and MCA, the user has access to higher level programming, which can be used to control any accelerator parameter with PV. For example, the beam images can be transferred from the camera to MATLAB code directly, and can be used for computation and plotting with MATLAB. This is an extremely useful tool that can be used in many accelerator applications and measurements. By using the real time video capabilities of SampleViewer, the beam imaging system was used to optimize the THz radiation at the IAC [7, 8].

ACKNOWLEDGEMENTS

We would also like to give our sincere thanks to Dr. Mark Rivers of APS for his advice and guidance on AreaDetector and MCA. In addition, we would like to thank Dr. Henrik Loos of SLAC for his strong interest, advice and encouragement in this project.

REFERENCES

[1] http://www.iac.isu.edu

[2] http://www.thorlabs.us/index.cfm?

[3] http://www.alliedvisiontec.com/us/products/cameras/gigabit-ethernet/prosilica-gc/gc1290.html

[4] A. Andrews *et al.*., in these proceedings.

[5] A. Terebilo *et al.*., in Proc. ICALEPCS 2001, THAP030, San Jose, CA, USA.

[6] C. Eckman *et al.*., in Proc. IPAC2012, TUPPC055,New Orleans,LA

[7] Y. Kim *et al.*., in these proceedings.

[8] A. V. Smirnov *et al.*., in these proceedings.