BEAMVIEW - A DATA ACQUISITION SYSTEM FOR OPTICAL BEAM INSTRUMENTATION*

R. Haseitl, C. Andre, F. Becker, P. Forck, GSI, Darmstadt, Germany.

Abstract

At the GSI accelerator facility, several optical beam instrumentation devices for transversal profile measurement are installed. Their readout is done using FireWire CCD cameras attached to a compact embedded device, specialized for image processing tasks. The resulting data like projections, histograms and images is sent via Ethernet to a Windows or Linux PC, reaching frame rates above 30 frames per second at VGA resolution. In this paper we present the system components and software design to control CCD cameras and various other devices. First experiences of operation with beam are presented as well.

MOTIVATION

Existing System

In aspects of optical beam instrumentation, GSI mainly uses scintillator screens, being read out with analogue cameras. The video signals are transmitted into the main control room via long cables and multiplexers, bringing the pictures on several monitors. This system is simple and cheap, but suffers from some disadvantages:

The pictures are sent from the camera onto the screen and are then 'lost' lacking the possibility of any offline analysis, since no frame grabbers are installed. Determining the properties of short beam pulses on a scintillator is not possible, because just a bright flash over the whole image will be observed. Since the images are not stored, a comparison to earlier settings is also impossible. Therefore a new, digital system was designed, to meet the requirements for machine operation.

Requirements for the new System

Not only scintillator screens (SCR) are observed by CCD cameras, also Beam Induced Fluorescence (BIF) monitors are read out digitally. BIF is a non-destructive method to determine the beam profile by ionizing gas molecules in the beam pipe [1]. The ionized molecules emit a few photons which are detected by a CCD camera, equipped with an image intensifier. Presently, FireWire (IEEE1394a) CCD cameras are used for these purposes, running at VGA resolution (640x480 pixel).

To meet the demands of BIF and SCR, the following features should be offered by the new system called ‘BeamView’:

- view and save images in triggered mode or with predefined frame rates
- precise triggering at machine events and control of exposition
- low-level switch to suppress triggers, if camera is still busy
- capability of buffering images to overcome temporary bottlenecks
- provide online analysis (e.g. projection and histogram calculation)
- control camera parameters (gain, integration time, resolution), camera trigger, remote-controllable iris, image intensifier amplification and gas pressure out of one single software
- fit into the existing infrastructure: Linux and X-Terminals in the main control room, Windows at most experiments
- easy to use standard interface with 'Expert Mode' for advanced users
- standard hard- and software interfaces to provide interoperability for later system upgrades

SYSTEM DESIGN

Hardware

To meet the requirements mentioned above, a two-stage hardware design was anticipated. The existing FireWire cameras are connected to dedicated image processing hardware, acting as middle layer between the camera and the computer running the Graphical User Interface (GUI) as a client. For this purpose, the embedded system from National Instruments Compact Vision System 1456 (CVS) is used (see Fig. 1) [2]. This embedded computer is equipped with FireWire interfaces, an FPGA with digital I/Os, network connectivity, a 733 MHz Intel Celeron CPU, 128MB RAM and 256MB non-volatile storage. The CVS is capable to control up to 16 cameras connected via FireWire hubs, with four of them running simultaneously. It is programmed via LabView Realtime, FPGA and Vision packages [2].

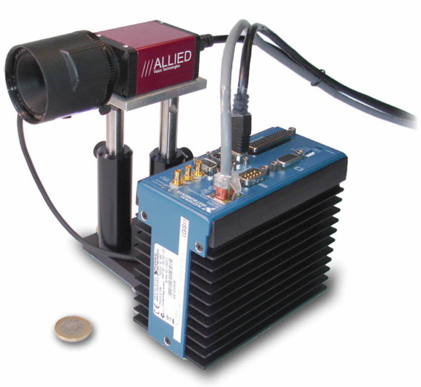


Figure 1: National Instruments Compact Vision System 1456 [2] with attached FireWire camera and remote controlled iris lens system.

*Work supported by EU, project FP6-CARE-HIPPI

To trigger the cameras to the beam pulse, they are attached to a programmable timing interface. This device has a network connection to choose the particular machine event. Furthermore every camera system is equipped with a variable iris, which is remote controlled via an Ethernet connected DAC electronics. The same DAC system controls the amplification stage in the image intensified cameras. The communication to a commercial pressure control unit (Pfeiffer RVC300) is performed via an RS232 to Ethernet converter (Lantronix UDS2100). To minimize radiation damage, most of the active electronics is placed outside the accelerator tunnel. FireWire extenders using optical fibers are used for this purpose. The schematic of the hardware is depicted in Fig. 2.

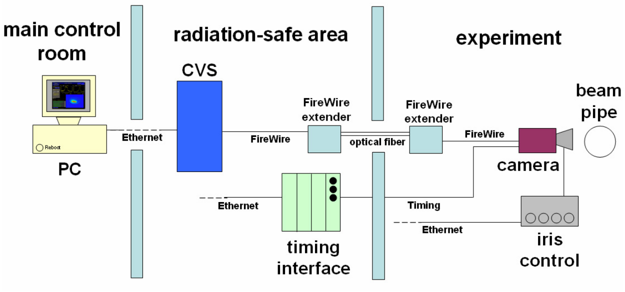


Figure 2: Arrangement of the SCR hardware.

Software on the CVS

Based on the distributed hardware, a distributed software design was developed. The part of the software running on the CVS is responsible for acquiring images and forwarding them over the network. The PC based software receives the images, displays them and gives the user full control over CVS and camera parameters.

A specification of the requirements for the CVS software has been made and given to an external company which developed the LabView application for the CVS and a C++ API for the client [3]. The CVS software offers the following features:

- control up to four cameras simultaneously
- preprocess incoming images and - if demanded - execute the following operations: histogram and projection calculation, image compression, dead-pixel correction, region of interest, perspective correction, mirroring, rotation
- send out the results via Ethernet
- buffer data in case of heavy network traffic or unavailability of the client
- disable the trigger in the FPGA in case the camera is busy processing the last frame
- XML interface for CVS and camera control
- standard TCP/IP socket communication to the outside world

The functionality is realized with four independent image acquisition threads, each controlling one active camera. In case the user wants to receive not only the original image but also other data like projections or modified images, additional calculations are activated in the corresponding loop. The system is controlled by human-readable XML commands being sent over the same socket connection than all the other data.

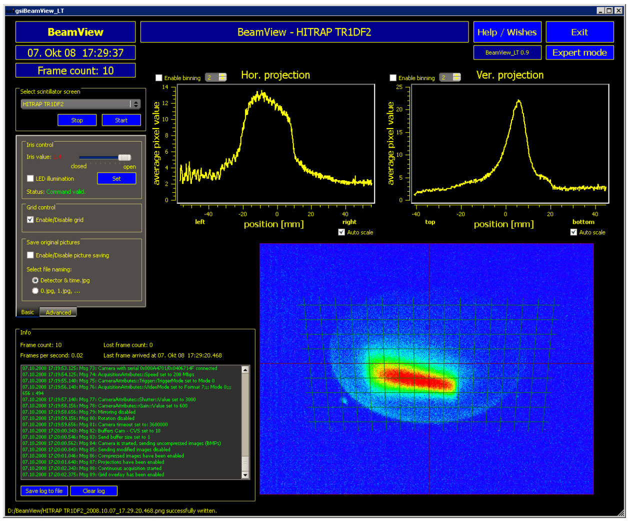
To deliver images, processed data and command responses, an application layer protocol is used to distinguish between XML and binary data. An API to the protocol allows easy access to the arriving data and the submission of commands. Communication takes place over standard Ethernet sockets using the TCP/IP protocol. Since Ethernet is a non-deterministic technology, buffers are implemented on the CVS. When there is heavy network traffic or the client is not ready to receive data, the CVS stores all data in a configurable ring buffer. The submission is continued as soon as the network is not overloaded any more or the client acknowledges the reception of the last data package.

Software on the PC

Using the API for the network protocol, a wrapper class for cameras has been coded to provide easy access to their complete functionality. A typical startup of a camera in triggered mode would look like that:

- connect to the CVS via a TCP/IP socket
- tell the CVS to use a specific camera
- initialize the camera in triggered mode
- optional: enable compression, projections/mirror, etc.
- start the acquisition

As soon as a trigger arrives at the camera, a picture is taken and sent to the client. A screenshot of the software is shown in Fig. 3. The original grey-scale images can be color-coded by the client, including a spreading of the color values. This is especially useful when the signal-to-noise ratio is low. Also a binning for the projections can be enabled to decrease statistical fluctuations.



07. Okt 08 17:29:37

Frame count: 10

HITRAP TR1DF2

Iris value: 1.4

□ LED illumination

Status: Command valid.

☑ Enable/Disable grid

□ Enable/Disable picture saving

● Detector & time.jpg

○ 0.jpg, 1.jpg, ...

□ Enable binning 2

Frame count: 10 Lost frame count: 0

Frames per second: 0.02 Last frame arrived at 07. Okt 08 17:29:20.468

```
07.10.2008 17:19:53.125: Msg 73: Camera with serial 0x000A4701/0x0406714F connected
07.10.2008 17:19:54.125: Msg 74: AcquisitionAttributes::Speed set to 200 Mbps
07.10.2008 17:19:55.140: Msg 75: CameraAttributes::Trigger::TriggerMode set to Mode 0
07.10.2008 17:19:56.140: Msg 76: AcquisitionAttributes::VideoMode set to Format 7;;; Mode 0;;;
656 x 494
07.10.2008 17:19:57.140: Msg 77: CameraAttributes::Shutter::Value set to 3000
07.10.2008 17:19:58.156: Msg 78: CameraAttributes::Gain::Value set to 600
07.10.2008 17:19:58.656: Msg 79: Mirroring disabled
07.10.2008 17:19:59.156: Msg 80: Rotation disabled
07.10.2008 17:19:59.656: Msg 81: Camera timeout set to: 3600000
07.10.2008 17:20:00.343: Msg 82: Buffers Cam - CVS set to 10
07.10.2008 17:20:00.546: Msg 83: Send buffer size set to 1
07.10.2008 17:20:00.562: Msg 84: Camera is started, sending uncompressed images (BMPs)
07.10.2008 17:20:00.843: Msg 85: Sending modified images disabled
07.10.2008 17:20:01.046: Msg 86: Compressed images have been enabled
07.10.2008 17:20:01.640: Msg 87: Projections have been enabled
07.10.2008 17:20:02.343: Msg 88: Continuous acquisition started
07.10.2008 17:20:02.375: Msg 89: Grid overlay has been enabled
```

D:/BeamView/HITRAP TR1DF2_2008.10.07_17.29.20.468.png successfully written.

Figure 3: BeamView in operation at the HITRAP [6] experiment.

The complete programming of the client application including the API was done in C++ using the Qt framework [4]. Qt provides classes for platform-independent GUI, network and XML functionality, and is available for all mayor operating systems. The BeamView client has been compiled under Windows as well as Linux without changing a single line of code. It consists out of two layers with one of them being the C++ API. The API is derived from the Qt socket class (QTcpSocket) and handles incoming and outgoing data, XML commands as well as binary data like images. To process data on the client, extensive use of the Qt signal/slot mechanism is made. When data has been read from the network, a Qt signal, comparable to an interrupt, is raised, with a pointer to the new data as an argument. The GUI layer receives this signal and updates itself with the new data. To avoid read/write concurrency issues, memory locking and acknowledge-mechanisms have been implemented.

For all additional devices which can be controlled via Ethernet such as iris, pressure control and triggering, separate C++ classes have been created. Communication takes place over separate TCP/IP connections which are independent from the connection to the CVS. Instances of these classes are member variables of the camera class and get called on user GUI interaction.

SYSTEM PERFORMANCE AND USER EXPERIENCE

Before installing the system at the accelerator, the performance was tested extensively. This concerns the CVS itself, as well as the export of the screen output from the client to other displays.

On the CVS side, the limiting factor seems to be network communication in combination with the CPU. For the tests, the camera has been triggered with variable frame rates. The resulting CPU load of the CVS has been measured, transmitting compressed (JPG) or uncompressed (BMP) images with projections. Additionally the highest achievable frame rate has been determined, before frames get lost. The results are shown in Table 1. It can be seen, that the CPU load is higher when sending great amounts of image data, although the compression step is omitted. The compression step seems to be faster than the transmission of data. When uncompressed images are sent to the client, frame losses occur at ~16 frames per second (fps) at 42 MBit network load. When compressed images are requested, the maximum frame rate is between 30 and 35fps with ~11MBit network load. The maximum fps of the compressed images jitters slightly due to the variable size of the compressed image. These results are valid, when the client displays the images on the local screen, in contrast to exporting the screen to another display.

Table 1: Measurement of the CVS CPU load and network traffic, comparing uncompressed and compressed images

fps	3,75	7,5	15	30	max. fps
BMP	21% 9MBit	43% 18MBit	88% 36MBit	100% 40MBit	16fps 42MBit
JPG	13% 1MBit	25% 3MBit	52% 5,5MBit	74% 10MBit	35fps 11MBit

The X-Terminals used in the GSI main control room are another bottleneck. When an application is executed, it will be started on a Linux machine in a server room and has to export its display to an X-Terminal. Presently, the native Linux X-technology seems to be too slow transmitting VGA image streams effectively: When the screen is exported to an X-Terminal, the maximum frame rate is between 4 and 15fps, depending of the color depth of the X-Terminal (8 bits per pixel vs. 24 bits per pixel). Tests to speed up the display export by using an export mechanism with integrated compression are currently under investigation.

In daily operation, high frame rates are rarely required. BeamView has been used to collect data for different experiments, e.g. scintillator screen investigations with frame rates typically between 2 and 15 fps. [5]. Over 28 GB of images have been stored without data loss during the whole measurement of six hours.

The HITRAP experiment [6] used an installation of six cameras with scintillator screens to monitor the beam through a decelerating beamline. Here the false-color mode was of great benefit, since the light yield of the beam was extremely low, proving the necessity of a high performance digital camera system.

CONCLUSION AND OUTLOOK

BeamView is already in use at GSI. The current version is optimized for the observation of scintillator screens with a single active camera. The expansion for BIF applications is currently under development, allowing to view two cameras per BIF station and multiple BIF stations along the beamline simultaneously.

REFERENCES

[1] F.Becker et al., "Beam Induced Fluorescence Monitor for Transverse Profile Determination", DIPAC'07, Venice, (2007).
[2] Company National Instruments, www.ni.com
[3] Company Ingenieurbüro Hagel, www.ib-hagel.de
[4] Company Qt Software, www.trolltech.com
[5] P.Forck, E.Gütlich, R.Haseitl, "Scintillation Screen Investigations for High Current Ion Beams at GSI Linac", BIW'08, Lake Tahoe, (2008).
[6] H.-J. Kluge et al., "HITRAP: A Facility at GSI for Highly Charged Ions", Advances in Quantum Chemistry, Volume 53, Pages 83-98, (2008).