TINE/ACOP STATE-OF-THE-ART VIDEO CONTROLS AT PETRA III

J. Bobnar, I. Križnar, T. Kusterle, Cosylab, Ljubljana, Slovenia
D. Melkumyan, S. Weisse, DESY Zeuthen, Zeuthen, Germany
P. Duval, G. Kube, J. Wilgen, DESY, Hamburg, Germany

Abstract

The TINE/ACOP video system is a complete state-of-the-art solution for streaming beam video, featuring live analysis and live beam image display inside ACOP video component, which can be placed in any Java Swing panel. After a number of iterative improvements and embellishments, the system has matured to stable production quality in the beginning of year 2010. The system consists of the following components: a TINE device server captures a video image [1] and encodes it to the standard TINE IMAGE format. The TINE transport layer streams the IMAGE objects to clients as it would any other data chunk [2]. The Java TINE client passes the IMAGE object through the analysis Java bean, which then performs fast statistical analysis of beam position and size. The streamed image plus analysis data are displayed in the Java video component, which is part of the ACOP components. Additional capabilities are background subtraction, automatic or manual threshold subtraction, enhanced coloring and saving snapshot as PNG file. Optionally, the analysis bean can be used standalone as a common service and results are further distributed via an intermediate TINE server written in Java.

INTRODUCTION

The origin of the TINE Video System goes back to the design of the Photo Injector Test Facility Zeuthen (PITZ), which is a test facility for research and development on laser driven electron sources for Free Electron Lasers and linear colliders [1]. The optimization of an electron gun is only possible with the help of an extensive diagnostic system, including the video system.

The whole video system includes a rich set of components, covering the low level hardware integration and image grabbing, to the transport protocol and data visualization tools.

In this article we will focus on the upper level of components, which have recently been upgraded and put to use also at DESY Hamburg.

DATA ACQUISITION AND TRANSPORT

The image acquisition is implemented in a grabber server written in C++. The main purpose of this server is to acquire grayscale images from the image source and pre-process the data (e.g. compression).

The transfer of the high resolution image (up to 2 megapixels) is done using the TINE transport protocol. TINE allows various choices of data transport including multicasting, unicast UDP and TCP. Combining this with compression algorithms the TINE video system easily achieves updates at 10 frames per second.

The image transported by TINE is packed into a dedicated IMAGE data type, which is composed of an image header providing meta information about the image (frame size, bit depth etc.) and the actual image data of variable size – TINE is not limited to the transport of a fixed size image, but can be used to transfer any size one desires (within the limits of the network traffic). The IMAGE data type can also be embedded within TINE structures and is used as a standard method of exchanging image data between video system components.

IMAGE VISUALIZATION AND ANALYSIS

Java has been selected as the target platform/technology for the video system clients. The client side is responsible for visualization of the image as well as performing the data analysis and processing of the image data. In some respects we might expect Java to reduce the execution speed of the software, which would be a trade off for platform independence. This does in fact play a role regarding for instance graphics or low-level networking functionality. However, due to the high processing power of today’s desktop computers, this is no longer a serious drawback and Java has proven to be very powerful and easy to use for writing the video clients.

A dedicated AcopVideo bean has been implemented, following the conventions and standards of the ACOP framework [3]. This automagically provides some common functions and tools (e.g. connection selection, drag and drop), as well as makes it easy for other developers to provide rich-clients that deal with the video.

The AcopVideo bean was implemented in pure Java, which means that it doesn’t use any native resources (besides the standard ones provided by JVM) and is completely platform independent. The AcopVideo bean was designed with performance in mind, which drove the architecture and implementation of the drawing algorithm. The performance of the video bean today easily satisfies the requirements of the operation control.

In addition to high performance, the video bean provides much functionality, which is not available in the older native or commercial video clients. The AcopVideo can display any TINE video channel or still image, which can be either loaded from several standard image files (JPEG, PNG, etc.) and quality (8 to 24 bits per pixel), or provided through the TINE channel (using the event notification system in order to minimize the necessary network traffic). The AcopVideo also offers several other options for image visualization and enhancements, such as different color modes for luminosity data, histogram equalization, aspect ratio changes and zooming, display of meta information etc.

IMAGE PROCESSING AND ANALYSIS

For better understanding and easier interpretation of the image additional analysis is required. From the image of the beam one can extract the emittance as well as other properties, which the control operators are interested in. However, the extraction of such data requires a deep analysis and statistics calculation on the image data on the fly while the image is streamed from the server. Such analysis can by itself be extremely time consuming, and it can be also very difficult to perform if the beam does not have ‘regular’ shape (round or at least elliptical). Consequently much effort has been put into finding the most reliable and fastest analysis solution.

Statistical Analysis

The basic analysis of the image is done by calculating the statistical parameters of the beam. Using simple statistical algorithms (assuming that the beam has a non-sparse approximately elliptical shape) the mean value and standard deviation of the beam profile are calculated. The 2-dimensional analysis of the image also provides the rotational parameter of the beam ellipse.

In addition to this analysis, a side-view projection of the image is also analyzed. The pixels in a single row (and column) are summed together, what leads to two 1-dimensional profiles – one for horizontal axis and one for vertical. Similar as for the 2-dimensional analysis, the statistical parameters are also calculated for the side projections.

The calculated parameters can be used for the first approximation of the image interpretation. They provide reliable information when the image has low noise and no additional artifacts such as side light or camera pixel gain defects, split beam etc. AcopVideo bean provides functions for easy display of these parameters together with the live image; crosshair marker is used for display of the mean and standard deviation, while the side projections are plotted at the bottom and side of the image (see Figure 1). The calculated parameters can also be extracted separately and displayed for example in a dedicated table or used in further analysis.

Analysis Improvements

When the image is noisy (or generally not regular), additional algorithms have to be used to obtain better results. Thus, a Region of Interest (ROI) was introduced. When the beam is localized to a small part of the total image one can select a narrow area around the beam peak to reduce the size of the image that needs to be analyzed. Usage of the ROI significantly improves the statistical parameters, since it eliminates the contribution of the noise or other artifacts in the distant regions from the beam peak (see Figure 1).

Another improvement is the usage of a threshold value, which defines the minimum values that a pixel has to have in order to be included in the calculation. This eliminates low amplitude noise in the dark areas and puts more weight on the bright area, where the beam is located. The threshold value can be either explicitly specified by the user or calculated automatically. In the former case the threshold value is a constant in time; in the latter case the user specifies a region within the image (usually the dark region) and that region is used to calculate the mean pixel value. The mean value is then used as the threshold value during the analysis. In this case the value changes at each image update.

Next round of improvement introduced the use of the background image subtraction. A still image representing the background (the image area the beam turned off) is subtracted from each frame in order to eliminate permanent artifacts of image background. User can choose between a pre-stored image from the file system or grab a live image from the TINE channel (in the latter case the beam has to be turned off during that time in order to obtain the only background). The selected image is then subtracted from the original live image, which produces an artifact-free image used for further processing.

Further improvement of analysis was achieved by introducing a smoothing algorithm. When the image is extremely noisy, smoothing can be used to average out the noise. For each pixel the new value is calculated as the average value of a few points around the particular pixel. This can lead to more stable and reliable statistical analysis results.

Best Fit Analysis

In certain cases it turned out that even with all the aforementioned improvements, the statistical analysis still does not provide good enough results. While the mean value is approximately correct, the standard deviation might overshoot. To overcome this problem an additional algorithm has been implemented, which calculates the beam properties more precisely.

Least square curve fitting algorithm was implemented to find the best fit for the beam image side projections. We decided to use a Gauss function with linear background:

$$y = A \exp\left(\frac{(x - \overline{x})^2}{2\sigma^2}\right) + k\,x + n,$$

where

$$A, \overline{x}, \sigma, k, n$$

are the fitted parameters. To find the numerical solution to these parameters we have implemented the Levenberg-Marquardt algorithm [4]. In most cases the algorithm converges to the proper solution, but to guarantee better stability good starting values should be provided. For the first guess the statistical analysis results posed as a good guess and after the curve is fitted for the first time all consequent fits can be obtained starting with the previous results, since the beam changes are usually very slow (two consequent frames do not differ much).

This algorithm has proven to be much more reliable and trustworthy than the statistical analysis already without the use of improvements discussed in the previous sections. If combined with the background image and threshold calculation, the algorithm produces very stable and accurate results (see Figure 1).

The drawback of the least-square fitting algorithm is that it consumes significantly more time than the statistical analysis. On a desktop PC it is still possible to observe the live image up to about 2 fps, which is in most cases enough for normal operations, but at higher rates frame drop might occur. Therefore, the user has the option to turn on or off each individual feature in order to display only the values he is interested in.

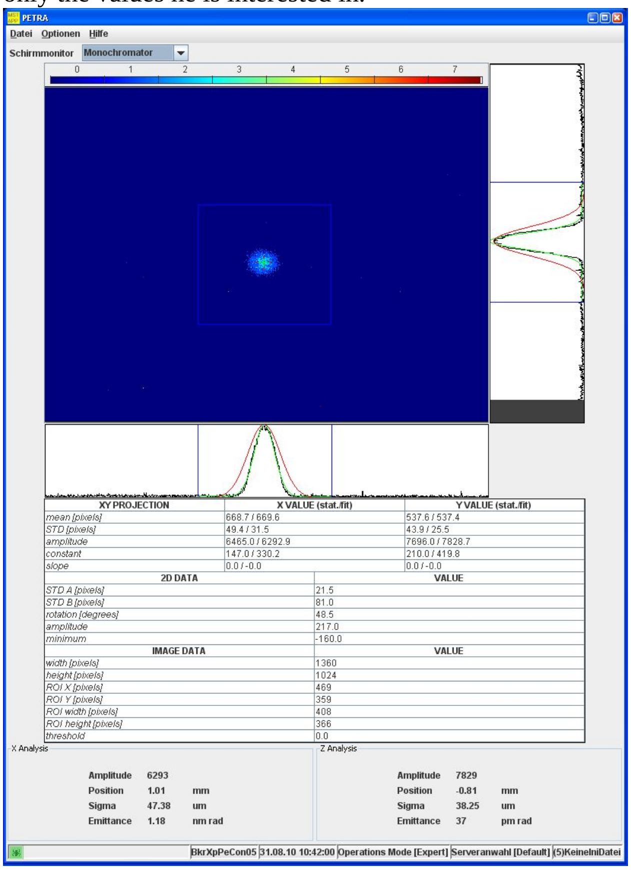

Figure 1: AcopVideo bean displaying a live beam and its profile. A region of interest is chosen around the peak of the beam (blue rectangle). Red curve is the result of statistical analysis; green curve is fitted gauss function. The table is used to show the numerical values of the analysis results.

Modularity and Analysis Server

The image analysis has been implemented independently of the video bean. This allows its usage at any level on which someone is interested into the beam analysis. The analysis bean can simply be used as an extra layer between the source of the image and the destination.

The modularity of the analysis has been used by the general analysis server, which can be used to perform the analysis instead of a desktop computer. The analysis server is written as a regular TINE server, which is registered in the TINE Equipment Naming Service and can be used as a source for the AcopVideo. All that the server requires is the TINE channel which supplies the live image and the output of the server is a dedicated data object, holding the original image and all the calculated parameters. The use of the analysis server lowers the CPU usage on the client PC, which is particularly useful when one wants to observe the analysis by several different clients (on the same or on multiple computers). However, the downside of the server is that it generates a bit more network traffic since it has to send more data (including the side view profiles etc.).

AcopVideo bean is designed in a way that it can use both the local analysis (on the client computer, where the AcopVideo is running) and the remote analysis (connected to a remote server). User is able to switch between the two options in run-time and use the one that is more appropriate at any given time.

CONCLUSION

Much progress has been done on the video system since the beginnings. A lot of effort has been put into development of high performance tools, which can be used in day-to-day operations in the control room. The recent image analysis implementation made the video applications much more than just a simple visualization tools – it became a powerful diagnostic tool for online emittance diagnostics at PETRA which tremendously helps the operators in the control room to achieve full accelerator performance.

Nevertheless, there is still a lot of room for improvements. The next step is the optimization of the transport and compression of the image, which might consequently require the optimization of the analysis algorithms. The analysis itself also leaves options for further development, such as for example 2-dimensional Gaussian fit.

REFERENCES

[1] S. Weisse et al., "Status of a versatile Video System at PITZ, DESY-2 and EMBL Hamburg", ICALEPCS 2007 Proceedings, Knoxville, TN, USA

[2] S. Weisse et al., “TINE Video System: proceedings on redesign”, ICALEPCS 2009 Proceedings, Kobe, Japan

[3] J. Bobnar et al., “The ACOP Family of Beans: A Framework Independent Approach”, ICALEPCS 2007 Proceedings, Knoxville, TN, USA

[4] W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery, “Numerical Recipes in C, The art of Scientific Computing, 2nd Edition”, Cambridge University 1992