GigE カメラを用いたスクリーンモニタの EPICS 制御システムの開発

DEVELOPMENT OF EPICS CONTROL FOR SCREEN MONITOR USING GIGE VISION CAMERA

田丸哲也#, A), 内藤孝 B), 塚田義則 A),早川厚 A),坂入崇 C)
Tetsuya Tamaru#, A), Takashi NaitoB), Yoshinori TsukadaA), Atsushi HayakawaA), Takashi Sakairi C)
A) Kanto Information Service (KIS)
B) High Energy Accelerator Research Organization (KEK)
C) Iwac Service Ino

Abstract

KEK's Accelerator Test Facility (ATF) has 15 screen monitors installed in ATFLINAC and 8 screen monitors installed in BT line. A random shutter camera (NTSC signal output) was used as an imaging camera, but 1) image quality problems due to signal transmission using the CATV system, 2) camera malfunction problems due to noise, 3) Due to problems such as maintenance of the CATV system, updating the camera has been considered. Therefore, we started to introduce a GigE Vision standard camera that can transmit image signals over a long distance from around 2012. The software uses a camera library created by Aravis, an open source library that supports GigE Vision, and developed camera control by EPICS. The operation speed of image acquisition by this system was 42 ms. Since February 2020, all cameras have been replaced with GigE Vision standard cameras and operation has been started and they are operating without problems. In this presentation, we will report the configuration of the updated screen monitor, its operation status, and future prospects.

1. はじめに

KEK の先端加速器試験施設(ATF)では、ATFLINAC に 15 台、BT ラインに 8 台のスクリーンモニタが設置されている。 撮像用のカメラとしてランダムシャッターカメラ(NTSC 信号出力) が用いられていたが、CATV システムを用いて信号伝送しているための画質の問題、ノイズによるカメラの誤動作の問題や CATV システムのメンテナンスの問題等がありカメラの更新が検討されてきた。 その為、画像信号の長距離伝送が可能な GigE Vision 規格[1]の JAI[2]製カメラ CM-030GE[3]の導入を 2012 年頃から開始した。導入は進められていたが生産中止となったため、同じ JAI 製カメラの GO-2401M-PGE[4]を検討し、導入を進めた。

GO-2401M-PGE を制御するにあたり、ソフトウェアは GigE Vision をサポートしているオープンソースライブラリ Aravis[5]を利用してカメラライブラリを作成し、これを利用した EPICS によるカメラ制御を開発した。 EPICS を経由したこのシステムによる画像取得の最短動作速度はトリガーOFF で 42 ms であった。2020 年 2 月からカメラを全て GigE Vision 規格のカメラに置き換えて運転を開始し問題なく運用されている。

本発表では、更新したスクリーンモニタの構成と、運用状況、今後の展望について報告する。

2. 開発

2.1 GO-2401M-PGE について

GO-2401M-PGE は JAI 製の小型エリアスキャンカメラである。全長は C マウント部含め 51.5 mm、幅 29 mm と小型で、重量も 46 g と軽量である。ピクセル数は縦 1216 横 1936 の 235 万画素で、最大フレームレートは 41 fps である。カラーはモノクロ、通信はギガビットイーサネットで、外部トリガーにも対応している。プロトコルは GigE Vision で、制御は JAI から提供されている Windows アプリケーションを使用するか、先述した Aravis で制御出来る。

実際に使用しているカメラを Fig. 1 に示す。

Figure 1: JAI GO-2401M-PGE.

現行品で、外部トリガーに対応しているスタンダードな製品はいくつかあるが、コストも比較的安価で、画素数も多く高性能という事から、GO-2401M-PGE を選択した。

GO-2401M-PGE は PoE[6]に対応している為、PoE（IEEE802.3af）対応のケーブルとハブを用意出来れば別途電源を用意する必要は無く、簡単に設置が可能だが、今回は設置環境やコストを考慮して別途電源を用意した。

2.2 規格と Aravis ライブラリ[7]

GigE Vision とは、イーサネット経由で産業用カメラの制御、撮像した映像信号をパソコン等に伝送するためのプロトコルである。ギガビットイーサネットでの伝送が可能であり、I/O ボードを必要としないため、イーサネットケーブル 1 本で導入が可能である。最大伝送距離は 100 m で、スイッチングハブを増設する事で距離の延長が可能など、ギガビットイーサネットの利点を受け継いだ規格である。

Aravis は GenICam(Generic Interface for Camera)[8]規格に準拠した、glib/gobject ベースのオープンソースライブラリである。GenICam は、GigE Vision や USB3 Vision といった異なるプロトコルであっても共通の API で制御することが出来るソフトウェアインターフェースである。これにより Aravis を用いて、異なるプロトコルでも制御が可能となっている。最新のバージョンは 0.8 で、カメラの機能であるキャプチャ、シャッタースピード、ゲイン、ビニング等のカメラの基本機能が Aravis を用いれば利用可能である。ただし、設定値の保存や、ネットワークの設定に関しては設定が出来ないため、ベンダーから提供されているアプリケーションを使用して設定をする必要がある。

2.3 開発環境

カメラ制御の開発に用いた言語、ライブラリ等を Table 1、スクリーンソフトを Table 2 に示す。

Table 1: Languages and Libraries Used to Develop Camera Control

Development language	C
EPICS version	3.14.12.5
Library	Aravis 0.6

Table 2: Languages and Libraries Used to Develop Screen Software

Development language	C++
GUI Editor	QtCreator-3.1.2
Library	Qt-4.8.7,cfitsio-3.48

カメラ制御はオープンソースライブラリ Aravis を用いて作成し、その制御を EPICS で行う様にした。カメラは複数台あるので、マクロでスクリーン毎にレコードを用意した。

撮像した画像を解析するスクリーンソフトは Qt というアプリケーションプラットフォームで作成した。fits フォーマット[9]で画像のセーブ、ロードが出来る様に、cfitsio を利用した。

Linux ディストリビューションは 64 bit の CentOS7.6 を採用した。メモリは 4 GB で、ディスク容量を考慮して、サーバーを NFS マウントした先に EPICSIOC やライブラリ等を保存する様にした。

3. スクリーンソフト

スクリーンソフトを Fig. 2 に示す。

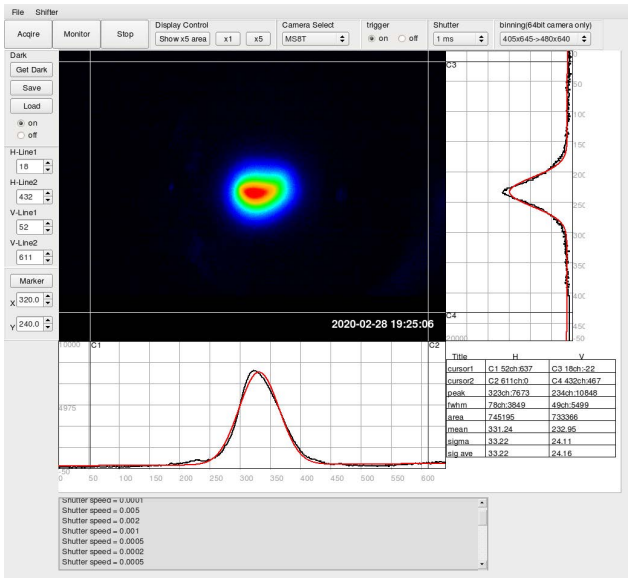

Figure 2: Screen Software.

スクリーンソフトでは、トリガーの ON/OFF や、シャッタースピードの変更といった、カメラの機能が設定できる。スクリーンソフトの機能としては、キャプチャ機能、ビニング、画像のプロファイル、ダーク補正、拡大機能や画像データのセーブ、ロードが出来る。ビニングは 1216x1936 の 3x3 マスを 1 マスとして平均して 405x645 に直し、480x640 でスクリーンソフト上に表示している。拡大機能は 2 通りの拡大ができ、480x640 から 96x128（x5 倍率）の 1 マスを 5x5 マスとして 480x640 で表示する機能と、1216x1936 から 480x640 を取得して表示する事が出来る。また、スクリーンソフト上に表示されている白線をドラッグする事でプロファイルする範囲を決める事が出来る。

スクリーンソフトは、他の画像解析でも利用している XSR カメラや STREAK カメラの画像解析ソフトを参考にして作られており、基本機能を共通にして開発思想を統一化する事で、開発効率を上げる事が出来た。

4. 構成

ATF で設置されているカメラの構成を Fig. 3、Fig. 4 に示す。

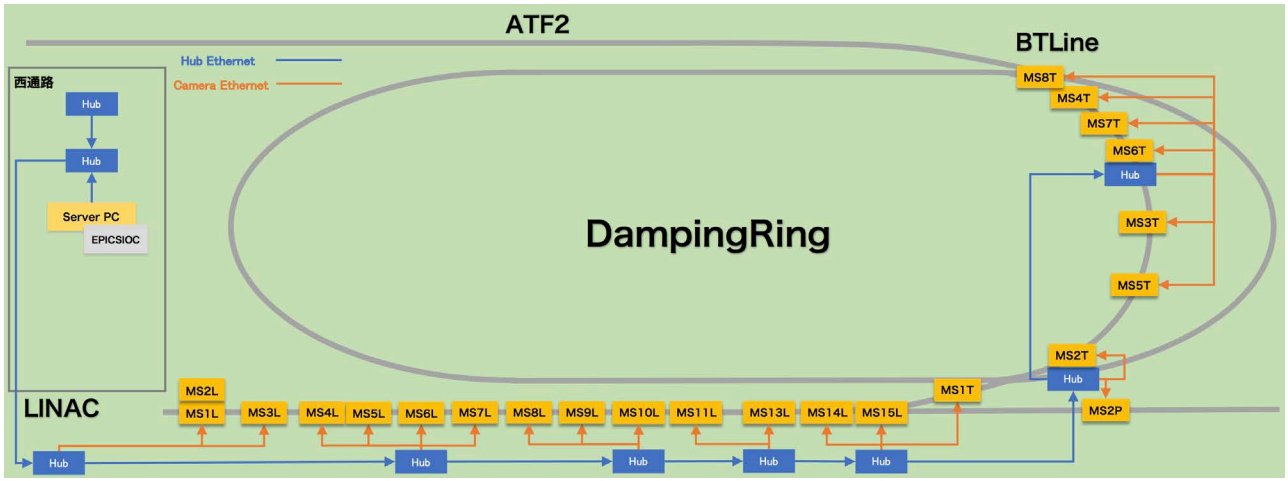

Figure 3: Camera Installation in ATF tunnel.

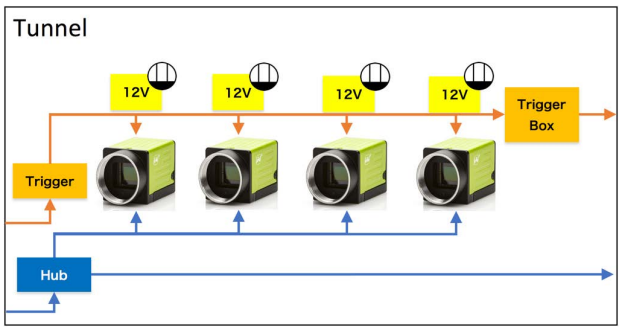

Figure 4: Camera Configuration.

カメラは LINAC から BT Line までに計 22 台設置されている。電源は 12 V のスイッチング電源で供給をしている。スイッチングハブを適宜、間に挟む事でイーサネットケーブルの延長と、敷設を容易に行える様にした。DHCP が有効になっているため、仮に故障しても、EPICS IOC のスタートアップスクリプトを編集して、カメラを交換するだけですぐに対応出来る様になっている。

カメラのネットワークは Server PC に直結されたローカルネットワークではなく、他サーバーが繋がっているネットワークに接続している。スイッチングハブを設置してあるので、他サーバーへのネットワークトラフィックの影響は少ない。

Server PC でのトラフィック量が大きすぎると EPICS のチャンネルアクセスにも影響を及ぼしかねないので、カメラ設置後のネットワークトラフィックの調査を行った。結果を 5.1 に示す。

5. 性能評価

5.1 ネットワークトラフィック

iftop コマンドでネットワークトラフィックを調べた。1 台辺りのトラフィック量を Table 3 に示す。

Table 3: Camera Network Traffic

Action	Server PC -> Camera	Camera -> Server PC
Waiting	40 B/sec	46 B/sec
Capture	128 B/sec	2.30 MB/sec

カメラとの接続が完了した後、通信待機中での通信量は合わせて 86 B/sec 程度だった。これは定期的に送られる HeartBeat が含まれているからだと思われる[10]。待機中における 22 台全体の通信量は、1892 B/sec 程度かかる。1216x1936 をキャプチャする 1 台の通信量は 2.30 MB/sec 程度であった。カメラは同時に 1 台しか使用しないので、2.30 MB/sec が実質のトラフィック量である。この結果から、トラフィックの影響はほぼ無いと判断した。現在でもカメラ敷設によるネットワークへの影響は起きていない。

5.2 トリガー時の動作速度

ATF 加速器のマシンサイクルは 0.3 秒となっており、0.3 秒のトリガーを検知出来る必要がある。そこで、100 μs と、10 ms のシャッタースピードで、パルス幅を 100 μs とし、安定してキャプチャ可能な最短周期のトリガーを調査した。

キャプチャを行う EPICS レコードに対して 50 回 caput をループさせるプログラムを 2 セット行い、カメラ制御プログラム内に、gettimeofday 関数で時間の計測を行った。トリガー周期が 60 ms でシャッタースピードが 100 μs の結果を Fig. 5 に、トリガー周期が 70 ms でシャッタースピードが 10 ms の結果を Fig. 6 に示す。

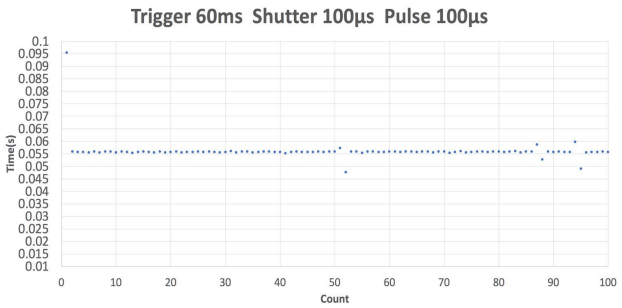

Figure 5: Trigger 60 ms Shutter Speed 100 μs Measurement Result.

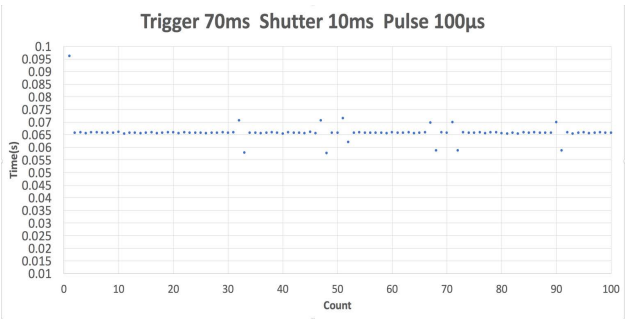

Figure 6: Trigger 70 ms Shutter Speed 10 ms Measurement Result.

トリガーに対して取得出来る最短周期は、シャッタースピードが 100 μs で、トリガーの周期が 60 ms の場合、動作時間は 55 ms 程度で取得している。シャッタースピードが 10 ms で、トリガーの周期が 70 ms の場合、動作時間が 65 ms 程度で取得しており、3 Hz のトリガーに対し動作可能である事が分かった。

時間が稀に遅くなっている時、キャプチャデータ取得からビニングまでの時間が通常より遅いケースが見受けられた。他にもカメラとの通信状況、プログラムの実行速度やオーバーヘッド等の影響が含まれているものと思われる。その後の動作時間が速くなっているのは、遅くなった分だけ次のトリガーを待つ時間が短くなった為かと思われる。1 セット、2 セットの 1 回目はキャプチャのタイミングがランダムのため、動作時間はキャプチャするタイミングによってブレが生じる。設定出来る最小のシャッタースピードの 73 μs を設定しても結果はほぼ同じだった。

この周期より 5 ms 早くした場合、動作時間が設定したトリガー時間の 2 倍時間がかかるケースが発生した。これは処理中に最初のトリガーを検知出来ず、次のトリガーを検知したためだと思われる。

次にトリガーOFF 時の測定結果を Fig. 7 に示す。

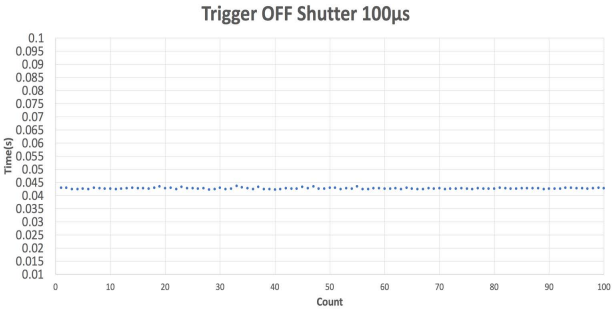

Figure 7: Trigger OFF Measurement Result.

トリガーOFF 時でシャッタースピードが 100 μs の場合だと 42 ms 程度で取得が出来る。トリガーの有無を除いた場合、42 ms が最短の速度となる。

6. まとめ

ランダムシャッターカメラを GigE カメラに全て置き換え、特に問題無く運用を続けている。故障したケースは起きていないが、放射線による影響や、外的要因による影響を考慮して、耐久性を調べていく予定である。

今後、安定に運用する為に、ハードウェア、ソフトウェアの両面で様々なトラブルに対応出来る様に予備機の用意や、故障した際に短時間で交換出来る様にソフトウェア、マニュアルを整える予定である。

参考文献

[1] https://www.baslerweb.com/jp/vision-campus/interfaces-and-standards/gigabit-ethernet/
[2] 株式会社ジェイエイアイコーポレーション
[3] https://www.jai.com/jp/products/cm-030-ge
[4] https://www.jai.com/jp/products/go-2401m-pge
[5] https://github.com/AravisProject/aravis
[6] Power over Ethernet
[7] https://www.baslerweb.com/jp/vision-campus/interfaces-and-standards/genicam-standard/
[8] http://jiia.org/wp-content/themes/jiia/pdf/report2007_01.pdf
[9] https://heasarc.gsfc.nasa.gov/fitsio/
[10] https://sentech.co.jp/information/faq/oldfaq-faq/gige-old/software-gige/457