LCLS

title: # XAL ProfileMonitor for LCLS

document:

revision: 0.05

project: LCLS

authors: Sergei Chevtsov

pages: 23

LINAC COHERENT LIGHT SOURCE STANFORD LINEAR ACCELERATOR CENTER

XAL ProfileMonitor for LCLS

Revision History:

Revision	Revised By	Revision Date	Description of Changes
0.01	Sergei Chevtsov	Mar-12-2008	Initial Draft
0.02	Sergei Chevtsov	Mar-26-2008	Incorporated Debbie Rogind's comments (incl. change of the document/application title)

0.03	Sergei Chevtsov	Mar-31-2008	Added a use case for saving the average of background images; mentioned that a GUI shall contain reasonable default values; added an optional use case for manually annotating images.
0.04	Sergei Chevtsov	Apr-01-2008	Added/modified information after the initial review: starting/stopping streaming of live images; 1 Hz minimum for rate of live streams; a higher rate of buffered acquisition by laser cameras; support of acquisition of only 1 background image per dataset; time constraints on loading data from an RDB; new calculated beam parameters, beam fit types (lineout), data quality indicators, algorithm; manual cropping via halving of current image area; saving to Elog simultaneously prompts to save image data; exported Matlab structure as shallow as possible.
0.05	Sergei Chevtsov	Apr-29-2008	Each camera belongs to an area; all images are processed in full and in sliced modes- user selects which he wants to see; save per camera the default number of images and whether to retrieve a new background image; during acquisition, show images as part of the progress widget.

Table of Contents

1 Project Drivers

1.1 Purpose of this Project

1.1.1 Background of the Project Effort

LCLS features two types of beam cameras: “screen cameras” and “laser cameras”. Each screen camera points at a screen that (when inserted into the path of an electron beam) acts as a beam profile monitor. Laser cameras are at the end of laser beam lines; they are used for beam alignment.

Each LCLS camera continuously broadcasts live grayscale images as well as supporting parameters at a rate of up to 2 Hz; data is available via EPICS PVs.

In addition, each screen camera provides the capability to buffer images of electrons hitting the previously inserted screen at a much higher rate (~10 Hz), while each laser camera can buffer images at a rate of up to 30Hz. Buffered images from screen cameras are stored in a common memory that due to budgetary constraints is located on a single EPICS IOC.

LCLS physicists have written Matlab scripts that extract various beam parameters from a beam image (such as the coordinates of the beam centroid, statistical profiles and fits of the electrons distribution, etc.).

One of our ultimate goals is to measure the beam emittance in order to improve the quality of LCLS beam.

1.1.2 Goals of the Project

- To create an XAL application that helps to acquire and analyze images from LCLS cameras.
- To develop an expandable Java library (API) for acquiring and processing beam images, with the results of image processing to be available to other applications (such as “bunch length measurement”, “emittance measurement”, etc.).
- To document the image acquisition algorithm.

1.2 Client, Customer and Other Stakeholders

1.2.1 The Client

- LCLS Controls (Hamid)

1.2.2 The Customer

- Physicists, operators, high-level application developers.

1.2.3 Other Stakeholders

- LCLS users
- XAL community

1.3 Users

1.3.1 The Hands-on Users of the Product

Physicists, operators, and high-level application developers

1.3.2 Priorities Assigned to Users

All users have the same priority.

1.3.3 User Participation

A number of physicists and at least two high-level application developers (the developers of "the bunch length measurement" and "emittance measurement" applications) will provide input for requirements.

1.3.4 Maintenance Users and Service Technicians

People from controls department that will be responsible for routine maintenance of ProfileMonitor are:

- System Administrators (as part of XAL maintenance on LCLS production machines)
- Database Administrators (maintain the RDB tables for camera configurations)

2 Project Constraints

2.1 Mandated Constraints

2.1.1 Solution Constraints

ProfileMonitor must run on hosts in the LCLS high-level apps production environment, i.e.

- ProfileMonitor must fit into the XAL framework.
- ProfileMonitor must run on Linux.
- ProfileMonitor must be able to read from and write to all relevant EPICS PVs at all times.
- ProfileMonitor must have access to the file system and to the relational database (note: an RDB schema is going to be developed as part of this project).
- ProfileMonitor must be operational 24X7, or as close to this as possible

Camera configurations (incl. the EPICS PV names) must be kept up-to-date.

2.1.2 Partner or Collaborative Applications

- XAL framework for the LCLS control room
- Relational database (for storing camera configuration parameters)
- Bunch-length measurement application via the ProfileMonitor API
- Emittance measurement application via the ProfileMonitor API
- Correlation plots via the ProfileMonitor API

2.1.3 Off-the-Shelf Software

- Matlab Java builder

2.1.4 Anticipated Workplace Environment

OPIs in the LCLS control room, SLAC office computers over X11 (when an EPICS gateway is provided, ProfileMonitor may also be directly installed on SLAC Windows PCs).

2.1.5 Schedule Constraints

- ProfileMonitor must be released in June 2008.
- Development must fit in with other important items in the developers' schedule.

2.1.6 Budget Constraints

- Java Matlab builder might need to be purchased at the cost of $9000. A free trial version will be used first.

2.2 Relevant Facts and Assumptions

2.2.1 Facts/Assumptions

- Every LCLS camera is controlled via a similar set of EPICS PVs.
- A grayscale image is available as an EPICS waveform; necessary supporting parameters (such as image width/height) are provided.
- ProfileMonitor will reuse the functionality of available Matlab scripts for processing beam images. In the future, beam images could also be processed in camera IOCs directly, with the results made available via EPICS PVs.

3 Functional Requirements

3.1 The Scope of the Work

3.1.1 The Current Situation

ProfileMonitor already exists as a stand-alone Matlab application. It works in the LCLS control room without problems, but is difficult to maintain and extend. Also, all calculations run in the same thread, thus the performance isn't as good as it could be. Finally, ProfileMonitor uses commercial Matlab libraries, for which expensive licenses must be bought.

3.1.2 Product Boundary

ProfileMonitor is an XAL application for the LCLS control system.

3.1.3 Product Use Case List

	User(s)	Scenario (what they want to do)	Description (how they do it)	Fit Criterion (what does ProfileMonitor supply?)
1	Operator, physicist	User starts/stops to view live images from a particular camera at 1Hz.	Select a camera from the list of camera labels and press the "Start Live Streaming"/"Stop Live Streaming". Each camera belongs to an area; select an area to limit the list of available cameras.	When launched, ProfileMonitor shall load the information about accelerator areas and available camera (such as their labels and corresponding PVs) from an RDB. ProfileMonitor shall provide a "Live Image Viewer" for streaming live images at 1Hz. ProfileMonitor shall start/stop streaming on user's demand.
2	Operator, physicist	User takes snapshot of a live image for further analysis.	Press the "Take Snapshot…" button to open ImageAnalyzer for the current live image.	ProfileMonitor shall be able to open an undocked instance of ImageAnalyzer for the snapshot of a live image.

3	Operator, physicist	User acquires a new background image and a number of beam images from a particular screen camera. (1) User acquires a previously saved background image and a number of beam images.	Verify first whether another user has reserved an acquisition. If desired, change the default name of the requested dataset by editing the corresponding text field. Insert the screen into the beam line. Check "Acquire new background image" box. Enter a positive number of desired beam images into the corresponding text fields. Press the "Acquire" button. (1) Uncheck "Acquire new background image" box. Optionally, save the setup for the selected camera;	ProfileMonitor shall report on all running instances of the application whether there is an on-going image acquisition. ProfileMonitor shall present default dataset names to the user, who shall be able to modify them. ProfileMonitor shall use EPICS PVs to control screens that are associated with screen cameras. The names of PVs shall be stored in an RDB. ProfileMonitor shall check the validity of the entered number of beam images. ProfileMonitor shall be able to limit the number of beam images that user can request per acquisition (to avoid screen damage or buffer overflow). ProfileMonitor shall report the status of the current image acquisition that was launched by the same instance of the application (incl. current image, a progress bar, the estimated time until completion, and status info). ProfileMonitor shall allow user to cancel an acquisition at any time (technical note: multi-threaded implementation required). ProfileMonitor shall inform user about the number of datasets that are loaded in its memory and the number of datasets that have been saved. ProfileMonitor shall be able to save the acquisition setup per camera. (1) ProfileMonitor shall be able to retrieve a previously saved background image for each screen camera from an RDB.

4	Operator, physicist	User saves acquired image datasets as well as parameters for and results of image processing to an RDB. (1) User saves a newly acquired background image from a selected image dataset as the reference background image for the camera.	Click on the "Save" button in the window menu. Fill in the required fields in the dialog that pops up. (1) Click on the "Save Background Image" button.	ProfileMonitor shall save raw data as well as parameters for and results of image processing to RDB. ProfileMonitor shall display the progress when saving data. ProfileMonitor shall update the information about the loaded/saved image datasets. (1) ProfileMonitor shall be able to save a newly acquired background image from a dataset to an RDB for future acquisitions.
5	Operator, physicist	User loads image datasets as well as parameters for and results of image processing from an RDB. (1) User exports image data to a Matlab file.	Click on the "Load" button in the window menu. Select the desired data in the dialog that pops up. (1) Click on the "Export…" button and enter a file name.	ProfileMonitor shall present to user a list of available image data (image datasets/parameters for and results of image processing). ProfileMonitor shall display the progress when loading data. ProfileMonitor shall update the status of loaded/saved datasets, and the default name of the dataset that might be acquired next. (1) ProfileMonitor shall pre-fill the name of the export file. User shall be able to modify the file name pattern. ProfileMonitor shall keep track of a user home directory.
6	Operator, physicist	User browses through image datasets. (1) User "deletes" the current dataset.	Press the "Browse…" button to switch ProfileMonitor to the "Image Browser" mode. Select a tab with the label of your desired dataset. (1) Press the button labeled with a red "x" to the left of the dataset tabs. Confirm the "deletion" in the dialog.	ProfileMonitor shall provide an "Image Browser". ProfileMonitor shall use Swing to support tabbed browsing (see Mozilla Firefox). ImgBrowser shall report the progress of processing dataset images. ProfileMonitor shall be able to update tabs as the datasets get deleted/added (technical note: MVC-pattern required). ProfileMonitor shall "delete" a dataset by flagging it "unselect-able". ProfileMonitor must not permanently remove any data (in case a rollback is needed).

7	Operator, physicist	User browses through images of the selected dataset. (1) User marks an image as invalid.	In ImageBrowser, select a dataset. Use the scrollbar to browse through images. (1) Uncheck the box next to the image.	ProfileMonitor shall use Swing to provide the scrollbar widget. (1) ProfileMonitor shall cross out invalid images, but otherwise not change them, so that user can quickly determine the reason for invalidity during a future analysis.
8		User (re-) defines a manual crop area for an image. (1) User (re-) defines a master crop area for an image dataset.	Check the "Manually Crop Image" box to activate the cropping; rubber-band around the desired crop area. To show the full image, uncheck the "Manually Crop Image" box. (1) Press the "Master Crop Area…" button and rubber-band around the desired region of the image that pops-up.	ProfileMonitor shall support manual cropping of an image via rubber-banding; ProfileMonitor shall be able to quickly switch from a cropped image to a full image. (1) ProfileMonitor shall support the selection of a master crop area for an image dataset via rubber- banding. ProfileMonitor shall average all beam images to help user select a master crop area.
9	Operator, physicist	User analyzes a particular image from a dataset. (1) User analyzes a different image from the same dataset. (2) User compares results of analysis from different images.	Press the "Analyze" button under the desired image to switch ProfileMonitor to the "Image Analyzer" mode. (1) Select the index/timestamp of the desired image from the "Images" list in ImageAnalyzer. (2) Undock the current ImageAnalyzer panel. Press the "Browse" button to switch to ImageBrowser. Select a different image for analysis.	ProfileMonitor shall provide an "Image Analyzer". (1) ProfileMonitor shall index each image in a dataset. (2) ProfileMonitor shall support undocking of its sub-panels, e.g. ImageAnalyzer.
10	Operator, physicist	User adjusts the crop area of an image.	Press the "Adjust Crop Area…" button on ImageAnalyzer. Move the image, or change its width/height in the dialog that pops up.	ProfileMonitor shall support moving and changing dimensions of rubber-banded crop area. ProfileMonitor shall be able to switch between an individual crop area of each image and the master crop area of the dataset that the image belongs to.

11	Operator, physicist	User changes image-processing parameters for a particular image; specifically, user can: Apply a pre-defined color map; Apply filters; Display a cropped image (either calculated, or a 'manually selected', i.e. previously defined by the user, area; also, halving of current area shall be supported); Display the beam fits (lineout as well as profile)/parameters incl. sigma, current/golden/laser beam centroid, FWHM, one over e square, 3rd and 4th momenta, correlation coefficient, a quality indicator for all data; Select whether to view beam parameters for the full image or for a particular slice; change the number of slices/slice index/slice plane per image; Select one of 7 algorithms for image processing (Gaussian, Asymmetric, RMS, RMS cut peak, RMS cut area, RMS floor, Super-Gaussian); Select units (microns, pixels); Subtract background noise (either calculated, or taken from a background image). (1) User manually annotates image with text and/or arrow.	Use the widgets on the image-processing panel to change the values of image processing parameters.	Next to an image, ProfileMonitor shall display panels for setting the image processing parameters. ProfileMonitor shall show a processed image according to the selected set of image processing parameters. ProfileMonitor shall print the results of image processing and display beam fits for each beam image. ProfileMonitor shall support three color-maps (gray, hot, and jet- see Matlab) and calculate the optimal number of colors for each image and/or dataset; user shall be able to adjust the number of colors. ProfileMonitor shall support two image filters: floor and median (see Matlab); the floor filter shall always be applied. ProfileMonitor shall support auto-cropping of beam images. ProfileMonitor shall support manual cropping of images either via rubber-banding or via "halving" of current area. ProfileMonitor shall display x/y lineout and profile fits of beam data as plots on top of images, or properly aligned to the left/bottom of an image; beam centroids as ellipses with crosshairs of a specifiable color; all beam parameters (centroid, sigma, FWHM, one over e square, 3rd and 4th momenta, correlation coefficient, a quality indicator) as text. For lineout fits, user shall be able to select a point on the image or choose the brightest point as calculated by ProfileMonitor. ProfileMonitor shall draw crosshairs that intersect at the point used for lineout fits. ProfileMonitor shall support vertical and horizontal slicing of images; ProfileMonitor shall calculate both, beam parameters for the full image, and for a particular slice; ProfileMonitor shall support displaying either data. ProfileMonitor shall support pixels and microns as units for axes/images. ProfileMonitor shall support the subtraction of background

12	Operator, physicist	User prints a window.	Click on the “Print…” button in the window menu.	ProfileMonitor shall support printing.
13	Operator, physicist	User saves data to Elog.	Click on “To Elog” button.	ProfileMonitor shall print the current window to Elog and simultaneously start the sequence to save data.
14	Operator, physicist	User consults online help.	Click on the “Help” button in the window menu.	ProfileMonitor shall include a comprehensive online help.
15	High-level apps developer	User calls parts of ProfileMonitor programmatically.	Read Programmer’s Guide and JavaDoc for details.	ProfileMonitor shall provide a public API for image acquisition, browsing, and analysis.

3.2 Functional and Data Requirements

3.2.1 Functional Requirements

Note: The following requirements are ordered alphabetically by categories- and inside each category, by priority.

Requirement ID: API.1

Use Case #: 14

Description: ProfileMonitor shall include a documented public API (application programmer’s interface) for acquiring, processing, browsing, and analyzing LCLS images. A Programmer’s Guide and JavaDoc files shall be created. Examples of how to use the API shall be written.

Rationale: High-level applications (such as Bunch Length Measurement and Emittance Measurement) will re-use some of the functionality that ProfileMonitor implements.

Source: Mike Zelazny, Greg White

Fit Criterion: Public Java classes, JavaDoc, Programmer’s Guide.

Dependencies/Conflicts:

History:

Requirement ID: Camera.1

Use Case #: All

Description: ProfileMonitor shall retrieve the configurations of LCLS cameras from an RDB. An administrator shall be able to add/delete cameras and/or to change existing camera parameters.

Each camera configuration shall include the following parameters:

Area, buffer size (in number of images), damage factor (max nr. of beam images per acquisition), default nr. Of beam images, EPICS PV prefix, “is on DEV network?”-flag*, “is on PROD network?”-flag*, “is screen camera?”- flag, label, “publish beam fits?” – flag, “retrieve new background image?”-flag, saved centroids (golden orbit/laser beam); image color depth, image height/width, image offset (x/y), image orientation (horizontal/vertical flip), image origin (x/y), image resolution (um/pix).

Some of these values shall be retrieved from an RDB directly, other from EPICS PVs (in which case the PV suffixes shall be stored in the RDB).

*Only configurations for cameras that are on the current user network shall be loaded.

Rationale: This is the very basis of ProfileMonitor.

Source: Sheng Peng, Henrik Loos, Mike Zelazny

Fit Criterion: User sees live images.

Dependencies/Conflicts:

History:

Requirement ID: GUI.1

Use Case #: All

Description: ProfileMonitor shall display the timestamp and the label of the original camera next to each image.

Rationale: ProfileMonitor shall be user-friendly and easy to debug.

Source: Henrik Loos, Mike Zelazny

Fit Criterion: ProfileMonitor displays all of the information above.

Dependencies/Conflicts:

History:

Requirement ID: GUI.2

Use Case #: All

Description: ProfileMonitor shall provide GUIs with modern user-friendly features, such as: re-sizeable windows, undockable panels, window menus (File, Edit etc.). ProfileMonitor GUI shall present reasonable defaults to the user and be designed in close cooperation with LCLS physicists.

Rationale: ProfileMonitor in XAL must be an improvement on the current Matlab version.

Source: Greg White, Paul Emma

Fit Criterion: User considers the XAL ProfileMonitor friendlier than the Matlab ProfileMonitor.

Dependencies/Conflicts:

History:

Requirement ID: GUI.3

Use Case #: 12, 13

Description: Each ProfileMonitor window shall be printable, incl. to Elog. When user saves the screenshot of a window to Elog, ProfileMonitor shall simultaneously initiate the sequence to save data.

Rationale: Physicists need to publish papers of their research.

Source: Paul Emma, Greg White

Fit Criterion: User prints ProfileMonitor windows.

Dependencies/Conflicts:

History:

Requirement ID: GUI.4

Use Case #: All

Description: ProfileMonitor status panels shall display the following information:

ProfileMonitor version (incl. build date), the network on which the application is running (prod or dev), the timestamp of the saved background image (per camera)

Rationale: ProfileMonitor shall provide some basic debugging info.

Source: Greg White, Mike Zelazny

Fit Criterion: ProfileMonitor displays all of the information above.

Dependencies/Conflicts:

History:

--

Requirement ID: **GUI.5**

Use Case #: All

Description: ProfileMonitor shall report errors/warning/useful status information to the user in a dialog and/or on a status panel.

Rationale: ProfileMonitor shall be user-friendly.

Source: Sergei Chevtsov, Greg White

Fit Criterion: ProfileMonitor displays all of the information above.

Dependencies/Conflicts:

History:

--

Requirement ID: **GUI.6**

Use Case #: All

Description: ProfileMonitor shall display the progress status of each long task (one that take >5 seconds to complete) on a progress panel that includes a progress bar and the estimated time left. During acquisition, ProfileMonitor shall display the latest acquired image.

Rationale: ProfileMonitor shall be user-friendly.

Source: Sergei Chevtsov

Fit Criterion: ProfileMonitor displays all of the information above.

Dependencies/Conflicts:

History:

Requirement ID: **GUI.7**

Use Case #: 11

Description: ProfileMonitor shall display plots of beam fits with the appropriate units (mm or pix), number of ticks, and scale.

Rationale: ProfileMonitor shall be user-friendly.

Source: Paul Emma, Henrik Loos

Fit Criterion: ProfileMonitor display all of the information above.

Dependencies/Conflicts:

History:

Requirement ID: **GUI.8**

Use Case #: All

Description: Before the last ProfileMonitor window is closed, ProfileMonitor shall pop up a confirmation dialog that asks the user whether he wants to exit ProfileMonitor.

Rationale: ProfileMonitor shall be user-friendly.

Source: Sergei Chevtsov, Mike Zelazny

Fit Criterion: ProfileMonitor pops-up a confirmation dialog before it exits.

Dependencies/Conflicts:

History:

Requirement ID: **GUI.9**

Use Case #: 8, 10

Description: When user rubber-bands to select a new crop area, ProfileMonitor should display the previous crop area as reference.

Rationale: ProfileMonitor shall be user-friendly.

Source: Sergei Chevtsov

Fit Criterion: ProfileMonitor displays the current crop area (if applicable), when user selects a new crop area.

Dependencies/Conflicts:

History:

Requirement ID: **GUI.10**

Use Case #: All

Description: User should be able to change some of ProfileMonitor's look (fonts, colors) via preferences.

Rationale: ProfileMonitor shall be user-friendly.

Source: Sergei Chevtsov

Fit Criterion: ProfileMonitor successfully applies a different set of UI preferences to the running application.

Dependencies/Conflicts:

History:

Requirement ID: **Help.1**

Use Case #: 13

Description: ProfileMonitor shall include comprehensive help: online help, video tutorials, user's/programmer's guides, and JavaDoc.

Rationale: ProfileMonitor shall be user-friendly.

Source: Greg White, Paul Chu.

Fit Criterion: Users read help before asking me questions.

Dependencies/Conflicts:

History:

Requirement ID: ImageAcquisition.1

Use Case #: 3

Description: User shall be able to acquire LCLS images from a selected screen camera via ProfileMonitor.

To enable acquisition, user shall put the screen of the selected camera IN the beam line and all upstream screens OUT of the beam line. Then, user shall specify whether he wants a new or a previously saved background image (one per camera). Next, user shall specify the number of beam images. ProfileMonitor shall put a limit on the number of images that is related to the damage factor and the buffer size of the camera.

ProfileMonitor shall put a uniquely identifiable lock on each acquisition and inform other users of an on-going acquisition.

User shall have the option of downloading acquired images and creating a dataset in local memory. ProfileMonitor shall assign a default name, which user can change, and the name of the origin camera to the dataset.

User shall be able to cancel an acquisition.

Rationale: This is another basic functionality of ProfileMonitor.

Source: Henrik Loos, Paul Emma, Mike Zelazny.

Fit Criterion: User acquires an image dataset.

Dependencies/Conflicts:

History:

Requirement ID: ImageAcquisition.2

Use Case #: 4

Description: ProfileMonitor shall be able to save a number of image datasets (>=1) to an RDB as well as to export them to a compressed .MAT file. The depth of the .MAT structure should be as small as possible.

ProfileMonitor shall provide a default name for user's data; user should be able to customize the pattern for default data names.

ProfileMonitor shall be able to save a newly acquired background image of the selected image dataset to an RDB for future acquisitions.

User shall be able to retrieve his data from an RDB in a reasonable amount of time (about as quickly as from a local hard disk).

Rationale: User needs to analyze image datasets later off-line.

Source: Paul Emma, Henrik Loos, Mike Zelazny, Patrick Krejcik

Fit Criterion: ProfileMonitor successfully saves image datasets.

Dependencies/Conflicts:

History:

Requirement ID: ImageAnalysis.1

Use Case #: 6

Description: User shall be able to browse through image datasets and mark a dataset as "deleted" (note: do not actually delete any data); user shall also be able to select a dataset to browse through its images.

User shall be able to override the algorithmically determined quality of image data.

User shall be able to select an image for a more detailed analysis ("ImageAnalyzer" mode); in that mode, user shall be able to quickly jump to a different image from the same dataset.

Rationale: User needs to analyze acquired images.

Source: Paul Emma, Henrik Loos, Mike Zelazny

Fit Criterion: All of the above works.

Dependencies/Conflicts:

History:

--

Requirement ID: ImageAnalysis.2

Use Case #: 4

Description: User shall be able to save changes in image data (image datasets as well as parameters for and results of image processing) to RDB; some of the actual results of image processing shall be published via EPICS PVs.

ProfileMonitor shall display a confirmation dialog, if RDB data is about to be overwritten.

Rationale: Results of image processing shall be available for analysis by other applications and/or debugging; results of image processing shall be archived.

Source: Mike Zelazny, Henrik Loos

Fit Criterion: ProfileMonitor saves all image-related data to RDV; results of image processing are available via EPICS PVs.

Dependencies/Conflicts:

History:

--

Requirement ID: ImageAnalysis.3

Use Case #: 5

Description: User shall be able to load the image data (image datasets as well as parameters for and results of image processing) from RDB into ProfileMonitor.

Rationale: User needs to analyze off-line data.

Source: Paul Emma

Fit Criterion: User loads the image data into ProfileMonitor.

Dependencies/Conflicts:

History:

--

Requirement ID: ImageAnalysis.4

Use Case #: 5

Description: User should be able to export image data to a Matlab (.mat) file.

Rationale: User needs to analyze image data inside Matlab.

Source: Greg White

Fit Criterion: User exports image data to a Matlab file.

Dependencies/Conflicts:

History:

Requirement ID: **ImageAnalysis.5**

Use Case #: 11

Description: User should be able to manually annotate images with text and/or arrows.

Rationale: ProfileMonitor should be user-friendly..

Source: Greg White

Fit Criterion: Users annotates images as he desires.

Dependencies/Conflicts:

History:

Requirement ID: **ImageProcessing.1**

Use Case #: 11

Description: ProfileMonitor shall be able to subtract the background noise from a beam image.

ProfileMonitor shall be able to subtract a background image from a beam image. ProfileMonitor shall support subtracting background images of different size, with a different origin, or orientation.

Rationale: ProfileMonitor must help with analysis of the beam.

Source: Paul Emma, Henrik Loos, Mike Zelazny

Fit Criterion: ProfileMonitor subtracts either background noise, or a background image from beam images.

Dependencies/Conflicts:

History:

Requirement ID: **ImageProcessing.2**

Use Case #: 11

Description: ProfileMonitor shall be able to display the current beam centroid on a beam image.

ProfileMonitor shall be able to display the previously saved golden orbit/laser beam centroid on a beam image. ProfileMonitor shall draw a centroid as an ellipse with crosshairs.

User shall be able to specify the color of a centroid.

Rationale: ProfileMonitor shall be user-friendly and help with quick analysis of beam images.

Source: Henrik Loos

Fit Criterion: ProfileMonitor draws all of the centroids mentioned above.

Dependencies/Conflicts:

History:

Requirement ID: **ImageProcessing.3**

Use Case #: 11

Description: ProfileMonitor shall be able to apply one of the three color-maps (gray, hot, or jet- see Matlab functions) to an image.

ProfileMonitor shall automatically determine the optimal number of colors. User shall be able to adjust the number of colors.

Rationale: ProfileMonitor must facilitate the analysis of beam images.

Source: Henrik Loos

Fit Criterion: Function works.

Dependencies/Conflicts:

History:

Requirement ID: **ImageProcessing.4**

Use Case #: 11

Description: ProfileMonitor shall be able to automatically crop a beam image, so that only relevant data is analyzed. User shall be able to manually crop an image via rubber-banding, or via halving of the current area..

ProfileMonitor shall be able to store a master crop area for an image dataset. ProfileMonitor shall average the images of a dataset, before user selects a master crop area. User shall be able to adjust the height/width/centroid of the master crop area for each image in the dataset; ProfileMonitor shall display the current crop area during adjustment.

Rationale: ProfileMonitor must be user-friendly and help with image analysis.

Source: Henrik Loos, Paul Emma

Fit Criterion: All of the above works.

Dependencies/Conflicts:

History:

Requirement ID: **ImageProcessing.5**

Use Case #: 11

Description: ProfileMonitor shall be able to apply a floor and, optionally, a median filter to each image.

Rationale: ProfileMonitor shall facilitate the image analysis.

Source: Henrik Loos

Fit Criterion: Applying floor and/or median filter works.

Dependencies/Conflicts:

History:

Requirement ID: **ImageProcessing.6**

Use Case #: 11

Description: ProfileMonitor shall be able to automatically determine the quality of image data. User shall be able to override such flag.

Rationale: ProfileMonitor must facilitate the analysis of beam images.

Source: Henrik Loos, Paul Emma

Fit Criterion: Function works.

Dependencies/Conflicts:

History:

--

Requirement ID: ImageProcessing.7

Use Case #: 11

Description: For each beam image, ProfileMonitor shall display beam lineout and profile fits in each projection (xcoord profX fitX/ ycoord profY fitY).). For lineout fits, user shall be able to specify a point on an image, or select the brightest point as calculated by ProfileMonitor. ProfileMonitor shall draw crosshairs that intersect at the point for which lineout fits are calculated. User shall be able to select one of the 7 fit algorithms by its name (Gaussian, asymmetric, RMS raw, RMS cut peak, RMS cut area, RMS floor, Super-Gaussian).

ProfileMonitor shall display beam parameters as text (xmean, ymean, xrms, yrms, corr, sum, sigma, FWHM, one over e square, 3rd and 4th momenta, correlation coefficient, quality indicator).

ProfileMonitor shall be able to display all data plots in either microns, or pixels.

Rationale: ProfileMonitor must facilitate the analysis of beam images.

Source: Henrik Loos, Paul Emma

Fit Criterion: All of the above works.

Dependencies/Conflicts:

History:

--

Requirement ID: ImageProcessing.8

Use Case #: 11

Description: ProfileMonitor shall be able to calculate beam parameters for both, the full image and a particular slice of it (vertical/horizontal). User shall be able to select whether ProfileMonitor shows beam parameters for the full image or for a particular slice. ProfileMonitor should display the outline of a slice on the image. The width and the color of the outline frame should be configurable.

Rationale: ProfileMonitor must facilitate the analysis of beam images.

Source: Henrik Loos, Paul Emma

Fit Criterion: Slicing of images works as described above.

Dependencies/Conflicts:

History:

--

Requirement ID: ImageProcessing.9

Use Case #: 11

Description: ProfileMonitor should have the option to immediately apply changes of image processing parameters. ProfileMonitor should provide an APPLY button to apply changes of image processing parameters on demand.

Rationale: ProfileMonitor must be user-friendly.

Source: Sergei Chevtsov

Fit Criterion: Immediate APPLY and APPLY button work.

Dependencies/Conflicts:

History:

--

Requirement ID: **LiveImage.1**

Use Case #: 1

Description: User shall be able to select a camera from which to view live images at 1 Hz. User shall be able to start/stop streaming of live images. ProfileMonitor shall display all beam parameters/fits for each live image.

Rationale: ProfileMonitor must be user-friendly.

Source: Sergei Chevtsov

Fit Criterion: Displaying live images and beam fits/parameters works.

Dependencies/Conflicts:

History:

--

Requirement ID: **LiveImage.2**

Use Case #: 2

Description: User should be able to take snapshots of live images for further analysis.

Rationale: ProfileMonitor must be user-friendly.

Source: Sergei Chevtsov

Fit Criterion: Taking snapshots of live images works.

Dependencies/Conflicts:

History:

--

Requirement ID: **Logging.1**

Use Case #: All

Description: ProfileMonitor shall log all messages to cmlog.

Rationale: ProfileMonitor must be easy to debug.

Source: Greg White

Fit Criterion: Logging to cmlog works.

Dependencies/Conflicts:

History:

3.2.2 Data Requirements

The main data elements are:

- Camera configurations (stored in RDB)
- Image datasets (saved to a file)
- Results of image processing (exist in memory only)

4 Non-functional Requirements

4.1 Look and Feel Requirements

4.1.1 Appearance Requirements

ProfileMonitor shall be part of XAL framework.

4.2 Usability and Humanity Requirements

4.2.1 Ease of Use

ProfileMonitor in XAL shall be much more user-friendly than the current Matlab version.

4.3 Performance Requirements

4.3.1 Speed and Latency Requirements

ProfileMonitor shall provide a visual response within 1 second. This shall consist of a "working" message or an hourglass, or a result notification.

4.3.2 Safety-Critical Requirements

ProfileMonitor shall not be used for safety-critical functions. It is assumed that devices are either self-protecting, or protected by the MPS, and that humans in the vicinity are protected by the PPS.

4.3.3 Reliability and Availability Requirements

In general, ProfileMonitor shall be available 24x7.

4.3.4 Robustness or Fault-Tolerance Requirements

ProfileMonitor shall perform accurately. Errors encountered shall be fully and immediately reported.

4.4 Operational and Environmental Requirements

4.4.1 Expected Physical Environment

LCLS Accelerator control room.

4.5 Maintainability and Support Requirements

4.5.1 Maintenance Requirements

- The camera configurations that are stored in an RDB shall be kept up-to-date.

4.6 Security Requirements

4.6.1 Access Requirements

ProfileMonitor needs to set values of some EPICS PVs.

ProfileMonitor shall provide security levels based on user ID.

5 Project Issues

5.1 Off-the-Shelf Solutions

5.1.1 Products That Can Be Copied

An existing implementation in Matlab can be considered a prototype.

5.2 Risks

- Schedule of project vs. schedule of potential developers.
- Concurrent development project dependencies (SEAL/CSS, RDB, AIDA, Multiknob, as relevant)
- User non-acceptance