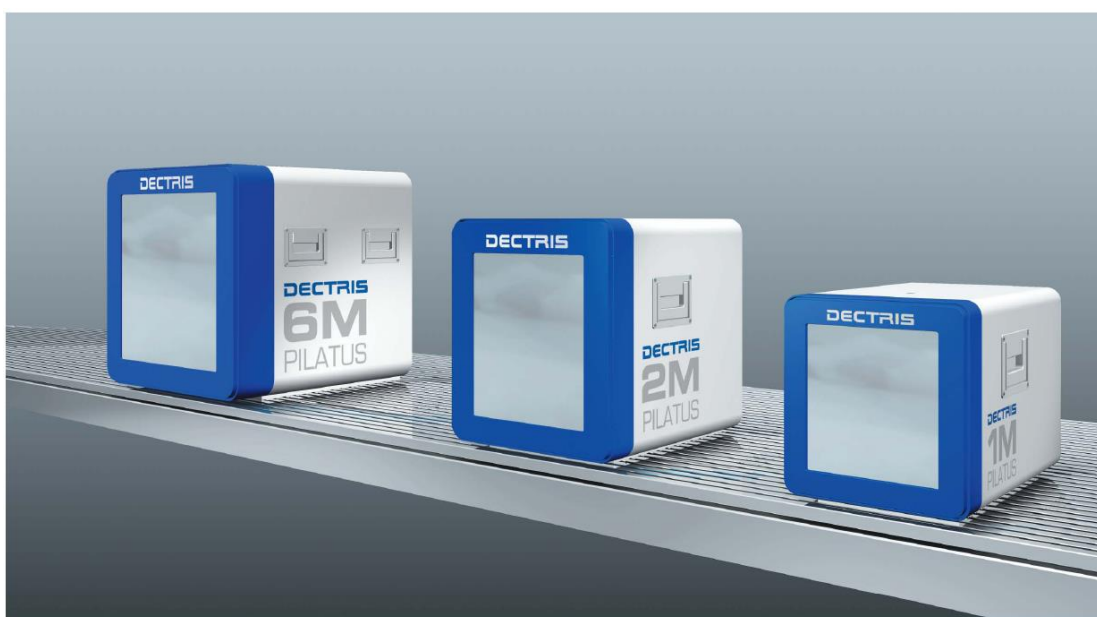
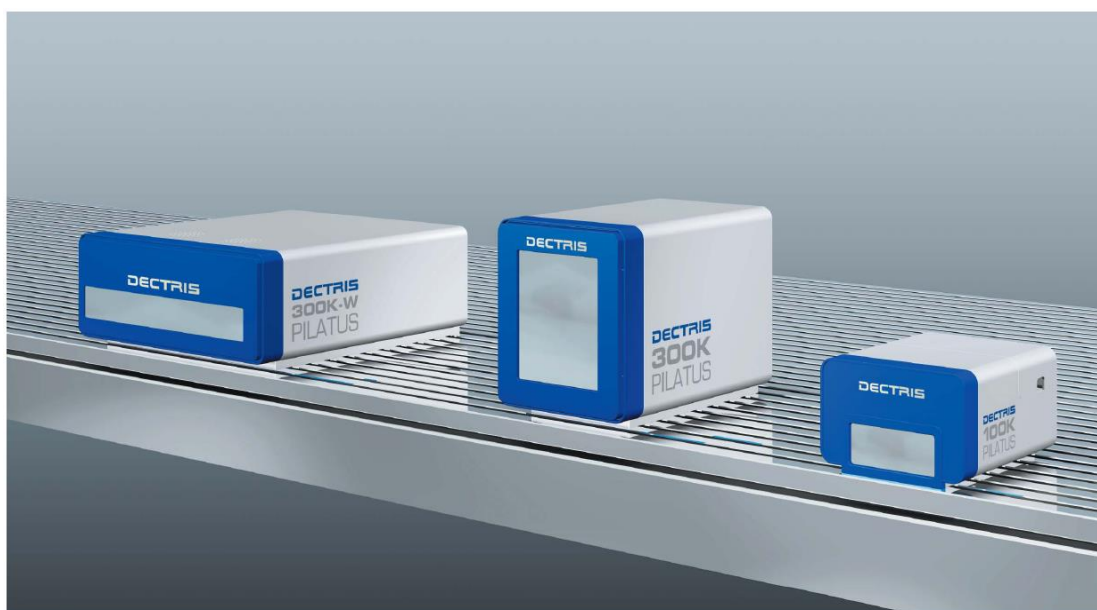


# USER MANUAL

## PILATUS Detector Systems



## Table of Contents

|   |    |
|---|----|
| 1 Document History .....  | 3  |
| 1.1 Changes .....   | 3  |
| 2 How to use this Manual .....  | 4  |
| 2.1 Address and support .....   | 4  |
| 2.2 Explanation of Symbols .....  | 4  |
| 2.3 Convention for Commands .....   | 5  |
| 2.4 Disclaimer .....  | 5  |
| 3 Warnings .....  | 6  |
| 4 System Description .....  | 7  |
| 4.1 Overview .....  | 7  |
| 4.2 Hardware .....  | 7  |
| 4.3 Software .....  | 11 |
| 4.3.1 Overview of TVX .....   | 12 |
| 4.3.2 Overview of Camserver .....   | 12 |
| 4.3.3 Description of the File Structure and Configuration Files on the<br>Pilatus Detector Server ..... | 13 |
| 5 Quick Start Guide .....   | 14 |
| 6 Control the Detector .....  | 16 |
| 6.1 From the Detector Server .....  | 16 |
| 6.2 From a Specific Environment .....   | 16 |
| 6.2.1 Steps to Bring Up a PILATUS Detector in a New Environment ..                                      | 17 |
| 6.2.2 Testclients .....   | 18 |
| 7 How to use the Pilatus Detector through Camserver .....   | 19 |
| 7.1 Main Commands .....   | 19 |
| 7.1.1 Variables .....   | 20 |
| 7.2 Image formats .....   | 20 |
| 7.3 External Triggering .....   | 22 |
| 7.3.1 Necessary Camserver Time Settings .....   | 22 |
| 7.3.2 External Trigger Mode .....   | 23 |
| 7.3.3 External Multi Trigger Mode .....   | 25 |
| 7.3.4 External Enable Mode .....  | 26 |
| 7.3.5 Multiple Exposure Mode .....  | 28 |
| 8 Trimming the Detector .....   | 29 |
| 8.1 Principle .....   | 29 |
| 8.2 Simple Trimming Method .....  | 29 |
| 8.3 Set the Threshold with more Control .....   | 30 |
| 9 Bad Pixel Mask and Module Gaps .....  | 32 |
| 9.1 Using the Bad Pixel Mask .....  | 32 |
| 9.1.1 Adding new Bad Pixels to the Mask .....   | 32 |
| 9.1.2 Make a new Bad Pixel Mask from an Uniform Illumination .....                                      | 33 |
| 9.2 Flag the Module Gaps (Only for Multi Module Detectors) .....  | 33 |
| 10 Adjust Crystallography Parameters .....  | 34 |
| 11 Flat Field Image .....   | 35 |
| 11.1 Using the Flat Field Correction Image in Camserver .....   | 35 |
| 11.2 Creating a new Flat Field Image .....  | 36 |
| 11.3 Using the Flat Field Correction Image in TVX .....   | 36 |

|  |    |
|--|----|
| 12 How to Use the Pilatus Detector through TVX ..... | 37 |
| 12.1 Description of the Image Display .....          | 40 |
| 12.2 Analysis commands .....                         | 44 |
| 12.3 Mask files .....                                | 45 |
| 12.4 User defined commands.....                      | 47 |
| 12.5 Glossary files.....                             | 48 |
| 12.6 Example .....                                   | 48 |
| 13 Factory Calibration and Correction .....          | 49 |
| 14 Camserver Commands .....                          | 51 |
| 15 Camserver Test Client .....                       | 62 |

# 1 Document History

Actual document

| Version | Date       | status   | prepared | checked | released |
|---------|------------|----------|----------|---------|----------|
| 1.4     | 01.05.2013 | released | PS, BS   | MS      | SB       |

## 1.1 Changes

| Version | Date       | Changes              |
|---------|------------|----------------------|
| 1.1     | 31.08.2007 | Various improvements |
| 1.2     | 06.02.2009 | Various improvements |
| 1.2.1   | 28.09.2010 | Various improvements |
| 1.3     | 23.08.2011 | Major revision       |
| 1.4     | 01.05.2013 | Various improvements |

## 2 How to use this Manual

Before you start to operate the PILATUS detector system please read this manual thoroughly.

The user manual and the technical specification together form the user documentation.

### 2.1 Address and support

DECTRIS Ltd.  
Neuenhoferstrasse 107  
5400 Baden  
Switzerland  
Phone: +41 56 500 21 02  
Fax: + 41 56 500 21 01



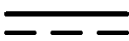


Email: [support@dectris.com](mailto:support@dectris.com)

Should you have questions concerning the system or its use, please contact us via phone, mail or fax.



Do not ship the system Back before you receive the necessary transport and shipping information!

### 2.2 Explanation of Symbols

| Symbol  | Description   |
|---|---|
|  | Important or helpful notice   |
|  | Caution. Please follow the instructions carefully to prevent equipment damage or personal injury. |
|  | DC-current  |
|  | AC-current  |
|  | Ground  |

## 2.3 Convention for Commands

| Example                      | Description  |
|------------------------------|--|
| <i>cd ..</i>                 | Shell commands are written in blue and italic      |
| <i>ExpTime</i>               | Camserver commands are written italic              |
| <b><i>disp</i></b>           | TVX commands are written in bold italic            |
| <b><u><i>exposem</i></u></b> | TVX macros are written in underlined, bold, italic |

## 2.4 Disclaimer

DECTRIS has carefully compiled the contents on this manual according to the current state of knowledge. Damage and warranty claims arising from missing or incorrect data are excluded.

DECTRIS bears no responsibility or liability for damage of any kind, also for indirect or consequential damage resulting from the use of this system.

DECTRIS is the sole owner of all user rights related to the contents of the manual (in particular information, images or materials), unless otherwise indicated. Without the written permission of Dectris it is prohibited to integrate the protected contents published in these applications into other programs or other Web sites or to use them by any other means.

DECTRIS reserves the right, at its own discretion and without liability or prior notice, to modify and/or discontinue this application in whole or in part at any time, and is not obliged to update the contents of the manual.

## 3 Warnings



Please read these warnings before operating the detector

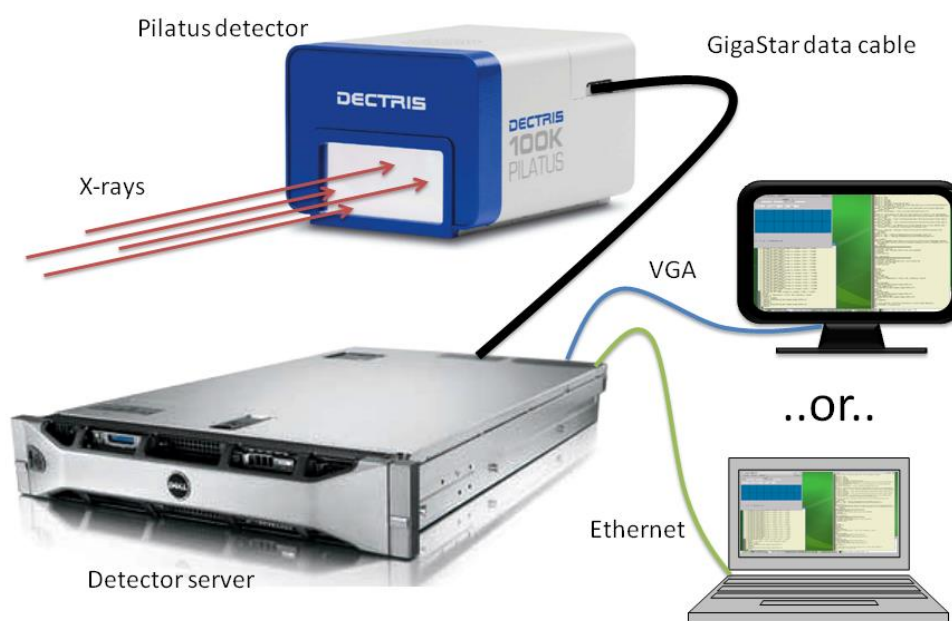
- Before turning the power supply on, check the supply voltage against the label on the power supply. Using an improper main voltage will destroy the power supply and damage the detector.
- Power down the detector system before connecting or disconnecting any cable.
- Make sure the cables are connected and properly secured.
- Avoid pressure or tension on the cables.
- The air inlets and outlets for the detector fan should not be blocked.
- The detector system should have enough space for proper ventilation. Operating the detector outside the specified ambient conditions could damage the system.
- The detector is not specified to withstand direct beam at a synchrotron. Such exposure will damage the exposed pixels.
- Place the protective cover on the detector when it is not in use.
- Opening the detector or the power supply housing without explicit instructions from DECTRIS will void the warranty.
- The Linux operating system, on the DETECTOR SERVER has a customized Kernel to improve data throughput.
- DO NOT UPDATE THE OPERATING SYSTEM OR THE KERNEL
- DO NOT TOUCH THE ENTRANCE WINDOW OF THE DETECTOR

## 4 System Description

### 4.1 Overview

A PILATUS detector system consists of the following components (see also Figure 1):

- Detector
- Detector server with a customized Linux operating system, the data acquisition tool *Camserver* and the data analysis tool *TVX*
- Power supply (where applicable)
- Connecting cables



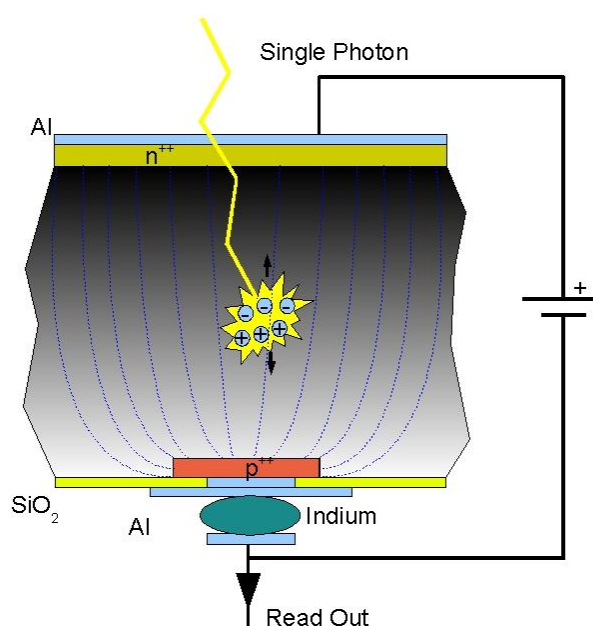
**Figure 1. Overview of the PILATUS detector system setup (Monitor or laptop are not included).**

### 4.2 Hardware

DECTRIS X-ray detector systems operate in "single photon counting" mode and are based on the newly developed CMOS hybrid pixel technology. The main difference with respect to existing detectors is that the X-rays are directly



transformed into electric charge (Figure 2) and processed in the CMOS readout chips. This new design has no dark current or readout noise, a high dynamic range of  $1'048'576$  (20 bits), a read-out time of less than 3 ms, a frame rate of over 300 images/s\* and an excellent point spread function of 1 pixel. The quantum efficiency of the 0.32 mm thick silicon sensor is optimal for experiments in the energy range from 3-12 keV; however the detectors can be used for energies of up to 30 keV or more. The counting rate is greater than  $2 \times 10^6$ /s/pixel, enough to perform many experiments using the high flux of modern synchrotron light sources. However, the detector cannot withstand a direct synchrotron beam.

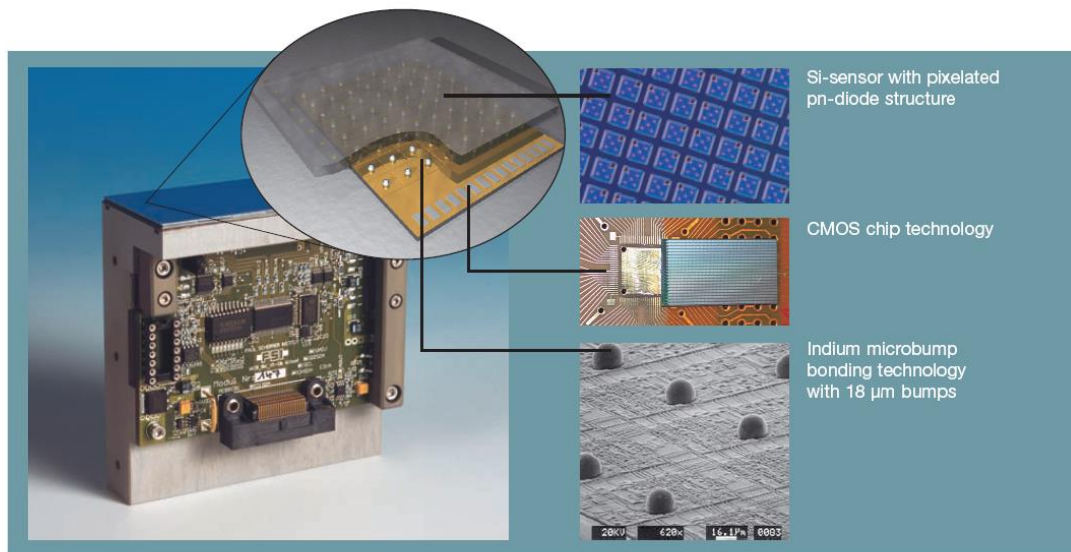


**Figure 2. Principle of direct detection.**

A DECTRIS hybrid pixel detector is composed of a silicon sensor, which is a two-dimensional array of pn-diodes processed in high-resistivity silicon, connected to an array of readout channels designed with advanced CMOS technology (Figure 3). Each readout channel is connected to its corresponding detecting element through a microscopic indium ball, with a typical diameter of  $18 \mu\text{m}$ . This connection process is called 'bump-bonding'.

---

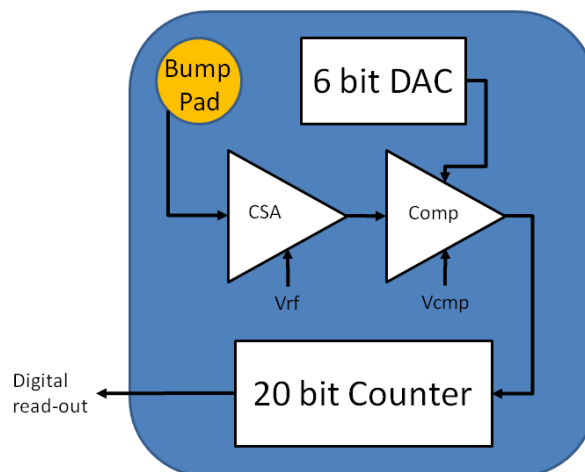
\* This maximum frame rate is only valid for 100k and 300k systems.



**Figure 3. The detector module, the basic element of all DECTRIS area detector systems**

The great advantage of this approach is that standard technologies are used for both the silicon sensor and the CMOS readout chips, which guarantees highest quality. Both processes are optimized separately, as the best silicon substrates for X-ray detection and for high-speed/high-quality electronics are very different. Moreover, the small size of the pixel and the interconnection results in a very low capacitance, which has the beneficial effect of reducing the noise and power consumption of the pixel readout electronics.

X-ray data collection can be improved with detectors operating in single photon counting mode. A hybrid pixel which features single photon counting comprises a charge-sensitive preamplifier (CSA), which amplifies the signal generated in the sensor by the incoming X-ray, and a comparator (Comp), which produces a digital signal if the incoming charge exceeds a pre-defined threshold. The comparator feeds a 20 bit counter, which then leads to completely digital storage and noiseless readout of the number of detected X-rays in each pixel (Figure 4).



**Figure 4. Design of each pixel. See text for details.**

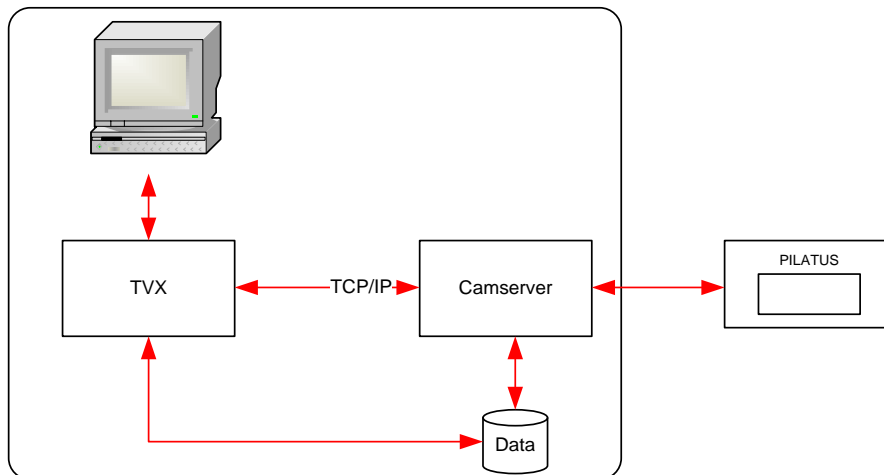
The fundamental unit of the DECTRIS detectors, the module, consists of a single fully depleted monolithic silicon sensor with an 8 x 2 array of CMOS readout chips bump-bonded to it. Each sensor is a continuous array of  $487 \times 197 = 94965$  pixels without dead areas and covers an active area of  $83.8 \times 33.5 \text{ mm}^2$ . The readout chips are wire-bonded to the underlying print which is glued to the mounting bracket. Together with its readout control electronics the sensor with readout chips forms the complete module (Figure 3).

## 4.3 Software

The operating software for the PILATUS detector system consists of two components:

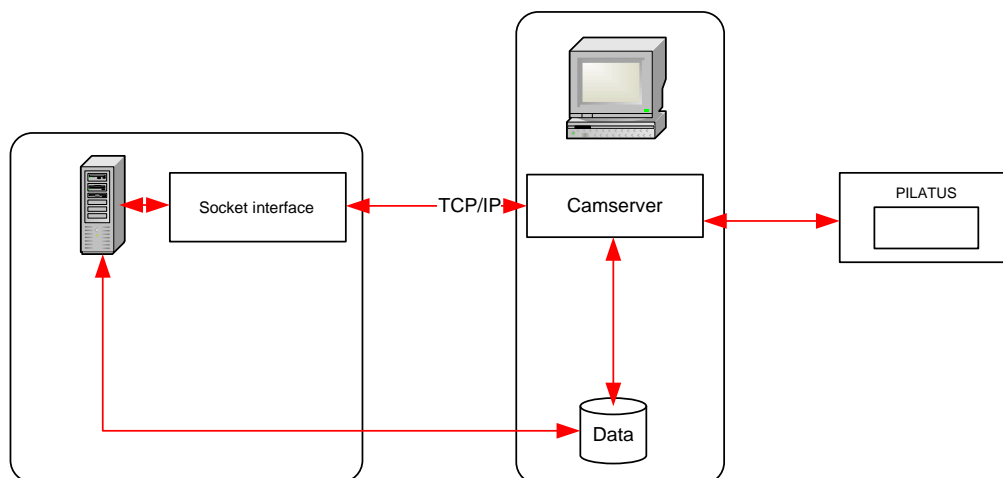
- **TVX**  
Data analysis software and control program
- **Camserver**  
Operating software for the detector hardware

These two software packages are normally installed on one detector server and communicate with each other through an internal socket connection.



**Figure 5. Normal operation with TVX and Camserver on one computer.**

It is also possible to operate the detector without TVX and access Camserver directly via a socket connection from another PC.



**Figure 6. Operation with TVX and Camserver on separate computers.**

### **4.3.1 Overview of TVX**

TVX is a free, open source, data acquisition and control software suite tailored to X-ray science. TVX is an attempt to provide a flexible user interface that is easily adapted to control a broad range of 2-D X-ray detectors as well as a powerful collection of analysis tools.

The suite operates by distributing the tasks of data analysis and hardware control between two separate programs. The first program, which is referred to as TVX, contains the user interface and analysis tool suite. The other, which is referred to as the Camserver, is responsible for controlling the hardware of the specific data acquisition system. These two programs communicate over a TCP/IP connection, as shown in Figure 5, and thus do not need to run on the same machine; see Figure 6.

An added benefit of this model is that it allows the experimenters to do their analysis wherever and whenever it is most convenient for them, be it at the beam line while the data are being taken or back at their home institution or corporation. TVX compiles and operates on Linux. Camserver requires specific camera hardware for operation.

### **4.3.2 Overview of Camserver**

Camserver is a freestanding program that controls the Pilatus detector and provides a simple user interface for "atomic" (single function) commands. It is intended to provide a minimal, but fully functional, low level interface to camera hardware.

On invocation, Camserver takes a single command-line argument, the path to its resource file, by default called 'camrc'. Camserver will also use the same path to open its debugging file, 'camdbg.out'.

A major function of Camserver is to accept socket connections from a high level controller (e.g., 'TVX'), which can provide high level services to this or other cameras. The interface is a simple text-based message passing system. Images - the ultimate product of a working area X-ray detector - do not pass through the socket interface, but are written to a configurable location (e.g., a NFS mount) where any program can access them.

### 4.3.3 Description of the File Structure and Configuration Files on the Pilatus Detector Server

In the default setup, all data necessary for the use of the PILATUS detector system is in the directory /home/det/p2\_det.

| Directory | Sub Directory | Description                                  |
|-----------|---------------|--|
| config    |               | Configuration Data                           |
|           | cam_data      | Definitions of the camera / calibrations     |
|           | calibration   | All calibration details including Mask files |
|           | camstat       | Status of the detector                       |
| images    |               | Default image directory                      |
| programs  |               | Program code for TVX and Camserver           |
| graphs    |               | Default directory for TVX graphs             |
| correct   |               | Masks for statistical analysis in TVX        |
| docs      |               | Documentation of TVX and Camserver           |

Important configuration files are listed in the followings table. The directory is given relative to the default path ~/p2\_det/.

| File        | Directory       | Description  |
|-------------|-----------------|--|
| camrc       | ./              | configuration file for Camserver                     |
| tvxrc       | ./              | configuration file for TVX                           |
| camdbg.out  | ./              | Debug file for Camserver                             |
| p2det.set   | config/cam_data | Startup file for Camserver                           |
| default.gl  | config          | Startup glossary of TVX                              |
| det_spec.gl | config          | Detector specific parameters for TVX                 |
| user.gl     | config          | User shortcuts for TVX                               |
| startup.gl  | config          | Final startup glossary executing detector self-tests |

## 5 Quick Start Guide

Before you turn on the system, make sure you have read this manual, the technical specification, and set up the detector accordingly.

- Turn on the detector server.
- Log in procedure:  
User and password: see additional sheet in your user documentation
- Open a shell.
- Change to the p2\_det directory. All following paths are given relative to this!  
`cd ~/p2_det/`
- Run TVX by typing  
`./runTVX`

After the initialization (up to 30s) you should get the following screen:

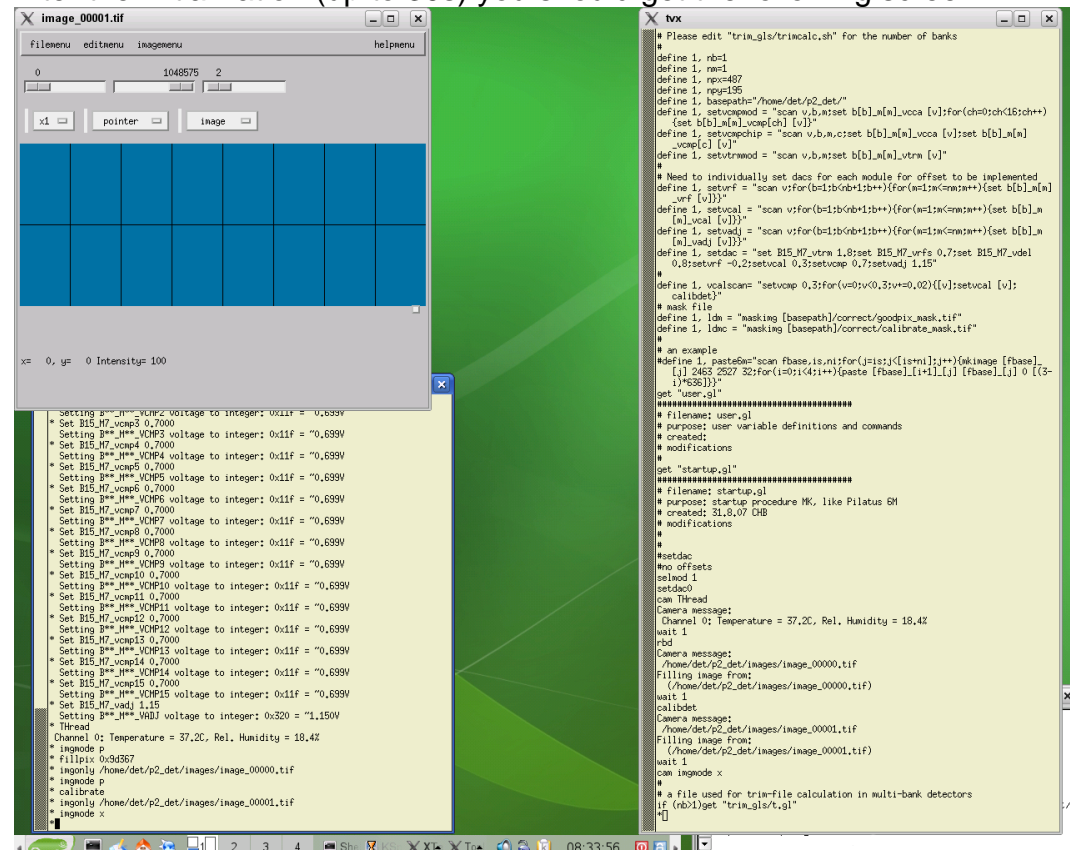


Figure 7. Startup screen after executing `runTVX` (example for a 100k detector)

- The command *runTVX* is a shell script which starts the programs Camserver (yellow window on the left side of figure 9) and TVX (yellow window on the right side of figure 9) as well as manages all log files.

- During startup the detector sets several parameters in the startup scripts (details in 6.1).

Once two images (green and blue) appear on the screen the detector is ready to operate. In Figure 7 only one (blue) is visible since it is above the other (green) one. Both terminals (TVX and Camserver) have their prompt, indicated by an asterisk symbol (\*).

- To operate the detector correctly you

### MUST TRIM THE DETECTOR

according to the incoming X-ray radiation. We refer to trimming as the process of adjusting the settings (voltages and trimbits) of each pixel to the specified gain and energy (for details see 8). This trimming can be done by the *SetEnergy* or the *SetThreshold* command. These commands trim the detector to a uniform threshold across the detector surface. The threshold should generally be set to 50% of the incoming X-ray energy (more details can be found in section 4.2 and 6.1). If you operate the detector every time at the same energy (e.g. an X-ray tube with Cu-anode) you can add such a command at the end of the startup glossary which gets executed during each start up sequence (for details see 6.1).

For example, if you have a 8.048keV incoming beam, type in the TVX window (yellow window on the right side in Figure 7) the following command:

**cam SetEnergy 8048↵**

Details on this command can be found in section 8. For a finer control you can use the *SetThreshold* command (see section 8.3). For more specific questions and detailed advice, please contact DECTRIS Ltd. for support. Note that the *SetEnergy* command takes a relatively long time. On a large multi module system it can take up to 2 minutes.

- The detector is now ready for operation.

- To take an image you can type:

**expose 10↵**

in the TVX window, where 10 is the exposure time in seconds.

- Further information:

Details on how to **control the detector from a specific environment** (e.g. at a synchrotron) can be found in section 6.2.

Details on how to **trigger the detector with an external signal** can be found in section 7.3.

For information concerning the **dead pixels and gaps** please see section 9.

For **image manipulation with TVX** please see section 12.



## 6 Control the Detector

### 6.1 From the Detector Server

The Pilatus detector can be controlled from the delivered detector server. Just follow the instructions in the Quick Start Guide (Section 5) to start up the detector.

The `runTVX` command is a shell script which starts the actual programs: Camserver and TVX. It also handles the log-files which can be found in the `p2_det` directory

During the Camserver startup a connection to the detector is established and verified.

The TVX startup is carried out via the file called `default.gl`. This is a so called glossary file (\*.gl), which is a convention used for files which are processed by TVX.

This `default.gl` file (stored in `"$HOME/p2_det/config"`) makes several definitions (short-cuts) which can be used further on in TVX.

For example the `exposem` command makes an endless loop and writes images in a temporary file (see also section 12). This is a simple live-view tool of TVX.

At the end of the startup glossary three more glossaries are called. The first one (`det_spec.gl`) loads detector specific parameters such as camera size and number of modules. The second one (`user.gl`) is used to enter user short-cuts once you start to define your own ones. The last one (`startup.gl`) sets voltages on the actual X-ray sensitive elements and carries out a test of the digital part and the analog part of each pixel. The last two tests are done with the TVX definitions `rbd` (read back detector) and `calibdet`, respectively. The results are shown in a green (digital test with 1000 counts loaded into the counter of each pixel) and a blue (analog test with 100 simulated pulses fed into each pixel) image. These two images show up automatically at the end of the startup procedure and verify the full functionality of the detector.

### 6.2 From a Specific Environment

In the previous section the stand alone operation is shown. However, often there is a need to integrate the detector in an existing environment.

The Pilatus detector can be easily integrated into any system. To do this, one has to send commands through a socket connection to Camserver. Any client can connect to Camserver via a socket connection, and issue plain text commands. However, only the first connection will get full control and can execute commands. All following connections will only have read access. The command syntax (see section 7) over the socket is identical to the syntax to

be typed directly in the Camserver window. Thus, direct typing is helpful for testing.

The reply from Camserver (acknowledgement) consists of a command index number, followed by a space and either "OK" or "ERR", followed by another space and possibly a message. The acknowledgement arrives after the requested action is completed, typically in 1-2 ms; some commands, such as *SetThreshold*, take longer, especially for a big detector. All acknowledgements end in 0x18 (ASCII 'CAN') without a newline; there may be internal newlines in long messages. (Since there is no terminating linefeed, MS Windows sockets must be opened in binary mode; this is not a consideration for UNIX-like systems.)

Because of the socket connection protocol, the camera hardware and server can reside on a different machine from the high level controller.

Camserver implements a token mechanism to prevent more than one outside process from having control over the hardware. The Camserver window has full control at all times.

There is a debug facility to help with setting up the interface. If you type "*dbg/vl 5*", the file 'camdbg.out' will contain many messages, including the exact contents of socket messages. Be sure to set "*dbg/vl 1*" (the default) before doing real work, else 'camdbg.out' can grow without limit. If there are difficult problems with the detector, a run with "*dbg/vl 6*" reproducing the error can be helpful for diagnosis. Simply capture 'camdbg.out' and send it to DECTRIS.

The Camserver program of the PILATUS detector provides a simple to use interface for either EPICS or SPEC. Several clients for these protocols have been written at the Swiss Light Source (SLS) at the Paul Scherrer Institut (PSI) and by Mark Rivers of the University of Chicago: <http://cars9.uchicago.edu/software/epics/pilatusROIDoc.html>

### **6.2.1 Steps to Bring Up a PILATUS Detector in a New Environment**

- 1) If needed, change the hostname to be compatible with the local network. This can be done conveniently in SuSE linux with YAST2, or directly with vi.
- 2) Set up the detector on the network, if needed. Note that the detector does not require an external network.
- 3) Configure Camserver, and the client TVX, as needed. Probably the defaults will be adequate, but many parameters can be adjusted in "camrc", and "tvxrc", both of which reside in the \$HOME/p2\_det directory.
- 4) Start the detector by *runTVX* in the \$HOME/p2\_det directory.

5) If you are using your own client (see section 6.2.2) for Camserver (e.g., SPEC or EPICS) disconnect TVX (type “**disconnect**” in TVX). This can be done automatically in the startup script after the test images are shown. Connect your client and begin issuing commands (see section 1).

Further sources of information can be found in \$HOME/p2\_det/programs/tvx/docs/.

To see the details of the current configuration of Camserver check the hardware version file in \$HOME/p2\_det/config/camstat/HWVersions.

## 6.2.2 Testclients

The detector can be controlled from any client which opens a socket connection to Camserver.



Make sure TVX is disconnected (type “**disconnect**” in TVX) before you connect your own client.

The most simple client perhaps is *telnet*. It is possible to test it on the detector server itself by executing “*telnet localhost 41234*”. Here localhost is the “IP” of the detector and 41234 the port where *Camserver* is listening to.

A basic test client written in C can be found under section15.

Another test client, called “camclntt” in “p2\_det/programs/tvx/camera” can also be used to issue commands to Camserver.

For more information please contact Dectris at [support@dectris.com](mailto:support@dectris.com).

## 7 How to use the Pilatus Detector through Camserver

Camserver is a completely freestanding program that controls the detector and provides a simple user interface for "atomic" (single function) commands. It is intended to provide a minimal, but fully functional, low level interface to camera hardware.

To get help on the Camserver commands use the help facility of TVX (see section 12).

All commands in Camserver (unlike TVX) can be abbreviated to the minimum number of letters that make the command unambiguous; below we use only the full names for clarity. As in TVX, commands are case-insensitive, but pathnames are case-sensitive. A full list of commands can be found in section 1.



We recommend that full command names are used for clarity.

### 7.1 Main Commands

| Command                     | Description   |
|-----------------------------|---|
| <i>ExpTime</i>              | Set the exposure time ( $10^{-6}$ to $10^6$ sec).   |
| <i>Exposure [filename]</i>  | Make an exposure with the exposure time and exposure period predefined with the command <i>ExpTime</i> and <i>ExpPeriod</i> respectively. The format of the file is determined from the supplied extension. The file is stored relative to the path defined by the <i>imgpath</i> unless an absolute path is given. |
| <i>ExtTrigger [fname]</i>   | Start exposure with defined variables after receiving an external trigger and store images [fname].   |
| <i>ExtMTrigger [fname]</i>  | Start multiple exposures with defined variables after receiving multiple external triggers and store images [fname] (see section 7.3.3).  |
| <i>ExtEnable [fname]</i>    | Start exposure defined by external signal and store images in [fname].  |
| <i>help exposure naming</i> | Type this in the TVX window for a discussion of how exposure series are named.  |
| <i>dcb_init</i>             | Re-initialize the detector control board.   |

### 7.1.1 Variables

The following variables can be viewed just by typing them; all times are in seconds.

| Variable                | Description  |
|-------------------------|--|
| <i>NImages [N]</i>      | Query or set the number of images in a sequence.   |
| <i>ExpTime [time]</i>   | Query or set the exposure time.  |
| <i>ExpPeriod [time]</i> | Query or set the exposure period for serial exposures. The exposure period must be at least 2.3 ms longer than the exposure time.  |
| <i>imgpath [path]</i>   | Query or set the default imgpath.  |
| <i>Delay [time]</i>     | Query or set the external trigger delay. This is the time to wait after the external trigger before taking the first image. The delay may not be greater than the exposure period.   |
| <i>nexpframe [N]</i>    | Number of exposures per frame<br>This is a so called multi exposure mode. nexpframe sets the number of exposures before the detector is read out e.g. “ <i>nexpframe 3</i> ” exposes the detector 3 times before reading out an image of the 3 combined exposures. See point 7.3.5 for more details. |

The usual way is to set all mentioned variables and then execute a command from the section above.

## 7.2 Image formats

Due to the high dynamic range of 20 bits (1'048'576) of the PILATUS detectors, images are stored as 32 bit (signed) integers. These images can be viewed and analyzed with TVX or other image viewers. Many viewers do not support 32 bit TIFF files; however, these images may be read in IDL or MATLAB.

The default image file-type for TVX is set in tvxrc; however, any file-type can be specified explicitly. Camserver has no default, so the file type must be specified explicitly for each exposure.

TVX supports the following image formats:

| Format | Description                    |
|--------|--------------------------------|
| .tif   | 32 bit TIFF files              |
| .edf   | ESRF data format               |
| .cbf   | Crystallographic binary format |

|                            |                 |
|----------------------------|-----------------|
| .img                       | raw data format |
| no extension or misspelled | raw data format |

Of these four image formats .tif, .edf, and .img are uncompressed and .cbf is the only (lossless) compressed format. The .cbf and .edf headers are pure text and therefore human-readable. The .tif headers are encoded, so only the comment section is human-readable.

- The .img file contains only the uncompressed data.
- The .edf file starts with the header according to the ESRF data format followed by the uncompressed data.
- The .tif file contains the standard TIF header including a Pilatus section followed by the uncompressed data. The Pilatus section contains at least the following items (example):
  - # Silicon sensor, thickness 0.000320 m
  - # Exposure\_time 1.0000000 s
  - # Exposure\_period 1.00230000 s
  - # Tau = 199.1e-09 s
  - # Count\_cutoff 1280469 counts
  - # Threshold\_setting: 5900 eV
  - # Gain\_setting: mid gain (vrf = -0.200)
  - # N\_excluded\_pixels = 4
  - # Excluded\_pixels: badpix\_mask.tif
  - # Flat\_field: (nil)
  - # Trim\_file: p100k0273\_T5900\_vrf\_m0p20.bin
  - # Image\_path: /home/det/p2\_det/images/

This list can be extended with several items described in section 10. The TIF header is of a fixed length of 4096 bytes, so it is possible to make a file reader by stripping off the header and reading the row-major block of data.

- The .cbf format is the only compressed image format and is recommended in situations where I/O bandwidth may be limited. The byte-offset-compression algorithm usually reduces the file size to a quarter of the .tif image. The initial header data (this time a .cbf header) is followed by a Pilatus section. Exactly as in the .tif format, this section can be extended with additional items (see section 10). Just before the actual compressed data there is additional information for the .cbf format. It should be mentioned that the full cbf header can be achieved through the commands described in section 10. More details to the .cbf format can be found <http://www.bernstein-plus-sons.com/software/CBF/>.

## 7.3 External Triggering

External triggering can be separated into three different modes:

- External Trigger: triggers a predefined series of commands after the detector receives a positive edge.
- External Multitrigger: triggers each exposure with an external pulse, but times the exposures using the internal timer.
- External Enable: gates the detector's images on the positive signal applied to the external enable input of the detector.

### 7.3.1 Necessary Camserver Time Settings



All exposure commands need to have set the *ExpPeriod*, *ExpTime*, and *NImages* variables. In some circumstances the times given can be approximate.

These values are used by the control program to determine the mode of operation of the system. If the timing parameters are sufficiently relaxed, the system uses a "polling" interface to watch for changes in state of the external enable signal and then responds appropriately. This requires that the exposure time must be  $>33.9$  ms ( $\sim 30$  Hz), and the "dwell" time between exposures (*ExpPeriod* minus *ExpTime*) must be  $>13$  ms. This mode of operation is especially well suited for long exposure times, but with adequate time between exposures.

If the exposure time or the read-out time (exposure period minus exposure time) are short, the system must predict the end of the exposure and the beginning of the next exposure with some accuracy; the requirement for accuracy becomes more stringent as the exposure time becomes longer while the time between exposures is held short.

In most circumstances, there is at least a  $\pm 30\%$  tolerance on the timing parameters input to Camserver as compared to the time actually generated by the external pulse generator. However, bigger differences can cause an error. This is especially the case if the delays add up to more than the timeout of the communication protocol (DMA timeout, displayed during startup of Camserver; either 8 or 16 seconds).

Frame Rate  $< 30$  Hz (only in TVX 7.2.70-110214 or later)

If the exposure period is more than 33.9 ms (less than  $\sim 30$  Hz) AND the read-out time is more than 13 ms it is possible to have large variations in the times which are set in Camserver and the applied pulses on the trigger input of the camera. Moreover, there is no timeout after executing a trigger command in Camserver: the detector will wait until a signal arrives. This state can only be

interrupted by transmitting a 'K' to Camserver over the socket connection. The exposure time and exposure period still have to be set to the approximate trigger structure.

If the exposure time is longer than the DMA timeout (see above) the polling method is enforced. This causes that the read-out time has to be set to more than 13 ms for the external enable and external multi trigger mode.

### 7.3.2 External Trigger Mode

The external trigger mode is exactly the same as an exposure except that an external pulse is used rather than the enter key on the keyboard.

External trigger mode is activated with the command "*ExtTrigger imagename.tif*" where *imagename.tif* is the name of the images you wish to be taken. The first image name in a series will be "imagename\_00000.tif" unless otherwise specified. If *NImages* > 1, the image number will be incremented for each image in the series (see footnote in section 1 for details).



The settings that are necessary for external triggering are:

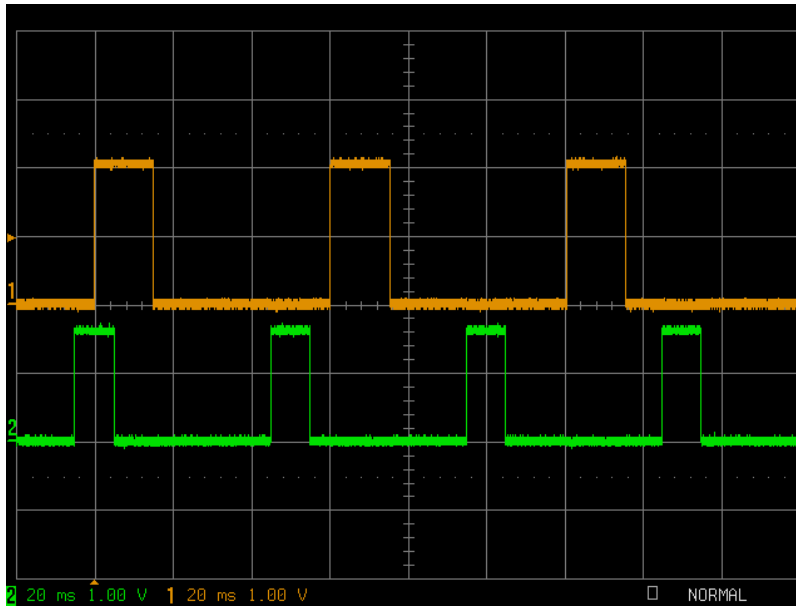
- *NImages*
- *ExpTime*
- *ExpPeriod*
- *Delay* (optional)

After receiving a trigger on the positive edge, the detector will wait a period of time defined by "Delay", take an exposure of length "*ExpTime*", readout the image and after a period defined by "*ExpPeriod*" will repeat the cycle for "*NImages*" images.



The image number is only incremented during the exposure series; if you reissue the command "*ExtTrigger imagename.tif*" the system will start writing images from "imagename\_00000.tif" and overwrite existing data.

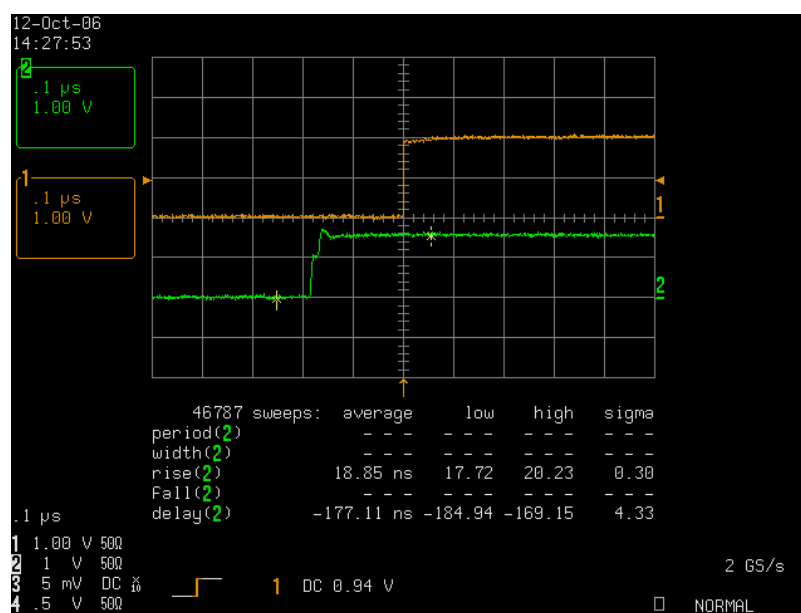




**Figure 8. Oscilloscope trace of an external trigger. Orange: enable signal of the detector; Green: trigger.**

In Figure 8 the upper trace is the exposure (enable) signal, the lower trace is from the pulse generator being used as a trigger. For this external trigger, “*NImages*” is 3, the “*Delay*” is 0.005 s, “*ExpTime*” is 0.016 s and “*ExpPeriod*” is 0.06 s. Note that only the first positive edge of the trigger is used in this example.

Because the external trigger relies upon the module’s internal clock signal to start the timing of the exposure, there is a delay and jitter between the trigger signal and the start of the first exposure. The maximum jitter is ~15 ns with an average delay of 177 ns (see Figure 9).



**Figure 9. Delay and jitter. Orange: enable signal of the detector; Green: trigger.**

### 7.3.3 External Multi Trigger Mode

External multi trigger mode is started with the command “*ExtMTrigger imagedname.tif*” where *imagedname.tif* is the name of the images you wish to be taken. The image name will be *imagedname\_00000.tif* unless otherwise specified. If *NImages* >1 the image number will be incremented for each image in the series.

After issuing the “*ExtMTrigger imagedname.tif*” command the detector will monitor and take a number of images defined below, on the level of the trigger pulse.



The settings that are necessary for external multi triggering are:

- *NImages*
- *ExpTime*
- *ExpPeriod*
- *Delay* (optional)

After receiving a trigger on the positive edge, the detector will wait a period of time defined by “*Delay*”, take an exposure as defined by “*ExpTime*”, readout the image and will rearm to take another images. This will be repeated “*NImages*” times.



The image number is only incremented during an exposure series; if you reissue the command “*ExtTrigger imagedname.tif*” it will start writing images from “*imagedname\_00000.tif*” and overwrite existing data.

### 7.3.4 External Enable Mode

External enable mode is started with the command “*ExtEnable imagename.tif*” where *imagename.tif* is the name of the images you wish to be taken. The first image name will be *imagename\_00000.tif* unless otherwise specified. If *NImages* > 1 the image number will be incremented for each image in the series.

After issuing the “*ExtEnable imagename.tif*” command the detector will monitor and take a number of images defined by *NImages* gated on the level of the trigger pulse.

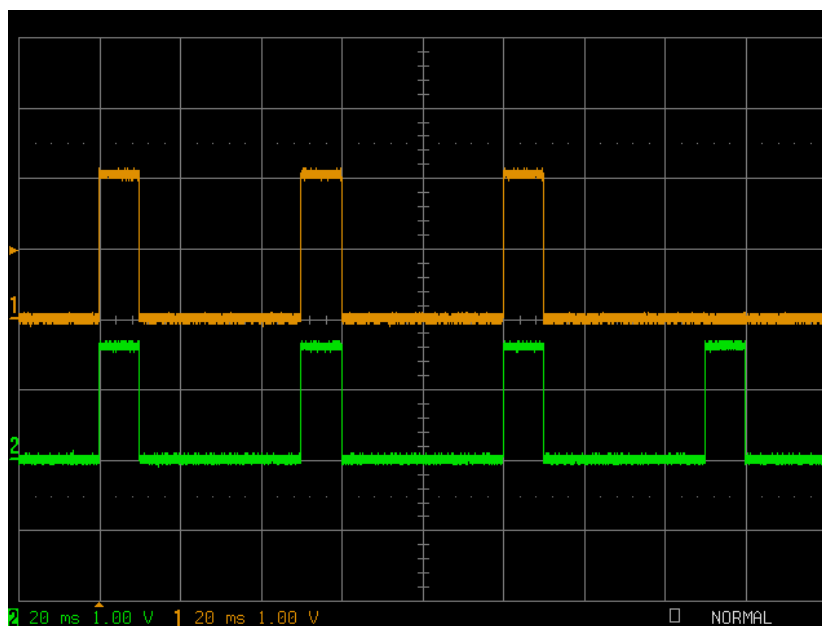


The settings that are necessary for external enable mode are:

- *NImages*
- *ExpTime*
- *ExpPeriod*



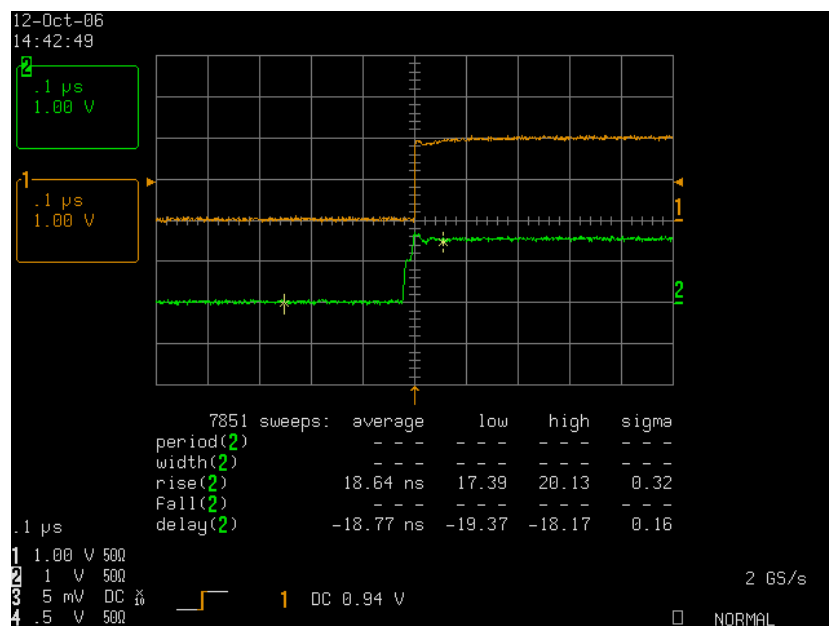
The image number is only incremented during an exposure series; if you reissue the command “*ExtEnable imagename.tif*” it will start writing images from “*imagename\_00000.tif*” and overwrite existing data.



**Figure 10.** Oscilloscope image of an external enable. Orange: enable signal of the detector; Green: external gate.

In this example using external enable “*NImages*” was set to 3.

Because external enable gates the counter directly, it does not rely upon the detector’s internal clock. This means that the Delay between the enable and start of exposure is negligible and mostly given by the rise time of the enable provided to the detector. This can be seen in the oscilloscope image below.



**Figure 11. Oscilloscope trace of the typical delay between enable signal and exposure. Orange: enable signal of the detector; Green: trigger.**

### 7.3.5 Multiple Exposure Mode

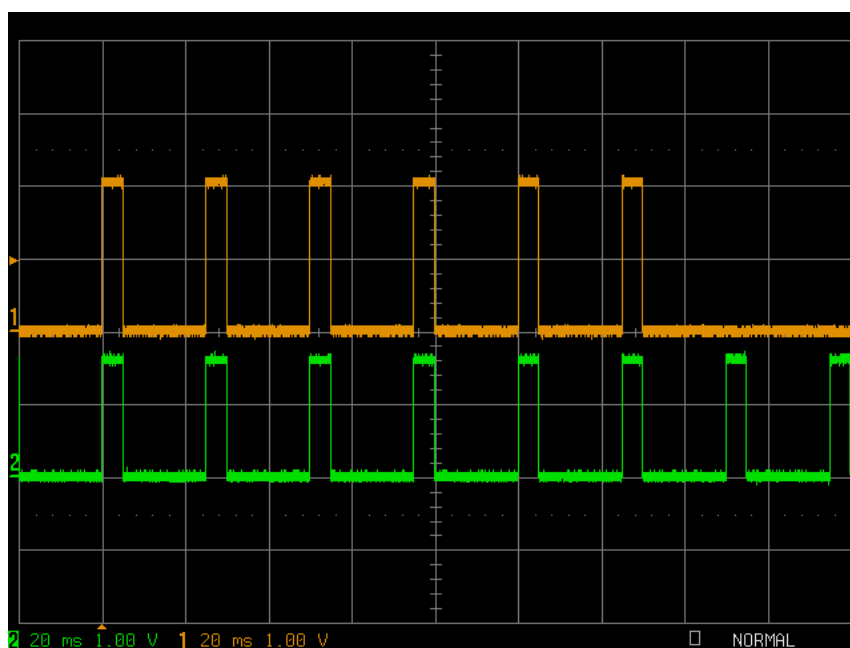


This mode is useful to capture data from a rapidly repeating event which generates only a few X-rays per pixel for each event, such as pump-probe experiments. For example, it is possible to synchronize to the bunch structure of a storage ring providing that an appropriate gate is available from the ring control system. The data are accumulated in the pixel and read out after a certain number, e.g. 250'000, of events is collected.

To use this mode, set the variable *nexpframe* to the desired value. The default value is 1; all exposure modes use this variable.

If the detector is used in this mode and it is synchronized to the bunch structure of a storage ring (high count rate), the rate correction is probably invalid and it is advisable to turn it off (see *Tau* in section Camserver Commands).

In the following example '*nexpframe 2*' is set. The detector will take exposures in the same way as described for external enable (also possible with external trigger and external multi trigger), but will additively bundle 2 exposures in each readout. If *NImages* is defined to be 3 and *nexpframe* is defined to be 2, then the detector will take 6 exposures and generate 3 images. It is necessary to provide 6 pulses or positive edges to achieve a successful readout (only a single pulse is required for the *ExtTrigger* mode).



**Figure 12.** Oscilloscope image of the multiple exposure mode. Orange: enable signal of the detector; Green: external gate.

## 8 Trimming the Detector

### 8.1 Principle

PILATUS detectors possess an adjustable energy threshold which has to be set due to the working principle of the detector. This threshold is controlled by the comparator voltage of the detector chip. Furthermore, the threshold of every pixel can be individually trimmed with six trim bits (6-bit DACs) which allow  $2^6 = 64$  different values. The magnitude of the influence of these trim bits is determined by the trim voltage of each chip. Since the process of threshold trimming is rather complex it is done during factory calibration. To be able to operate the detector appropriately, the determined trim-parameters have to be applied with one of the methods described below.



Every detector is calibrated at the factory.

### 8.2 Simple Trimming Method

The easiest method to trim the detector is to use the `SetEnergy` command. Use the command `SetEnergy` and supply the information of the incoming X-ray energy (in electron volts) as an argument. This will condition/trim the detector in most appropriate way for your measurement.

| Command                         | Description  |
|---------------------------------|--|
| <code>SetEnergy [energy]</code> | <p><code>SetEnergy</code> - - set or query the simplified gain and threshold setting</p> <p>Usage: <code>SetEnergy [energy in eV]</code></p> <p>If energy is out of range, a default is set and a warning issued. The requested energy is used to calculate appropriate gain and threshold settings for the detector. This command internally uses the <code>SetThreshold</code> command to set the threshold. For finer control over detector settings, use <code>SetThreshold</code></p> |

This command allows you to operate the detector in the best manner after issuing only one command.

## 8.3 Set the Threshold with more Control

In order to have more control over the threshold the *SetThreshold* command can be used.

**Background:** The detector systems require that an energy threshold is set. This threshold is usually set to 50% of the incoming X-ray energy for reasons which are explained below. However, it is also possible to set the threshold at an arbitrary energy.

**What happens:** All X-ray photons with an energy higher than the threshold energy will be counted by the detector whereas photons with an energy below this threshold will be cut off and therefore not counted. This transition is not perfectly sharp and follows an s-shaped curve with a derivative of about 1keV (FWHM). This means that if you set the threshold energy at exactly the incoming (monochromatic) energy you will count ~50% of the X-rays. At higher energies the asymptote of this s-shape is not a constant but still slightly increasing due to various reasons.

As mentioned before, the optimum of the threshold value is at 50% of the incoming energy. This arises from the fact that it is possible that an X-ray will be absorbed at the boundary between two pixels and the charge is divided to both of them. If the threshold energy is set below 50% of the incoming X-ray energy it is possible that one photon is counted twice in two neighboring pixels.

**Best usage:** For the mentioned reasons it is best to set the threshold energy for the Pilatus detector between 50% and 80% of the incoming X-ray energy. The upper limit of 80% is not a hard criterion but above that the energy resolution gets worse. Moreover, one should not set the threshold much closer than 1keV to the incoming X-ray energy due to the suppression of the primary incoming energy.

However, it is not a problem to set the threshold e.g. to 60% of the incoming energy. At this threshold there are only about 3-10% (depending on the energy) fewer counts as compared to a threshold of 50%. This decrease is continuous and almost linear until the threshold is ~80% of the incoming energy.

**The gain:** The adjustment of the threshold is limited by the detector electronics. To be able to span a threshold range from 4 keV to 18 keV it is necessary to change the gain, i.e. the amplification level of the detected signal. There are three gains (high, mid and low-gain) implemented which all have a certain threshold range. These ranges are overlapping for large energy ranges. However, lower gain settings should be preferred since they allow higher count rates and better count rate correction. This is especially true if the count rate exceeds several  $10^5$  counts/s/pixel. The previously mentioned *SetEnergy* command (8.2) automatically calculates the threshold to 50% of the supplied incoming X-ray energy and chooses the lowest gain for the desired threshold energy. However, for certain experiments it may be desired to stay at the same gain or just vary the threshold. Then it is useful to use the *SetThreshold* command as described below.

| Command                            | Description  |
|------------------------------------|--|
| <i>SetThreshold gain threshold</i> | SetThreshold - set gain (energy range) and threshold energy<br><br>Usage:<br><i>SetThreshold</i> [[gain] threshold]<br>1) if parameters are omitted, the current settings are shown<br>2) gain is 'highG', 'midG' or 'lowG'<br>3) if gain is omitted, the previous setting is retained<br>4) threshold is in eV<br>5) this command builds a script in "/tmp/setthreshold.cmd" and then loads it. |

The following table shows more details on the three different gains: The optimum energy range for their application, the corresponding amplifier voltage (Vrf) and the time structure used for rate correction (Tau). Details on the latter can be found in section 1.

| gain  | Vrf [V] | Description   | Tau [ns] |
|-------|---------|---|----------|
| highg | -0.15   | Slow setting with high gain for best resolution at low energies 4 – 5 keV                   | 384      |
| midg  | -0.2    | Standard setting for energies between 5 – 7 keV   | 200      |
| lowg  | -0.3    | Fast setting with low gain for best performance at high energies and high rates: 7 – 18 keV | 125      |

The directories with the trim files are stored in:  
 /home/det/p2\_det/config/calibration.

The information about the relevant trimfiles is stored in  
 /home/det/p2\_det/config/cam\_data/calibrations.def  
 In this file, the names of the trimfiles and the corresponding parameters are stored.



## 9 Bad Pixel Mask and Module Gaps

During factory calibration the bad pixels (non-responding or noisy) are determined and stored in a tif image. This image contains 0 for good pixels and 1 for bad pixels. Once the bad pixel mask is loaded, bad pixels are flagged with -2 in the final image. This can be desired or not since once the pixels are flagged, their data are lost. It is also easy to incorporate these data as a post-acquisition step.

### 9.1 Using the Bad Pixel Mask



The bad pixel mask created during factory tests is automatically applied to each image that you take. The bad pix mask is loaded every time the detector is trimmed via the *SetThreshold* command (or any other command which calls this command).

To NOT apply this bad pixel mask automatically, please insert a “#” sign in front of the lines with a *bpmap* in the file */home/det/p2\_det/config/cam\_data/calibrations.def* (usually 9 times).

To remove this function temporally you can also use the command *LdBadPixMap* with a 0 as argument. For further details please see Camserver command “*LdBadPixMap*” in section 1.

#### 9.1.1 Adding new Bad Pixels to the Mask

It is possible that one or more noisy pixels can appear over time. To add such a bad pixel to the mask you have to modify the actual (bad pixel) mask and put at the corresponding x y position a one. This can be easily done with TVX.

To prevent loss of information make a backup of the old mask. In a shell simply type (as one line) where DDMMYY is the actual date:

```
cp /home/det/p2_det/config/calibration/Mask/badpix_mask.tif  
/home/det/p2_det/config/calibration/Mask/badpix_mask_untilDDMMYY.tif
```

Then display the bad pixel mask with TVX:

In TVX: **disp** /home/det/p2\_det/config/calibration/Mask/badpix\_mask.tif

Add the bad pixels, e.g. at the coordinates x: 17, y: 126 by the following command:

In TVX: **pixfill** 1 17 126 17 126

The coordinates are given twice because it is also possible to define a box which will be filled with the first value after the **pixfill** command.

### 9.1.2 Make a new Bad Pixel Mask from an Uniform Illumination

If you expose the detector with a flat field you can use this image to create a new mask with TVX by defining upper and lower limits. Pixels in the detector that are either dead, too noisy or behave in a non-desirable manner can be masked out.

If you have now an image called e.g.: `img_01.tif` with an average count (see `boxall` in 12.2 for the statistical analysis) of e.g.: 1000 counts in the image directory `~/p2_det/images` you can use the following command in TVX to get a mask image:

In TVX: **`mkmask`** `img_01.tif` `goodpixel_mask.tif` 600 1400

This will give you a file called `goodpixel_mask.tif` with 0 where the pixel values are outside the mentioned boundaries of 60% and 140% of the mean value of the provided image, `img_01.tif`. To obtain the appropriate format for Camserver please invert the image:

In TVX: **`move`** `badpixel_mask.tif=1-goodpixel_mask.tif`

This created `badpixel_mask.tif` should then be copied to `~/config/calibration/Mask`. Please make a backup copy of the old mask.

## 9.2 Flag the Module Gaps (Only for Multi Module Detectors)

On multi module detectors the gaps between the modules can be filled either with 0 or with -1. Which number will be filled in is controlled by the Camserver command called *GapFill* (see Camserver command at section 1).

During startup with the `runTVX` script *GapFill* is set to -1. This causes the gaps to be flagged with -1 in the final image. To change the default behavior during startup please remove the line which contains *Gapfill* command from the startup file (`/home/det/p2_det/config/startup.gf`).



At TVX/Camserver versions before TVX-7.2.70-110214 the `SetThreshold` command resets the *Gapfill* value to 0. From TVX-7.2.70-110214 on the previous state is remembered.

## 10 Adjust Crystallography Parameters

The *MxSettings* command allows the user to store more information (as comments or in the CBF template) inside the image header of TIF and CBF images. The principal usage of the *MxSettings* command is the following:

```
mxsettings [parm_name value] [parm_name value] ...
```

It can be used to enter one or more of the following parameters with its value:

Available parameter names:

Wavelength; Energy\_range; Detector\_distance; Detector\_Voffset; Beam\_xy;  
Beam\_x; Beam\_y; Flux; Filter\_transmission; Start\_angle; Angle\_increment;  
Detector\_2theta; Polarization; Alpha; Kappa; Phi; Phi\_increment; Chi;  
Chi\_increment; Oscillation\_axis; N\_oscillations; Start\_position;  
Position\_increment; Shutter\_time;  
and CBF\_template\_file.

It is possible to specify more than one parameter in each command and they can be in any order. The parameters 'Beam\_xy' and 'Energy\_range' accept 2 values. The parameter 'Oscillation\_axis' is text up to 18 chars, e.g. "X, CW".

The command with no parameters will print the entire current list. If only one parameter without value is given the corresponding value is printed.

In an automatic sequence, starting values are automatically incremented by their corresponding increments (Start\_angle is incremented by Angle\_increment, Phi by Phi\_increment, Chi by Chi\_increment and Position by Position\_increment).

The values of these parameters may alternatively be specified in the CBF header template. The parameter 'CBF\_template\_file' gives the full path to the template. Note that the CBF template may be used to set variables even when TIFF images are to be written.

'*MxSettings* CBF\_template\_file 0' can be used to turn off this setting.

See also the `cbf_template_HOWTO.txt` in the `$HOME/p2_det/tvx/docs` directory for more information.

# 11 Flat Field Image

## 11.1 Using the Flat Field Correction Image in Camserver



The flat field images recorded during factory calibration are NOT automatically applied to the images that you take.



The factory recorded flat field is only appropriate to use at the energy and threshold settings at which it is recorded (no special correction, recommended usage only at 100K and 300K detectors). The energy and threshold values are encoded in the directory name where the correction file FFcorr.tif is stored and these directories are located in /home/det/p2\_det/config. In the following example, FF\_pxxxkyyyy\_E8p0\_T5p0\_vrf\_m0p2, E8p0 stands for an incoming X-ray energy of 8.0keV, T5p0 a threshold of 5.0keV and vrf\_m0p2 means Vrf = -0.2 V which is mid gain. The last part can be vrf\_m0p3, vrf\_m0p2 and vrf\_m0p15 which is low gain, mid gain and high gain respectively.

| Command                          | Description   |
|----------------------------------|---|
| <i>LdFlatField<br/>imagename</i> | The Camserver command " <i>LdFlatField</i> " will cause Camserver to apply a specified flat field correction to every image taken. The file is a floating-point TIFF file with a mean value near 1.0. |

The command applies the flat field ONLY temporarily. The information about the flat field file is lost once Camserver is restarted or a new threshold/energy (using the *SetThreshold* or the *SetEnergy* command) is set.

To automatically apply the flat field image please remove the "#" signs in front of all lines which contain "ffield" in the file /home/det/p2\_det/config/cam\_data/calibrations.def. This will cause the *SetThreshold* or the *SetEnergy* command to apply the settings stored in calibration.def file.

## 11.2 Creating a new Flat Field Image

Intensity correction images can be created by the user. To get an appropriate flat field from an homogeneous illumination one can calculate it manually or use a TVX command.



First set up a very uniform X-ray illumination and verify that it is as flat as possible. This can be very difficult, but is critical to success.

- Load the required threshold settings.
- Record a picture with the desired statistics; at least  $10^4$  counts per pixel is recommended.



The longer the exposure time, the better the statistics.

- Divide the mean value of the recorded flat field image by the image itself. To perform this operation with TVX see section 12.6.
- The resulting image should contain values of around 1.

To use the flat field file please see the previous section 11.1.

## 11.3 Using the Flat Field Correction Image in TVX

To apply the correction to an existing image, issue the command "move new\_image=image!imi". Where "new\_image" is the new image that will be created and "image" is the image that is to be corrected.

When you first ask for an intensity correction, TVX asks for a file, which becomes permanent until exiting the program. The command "**setint** correction\_image" can be used to change it. By default the file is assumed to be in '~/p2\_det/correct/'. By explicitly stating the path and image you can specify an image in a different directory. Issuing the command "**setint**" without any argument will list the currently used correction image, if any.

## 12 How to Use the Pilatus Detector through TVX


TVX is a powerful tool for data acquisition and analysis. This section describes only the most commonly used commands in TVX. All commands are case-insensitive; however, filenames are case-sensitive.



An 'object' in TVX may be an image or a graph. Many commands, such as *move*, will work on objects of either kind. Objects can be combined with standard arithmetic operators (+, -, \*, /, +=, etc.), logical operators (<, >, <=, >=, |, ||, &, &&) and special operators (<<, >>, !, :, <=<, etc.) in arbitrarily complex expressions to perform sophisticated analyses and to construct custom scripts. In case of doubt, try it out: you can't hurt anything.

Many commands in TVX require an input value or argument. Without the declaration of a value, the currently set value is shown.



In this manual *input values* are shown in *Italic*.

| <b><i>Command or <u>Macro</u></i></b>       | <b>Description</b>   |
|---|--|
| <b><i>menu</i></b>                          | Shows all commands<br>It is divided in 5 parts: <ul style="list-style-type: none"> <li>• <b>Reserved Words:</b> (do not use as variables)</li> <li>• <b>External Procedures:</b></li> <li>• <b>User Commands in current directory:</b> All TVX commands; use <i>man command</i> to get a detailed description)</li> <li>• <b>Defined variables &amp; strings:</b> This are macros which are created during startup (e.g. in default.gl); use <i>show macroname</i> to see more details</li> <li>• <b>User Variables:</b> assigned variables</li> </ul> |
| <b><i>help command -or- man command</i></b> | Displays the help text for the <i>command</i> . Help <i>help</i> is a good way to start.   |
| <b><i>show <u>macroname</u></i></b>         | Displays the definition of the macro   |
| ESC-button                                  | Stop a running task and return to the TVX line interpreter   |
| CTRL-C                                      | Full reset of TVX<br><br> Do not use this in Camserver!   |

| <b><u>Command or Macro</u></b>                     | <b>Description</b>  |
|--|---|
| <b><u>rbd</u></b>                                  | <p>Read Back Detector.<br/>Self-test of the digital part of the detector. Sends a digital pattern to each pixel, reads it out and displays the image.</p>  <p>Use this command always after a startup.</p> <p>Every pixel shows 1000 counts.</p>             |
| <b><u>calibdet</u></b>                             | <p>Self-test of the detector.<br/>Sends 100 calibrate pulses to the analog part of each pixel, reads back the recorded values as an image and displays the result.</p>  <p>Use this command always after a startup.</p> <p>Every pixel shows 100 counts.</p> |
| <b><u>setdac</u></b>                               | Sets all Digital Analog Converters (DAC) to the predefined values.  |
| <b><u>imagepath</u> path</b>                       | <p>Image Path<br/>Without the input of a path it displays the current default path. With a declaration it changes the default path for images. The imagepath command also sets the autaname to the new path. The keyword 'image path' can be used in expressions as [image path].</p>   |
| <b><u>grafpath</u> path</b>                        | <p>Display or change the default path for graphs. The keyword 'grafpath' can be used in expressions as [grafpath]</p>   |
| <b><u>expose</u> exposure time</b><br>(in seconds) | <p>expose 1: makes an image with an exposure time of 1 sec.<br/>Shortest exposure time: &gt;0.000 001s.<br/>Shows the exposed image and its name immediately after completion.</p>  |
| <b><u>exposem</u></b><br>exposure time             | <p>continuous camera mode without saving images.<br/>Takes images until any key is pressed. The last image is stored in temp.tif</p>  |
| <b><u>disp</u> filename</b>                        | Display an image (see Section 12.1). Opens up to 3 windows with successive invocations.   |
| <b><u>disp1</u> filename</b>                       | Displays an image reusing the last window. Useful in loops.   |

| <b><i>Command or Macro</i></b>            | <b>Description</b>  |
|---|---|
| <b><i>graf</i></b> <i>fn1[fn2[ fn3]]</i>  | Graph up to 3 graphs in a window  |
| <b><i>Convert</i></b> <i>[S][D][type]</i> | Change the image data type. S...source image, D...destination image. Type... Char, Short, Int, Float  |
| <b><i>define</i></b>                      | DEFINE name="instruction1; instruction2;....."<br>Defines user symbol name and value.<br>E.g. define tpict="zpict; move imt=im3" defines symbol tpict as a combination of 'zpict', and the built-in move instruction. |
| <b><i>CaptureIM</i></b> <i>filename</i>   | Capture the default image to filename<br>captures a displayed image (and its zoom) as a .ppm (portable pixmap) file, including coloration and contrast adjustments.   |
| <b><i>CaptureGR</i></b> <i>filename</i>   | Capture the default graph to filename<br>Captures a displayed graph (and its zoom) as a .ppm (portable pixmap) file.  |
| <b><i>connect</i></b> <i>[ip_address]</i> | Connect the socket connection from TVX to the Camserver at <i>[ip_address]</i> . The IP is not necessary if TVX is running on the same computer.  |
| <b><i>disconnect</i></b>                  | Disconnect the socket connection from TVX to Camserver. E.g, so that a beamline operating system like EPICS can take control over the Camserver.  |



## 12.1 Description of the Image Display

The TVX command *disp* allows display of images. It contains several options to adjust the contrast and the min-max values. Moreover, it is possible to display the image in different color schemes. This image display will automatically show up after you execute the *expose* macro in TVX.

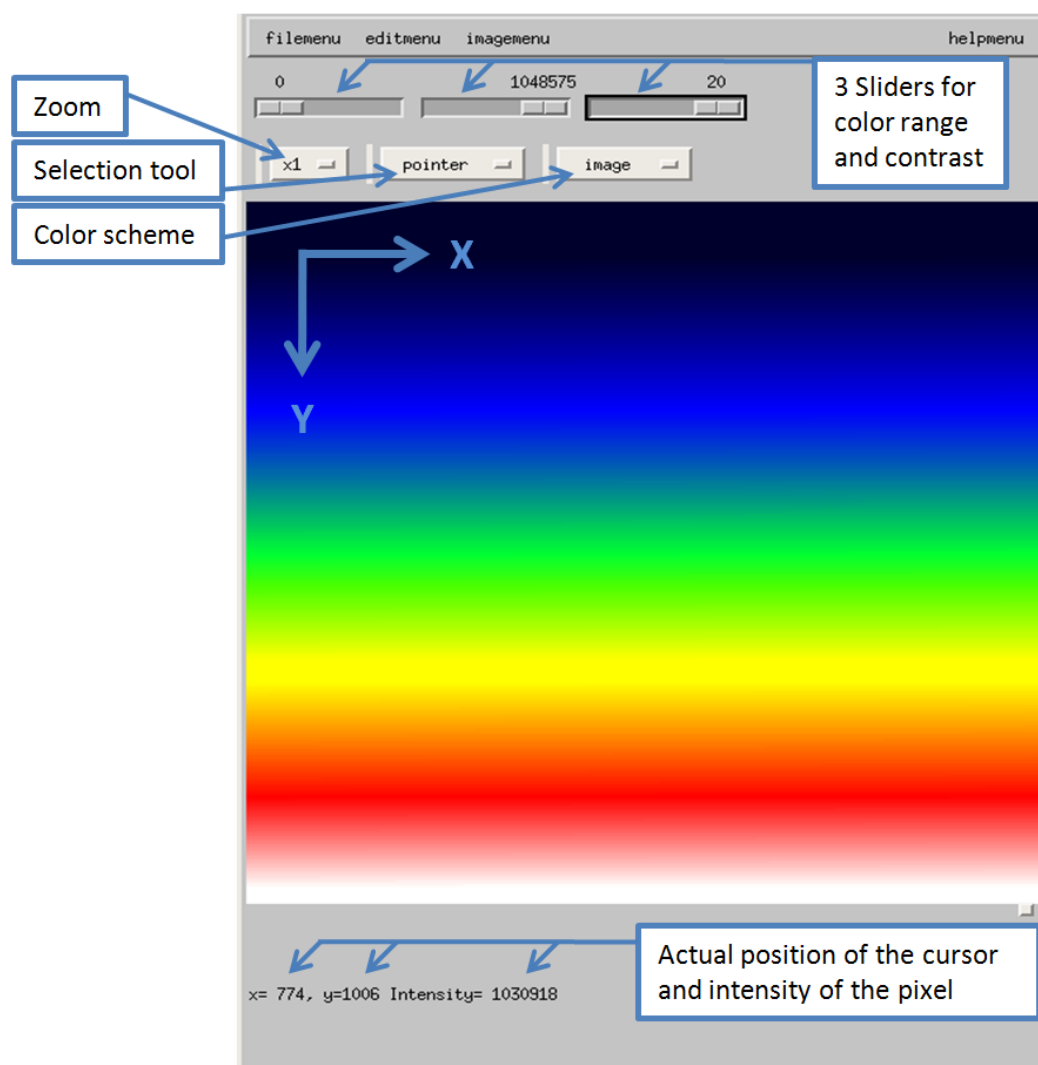


Figure 13. Image display from TVX

| Display tool           | Description   |
|------------------------|---|
| (sliders)              | <p>Define the color and the contrast of the image. For every value of a pixel a color from a lookup table will be displayed.</p> <p>With the two left sliders the cut off for the low and high values can be set. Values outside this range are displayed with the same color. The third slider defines the contrast factor.</p> <p>The sliders can be moved with the mouse or by putting the mouse on the slider and adjusting the value with the left and right cursor buttons. They can also be set from the command line using the <i>disp</i> command (use <i>man disp</i>).</p> |
| zoom                   | A magnification can be chosen and the enlarged area is shown in a new window. The zoom outline in the main window can be positioned by clicking or dragging with the mouse with the right button depressed.   |
| <b>Selection tools</b> |   |
| pointer                | normal pointer  |
| annulus                | <p>Allows analysis of circular areas.</p> <p>The sizes of the circles can be adjusted with the mouse or directly by the setting the values in the image window.</p>   |
| box                    | <p>Allows analysis of rectangular areas. Move the box with the right mouse or place the center of the box with the left mouse button. The size of the box can be adjusted with the mouse or directly by setting the values in the image window.</p>   |
| butterfly              | <p>Allows analysis of special shaped areas.</p> <p>The shape of the area can be adjusted with the mouse or directly by the setting the values in the image window. The circle is only for alignment purposes.</p>   |
| Line                   | Distance measuring tool. Requires that the correct pixel size be set in detector setup file (~/.p2_det/tvxrc)   |
| resolution             | Resolution circles for crystallographic patterns. Calculates the resolution of the image. The correct parameters for the detector should be set in the detector setup file or from the command line, (det_dist, lamdba and pixel size)  |
| <b>Display mode</b>    |   |
| grays                  | color lookup table with gray scale.   |
| spectral               | color lookup table with a spectral distribution (blue and   |

| Display tool | Description   |
|--------------|---|
|              | black near zero, red fading to pink and white at the high end)  |
| thermal      | color lookup table going from blue through yellow and red, but no greens  |
| decades      | The values between Min and Max are displayed linearly, but with the scale wrapping around Scal number of times. Thus, Scal = 1 is linear, Scal = 5 covers the range Min to Max with 5 linear segments going from 0 to 255, 0 to 255, etc.<br>This gives lots of artificial contrast that is good for smoothly-varying SAXS data, but is otherwise rather non-intuitive.   |
| power        | The image is displayed between Min and Max using the transfer function:<br>$(\# \text{ grays}) * ((\text{value} - \text{min}) / (\text{max} - \text{min})) * (\text{Scal} / 15)$<br># grays is usually 256. Thus, a small value of Scal (~3) gives a very steep transfer function at low values, and very little contrast at high values.<br>Scal = 15 is a linear transfer function; Scal > 15 is useful only in special cases |
| reverse      | The values are reversed - X-rays in the image become black rather than white.<br>Useful for crystallographic images.  |

Several test images and graphs are included in the system.



Try the following:

**imagepath** examples  
**disp** gray20bit.tif

**grafpath** examples  
**grafdemo**

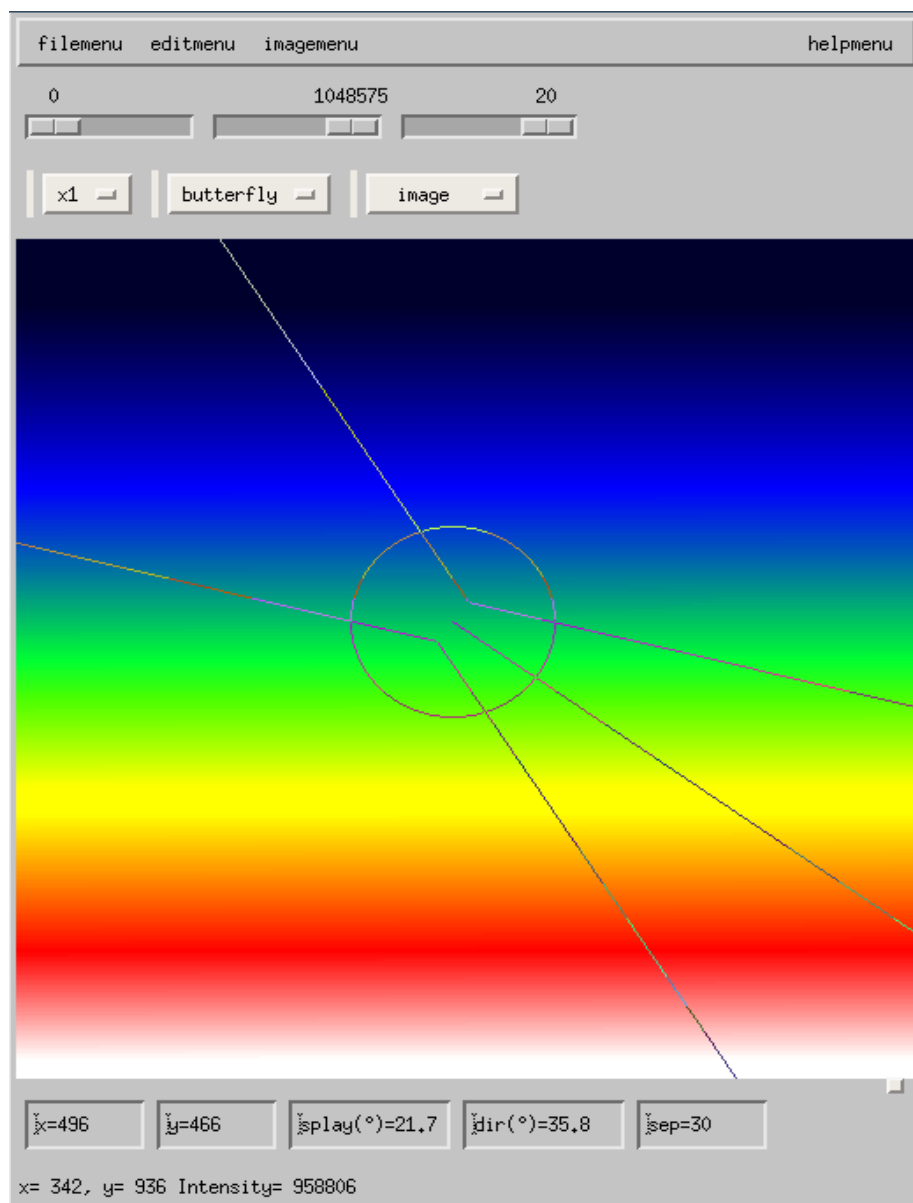
More examples are in:

/home/det/p2\_det/programs/tvx/test/images -and-  
/home/det/p2\_det/programs/tvx/test/graphs



### Example: Butterfly selection tool

This selection tool is useful for straight line integrations (densitometer traces) and for integrating small angle scattering patterns from either a line or a point X-ray source.



**Figure 14. Example of a the butterfly selection tool**

The size and position can be adjusted directly with the mouse or by typing the values directly into the boxes. The circle is used only as a positioning aid. Use the keyword *integrate* in the TVX window to display the result.

## 12.2 Analysis commands

TVX offers a large variety of image analyzing and processing commands. The most important commands are described in this document. All created numeric data are stored in the directory given by “**grafpath**”; image data are stored in the directory given by “**imagepath**”.

| Command                                  | Description  |
|--|--|
| <b><i>move</i></b> <i>fn1=fn2</i>        | The basic image manipulation command. In the simple form shown, this copies an image to a new name or directory. However, <i>fn2</i> can be any arithmetic expression of images and constants.   |
| <b><i>Integrate</i></b>                  | integrate the pixels selected by the current selection tool - box, butterfly (includes straight-line case), or spot (annulus tool) - and show the resulting graph.<br>Usage: integrate [IM] [graph_name]<br>For the butterfly, the graph name can be given as the second parameter; in this case the image name must be specified. In other cases, the default image is used if no image is specified.   |
| <b><i>histogram</i></b> <i>lo hi int</i> | Histogram of the pixels selected by the box tool on the image. Alternatively, specify the image and region-of-interest on the command line.<br><br>Usage: histogram [IM] lo hi int [x1 y1 x2 y2] [graph_name], where lo is the first value to use, hi is the last value, and int is the interval. If IM is not specified, the default IM is used. [x1 y1 x2 y2] are the coordinates of the box to be histogrammed. If no graph_name is specified, the histogram is placed in file 'hist[n].dat' in the default graph directory, where n rotates through the values 0...5. This file can be then be moved to a permanent file by a command such as "move myhist=hist1". The histogram parameters are remembered, so subsequent operations with the same parameters can be obtained by just typing 'histogram'. If the coordinates are specified on the command line, the parameters must also be specified. If the file name is specified, either the image name must also be given, or 3 (or 7) numeric parameters must be specified. In the integral mode, the integral is written to 'hist[n+1].dat'. if the name is specified, it is appended with "_i" for the integral.<br>See also 'histset' |

|                                  |  |
|----------------------------------|--|
| <b><i>box</i></b>                | <p>Print statistics from the current box selection tool on the image. Alternatively, specify image name and box coordinates on the command line.</p> <p>Usage: <code>box [IM] [x1 y1 x2 y2]</code><br/>         If IM is not given, the default IM is used.<br/>         [x1 y1 x2 y2] are the coordinates of box to be examined. If not given, use the box selection tool on image. If given, the box selection tool is created or updated on the image, if it is displayed. If the box is set with the mouse, 'box' and 'integrate' give the same result. Several system variables are set: counts (total counts in box), area, mean, minimum, maximum, stdev (rms), var (variance), xcen &amp; ycen (centroid), box_x1, box_x2, box_y1 &amp; box_y2 (corners of box).</p> |
| <b><i>boxall</i></b>             | Print statistics from the whole (current) image.   |
| <b><i>format n1[.n2]</i></b>     | Control the number of digits to be printed (n1) or the number of decimal places (n2).  |
| <b><i>deleteallobjs</i></b>      | Delete the TVX record of all objects – the objects themselves are untouched. Images are stored in the TVX memory up to the limit specified in <code>tvxrc</code> , which can consume significant resources; use this command to free up memory. In addition, one can create files with identical names in various directories. To avoid the necessity of always specifying full path names, use this command to clear the TVX memory.  |
| <b><i>deleteobj filename</i></b> | Deletes the specified object from the TVX memory. The file on disk is untouched.   |
| <b><i>maskimg</i></b>            | Specify a mask image to be used by many TVX commands, such as box and histogram. See below.  |

## 12.3 Mask files

Setting a mask image is useful when you are looking at the statistics of images from the detector. Pixels in the detector that are either dead, too noisy or behave in a non-desirable manner can be masked out. After a pixel has been masked, it will no longer be considered when using statistical routines in TVX to analyze your image so that your results will not be distorted by pixels with too high or too low values.

A mask file is an image file that uses only two distinct values for each pixel. Every pixel that is to be masked out is given a value of 0, every other pixel is given a value of 1. You can create a mask file from another image by using the command "mkmask".

| Command                           | Description   |
|-----------------------------------|---|
| <b><i>mkmask</i></b>              | <p>Make a mask from an image between two limits, inclusively.</p> <p>Usage: mkmask [IM [IMout]] low high</p> <p>The result is a mask of 1's and 0's which can be used to select pixels of an image by multiplication. If no image is supplied, the default is used. Note that a float input object returns a 32-bit integer mask.</p> <p>Because the generated file is a normal image you can use any of the image manipulation tools supplied in TVX to alter your mask image if you wish.</p> |
| <b><i>masking</i></b>             | <p>Declare, inquire about, or turn off the current mask image.</p> <p>Usage: masking [im] -or- masking 0</p> <p>If present, the mask is used to blank out bad pixels in statistical routines such as box, integrate, spot &amp; histogram. Zeros in the mask are excluded from the analysis, non-zeroes are included.</p> <p>With no argument, displays the current mask image name, if any. With numeric argument (e.g. 0), turn off the mask image.</p>                                       |
| <b><u>ldm</u></b>                 | <p>Ldm is short for load mask (a macro) and uses the masking command and the factory produced mask (stored in \$HOME/p2_det/correct/goodpix_mask.tif).</p>  |
| <b><i>pixlfill</i> [IM] value</b> | <p>Set pixels in <i>IM</i> to <i>value</i> using the current box (coordinates) as a template. This permits you to manually alter a mask image based on observations on a different image. Alternatively, the box coordinates can be specified as described in section 9.1.1.</p>  |



If the command "deleteallobjs" is used after you have loaded a mask image your masking will be reset; of course, the stored image is untouched.

## 12.4 User defined commands

TVX supports complex C-like commands in the command line.



### Example:

To display a series of images as a movie:

```
format 2; for (i=0;i<100;i++){disp1 image_000[i]; wait 0.5}
```

Displays image\_00000 to image\_00099 and waits 0.5 seconds between each picture. The brackets [ ] mean to substitute the enclosed argument as text with the number of digits specified by the format.

With **define** one can create custom commands for the current session and eventually save them for reuse.



### Example:

```
define test1="format 2; for (i=0;i<100;i++){disp1 image_000[i]; wait 0.5}"
```

| Command  | Description   |
|--|---|
| <b>define</b><br><i>name="string"</i><br><b>define</b> <i>name=value</i> | Define a name (value or command) which can be used in the current session. They are not saved when TVX closes.  |
| <b>save</b> "myfile.gl"  | Saves the currently defined commands in <i>myfile.gl</i> as text. Such files are called glossaries. Glossaries may also have executable commands edited in following all the definitions; these are preserved when the file is overwritten. |
| <b>get</b> "myfile.gl"   | Load the definitions from <i>myfile.gl</i> , and execute any commands appended after the definitions.   |



## 12.5 Glossary files

When TVX is started, a glossary is automatically started up called */home/det/p2\_det/config/default.gl*.

In this glossary, the main commands for using the detector are defined. Three other glossaries are called from *default.gl* (all in config):

| Glossary    | Description  |
|-------------|--|
| det_spec.gl | Detector specific definitions. In case of multi module detectors number of banks, modules, tools for addressing modules and analyzing module specific data.  |
| user.gl     | User specific commands   |
| startup.gl  | Commands which are automatically loaded at startup, e.g. <i>setdac</i> , <i>rbd</i> , <i>calibdet</i> . For usage at the beamline, usually the last command is Disconnect, which allows remote control of Camserver. |

## 12.6 Example

The following line is a simple example of using TVX to create a flat field correction file (corr.tif) out of a high intensity count image (image\_00001.tif). After you recorded the image with adequate statistics and stored it in \$Home/p2\_det/images, you can use the following commands:

- ***disp*** image\_00001.tif
- ***ldm***
- ***boxall***
- ***convert*** image\_00001.tif corr\_image\_float.tif float
- ***move*** corr.tif=[mean]/corr\_image\_float.tif

If the image\_00001.tif was an appropriate “flat” image (see section 11 for details) the 5 lines create a proper correction image (corr.tif) which can be used for flat field correction.

## 13 Factory Calibration and Correction

The following calibrations are done at the DECTRIS premises:

### 1) Threshold Calibration

The PILATUS detector systems come fully calibrated. See the system information sheet in your user handbook for more information about the calibrated energies and settings.

The discriminator thresholds in the individual pixels are set by an automated procedure (described above).

### 2) Rate Correction

The counter in the pixels is a classical paralyzable counter with a dead time that depends on the gain (amplifier) settings. The correction required is negligible up to  $\sim 10^5$  counts/s/pixel, but becomes quite significant approaching  $10^6$ /s; above  $\sim 2 \cdot 10^6$ /s the conversion is cut off at a "saturation" value. This value is printed in the header and can be used as a flag in analysis software. Rate correction is optionally turned off in the control software (e.g. triggering on short single X-ray bursts).

### 3) Distortion (only for Multi Module Systems)

A text file with a map of the offset in position and angle of each module with respect to some common origin will be provided.

The processing program must incorporate this information, using it as a lookup table to map sought reciprocal space positions onto detector positions. However, due to the high manufacturing quality of the detector and the direct detection, geometrical distortions are minimal and excellent results are obtained without correction.

### 4) Parallax

The silicon sensor is 0.320 mm thick. The parallax correction as a function of energy and angle of incidence has been well modeled and is about 1 pixel displacement at an angle of incidence of 45 deg. Parallax actually improves spatial resolution because a spot that is spread over a few of pixels can be localized better than a spot in just 1 pixel.

### 5) Flat Field

Different modules in a multi-module detector differ in sensitivity. This calibration needs to be done as a function of beam energy and energy to threshold ratio. The flat field map can be loaded into the detector controller, which performs the correction as the data is read; however, it is difficult to restore uncorrected data from a corrected image. Alternatively, crystallographic processing programs such as XDS can correct for variations in detector sensitivity which has a similar effect as a flat field correction.

## **6) Bad Pixels**

The software permits reading a bad pixel mask and flagging defective pixels as -2 in the data. The gaps between the modules can optionally be flagged with -1 (zero is the default). Both of these flags are used by XDS.

## 14 Camserver Commands

The following list presents the Camserver commands with a short description. For detailed usage of the detector system please see sections 7 through 11.

| Command           | Arguments<br>(unit)<br>[default values]         | Description  | Socket<br>connection<br>return code          | Socket connection<br>return text  |
|-------------------|---|--|--|---|
| <i>Exposure</i>   | file_base_name +<br>file extension <sup>1</sup> | <p>Make an exposure</p> <p>ExpTime, ExpPeriod, ImgPath and NImages have to be set beforehand to the desired values. The image is written to the specified file_base_name<sup>1</sup> relative to ImgPath, or to an absolute path if given. The format of the image is derived from the filename extension if given (tif, cbf or edf); otherwise a raw image is written. If an exposure series is set up (NImages &gt;1), an image number is inserted before the extension<sup>1</sup>.</p> | <p>at the start: 15</p> <p>at the end: 7</p> | <p>at start: starting xxx<br/>second background:<br/>&lt;date &amp; time&gt;</p> <p>at the end: full path name<br/>of last image</p>            |
| <i>ExtTrigger</i> | file_base_name +<br>file extension <sup>1</sup> | <p>Arm the detector for an exposure or an exposure series started by one external trigger.</p> <p>Before execution, set timing parameters by the commands ExpTime and ExpPeriod. To specify a delay between the trigger and the start of the exposure a "Delay" time can be set.</p> <p>The time from arming the system until the arrival of the first trigger is unlimited. Use 'K' transmitted over the socket connection to interrupt this state.</p>                                   | <p>at the start: 15</p> <p>at the end: 7</p> | <p>at start: Starting externally<br/>triggered exposure(s):<br/>&lt;date &amp; time&gt;</p> <p>at the end: full path name<br/>of last image</p> |

|                    |   |   |  |   |
|--------------------|---|---|--|---|
| <i>ExtMtrigger</i> | file_base_name +<br>file extension <sup>1</sup> | <p>Arm the detector for an exposure series where each exposure is started by an external trigger.</p> <p>Set timing parameters by the commands ExpTime and ExpPeriod before execution. Each exposure is triggered by the external trigger, but uses the internal timer for the exposure time. Individual exposures can be added up within one readout/image by specifying NExpFrame prior to execution (see Multiple Exposure Mode 7.3.5).</p> <p>The time from arming the system until the arrival of the first trigger is unlimited. Use 'K' transmitted over the socket connection to interrupt this state.</p> <p>If the time between ExpPeriod and ExpTime is shorter than 13 ms and the ExpPeriod is shorter than 33.9 ms (faster than ~30 Hz) all trigger signals have to arrive approximately within +/- 30% of the set ExpPeriod and ExpTime. The required minimum time difference between ExpPeriod and ExpTime (readout time) is 2.28 ms (see command ExpPeriod).</p> <p>If the frame rate is less than 30 Hz<sup>3</sup> and the readout time is more than 13 ms the trigger signals do not have to arrive within the mentioned +/-30%, but the ExpTime and ExpPeriod still have to be defined similar to the shortest expected signal repetitions. Use 'K' transmitted over the socket connection to interrupt this state.</p> | <p>at the start: 15</p> <p>at the end: 7</p> | <p>at start: Starting externally multi-triggered exposure(s): &lt;date &amp; time&gt;</p> <p>at the end: full path name of last image</p> |
|--------------------|---|---|--|---|

|                  |   |  |  |   |
|------------------|---|--|--|---|
| <i>ExtEnable</i> | file_base_name +<br>file extension <sup>1</sup> | <p>Make an exposure or an exposure series using an external gate signal.</p> <p>Set timing parameters by the commands ExpTime and ExpPeriod before execution. Each exposure is started when the signal changes to high and is finished when the signal changes to low again. Individual exposures can be added within one readout/image by specifying NExpFrame prior to execution (see Multiple Exposure Mode 7.3.5).</p> <p>If the readout time, i.e. the difference between ExpPeriod and ExpTime, is shorter than 13 ms and the ExpPeriod is shorter than 33.9 ms (faster than ~30 Hz), all trigger signals have to arrive approximately within +/- 30% of the set ExpPeriod and ExpTime. The required minimum readout time is 2.28 ms (see command ExpPeriod).</p> <p>If the frame rate is less than 30 Hz<sup>3</sup> and the readout time is more than 13 ms the trigger signals do not have to arrive within the mentioned +/-30%, but the ExpTime and ExpPeriod still have to be defined similar to the shortest expected signal repetitions. Use 'K' transmitted over the socket connection to interrupt this state.</p> | <p>at the start: 15</p> <p>at the end: 7</p> | <p>at start: Starting externally enabled exposure(s):<br/>&lt;date &amp; time&gt;</p> <p>at the end: full path name of last image</p> |
| <i>ExpTime</i>   | Time (s) [1.0]                                  | Set the exposure time; time < 60days.  | 15   | Exposure time set to:<br>xxx sec.   |
| <i>ExpPeriod</i> | Time (s) [1.05]                                 | <p>Set the exposure period</p> <p>Time must be longer than exposure time + readout time (2.28 ms).</p> <p>The frame time is ExpPeriod*(NExpFrame-1) + ExpTime</p> <p>Time &lt; 60 days</p>   | 15   | Exposure period set to:<br>xxx sec  |

|                |                                    |   |    |  |
|----------------|------------------------------------|---|----|--|
| <i>ImgPath</i> | Path [/home/det/<br>p2_det/images] | <p>Change the image path</p> <p>If the directory does not exist, it will be created if it is possible to do so with write permission. A path relative to the current path is accepted; '..' is accepted. If 'imgpath test' is given, and the current directory named is 'test', a new directory is NOT created. If such a new directory is desired, it may be specified by 'test/test'. If 'imgpath test1/test2' is given, and the current path is '.../test1/test2', a new directory is NOT created.</p>   | 10 | the path   |
| <i>NImages</i> | Number (#) [1]                     | <p>Set the number of images for an automatic sequence</p> <p>Maximum number of images is 65535</p>  | 15 | N images set to: nn  |
| <i>Delay</i>   | Time (s) [0] dcb-<br>Number        | <p>Set the delay from the external trigger until start of the first exposure</p> <p>The time must be shorter than 64 s<br/>The delay is reset to 0 for ordinary exposures and external enable</p> <p>In a multi-dcb (detector control board ) detector, the dcb's can be individually delayed.<br/>If dcb-number is not given, t is applied to all dcb's.<br/>Delays of successive dcb's are cumulative; e.g., the sequence<br/>    Delay 0.0 0<br/>    Delay 0.1 1<br/>    Delay 0.1 2<br/>results in dcb 2 starting 0.2 s after the trigger, not 0.1 s</p> <p>ExpPeriod &gt; ExpTime + max_delay + readout_time,<br/>    where max_delay = 0.2 s in the above example and<br/>    readout_time = 0.00228 s.</p> | 15 | <p>Delay time set to: n.n sec</p> <p>-or-</p> <p>dcb 0 delay time set to:<br/>0.000000 sec.</p> <p>for all dcb's</p> |

|                   |                            |  |    |                                |
|-------------------|----------------------------|--|----|--------------------------------|
| <i>NExpFrame</i>  | Number (#) [1]             | <p>Set the number of exposures to accumulate per frame/read out.</p> <p>This is a method summing up images within the detector chip. A value &gt;1 is a waste of X-rays except when using external enable or external multi-trigger to synchronously capture an event. If <i>nexpframe</i>&gt;1, the reported 'measured exposure time" applies to the last exposure only. The maximum possible number is 2<sup>32</sup>-1.</p>   | 15 | Exposures per frame set to: nn |
| <i>MXsettings</i> | Mxparameters (text) [none] | <p>Set crystallographic parameters reported in the image header</p> <p><i>mxsettings</i> [parm_name value] [parm_name value] ...</p> <p>Possible:<br/>Wavelength, Energy_range, Detector_distance, Detector_Voffset, Beam_xy, Beam_x, Beam_y, Flux, Filter_transmission, Start_angle, Angle_increment, Detector_2theta, Polarization, Alpha, Kappa, Phi, Phi_increment, Chi, Chi_increment, Omega<sup>7</sup>, Omega_increment<sup>7</sup>, Oscillation_axis, N_oscillations, Start_position, Position_increment, Shutter_time, CBF_template_file</p> <p>Not settable with <i>mx_settings</i>, but provided to templates from detector settings:<br/>Timestamp, Exposure_period, Exposure_time, Count_cutoff, Compression_type, X_dimension, Y_dimension</p> | 15 | None set or current settings   |



|                               |                                    |   |    |   |
|-------------------------------|------------------------------------|---|----|---|
| <i>SetThreshold</i>           | Threshold parameters (text) [none] | <p>Set gain and threshold</p> <p>This allows setting the threshold energy in eV with following method:<br/>setthreshold [[gain] threshold]</p> <ul style="list-style-type: none"> <li>* If parameters are omitted, the current settings are shown.</li> <li>* Gain is 'highG', 'midG', 'lowG'.</li> <li>* If gain is omitted, the previous setting is retained.</li> <li>* setthreshold 0 turns off (invalidates) the current remembered settings; however nothing is transmitted to the detector. This may be used to force a reload of the trims</li> </ul> <p>This command builds a script in "/tmp/setthreshold.cam" and then loads it. The data for building the script are read from ~/p2_det/config/cam_data/calibration.def.</p>  | 15 | <p>Setting the threshold: pathname of file</p> <p>Without argument (once set):<br/>Settings: xxx gain; threshold: xxxx eV; vcmp: x.xxx V Trim file: /path/to/trim/file/abc.bin</p> <p>Without argument (never set):<br/>Threshold has not been set</p>                  |
| <i>SetEnergy</i> <sup>4</sup> | Energy (eV) [none]                 | <p>Simplified method to set gain and threshold</p> <p>The requested energy is used to calculate appropriate gain and threshold settings for the detector.</p> <ul style="list-style-type: none"> <li>* If energy is out of range, a default is set and a warning issued.</li> </ul> <p>If energy is omitted, the current energy &amp; threshold settings are shown</p> <p>If energy is 0, the energy setting is reset. The energy can be entered in 3 ways: this command, through the setthreshold command, or through MXsettings "wavelength [val] set". If wavelength is set, it is assumed to be definitive, and SetEnergy and the energy option in setThreshold have no effect. Only "setenergy 0" will overcome this lock. For finer control over detector settings, use SetThreshold.</p> | 15 | <p>Setting the energy: pathname of file</p> <p>Without argument (once set): Energy setting: xxxx eV Settings: xxx gain; threshold: xxxx eV; vcmp: x.xxx V Trim file: /path/to/trim/file/abc.bin</p> <p>Without argument (never set):<br/>Threshold has not been set</p> |

|                                |                           |  |         |   |
|--------------------------------|---------------------------|--|---------|---|
| <i>K</i>                       |                           | Stop data taking after current exposure  | 13<br>7 | ERR kill<br>full path name of last<br>image |
| <i>LdBadPixMap<sup>5</sup></i> | Filename (text)<br>[none] | Load a mask image giving bad pixels to be flagged<br><br>Filename must be a full pathname.<br><br>If filename is not given, the current setting is shown. If filename is '0' or "off", the pixel flagging function is turned off. The maximum number of bad pixels is 5000; the flag value is -2.                              | 15      | none or pathname                            |
| <i>LdFlatField<sup>6</sup></i> | Filename (text)<br>[none] | Load the flat-field correction file<br><br>Filename must be a full pathname.<br>File must be a 32-bit floating-point TIFF image.<br><br>If filename is not given, the current setting is shown. If filename is '0' or "off", the flat field function is turned off. The image is pixel-wise multiplied by the correction file. | 15      | none or pathname                            |
| <i>GapFill</i>                 | Number (0,-1) [0]         | Set the value to be used in pixels between modules. This only applies to multi-module detectors.<br><br>Number can only be 0 or -1.<br><br>If n is omitted, the current value is printed. For TVX-7.2.70-110214 and earlier: The SetThreshold command resets the Gapfill value to 0.   | 15      | Detector gap-fill is: nn                    |
| <i>THread</i>                  | Channel (n) [all]         | Read one of the temperature and humidity sensors<br><br>Channels are numbered 1-6 on the first detector control board, 7-12 on the second. If channel is not specified, # 0 is addressed. If no sensor is connected, -99 is printed.   | 215     | temperature and humidity                    |

|                     |                      |   |    |  |
|---------------------|----------------------|---|----|--|
| <i>Tau</i>          | Time (s) [0]         | <p>Control in-line count rate correction.</p> <p>A value of 0.0 turns off the in-line rate correction.</p> <p>The count rate correction value is threshold dependent and thus set by the SetThreshold command to an appropriate value. At very high counting rates (<math>&gt;1\text{e}6/\text{s}</math>) Tau can vary with storage ring fill pattern. Rate correction is invalid when the detector is synchronized to the bunch structure of a storage ring by an external enable signal. In this case it is advised to set Tau to zero after setting the threshold.</p> | 15 | Rate correction is on; tau = $\text{xxxe-}9$ s, cutoff = xxxx counts |
| <i>SetAckInt</i>    | Number (#) [0]       | <p>Set the interval for acknowledgements over the socket.</p> <p>This causes Camserver to acknowledge every n-th image.</p> <p>If N is omitted, the current value is shown. At the default (n=0) only the last exposure of a series is acknowledged. The initiating command is always acknowledged, so for 1 or more images, there is an acknowledgement before the start and at the end of a series. There are some restrictions at high frame rate: n cannot be too low.</p>  | 15 | none or current setting  |
| <i>ResetCam</i>     |                      | Reset the camera  | 15 | none   |
| <i>DebTime</i>      | Time (s) [0]         | <p>Set the contact debounce time for external enable mode</p> <p>If t is not given, the current setting is echoed. This is useful when the external enable is not “clean”, e.g., derived from a mechanical switch. The external enable pulse must be shorter than 85 sec.</p>   | 15 | Debounce time set to: n.n sec  |
| <i>HeaderString</i> | String (text) [none] | <p>Give a string to be included in the image header.</p> <p>The maximum length is 68 characters, no formatting permitted. Enclose the text in quotes to transmit non-alpha characters.</p>  | 15 | none   |

|                       |             |  |    |   |
|-----------------------|-------------|--|----|---|
| <i>DiscardMultilm</i> | yes/no [no] | <p>Discard multiple images</p> <p>During (too) fast images collections it can happen that the system loses images. It is possible to sum the counted photons in the following image or zero this.</p> <p>no/n/0 ... Multiple images will be preserved<br/>yes/y/1 ... Multiple images will be zeroed</p> | 15 | status  |
| <i>Exit, Quit</i>     |             | Close the socket connection  |    | none  |
| <i>Df</i>             |             | Show the number of 1 KB blocks available on ImgPath  | 5  | 1K blocks available   |
| <i>Dcb_init</i>       |             | Attempt full (re)initialize of the dcb (detector control board).   | 15 | dbglvl 1  |
| <i>ExpEnd</i>         |             | Return the filename with which the exposure ended  | 6  | full path name of last image  |
| <i>CamSetup</i>       |             | Report camera setup  | 2  | Camera definition:<br>Camera name:<br>Camera state:<br>Target file:<br>Time left:<br>Last image:<br>Master PID is:<br>Controlling PID is:<br>Exposure time:<br>Last completed image:<br>Shutter is: |
| <i>Telemetry</i>      |             | Report camera telemetry  | 18 | Image format:<br>and additional camera messages   |
| <i>Version</i>        |             | Print the TVX/Camserver version  | 24 | version   |
| <i>ShowPID</i>        |             | Show the PID of the process receiving the command  | 16 | the pid   |

**1...file\_base\_name** Exposure commands take a filename or file basename as their argument. For single images, the filename is used as typed; if a recognized image file format extension is present (.tif, .edf, .cbf), the file will be created in that format. Otherwise, a raw image is produced. For multiple exposure series (NImages > 1), the typed name is used as a basename; again, the extension, if given, is used to set the image format. The following examples show the interpretation of the basename:

| basename         | files produced        |                           |
|------------------|-----------------------|---------------------------|
| test6.tif        | test6_00000.tif,      | test6_00001.tif, ...      |
| test6_.tif       | test6_00000.tif,      | test6_00001.tif, ...      |
| test6_000.tif    | test6_000.tif,        | test6_001.tif, ...        |
| test6_014.tif    | test6_014.tif,        | test6_015.tif, ...        |
| test6_0008.tif   | test6_0008.tif,       | test6_0009.tif, ...       |
| test6_2_0035.tif | test6_2_0035.tif,     | test6_2_0036.tif, ...     |
| test6_014B.tif   | test6_014B_00000.tif, | test6_014B_00001.tif, ... |

I.e., the numbers following the last '\_' are taken as a format template, and as a start value. The minimum number of digits in the format is 3; there is no maximum; the default is 5. The format is also constrained by the requested number of images.

**2...Maximum time out during fast exposures (> 30 Hz)** Due to the Gigastar communication between the detector and the computer it is necessary that the first pulses arrives on the detector within the DMA (Direct Memory Access) timeout. Since this DMA timeout can vary from detector to detector it is displayed during Camserver start (e.g. Maximum DMA timeout = 8.1 s). See also section 7.3.1 for details.

**3...Tolerant pulse repetition (< 30 Hz)** TVX versions TVX-7.2.70-110214 and later are much more tolerant against variations in the trigger sequence for slow frame rates (< 30 Hz). The provided signal to the external trigger input on the detector can deviate from the adjusted ExpTime and ExpPeriod more than the usual +/-30%. However, the trigger signal must be still logical; e.g. a too early trigger signal (during the ExpTime) in the ExtMTrigger mode will be ignored. This may cause that at the end of the series the detector is still is waiting for one trigger pulse.

**4 ...** Only available in TVX versions TVX-7.2.70-101111 and newer.

**5 ...** Automatically set by the SetThreshold command.

**6 ...** The command **NFrameImg** is only available at TVX versions TVX-7.2.70-110331 and older. It controls or queries the number of frames to be summed to an image. n frames (readouts) are summed in software into a single image. The frames are individually rate-corrected. The resulting dynamic range is up to 31 bits. The maximum pixel value is  $2^{32}$ . It can be used with an integer number and will write back a return code of 15 and the following message: "Frames per image set to: nn"

**7 ...** Only available in TVX versions TVX-7.3.13-121212 and newer.



# 15 Camserver Test Client

/\*\*\*\*\*\

Name: cam\_client.c  
Created by: Sebastian Commichau, May 2009  
Modified by: Stefan Brandstetter, Feb 2011  
Purpose: Client for Camserver  
Compile with: gcc -o cam\_client cam\_client.c

DECTRIS Ltd.  
Neuenhoferstrasse 107  
CH-5400 Baden  
www.dectris.com

\\*\*\*\*\*/

```
#include <stdio.h>
#include <netdb.h>
#include <signal.h>
#include <unistd.h>
#include <string.h>

#define BUFSIZE 1024

int main() {

    // Change to required IP or hostname
    char server[64] = "localhost";

    // Change to required port
    int port = 41234;

    char buffer[BUFSIZE];

    int s;
    struct sockaddr_in s_addr;
    fd_set rfd;

    // Open socket descriptor
    s = socket(PF_INET, SOCK_STREAM, 0);

    // Resolve hostname and try to connect to server
    struct hostent *hostent = gethostbyname(server);
    s_addr.sin_family = PF_INET;
    s_addr.sin_port = htons((unsigned short) port);
    s_addr.sin_addr = *(struct in_addr*) hostent->h_addr;

    // Connect to socket
    if (connect(s, (struct sockaddr *) &s_addr, sizeof(s_addr)) < 0)
        return;

    printf("\n**** Command line socket interface for camserver ****\n\n");
    printf("Type 'exit' to quit\n");
```

```
// Main loop processing user input and socket input
while (1) {

    printf("cam_client> ");
    fflush(STDIN_FILENO);

    // Wait for data either from terminal (stdin) or from socket
    FD_ZERO(&rfd);
    FD_SET(s, &rfd);
    FD_SET(STDIN_FILENO, &rfd);

    if (select(((int) s)+1, &rfd, NULL, NULL, NULL)==-1)
        break;

    // Data from stdin
    if (FD_ISSET(STDIN_FILENO, &rfd)) {

        fgets(buffer, BUFSIZE, stdin);

        if (!strcmp(buffer,"exit\n"))
            break;

        // Replace carriage return by null character
        if (buffer[strlen(buffer)-1] == '\n')
            buffer[strlen(buffer)-1] = '\0';

        // Write to socket
        write(s, buffer, strlen(buffer)+1);

        bzero(buffer, sizeof(buffer));

    }
    // Data from socket
    else if (FD_ISSET(s, &rfd)) {

        // Read from socket
        if (read(s, buffer, BUFSIZE)==0) {
            printf("server not existing anymore, exiting...\n");
            break;
        }

        printf("%s\n",buffer);

        bzero(buffer, sizeof(buffer));

    }

}

close(s);

return 0;
}
```