USER MANUAL

PILATUS

This user manual is valid for the following PILATUS detectors:

- PILATUS 200K
- PILATUS 300K
- PILATUS 300K-W

DECTRIS®

Table of Contents

1 Document History

Actual document

Version	Date	Status	Prepared	Checked	Released
1.0	01.05.2013	released	SB, SC	SC, SB	CS

1.1 Changes

Version	Date	Changes
1.0	01.05.2013	First version

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

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.

2.3 Convention for Commands

Example	Description
cd ..	Shell commands are written in blue and italic
ExpTime	Camserver commands are in written italic
disp	TVX commands are written in bold italic
<u>***exposem***</u>	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

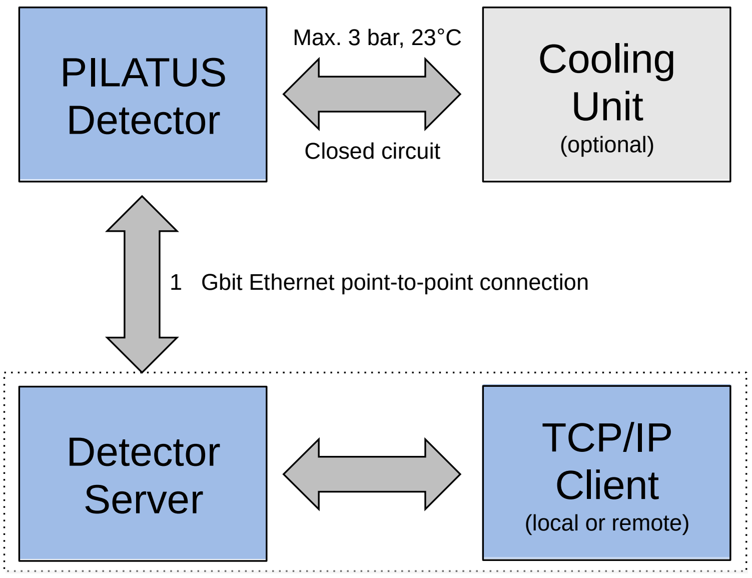
- Before turning the power supply on, check the supply voltage against the label on the power supply. Using an improper main voltage will destroy the power supply and damage the detector.
- Power down the detector system before connecting or disconnecting any cable.
- Make sure the cables are connected and properly secured.
- Avoid pressure or tension on the cables.
- The 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.
- DO NOT INSTALL ADDITIONAL SOFTWARE OR CHANGE THE OPERATING SYSTEM on the detector server except necessary network configurations.
- DO NOT TOUCH THE ENTRANCE WINDOW OF THE DETECTOR!

4 System Description

4.1 Overview

A PILATUS detector system consists of the following components:

- Detector
- Detector server
- Cooling unit



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 1) and processed in the CMOS readout chips. This new design has no dark current or readout noise, a high dynamic range of 20 bits a read out time of 7 ms, a frame rate of up to 20 images/s and an excellent point spread function of 1 pixel. The quantum efficiency depends on the silicon sensor thickness; however the detectors can be used for energies of up to 30 keV or more. The counting rate is greater than $2x10^6$/s/pixel, enough to perform any experiments using the high flux of any laboratory sources. However, the detector cannot withstand a direct beam.

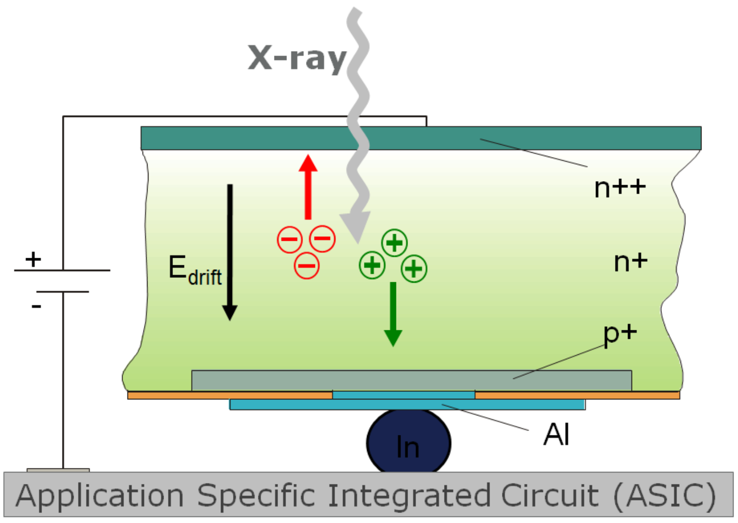
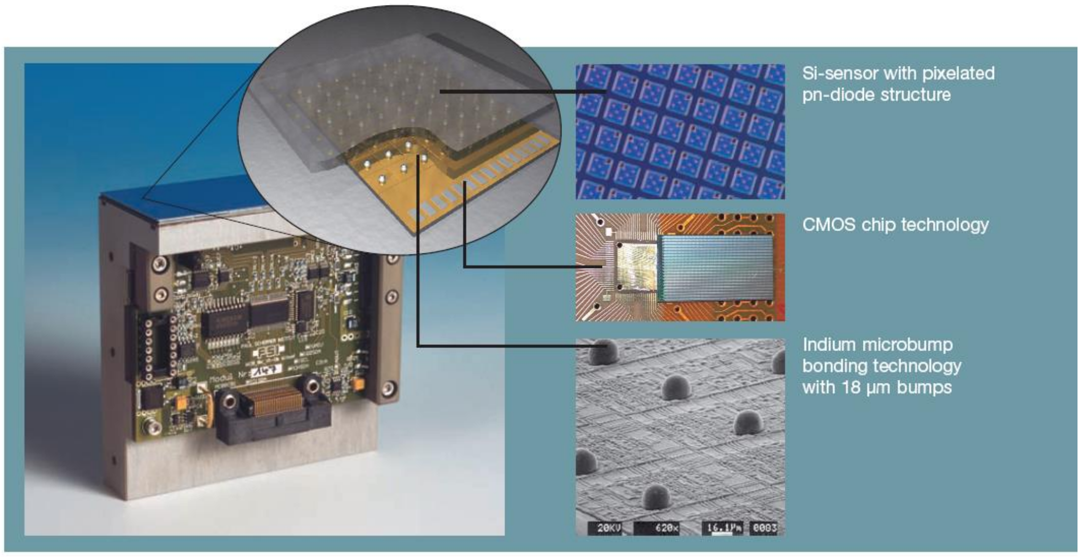


Figure 1. Principle of direct detection.

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 2). Each readout channel is connected to its corresponding detecting element through a microscopic indium ball, with a typical diameter of 18 μm. This connection process is called ‘bump-bonding’.



Si-sensor with pixelated pn-diode structure
CMOS chip technology
Indium microbump bonding technology with 18 μm bumps

Figure 2. 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 3).

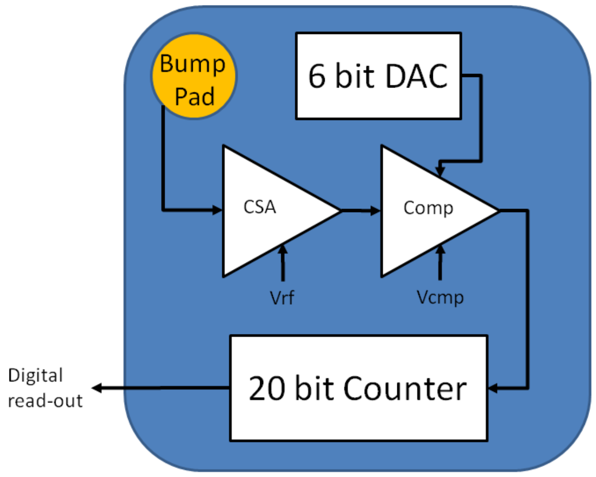


Figure 3. Design of each pixel. See text for details.

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 x 197 = 94965 pixels without dead areas and covers an active area of 83.8 x 33.5 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 2).

4.3 Software

The PILATUS detector system is controlled via Camserver.

4.3.1 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 feature of Camserver is to accept socket connections from a high level controller, which can provide high level services to this or other cameras. (see section 7 and 14 for more details). 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.

4.3.2 Description of the File Structure and Configuration Files on the PILATUS Detector Server

In the default setup, all necessary data 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 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
camdbg.out	./	Debug file for Camserver
p2det.set	config/cam_data	Startup file for Camserver

4.3.3 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. TVX contains a user interface and analysis tool suite. TVX communicates over a TCP/IP connection to Camserver.

4.3.4 Configuration Files for TVX

In the default setup, all data for the use of the PILATUS detector system are in the directory /home/det/p2_det. The following directories are relative to it.

Directory	**Sub Directory**	**Description**
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**
tvxrc	./	configuration file for TVX
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 documentatin

● Open a shell.

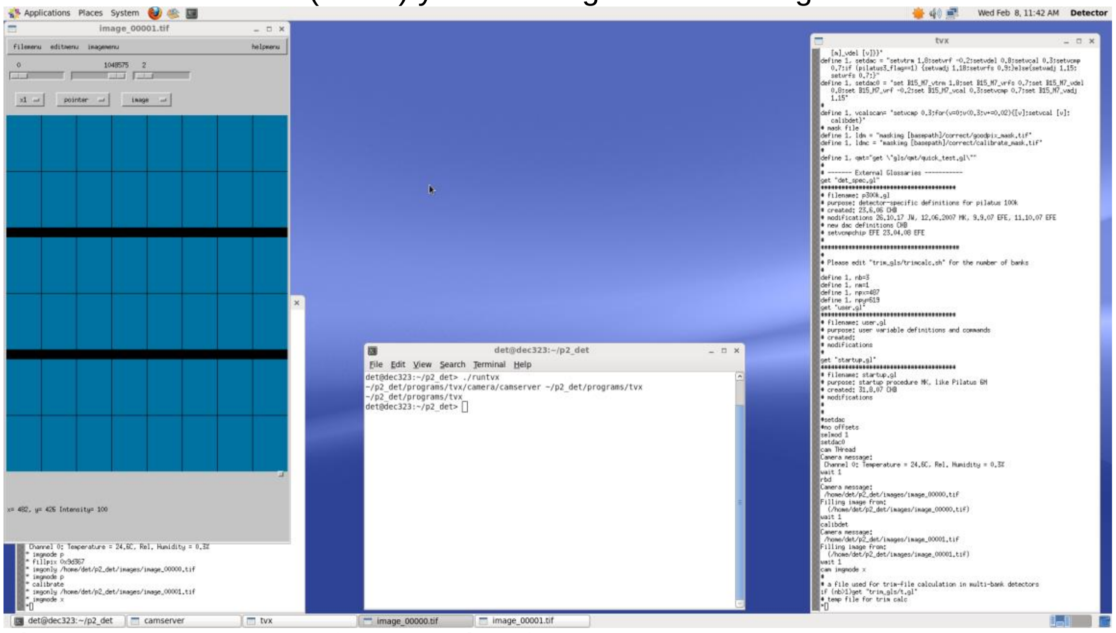
● Change to the p2_det directory. All following paths are given relative to this!

cd ~/p2_det↵

● Run TVX by typing

./runtvx↵

After the initialization (~20 s) you should get the following screen:


Applications Places System
Wed Feb 8, 11:42 AM Detector
image_00001.tif
filemenu editmenu imagemenu helpmenu
0 1048575 2
x1 pointer image
x= 482, y= 426 Intensity= 100
det@dec323:~/p2_det
File Edit View Search Terminal Help
tvx
det@dec323:~/p2_det camserver tvx image_00000.tif image_00001.tif

```
 Channel 0: Temperature = 24.6C, Rel. Humidity = 0.3%
* imgmode p
* fillpix 0x9d367
* imgonly /home/det/p2_det/images/image_00000.tif
* imgmode p
* calibrate
* imgonly /home/det/p2_det/images/image_00001.tif
* imgmode x
*
```

```
det@dec323:~/p2_det> ./runtvx
~/p2_det/programs/tvx/camera/camserver ~/p2_det/programs/tvx
~/p2_det/programs/tvx
det@dec323:~/p2_det>
```

```
   [m]_vdel [v]])"
define 1. setdac = "setvtrm 1.8;setvrf -0.2;setvdel 0.8;setvcal 0.3;setvcmp
   0.7;if (pilatus3_flag==1) {setvadj 1.18;setvrfs 0.9;}else{setvadj 1.15;
   setvrfs 0.7;}"
define 1. setdac0 = "set B15_M7_vtrm 1.8;set B15_M7_vrfs 0.7;set B15_M7_vdel
   0.8;set B15_M7_vrf -0.2;set B15_M7_vcal 0.3;setvcmp 0.7;set B15_M7_vadj
   1.15"
#
define 1. vcalscan= "setvcmp 0.3;for(v=0;v<0.3;v+=0.02){[v];setvcal [v];
   calibdet}"
# mask file
define 1. ldm = "masking [basepath]/correct/goodpix_mask.tif"
define 1. ldmc = "masking [basepath]/correct/calibrate_mask.tif"
#
define 1. qmt="get \"gls/qmt/quick_test.gl\""
#
# ------- External Glossaries -----------
get "det_spec.gl"
######################################
# filename: p300k.gl
# purpose: detector-specific definitions for pilatus 100k
# created: 23.6.06 CHB
# modifications 26.10.17 JW, 12.06.2007 MK, 9.9.07 EFE, 11.10.07 EFE
# new dac definitions CHB
# setvcmpchip EFE 23.04.08 EFE
#
######################################
#
# Please edit "trim_gls/trimcalc.sh" for the number of banks
#
define 1. nb=3
define 1. nm=1
define 1. npx=487
define 1. npy=619
get "user.gl"
######################################
# filename: user.gl
# purpose: user variable definitions and commands
# created:
# modifications
#
get "startup.gl"
######################################
# filename: startup.gl
# purpose: startup procedure MK. like Pilatus 6M
# created: 31.8.07 CHB
# modifications
#
#
#setdac
#no offsets
selmod 1
setdac0
cam THread
Camera message:
 Channel 0: Temperature = 24.6C, Rel. Humidity = 0.3%
wait 1
rbd
Camera message:
 /home/det/p2_det/images/image_00000.tif
Filling image from:
  (/home/det/p2_det/images/image_00000.tif)
wait 1
calibdet
Camera message:
 /home/det/p2_det/images/image_00001.tif
Filling image from:
  (/home/det/p2_det/images/image_00001.tif)
wait 1
cam imgmode x
#
# a file used for trim-file calculation in multi-bank detectors
if (nb>1)get "trim_gls/t.gl"
# temp file for trim calc
*
```

Figure 4. Startup screen after executing *./runtvx*

● The command *./runtvx* is a shell script which starts the programs Camserver (window on the left side of Figure 4) and TVX (window on the right side of Figure 4) 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 4 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 (*).

● The detector is now ready for operation with Copper (8keV) radiation. If you would like to change to other energies please see section 8.

● 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** 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.

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 14) 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 trimming commands (e.g. *SetCu*), 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 "*dbglvl* 5", the file 'camdbg.out' will contain many messages, including the exact contents of socket messages. Be sure to set "*dbglvl* 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 "*dbglvl* 6" reproducing the error can be helpful for diagnosis. Simply capture 'camdbg.out' and send it including a description of the issue 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 Institute (PSI) and by Mark Rivers of the University of Chicago:
http://cars.uchicago.edu/software/epics/areaDetector.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.

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 14).

Alternatively it also possible to start only Camserver with the command *./camonly* in the $HOME/p2_det directory.

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 "camclientt" 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.

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 section12).
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 14.

We recommend that full command names are used for clarity.

7.1 Main Commands

Command	Description
Exposure [filename]	Make an exposure with defined variables (e.g. exposure time and exposure period, see 7.1.1 and 14 for more details). The format of the file is determined from the supplied extension. The file is stored relative to the path defined by the *imgpath* unless an absolute path is given.
ExtTrigger [fname]	Start exposure with defined variables after receiving an external trigger and store images [fname].
ExtMTrigger [fname]	Start multiple exposures with defined variables after receiving multiple external triggers and store images [fname] (see section 7.3.2).
ExtEnable [fname]	Start exposure defined by external signal and store images in [fname].
help exposurenaming	Type this in the TVX window for a discussion of how exposure series are named.

7.1.1 Variables

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

Variable	Description
NImages [N]	Query or set the number of images in a sequence.
ExpTime [time]	Query or set the exposure time (10^{-6} to 10^{6} sec).

ExpPeriod [time]	Query or set the exposure period for serial exposures. The exposure period must be at least 0.95 ms longer than the exposure time.
imgpath [path]	Query or set the default imgpath.
Delay [time]	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.
nexpframe [N]	Number of exposures per frame This is a so called multi exposure mode. nexpframe sets the number of exposures before the detector is read out e.g. "*nexpframe 3*" exposes the detector 3 times before reading out an image of the 3 combined exposures. See point 7.3.4 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 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 TIF 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 TIF 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 1.0023000 s
    - # Tau = 0 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/
    - # Ratecorr_lut_directory: (nil)
    - # Retrigger_mode: 0

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.

External Trigger and External Multitrigger exposure commands need to have set the *ExpPeriod*, *ExpTime*, and *NImages* variables. The external enable mode only requires the *NImages* variable. However, *ExpPeriod*, *ExpTime* should be set for an adequate rate correction.

There is no timeout after executing a trigger command in Camserver: the detector will wait until a trigger signal arrives. This state can only be interrupted by transmitting a '*K*' to Camserver over the socket connection.

7.3.1 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.

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.

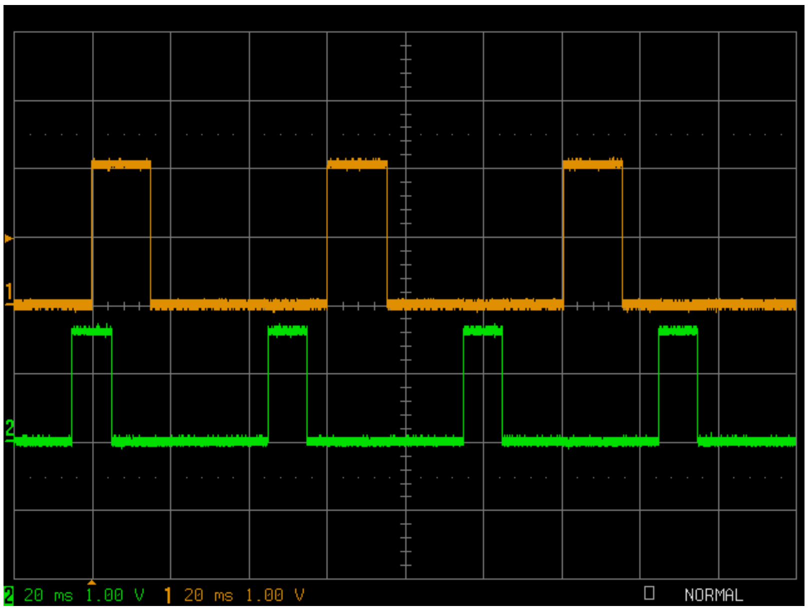
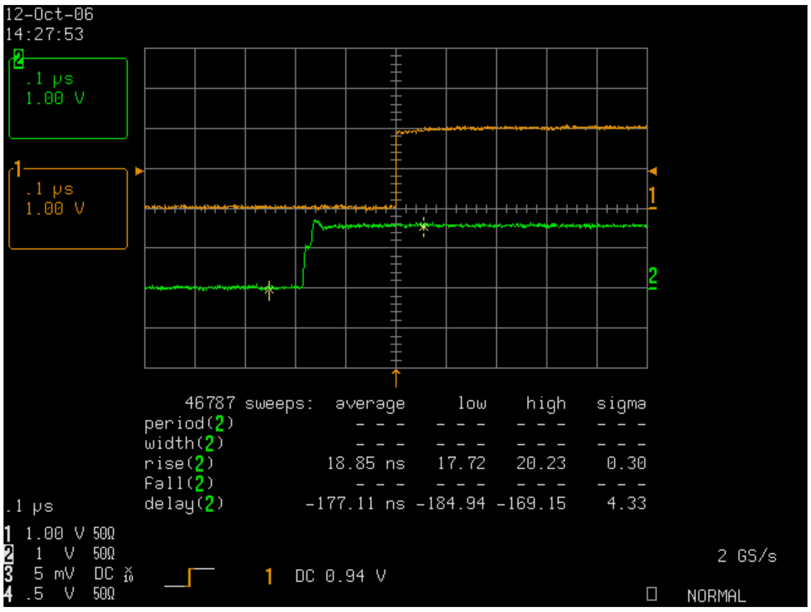


Figure 5. Oscilloscope trace of an external trigger. Orange: enable signal of the detector; Green: trigger.

In Figure 5 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 6).



```
46787 sweeps:  average      low     high    sigma
period(2)        - - -     - - -    - - -    - - -
width(2)         - - -     - - -    - - -    - - -
rise(2)        18.85 ns    17.72    20.23     0.30
fall(2)          - - -     - - -    - - -    - - -
delay(2)     -177.11 ns  -184.94  -169.15     4.33
```

```
.1 µs
1  1.00 V  50Ω
2    1  V  50Ω
3    5 mV  DC
4   .5  V  50Ω
1  DC 0.94 V
2 GS/s
NORMAL
```

Figure 6. Delay and jitter. Orange: enable signal of the detector; Green: trigger.

7.3.2 External Multi Trigger Mode

External multi trigger mode is started with the command "*ExtMTrigger imagename.tif*" where *imagename.tif* is the name of the images you wish to be taken. The 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 "*ExtMTrigger imagename.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" – for the first edge –, take an exposure as defined by "*ExpTime*", readout the image and will wait for the next trigger edge. This will be repeated "*NImages*" times.

The image number is only incremented during an exposure series; if you reissue the command "*ExtTrigger imagename.tif*" it will start writing images from "imagename_00000.tif" and overwrite existing data.

7.3.3 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* (optional, should be set for an adequate rate correction)
- *ExpPeriod* (optional, should be set for an adequate rate correction)

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.

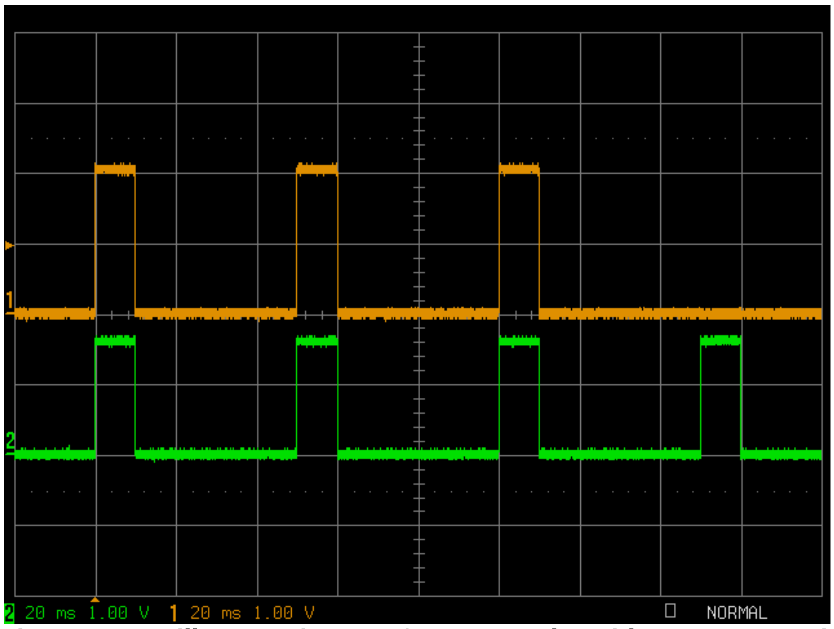
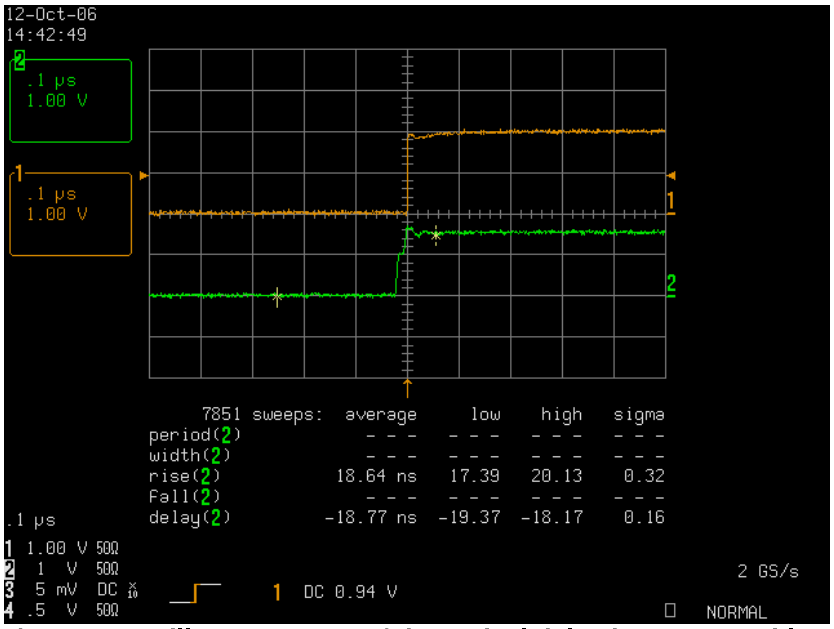


Figure 7. Oscilloscope image of an external enable. Orange: enable signal of the detector; Green: external gate.

In this example 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 signal provided to the detector. This can be seen in the oscilloscope image below.



```
        7851 sweeps:  average     low    high   sigma
period(2)               - - -   - - -   - - -   - - -
width(2)                - - -   - - -   - - -   - - -
rise(2)              18.64 ns   17.39   20.13    0.32
fall(2)                 - - -   - - -   - - -   - - -
delay(2)            -18.77 ns  -19.37  -18.17    0.16

1  1.00 V  50Ω
2   1   V  50Ω
3   5  mV  DC
4  .5   V  50Ω
                    1  DC 0.94 V
```

Figure 8. Oscilloscope trace of the typical delay between enable signal and exposure. Orange: enable signal of the detector; Green: trigger.

7.3.4 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.

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).

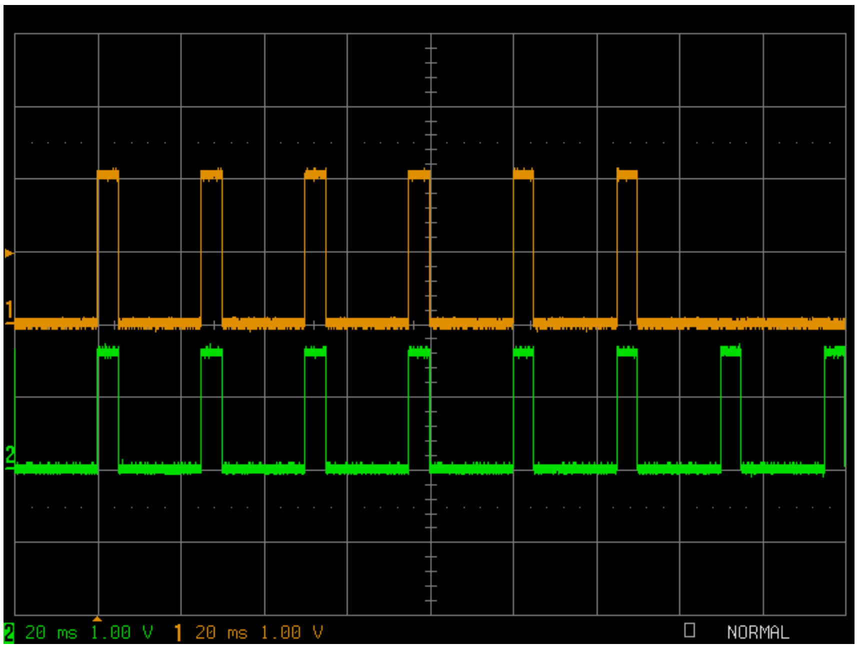


Figure 9. Oscilloscope image of the multiple exposure mode. Orange: enable signal of the detector; Green: external gate.

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 Trimming Method

The detector will be automatically trimmed for Copper radiation when started. During startup of Camserver the command *SetCu* is executed. This causes that the detector is trimmed optimal for Copper radiation.
To change the default behavior during startup please replace the line SetCu with another trimming command (e.g. SetMo for Molybdenum radiation) in the Camserver startup file (*/home/det/p2_det/*config/cam_data*/p2_det.set*).

The trimming of a running detector can be switched by using the corresponding trimming command.

Command	**Description**
SetCu	Appropriate for operating with Copper radiation
SetMo	Appropriate for operating with Molybdenum radiation
*SetCr**	Appropriate for operating with Chromium radiation
*SetFe**	Appropriate for operating with Iron radiation
*SetAg**	Appropriate for operating with Silver radiation
*SetEnergy [energy]**	SetEnergy - set or query the threshold setting for variable energy Usage: SetEnergy [X-ray energy in eV]

*… depending on availability of energy calibrations

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.

Energy: The energy has to be set via the *SetThreshold* command to get an optimized flat field since the flat field is energy and also energy to threshold dependent. If the *SetThreshold* command is used and the threshold is not set to 50% of the incoming energy, it is necessary to specify the correct energy for an adequate flat field.

Command	Description
SetThreshold [energy energy_value] gain threshold *	Allows to set the threshold energy as well the incoming energy for an optimized flat fields Usage: *SetThreshold* [energy energy_value] [[gain] threshold] 1. if parameters are omitted, the current settings are shown 2. if energy (with an energy_value) is given, energy_value is recorded as incident energy, else incident energy is assumed to be 2x the threshold energy. This is important for the correct usage of the flat field, since it needs both parameters for correct application. 3. gain is only included to obtain compatibility with earlier version and has no effect. 4. threshold is in eV 5. this command builds a script in "/tmp/setthreshold.cmd" and then loads it. 6. 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 For example: setthreshold 7400 setthreshold energy 14200 7500 If gain settings are used, they are ignored. The following to examples are equivalent to the two examples above and do exactly the same: setthreshold midG 7400 setthreshold energy 14200 lowG 7500

*…depending on availability of energy calibrations

9 Bad Pixel Mask and Module Gaps

9.1 Using the Bad Pixel Mask

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.

The bad pixel mask created during factory tests is automatically applied to each image that you take. The bad pixel mask is loaded every time the detector is trimmed (e.g. SetCu).
To NOT apply this bad pixel mask automatically, you have to use the command *LdBadPixMap* with a 0 as argument after a trimming command (e.g. SetCu). For further details please see Camserver command "*LdBadPixMap*" in section 14.

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: ***pixlfill*** 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 ***pixlfill*** 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

On the PILATUS 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 14).
During startup the *GapFill* command is set to -1. This causes the gaps to be flagged with -1 in the final image. To change the default behavior please remove the line which contains *Gapfill* command from the Camserver startup file (*/home/det/p2_det/* config/cam_data/*p2_det.set*).

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 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; Omega; Omega_increment; Oscillation_axis; N_oscillations; Start_position; Position_increment; Shutter_time;
and CBF_template_file.

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, Omega by Omega_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 TIF images are to be written.
'*MXsettings* CBF_template_file 0' can be used to turn off this setting.

More information can be found in the cbf specification available from www.dectris.com.

11 Flat Field Image

11.1 Using the Flat Field Correction Image in Camserver

Flat field corrections files recorded during factory calibration are automatically applied to all acquired images.

This variations in the sensitivity are a function of beam energy and energy to threshold ratio. To correct these sensitivity variations within and between different modules the X-ray energy and the threshold energy settings are set automatically via the trim commands (SetCu, SetMo, SetFe, SetAg, SetCr, andSetEnergy).
If the *SetThreshold* command is used additional to the threshold energy also the used X-ray energy has to be specified to obtain an optimized flat field (see *SetThreshold* command in section 7 and section 14 for more details).

To turn off the flat field correction you can use the Camserver command *LdFlatField 0*. To turn it on again it is best to restart Camserver.

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*.

Command **or** ***<u>Macro</u>***	**Description**
menu	Shows all commands It is divided in 5 parts: • **Reserved Words:** (do not use as variables) • **External Procedures:** • **User Commands in current directory:** All TVX commands; use *man command* to get a detailed description) • **Defined variables & strings:** This are macros which are created during startup (e.g. in default.gl); *use show macroname* to see more details • **User Variables:** assigned variables
help command *-or-* ***man command***	Displays the help text for the *command*. Help *help* is a good way to start.
show <u>macroname</u>	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 Do not use this in Camserver
<u>rbd</u>	Read Back Detector. Self-test of the digital part of the detector. Sends a digital pattern to each pixel, reads it out and displays the image.

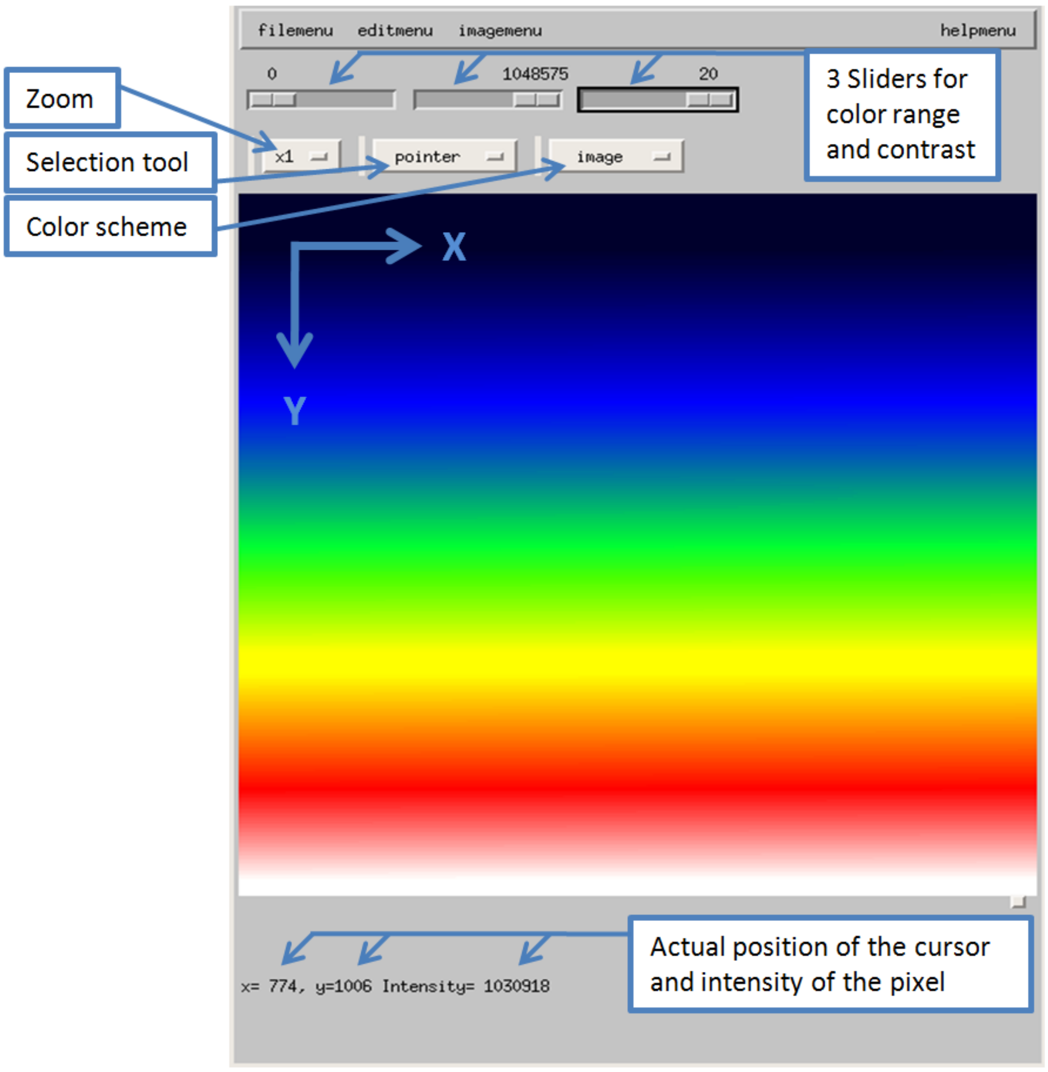
DECTRIS®

	ℹ Use this command always after a startup. Every pixel shows 1000 counts.
calibdet	Self-test of the detector. Sends 100 calibrate pulses to the analog part of each pixel, reads back the recorded values as an image and displays the result. ℹ Use this command always after a startup. Every pixel shows 100 counts.
setdac	Sets all Digital Analog Converters (DAC) to the predefined values.
imagepath *path*	Image Path 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 autoname to the new path.
grafpath *path*	Display or change the default path for graphs. The keyword 'grafpath' can be used in expressions as [grafpath]
expose *exposure time* (in seconds)	expose 1: makes an image with an exposure time of 1 sec. Shortest exposure time: >0.000 001s. Shows the exposed image and its name immediately after completion.
exposem *exposure time*	continuous camera mode without saving images. Takes images until any key is pressed. The last image is stored in temp.tif
disp *filename*	Display an image (see Section12.1). Opens up to 3 windows with successive invocations.
disp1 *filename*	Displays an image reusing the last window. Useful in loops.
graf *fn1[fn2[fn3]]*	Graph up to 3 graphs in a window
Convert *[S][D][type]*	Change the image data type. S…source image, D…destination image. Type… Char, Short, Int, Float
define	DEFINE name="instruction1; instruction2;....." Defines user symbol name and value. E.g. define tpict="zpict; move imt=im3" defines symbol tpict as a comination of ‘zpict’, and the built-in move instruction.
CaptureIM *filename*	Capture the default image to filename captures a displayed image (and its zoom) as a .ppm (portable pixmap) file, including coloration and contrast adjustments.
CaptureGR *filename*	Capture the default graph to filename Captures a displayed graph (and its zoom) as a .ppm

	(portable pixmap) file.
connect [*ip_address]*	Connect the socket connection from TVX to the Camserver at [*ip_address].* The IP is not necessary if TVX is running on the same computer.
disconnect	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.


filemenu
editmenu
imagemenu
helpmenu
0
1048575
20
3 Sliders for color range and contrast
Zoom
Selection tool
Color scheme
x1
pointer
image
X
Y
Actual position of the cursor and intensity of the pixel

```
x= 774, y=1006 Intensity= 1030918
```

Figure 10. Image display from TVX

Display tool	**Description**
(sliders)	Define the color and the contrast of the image. For every value of a pixel a color from a lookup table will be displayed. 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. 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 *disp* command (use *man disp*).
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.
Selection tools	
pointer	normal pointer
annulus	Allows analysis of circular areas. The sizes of the circles can be adjusted with the mouse or directly by the setting the values in the image window.
box	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.
butterfly	Allows analysis of special shaped areas. 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.
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)
Display mode	
grays	color lookup table with gray scale.
spectral	color lookup table with a spectral distribution (blue and 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. 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:
	
	(# grays)*((value - min)/(max - min))*(Scal/15)
	# 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.
	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. Useful for crystallographic images.

Several test images and graphs are included in the system.

Try the following:

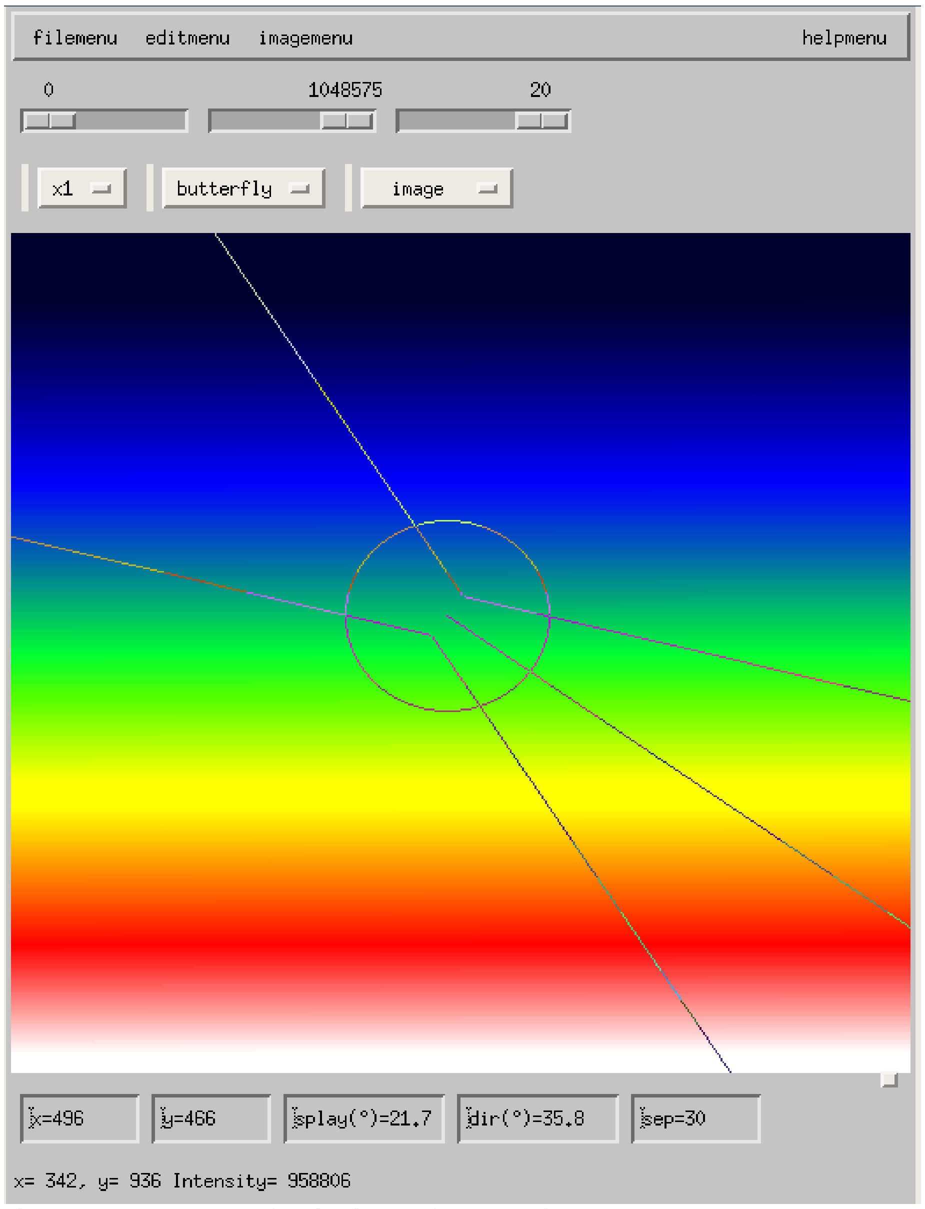
imagepath examples
disp gray20bit.tif

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.



x=496 y=466 splay(°)=21.7 dir(°)=35.8 sep=30

x= 342, y= 936 Intensity= 958806

Figure 11. 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**
move *fn1=fn2*	The basic image manipulation command. In the simple form shown, this copies an image to a new name or directory. However, *fn2* can be any arithmetic expression of images and constants.
Integrate	integrate the pixels selected by the current selection tool - box, butterfly (includes straight-line case), or spot (annulus tool) - and show the resulting graph. Usage: integrate [IM] [graph_name] 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.
histogram *lo hi int*	Histogram of the pixels selected by the box tool on the image. Alternatively, specify the image and region-of-interest on the command line. 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. See also 'histset'

box	Print statistics from the current box selection tool on the image. Alternatively, specifiy image name and box coordinates on the command line. Usage: box [IM] [x1 y1 x2 y2] If IM is not given, the default IM is used. [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 & ycen (centroid), box_x1, box_x2, box_y1 & box_y2 (corners of box).
boxall	Print statistics from the whole (current) image.
format *n1[.n2]*	Control the number of digits to be printed (n1) or the number of decimal places (n2).
deleteallobjs	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 *tvxrc,* 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.
deleteobj *filename*	Deletes the specified object from the TVX memory. The file on disk is untouched.
maskimg	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
mkmask	Make a mask from an image between two limits, inclusively. Usage: mkmask [IM [IMout]] low high 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. 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.
maskimg	Declare, inquire about, or turn off the current mask image. Usage: maskimg [im] -or- maskimg 0 If present, the mask is used to blank out bad pixels in statistical routines such as box, integrate, spot & histogram. Zeros in the mask are excluded from the analysis, non-zeroes are included. With no argument, displays the current mask image name, if any. With numeric argument (e.g. 0), turn off the mask image.
ldm	Ldm is short for load mask (a macro) and uses the maskimg command and the factory produced mask (stored in $HOME/p2_det/correct/goodpix_mask.tif).
pixlfill *[IM] value*	Set pixels in *IM* to *value* 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.

If the command "deleteallobjs" is used after you have loaded a mask image your maskimg 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**
define *name="string"* ***define*** *name=value*	Define a name (value or command) which can be used in the current session. They are not saved when TVX closes.
save *"myfile.gl"*	Saves the currently defined commands in *myfile.gl* 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.
get *"myfile.gl"*	Load the definitions from *myfile.gl*, 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. *setdac, rbd, calibdet*. 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 counting mechanism introduces a short dead-time after each hit, which becomes significant for rates above a few 10^5 counts/s/pixel. The resulting loss in the number of counted photons is corrected by applying a corresponding rate correction. This depends on the threshold and time settings. If the uncertainty of the correction becomes too large, the correction is cut off at a "saturation" value which is also printed in the header. The detector is shipped with an optimized correction automatically set by any trim command (section 8).

3) Flat Field
Sensitivity variations within and between different modules will be corrected by an appropriate flat field. The flat field correction is loaded automatically when trimming the detector (see section 11 for more details).

4) 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 (unit) [default values]	Description	Socket connection return code	Socket connection return text
Exposure	file_base_name + file extension[1]	Make an exposure ExpTime, ExpPeriod, ImgPath and NImages have to be set beforehand to the desired values. The image is written to the specified file_base_name[1] 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 >1), an image number is inserted before the extension[1].	at the start: 15 at the end: 7	at start: starting xxx second background: <date & time> at the end: full path name of last image
ExtTrigger	file_base_name + file extension [1]	Arm the detector for an exposure or an exposure series started by one external trigger. 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. 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.	at the start: 15 at the end: 7	at start: Starting externally triggered exposure(s): <date & time> at the end: full path name of last image

DECTRIS®

ExtMtrigger	file_base_name + file extension [1]	Arm the detector for an exposure series where each exposure is started by an external trigger. Set timing parameters by the commands ExpTime and ExpPeriod before execution. Each exposure is trigged 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 7.3.4). 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.	at the start: 15 at the end: 7	at start: Starting externally multi-triggered exposure(s): <date & time> at the end: full path name of last image
ExtEnable	file_base_name + file extension [1]	Make an exposure or an exposure series using an external gate signal. Each exposure is started when the signal changes to high and is finished when the signal changes to low again. 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.	at the start: 15 at the end: 7	at start: Starting externally enabled exposure(s): <date & time> at the end: full path name of last image
ExpTime	Time (s) [1.0]	Set the exposure time; time < 60days.	15	Exposure time set to: xxx sec.
ExpPeriod	Time (s) [1.05]	Set the exposure period The exposure period must be longer than or equal 50 ms. The minimum time between exposure time and exposer period must be 7 ms. The frame time is ExpPeriod*(NExpFrame-1) + ExpTime Time < 60 days	15	Exposure period set to: xxx sec

ImgPath	Path [/home/det/ p2_det/images]	Change the image path 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.	10	the path
NImages	Number (#) [1]	Set the number of images for an automatic sequence Maximum number of images is 65535	15	N images set to: nn
Delay	Time (s) [0]	Set the delay from the external trigger until start of the first exposure The time must be shorter than 64 s The delay is reset to 0 for ordinary exposures and external enable	15	Delay time set to: n.n sec
NExpFrame	Number (#) [1]	Set the number of exposures to accumulate per frame/read out. This is a method summing up images within the detector chip. A value >1 is a waste of X-rays except when using external enable or external multi-trigger to synchronously capture an event. If nexpframe>1, the reported 'measured exposure time" applies to the last exposure only. The maximum possible number is 2^32-1.	15	Exposures per frame set to: nn

MXsettings	Mxparameters (text) [none]	Set crystallographic parameters reported in the image header mxsettings [parm_name value] [parm_name value] ... Possible: 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, Omega_increment, Oscillation_axis, N_oscillations, Start_position, Position_increment, Shutter_time, CBF_template_file Not settable with mx_settings, but provided to templates from detector settings: Timestamp, Exposure_period, Exposure_time, Count_cutoff, Compression_type, X_dimension, Y_dimension	15	None set or current settings
SetCu		Command to trim the detector optimal for usage with Copper radiation	15	OK /tmp/setthreshold.cmd
SetMo		Command to trim the detector optimal for usage with Molybdenum radiation	15	OK /tmp/setthreshold.cmd
SetCr		Command to trim the detector optimal for usage with Chromium radiation	15	OK /tmp/setthreshold.cmd
SetFe		Command to trim the detector optimal for usage with Iron radiation	15	OK /tmp/setthreshold.cmd
SetAg		Command to trim the detector optimal for usage with Silver radiation	15	OK /tmp/setthreshold.cmd

SetThreshold	Threshold parameters (text) [none]	Allows to set the threshold energy as well the incoming energy for an optimized flat fields Usage: SetThreshold [energy energy_value] [[gain] threshold] 1. if parameters are omitted, the current settings are shown 2. if energy (with an energy_value) is given, energy_value is recorded as incident energy, else incident energy is assumed to be 2x the threshold energy. This is important for the correct usage of the flat field, since it needs both parameters for correct application. 3. gain is only included to obtain compatibility with earlier version and has no effect. 4. threshold is in eV 5. this command builds a script in "/tmp/setthreshold.cmd" and then loads it. 6. 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 For example: setthreshold 7400 setthreshold energy 14200 7500 If gain settings are used, they are ignored. The following to examples are equivalent to the two examples above and do exactly the same: setthreshold midG 7400 setthreshold energy 14200 lowG 7500	15	Setting the threshold: pathname of file Without argument (once set): Settings: xxx gain; threshold: xxxx eV; vcmp: x.xxx V Trim file: /path/to/trim/file/abc.bin Without argument (never set): Threshold has not been set

SetEnergy	Energy (eV) [none]	Simplified method to set the threshold The requested energy is used to calculate appropriate threshold settings for the detector.	15	Setting the energy: pathname of file 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 Without argument (never set): Threshold has not been set
K		Interrupts an exposure or an exposure series	13 7	ERR kill full path name of last image
LdBadPixMap[2]	Filename (text) [none]	Load a mask image giving bad pixels to be flagged Filename must be a full pathname. 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 9000; the flag value is -2.	15	none or pathname
LdFlatField[2]	Filename (text) [none]	Load the flat-field correction file Filename must be a full pathname. File must be a 32-bit floating-point TIF image. 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

ReadoutTime		Show detector readout time in milliseconds.	15	Detector readout time [ms]: time in milliseconds
GapFill	Number (0,-1) [0]	Set the value to be used in pixels between modules. Number can only be 0 or -1. If n is omitted, the current value is printed.	15	Detector gap-fill is: nn
THread	Channel (n) [all]	Read one of the temperature and humidity sensors 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
SetAckInt	Number (#) [0]	Set the interval for acknowledgements over the socket. This causes Camserver to acknowledge every n-th image. 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.	15	none or current setting
ResetCam		Reset the camera.	15	none
DebTime	Time (s) [0]	Set the contact debounce time for external enable mode. If it 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.	15	Debounce time set to: n.n sec
HeaderString	String (text) [none]	Give a string to be included in the image header. The maximum length is 68 characters, no formatting permitted. Enclose the text in quotes to transmit non-alpha characters.	15	none

Exit, Quit		Close the socket connection.		none
Df		Show the number of 1 KB blocks available on ImgPath.	5	1K blocks available
ExpEnd		Return the filename with which the exposure ended.	6	full path name of last image
CamSetup		Report camera setup.	2	Camera definition: Camera name: Camera state: Target file: Time left: Last image: Master PID is: Controlling PID is: Exposure time: Last completed image: Shutter is:
Telemetry		Report camera telemetry.	18	Image format: and additional camera messages
Version		Print the TVX/Camserver version.	24	version
ShowPID		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 … Automatically set by the trimming command (e.g. SetCu).

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;
}
```