areaDetector: A module for EPICS area detector support

New developments

Mark Rivers

GeoSoilEnviroCARS, Advanced Photon Source

University of Chicago

areaDetector Talk Outline

- Motivation & goals for areaDetector module
- Overview of architecture
- Drivers for detectors & cameras
- Plugins for real-time processing
- Viewers and other clients
- What's new in R1-6

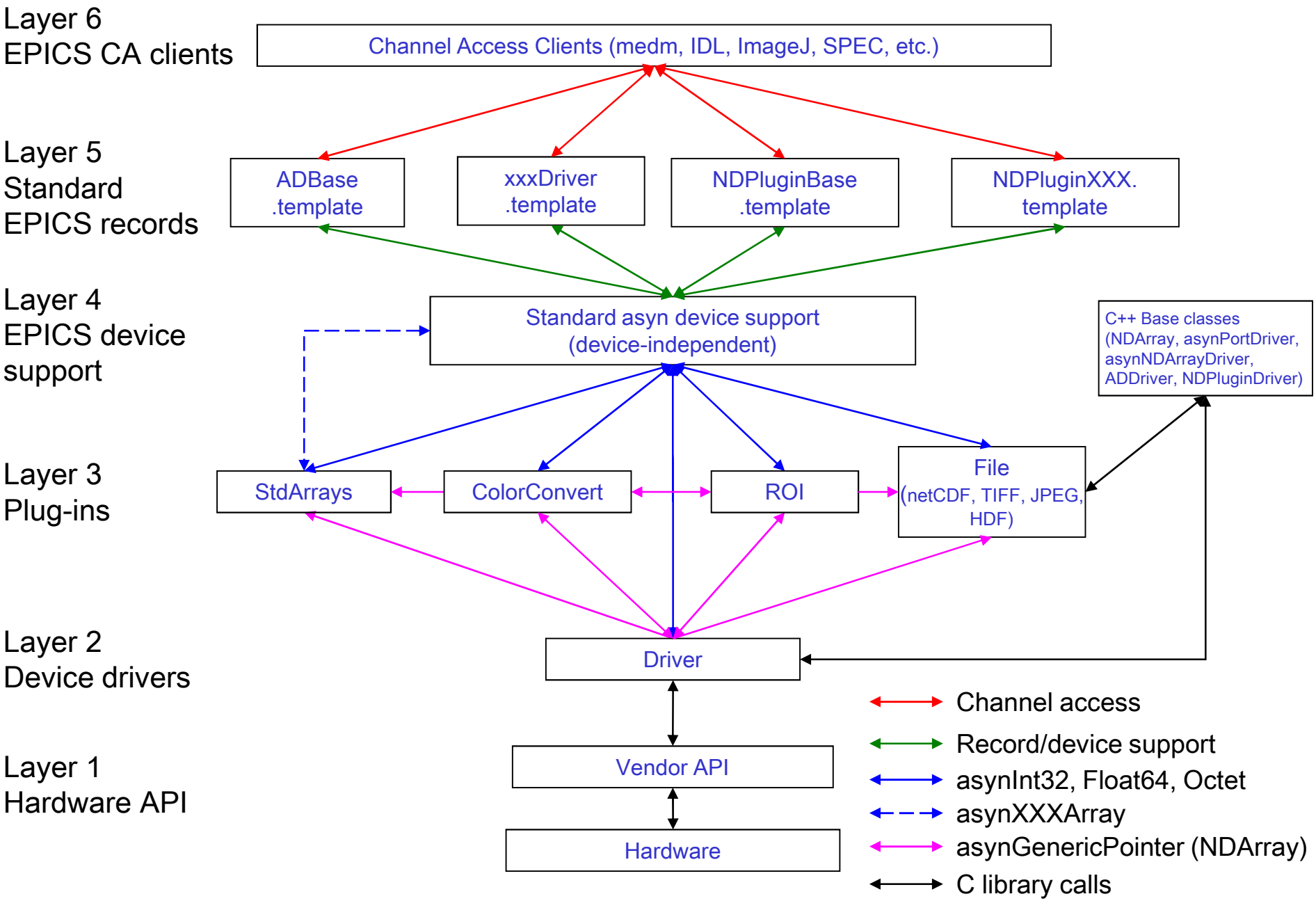
areaDetector - Implementation

- Drivers for many detectors popular at synchrotron beamlines
  - Handle detectors ranging from >500 frames/second to <1 frame/second
- Basic parameters for all detectors
  - E.g. exposure time, start acquisition, etc.
  - Allows generic clients to be used for many applications
- Easy to implement new detector
  - Single device-driver C++ file to write. EPICS independent.
- Easy to implement detector-specific features
  - Driver understands additional parameters beyond those in the basic set
- EPICS-independent at lower layers
  - Soleil is investigating using lower layers under Tango
- Middle-level plug-ins to add capability like regions-of-interest calculation, file saving, etc.
  - Device independent, work with all drivers
  - Below the EPICS layer for highest performance

areaDetector – Data structures

- NDArray
  - N-Dimensional array.
    - Everything is done in N-dimensions (up to 10), rather than 2. This is needed even for 2-D detectors to support color.
  - This is what plug-ins callbacks receive from device drivers.
- NDAttribute
  - Each NDArray has a list of associated attributes (metadata) that travel with the array through the processing pileline. Attributes can come from driver parameters or any EPICS PV; e.g. can store motor positions, temperature, ring current, etc. with each frame.
- NDArrayPool
  - Allocates NDArray objects from a freelist
  - Plugins access in readonly mode, increment reference count
  - Eliminates need to copy data when sending it to callbacks.

EPICS areaDetector Architecture



Detector drivers

- ADDriver
  - Base C++ class from which detector drivers derive. Handles details of EPICS interfaces, and other common functions.
- Simulation driver
  - Produces calculated images up to very high rates. Implements nearly all basic parameters, including color. Useful as a model for real detector drivers, and to test plugins and clients.
- Prosilica driver
  - Gigabit Ethernet cameras, mono and color
  - High resolution, high speed, e.g. 1360x1024 at 30 frames/second = 40MB/second.
- Firewire (IEEE-1396 DCAM)
  - Vendor-independent Firewire camera drivers for Linux and Windows
- Roper driver
  - Princeton Instruments and Photometrics cameras controlled via WinView

Detector drivers (continued)

- PVCAM driver
  - Princeton Instruments and Photometrics cameras controlled via PVCAM library
- Pilatus driver
  - Pilatus pixel-array detectors.
- marCCD driver
  - Rayonix (MAR-USA) CCD x-ray detectors
- ADSC driver
  - ADSC CCD detectors
- mar345 driver
  - marResearch mar345 online image plate
- Perkin-Elmer driver
  - Perkin-Elmer amorphous silicon detectors

Plugins

- Designed to perform real-time processing of data, running in the EPICS IOC (not over EPICS Channel Access)
- Receive NDArray data over callbacks from drivers or other plugins
- Plug-ins can execute in their own threads (non-blocking) or in callback thread (blocking)
  - If non-blocking then NDArray data is queued
    - Can drop images if queue is full
  - If executing in callback thread, no queuing, but slows device driver
- Allows
  - Enabling/disabling
  - Throttling rate (no more than 0.5 seconds, etc)
  - Changing data source for NDArray callbacks to another driver or plugin
- Some plugins are also sources of NDArray callbacks, as well as consumers.
  - Allows creating a data processing pipeline running at very high speed, each in a difference thread, and hence in multiple cores on modern CPUs.

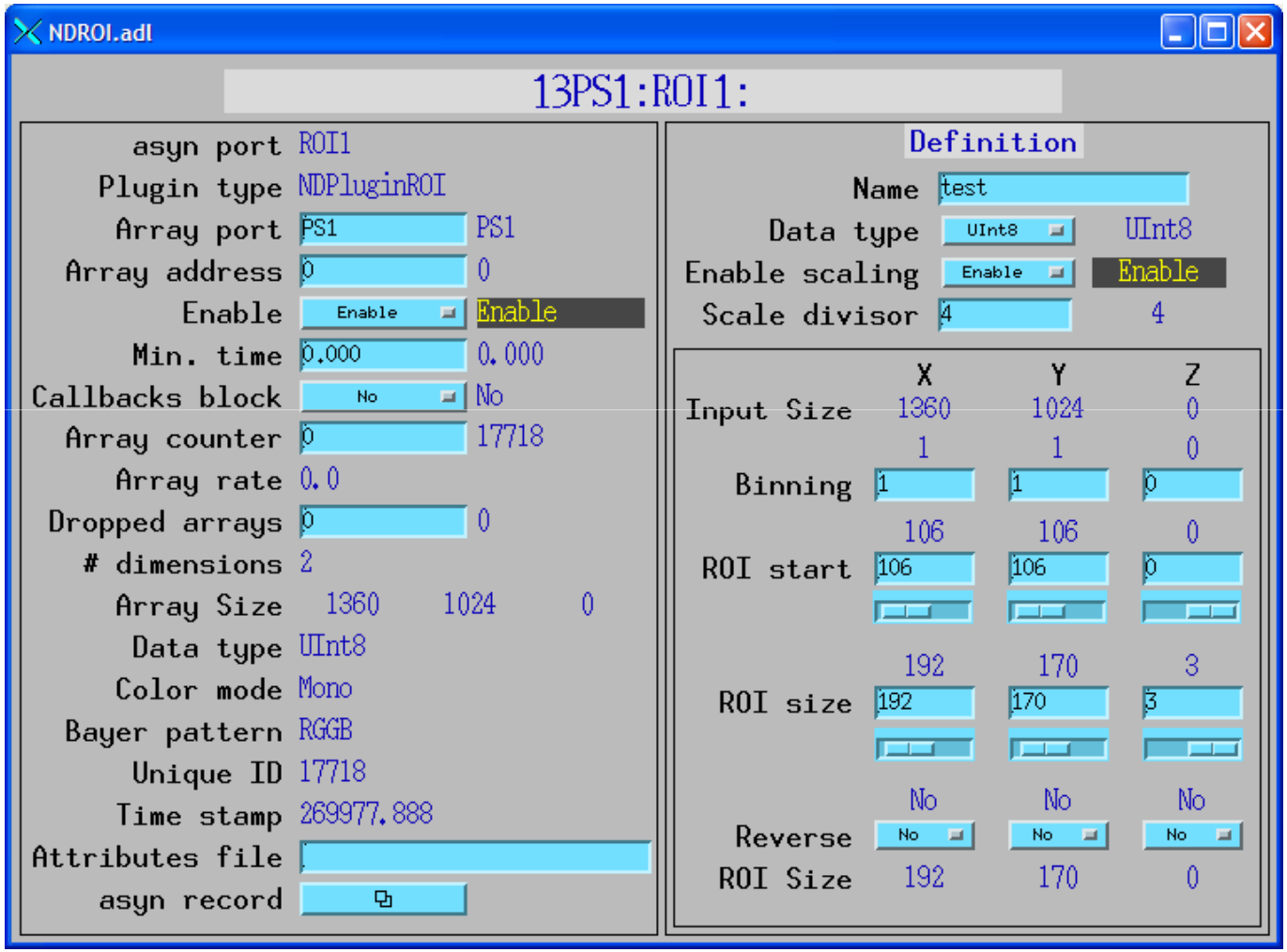
Plugins (continued)

- NDPlugInStdArrays
  - Receives arrays (images) from device drivers, converts to standard arrays, e.g. waveform records.
  - This plugin is what EPICS channel access viewers normally talk to.
- NDPluginROI
  - Performs region-of-interest calculations
  - Select a subregion. Optionally bin, reverse in either direction, convert data type.
  - R1-6
    - Previously the ROI plugin supported multiple ROIs, performed statistics calculations, and highlighted the ROIs.
    - New version is much simpler; it supports only a single ROI, and does not calculate statistics or do highlighting. Those functions have been moved to new plugins. One new function has been added, the ability to divide the array by a scale factor, which is useful for avoiding overflow when binning.
- NDPluginColorConvert
  - Convert from one color model to another (Bayer, RGB pixel, row or planar interleave)
  - R1-6
    - Added conversions from mono to RGB1, RGB2, and RGB3, and from RGB1, RGB2, and RGB3 to mono.
    - Previously this plugin only built on Linux and WIN32. Now it builds and does all conversions except Bayer on all architectures. Bayer conversion is restricted to Linux and WIN32.
- NDPluginMJPEG
  - MJPEG server that allows viewing images in a Web browser.

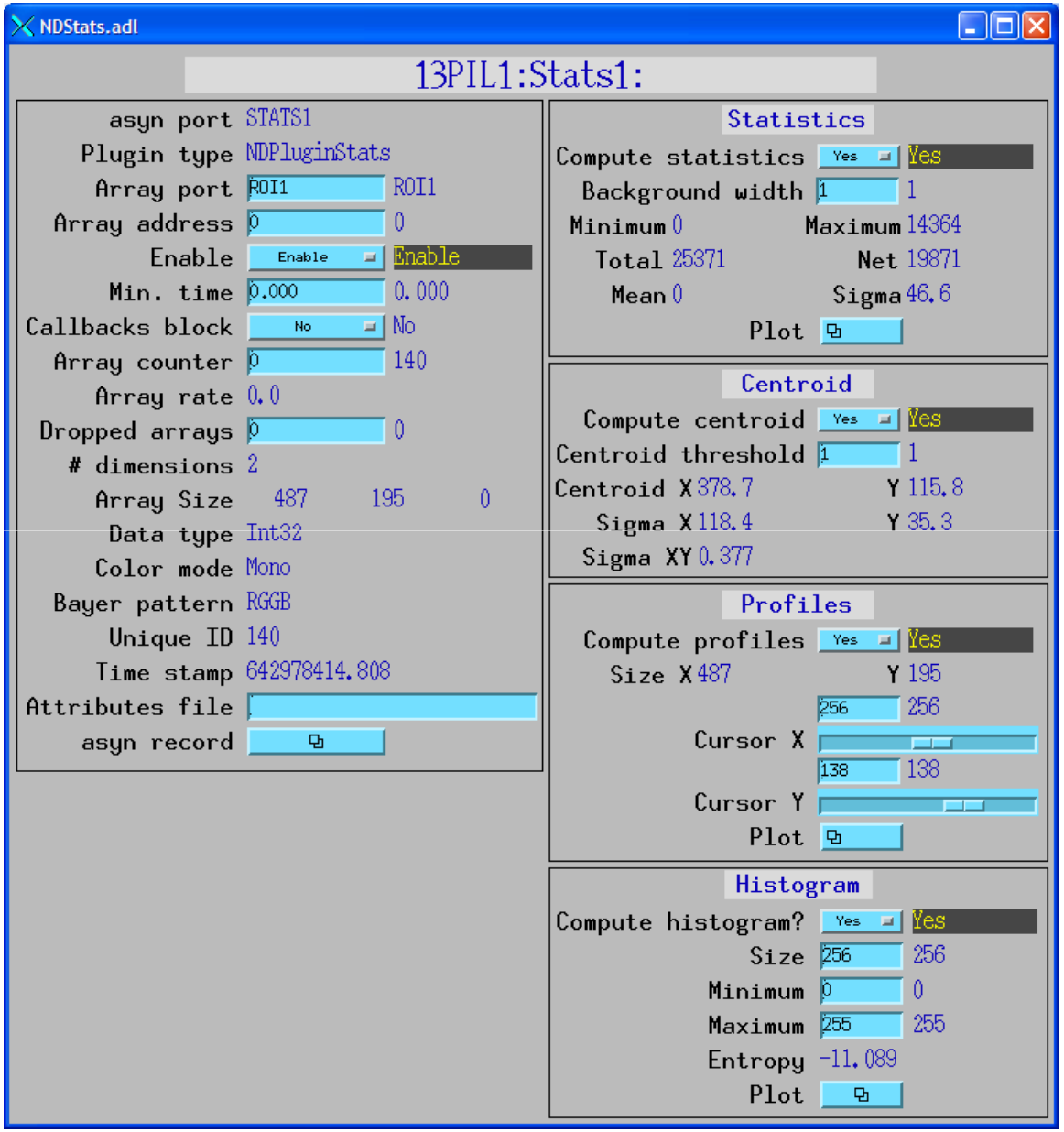
Plugins (R1-6)

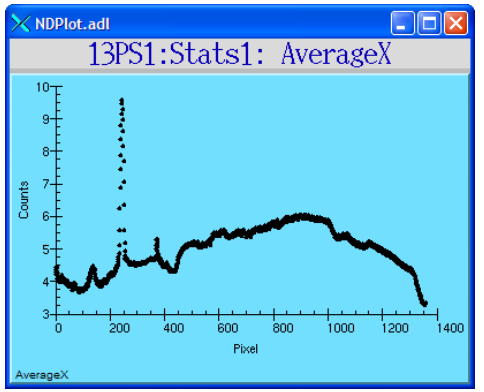
- New NDPluginStats plugin
  - Calculates statistics on an array
  - Replaces the statistics calculations that were previously performed in the ROI plugin.
  - Adds new statistics, including the centroid position and width.
  - Computes X and Y profiles, including average profiles, profiles at the centroid position, and profiles at a user-defined cursor position.
- New NDPluginProcess plugin
  - Does arithmetic processing on arrays
  - Background subtraction.
  - Flat field normalization.
  - Offset and scale.
  - Low and high clipping.
  - Recursive filtering in the time domain.
  - Conversion to a different output data type.
- New NDPluginOverlay plugin
  - Adds graphic overlays to an image.
  - Replaces the "Highligh ROIs" function that was previously provided in the ROI plugin.
  - Much more general, and can be used to display not only ROIs, but multiple cursors, user-defined boxes, etc.

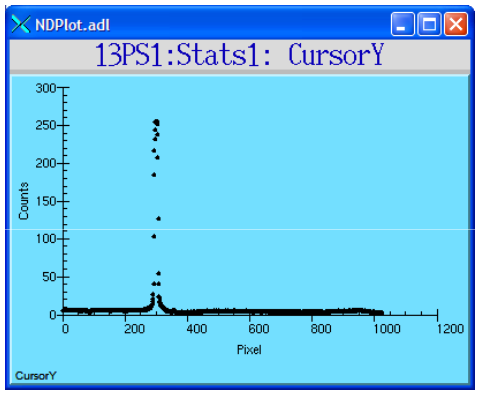
ROI plugin

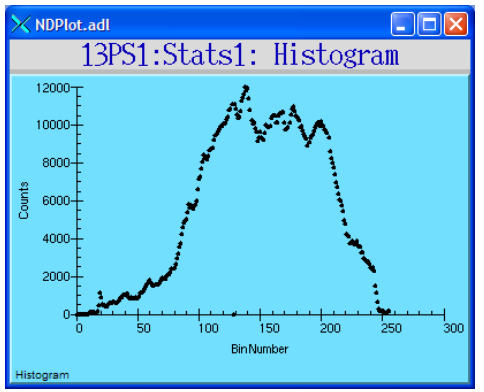

NDROI.adl
13PS1:ROI1:
asyn port ROI1
Plugin type NDPluginROI
Array port PS1 PS1
Array address 0 0
Enable Enable Enable
Min. time 0.000 0.000
Callbacks block No No
Array counter 0 17718
Array rate 0.0
Dropped arrays 0 0
dimensions 2
Array Size 1360 1024 0
Data type UInt8
Color mode Mono
Bayer pattern RGGB
Unique ID 17718
Time stamp 269977.888
Attributes file
asyn record
Definition
Name test
Data type UInt8 UInt8
Enable scaling Enable Enable
Scale divisor 4 4
X Y Z
Input Size 1360 1024 0
1 1 0
Binning 1 1 0
106 106 0
ROI start 106 106 0
192 170 3
ROI size 192 170 3
No No No
Reverse No No No
ROI Size 192 170 0

Statistics plugin

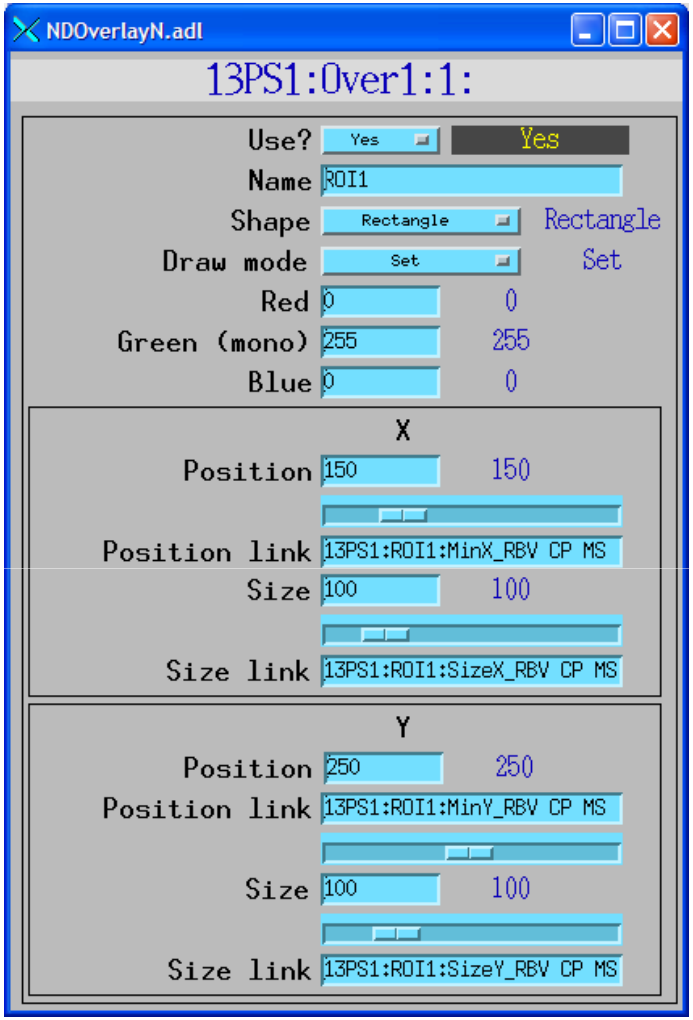








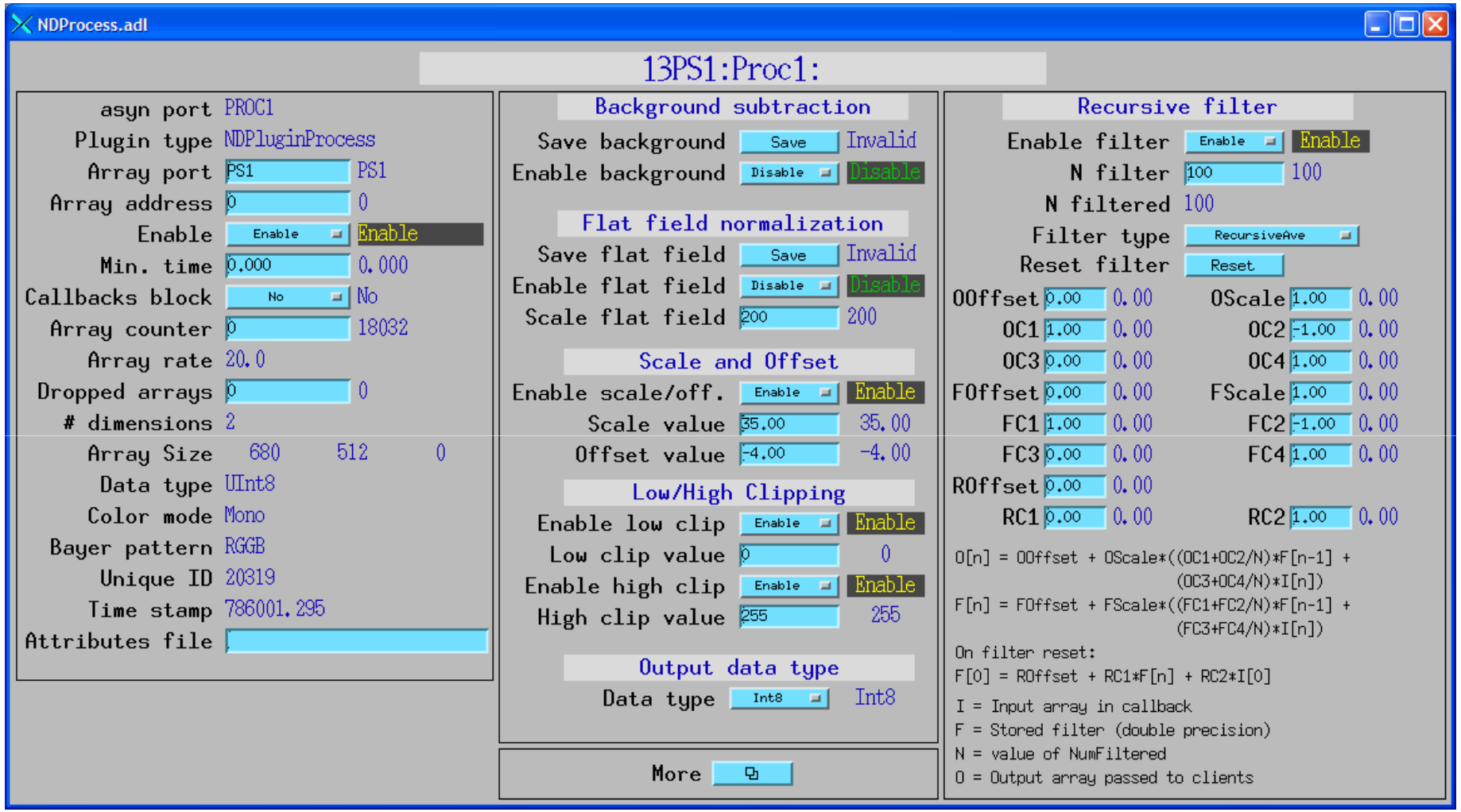
Overlay plugin



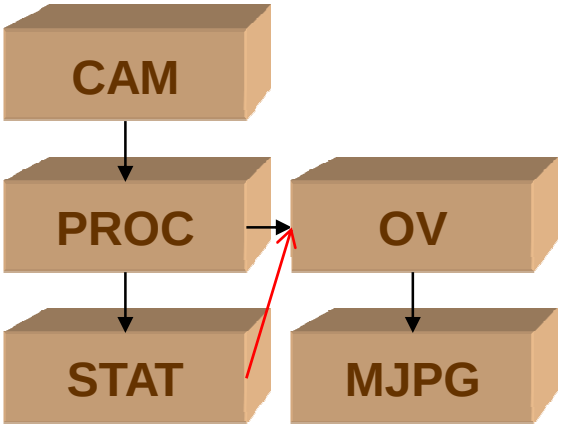
Centroid of laser pointer calculated by statistics plugin

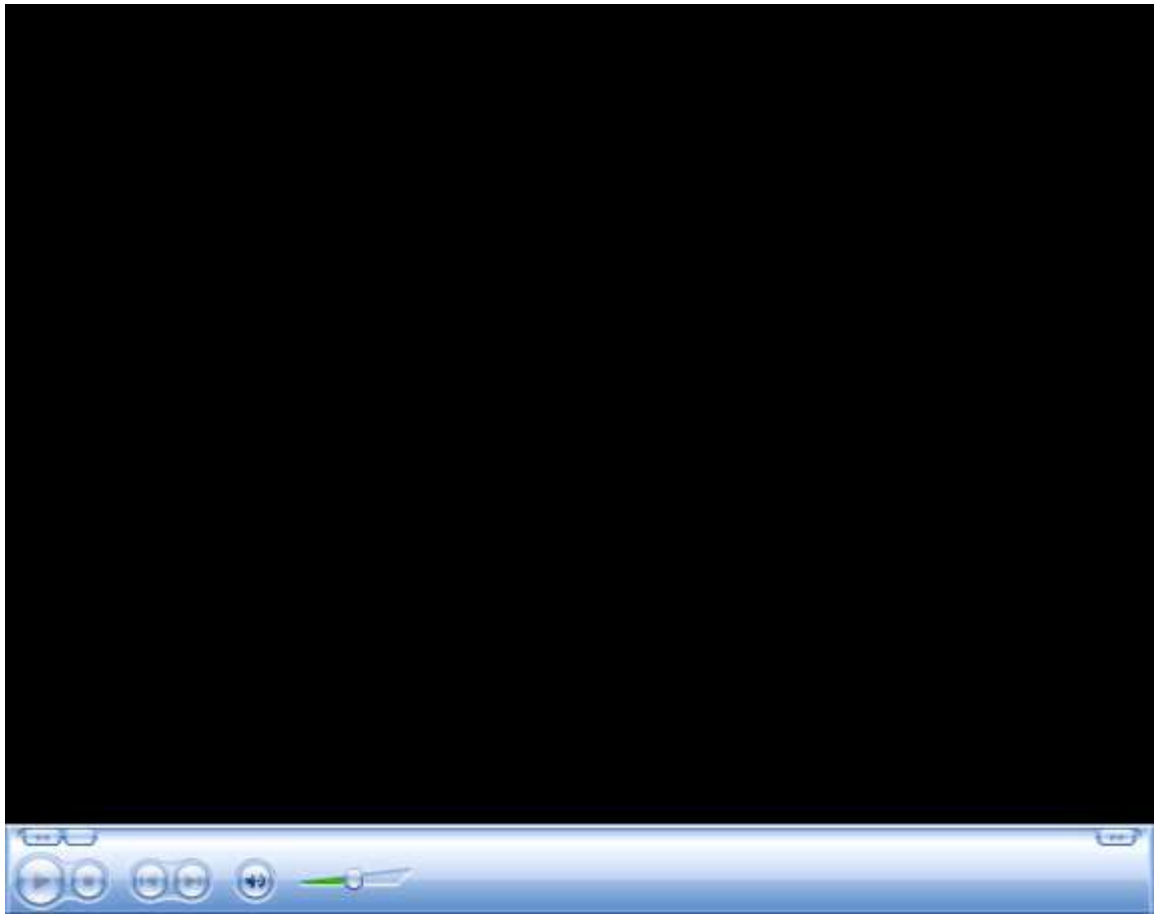
Cursor overlay X, Y position linked to centroid

Processing plugin



Example processing chain

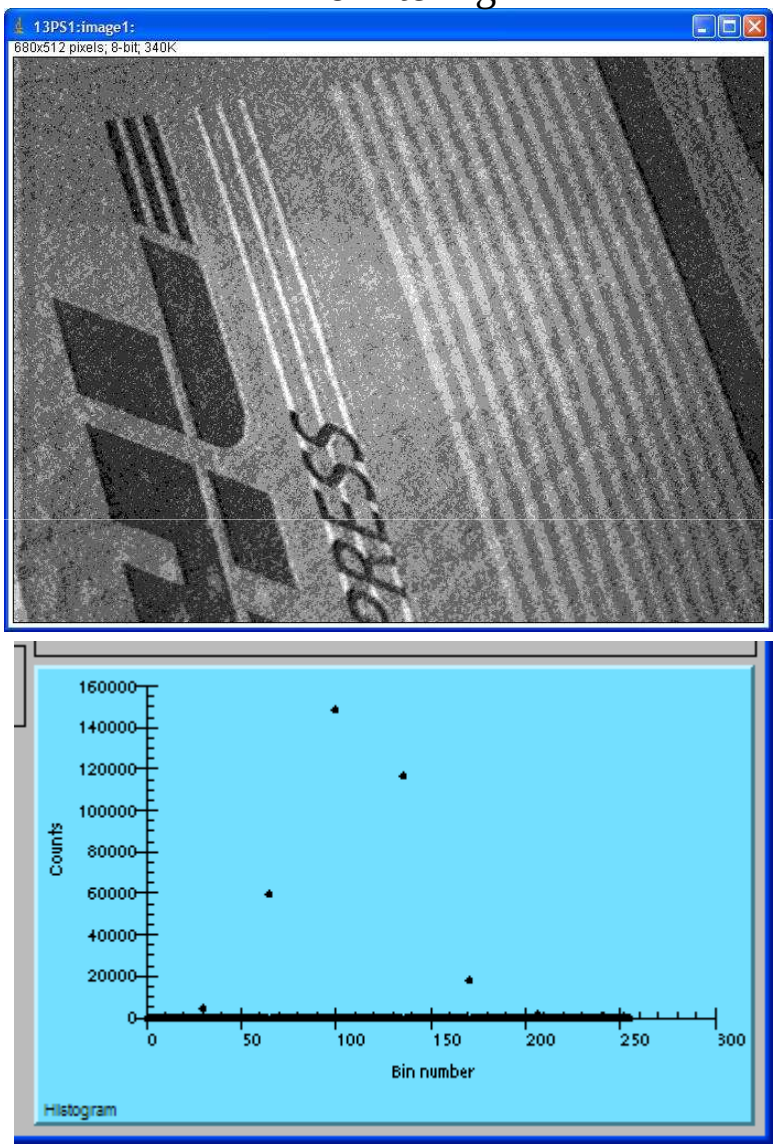




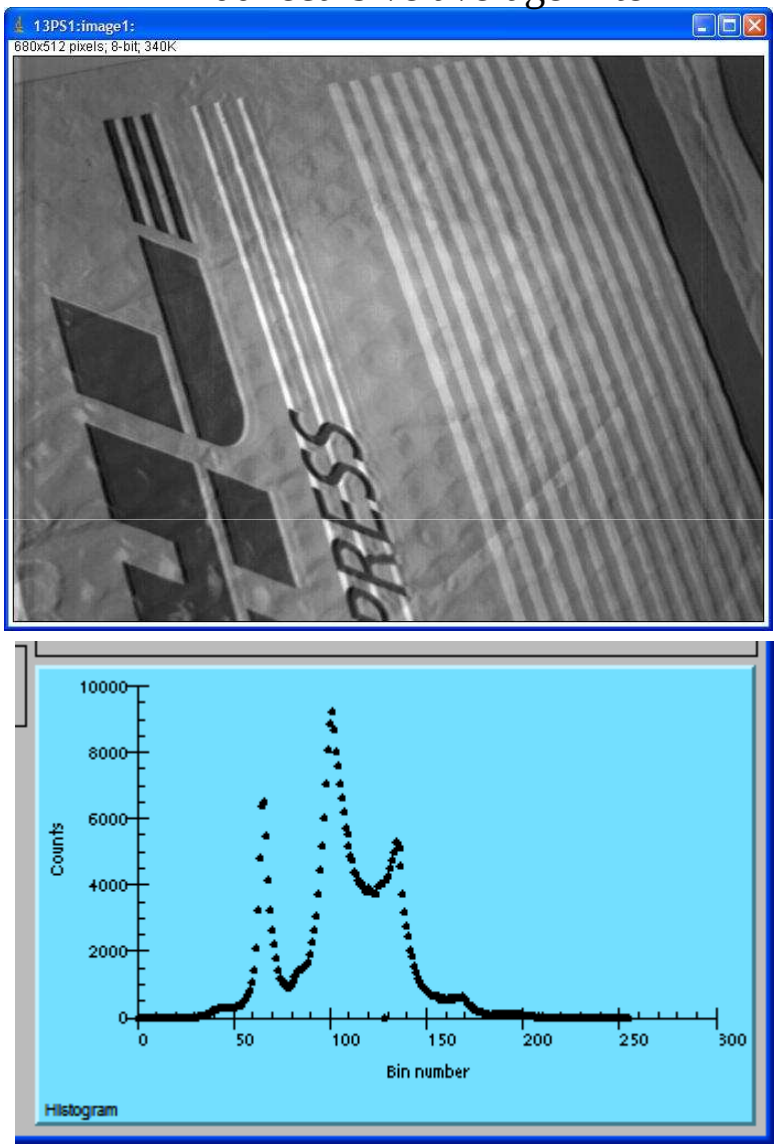
diamond

Processing plugin
30 microsec exposure time

No filtering



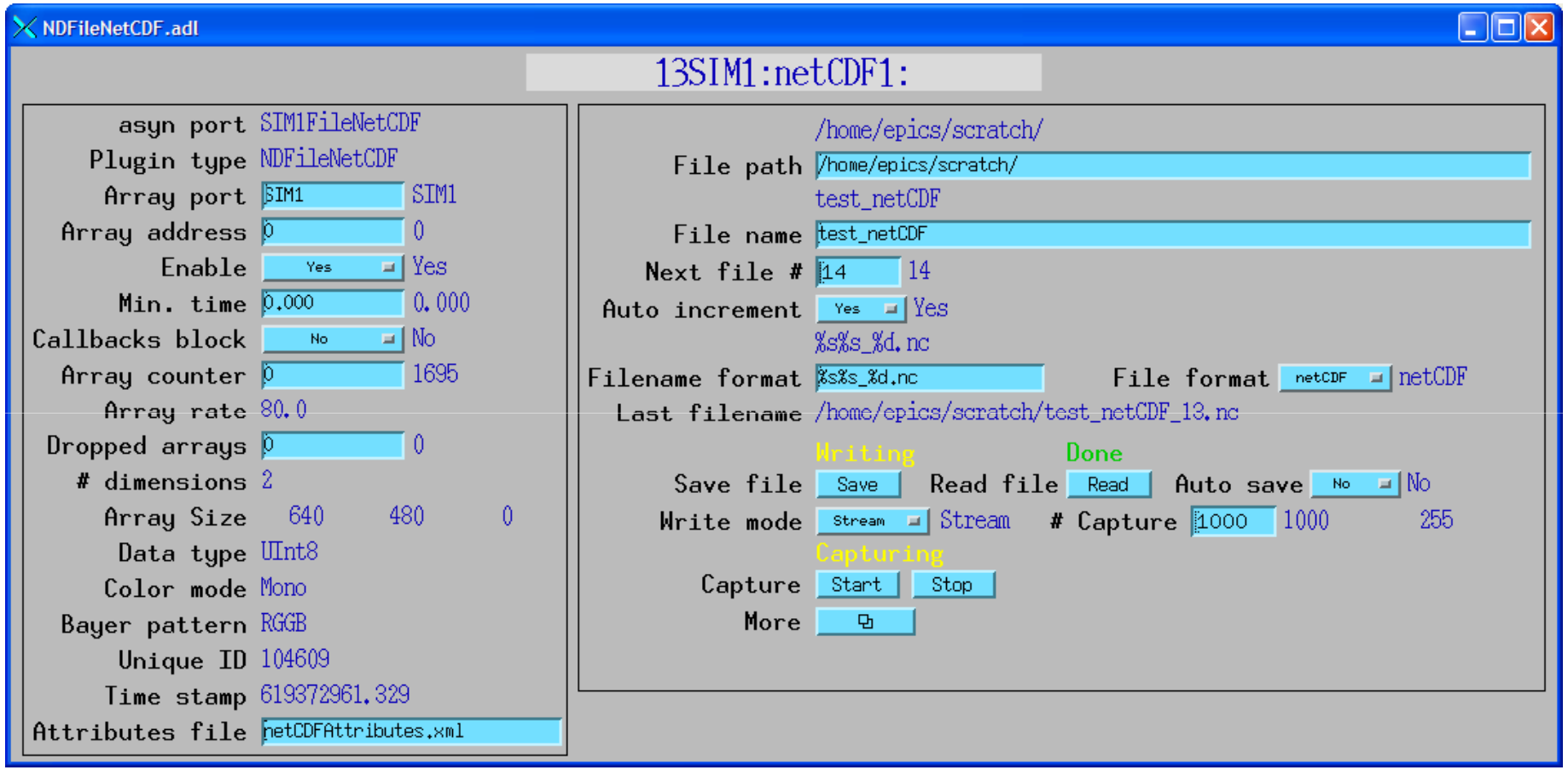
N=100 recursive average filter



Plugins: NDPluginFile

- Saves NDArrays to disk
- 3 modes:
  - Single array per disk file
  - Capture N arrays in memory, write to disk either multiple files or as a single large file (for file formats that support this.)
  - Stream arrays to a single large disk file
- File formats currently supported
  - TIFF
  - JPEG (with compression control)
  - netCDF (popular self-describing binary format, supported by Unidata at UCAR)
  - NeXus (standard file format for neutron and x-ray communities, based on HDF, which is another popular self-describing binary format, richer than netCDF).
- In addition to file saving plugins, many vendor libraries also support saving files (e.g. marCCD, mar345, Pilatus, etc.) and this is supported at the driver level.
- File saving plugin can be used instead of or in addition to vendor file saving
  - Can add additional metadata vendor does not support
  - Could write JPEGS for Web display every minute, etc.

NDPluginFile display

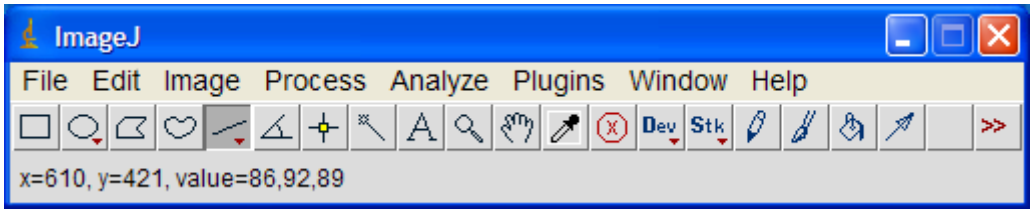


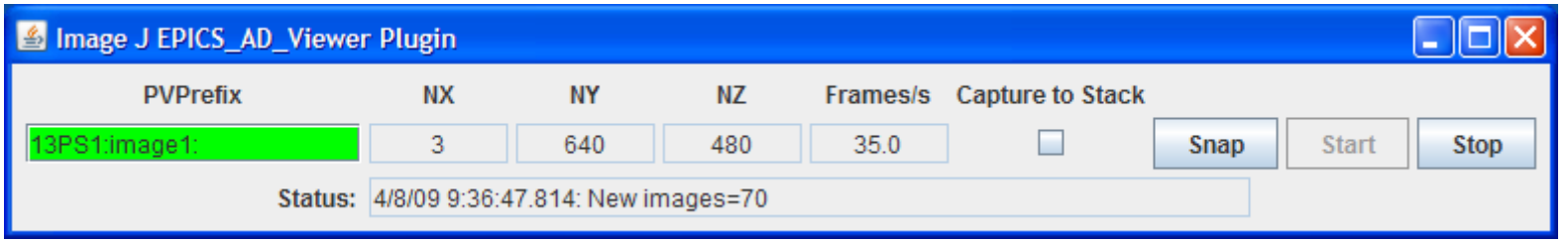
Example: streaming 80 frames/second of 640x480 video to netCDF file, no dropped frames.

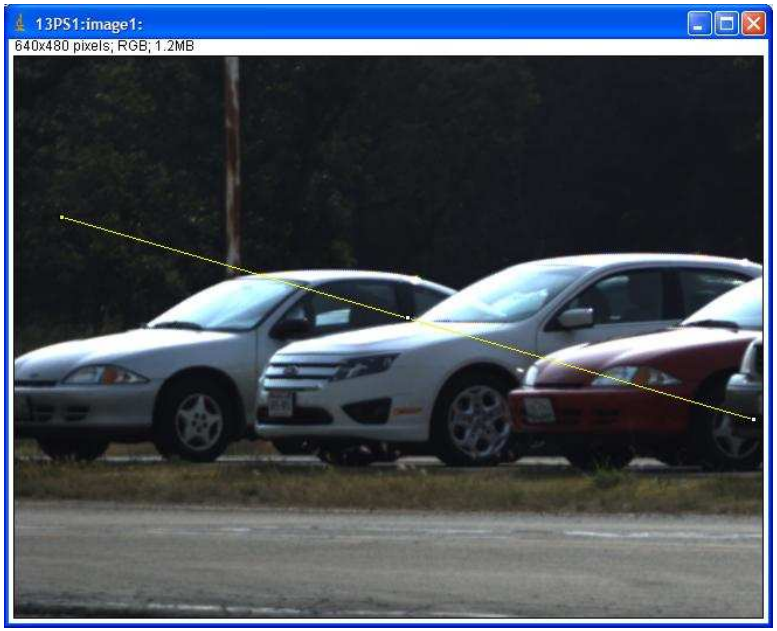
Viewers

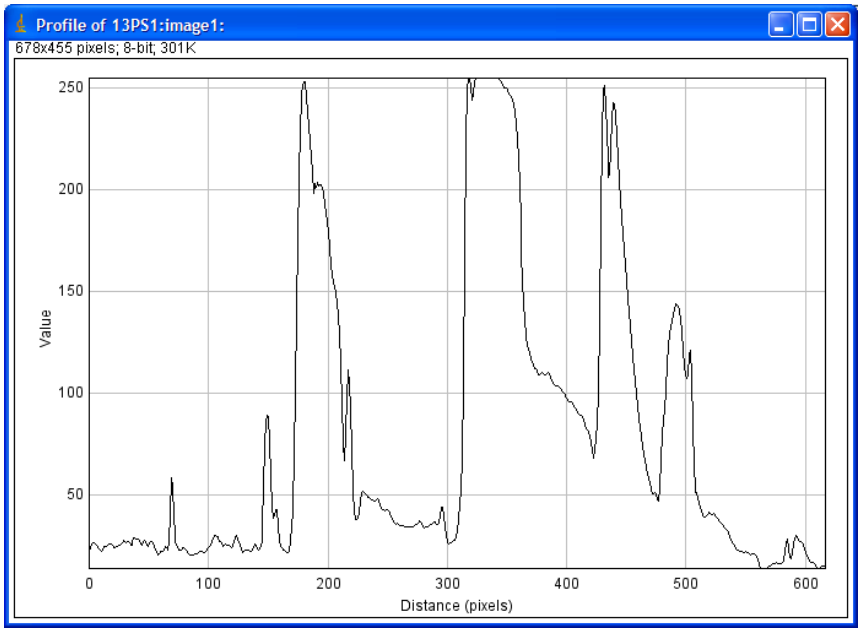
- areaDetector allows generic viewers to be written that receive images as EPICS waveform records over Channel Access
- Current viewers include:
  - ImageJ plugin EPICS_AD_Display. ImageJ is a very popular image analysis program, written in Java, derived from NIH Image.
  - IDL EPICS_AD_Display.
  - IDL areaVision (Stephen Mudie from AS)
  - MJPEG server allows image display in any Web browser

ImageJ Viewer



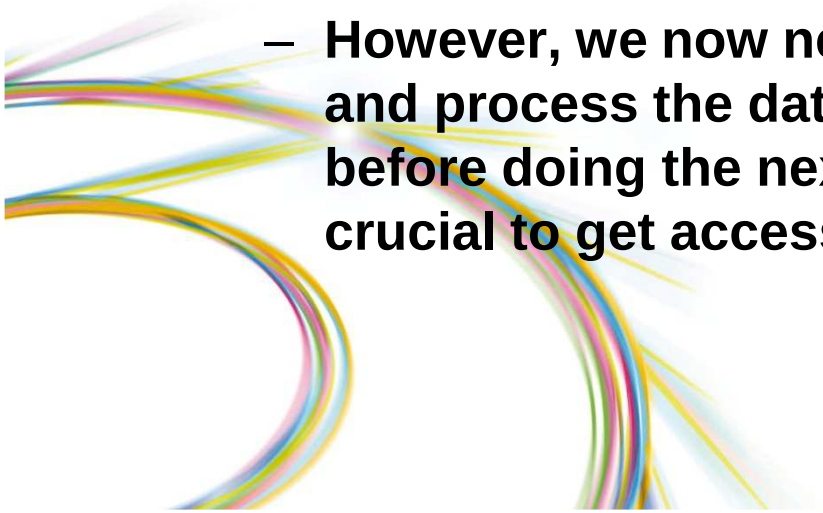






Area detector issues

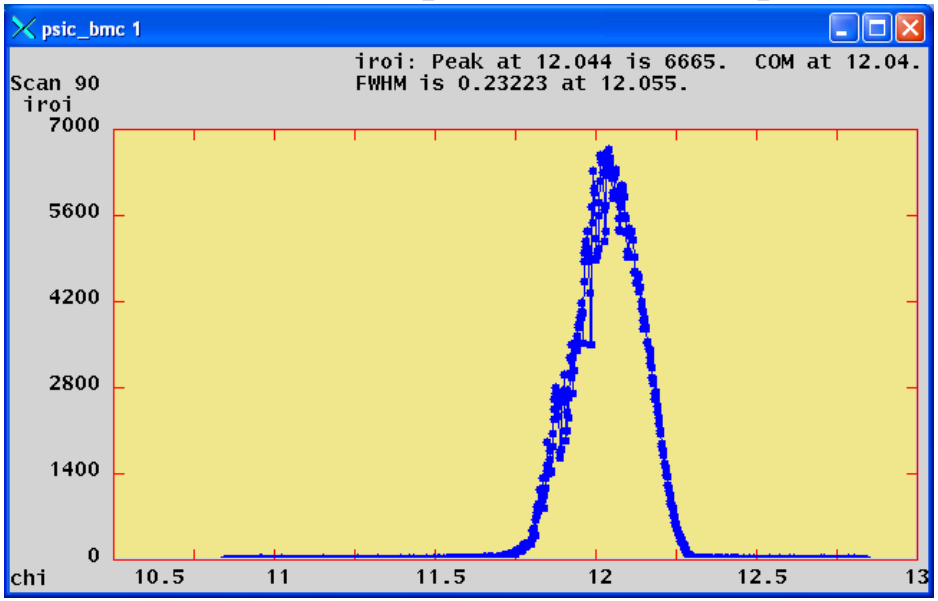
- **Vendors who don't give you access to the pixel data...**
  - **I16 often use their detector to find individual reflections from single-crystal samples. For these applications they are not that interested in keeping the images. They only need some quick measure of the number of counts and location of the brightest spot in a given region.**
  - **When stepping and exposing at 2-3 Hz we are able to write the files across the network, read them elsewhere and process them during the scan.**
  - **However, we now need to do fast scans at up to 200 Hz and process the data in real time, to get centres etc, before doing the next measurement. It is therefore crucial to get access to the pixel data.**



diamond

Performance Example with Pilatus driver

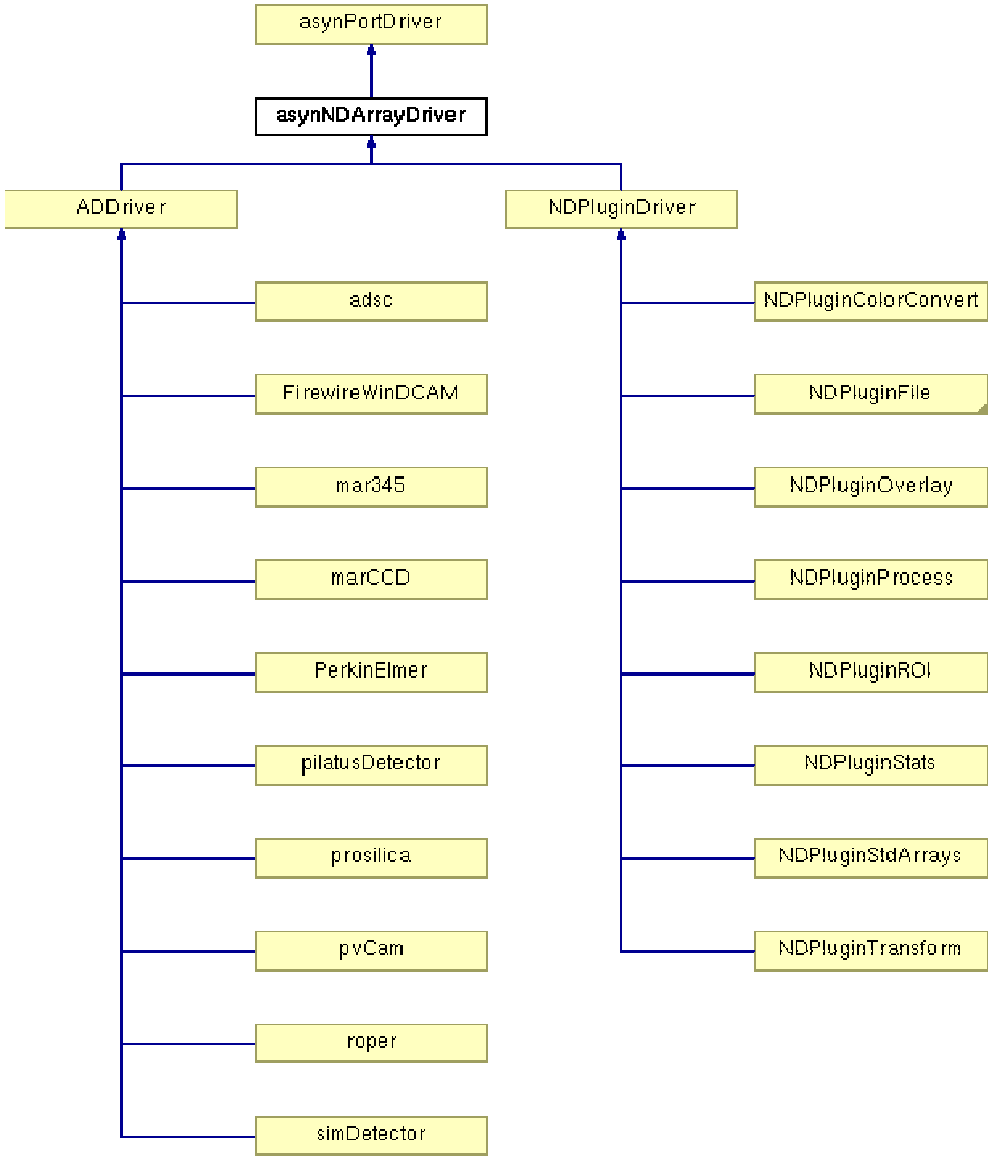
- SPEC used to collect 1000 points using trajectory scanning mode with the Newport XPS motor controller. Hardware trigger of Pilatus from XPS.
- Relative scan of the chi axis from -2 degrees to +2 degrees with 1000 points at .02 seconds/point
- Coordinated motion of the phi, kappa and omega axes.
- Theoretical time 20.0 second, actual time 20.8 seconds
- Includes time to save all 1000 images to disk (366 MB), Pilatus driver to read each file, correct bad pixels and flat field, compute ROIs, and post the ROIs and 1000 images to EPICS.



Conclusions

- Architecture works well, easily extended to new detector drivers, new plugins and new clients
- Base classes, asynPortDriver, asynNDArrayDriver, asynPluginDriver actually are generic, nothing “areaDetector” specific about them.
- They can be used to implement any N-dimension detector, e.g. the XIA xMAP (16 detectors x 2048 channels x 512 points in a scan line)
- Can get released source code and pre-built binaries (Linux, Windows, Cygwin) from our Web site:
  - http://cars.uchicago.edu/software/epics/areaDetector
- Can also get code on APS BCDA SVN repository
  - https://subversion.xor.aps.anl.gov/synApps/areaDetector/trunk

Internals – class hierarchy



Internals – class hierarchy

Look at Doxygen generated documentation for asynNDArrayDriver, ADDriver, pluginDriver, simDetector

Look at source code for ADDriver, simDetector, marCDD

Acknowledgments

- Brian Tieman (APS) Roper PVCAM driver
- John Hammonds (APS) Perkin-Elmer driver and NeXus file saving plugin
- Tim Madden (APS) initial version of ImageJ viewer
- Stephen Mudie (Australian Synchrotron) areaViewer IDL-based viewer
- Ulrik Pedersen and Tom Cobb (Diamond) MJPEG plugin, Linux Firewire driver
- Lewis Muir (U Chicago) ADSC CCD driver
- NSF-EAR and DOE-Geosciences for support of GSECARS where most of this work was done
- Thanks for your attention!!!