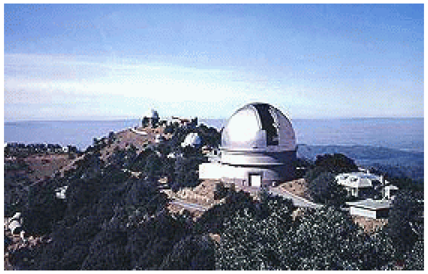
Extending and Customising DS9

a case study and notes from UCO/Lick Observatory

D. A. Clarke, S. L. Allen, W. Joye



Lick Observatory

Keck Observatory

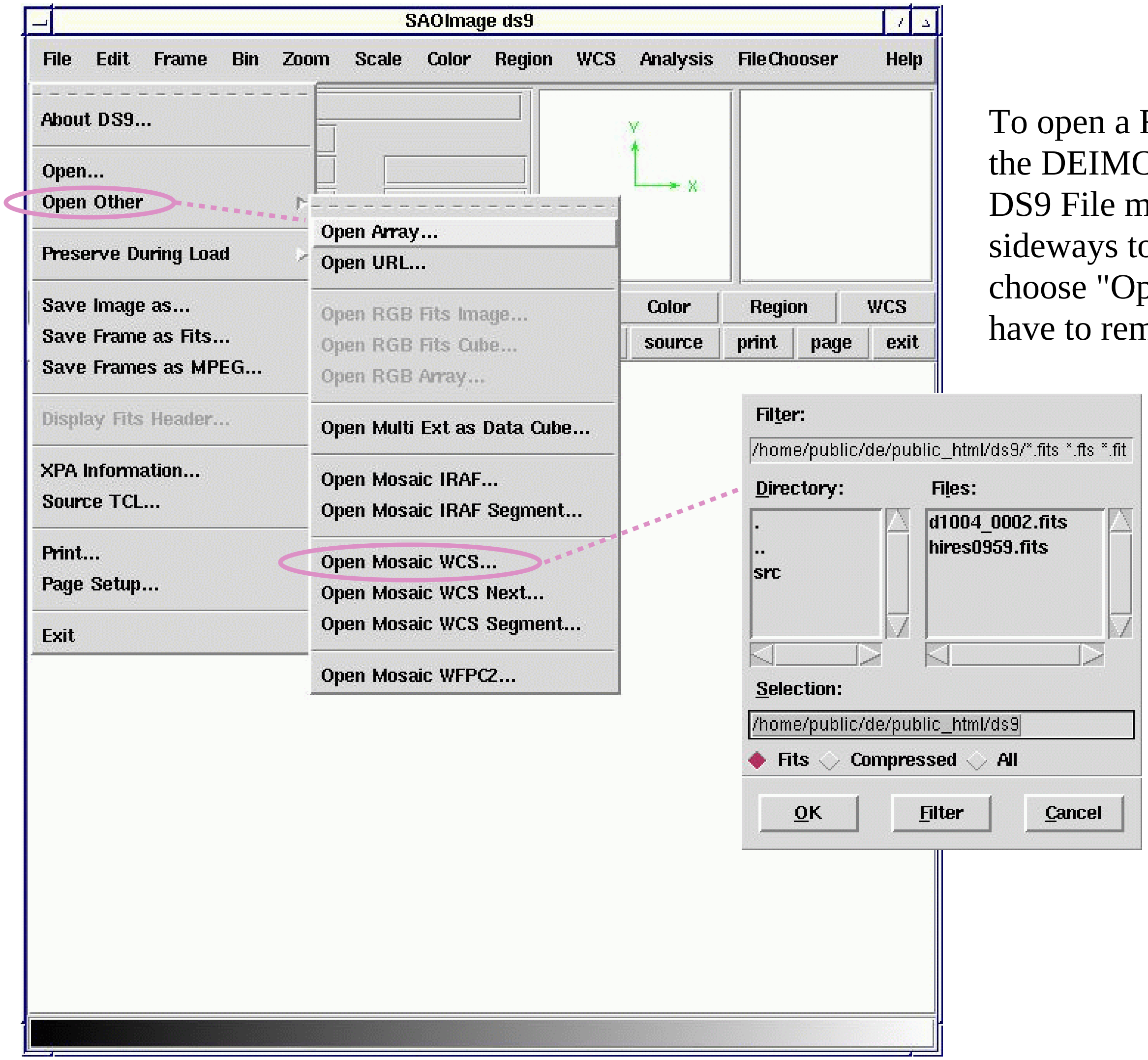
UCO/Lick Observatory adopted DS9 as the standard real-time and quick-look display for DEIMOS (commissioned Summer 2002) and subsequent instruments.

Lick software staff made several modifications and extensions to DS9, taking advantage of its open architecture and the ease of adding Tcl/Tk code to the DS9 core at runtime. We worked in close collaboration with DS9 author Bill Joye on both core modifications and external ’plug-ins’.

Our poster describes features added to DS9 (in the core or externally), and strategies for integrating DS9 with real time image acquisition. The Lick software team invites other DS9 users to form a community and discuss code-sharing, APIs and other DS9 issues.

Extending and Customising DS9

a more featureful file chooser



Filter: /home/public/de/public_html/ds9/*.fits *.fts *.fit

Selection: /home/public/de/public_html/ds9

● Fits ◇ Compressed ◇ All

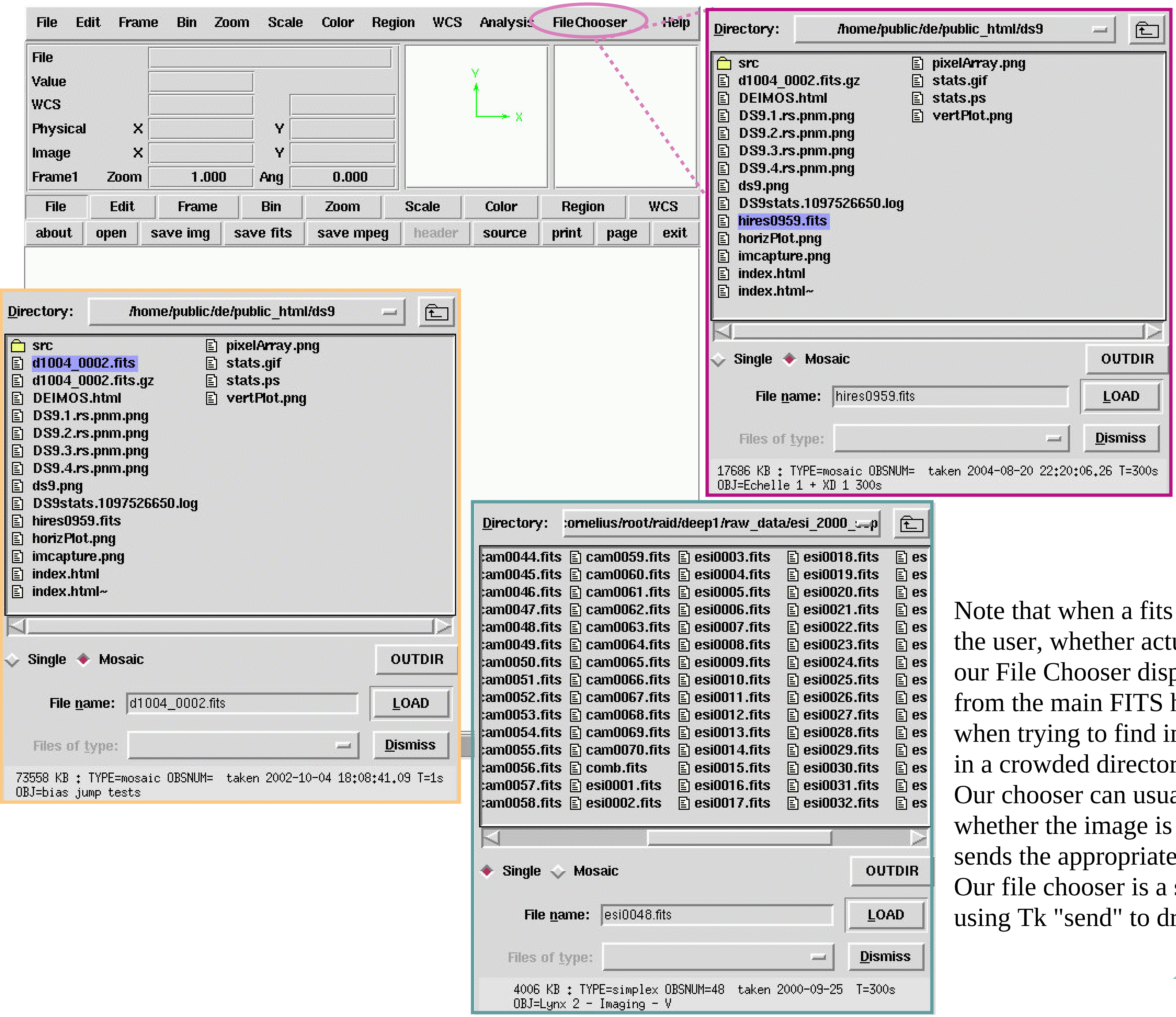
To open a FITS mosaic image file such as we produce with the DEIMOS and upgraded HIRES instruments using the DS9 File menu, we have to open the File menu, then slide sideways to "Open Other", then navigate downwards to choose "Open Mosaic WCS". For non-mosaic images we have to remember to use the plain "Open" menu option.

Either way, we get the file chooser widget shown at left.

Our users were accustomed to a more featureful file chooser, which we had been using with Figdisp for a while. We decided to graft this local file chooser onto DS9.

Extending and Customising DS9

a more featureful file chooser, part II


File
Edit
Frame
Bin
Zoom
Scale
Color
Region
WCS
Analysis
FileChooser
Help
File
Value
WCS
Physical X Y
Image X Y
Frame1 Zoom 1.000 Ang 0.000
Y
X
File
Edit
Frame
Bin
Zoom
Scale
Color
Region
WCS
about
open
save img
save fits
save mpeg
header
source
print
page
exit
Directory: /home/public/de/public_html/ds9
src
d1004_0002.fits
d1004_0002.fits.gz
DEIMOS.html
DS9.1.rs.pnm.png
DS9.2.rs.pnm.png
DS9.3.rs.pnm.png
DS9.4.rs.pnm.png
ds9.png
DS9stats.1097526650.log
hires0959.fits
horizPlot.png
imcapture.png
index.html
index.html~
pixelArray.png
stats.gif
stats.ps
vertPlot.png
Single
Mosaic
OUTDIR
File name:
LOAD
Files of type:
Dismiss
Directory: /home/public/de/public_html/ds9
src
d1004_0002.fits.gz
DEIMOS.html
DS9.1.rs.pnm.png
DS9.2.rs.pnm.png
DS9.3.rs.pnm.png
DS9.4.rs.pnm.png
ds9.png
DS9stats.1097526650.log
hires0959.fits
horizPlot.png
imcapture.png
index.html
index.html~
pixelArray.png
stats.gif
stats.ps
vertPlot.png
Single
Mosaic
OUTDIR
File name:
LOAD
Files of type:
Dismiss
Directory: :ornelius/root/raid/deep1/raw_data/esi_2000_...p
:am0044.fits cam0059.fits esi0003.fits esi0018.fits es
:am0045.fits cam0060.fits esi0004.fits esi0019.fits es
:am0046.fits cam0061.fits esi0005.fits esi0020.fits es
:am0047.fits cam0062.fits esi0006.fits esi0021.fits es
:am0048.fits cam0063.fits esi0007.fits esi0022.fits es
:am0049.fits cam0064.fits esi0008.fits esi0023.fits es
:am0050.fits cam0065.fits esi0009.fits esi0024.fits es
:am0051.fits cam0066.fits esi0010.fits esi0025.fits es
:am0052.fits cam0067.fits esi0011.fits esi0026.fits es
:am0053.fits cam0068.fits esi0012.fits esi0027.fits es
:am0054.fits cam0069.fits esi0013.fits esi0028.fits es
:am0055.fits cam0070.fits esi0014.fits esi0029.fits es
:am0056.fits comb.fits esi0015.fits esi0030.fits es
:am0057.fits esi0001.fits esi0016.fits esi0031.fits es
:am0058.fits esi0002.fits esi0017.fits esi0032.fits es
Single
Mosaic
OUTDIR
File name:
LOAD
Files of type:
Dismiss

File name: d1004_0002.fits

```
73558 KB : TYPE=mosaic OBSNUM=  taken 2002-10-04 18:08:41.09 T=1s
OBJ=bias jump tests
```

File name: hires0959.fits

```
17686 KB : TYPE=mosaic OBSNUM=  taken 2004-08-20 22:20:06.26 T=300s
OBJ=Echelle 1 + XD 1 300s
```

File name: esi0048.fits

```
4006 KB : TYPE=simplex OBSNUM=48  taken 2000-09-25  T=300s
OBJ=Lynx 2 - Imaging - V
```

The new "File Chooser" item in the top menubar eliminates sliding through cascading menus. One click launches the local FileChooser.

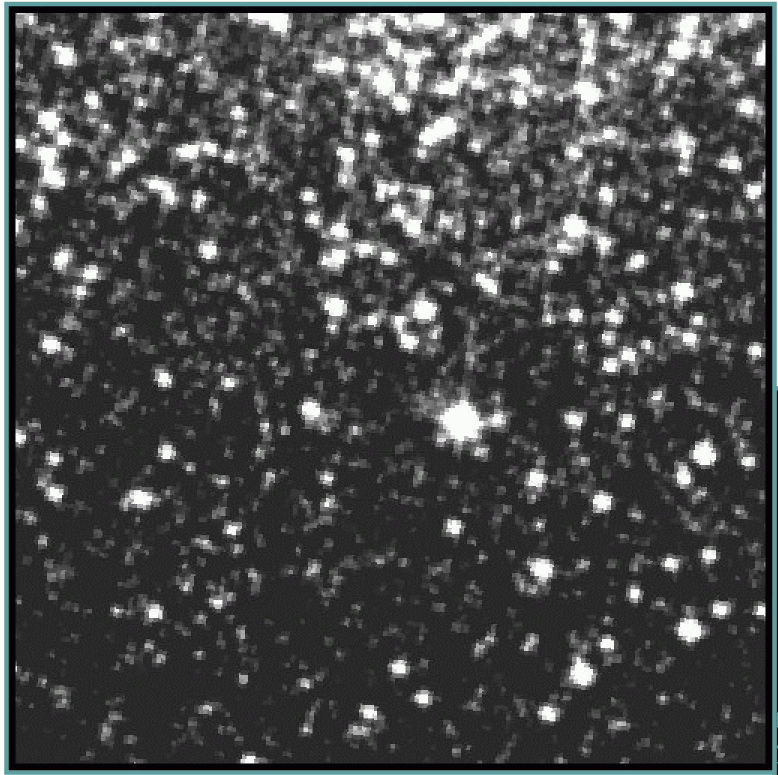
Unlike the native DS9 file chooser, which is a focus-grabbing popup (DS9 is disabled while this chooser is displayed), ours is a separate process and DS9 remains responsive while it is up. It can survive the death and restart of DS9.

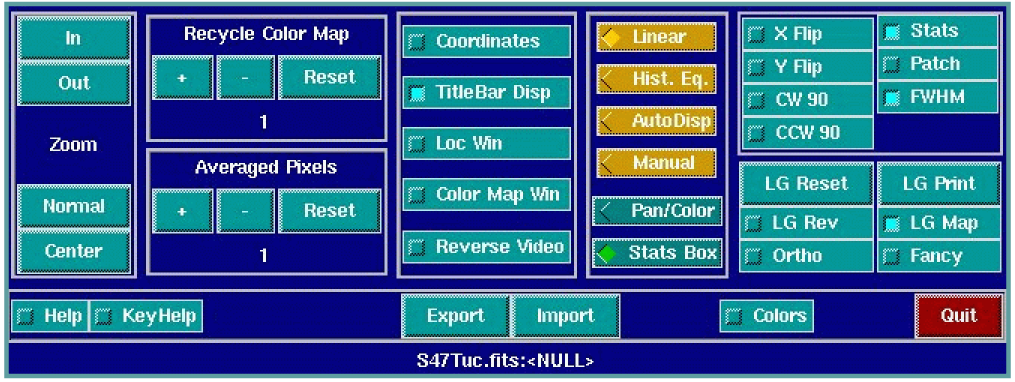
Note that when a fits file is selected by the user, whether actually loaded or not, our File Chooser displays selected keywords from the main FITS header. This is helpful when trying to find images of a certain type in a crowded directory of large FITS files. Our chooser can usually detect automatically whether the image is mosaic or simplex, and sends the appropriate load command to DS9. Our file chooser is a simple Tk application using Tk "send" to drive DS9.

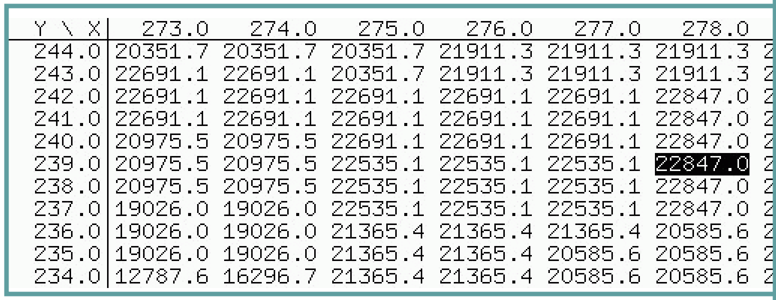
Extending and Customising DS9

basic statistics and centroiding: replacing Figdisp

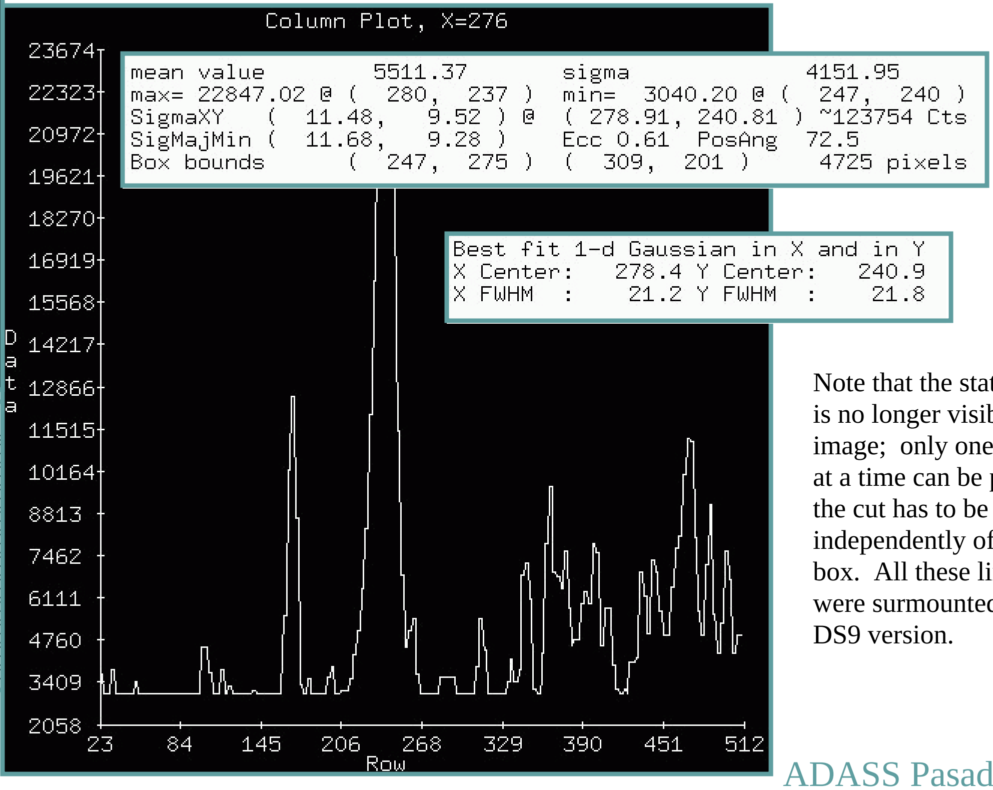
For many years Keck Observatory instruments and observers used an image display tool called ’figdisp’ for live image readout and quick-look, mimimal data reduction. Figdisp had a simple stats/centroiding feature (shown here). DS9 at the time of adoption had no such features, being envisioned as a pure-display tool with no reduction capabilities.







```
 Y \ X|   273.0    274.0    275.0    276.0    277.0    278.0
 244.0|20351.7  20351.7  20351.7  21911.3  21911.3  21911.3
 243.0|22691.1  22691.1  20351.7  21911.3  21911.3  21911.3
 242.0|22691.1  22691.1  22691.1  22691.1  22691.1  22847.0
 241.0|22691.1  22691.1  22691.1  22691.1  22691.1  22847.0
 240.0|20975.5  20975.5  22691.1  22691.1  22691.1  22847.0
 239.0|20975.5  20975.5  22535.1  22535.1  22535.1  22847.0
 238.0|20975.5  20975.5  22535.1  22535.1  22535.1  22847.0
 237.0|19026.0  19026.0  22535.1  22535.1  22535.1  22847.0
 236.0|19026.0  19026.0  21365.4  21365.4  21365.4  20585.6
 235.0|19026.0  19026.0  21365.4  21365.4  20585.6  20585.6
 234.0|12787.6  16296.7  21365.4  21365.4  20585.6  20585.6
```



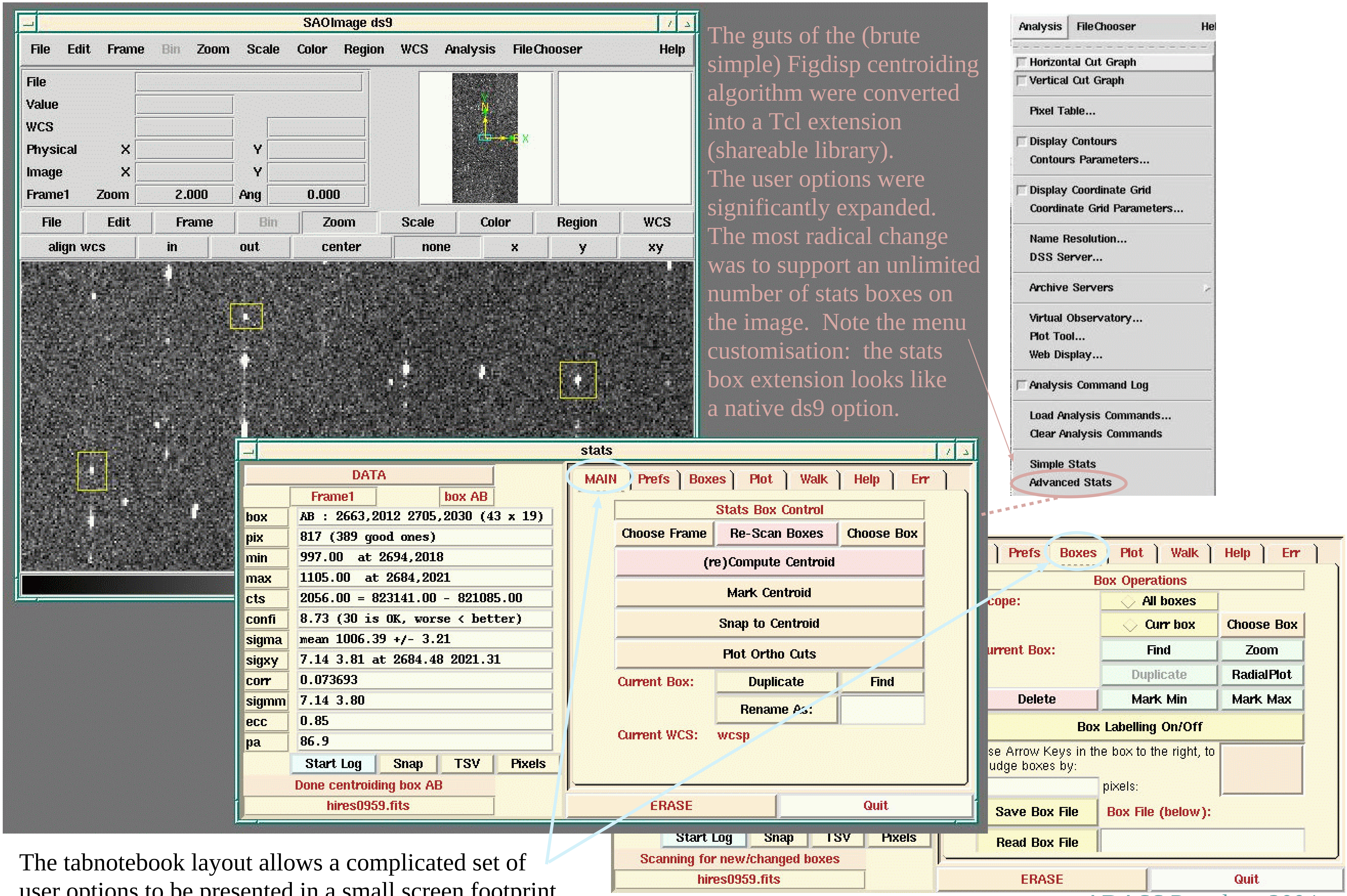
```
mean value         5511.37         sigma              4151.95
max= 22847.02 @ (   280,   237 )   min=   3040.20 @ (   247,   240 )
SigmaXY     (   11.48,    9.52 ) @  ( 278.91, 240.81 ) ~123754 Cts
SigMajMin (   11.68,    9.28 )      Ecc 0.61   PosAng   72.5
Box bounds         (   247,   275 )   (   309,   201 )      4725 pixels
```

```
Best fit 1-d Gaussian in X and in Y
X Center:    278.4 Y Center:    240.9
X FWHM   :    21.2 Y FWHM   :     21.8
```

Note that the stats box border is no longer visible on the image; only one stats box at a time can be processed; the cut has to be requested independently of the stats box. All these limitations were surmounted in the DS9 version.

Extending and Customising DS9

ds9 stats box: Figdisp on steroids



The guts of the (brute simple) Figdisp centroiding algorithm were converted into a Tcl extension (shareable library). The user options were significantly expanded. The most radical change was to support an unlimited number of stats boxes on the image. Note the menu customisation: the stats box extension looks like a native ds9 option.

stats

DATA

Frame1 box AB

```
box    AB : 2663,2012 2705,2030 (43 x 19)
pix    817 (389 good ones)
min    997.00  at 2694,2018
max    1105.00  at 2684,2021
cts    2056.00 = 823141.00 - 821085.00
confi  8.73 (30 is OK, worse < better)
sigma  mean 1006.39 +/- 3.21
sigxy  7.14 3.81 at 2684.48 2021.31
corr   0.073693
sigmm  7.14 3.80
ecc    0.85
pa     86.9
```

Done centroiding box AB

hires0959.fits

Current WCS: wcsp

Scanning for new/changed boxes

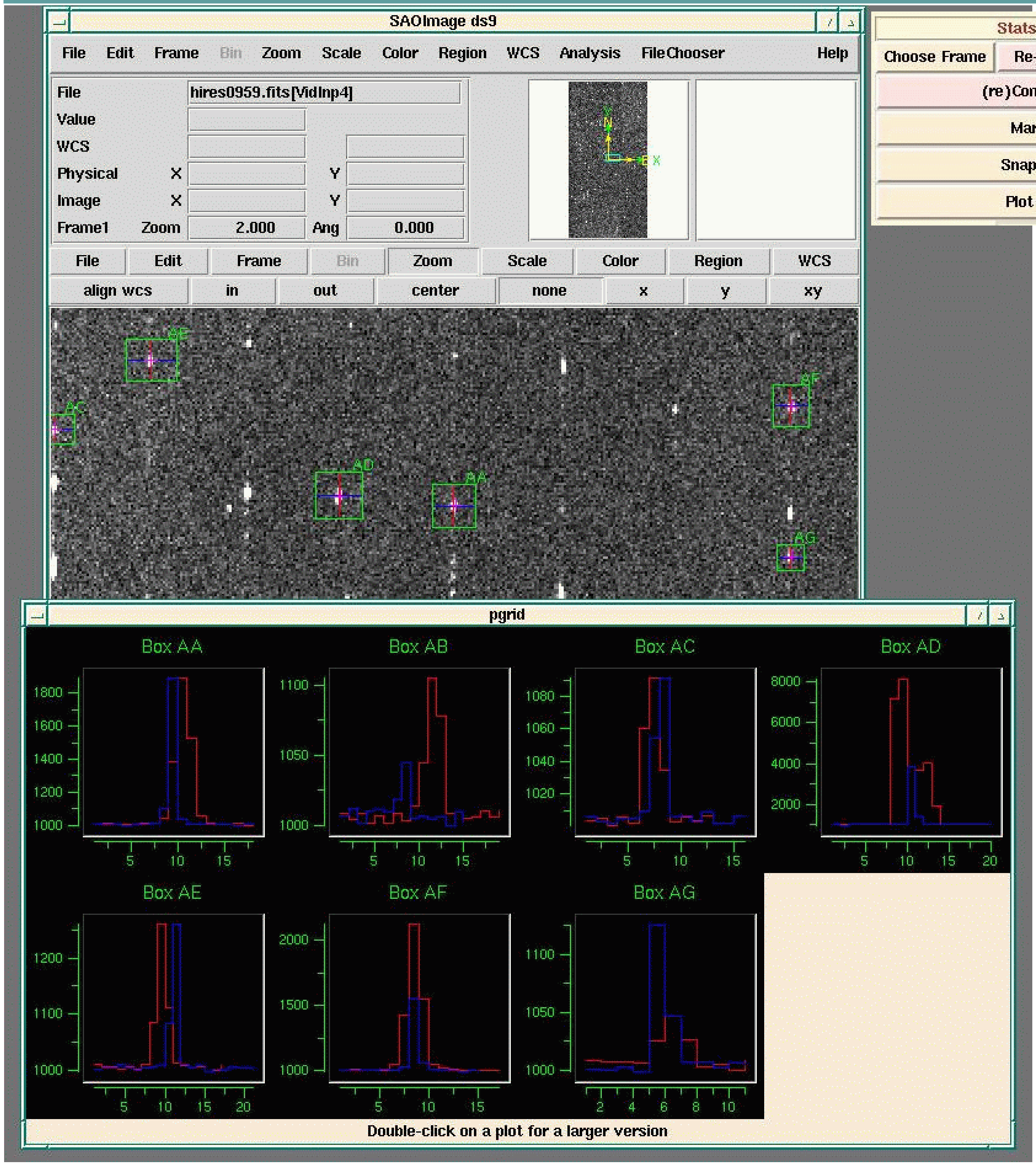
hires0959.fits

se Arrow Keys in the box to the right, to udge boxes by: pixels:

The tabnotebook layout allows a complicated set of user options to be presented in a small screen footprint.

Extending and Customising DS9

ds9 stats box: more features



File hires0959.fits[VidInp4]

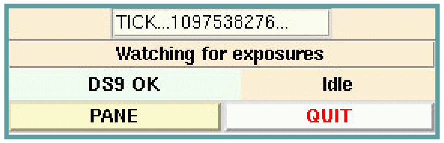
Frame1 Zoom 2.000 Ang 0.000

From the main options panel the user can compute centroids for any number of boxes, mark all centroids, snap the boxes to centre on the new centroids, and view line cuts through the centroid for each box. Box labels can be turned on and off at will. The line plots are ’live’: if a box is dragged or resized, the line plot will automatically reflect the changes in real time.

All these features are implemented in fairly straightforward Tcl code loaded at startup time. No C coding was required except for the actual stats/centroid computation, which had to be coded in C for speed. The DS9 source code remains untouched: all modifications are made by loading in additional Tcl code at startup time.

Extending and Customising DS9

live image readout

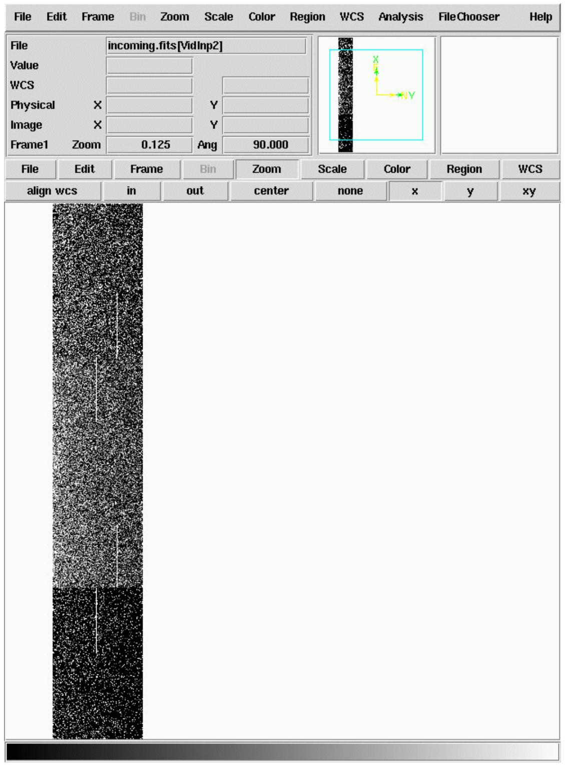


TICK...1097538276...

Watching for exposures

DS9 OK | Idle

A separate Tk process watches (via an asynch TCP based keyword API) for the start of a new exposure. It uses Tk ’send’ to initiate a readout loop in the DS9 interp which uses other keywords to determine which subraster of a shared memory cache of the image to update to screen. The image grows as the readout continues.

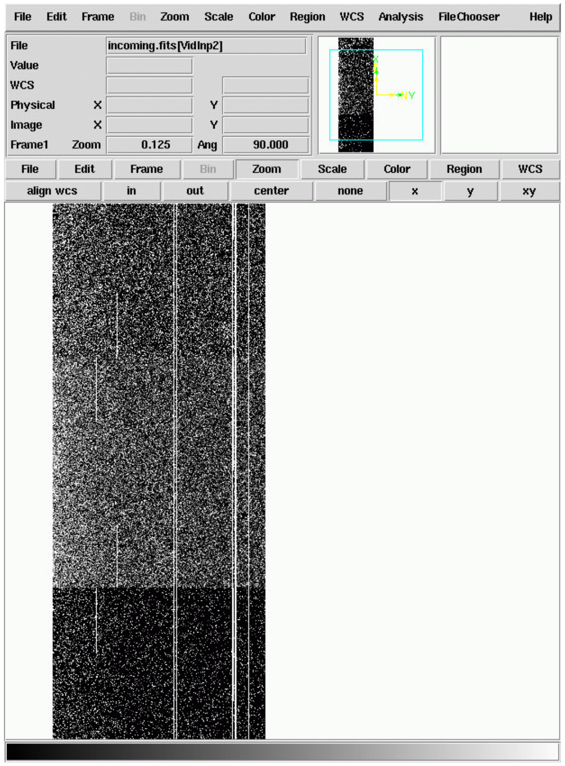


File incoming.fits[VidInp2]

Frame1 Zoom 0.125 Ang 90.000

File Edit Frame Bin Zoom Scale Color Region WCS Analysis FileChooser Help
Value
WCS
Physical X Y
Image X Y
File Edit Frame Bin Zoom Scale Color Region WCS
align wcs in out center none x y xy

approx. 800 rows

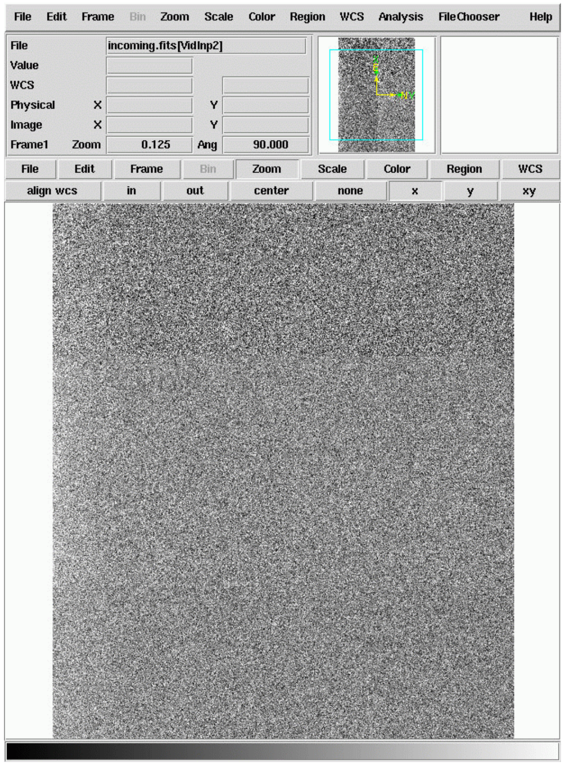


File incoming.fits[VidInp2]

Frame1 Zoom 0.125 Ang 90.000

File Edit Frame Bin Zoom Scale Color Region WCS Analysis FileChooser Help
Value
WCS
Physical X Y
Image X Y
File Edit Frame Bin Zoom Scale Color Region WCS
align wcs in out center none x y xy

approx 2000 rows



File incoming.fits[VidInp2]

Frame1 Zoom 0.125 Ang 90.000

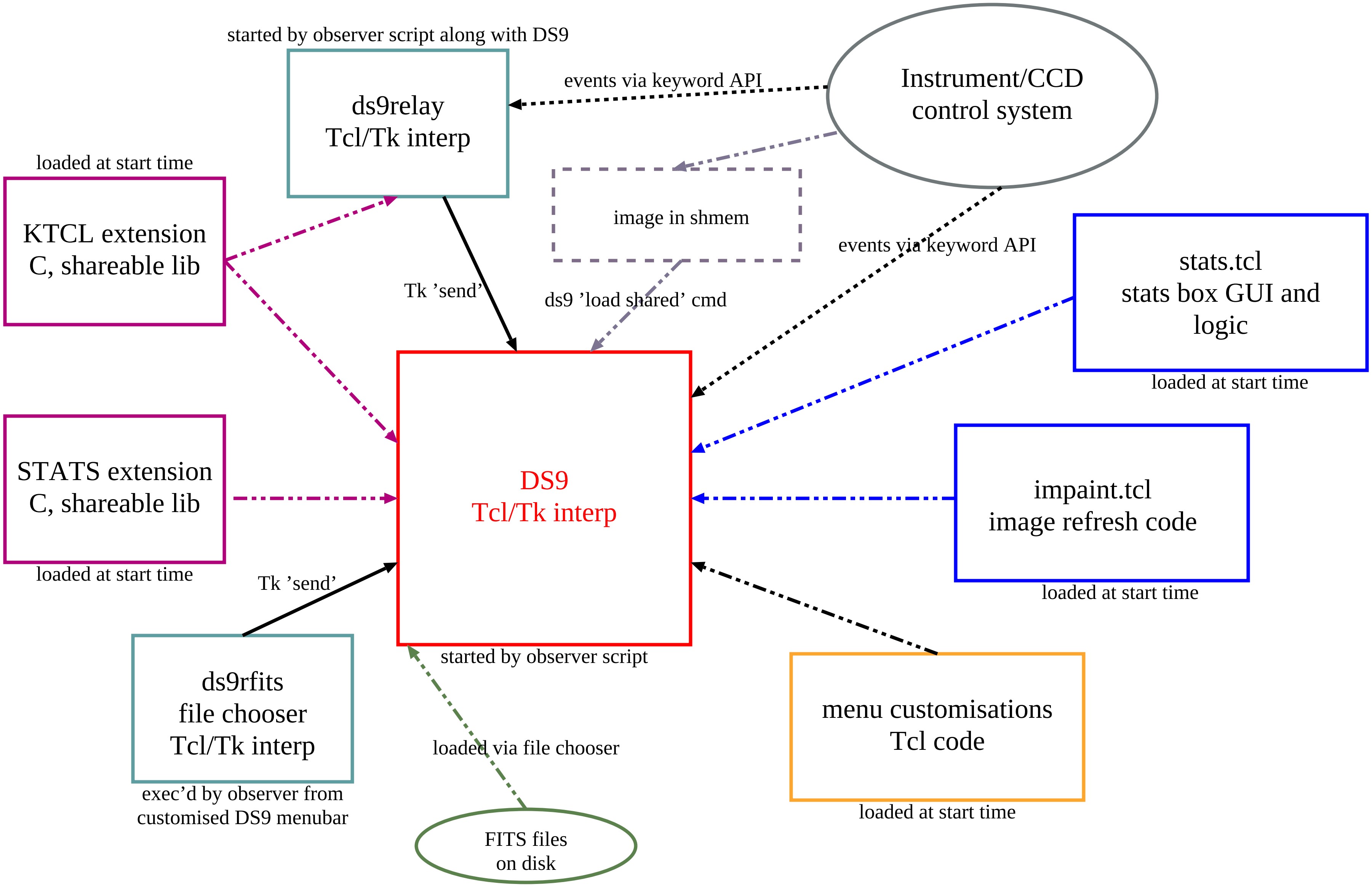
File Edit Frame Bin Zoom Scale Color Region WCS Analysis FileChooser Help
Value
WCS
Physical X Y
Image X Y
File Edit Frame Bin Zoom Scale Color Region WCS
align wcs in out center none x y xy

all 4096 rows, readout complete

Because DS9 is a Tcl/Tk interpreter it was easy to load the ’ktcl’ (Tcl extension for the Keck keyword API) shareable library at startup time, which enables the interp to monitor and respond to instrument control and exposure events. However, the job of watching for new exposure events is farmed out to a sub process. This ensures that the RTD remains maximally responsive to user events when no readout is in progress: only when a live readout is in progress does DS9 spend any cycles monitoring it. While DS9 is theoretically still "live" for user events during readout, in practise the screen updates are sufficiently time consuming that its response to pan, zoom, and other user actions is sluggish. It would be nice to improve this.

Extending and Customising DS9

live image readout: how the observer's DS9 is equipped with custom features



started by observer script along with DS9

ds9relay
Tcl/Tk interp

events via keyword API

Instrument/CCD
control system

loaded at start time

KTCL extension
C, shareable lib

image in shmem

events via keyword API

stats.tcl
stats box GUI and
logic

Tk 'send'

ds9 'load shared' cmd

loaded at start time

DS9
Tcl/Tk interp

STATS extension
C, shareable lib

impaint.tcl
image refresh code

loaded at start time

Tk 'send'

loaded at start time

started by observer script

ds9rfits
file chooser
Tcl/Tk interp

loaded via file chooser

menu customisations
Tcl code

exec'd by observer from
customised DS9 menubar

FITS files
on disk

loaded at start time

9 9

An Invitation to All DS9 Users

you're invited to help start a DS9 user community

MAILING LIST (mailman)

ds9@ucolick.org

WIKI

ds9wiki.ucolick.org

Newbies: get help and advice

Learn about new uses of DS9

Interested?
join the mailing list

Experienced Users: share tips and tricks

Share code for DS9 extensions and "plugins"

Share bug reports and workarounds

Have expertise?
contribute to the
Wiki...

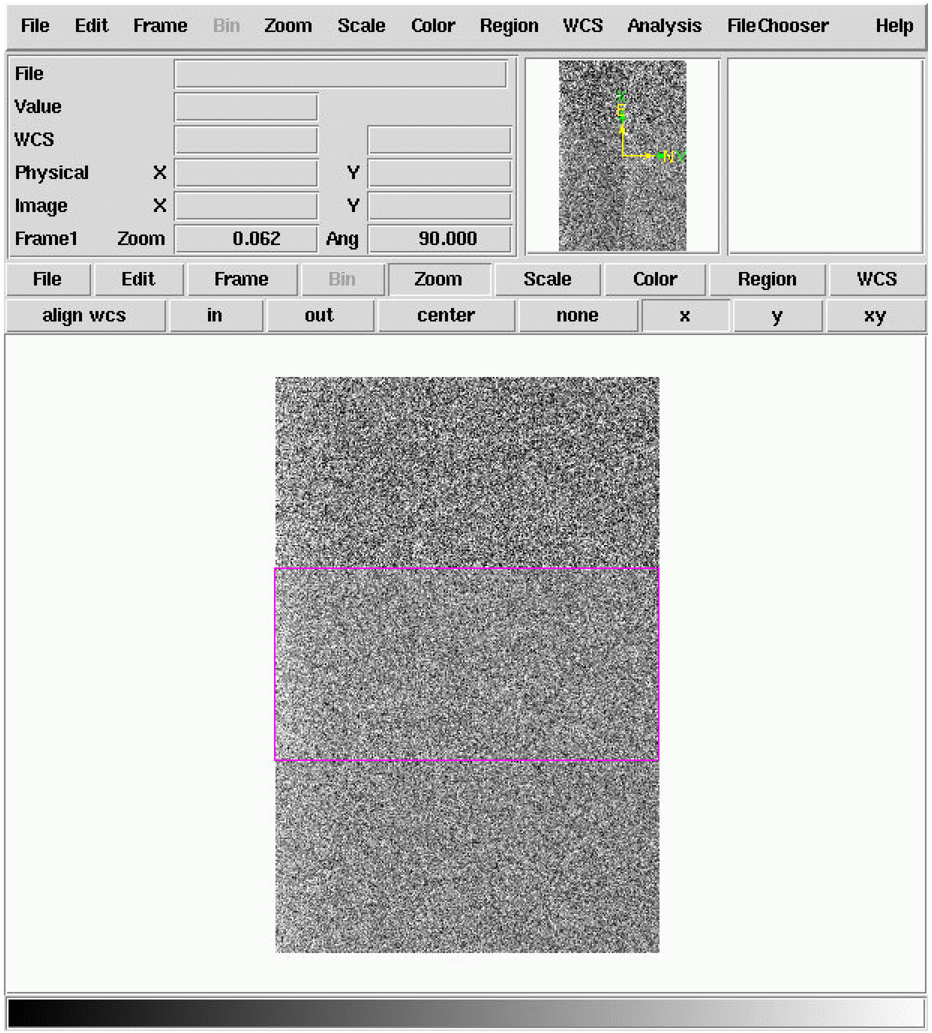
Discuss new feature requests

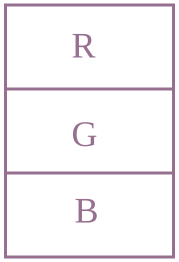
Discuss DS9 APIs and coding standards

ADASS Pasadena 2004

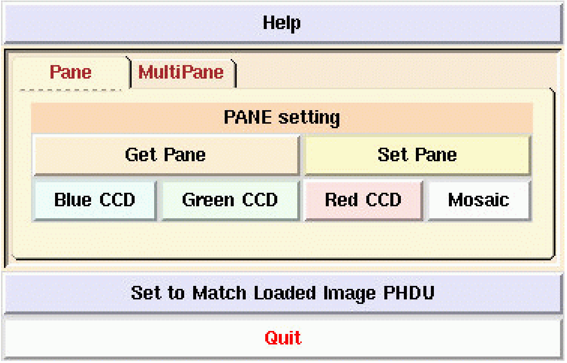
Extending and Customising DS9

subraster readout control via ds9





new HIRES mosaic
detector layout: 3
2048x4096 CCDs

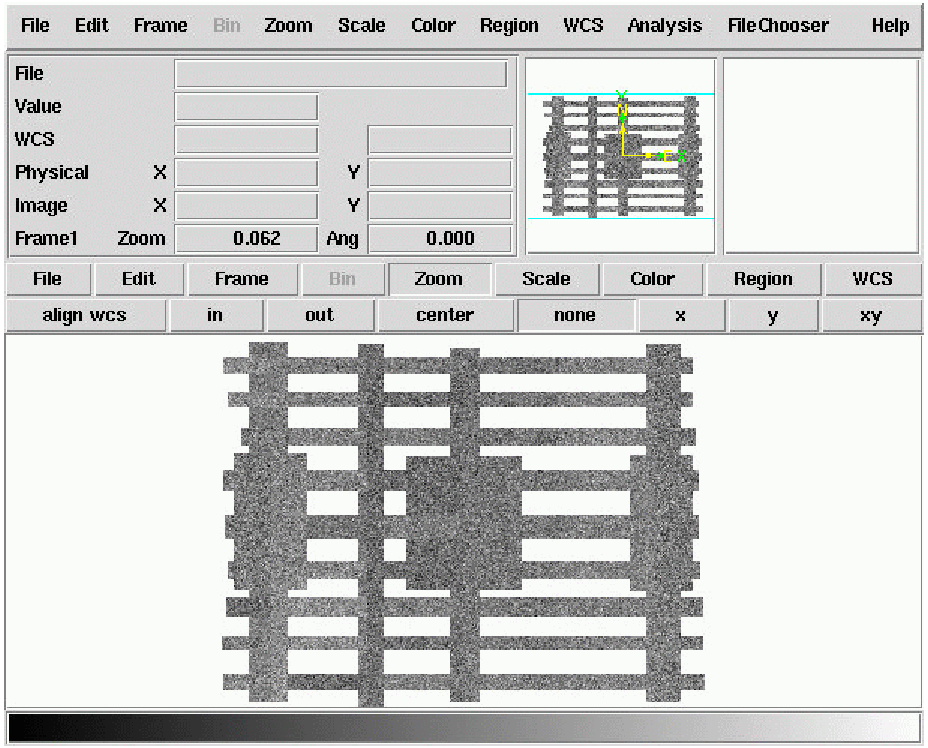


The subraster readout area is shown by the magenta box. The small control panel above is part of the separate process ’ds9relay’ which watches for new exposure events. Since it is equipped with the appropriate Tcl extension for using the keyword API, it can easily read and write the PANE and PANEn keywords which define subrasters. It communicates with DS9 via the Tk "send" mechanism to derive (or to draw) subraster regions.

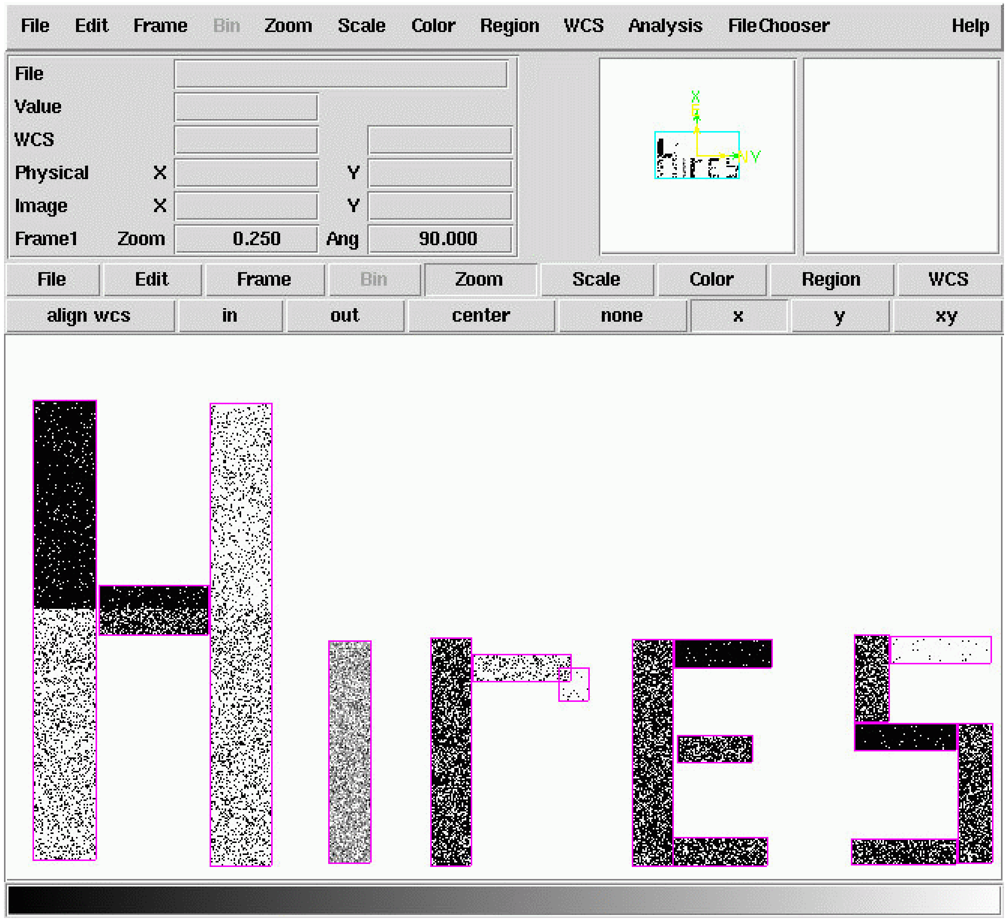
In this example the user has pressed the "Green CCD" button on the control panel and the magenta outline defines the boundary of the green CCD. The user can also use the mouse to draw, interactively, a rubberband box region which defines an arbitrary subraster. When the "Set Pane" button is pressed the PANE keyword will be written with the appropriate values (requested from DS9 via region query commands) and the next readout will be of that subraster.

Extending and Customising DS9

multiple subrasters defined using ds9

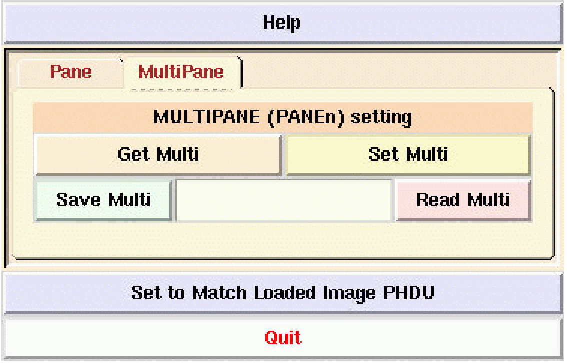


Frame1 Zoom 0.062 Ang 0.000



Frame1 Zoom 0.250 Ang 90.000

Here are a couple of silly examples showing how easy it is to define complex multiple-subraster readouts by drawing box regions in DS9. Note that each combination of region and amplifier results in one image HDU in the mosaic FITS file. With 16 panes and 6 amplifiers it is possible to create quite a baroque mosaic file! However, DS9 has no difficulty reading the file, and the pane control box makes it easy to recover the subrasters as specified for that image (using the "Set to Match..." button).



Extending and Customising DS9

references, acknowledgments and URLs

This poster, the accompanying paper, and supplemental materials and FITS files will be found at

http://spg.ucolick.org/Docs/ADASS/2004/ClarkeEtAl

Useful references:

"Data Format for the NOAO Mosaic", F. Valdes, ADASS VI. *ASP Conf. Ser.* Vol 125, 1997
"Representations of world coordinates in FITS", Greisen, E.W. & Calabretta, M.R. (2002),
Astronomy & Astrophysics, 375, 1061-1075
"Representations of celestial coordinates in FITS", Calabretta. M.R., & Greisen, E.W., (2002),
Astronomy & Astrophysics, 375, 1077-1122

Find out more about DEIMOS and HIRES at

http://www2.keck.hawaii.edu/inst/deimos/
http://www2.keck.hawaii.edu/inst/hires/hires/html
http://deimos.ucolick.org
http://www.ucolick.org/~hires/

Acknowledgments and many thanks to:

many contributors to the DS9 core --
DEIMOS and HIRES users who adapted to DS9 and suggested many improvements --
Jeff Hobbs and the Tcl/Tk core maintenance crew --
Bill Chen (author of Tgif, used to prepare this poster) --
Eric Mandel for allowing us to monopolise large chunks of Bill Joye's time and attention --
the usual suspects: network and computer support folks at Keck and Lick, the Keck summit staff,
and particularly the instrument scientists for DEIMOS and HIRES

Extending and Customising DS9

Introduction and Caveats

There are several ways to extend or customise DS9. The DS9 executable is a wish interpreter, which means that

- Additional custom Tcl code may be easily added at startup time using the -source flag
- Additional custom C code may be added as a Tcl extension and loaded at startup time or on demand, by ’package require’ or ’load’
- Another Tk process can ’drive’ DS9 via the ’Tk send’ mechanism by way of the X server, allowing entirely separate Tk applications to control DS9.

DS9 can also be controlled, and fed data, via the XPA API provided by the author. This requires an XPA client/server setup and use of predefined XPA "access points" into the interpreter. We at UCO/Lick did not like being restricted to a predefined set of access points or functions. We wanted direct access to all the DS9 features on demand, and being already literate in Tcl/Tk we found the options above to be more attractive than XPA which, in addition to being potentially limiting, was unfamiliar and presented a learning curve.

On the whole this decision has worked out well, but it has not been without some costs. We are more vulnerable to inter-version changes to the syntax of DS9 internal commands, whereas (we assume) more effort is made to endure inter-version compatibility of defined and published XPA access points. Despite occasional porting issues when a new version is installed, for the kind of intrusive tinkering we like to do, direct access via Tcl code remains preferable.

Examples of the above techniques follow. We used additional custom Tcl code loaded at runtime to customise DS9 menus, to add basic image stats functions, and to add live image readout capability. We used loaded shared libraries to provide the centroiding algorithm for the stats features, and to load the instrument control API library for live image readout synchronisation. We used the ’Tk send’ mechanism to make an external file chooser load FITS files into DS9, and to write a ’watcher’ that drives DS9 to load incoming images at the appropriate moment in the observing activity cycle. Screen shots and more details appear above. All of this code is now in production use for observing with DEIMOS and HIRES (with mosaic detector upgrade). In addition, we needed modifications to the DS9 core code for better support of mosaic images and multiple WCS, which Bill Joye provided.

Extending and Customising DS9

it’s not all DIY: core features and patches from DS9 author (Bill)

Not everything we wanted to change or enhance about DS9 was within our power. There were two reasons why we shrank from any attempt to patch the distribution code.

1) we did not want to be re-installing our patches every time Bill released a new version

2) the core of DS9 is very large, and is written in C++ using yacc and lex (bison and flex) for command parsing. Fairly deep knowledge is required to patch without breaking it.
It is much less intimidating, and less of a time sink, for us to keep our extensions at or above the Tcl layer of DS9.

Therefore we were dependent on author Bill Joye for many features, bugfixes, etc. required to support our instruments and keep our users happy, as well as to facilitate some of the pure-Tcl extensions we wrote. A partial list of author-provided upgrades and extensions follows:

- ☆ support for mosaic images with multiple WCS
- ☆ display of mosaic images using selected WCS for layout
- ☆ refinement of heuristics for loading NOAO MEF files
- ☆ support for FITS image files in shared memory
- ☆ addition of command and options at Tcl layer to facilitate extension/enhancement
- ☆ expansion of user preferences for increased user-friendliness
- ☆ improvements to ’make’ scheme

Extending and Customising DS9

conclusions

The decision to adopt DS9 as the RTD for DEIMOS rather than upgrade older, familiar code to handle mosaic images was not undertaken lightly. One of the strongest arguments for gambling on a new, unknown RTD (several years ago) was that its Tcl/Tk nature would make it easily extended and customised, thus reducing the cost of ownership and potentially offering observers a rich and evolving feature set. It is now seven years since the DEIMOS critical design review, when our intention to switch to DS9 was formally announced. Our experience since then suggests that we were correct in believing that DS9's flexibility and ease of customisation/enhancement would more than compensate for the costs of abandoning our old familiar RTD.

We have successfully customised DS9 to add several idiosyncratic features specific to our instruments or our science.

This customisation has cost less, and design iteration has been easier, than with any other RTD we have used.

We have received excellent support from the author (Bill Joye) and the CFA group at Harvard.

We feel confident that we will be able to extend and customise DS9 to support future instruments and projects.

We feel that DS9 is well-positioned to become a *de facto* standard image display for astronomical applications, for both observing and data reduction.

We feel that we, and probably others, would benefit from a wider community effort to share information (and code) from various DS9 customisation efforts. One of the goals of this year's poster and BoF is to encourage the formation of a DS9 users' community for code sharing, discussion of Tcl and other APIs, community help and tutoring for newbies, tips-and-tricks archives, and joint discussion of new feature requests.