#

NAME

ximtool – interactive image display program for the X Window System

SYNOPSIS

ximtool [–*toolkitoption* ...] [*options* ...]

OPTIONS

-basePixel *N*

The base colormap cell used by the colormap. This essentially allows you to reserve *basePixel* colors in the global colormap for other applications. The default is 64, if changed you'll need to also specify the *-cmapInitialize* option or resource.

-cmap1 *file*

User colormap 1. This flag allows you to specify a colormap to be made available at task startup.

-cmap2 *file*

User colormap 2. This flag allows you to specify a second colormap to be made available at task startup.

-cmapDir1 *dir*

User colormap directory 1. Specifies a directory to be searched for colormaps.

-cmapDir2 *dir*

User colormap directory 2. Specifies a directory to be searched for colormaps. By default this points to the system directory /usr/local/lib/imtoolcmap, allowing a set of site default colormaps to be defined here.

-cmapInitialize *bool*

Initialize the ximtool colormap at startup. When setting the *basePixel* option or resource this is required in order to force the Gterm widget to update its global colormap resource in the X server. The default is *false*.

-cmapName *name*

Name used for private colormap. The default for all IRAF imaging applications is *image*. Gterm widget based imaging applications which have the same value of cmapName will share the same colormap, minimizing colormap flashing and allowing multiple applications to be run at the same time.

-config *N*

Initial frame buffer configuration number. The default value is 1, indicating a 512x512 frame buffer with 2 frames. See below for information on the frame buffers.

-defgui

Print the default GUI to the stdout. The GUI is a Tcl program that may be customized by the user and reloaded using the *-gui* option or the *gui* resource parameter.

-displayPanner *bool*

Display panner marker window at startup. If set, a panner window showing the full frame buffer will appear in the upper-right side of the main display window.

-displayCoords *bool*

Display WCS coordinate marker window at startup. If set, a coordinate readout text marker showing will appear in the lower-right side of the main display window.

-fifo *pipe*

Specifies the name of the fifo pipe to be used, the *i* and *o* suffixes will be added automatically. The default pipe names will be /dev/imt1i (input pipe) and /dev/imt1o (output pipe).

-fifo_only

If set, only fifo pipes will be used for communication with a client program, sockets will be disabled.

-gui *file*
Specifies the GUI file to be used.

-help Print a summary of command line options to the screen.

-imtoolrc *file*
Specifies the frame buffer configuration file to be used. See below for information on frame buffers.

-inet_only
If set, only inet sockets will be used for communication with a client program, fifo pipes and unix sockets will be disabled.

-invert
Start XImtool using inverted colormaps. When set, a "normalized" display will always be the inverse of the selected colormap.

-ismdev *dev*
Specifies the plug-in ISM connection socket. This should be a unix domain socket of the form "*/tmp/.ISM%d*", where the *%d* will be replaced by the user id. Once an ISM has connected this port is freed to accept other connections.

-maxColors *N*
Specify the max number of colors to be used for the display.

-memModel *type*
Determines how ximtool uses memory in the ximtool client and the X server. The options are *fast*, *beNiceToServer*, and *small*. The default is *fast*, which uses server pixmaps to make frame blink fast. This is recommended unless server memory is very limited. Note that even in fast mode, the server pixmap is only the size of the display window, so memory usage is reasonable even if the frame buffer is very large.

-nframes *N*
Specifies the number of frame buffers to configure at startup. By default there will be 2 frames available, a maximum of 4 frames are allowed.

-port *N*
Specifies the port number to use when connecting through an inet socket.

-port_only
Same as *-inet_only* option. If set, only inet sockets will be used for communication with a client program.

-printConfig *name*
Specifies the printer configuration file to use. By default this will be /usr/local/lib/ximprint.cfg. See below for more information on configuring output devices.

-tile The default display mode is to view one frame at a time. In tile frames mode, 2 or 4 frames may be viewed simultaneously in the display window. All the usual operations (zoom and pan, colortable enhancement, cursor readback, etc.) still work for each frame even when in tile frames mode.

-unix *name*
Specifies the unix domain socket name to use. A "%d" in the filename will be replaced with the user id.

-unix_only
If set, only unix domain sockets will be used for communication with a client program, inet sockets and fifos will be disabled.

APPLICATION RESOURCES

XImtool is implemented as a client program which is responsible for loading the frame buffers/colormaps, communicating with clients, etc, and a user-modifiable GUI file written as a Tcl script which handles all the user interface details. The *client resources* described below will be common to any user-defined GUI, the *gui resources* may change depending on how extensively the GUI has been modified by the user. Each of these components has its own set of resources, but to the user setting them is the same as with any other application.

Gterm widget resources (i.e. those for the main image window or colorbar) may be set as either client or GUI resources. See the *xgterm(1)* man page for a complete description of *Gterm* widget resources.

CLIENT RESOURCES

The client resources generally define the initial state of the application or set configuration parameters.

Resource Name	**Default Value**
defConfig	1
defNFrames	0
tileBorderWidth	3
tileBorderColor	9
autoscale	false
antialias	false
antialiasType	boxcar
tileFrames	false
highlightFrames	true
gui	default
imtoolrc	/usr/local/lib/imtoolrc
invert	false
memModel	fast
basePixel:	64
maxColors:	216
cmapInitialize:	false
cmap1	none
cmap2	none
cmapDir1	none
cmapDir2	/usr/local/lib/imtoolcmap
input_fifo	/dev/imt1i
output_fifo	/dev/imt1o
unixaddr	/tmp/.IMT%d
port	5137
ism_addr	/tmp/.ISM%d
ism_task	"ism_wcspix.e wcspix &"

Description of ximtool client resources:

defConfig Default frame buffer configuration number on startup. See below for more information on frame buffers.

defNFrames Default number of frames on startup. Set to zero to use the value from the frame buffer configuration (*imtoolrc*) file.

tileBorderWidth
tileBorderColor Used by the tile frames option. Specifies how far apart to space the frames in tile frames mode. Color "9" refers to the Gterm widget resource color9, which is assigned a color with its own resource.

autoscale Enable/disable the autoscale option.

antialias Enable/disable the antialias option.

antialiasType Type of antialiasing. Options include *boxcar* (default), *bilinear*, *nearest*, *area*, *blkavg*, *lowpass*, and *gaussian*.

tileFrames
Enable/disable the tile frames option.

highlightFrames
Determines whether the current frame is highlighted when in tile frames mode.

gui
The GUI to be executed. "default" refers to the default, builtin ximtool GUI. You can replace this with your own GUI file if you are bold enough, and completely change the look and functionality of the GUI if desired.

imtoolrc
Where to find the imtoolrc file. This defines the recognized frame buffer configurations.

invert
Start Ximtool using an inverted colormap. When set, a "normalized" display will always be the inverse of the selected colormap.

memModel
Determines how ximtool uses memory in the ximtool client and the X server. The options are *fast*, *beNiceToServer*, and *small*. The default is *fast*, which uses server pixmaps to make frame blink fast. This is recommended unless server memory is very limited. Note that even in fast mode, the server pixmap is only the size of the display window, so memory usage is reasonable even if the frame buffer is very large.

basePixel
maxColors
These two resources determine the region of colormap space used to render image pixels.

cmapInitialize
Initialize the ximtool colormap at startup. This is a required resource to clear a previous ximtool colormap allowing a new **basePixel** and **maxColors** to take effect.

cmap1
cmap2
User colormap files. The intent here is to allow individual colormaps to be conveniently specified as a resource.

cmapDir1
cmapDir2
User or system colormap directories. By default cmapDir2 points to the system directory /usr/local/lib/imtoolcmap, allowing a set of site default colormaps to be defined here. This leaves cmapDir1 available to a user colormap directory.

input_fifo
output_fifo
The input and output fifos for fifo i/o. "Input" and "output" are from the client's point of view. Note that only one display server can use a fifo-pair at one time.

unixaddr
Template address for unix domain socket. The user must have write permission on this directory, or the file must already exist. *%d*, if given, is replaced by the user's UID.

port
TCP/IP port for the server. Note that only one server can listen on a port at one time, so if multiple ximtool servers are desired on the same machine, they should be given different ports.

ism_addr
Template address for ISM unix domain socket. The user must have write permission on this directory, or the file must already exist. *%d*, if given, is replaced by the user's UID.

ism_task
Command string to execute for the real-time pixel and WCS readout ISM (Image Support Module) task.

GUI RESOURCES

In principle ximtool can have any number of different GUIs, each of which defines its own set of resources. GUIs typically define a great many resources, but most of these are not really intended for modification by the user (although one can modify them if desired).

The following are some of the more useful resources used by the default ximtool GUI. The *imagewin* resources are Gterm widget resources.

Main Display Gterm Widget Resources

Resource Name	**Default Value**
∗imagewin.cmapName:	image
∗imagewin.basePixel:	64
∗imagewin.warpCursor:	True
∗imagewin.raiseWindow:	True
∗imagewin.deiconifyWindow:	True
∗imagewin.ginmodeCursor:	circle
∗imagewin.ginmodeBlinkInterval:	500
∗imagewin.color0 (background):	black
∗imagewin.color1 (foreground):	white
∗imagewin.color8 (panner highlight):	#7c8498
∗imagewin.color9 (tileFrame color):	SteelBlue
∗imagewin.width:	512
∗imagewin.height:	512

GUI Resources

Resource Name	**Default Value**
∗autoscale:	True
∗zoomfactors:	1 2 4 8
∗displayCoords:	True
∗displayPanner:	True
∗displayMagnifier:	True
∗blinkRate:	1.0
∗pannerArea:	150∗150
∗pannerGeom:	-5+5
∗magnifierArea:	100∗100
∗magnifierGeom:	+5+5
∗wcsboxGeom:	-5-5
∗maxContrast:	5.0
∗warnings:	True
∗centerBoxSize:	5
∗peakCentroid:	True

Alternate GUI Resources

Resource Name	**Default Value**
∗showToolBar:	False
∗showPanelBar:	False

Description of selected resources:

∗**cmapName** Name used for private colormap. The default for all IRAF imaging applications is "image". Gterm widget based imaging applications which have the same value of cmapName will share the same colormap, minimizing colormap flashing and allowing multiple applications to be run at the same time.

∗**basePixel** The base colormap cell used by the display colormap.

∗**imagewin.warpCursor** Warp pointer into image window when initiating a cursor read.

∗**imagewin.raiseWindow** Raise image window when initiating a cursor read.

∗**imagewin.deiconifyWindow**
Deiconify image window if necessary when initiating a cursor read.

∗imagewin.ginmodeCursor
Type of cursor when a cursor read is in progress. The default is a circle. Any selection from the X cursor font can be used. A special case is "full_crosshair" which is the full crosshair cursor of the Gterm widget.

∗imagewin.ginmodeBlinkInterval
Determines whether the cursor blinks when a cursor read is in progress. The value is given in milliseconds.

∗imagewin.color0	Background color.
∗imagewin.color1	Foreground color.
∗imagewin.color8	Color assigned the panner window.
∗imagewin.color9	Color used for the tileFrames highlight.
∗imagewin.width	Width of the main image window.
∗imagewin.height	Height of the main image window.
∗pannerArea	Area in pixels of the panner window.
∗pannerGeom	Where to place the panner window.
∗wcsboxGeom	Where to place the coords box.
∗maxContrast	Maximum contrast value.

DESCRIPTION

As a display server, XImtool is started as a separate process from client software such as IRAF. Once it is running it will accept client connections simultaneously on fifo pipes, unix domain sockets, or inet sockets. A display client like the IRAF *DISPLAY* task makes a connection and sends the image across using a modified IIS Model 70 protocol. Once the image is loaded in the display buffer it may be enhanced, saved to a disk file in a number of different formats, or printed as Encapsulated Postscript to a printer or disk file. Up to sixteen frame buffers are allowed, these may be displayed simultaneously in a tiled mode, or blinked frame-to-frame. Each frame may have its own colormap or brightness/contrast enhancement. Pan/Zoom and cursor readout are permitted using *markers*, on-line help is also available.

When run in standalone mode, images (currently IRAF OIF, GIF, Sun Rasterfiles or simple FITS (i.e. excluding MEF files) formats are permitted) may be loaded on the command line or by using the Load Panel. This allows you to browse images and perform the same manipulations as if they had been displayed by a client.

MOUSE OPERATIONS

Clicking and dragging MB1 (mouse button 1) in the main image window creates a rectangular region marker, used to select a region of the image. If you do this accidentally and don’t want the marker, put the pointer in the marker and type DELETE or BACKSPACE to delete the marker. With the pointer in the marker, MB3 will call up a marker menu listing some things you can do with the marker, like zoom the outlined region. MB1 can be used to drag or resize the marker. See below for more information on markers.

Clicking on MB2 in the main image window pans (one click) or zooms (two clicks) the image. Further clicks cycle through the builtin zoom factors. Moving the pointer to a new location and clicking moves the feature under the pointer to the center of the display window. Holding down the Shift key while clicking MB2 will cause a full-screen crosshair cursor to appear until the button is released, this can be useful for fine positioning of the cursor.

MB3 is used to adjust the contrast and brightness of the displayed image. The position of the pointer within the display window determines the contrast and brightness values. Click once to set the values corresponding to the pointer location, or click and drag to continuously adjust the display.

KEYSTROKE ACCELERATORS

The following keystrokes are currently defined in the GUI:

-------------------- **Misc Functions** ---------------------

Ctrl-b	Previous (back) frame
Ctrl-c	Center frame
Ctrl-f	Forward frame
Ctrl-i	Invert colormap
Ctrl-m	Toggle magnifier
Ctrl-n	Normalize
Ctrl-p	Toggle panner
Ctrl-r	Register
Ctrl-s	Match LUT scaling
Ctrl-t	Tile frames toggle
Ctrl-u	Unzoom (zoom=1)
Ctrl-x	Flip X
Ctrl-y	Flip Y
Ctrl-=	Print using current setup
Ctrl-<	Decrease blink rate (blink faster)
Ctrl->	Increase blink rate (blink slower)
Ctrl-+	Zoom in
Ctrl--	Zoom out
Alt-1 thru Alt-4	Set frame to be displayed
Ctrl-1 thru Ctrl9	Set integer zoom factor
Ctrl-Alt-q	Quit
Ctrl-Alt-f	Fitframe

--------------------- **Panel Toggles** ---------------------

Alt-b	Blink frames
Alt-c	Control panel
Alt-h	Help popup
Alt-i	Info box popup
Alt-l	Load file popup
Alt-p	Print popup
Alt-s	Save popup
Alt-t	TclShell popup

------------------- **Cursor Positioning** ------------------

Ctrl-h / Ctrl-Left	Move cursor one pixel left
Ctrl-j / Ctrl-Down	Move cursor one pixel down
Ctrl-k / Ctrl-Up	Move cursor one pixel up
Ctrl-l / Ctrl-Right	Move cursor one pixel right
Shift-Ctrl-h / Shift-Ctrl-Left	Move cursor ten pixels left
Shift-Ctrl-j / Shift-Ctrl-Down	Move cursor ten pixels down
Shift-Ctrl-k / Shift-Ctrl-Up	Move cursor ten pixels up
Shift-Ctrl-l / Shift-Ctrl-Right	Move cursor ten pixels right

------------------- **Auto-Registration** -------------------

Ctrl-a	Toggle auto-registration
Ctrl-o	Set frame offset

-------------------- **Frame Positioning** ------------------

Ctrl-Left	Shift one full frame left
Ctrl-Down	Shift one full frame down
Ctrl-Up	Shift one full frame up
Ctrl-Right	Shift one full frame right
Ctrl-Alt-Left	Shift one half frame left
Ctrl-Alt-Down	Shift one half frame down
Ctrl-Alt-Up	Shift one half frame up
Ctrl-Alt-Right	Shift one half frame right

------------------- **Peak-Up Centroiding** -----------------

Ctrl-[	Decrease centroiding box size
Ctrl-]	Increase centroiding box size
Ctrl-0 (zero)	Centroid/find local maximum
Alt-Ctrl-0 (zero)	Find local minimum

------------------ **Mouse Button Events** ------------------

Shift-Btn1Down	Turn on magnifier
Shift-Btn1Up	Turn off magnifier
Shift-Btn2Down	Turn on crosshair cursor
Shift-Btn2Up	Turn off crosshair cursor
Btn1Down	Create a Marker
Btn1Motion	Resize marker being created
Btn2Down	Zoom/center on cursor position
Btn3Down/Motion	Brightness/contrast scale the image
Ctrl-Btn1Down	Create Ruler Marker
Ctrl-Btn1Motion	Resize Ruler Marker being created
Ctrl-Btn1Up	Destroy Ruler Marker
Alt-Motion	Freeze cursor readout

NOTE: These keystrokes only work with the cursor in the main image window, only a few of the commands are implemented to work within subwindows or markers to avoid conflicts with translations for those objects. If a command does not work, check the cursor location and try it again in the main display window.

FRAME BUFFER CONFIGURATIONS

XImtool starts up using default frame buffer size of 512x512 pixels, two (of 16 possible) frames will be created. When loading disk images (i.e. run in standalone mode) the frame buffer configuration file will be searched for a defined frame buffer that is the same size or larger than the current image, if no suitable buffer can be found a custom frame buffer the same size as the image will be created in an unused portion of the configuration table. When used as a display server the frame buffer configuration number is passed in by the client and loaded explicitly even if it means clipping the image. If a new frame buffer is a different size than previously defined frames, all available frames will be initialized and cleared prior to the display. The default frame buffer configuration file is /usr/local/lib/imtoolrc, this can be overridden by defining a IMTOOLRC environment variable naming the file to be used, by creating a using the *-imtoolrc* command line flag or *imtoolrc* application resource.

The format of the frame buffer configuration file is

configno nframes width height [extra fields]

e.g.

```
      1  2  512  512
```

```
2  2  800  800
3  1 1024 1024          # comment
:  :   :    :
```

At most 128 frame buffer sizes may be defined, each configuration may define up to 16 frames, configuration numbers need not be sequential.

NOTE: When defining a new frame buffer for use with client software such as IRAF the user must also remember to define those frame buffers in the IRAF *dev$graphcap* file.

SUPPORT FOR 16 DISPLAY FRAMES

As part of the extensive GUI changes with the V1.3 release, support for the full 16 frames allowed by the IIS protocol is now available. IRAF V2.11.4 or later client tasks (and CDL library) are required to take advantage of this frames. All changes are backwards compatible, older versions of IRAF will continue to work but cannot access more than the original four frames. The new DISPLAY task will automatically sense whether the display server being used supports 16 frames or the original 4 and adjust the 'frame' parameter maximum accordingly. The changes are fully backwards compatible for other servers.

More frames are possible if needed but will require further changes to the client IRAF code to be effective. Allowing creation of more than 16 frames by the Load panel can be done independently but would also require numerous code change to XImtool. Please contact site support if there is a need for this, or for workaround suggestions depending on your application.

MARKERS

Although ximtool doesn't do much with markers currently, they are a general feature of the *Gterm* widget and are used more extensively in other programs (e.g. the prototype IRAF science GUI applications). XImtool uses markers for the marker zoom feature discussed above, and also for the panner and the coords box. All markers share some of the same characteristics, so it is worthwhile learning basic marker manipulation keystrokes.

- MB1 anywhere inside a marker may be used to drag the marker.
- MB1 near a marker corner or edge, depending on the type of marker, resizes the marker.
- Shift-MB1 on the corner of most markers will rotate the marker.
- Markers stack, if you have several markers and you put one on top of the other. The active marker is highlighted to tell you which of the stacked markers is active. If the markers overlap, this will be marker "on top" in the stacking order.
- MB2 in the body of a marker "lowers" the marker, i.e. moves it to the bottom of the stacking order.
- Delete or backspace in a marker deletes it.
- Markers have their own translation resources and so the default keystroke commands will not be recognized when the cursor is in a marker.

  For example, try placing the pointer anywhere in the coords box, then press MB1 and hold it down, and drag the coords box marker somewhere else on the screen. You can also resize the coords box by dragging a corner, or delete it with the delete or backspace key. (The Initialize button will get the original coords box back if you delete it, or you can reset the toggle in the control panel).

PANNER MARKER

The panner window always displays the full frame buffer. Try setting the frame buffer configuration to a nonsquare frame buffer (e.g. imtcryo) and then displaying a square image (e.g. dev$pix) and the panner will show you exactly where the image has been loaded into the frame.

The panner window uses two markers, one for the window border and one to mark the displayed region of the frame. Most of the usual marker keystrokes mentioned below apply to these markers as well, e.g. you can use MB1 to reposition on the panner window within the main image display window, or to drag the region marker within the panner (pan the image). Resizing the region marker zooms the image; this is a non-aspect constrained zoom. The panner window itself can be resized by dragging a corner with MB1. Typing delete or backspace anywhere in the panner window deletes the panner.

A special case is MB2. Hitting MB2 anywhere in the panner window pans the image to that point. This is analogous to hitting MB2 in the main display window to pan the image.

The panner marker can be disabled by defining the *displayPanner* GUI resource, its size and location can be controlled using the *pannerArea* and *pannerGeom* GUI resources respectively.

MAGNIFIER MARKER

The magnifier marker can be used to zoom in on a small area around the cursor. It will be updated as the cursor moves but only for small motions (either mouse movement or with the cursor movement keystrokes) to minimize the impact on the system. The zoom factor is expressed as some fraction of the size of the magnifier marker itself. The default zoom is 4, i.e. the area in the marker represents and area in the image that's one-fourth the size of the marker. Other zoom factors may be selected using the popup menu created by hitting MB1 in the marker.

By default the magnifier marker is not visible, to toggle it select the *Magnifier* option from the *Options* menubar button. Alternatively, for just a quick look holding down the Shift and MB1 buttons will display the marker until the button is released.

The magnifier marker can be disabled by defining the *displayMagnifier* GUI resource, its size and location can be controlled using the *magnifierArea* and *magnifierGeom* GUI resources respectively.

COORDS BOX MARKER

XImtool provides a limited notion of world coordinates, allowing frame buffer pixel coordinates and pixel values to be converted to some arbitrary linear client-defined coordinate system. The coords box feature is used to display these world coordinates as the pointer is moved about in the image window.

The quantities displayed in the coords box are X, Y, and Z: the X,Y world coordinates of the pointer, and Z, the world equivalent of the pixel value under the pointer. All coordinate systems are linear. The precision of a displayed quantity is limited by the range of values of the associated raw frame buffer value. For example, if the display window is 512x512 only 512 coordinate values are possible in either axis (the positional precision can be increased however by zooming the image). More seriously, at most about 200 pixel values can be displayed since this is the limit on the range of pixel values loaded into the frame buffer. If a display pixel is saturated a "+" will be displayed after the intensity value.

The coords box is a text marker, it can be moved and resized with the pointer like any other marker. The coords box marker can be disabled by defining the *displayCoords* GUI resource, its location can be controlled by the *wcsboxGeom* GUI resource.

MARKER MENU OPTIONS

Except for the panner and WCS markers, MB3 (mouse button 3) calls up the marker menu providing a limited set of functions common to all markers:

- **o** Zoom does an equal aspect zoom of the region outlined by the marker. In this way you can mark a region of the image and zoom it up.
- **o** Fill exactly zooms the area outlined by the marker, making it fill the display window. Since the

marker is not likely to be exactly square, the aspect ratio of the resultant image will not be unitary.

- **o** Print prints the region outlined by the marker to the printer or file currently configured by the Print Panel.
- **o** Save saves the region outlined by the marker to the file currently configured by the Save Panel.
- **o** Info prints a description of the marked region. The text is printed in the Info Panel.
- **o** Unrotate unrotates a rotated marker.
- **o** Color is a menu of possible marker colors.
- **o** Type is a menu of possible marker types. This is still a little buggy and it isn't very useful, but you can use it to play with different types of markers.
- **o** Destroy destroys the marker. You can also hit the delete or backspace key in a marker to destroy the marker.

RULER MARKERS

Holding down the Ctrl key and the Left-Mouse-Button while moving the mouse will drag out a "ruler marker" measuring the distance from the initial point to the current mouse position. Releasing the Ctrl key before lifting the mouse button will leave the marker on the display, otherwise it will be erased automatically once the mouse button is released. Any number of ruler markers can be created in the frame.

Distances are measured by default in image logical pixels however the Right-Mouse-Button can be used inside the marker to popup a menu of options:

Sticky By default rulers are destroyed whenever the display changes due to a pan, zoom, flip, or frame change. This option will make the ruler "sticky" so it will not be erased, subsequent use of the menu to shows this option to be "UnSticky" to remove this feature.

Units Sub-menu to select the units of the display. If the ISM is enabled and a WCS is present in the image and selected as one of the readout options, distances may also be read out in units of arcseconds, arcminutes, or degrees instead of the default logical pixels. All markers created after the unit change will readout in the new units as their default.

Color Select the color of the marker.

Draw into Frame (*Not Yet Implemented*) Draw the marker as overlay graphics in the frame. Doing so will retain the marker when printing a hardcopy of the display.

Destroy Destroy the marker.

The marker can also be destroyed by hitting the Delete or Backspace key while the cursor is in the marker. There is presently no way to move the marker to a new position in the frame.

REAL-TIME WCS/PIXEL-VALUE READOUT

XImtool now has the ability to display the actual pixel value of an image (as well as the scaled value previously shown) and the cursor position in image WCS values (e.g. RA/DEC, GLAT/GLONG, etc). This is done using an external task (the 'ism_wcspix.e' binary in the new distribution) to access the image and pass the coordinate/pixel information to the GUI.

WCS readout is enabled by default but can be toggled or reset using the *WCS/Pix* button on the Coords tab in the control panel or the *ISM* toggle on the alt-gui menubar. When enabled, images currently in the server or subsequently displayed will be passed to the external process where they are cached for access. Cursor movements generate an event that maps the current frame buffer position to a position in the cached image. The ISM (ISM is Image Support Module) task then reads the image to determine the pixel value (or a small table of values around the current position), and computes one or more coordinates from the image position. The ISM task also has access to the associated BPM images and can optionally return bad pixel information during the cursor readout.

By default, the logical and world image coordinates are displayed to both the Coords panel readout as well as the main display window wcsbox text marker. Alternate coordinate systems (e.g. transformation of equatorial to galactic coordinates or some other sky system, physical coords, amplifier coords, etc) can be selected for display by hitting the *Options* toggle on the Coords panel. Available coordinate systems are chosen using the *Type* menu on the panel, the readout format (sexigesimal, degrees, etc) using the *Format* menu, and the display to the current panel or main image window using the remaining toggles for each WCS. Up to four systems may be displayed at one time, the coordinate panel and wcsbox marker will adjust size automatically depending on the display.

By selecting the *BPM Data* toggle from the Coords.Options panel ximtool is able to flag pixels in images with an associated bad pixel mask. This bad pixel mask is currently assumed to be named in the image header "BPM" keyword by convention. If the cursor passes over a bad pixel in the mask, the Coords bpm display as well as the main window wcsbox will change to a red background color. Only the Coords display will show the value, any non-zero value will be flagged with the color change.

With the ISM enabled the Compass indicator will display a set of arrows showing North-East if a WCS is available, otherwise just the current X-Y axes are shown. The pixel table will display actual pixel values from the image, with the ISM off the pixel table displays the scaled image values from the frame buffer.

FREEZING CURSOR READOUT

Holding down the Alt key will now freeze the cursor display readout and draw crosshairs on the screen at the last position. This can be used for example to position the cursor but then allow the cursor to be moved to another window (to enter text, start a program, whatever) without losing the position information displayed on the screen.

CUT-GRAPHS

XImtool now has the ability to display horizontal and vertical cut-graphs of the display, these appear as "flip-out" panels that appear on the bottom and right side of the main display window and are controlled by the small "H" and "V" buttons in the lower right corner of the window. When both panels are enabled the corner area of the display also shows an options panel for the graphs. Current options are:

Better Speed Draw the graphics so they update at the fastest possible rate. This is done by subsampling pixels to produce a smoother graph but without sacrificing too much accuracy.

Better Accuracy Draw the graphics using all screen pixels to produce the most accurate display. On fast modern machines this can be enabled with no apparent loss of speed, however older machines may wish to use this only occassionally to limit any lag in the cursor tracking.

Image Pixels (*Not Yet Implemented*)

Jump Cursor If enabled, large jumps of the cursor do not update the graphics display, small movements around an object of interest will update the display continuously.

Smooth Cursor If enabled, all cursor movements cause the display to be updated. This is another option that can be set safely on faster machines but will cause a delay on slower ones.

Graphics Cursors If enabled, the graphics cursors in either of the plots are active and can be used to update the cursor readout on the main image window and the complementary cut-graph. This can be used for example to freeze the cursor in the main display using the Alt key (see above), then moving to one of the graphics windows to perform cut graphs in only one axis.

Graphs are (currently) drawn using only the scaled display values to avoid complications of accessing multiple images in a mosaic display. Both plots are labeled using the frame z1/z2 values and contain cursor indicators which update contuously.

PEAK-UP CURSOR CENTROID POSITIONING

Several new keystroke commands are available to reposition the cursor to a centroid or min/max pixel value within a bounding box of the cursor position, allowing you to approximate the position with the mouse and fine tune it quickly before typing the application keystroke command. The initial box size is controlled with a *centerBoxSize* GUI resource (defaults to 5 pixels) but can be adjusted interactively using the **Ctrl-[** and **Ctrl-]** commands to descrease/increase the box size respectively. A marker will flash briefly to indicate the box size.

The **Ctrl-0** (zero) key finds either a centroid or the local maximum pixel value within this box region, **Alt-Ctrl-0** (zero) will find the local minimum value. In either case the cursor is reposition to the computed value. The default peak-up action is to find the centroid position in the box however this can be changed to find the max pixel by selection the "*Centroid Peaks*" option from the main Display control panel or by resetting the *peakCentroid* GUI resource (defaults to True).

Centroiding is done using only the scaled screen pixel values and only pixels above the mean value within the box are used. It works best if the box size is set appropriately, the centroid position may appear to drift if the box is too large and includes too many background pixels.

Command Summary

Ctrl-0 (zero)	Reposition to centroid/max-pixel
Alt-Ctrl-0 (zero)	Reposition to min-pixel
Ctrl-[	Decrease centering box size (min of 5)
Ctrl-]	Increase centering box size

Resource Summary

peakCentroid = True	Compute the box centroid position, a ’False’ value force the max value to be used
centerBoxSize = 5	Size of the centroid box, used as cursor position +/- this value

AUTO-REGISTRATION OF IMAGES

The auto-register feature allows you specify a registration of two or more display frames with an offset. When enabled, this registration is maintained for all frames in the list if any one of them is panned or zoomed to a new location in the frame buffer.

For example, to use this feature do the following:

1) Enable Auto-Register (either on the Control Panel or the toolbar on the alt-gui) and pan/zoom to some star of interest.

2) Use Mouse-Button-2 to center the star in the frame.

3) Cycle through the frames and you may see a small shift of the star. For each frame, position the cursor on the star and type **Ctrl-o** to offset it to the center. Repeat as necessary.

Small corrections will be cumulatively added so you can use the **Ctrl-0** (Ctrl-zero) peak-up command to centroid each object in the frame before the **Ctrl-o** offset.

4) Pan around the image in one display frame, then switch frames and the new frame should also be panned to the new image with the proper offset.

5) A **Ctrl-a** command will toggle the feature, offsets are only allowed when autoreg is enabled.

Hitting **Register** will zero the offsets, as will toggling the auto-register function. What you should see is the object centered in the frame and as you blink through it remains registered but the panner box marker is moving around. Drag the panner around and all frames still remain registered with the given offset. The control/info panels now display what the offset is for each frame.

The register display list is shared with the blink list and can be set using the Display control panel. By default all frames are included in the list. For accessing more than four frames, use the box icon in the Blink/Register box of the Display control panel to bring up a new window with access to all 16 available frames.

Command Summary

Ctrl-o	Set the registration offset from center
Ctrl-a	Toggle the Auto-Register feature

CONTROL PANEL

XImtool has a control panel which can be used to exercise most of the capabilities the program has for image display. The control panel can be accessed either via the **Options** menu from the main window menubar, or by pressing the leftmost button in the row of buttons at the upper right side of the display in the standard GUI (in the alternate GUI the *Control Bar* accessed by the rightmost button on the menubar provides widgets for selecting the desired control panel).

The separate windows previously used for Control/Print/Load/Save/etc have now been integrated into a single window with the appropriate control panel selectable with a Tab widget. There are also new Tab panels for setting the frame tile configuration (see below), more detailed information on the server status, and selecting the WCS readout options (see below).

VIEW CONTROLS

The **Frame box** will list only the frame buffers you currently have defined. Currently, the only way to destroy a frame buffer is to change the frame buffer configuration, new frame buffers (up to 16) will be created automatically if requested by the client. The number of frame buffers created at startup can be controlled using the *-nframes* command-line switch or the *defNFrames* resource.

The **text display** window gives the field X,Y center, X,Y scale factors, the X,Y zoom factors, and the frame offset used in Auto-Registration. The scale factor and the zoom factor will be the same unless *autoscale* is enabled. The scale is in units of display pixels per frame buffer pixel, and is an absolute measure (it doesn't matter whether or not autoscale is enabled). Zoom is relative to the autoscale factor, which is 1.0 if autoscaling is disabled. This information is also presented in the Info panel.

The numbers in the **Zoom box** are zoom factors. Blue numbers zoom, red numbers dezoom. *Zoom In* and *Zoom Out* may be used to go to larger or smaller zoom factors, e.g. *Ctrl-5* followed by "Zoom In" will get you to zoom factor 10. Specific zoom factors may also be accessed directly as Control keystrokes, e.g. Ctrl-5 will set zoom factor 5. *Center* centers the field. *Toggle Zoom* toggles between the current zoom/center values, and the unzoomed image.

Aspect recomputes the view so that the aspect ratio is 1.0. Aspect also integerizes the zoom factor (use the version in the View menu if you don’t want integerization).

Fit Frame makes the display window the same size as the frame buffer. Note that autoscale has much the same effect, and allows you to resize the display window to any size you want, or view images too large to fit on the screen.

ENHANCEMENT CONTROLS

At the top is a scrolled list of all the available colormaps. Click on the one you want to load. You can add your own colormaps to this list by defining the *cmap[12]* or *cmapDir[12]* command line flags or application resources.

The two sliders adjust the contrast (upper slider) and brightness (lower slider) of the display. The *Invert* button inverts the colormap (multiples the contrast by -1.0). Note that due to the use of the private colormap the sliders are a bit sluggish when dragged to window the display. If this is annoying, using MB3 in the display window is faster.

The *Normalize* button (on the bottom of the control panel) will normalize the enhancement, i.e. set the contrast and brightness to the default one-to-one values (1.0, 0.5). This is the preferred setting for many of the pseudocolor colortables and for private colormaps loaded from disk images. The *Initialize* button does a reset of the server.

BLINK CONTROLS

Blink frames is the list of frames to be blinked. When blink mode is in effect ximtool just cycles through these frames endlessly, pausing "blink rate" seconds between each frame. The same frame can be entered in the list more than once. To program an arbitrary list of blink frames, hit the Reset button and click on each blink frame button until it is set to the desired frame number. The main control panel allows only the original four frames to be specified in the blink list, however access to the full list of 16 frames now supported is gained using the box icon button next the the *Reset* button to bring up a new control panel.

The *Blink Rate* can be adjusted as slow or as fast as you want using the arrow buttons. If you set the blink rate small enough it will go to zero, enabling single step mode (see below).

The *Register* button registers all the blink frames with the current display frame. Frames not in the blink list are not affected.

The *Match LUTs* button sets the enhancement of all blink frames to the same values as the display frame. Frames not in the blink list are not affected.

The *Blink* button turns blink on and off. When the blink rate is set to zero the Blink button will single step through the blink frames, one frame per button press.

NOTE: You can blink no matter what ximtool options are in effect, but many of these will slow blink down. To get the fastest blink you may want to turn off the panner and coords box, and match the LUTs of all the blink frames. All the ximtool controls are fully active during blink mode, plus you can load frames etc.

OPTIONS:

Panner

Toggles whether to display the Panner marker.

Magnifier

Toggles whether to display the Magnifier marker.

Coords Box

Toggles whether to display the coordinate box marker.

Autoscale

If autoscale is enabled then at zoom=1, the frame buffer will be automatically scaled to fit within the display window. With autoscale disabled (the default), the image scale is more predictable, but the image may be clipped by the display window, or may not fill the display window.

Antialias

When dezooming an image, i.e., displaying a large image in a smaller display window, antialiasing causes all the data to be used to compute the displayed image. If antialiasing is disabled then image is subsampled to compute the displayed image. Antialiasing can prevent subsampling from omitting image features that don't fall in the sample grid, but it is significantly slower than dezooming via subsampling. The default is no antialising.

Tile Frames

The default display mode is to view one frame at a time. In tile frames mode, 2 or 4 frames may be viewed simultaneously in the display window. All the usual operations (zoom and pan, colortable enhancement, cursor readback, etc.) still work for each frame even when in tile frames mode.

Warnings

The warnings options toggles whether you see warning dialog boxes in situations like overwriting an existing file, clearing the frame buffer, etc.

Centroid Peaks

If enabled, the **Ctrl-0** keystroke will reposition the cursor to the computed centroid of the centroiding box, otherwise the cursor is repositioned to the local maximum value within the box.

LOAD PANEL

The Load Panel allows you load images from disk directly to the frame buffer, this is analogous to loading an image on the command line except that browsing is possible. At present recognized formats include IRAF OIF format (i.e. *.imh* extension), simple FITS files, GIF, and Sun rasterfiles. The task will automatically sense the format of the image and load it appropriately. Images with private colormaps (such as GIF) will be loaded using the private colormap (meaning that changing the brightness/contrast enhancements will render an apparently random-colored image), all others will be loaded with a grayscale colormap.

When loading new images the frame buffer configuration table will be searched for a frame buffer that is the same size or larger than the new image size, if no frame buffer can be found a custom buffer exactly the size of the image will be created. This means that the image may not fill the display window when loaded, or you may see a subsection of the image in the main display window. Setting the *autoscale* option on the main Display panel will scale the entire image to fit the main display window, the full frame buffer will always be visible in the Panner marker window.

Images with more colors than can be displayed will automatically be quantized to the number of available colors before display. If the *Auto Grayscale* button is enabled any image colormap will be converted to grayscale and loaded as the standard grayscale colormap.

Formats which permit pixels larger than 8-bits/pixel will be sampled on a grid to determine an optimal range in the data to be used to compute a linear transformation to the number of display colors. This is the same z-scale sampling and transformation used by the IRAF *DISPLAY* task when computing the *z1/z2* values and provides a much better initial display than simple truncation to 8-bits. This scaling will be done automatically using a grid of *Nsample* points if the *Zscale* option is enabled. Otherwise, if the *Zrange* option is set the full data range will be used to scale the image. Lastly, is neither *Zscale* nor *Zrange* are enabled, the z1/z2 values may be set explicitly using the options box.

Directory Browsing

The load panel contains a list of files in the current directory that may be selected for loading by selecting with left mouse button. If the file is a directory the contents of the new directory will be loaded, if it's a plain file an attempt will be made to load it as an image otherwise an error popup will appear. Directories in the list are identified with a trailing '/' character, you will always see any subdirectories listed even if a filter is specified.

The *Root* button will reset the current directory to the system root directory. The *Home* button will reset the current directory to the user's login directory, the *Up* button moves up one directory level, and *Rescan* reloads the file list by rescanning the directory. The current working directory is given below the file selection window.

Selecting the *List Image Headers* option will change the display text to list all images in the current directory which match the filename filter. Directory browsing is disabled while this option is in effect.

File Patterns

By default all files and directories will be listed. You may specify a filter to select only those files with a given extension such as "*.fits" using the *Filter* text box. Directories will always be seen in the list and are identified with a trailing '/' character. Any valid unix pattern matching string will be recognized, multiple templates may be specified in a comma-delimited list such as "*.imh,*.fits" to list both OIF and FITS images.

Direct File Load

If you know exactly which file you wish to load, you may enter its name in the *Load File* text box and either hit <cr> or the Load button to load it. An absolute or relative path name may be given, if a simple filename is specified it will be searched for in the current working directory.

Frame Selections

By default images will be loaded into the current frame, you may choose a different frame using the Frame menu button to select from the available frames.

SAVE PANEL

The Save Panel lets you save the current contents of the main display window to a disk file (including the Panner/Coords markers, or overlay graphics displayed by the client program). Presently, only the contents of the main display window may be saved, there is no facility for saving the undisplayed contents of the entire frame buffer other than to enable the autoscale feature or zoom out so the whole buffer is in the display window. A limited number of formats are currently available, others will be added in future versions.

File Name The File Name text box allows you to enter the file name of the saved file. A "%d" anywhere in the name will be replaced by a sequence number allowing multiple frames to be saved with unique names.

Format The Format box allows you to choose the format of the image to be created however not all formats are currently implemented. The EPS format is similar to the *Print* option however there is no annotation.

Color The Color box lets you choose the color type of the image to be created. The options will change depending on the format, e.g. FITS doesn't allow color so no color options will be enabled. Formats which allow 24-bit images will be written using the current colormap after converting to a 24-bit image, pseudocolor images will be written with the current colormap.

PRINT PANEL

The Print Panel allows you dump the contents of the main display window as Encapsulated Postscript to either a named printer device or to a disk file. The *Print To* selects the type of output, the *Print Command* box will adjust accordingly, either as a Unix printer command or as a file name. A "%d" anywhere in the name for disk output will be replaced by a sequence number allowing multiple frames to be saved with unique names. Selecting printers from the installed list will automatically change the command to be used to generate the output. This command does not necessarily need to be a printer command, the printer configuration file lets you define any command string to process the image.

COLOR OPTIONS

The Color box lets you choose the color type of the image to be created. PseudoColor or 24-bit postscript will be created using the current colormap and enhancements.

POSTSCRIPT OPTIONS

Orientation Set the page orientation.

Paper Size Select the paper size to be used.

Image Scale Set the scale factor used to compute the final image size. No checking is done to make sure the image will fit correctly on the page.

PROCESSING OPTIONS

Auto Scale

Toggles whether or not the image is automatically scaled to fit the page. If not enabled, the image scale will be used to determine the output image size, otherwise the image will be scaled down (if necessary) to fit on the page.

Auto Rotate

Determines whether or not the image will be rotated to fit on the page. When set, an image larger than the current orientation will be rotated and possibly scaled to fit the page, otherwise the image may be scaled so that it fits in the current orientation.

Max Aspect

Automatically increases the scale so the image fills the page in the current orientation.

Annotate

The annotate option toggles whether or not the final file includes annotation such as the image title, a colorbar, and axis labels. There is currently no option for partial annotation.

ANNOTATION OPTIONS

Annotate

Selects whether Postscript image is to be annotated. **Title** Annotate with a title on the top of the image. **Borders** Annotate with borders surrounding the image giving image coordinates. **Colorbar** Annotate with colorbar at the bottom of the image **Title String** Title string to use when *title* is selected. The special value *imtitle* will force the title to be the currently displayed image title, otherwise it will be this user-selected field.

PRINTER SELECTION

The printer selection list lets choose the printer to be used. The printer configuration file is /usr/local/lib/ximprint.cfg by default or may be reset using the *-printConfig* command line switch or *printConfig* resource. The format of the file is simply

name\tcommand

The *name* value is what appears in the selection list and may be more than a single word, the *command* can be any command that accepts EPS input from a pipe, the two fields must be separated by a tab character. Normally the command will be a simple *lpr -Pfoo* or some such, but can also include converters or previewers. At most 128 printer commands may be used.

INFO PANEL

The Info panel was revised to provide a greater variety of status information. The type of output is controlled by the toggle buttons on the bottom of the frame, however all output is kept current as the program runs. Current info options include:

Frame Info about the current display frame.

Server Info about various server options, e.g. colormaps, memory model, antialias type, etc.

Clients Show currently connected clients. Lists available connection channels and active ISM clients.

WCS List all WCS and mappings for the current frame.

ISM Log of various ISM status messages.

Imtoolrc Show current frame buffer configuration table.

TILE PANEL (NEW)

With the additional frames, the default tiling scheme proved inadequate. A new control panel Tile frame now allows you to select from a number of tile configurations, the list of frames to be tiled, a *fill style* (left-to-right or top-to-bottom), as well as optional labels for each of the tiles (frame number, image title or image name).

Tile configuration will make use of all frames currently selected in the *Tile Frame* group in the following manner:

Disabled Do not tile the display.

Manual Tile according to *Manual Configuration* settings.

Best Optimize layout for frame buffer aspect.

Square Always force a square layout (2x2, 3x3, etc).

Horizontal Preferentially tile horizontally (6 frames ==> 3x2).

Vertical Preferentially tile vertically (6 frames ==> 2x3).

One Row Tile all in one row (Nx1).

One Column Tile all in one column (1xN).

COORDS PANEL (NEW)

The Coords Panel is meant to provide a full-featured readout as well as serve as a control panel for the various options. The display window contains the image name/title and frame buffer info, and a selection of coordinate and image pixel readouts. The intent is provide more infor- mation than can fit comfortably on the main image window while still taking up as little screen space as possible. To this end the "Options" button is used to hide most of the feature controls when not in use (see below). Other options on the main panel include:

WCS/Pix Toggle the real-time WCS/pixel readout capability (i.e. the ISM used to access the disk image). This must be enabled for certain other options to work.

Pix Table Open a panel showing an image pixel table. The panel shows an array of pixels surrounding the cursor position, either the actual pixel values if the ISM is enabled, or scaled display values otherwise. The size of the table may be selected from the menubar.

Header Display the current image header in a new panel. Both the entire image header as well as WCS-specific parts of the header are available under different tabs. This option is only active when the ISM is enabled.

Compass Draw an orientation compass on the display panner. If the ISM is enabled and a WCS is present in the header, the compass will indicate N/E according to the WCS, otherwise the X/Y axes of the image are drawn.

Options Pop-up/down the option control portion of the panel. When enabled, the Coords Panel will change size to reveal the options which can be changed (explained below).

The "Readout Values" group controls the selection of WCS type, location and format to be displayed. The "Type" menu always provides a selection of the image Logical, Physical or World systems, which may be identical depending on the image header. If a World system is supplied in the image addition entries for transformations to other sky systems, (e.g. FK5 to ICRS or galactic/ecliptic) will also be available. The selection is dependent on whether the ISM is running as well as WCS information present in the image. The "Format" menu allows the use to select a sexigesimal display, conversion to degrees or radians, or whichever format is most natural for the coordinate being display. The two toggle to the right control whether this WCS is to be displayed on the Panel (i.e. the Coords Panel window) or the ImgWin (i.e. the text marker on the main image window).

Other options below this group control whether or not to display the WCS labels, the image name/title, and frame buffer information in the main Coords Panel display. The "BPM Data" option controls whether or not the ISM will try to map any bad-pixel mask associated with the image. If enabled, a bad-pixel mask specified by the image header BPM keyword (currently fixed by convention but this may be selectable later) will be mapped along with the image. Aside from wcs/pixel readouts at each cursor position, any BPM data values found will also be displayed. A non-zero value will cause the BPM field of the Coords Panel readout as well as the main image window marker to switch to a red background color to flag the value.

The last box allows the user to specify a different ISM task to be executed or to reinitialize the current one. In most cases this won’t need to be changed, however a custom ISM could be started when using special data formats. This command string can also be controlled by the application "ism_task" resource.

TCLSHELL

The *TclShell* allows the user to type commands directly to the TCL interpreter, letting you send messages to the object manager or execute specific procedures in the TCL code that makes up the GUI. It is used as a development or debugging tool for the GUI, but for an example of what it does, bring it up and type a command such as

send fileButton set background red

COLORMAP SELECTION

By default XImtool will display images using either a grayscale colormap (e.g. if loaded by a client), or a private colormap when loading an image from disk that contains a colormap. Each frame defines its own colormap so you can define different colormaps or enhancements for each frame, they will change automatically as you cycle through the frames.

BUILTIN COLORMAPS

Once loaded, the colormap may either be changed using the builtin colormap menu under the View menu button on the main window, or from the Enhancement box on the control panel. XImtool has about a dozen colormap options builtin, other user-defined colormaps may optionally be loaded. It is not presently possible to save colormaps for later use.

USER-DEFINED COLORMAPS

The *cmap[12]* and *cmapDir[12]* resources (or command line arguments) are used to tell which specific colormaps to make available or where to look for colortables respectively. The colortables are loaded when ximtool starts up, or when it is reinitialized (e.g. by pressing the Initialize button in the control panel). XImtool will ignore any files in the colormap directory which do not look like colortables. New colortables will also be added automatically for each image loaded from disk.

The format of a user lookup table is very simple: each row defines one colortable entry, and consists of three columns defining the red, green, and blue values scaled to the range 0.0 (off) to 1.0 (full intensity).

```
R G B
R G B
(etc.)
```

Blank and comment lines (lines beginning with a ’#’) are ignored.

Usually 256 rows are provided, but the number may actually be anything in the range 1 to 256. XImtool will interpolate the table as necessary to compute the colortable values used in XImtool. XImtool uses at most 201 colors to render pixel data, so it is usually necessary to interpolate the table when it is loaded.

The name of the colortable as it will appear in the XImtool control panel is the root name of the file, e.g., if the file is "rainbow.lut" the colortable name will be "rainbow". Lower case names are suggested to avoid name collisions with the builtin colortables. Private colormaps for disk images will be have the same name as the image loaded. If the same colortable file appears in multiple user colortable directories, the first one found will be used.

MINIMIZING COLORMAP CONFLICTS

The Gterm widget used by XImtool (i.e. the main display window) uses a private global colormap for display, this allows it to have greater control over color cell allocation but can occasionally also cause "colormap flashing" as the mouse is moved in and out of the application. The problem here is that in a system with only an 8-bit colormap (256 colors) all applications must compete for colors, programs such as XV or Netscape allocate colors from the default colormap leaving only a few free cells for XImtool. Since XImtool defines a private global colormap it is still able to allocate the needed cells rather than failing, but it’s allocating cells already used by other applications. As the mouse moves out of the ximtool window those cells are once again defined in terms of the default colormap, so the ximtool window is then using a different colormap. It is this switching of the colormap context that causes the flashing to occur, but there are a few things that can be done to help minimize this.

XImtool logically defines 200 colors which the client image display program can use to render pixels. However, ximtool may or may not actually allocate all of those colors. By default it currently allocates only about 192 colors, to reserve 64 colors for the other windows on the screen. You don’t normally notice this as 1) usually the default screen colormap has enough free cells to allow ximtool to match the colors, and 2) the extra unallocated cells correspond to the brightest pixels in the rendered image, and these colors may not be used or usually only correspond to a few small regions near the saturated cores of bright objects.

You can eliminate this problem by setting the *basePixel* resource to e.g. 48 instead of 64, which will let the gterm widget allocate all 200 colors. However, this isn't recommended for normal use as it will increase the likelihood of colormap flashing. If you change *basePixel*, either restart the X server or set the resource *cmapInitialize=True* to force the gterm widget to update its global colormap resource in the X server. The colormap resource may also be deleted by using the command

xprop -root -remove GT_image

These options may also be set on the command line when first starting up.

In general one can set the Gterm widget resources *basePixel* and *maxColors* to specify the region of colormap space to be used for image display. If you set *maxColors* to a small value, the 200 logical colors defined by the widget will be mapped by the imtool color model into whatever number of colors are actually available to the widget. For example, in the default setup, 200 color values are really being mapped into 192 color cells used for display, the remaining colors are used for buttons, menus etc and are allocated from the default colormap by the X toolkit when the application starts up.

Even though the Gterm widget uses a private colormap, it is a private *global* colormap meaning that all Gterm widgets share the same colormap. An example of colormap sharing in ximtool is the main image window and the colorbar window. These are two separate gterm widgets that share the same colormap. They have to share the same colormap, as otherwise when you windowed the main image window the colorbar window would not accurately reflect the modified colormap. By default two separate ximtools would also share the same colormap meaning contrast enhancements in one window would affect the other. By resetting the *cmapName* command line option or resource you can change the name of the private colormap used causing separate ximtools to use different colormaps, but note this also creates colormap flashing between the two windows that cannot easily be avoided. By setting the *cmapName* to "default" the widget will allocate colors from the default colormap, but this is of little use at the moment.

There are a number of other resources that can be used to modify the behavior of the Gterm widget color management scheme, but these are the most useful ones. For question and further information feel free to contact *iraf@noao.edu*.

DISPLAY CLIENT CONNECTIONS

XImtool allows display clients to connect in any of the following ways:

fifo pipes

The traditional approach. The default global /dev/imt1[io] pipes may be used, or a private set of fifos can be specified using the *-fifo* command line argument or *∗fifo* resource. Values should be specified as the root pathname to a pair of fifo pipes whose last character is 'i' or 'o', these characters will be added automatically when opening the pipes. For example, to use the default pipes the path would be specified as simply "/dev/imt1". A value of "none" disables this connection.

tcp/ip sockets

Clients connect via a tcp/ip socket. The default port is *5137*, or a custom port may be specified using the *-port* command line switch or a *∗port* resource. This permits connecting to the server over a remote network connection anywhere on the Internet. A port number of 0 (zero) disables this connection.

unix domain sockets

Like a tcp/ip socket, but limited to a single host system. Usually faster than a tcp/ip socket, and comparable to a fifo. By default each user gets their own unix domain socket, so this option allows multiple users to run ximtools on the same host without having to customize things. The default value is "/tmp/.IMT%d", other sockets may be defined using the *-unix* command line switch or the *∗unixaddr* resource. Legal values should be specified as a filename to be used for the socket, up to two "%d" fields are allowed and will be replaced by the userid. An empty string value disables this connection.

By default ximtool listens simultaneously for client connections on all three types of ports. Clients may connect simultaneously by different means allowing up to three different displays to be loading at the same time into different frames.

COMMUNICATIONS PROTOCOL

The communications protocol used is a slightly modified version of that used by the IIS Model 70; other more modern protocols will likely be supported in the future. The IIS protocol is basically a command packet stream with a header describing the operation to be performed (select frame, load display, read cursor, etc), and an optional data packet containing e.g. pixels.

Beginning with XImtool V1.3 the protocol was modified even more to allow extra text at the end of the WCS string to define image mappings and to better support multiple world coordinate systems within a frame. For backwards compatability none of the existing IIS protocols were modified completely, however we take advantage of unused registers to flag the new features in existing functions (like read/write WCS). The WCS mapping changes required only that the unused ’x’ register be set to indicate the new behavior was desired, e.g. the wcs text containing the extra mapping data.

We also added two new WCS calls that allow us to query the WCS version, or query a WCS by a specific number corresponding to a mapping. The WCS version query will return a string such as "version=10" which can be parsed by the client to get a version number ’10’ (corresponding to version 1.0).

Because of the added mapping text the WCS string length was increased from 320 to 1024 bytes, the string length used internally depends on whether the ’x’ register has been set.

Support for the full 16 frames allowed by the bit-flag ’z’ register in the IIS header packet required the masking values be changed at various places in the code. This was more a limitation of the initial implementation than a required change to the protocol.

A complete summary of the XImtool IIS protocol implementation follows.

IIS PROTOCOL SUMMARY

All operations are initiated by sending a header packet containing a *thing id* (tid) and *subunit* selecting the function to be performed, optionally followed by data up to 32Kb long. The IIS header packet used is defined as

```
struct  iism70 {
      short   tid;
      short   thingct;
      short   subunit;
      short   checksum;
      short   x, y, z;
      short   t;
};
```

The *thing count* field contains the negative number of bytes of data that will be sent following the header packet. The IIS header checksum is computed as

```
checksum = 0177777 - (tid + subunit + thingct + x + y + z + t);
```

The four IIS registers are set differently depending on the operation, a summary of the header packets for each operation is summarized below.

IIS Header Packet Summary

	TID	Subunit	Tct	X	Y	Z	T	Data
Read Data	IIS_READ|PACKED	MEMORY	-NB	x	y	fr	-	NB
Write Data	IIS_WRITE|PACKED	MEMORY	-NB	x	y	fr	-	NB
Read Cursor	IIS_READ	IMCURSOR	-	-	-	-	-	-
Write Cursor	IIS_WRITE	IMCURSOR	-	x	y	wcs	-	-
Set Frame	IIS_WRITE	LUT|COMMAND	-1	-	-	-	-	2
Erase Frame	IIS_WRITE | fb	FEEDBACK	-	-	-	fr	-	-
Old Write WCS	IIS_WRITE|PACKED	WCS	-N	-	-	fr	fb	320
Old Read WCS	IIS_READ	WCS	-	-	-	fr	wcs	320
WCS Version?	IIS_READ	WCS	-	1	1	-	-	320
WCS by Number?	IIS_READ	WCS	-	1	-	fr	wcs	1024
New Write WCS	IIS_WRITE|PACKED	WCS	-N	1	-	fr	fb	1024
New Read WCS	IIS_READ	WCS	-	1	-	fr	wcs	1024

Where	NB	= number of bytes expected or written
	x	= x position of operation in frame buffer coords
	y	= y position of operation in frame buffer coords
	fr	= frame number (passed as bitflag (i.e. 1, 2 ,4 8, etc)
	fb	= frame buffer config number (zero indexed)
	N	= length of WCS string
	wcs	= WCS number (usually zero)
	Data	= the number of bytes of data to be read or written following the header packet.

IIS_WRITE	= 0400000
IIS_READ	= 0100000
COMMAND	= 0100000
PACKED	= 0040000
IMC_SAMPLE	= 0040000
MEMORY	= 001
LUT	= 002
FEEDBACK	= 005
IMCURSOR	= 020
WCS	= 021

TID fields can be logically OR'd with the PACKED flag indicating the number of data bytes is exactly *thingct* bytes long, otherwise *thingct* must be specified as half the number of data bytes. In a cursor read, if the IIS_READ flag is OR'd with IMC_SAMPLE the logical cursor position (i.e. the last value read or set) is returned immediately, otherwise the server will wait for a keystroke to be hit before returning a string containing the (x,y) position, wcs of the read, and the keystroke. When setting the frame you must send a short integer in the data containing the frame selected.

ISM COMMUNICATIONS

The ISM (Image Support Module) can be any external task which connects to XImtool over a socket. Communications are limited to simple null-terminated text strings. In most cases these strings are just the standard OBM messages sent to XImtool objects but can also include Tcl callback code (either ISM-specific callbacks, procedures which can be added to the callback list for existing XImtool objects, or even new GUI code to create panels and new objects).

ISM SOCKET CONNECTION

The ISM first requests a connection to XImtool on a dedicated socket whose default value is "/tmp/.ISM%d", where the ’%d’ is replaced by the userid allowing multiple users on a machine to have independent sockets. The XImtool ’ism_addr’ resource or "-ismdev" command-line option can be used to change this address, a value of ’none’ will disable ISM communications. The socket may also be set with an ISMDEV environment variable which will override the resource or command-line options.

Once a connection request is received, XImtool replies with a message telling the ISM to reconnect on a different socket, it then frees the initial connection allowing multiple other ISMs to request their own connection. The communications between XImtool and the ISM are carried out entirely over this second negotiated socket. Once connected, the ISM appears as just another named object which can receive OBM messages.

COMMUNICATIONS PROTOCOL

Messages from the ISM are written to the connection socket and must be preceeded by one of the following keywords:

callback	Negotiate a connection on another socket
ready	Client is ready to begin processing
quit	Client is shutting down and disconnecting
send	Send a message to another object

Where messages are of the form:

connect <*name*>	Request a connection for the <*name*> ISM
ready <*name*>	Reconnection request for the <*name*> ISM on negotiated socket, ISM is ready to processing.
send <*obj*> ’{’ <*msg*> ’}’	Send <*msg*> to the named <*obj*>. The message may be any valid string that will be understood by the recipient. The object may be any object in the GUI or OBM (see below).
quit	ISM is shutting down. The named is determined from the communications channel, ISM is responsible for any cleanup of it’s callbacks before issuing the shutdown.

All messages must be null-terminated. XImtool will buffer the text until a complete message is received. Once an ISM client has delivered a QUIT message no further messages will be sent the that ISM.

In OBM terminology the ISM is a named Client class object, where the name is set in the connection request. Messages sent to the ISM should use this name, messages sent to "client" are still interpreted to mean the XImtool client.

The content of messages delivered to the ISM are totally free-form and may contain any text the ISM is expected to understand.

GUI OBJECTS

While the ISM can send a message to any object in the task, there is a GUI Parameter object called ’ism_msg’ designed especially to process messages from the ISM. The callback in the GUI is expecting a message beginning with one of the following keywords:

source	Source message text as Tcl code
alert	Message contains error text to be displayed in the GUI ’alert’ box

deliver Message text should be passed to a callback routine specific to that ISM. This processing callback may have been previously uploaded. The message text may be any form the processing callback is expected to understand.

info Message text is status output intended for the XImtool ’info’ panel (connect/disconnect requests, etc)

In all cases the message is expected to be of the form

> *<cmd> <ism_name>* [*<arg1> <arg2>* <...>]

where <cmd> is one of the above keywords, <ism_name> is the name of the ISM sending the message. The remainder of the message is passed as an ’argv’ list to the processing callback uploaded for the ISM. The ISM is responsible for formatting these messages.

ENVIRONMENT

HOME	Specifies user login directory
DISPLAY	Specifies which display screen to use
IMTOOLRC or imtoolrc	Frame buffer configuration file
ISMDEV	ISM Connection socket
DEBUG_IIS	Debug IIS communications packets
DEBUG_ISM	Debug ISM communications packets
DEBUG_MAPPINGS	Debug WCS image mappings

FILES

/usr/local/lib/imtoolrc	Default frame buffer configuration file
/usr/local/lib/ximprint.cfg	Default printer configuration file
/usr/local/lib/imtoolcmap	Default colormap directory
/dev/imt1i	Default input display fifo
/dev/imt1o	Default output display fifo
/tmp/.IMT%d	Default unix display socket
/tmp/.ISM%d	Default unix ISM connection socket

BUGS

Users should report bugs to *iraf@noao.edu*.

SEE ALSO

xgterm(1), xtapemon(1)

COPYRIGHT

Copyright(c) 1986 Association of Universities for Research in Astronomy Inc.