NAME

bit – Bitmap Image Touchup program for SGI Workstations

SYNOPSIS

bit [**–vfabiIdenhCRrq –c** *file* **–s** *sec* **–p** *path* **–x** *f* **–y** *f* **–m** *level*] [files]

DESCRIPTION

bit is an interactive full color image viewer and editor based on Silicon Graphics GL.

As an image viewer, *bit* allows list of images to be viewed in their original type (24bits or color index) in any order and in any of the many built-in styles of display. With command line option –IN, *bit* simulates and combines the functionalities of *ipaste*(1G), *gifpaste*(1L) and *jpegpaste*(1L). Further, with the pan and zoom features, large images can be viewed in full without being limited by the window or screen size. Slideshow is also supported.

As an image editor, *bit* performs a large number of image editing and processing tasks accurately and efficiently. It keeps information loss at any stage of the editing process at an absolute minimum by performing subpixel sampling automatically while conserving the input image type whenever possible to avoid unnecessary promotion and/or quantization of input images. The paint function features brushes, free hand drawing, fill and many built-in patterns, all with 24bits color. In addition, arbitrarily colored and sized text and simple geometric figures can be rendered on top of an image for annotation with the option to make them part of the bitmap or output separately to take advantage of the higher printer resolutions.

Bit can also be used as a graphical user interface to an existing image processing program by defining an external binding to it which in essence turns a command line oriented program into a *bit* subfunctions and can be accessed by a press of a button, and best of all, the processed image will be read back and displayed immediately. Convolution using externally defined matrices of arbitrary order can be performed dynamically giving great flexibility in processing an image.

Upon startup, *bit* does the following and then waits for further command:

1. Reads the initial window position specification file *WinPosition* from user directory $HOME/.bitrc unless the command line option **–a** (Auto), **–F** (Full screen) **–i** (fit first Image) or **–I** (always fit Image) are present. See **Configuration** for details.

2. Reads the configuration file *Setup.opt* from system directory /usr/local/lib/bit and user directory $HOME/.bitrc in that order. This step is skipped or modified by the command line option **–n, –p** or **–c.**

3. Reads the command line filenames in order if present, and placing these files on a list (the *FileList*) for future use.

3a. Opens a tiny window on the upper-right corner of the screen with a clock and a mailbox if flag –C is present.

4. Opens the main window, i.e., the window in which images will be displayed, and clears the window to a pre-determined background color. The window position and size are read from the file *WinPosition* if –i, –I, –F and –a are not present.

5. Opens the file control panel (the *FilePanel*) on the lower-right corner of the screen (or the position specified in *WinPosition* file if flag –N is not present. The *FilePanel* is the most basic control panel, consisting of command buttons to set program options and to handle file IO.

At this point, if the *FileList* is not empty, the first file on the list will be loaded and displayed in the main window.

To display next file on the list, click *leftmouse* in the main window or press the *SPACE* key;

To display the previous file, click *rightmouse* in the main window or press the *p* key;

And to display the current image (again), click *middlemouse* in the main window or press the *c* key.

Besides adding files to the *FileList* via command line arguments, files can also be added or removed from the *FileList* by the *FilePanel* button labeled *load*, See Command Summaries for details.

To pause an indefinite interval at any point, press *PAUSE* key. To re-activate, press *PAUSE* again.

To exit *bit*, click the *Quit* button on *FilePanel* or press <ESC> key anywhere.

FEATURES

Bit performs all of its image manipulations interactively with a continuous progress report, and once the processing is complete, the processed image will be displayed. In most cases, transformation is reversible and a simple undo will recover the un-processed image.

Current features include

- Rotates an image by an arbitrary angle with the option of anti-aliasing and filling the "rotated out" region with an arbitrary color.
- Scales an image by arbitrary factors in x- and/or y-directions with the option of subpixel sampling. It can also scale the image to any requested size while keeping the aspect ratio by filling the empty region with an arbitrary color.
- Renders text of arbitrary color, size or orientation on top of an image with the option of rendering directly into the raster or saving the text and raster separately, such as into a PostScript file, to take advantage of a higher printer resolution. Besides normal text strings, many predefined special symbols can be input conveniently as in TeX(1), e.g., α\small$\beta$ produces Greek alphabet α followed by a slightly smaller Greek alphabet β Or you can input the character code directly if the font encoding scheme is known, e.g., for PostScript Symbol font, $\343$ produces the copyright symbol ©
- Show and manipulate the histogram of an image, including histogram equalization and direct histogram specification (not available in this release).
- Performs arbitrary 1-to-1 pixel transformation. The transformation function can be of arbitrary form and can be specified interactively by mouse or by arbitrary math expressions. Further, the transformation can be applied to RGB channels separately or simultaneously and to the entire image or a portion of it.
- Modify the RGB values of an individual pixel or a group of pixels.
- Places simple geometric figures (SGFs), e.g., arrows, circles, triangles etc., of arbitrary color, size and orientation into an image at arbitrary locations addressed either in absolute pixels or in some user definable coordinate system units, with the option of rendering directly into the raster or save the sgfs and raster separately (e.g, PostScript file). All SGFs can be scaled in x- and y-directions independently or simultaneously (this is how to get an ellipse from a circle).
- Full color paint with many built-in tools, which include brushes of many different shapes, air brush with adjustable nozzle size and flow rate, free-hand polygon, free-hand drawing, fill and many more.
- Crops a rectangular portion of an image, or add a solid frame to an image. The crop region can be larger than the window/screen size.
- Merges and/or concatenates several images to form a single image, with optional add/sub/diff/xor/mask operations on the fly.
- Cuts and pastes across images.

- Computes the pixel by pixel difference between two identically sized images and displays the results as an image, whose histogram provides quantitative information.
- Performs edge detection. In combination with image histograms and 1-to-1 transformation, very accurate result can be obtained.
- Performs convolution with externally defined (square) matrices of arbitrary order.
- Smoothes an image or a portion of it with thresholding.
- Sharpens an image or a portion of it.
- Median filtering an image or a portion of it.
- Magnifies any portion of an image by any amount.
- Displays a list of images in sequence with a user specifiable pause interval between images. (*SlideShow*)
- Measures RGB intensities of a pixel at any location or all pixels along an arbitrary line with the result presented as RGB intensities or as an intensity vs. distance plot.
- Performs image type conversions.
- Converts image formats.
- Filters an image by external programs and read the filtered image back and display it.
- Performs FFT on the entire image or a portion of it and display the resulting power spectra.

OPTIONS

The following command line options are interpreted by *bit* program and override any program defaults, including options set in the start up file.

–v Prints the current *Version* number of the *bit* program and exits. No graphics required.

–f Prints an enumerated list of all supported image *Formats*. No graphics required.

–a By default, *bit* searches file ˜/.bitrc/WinPosition for initial window positions. This flag sets initial location to *Automatic,* which is about 80% of the screen size.

–i This flag forces the initial window size to be that of the first image.

–I This flag forces the window size to be that of the current image.

–b prevents *bit* from running in Background so that compound command like *bit; other_command* can be guaranteed to run in sequence.

–F starts bit with main window occupying the entire screen(*FullScreen*).

–N starts bit with No initial control panels. Once bit started, you can get the FilePanel with m key.

–C starts bit with Clockmailbox window.

–e Brings up the *Editing* control panel automatically upon startup.

–n Instructs *bit Not* to read initialization file *Setup.opt.*

–c *filename*

Changes the initialization file name to *filename*. Default is *Setup.opt.*

–h Prints a brief *Help* message summarizing all command line options and what they do.

–d *ds*

Specifies the Display style. Valid styles are numbered from 0 to 16. Style 100 is magic in that it will cycle through all styles.

–s *sec*

Enables *SlideShow* mode where images are displayed one by one with *sec* being the pause time, in seconds, between each display. Due to resolution limit of the system and various system delays, a value less than 0.1 second is not terribly reliable (you can specify a value less than 0.1 second, however). Note any keyboard or mouse input takes precedence over the pause time specified, and in particular, keyboard s turns off the SlideShow.

–R Suppress progress report.

–r Instructs the *bit* program not to *Refresh* screen before displaying an image. The default is to refresh the screen.

–m *level*
Sets *Message* output level to *level*, with level loosely defined between 0 and 4. Level 0 suppresses all messages except for error messages; Level 1 will allow warning messages to be printed in addition to error messages; level 2 may be used to print some additional information, such as values of key parameters or transparent and recoverable failures. Level 3 and 4 are for debug and trace information. Note that level 3 and 4 may or may not be available depending on the installation. Default message level is 1, i.e., print error as well as warnings messages.

–q *Quiet* mode, same as -m0.

–x *fx* Where *fx* specifies the zoom factor in the *x* direction. Only the integral part will be used. A zero can be used to signal "AutoZoom" in which the zoomed x size will fit the screen and in addition, automatically sets the y zoom factor as to keep the aspect ratio.

–y *fy* Vertical zoom factor.

–p *dir*
Replaces the default *Path*, ˜/.bitrc, with *dir* for all user files, e.g., configuration, external bindings, and symbol definition etc. Note this flag does not prevent *bit* from searching the system default path, /usr/local/lib/bit, for installation specific options.

Important Commands

NOTE Typical manuals would give detailed command summaries here. Due to the length of the command summary and availability of the context sensitive online help in addition to the LaTeX document for *bit*, the command summary is presented in a later section so that more important comments can be given for easy reference.

Online Help

In almost all editing modes, up-to-date help is available, typically by selecting pop-up menu entry *help* if that particular mode is controlled by the pop-up or by clicking the mouse on the empty region of the control panel where no other objects such as buttons etc exist that might interact with the mouse.

Online help is presented in a form of a browser. You can scroll through the page by the mouse. In help mode, all control panels are deactivated and no interaction can take place until you exit from help mode by pressing the button on the title of the browser.

Special Handling

ClockMail interacts with the *bit* program through user signal 2. This interaction is provided for emergencies (whatever that might be) or when *bit* is not responsive, press <ESC> inside the ClockMail panel kills all *bit* related processes cleanly. Pressing Q inside the panel removes this panel.

Loading/Removing files

clicking mouse on the *load* button in the *FilePanel* will bring up a browser filled with files in the current directory that match the current selection pattern, with highlights on files that are currently on the *FileList*. To add or remove files, click the mouse on the filenames. You can walk the directories trees, either by selecting the desired directory (marked with a D in front) in the browser or by typing the directory name into the box labeled *dir*. (you might need to activate the input field by clicking the mouse on it first). Files in the browser will be updated to reflect the directory change. If the files to be added or removed from the *FileList* have common patterns, the *pat* input comes in handy: Simply type into the *pat* box the common pattern (regular expressions) and only filenames that match the pattern will be shown in the browser. To select all files shown in the browser, click the mouse on the *LoadAll* button and to de-select all files in the browser, click the mouse on the *UnloadAll* button. Click *Done* button to exit file selection. This will be enough to get you started viewing files in any order.

Generating Output

To write current image to disk, clicking the *Write* button in the *FilePanel* will bring up a panel listing all *bit* supported formats and a browser filled with all files in the current directory (subject to pattern matching with the current pattern). The default filename is the current filename with a possible different extension. Filename can be changed either by selecting it from the browser or by typing it into the field labeled *filename*.

To choose the desired format to write, click the mouse on the button that has the format name on it. Upon each new selection, several things will take place: First, the filename is modified to reflect the current selection by adding or modifying the extension, and if it already exists, a highlight is added to the name in the browser to warn about possible overwriting; Next, current settings for writing, if any, are shown in the box labeled *options*. If there is none, the *option* box shows the word "NONE". To change default settings, press the mouse on the option box. Note the options mentioned here are those that are specific to an image format, such as the Q-value in *jpeg* or interlace in *gif* etc. Image type conversion is transparent and taken care of automatically, e.g., requesting a *jpeg* with a gray image (either in colormap or rgb representation) will produce a single channel *jpeg* file with 8bit/sample.

Click the *OK* button to initiate output. If successful, the *write* control panel disappears.

SUPPORTED FORMATS

bit currently supports the following image formats

- IRIS RGB, native to SGI IRIS
- JPEG(JFIF) format
- MPEG Movie (ReadOnly)
- CompuServ GIF
- PNM(Portable Any Map), including PPM, PGM and PBM.
- XBM (X Window Bitmap)
- TIFF
- PostScript
- Sun Rasterfile

In addition, compressed version of aforementioned formats will be automatically decompressed and loaded.

CONFIGURATION

bit is very configurable. You can change any or all of the following flags either interactively or by modifying the *Setup.opt* file directly. Flags are case insensitive and their values can be either string or numerical. Use colon (:) to separate the flag and its value, e.g., both

DisplayStyle: 14
and
DisplayStyle: Sparkle

are valid and select the display style *sparkle*.

In the following listing, the numerical value is given in parenthesis. The best way to set up a configuration is to change the settings interactively and then write out the current settings to a file and name that file to be the default start-up file.

DisplayStyle

Many different styles are available to display an image. The simplest and fastest, also the default, is the *Block* style, which simply displays the image in one scoop using *lrectwrite*(3G) or *rectwrite*(3G) depending on the image type. Other styles are enumerated as follows:

Options	Effects
Block[0]	display in one scoop
Down[1]	from top to bottom
Up[2]	from bottom to top
Left[3]	from left to right
Right[4]	from right to left
DownUp[5]	from top and bottom toward center
UpDown[6]	from center to top and bottom
LR[7]	from left and right toward center
RL[8]	from center to right and left
RInt[9]	RowUniform interlace
CInt[10]	Column uniform interlace
Gint[11]	Interlace, GIF style
Rect1[12]	rectangular, towards center
Rect2[13]	rectangular, from center
Sparkle[14]	sparkle fade
Fade[15]	blending fade
Cycle[100]	cycle through all styles

Note Fade is applicable to RGB images and is ignored if the image is in CI.

DoubleBuffer

Turn (hardware) double buffering on and off. If hardware is not capable of double buffering, this flag has no effect. Valid options are Off[0] and On[1].

AlwaysClearScreen

The screen by default is cleared between successive images. To leave the screen as it is between images, toggle this flag (useful for animation). Valid options are No[0] and Yes[1].

AlwaysUseBorder

All windows, except for the main window and the initial *FilePanel*, do not have borders by default. To force a border, set this flag to Yes. Note this flag does not take effect immediately, i.e., it does not affect panels already on screen. Turning a panel off and then on again will reflect the settings of this flag.

SlideShow

Enable or disable the *SlideShow* mode. Numerical values given should be in seconds. 0 disables slideshow.

ProgressReport

How frequent to report progress in percentages. Built-in percentages are 2, 5, 10 and 20. A value of 0 turns the report off. Note however, cursor rotating can not be turned off.

PSResolution

Number of pixels to use in rasterizing PostScript files via Ghostscript interpretor *gs(1)* or psrender. Default is 96DPI.

AutoPan

Controls if to pan an image automatically if it is larger than that can be shown. Valid options are Never[0], SlidesOnly[1], and Always[2]. SlidesOnly means to pan automatically if the current display mode is SlideShow.

ShowCutBuffer

Controls if to show current cut buffer while looking for pasting locations.

PreserveWMColors

Controls if to preserve the colors window manager uses to minimize colormap flashing. *bit* does this by re-ordering the colormap and in certain cases, where the color index has specific meanings, this might not be desirable. Valid settings are No[0] and Yes[1].

AlwaysReadScrn

Cropping is normally done via internal memory copy to avoid reading a dithered framebuffer (unless there are text and SGFs on screen and in that case, screen read is performed and after that, the text and SGFs are deleted). Set this flag to Yes to override the default for special effects or for speed on better equipped machines.

AutoCropFill

Meaningful when the crop region is larger than image. For regions outside the image, you have the option to read the screen as it is, fill the region with a solid color, or interactively choosing one of the two. The valid options are No[0], Ask[1] and Yes[2].

AutoRemoveIcon

Controls if to delete image browser thumbnail images when original images no longer exist. The actual deletion could happen either at the time original image is deleted via Delete button on ControlPanel or when a directory is browsed.

QuantizationMethod

Three methods are implemented. The default is the Q256(1), which is very effective for reducing the image colors down to 256 colors (or very close to but less than 256 colors). General purpose but slower quantization method is the MedianCut[2]. Octree[3] is for demonstration purposes only. If the setting is *Ask[0]*, the method will be prompted interactively. Note in this pre-release, only Q256 is available. However, general quantization can be accomplished by using external bindings to *djpeg*(1) or *ppmquant*(1)

QuantDither
: Whether allow a dithering step after quantization. Valid options are No[0] and Yes[1]. Default is Yes.

HalftoneMethod
: Controls the method to use in converting images to B&W. Valid options are Ask[0], Ordered8[1] and FS[2]. If this flag is set to Ask, the method will be prompted interactively.

SimuDblBuffer
: Turn simulated double buffering on and off. If the hardware is capable of double buffering and *DoubleBuffer* is on, this flag has no effect. Valid options are Off[0] and On[1].

SimuOPDblBuf
: Turn simulated buffering in overlay/pupdraw on and off. This is only used in Marking. If the hardware has no overlay bitplanes, the simulation might not be correct in pupdraw.

AutoChangeExt
: Change or append an extension to filename according to the requested format when writing. Valid options are No[0] and Yes[1].

AutoRemoveFromList
: If for some reason, a file can not be loaded, by default, the file is removed from the *FileList* upon confirmation. This flag modifies the default. The valid options are No[0], Ask[1] and Yes[2].

ReportMouseLocation
: Controls whether to report mouse location in certain situations and if yes, select origin to which the distance is measured. The valid options are No[0], Window[1], Image[2]. Option Window sets the origin at the lower-left corner of the main window while option Image sets the origin at the lower-left corner of the image.

MessageVerbosity
: Controls the amount of messages to print. Valid options are Error[0], Warning[1], Info[2] and possibly Debug[3] and Trace[4] depending on the compiler option when *bit* was built.

KeepTextAndSgfs
: The duration of text and SGFs in deferred mode is by default limited to a particular image or up to when you do a cropping. Setting this flag to Yes overrides the default so that text and SGFs can be carried to the next image.

DelFromDiskConfirm
: When deleting files from disk, a confirmation will be prompted for by default. To turn confirmation off, set this flag to No[0].

ShowEditAuto
: Whether bring up the editing panel automatically upon startup. Valid options are No[0] and Yes[1].

SmoothLines

Turn on and off anti-aliased line drawing in marking and paint. If hardware is not capable, this flag has not effect. Valid options are No[0] and Yes[1].

Initial Window Positions

Initial window position and size are read from file $BITDIR/WinPosition unless command line option –a, –i, –I or –F is present. In file *WinPosition*, 8 numbers should be specified. The First 4 numbers are the main window parameters which specify the coordinates of the lower left corner of the main window and its width and height. The rest 4 numbers are the InfoPanel and ControlPanel locations. Note the size of the InfoPanel and ControlPanel can not be changed. The following is a valid specification (# is the comment character)

```
10 10 800 600  # main window size 800X600 with ll corner at (10,10)
820 10         # info window location
820 180        # edit panel location
```

You can interactively shape and move the windows and then write out the position via *SetUp* button on *InfoPanel*.

ENVIRONMENT VARIABLES

BITHELP

The directory name where online help, icon and system wide configuration and control files are located. Default is /usr/local/lib/bit. All configuration and control files in this directory are subject to overrides by those in BITDIR.

BITDIR The directory name where the personalized configuration file and external binding files are located. The default is $HOME/.bitrc. For any needed files, except for online help, *bit* will search BITHELP first, followed by current directory or BITDIR. If command line option –p is present, that directory or file will be searched or read instead of BITDIR. However, current directory will always be searched

TMPDIR

The directory where temporary files reside. Default is /usr/tmp.

COMMAND SUMMARIES

Some Definitions

A mouse click, unless explicitly specified, means the press and release of either the *leftmouse* button or *middlemouse* button. Menubutton always refers to the *rightmouse* button.

Dragging the mouse means moving the mouse while one of the mouse button is pressed.

A rubber band as used in *bit* refers to a simple geometric figure, typically a rectangle, whose size, location and possibly shape can be changed by mouse clicking or dragging.

A control panel (also known as dialog box) is a window with many built-in objects whose functions are to modify the attributes of an object, such as text color, or to modify task parameters, such as the scaling factor. Some of the control panels have no borders by default, to force a border, turn on the *AlwaysUseBorder* flag either through Setup button or ~/.bitrc/Setup.opt.

Unless explicitly specified, a keyboard input means the specific key, not the ASCII code, thus keyboard command is case insensitive.

Editing Panels

The edit control panel, *EditPanel*, can be toggled on and off by clicking the mouse on the *edit* button on the *FilePanel.* Command line option –e can be specified to bring up the control panel automatically upon startup.

To activate a specific editing mode, click the mouse on the button labeled by the function.

Type Conversions

bit internally works either in RGB mode or in colormap mode depending on the input image type. Typically, there is no need to convert type explicitly because *bit* converts image types transparently if it needs to. For unknown reasons, I decided to have these goodies available, maybe someone might find them useful or interesting to play with. Note that term RGB for image type does not mean that the image is in color. Rather, it refers to the current pixel representation. All type conversions are initiated by clicking the mouse on the desired types shown in *EditPanel*. The following is a summary of all types explicitly supported:

Color(RGB)
: Convert current image to RGB image. This is meaningful only if current image is in colormap mode. The converted type will be in color (as far as *bit* is concerned. Note that prior to version 0.74, the converted type is in Gray(RGB) if the input image is in grayscale). Conversion from Gray(RGB) to Color(RGB) is a simple operation that does not involve actual pixel manipulations. However, the conversion could manifest itself when generating output, e.g., writing a Gray(RGB) image into *JPEG* is very different from writing a Color(RGB) image even if the images are identical (except for the type).

Color(Map)
: If the current image is in RGB, this conversion will initiate quantization, whose parameters can be set interactively. If quantization is performed for the sole purpose of saving disk space, don't do it. JPEG is much better at that than quantization and JPEG compression retains much more information as well.

Gray(RGB)
: Convert current image to RGB gray regardless what the original type is.

Gray(Map)
: Convert current image to colormapped grayscale. Quantization will be invoked if necessary.

B&W
: Convert current image to black and white. Two methods are available, one is the ordered-dithering of order 8 (16 X 16 matrix), and the other is the Floyd-Steinberg error diffusion method which typically gives better looking results. B&W image can also be converted from other types by the 1-to-1 transformation with a step function.

Misc. Command

In the main *EditPanel*, two buttons labeled *Convolve* and *Filters* are handles to external bindings (i.e, external programs or matrices). To execute a binding, press the mouse on the *convolve* or *filter* button and if there are indeed any bindings defined, a panel contains all the definition will be shown, click on the bindings you want to execute, and *bit* will try to pipe the current image through the filter, and if successful, the results will be read back and displayed.

The definition files must be named *BIT_filters* and *BIT_convolve* respectively and they are searched in ˜/.bitrc or whatever the environment variable BITDIR points to, and concatenated with possible system wide definition found in /usr/local/lib/bit or HELPDIR with repeats removed.

Filter A filter is defined as an external program that reads at least one of *bit* supported formats from file specified by argv[1] (not *stdin* in order to guarantee *fseek*ability) and outputs the possibly transformed image to *stdout*. *bit* will intercept, load and display the output.

All bindings must start with at (@) character and any line that does not is considered to be comments and is silently ignored. The format for external bindings is as follows:

@label; program ; options; format1; [format2;[format3;] ..]

where *label* is the string that will label the action button; and *program* is the name of the program to run *(popen*(3S) will be called to run it). *options* are command line options to the *program*. If the option field contains a question mark (?) alone, the option will be prompted before the program is actually executed. *Format[0-4]* specifies the formats the program is capable of reading, such as *gif*, *ppm* etc. Note that it is very important to give the formats in decreasing generality, that is, *ppm; pgm; pbm;* **never** *pbm;pgm;ppm.*

For example, the following are some sample (valid) bindings:

Shrink the current image by 20%
@ Scale0.8; pnmscale -xscale 0.8 -yscale 0.8; ppm;pgm;pbm;

Quantize an image
@ Quantize256; ppmquant; -fs 256 ; ppm; pgm;
@ Jquant256; djpeg; -GIF; jpeg
@ Jquant; djpeg -colors ; ?; jpeg

The format name can be gotten by running *bit* with the –f flag. The format specifier is enclosed in parenthesis in the *bit* **–f** output.

Convolve
The convolution matrix must be defined using the following format:

@name; order; n1, n2, n3, ...;

where *name* is the button label and *order* is the rank of the convolution matrix, and *n* is an integer representing the matrix elements in column major, i.e.,*n1* is the (0,0) element, *n2* is the (0,1) element and so on. The matrix need not be normalized.

For example, you can define the following matrix to sharpen an image

@Sharpen; 3; 0,-1, 0, -1,10, -1, 0,-1, 0;

Magnify
Magnify a rectangular region of an image by any amount, limited only by the window size. In this mode, a moving rectangle defines the region of interest. Press *leftmouse* or *middlemouse* to magnify this region to the maximum window size. If left control key is down, *leftmouse* click increases the zoom factor by decreasing the size of the region of interest while *middlemouse* decreases the zoom factor by increasing the size of the region of interest. *menubutton* controls the pop-up menu, which can be used to set some common zoom factors.

The above method changes the magnification factor uniformly in the X- and Y-directions. To adjust separately the magnification factors, use the following keyboard command:

X: Increase x magnification by decreasing region size.
x: Decrease x magnification by increasing region size.
Y: Increase x magnification by decreasing region size.
y: Decrease x magnification by increasing region size.

For global magnification, see Section Configuration and –x|y flag.

Hide Hide the *EditPanel*. It can also be hidden by the *Edit* button on the *FilePanel*.

SysMap Install the default system color map.

Info Hide the *FilePanel*. Useful when doing a very large crop (with AlwaysReadScrn on). In case both the *FilePanel* and *EditPanel* are hidden, press *M* key inside the main window will bring up the *FilePanel*.

Select a color

To select a particular color from a colormap (if one exists) or by choosing RGB separately. Note that this is not a separate command with direct applications, but is indirectly used in many mode, e.g., text, rotate, scale and others.

In RGB mode, the three sliders representing the value of Red, Green and Blue intensities are shown and each slider is individually adjustable. The resulting color is shown on the *OK* button on the right-hand side of the panel. Some predefined named colors, mainly from /usr/lib/X11/rgb.txt, are available by pressing the *PreDefCol* button.

In colormap mode, a panel containing the current colormap with 64 entries per screen is shown and each individual entry is represented by a colored square inside the panel. Two large arrows on the sides of the panel can be used to scroll forward and backward through the color map. A particular color is selected by pressing the mouse on the square having the desired color. Current selection is marked by a cross and is also shown on the bottom-most button cabled *Selected*. The five small boxes on the bottom of the panel show, respectively, the index into the color map, red, blue and green intensities and luminosity.

In both RGB and colormap mode, *leftmouse* and *rightmouse* can be used to select any pixels on screen inside the main window as the current selected color by clicking on the pixel while holding down the CNTRL key. The selected pixels color will be shown in the appropriate boxes and/or buttons. The pixel picking mode is indicated by a cross cursor. This feature can also be used to get the intensity of a particular pixel by turning on the mouse position reporting from the Setup menu.

Crop In crop mode, a copy of the currently displayed image is made and all changes that occur are made to this copy until you either *save* the change or exit from the crop mode. At that point, the copy replaces the original image and all changes become permanent. Cropping a cropped image can be accomplished by using the save feature without exiting crop mode. The actual cropping can be of either internal (memory copy) or external (screen reads) operations. Screen reads are performed if there are text and SGF's on top of the image, otherwise internal cropping is done unless global option *AlwaysReadScrn* is on. In general, screen reads should be avoided on machines with less than 24 bitplanes for normal framebuffer. Note also that a screen read reads everything within the region of interest, as defined by the cropping rectangle, which may be undesirable and in some cases, you might need to hide all the control panels.

Upon entering crop mode, a rectangle with two small crosses on the opposite corners is shown. Clicking and dragging the cross move the corner of the rectangle and hence both the size and location of the crop region can be changed this way. For precise placement, the arrow keys and the keyboard key *hjkl* may be used (as in *VI*(1)). The corner to which the key applies is the one closest to the mouse. This behavior can be overridden by the shiftkey, i.e., if leftshift is down, *hjkl* always applies to the upper-left corner of the rectangle regardless which corner is last touched by the mouse. Similarly rightshift key will force the hjkl key to apply to the upper-right corner. The number of pixels each hjkl moves can be changed by key [1-9]. If *CONTROL* key is down, the aforementioned command will move the rectangle without changing its size. The process can be repeated until a satisfactory crop region is found.

The size of the rubber band can also be symmetrically increased and decreased by the keyboard key xXyX.

The size and placement of the crop region are reported continuously on the *FilePanel*.

Pressing the menubutton is the main window will bring up a pop-up menu having the following entries

DoIt discard the image outside the current rectangle and redraw the screen. If current region is larger than the image on any sides, a fill color will be prompted for (See Select a color). This fill feature can be used in conjunction with *frameN* command to add a border to the image. See also undo and save.

Help Detailed crop command summary, similar to the ones given here, but might contain more up-to-date info.

RBcolor
Cycle through all available colors of the rectangle. Exact number of colors available depends on the hardware.

RBobject
Change the bounding objects. Currently only circles are available, but cropping along the circle is yet to be implemented.

Frame1 Enlarge the rectangle by 1 pixel on all sides. Convenient to add a frame to the current image.

FrameN Enlarge the rectangle by n pixels on all sides. N will be prompted for.

Auto Search the image for solid regions (regions with same color), starting from all sides and the findings are indicated by a properly placed and sized rectangle.

Save Save all the changes made so far and mark the current image as the starting point for future undo.

Undo Undo cropping. Note that undo is of unlimited depth. Executing this command will undo everything since last save.

Done Return control to main.

To enlarge an image without filling the outside region, turn off the *CropFill* flag through configuration file or *Setup* menu. This can be very useful in combination with cut & paste to import cut buffers or to concatenate images.

Rotate There is no copies made in this mode, as soon as an image is rotated, the rotated image replaces the original image as the current image. Thus you can rotate a rotated image.

The rotation options are controlled by the rotation control panel, which consists of a counter to change the rotation angles (between -180 and +180 degrees measured counter-clockwise from the horizontal direction), and other buttons for convenience or other options. To change the rotation angle, simply click the mouse on either the single or double arrows in the counter. Leftarrow decrease the angle and right arrow increase the rotation angle. Rotation of multiple of 90 degrees or flipping about x- or y- axis are provided as shortcut buttons and can be set by clicking the mouse on appropriate buttons. After you have set the angle you can live with, click the mouse on the *OK* button.

For rotations other than multiple of 90 degrees, a fill color can be selected to fill the "rotated out" portions of the bounding rectangular region. To do so, click the mouse on *FillColor* button. In addition, for rotations other than multiple of 90 degrees, you can activate or deactivate subpixel sampling for anti-aliasing. Anti-aliasing always gives better results. The only situation where you do not want subpixel sampling is to keep the exact colors as the unrotated image. To activate or deactivate subpixel sampling, click the mouse on round *subpixel* button. If the subpixel is turned off, the rotated image will have the same type as the unrotated image.

For online help, click the mouse anywhere inside the rotating control panel where there is no other objects.

Cut&Paste

The cut & paste region is defined by a rubber band whose size and location can be modified interactively by the mouse and/or keyboard. Use leftmouse to cut (or C key) and middlemouse (or P key) to paste. Keyboard key uU undoes last pasting.

To change the size of the cut region, keyboard key xXyY can be used to decrease/increase the width and height of the cut region, key [1-9] controls the step size for each key press. The size can also be changed by clicking the menubutton, which will bring up a static rubber band with two crosses on it. You can change the size and location by dragging these two crosses. Keyboard key hjkl and the arrow keys can also be used to manipulate the rubber band. To resume cut & paste, click away from the crosses.

The cut buffer so generated (by leftmouse) may come in one of two ways. It may come as a result of a direct screen read or it may come as an internal memory copy depending on the settings of the ReadScr/Internal button. When ReadScr is active, everything that falls within the cut region will be read into the cut buffer while when Internal is active, the cut buffer always comes from the image in core. Screen reads may be useful in certain situations but in general it should be avoided on machines with less than 24 bitplanes.

A cut buffer may also be obtained by importing an external image in any of *bit* supported formats. To do so, click the *Import* button. The imported image behaves exactly like a normal cut buffer except that it can not be named, thus a new cut or new import overwrites the current imported image.

To save a cut buffer for later pasting, click *Save* button. The saved buffer will be available during the entire session of *bit.*

There are two different kinds of pasting, i.e., Overwrite and mask. In Overwrite mode, the cut buffer replaces completely the targeted image area, both on screen and in core while in mask mode, a mask operation is performed before replacing the pixels in the region. The mask is defined as NewPix = (PixInCutBuffer != MagicPix) ? PixInCutBuffer: OriginalPix for every pixel in the region (current MagicPix is 0). An obvious application of this feature is to import external images as logos or signatures and paste it into the current image.

Bit further has a Preview mode in Cut&Paste where all pasted buffers only alter the pixels in the framebuffer and leave alone the image in core. To alter in the framebuffer as well as altering the image in core, toggle on the Commit flag.

Note that in Commit mode or Preview with Masking turned on, both the cut and paste regions are not allowed outside the image area, thus if an image larger than the current image is imported in Commit or Preview with masking turned on, a lock up of the rubber band might occur, therefore, it is generally not a good idea to import an image larger than the current image, do the reverse instead.

For a simple cut&paste, see Paint function.

Scale Scale is controlled by the scaling control panel. By default, the aspect ratio is kept for scaling, thus changing one scale factor will change the other. If the purpose of scaling is to correct aspect ratio, click mouse on the *KeepASP* button to disable it, and x and y scaling factors will become independent and can be adjusted separately. The scales can be shown either as a ratio (current size/original size) or as the resulting size in pixels by toggling the *ShowSize* or *ShowScale* button. Subpixel sampling can be turned on or off by clicking the mouse on *Subpix* button.

In certain situations, it might be desirable to make the image having a particular size without changing the aspect ratio. To achieve this, toggle on the *Exact* button, and the scale factors become independent, adjust each until you've got the desired size, *bit* will scale the image to the requested size, using the smaller of the two factors, and filling the empty region with a fill color, which can be set by clicking on the *FillColor* button.

To start scaling, click the mouse on *OK* button. To exit scale mode, click the mouse on *Done* button.

Merge Load an image and concatenate with the currently displayed image. If the separation between the two images are negative one image is merged into the other.

To load the image to be merged, click on the Image2M button.

There are total of 12 alignments between two images (3 on each side, e.g., if we want to place the new new image on the left of the current image, within this constraint, there are center, top and bottom. If the new image has the same size as the current image, the three placements degenerate into a single location) and each is shown as a small arrow. The meaning of each arrow becomes clear if you consider the manual button as the current image, and all other buttons are the possible new image locations. The counter below the alignment button controls the separation between the two images, thus a negative separation merges two images. In performing merging, there are two ways to handle the overlapping part, the default is for the new image to overwrite the old one, i.e., new image on "top". To change the default, click on the Top/Bottom button.

In addition to overwrite, other operations are possible. Current implemented operations are defined as follows (assuming A is a pixel in the current displayed image and B is the corresponding pixel in the image to be merged):

Name	Definition
OverWrite	B
Mask	B ? B:A
Add	Min(B+A, 255)
Sub	Max(B-A, 0)
Diff	Abs(B-A)
XOR	BˆA

Switch B&A if the top/bottom setting is bottom.

The placement buttons and the separation counter provide adequate alignment between two images in most cases, nonetheless, a manual button is provided in case you want to control precisely where the image should be placed. To activate manual placement, click on the manual button. Then you can use the arrow keys, hjkl[1-9] to move the the rubber band, or you can use the mouse to move the rubber band.

Upon a satisfactory location is found, click on the Merge button to initiating merging/concatenation, and a new image is formed with all empty regions filled with the fill color, changeable by clicking on the FillCol button.

After merging/concatenating, although the image displayed and in core have the merged size, further alignment requests are still against the pre-merged image, not the merged one. To change this, click on the Update button and subsequent alignment requests will be against the current image. Depending on the situation and relative image sizes involved, Update button may prove to be very useful, e.g., to concatenate four identically sized images together, the only way to do it without manual placement is to concatenate the first two, and update the alignment reference size, and concatenate the rest without update.

To exit merging mode, click on the Done button.

PixTran Arbitrary 1-to-1 pixel (color) transformation, i.e., all pixels having value I will be replaced by F(I) in output pixel, where F(I) is the transformation functions. The transformation can be applied to the entire image or a rectangular region of it on RGB separately or simultaneously.

Upon entering PixTran mode, shown is a control panel with three XY graphs representing the RGB transformation function with the horizontal axis being the input pixel value and the vertical axis being the output pixel value. The initial transformation is an identity transformation, i.e., F(I) = I. To change the transformation function, click the mouse on the control points (shown as little squares) on the graphs and the cursor will change to a small cross. Keep mouse pressed and drag the control points to change the transformation function. The number of control points can be changed by the counter labeled *ControlPoints*.

Initial size is the entire image, as indicated by rubber band. To adjust the editing size, drag the small crosses on the rubber band or use keyboard key hjkl (or arrow keys) and xXyY.

On top-right of the control panel, three rounded buttons labeled *Red*, *Green*, and *Blue* are shown. Immediately below, there is a large button labeled *RGB*. These four buttons control to which component, or simultaneously all three components, the current transformation applies to. These four settings are of course mutually exclusive and setting one will unset the other three.

In principle, the interaction mentioned above with the transformation function is capable of any transformations imaginable, but in practice, it is not always easy to set the transformation to some known functions, nor to change all the control points by some given amount, such as gamma corrections and brightness adjustment. To solve these problems, a math expression parser and several common functions are provided as shortcuts. The parser is rather lenient in the syntax and error, but demands you name the variable x. Photo inversion, step functions, gamma correction are among the shortcuts. In addition, the intensity can be shifted either downward or upward. For large number of control points with upward shifts, the final transformation can be very interesting because of ignored overflows. The overall brightness of the image can be adjusted in two ways: the brightness can be increased by an absolute amount ($F(I) = I + \delta$ or by a percentage ($F(I) = (1 + \delta * I$. The selection can be changed by clicking the mouse on text *Brightness*.

Once a satisfactory transformation is found, click the mouse on the button labeled *Apply*. The current image will be transformed according to the transformation and the settings of which component or components the transformation applies to. The transformed image will then replace the original image as the current image. To undo the transformation, press the mouse on the button labeled *Undo*. Note, however, unlike in crop, only one undo is supported, that is, if you did two transformations in succession, only the last transformation can be undone.

To exit *PixTran* mode, press mouse on the *Done* button.

By using combination of the 1-to-1 transformations, some very pleasing effects can be achieved. For example, to simulate the effect of looking the image through a bluish glass, shift red and green intensity down by two bits, and shift the blue intensity down by one or zero bit, do the transformation. The result is very nice. Similar tricks can be applied for reddish glass or greenish glass. Be creative, when you find something interesting, please let me know. I might add a shortcut button for it.

Note when I said the interaction mentioned above is capable of any transformation imaginable, I lied. It can't simply because the transformation is three one-dimensional graphs instead of the correct one three-dimensional one. For this reason, two additional functions are supplied to swap two arbitrary pixels values (e.g., change black to white and white to black) or simply replace all pixels with a particular color with another color (e.g., change all dark gray pixels to bright red).

Text A panel with many objects is shown in text mode. Text color, size, and orientation can be set by clicking the mouse on appropriate objects inside the text control panel, and any changes in attribute take effect immediately.

Initial text is the current filename and can be changed by typing into the control panel field labeled *Text*. The new text is shown as soon as you type, both inside the field and in the main window. To move the location of the text string rendered, click the mouse inside the main window, the string will move to the location of the mouse. You can also drag the text string by keeping the mouse button down. For precise placement, use the arrow keys or hjkl.

Sometimes, it is desirable to render the text onto a fixed (uniform) background. To this end, turn on the *Bkgrnd* button, and a uniform background will appear beneath the text. The background color can be changed by clicking the mouse on the *BkColor* button. Note that if the background is not enabled, neither is the background color. Similarly, if you like to have a bounding frame around the text, turn on the frame feature by clicking the mouse the *Frame* button. In conjunction with *PixTran's* region feature, a special background can be created that gives the image as appears through semi-transparent glass.

Some special effects are available, currently, *Shadow, engraved* and *emboss* are implemented. For any of these to look nice, you need to play with the text and background color a bit. In general, for *shadow* to look nice, a strong text and background are needed and for *engraved* to look nice, a mild text and background should be used and text should be stronger than the background, and further, the background and text should be close. The same is true for *emboss* effects.

Note there are two modes of operation, one is the *immediate* and the other is the *deferred* method. In *immediate* mode, text are rendered into the raster while in *deferred* mode, text is saved separately from raster and may be output separately to get better looking hardcopies. The mode of operation can be toggled by pressing on the appropriate buttons. If text are saved in deferred mode and later on, you decide to render the text into the raster, you can do a crop.

If the purpose of adding text is to get an annotated hardcopy, justification is extremely important. Keep in mind that screen has a much lower resolution than a printer. The same string displayed left justified and center justified (with right shift of half of the string length) might look the same on the screen, but could be very different on hardcopy. Therefore, always use proper justification if hardcopy is desired. To preview what the test might look like on hardcopies, choose the PrinterFont by toggling the ScreenFont. This will switch the fonts used to printer matched ones which give better indication how long the text would be or how different lines of text would align on paper.

If special symbols are desired, you can input them just like in *TeX*(1), where special symbols are surrounded by dollar signs ($) to indicate font switching. Unlike in TeX, superscript and subscript do not work. Minor size changes on the same line can be achieved by using command \tiny \small, \normal, \large and \huge. For symbols not defined by *bit*, you can still get them by inputting their character code directly in octal preceded by backslash (\), provided you know the encoding scheme of font you are using. For text via PostScript, *ISOLatin1Encoding* is the default (unless printer is Level 1 only and in that case standard PS encoding is used).

Once everything, including the location, is satisfactory click the mouse on the *OK* button to place the the current text into the image (not necessarily into the raster, depending on immediate/deferred settings). To exit text mode, click the mouse on the *Done* button.

Note: in immediate mode, strings that are outside the image can not be rendered. To render a string outside the image, use deferred mode and do a crop following the text mode and fill the empty region with whatever color you might choose.

Marking

Mark locations on image with simple geometric figures, (arrows, circles, rectangles etc). This function is controlled by a control panel showing all available basic SGFs. Location, size, color, orientation and line thickness can be individually adjusted by clicking on appropriate buttons. By default, the origin of the coordinate system is at the lower left corner of the image and all distances are measured in pixels from origin.

Sometimes, the underline image may have intrinsic units associated with it, and working in that units might be easier, such as a map with one pixel-pixel distance equal to 0.83 km (kilometers). To switch to this unit, toggle on the *User* coordinate button and enter the conversion between pixels and this unit, defined as the ratio between 1 unit in the pixel coordinate system and the value it represents in the new unit system, and in this example, you would enter

1.20482 (which is 1/0.83), meaning 1.20482 km is represented by 1 pixel-pixel distance. Once in user coordinates, all distances and sizes reported and requested should be in the new units, for example, you can request to draw a circle of radius 120 km etc. The origin of the coordinate can be changed by pressing the mouse on the *SetOrigin* button, and the current location of the mark will be the new zero of the coordinate system.

To move the location of the sgfs, you can either click inside the main window where you want the sgf to be, or you can input the coordinates of the locations you desire, and the SGF will move to the new location. The input feature is extremely useful if user coordinate system is in force.

For speed reasons, only the x- and y- components of the current location is reported in the control panel by default. To get the distance from the origin, turn on the *Dreport* option.

To place a mark, press the mouse on the *OK* button, current object together with all of its attributes, such as color, location, etc, are saved. To exit mark mode, press *done* button.

The marks so "rendered" may or may not be rendered into the image depending on the settings of the *Immediate*/*Deferred* mode. In *Immediate* mode, sgf is rendered into the raster and *bit* has no record of it. In *deferred* mode, full vector information of the sgf is retained and the sgf is drawn on top of the image, but *not* rendered into the raster. The *deferred* mode is useful for generating hardcopies via PostScript where the vector info is used to scan convert the sgf on a printer which typically has a better resolution (>300dpi) than the computer screen (<100dpi). If sgfs are rendered in *deferred* mode, and later on there is a need to render them into the raster, do a crop.

Note in *immediate* mode, all sgfs that are outside the image are clipped. To render a mark outside the image, enlarge the image first via crop.

For convenience, the following keyboard keys can be used: xXyY have the usual meaning, that is xy decreases the width and height respectively and XY increases the the width and height. Arrow keys (or hjkl) can be used for precise placement of the sgfs. rR rotates the sgf in discrete steps which can be set by [1-9].

If the hardware is capable of drawing anti-aliased lines, *bit* will try to draw all lines anti-aliased. However, for colormap images or lines with width that is not 1, anti-aliasing will not be performed.

EditMap

Only meaningful if the current image is in colormap. Upon entering this mode, current colormap, 64 entries per page, and a color mixer are shown. To select a specific entries in the colormap, click on the little square inside the colormap. The current selection will be marked by a small cross and current selection's RGB intensity will be shown in the color mixer. You can change the color of current entry by dragging the specific color channels in the mixer and corresponding color in the map as well as all pixels in the image that match the current entry will reflect current settings. On the bottom of the colormap, there are five small boxes, showing respectively, the index into the map, the red, green, blue and luminosity of the entry. Entries in the map can be scrolled by the large arrows on the sides of the colormap.

To exit colormap editing, click the mouse on the *Done* button on the colormap panel.

Enhance
This is really histogram equalization generalized to color images. Instead of equalizing the RGB histograms separately with their own transformation function, a function based on the luminosity will be applied to the RGB components. There is no interaction with this function. A transformed image will be displayed and the original image will be replaced by the transformed one. In cases where the image of interest is superimposed on a strong (either black or white) background, histogram equalization gives poor results in the background region, maybe a thresholding before equalization is called for.

Smooth Apply a special convolution where each pixel is replaced by a weighted sum of itself and its nearest (3x3) and next nearest (5x5) neighbors.

Upon entering this mode, a copy of the current image is made and all changes that occur are made to this copy until return and at that point, the changes becomes permanent. To force the changes to be permanent before return, click on the Save button. Click on Undo to undo all changes since last Save.

By default, the smooth is performed on the entire image. To smooth a particular region of the image, drag the small crosses on the rubber band. See Crop for more details on the interactive sizing of the rubber band.

To preserve edges, a threshold may be used according to the following definition:

```
If(Abs(SmoothedPixel- OldPixel) >= threshold)
  NewPixel = SmoothedPixel
else
  NewPixel = OldPixel
```

Different thresholds for R,G,B can be used and are set by pressing on the counters.

To start smooth, click on the Smooth button. To finish up, click on the Done button.

Sharpen Apply a special convolution where each pixel is replaced by a weight sum of itself and its nearest neighbor. A rubber band defines the region to which the sharpen operation applies.

MedianFilter
Perform a 3x3 median filtering. Again, A rubber band defines the region of interests.

Histogram
Clicking on this button will bring up a window with four panels, showing red, green, blue and white graphs. These graphs are the separated histograms of the RGB components of the current image. The horizontal axis is the intensity and vertical axis is the population count. Note that this is not the true histogram because correlation is ignored, nevertheless, some insight can be gained by examining these graphs.

The graphs are static for efficiency reasons and therefore, even when image has been changed, the histogram is not updated automatically. To show the histogram of current (changed) image, clicking the mouse on the *Histogram* button again.

Click on the title bar brings up the pop-up menu, whose entries include re-scaling, change of plotting styles etc. Select Done to finish.

Edge
Uses a variant of orthogonal Sobel gradient operators. If the input image is in color, the output edge map will also be in color. The colormap used depends somewhat on the image itself. By looking at the histogram of the edge map, a good threshold value can be chosen and then a two-level step transformation will produce a bi-level image. No interaction.

Image Difference
Computes the differences between two identically sized images pixel by pixel and displays the results as an image. More information can be gained by showing the histogram of the difference image.

Image Panning
Image panning makes viewing images larger than the screen possible. The panning control panel consists of nine buttons labeled by the directions of the movement. The center dots will center the image both vertically and horizontally. The slider immediate below controls the pan step, in units of pixels. In SlideShow mode, image panning is automatically performed in both directions unless disabled by the initialization file.

Measuring Pixels Intensities Along a Line
The length and angle of the defining line can be changed by dragging the crosses at the two ends of the line. The origin of the line is marked by a circle around the cross. The intensity is presented as four graphs with horizontal axis being the distance, in pixels, along the line from the origin. The vertical axes are the Red, Blue, Green intensities and luminosity (or color index depending on the image type). To activate or update the reporting, click mouse inside the main window. *Menubutton* again controls the pop-up menu through which the line color can be changed or online help can be obtained.

Edit Individual Pixels
Upon initiating, a rectangle attached to cursor is shown and the region within the rectangle is magnified and shown in the *PixEdit* window. Once you have found the region you intend to edit, click *leftmouse* or *rightmouse* within the main window, the rectangle will cease to move with the cursor and the area within the rectangle will be the region you can edit.

To select the pixel to edit, click leftmouse on the pixel in the *PixEdit* window. Its value can be changed in the same way as in *Select a color*, i.e., by dragging the slider and/or changing the colormap entry.

To set a pixel to the current color (as shown in the mixer/colormap), click leftmouse on the pixel while holding down the ALT key or. by clicking middlemouse. Dragging will set all pixels along the path.

To change all pixels that have a particular value to a different one, edit one first, then use the pop-up entry Repeat. For example, if one wants to change all bluish pixels to red, select one bluish pixel and change it to red, then do a Repeat thru popup (To changing/swap pixels for the entire image or a large region of it, See PixTran function).

The pop-up menu has the following entries *Help, MarkCol, GridCol, Repeat, Cancel* and *Done. MarkCol* and *GridCol* cycles the the colors of the pixel mark and the grid; *Repeat* Do the last editing on *all* pixels in the PixEdit window. *Cancel* discard all changes and return. and *Done* saves all changes and return.

The following is not editing functions per se, but its interaction is complex enough to warrant an explanation:

Image Browser

Click on the *IBrowse* button to activate the image browser. If the thumbnail images exist (created by an earlier session of **bit**), they will be loaded and shown in the browser. If thumbnail images do not exist, *bit* will only show generic icons, i.e., the program will not generate thumbnail images upon startup.

To view an image, double click on the filename icon. If the image is successfully loaded, its thumbnail image will be made and used to replace the original generic icon. Double click on a directory will spawn another image browser. Total of 7 directories can be simultaneously displayed.

To generate thumbnail images for all visible filenames in the browser, click on the *Update* button. If a thumbnail has a date that is earlier than that of the image, the thumbnail is regenerated.

Directory entries are cached and to force a re-read after addition or deletion, click on the ReScan button. Note however, ReScan will also do an update. In addition, thumbnails whose original images no long exist will be deleted.

To move forward and backward within the browser, the arrow buttons and the slider can be used. For efficiency reasons, slider change is acted upon only on its release, that is, program does nothing while you are changing the slider. Arrows move one-screenful of files at a time, and thumbnails, if they exist, are loaded as you go. This could be slow if many thumbnails exist, to move back and forth quickly without loading any thumbnail images, keep CNTL key down while pressing on the arrow buttons.

To cleanup all thumbnail images, click on DelAll button. To cleanup manually, remove ./.btn directory and all entries in it.

There is no built-in buttons to manipulate (for example, deleting the original image) the original images. A mechanism via run time definition is provided to achieve this operation. Once a definition is registered, it operates on all marked files. To define an external binding, click *menubutton* on the button next to the Close button; to execute the binding, click *leftmouse* on the button.

The definition is consisted of two strings, and two booleans. The two strings are the button label and a shell command respectively and two booleans are Confirmation and MultiArgs. MultiArgs indicates if the shell command accepts multiple arguments and if Confirmation is set, **bit** will pause for a mouse click before the actual execution of the binding. The shell command takes the form

```
cmd [options] [?] [options]
```

where ? will be replaced by the filenames marked. For example,
mv ? /usr/people/good_dir

Paint

This function offers standard paint functions, namely, pencil, spray, brush and some other common operations.

Upon entering paint function, a panel containing all paint function icons are shown. Click on the icon to activate a specific function.

On the lower-right corner of the panel, current line width and fill pattern are shown. Two special entries are available to facilitate the uniform operation of outlining and fill. By default, outlines are drawn, possibly with a different color, around the filled area. To turn outlining off, select None (or zero) for line width. Similarly, all closed area are filled unless the fill pattern is None.

MPEG movies

This is one of the file formats that can hold more than one images in it (PostScript is another). Upon loading and displaying the first frame, a control panel is shown. Click on the Play button to show the movie non-stop, interruptible by the Stop or Done button. To play and rewind/loop, click on the Play&Loop button, again, interruptible by Stop/Done button. Number of frames per second is reported at the end of a sequence.

To display frame by frame (stepping), click on the NextFrame button.

The default magnification factor for MPEG movie is 2x2. To change it, click on the left-most box above the FrameControl box. Use menubutton for random selection.

To introduce a delay between two frames, click on the box next to the magnification box, click leftmouse for next entry, middle mouse for previous entry and rightmouse for random access.

To convert the currently displayed frame to other format, click on the Convert button. The destination format is selected by clicking the box on top of the Convert button. To convert all frames, starting from current frame, click on ConvertAll button.

All other buttons have the obvious meanings.

Also see sgimpeg(1L) for a simpler, stand-alone MPEG player.

Note: seeking of any kind does not work. Clicking on PrevFrame is the same as NextFrame.

PostScript Files

PostScript is supported via GhostScript, see *gs(1).* Again, upon converting and displaying first page, a control panel is shown if the document is multi-paged.

The following functions are not directly supported by *bit* program but can be done indirectly:

Measuring Pixel Intensities

In any mode where a color is prompted for, e.g., rotate, scale crop, text etc, click inside the main window while holding down CNTRL key will read the intensities of the pixel under the mouse. Again, the RGB, and luminosity (or CI) will be reported. The mouse position can be gotten by turning on the mouse reporting option through Setup menu. Also see Select a Color.

ACKNOWLEDGMENT

JPEG support is based on the Independent JPEG Group's library Version 4.

The FORMS library is developed by Mark Overmars.

MPEG support is derived from the Berkeley source.

FILES

/usr/local/lib/bit/∗.hlp
: All online helps. In case environment variable BITHELP exists, BITHELP will be used instead of the default.

$BITHELP/∗.dat
: Some control files

$BITHELP/Setup.opt
: Basic configuration. The first file *bit* reads unless -n is present.

˜/.bitrc/Setup.opt
: User configuration file which is read after reading the system default.

˜/.bitrc/BIT_filters
: The definition file for external filters. It will be concatenated with definitions read from $BITHELP/BIT_filter.

˜/.bitrc/BIT_convolve
: The external convolution matrices.

SEE ALSO

djpeg(1), cjpeg(1), ppm(5), pnm(5)

BUGS

Command line options are single letters.

Unable to load two images simultaneously due to loading module's re-entrant problems.

Can't run bit remotely, at least not reliably.

AUTHOR