Envision Toolbox Manual

Chapter 7

Getimage

Image Capture

Chapter 7 - Getimage

A. Introduction

Getimage allows you to capture an area of a workstation screen (running X Windows, version 11, release 4, 5, or 6), and save it as an image. A variety of output file formats are supported; the default is the Khoros 2.0 VIFF file format.

Getimage provides a convenient way in which to create a Khoros 2.0 VIFF file from images that may be displayed using non-Khoros methods. Images captured using getimage may be displayed with putimage or manipulated using editimage.

The *source* of image capture with getimage may be a specific window, or the screen as a whole. You may capture the entire source, or you may interactively specify a portion of the source to capture using a rubberbanding method.

Getimage supports capture of both 24-bit and 8-bit images. The number planes captured depends on the display from which the image is captured. For example, on a 24-bit display, capture of both 8-bit and 24-bit images is possible; the captured image will be 8-bit if the source of the captured image was 8-bit, or 24-bit if the source of the captured image was 24-bit. On an 8-bit display, an 8-bit image will be captured.

Getimage *cannot* handle capture of areas containing multiple or split visual displays, or areas containing multiple colormaps.

The following is a summary of the different ways in which getimage can be used.

1) Entire Screen
You may capture the screen in its entirety as an image.

2) Screen Area
You may capture a portion of the screen, regardless of the number of windows which it may include, interactively specifying the region to be captured by rubberbanding the area of interest.

3) Entire Window, Interactive Specification
You may capture a single window in its entirety from the screen, interactively specifying the window to be captured by clicking on it.

4) Entire Window, Specification by Name/ID
You may capture a single window non-interactively by specifying the window name or window ID.

5) Window Area, Interactive Specification
You may capture an area of interest from a particular window, where the window is specified by clicking on it and the area of interest is specified by rubber-banding.

6) Window Area, Specification by Name/ID

You may capture an area of interest from a particular window, where the window is specified by name or ID and the area of interest is specified by rubber-banding.

IMPORTANT NOTE: when a window is used as the image source, if the window indicated for image capture is obscured by any other windows, the image captured will also include the obscuring window(s).

B. Command Line Options

The syntax for **getimage** is as follows:

```
% getimage
  -o         (outfile) Output filename

  Required Mutually Exclusive Group - Specify exactly one of the 3 options:
    [-screen] (flag)   Use screen as source
    [-win]    (flag)   Use (interactively indicated) window as source
    [-wname]  (string) Name/ID of source window

  [-all]     (flag)    Save ALL of window or screen?
  [-decor]   (flag)    Grab the window manager decoration?
  [-b]       (flag)    Output image in bitmap format?

  [Optional Mutually Exclusive Group - Specify one or none of the 11 options:]
    [-ascii]   (flag) Output in ASCII format
    [-viff]    (flag) Output as VIFF (Khoros 2.0 Format)
    [-avs]     (flag) Output in Application Visualization System (AVS) format
    [-xbm]     (flag) Output in X11 Bitmap (XBM) format
    [-pnm]     (flag) Output in Portable Any Map (PNM) format
    [-xvimage] (flag) Output as Xvimage (Khoros 1.0 Format)
    [-eps]     (flag) Output in Encapsulated PostScript Format
    [-xwd]     (flag) Output in X11 Window Dump (XWD) format
    [-raw]     (flag) Output in Raw (headerless) format
    [-xpm]     (flag) Output in X Windows Pixmap (XPM) format
    [-rast]    (flag) Output in Sun Raster format
    [-gif]     (flag) Output in GIF format
```

Note that the standardized Khoros arguments [-V], [-U], [-usage], [-P], [-gui], [-form], [-x], [-y], [-A], [-a], and [-ap] are also supported by getimage.

-o {output file}

This is the name of the output file in which the captured image is to be saved. It is a required argument.

[-screen]

Provide the [-screen] argument if the source of the captured image is to be part or all of the entire screen, as opposed to a specific window. It is mutually exclusive with [-win] and [-wname].

[-win]

Provide the [-win] argument if the source of the captured image is to be part or all of a particular window, where you will be interactively specifying the window by clicking on it in response to the prompt. It is mutually exclusive with [-screen] and [-wname].

[-wname {window name/ID}]
Provide the [-wname] argument if the source of the captured image is to be part or all of a particular window, where the window will be specified using providing the name of the window or the window ID as the value of the [-wname] argument. It is mutually exclusive with [-screen] and [-win].

[-all]
Specify the [-all] flag if you want to capture ALL of the source as the output image. If the [-screen] flag is specified, this implies that the whole screen will be captured; if the [-win] or [-wname] arguments are specified, this implies that the whole window will be captured.

[-b]
Provide this flag if you want the captured image to be in bitmap format.

[-decor]
Provide this flag when capturing an entire window if you would like the window manager dressing to be included in the captured image.

Output File Format
The remaining flag arguments of getimage are for specifying the output file type. You may specify any file format that is supported by Khoros 2, including Ascii, Khoros 2.0 VIFF, AVS, X11 Bitmap, Portable Any Map, Khoros 1.0 VIFF, Encapsulated PostScript, X11 Window Dump, Raw, X Windows Pixmap, or Sun Raster format. The default format is Khoros 2.0 VIFF format.

C. Methods for Image Capture

The following sections detail the different methods for image capture that are supported by getimage, and how these different methods may be used by providing different combinations of arguments to getimage.

C.1. Entire Screen

This method provides a convenient way to capture the entire screen. Execute:

```
% getimage -o {filename} -screen -all
```

This will cause getimage to capture the entire screen automatically. Note that the entire screen can also be captured interactively using the "Area of Interest on Screen" method.

C.2. Screen Area

This method allows you to capture a specified portion of the screen, regardless of the number of windows which it may include.

```
% getimage -o {filename} -screen
```

You will be prompted to rubber-band around the area of the screen to be captured. To rubberband the area of interest, first click on the upper left hand corner of the desired portion of the screen; the cursor will change to look like the upper left hand corner of a box. *Keeping the mouse button down*, move the mouse to the lower right hand corner of the desired portion of the screen; the active portion will be indicated by rubber banding. The rubber-banded portion of the screen will be extracted into *filename* as soon as you release the mouse button.

C.3. Entire Window, Interactive Specification

This method lets you capture a single window in its entirety from the screen, interactively specifying the window that you want.

```
% getimage -o {filename} -win -all
```

You will be prompted to click on the window which is to be captured. After you click on the desired window, the window will be captured as *filename*, a Khoros 2.0 VIFF file.

C.4. Entire Window, Specification by Name/ID

This method lets you capture a single window in its entirety from the screen, specifying the window that you want by its name or ID. Being non-interactive, this method is better suited for automated image capture; it would be appropriate for execution from a script. Execute

```
% getimage -o {filename} -wname {window_name} -all
```

or

```
% getimage -o {filename} -wname {window_id} -all
```

If the name or ID of a window is unknown, you may use the X11 utility program "xwininfo" to obtain it. If a window with the specified name or ID can't be found, getimage will exit with an error message.

C.5. Window Area, Interactive Specification

This method lets you capture part of a single window, interactively specifying the window that you want, and then interactively specifying the area of interest within that window.

```
% getimage -o {filename} -win
```

First, you will be prompted to click on the window which is to be captured. After you click on the desired window, you will then be prompted to rubber-band the area of interest within that window. When rubberbanding, the cursor will be constrained to locations within the specified window, making it impossible to capture any area outside the window.

C.6. Window Area, Specification by Name/ID

This method lets you capture part of a single window, specifying the the window that you want by its name or window ID, and then interactively specifying the area of interest within that window. Execute:

```
% getimage -o {filename} -wname {window_name}
```

or

```
% getimage -o {filename} -wname {window_id}
```

As mentioned previously, if the name or ID of a window is unknown, you may use the X11 utility program "xwininfo" to obtain it.

You will be prompted to rubber-band the area of interest within the named window. When rubber-banding, the cursor will be constrained to locations within the specified window, making it impossible to capture any area outside the window.

D. Other Features

Bitmap Format
Getimage supports output of images in bitmap format. When the [-b] argument is used, getimage simply takes a pixmap and dithers it down to a bit (binary) image before outputting the image to the filename specified.

Inclusion of Window Manager Decoration
By default, when a window is captured, the window manager decoration is not captured with the image. The [-decor] argument specifies that the window manager decoration is to be included as part of the output image. Thus, if the command "% getimage -o {filename} -decor" is executed, the window title bar, borders, etc, will be included along with the contents of the window.

Output File Format
Using getimage, you may output the captured image in any file format supported by Khoros 2. You may specify any of:
[-ascii] ASCII format
[-viff] VIFF (Khoros 2.0 Format)
[-avs] Application Visualization System (AVS) format
[-xbm] X11 Bitmap (XBM) format
[-pnm] Portable Any Map (PNM) format
[-xvimage] Xvimage (Khoros 1.0 Format)
[-eps] Encapsulated PostScript Format
[-xwd] X11 Window Dump (XWD) format
[-raw] Raw (headerless) format
[-xpm] X Windows Pixmap (XPM) format
[-rast] Output in Sun Raster format
By default, the output file format is Khoros 2.0 VIFF.

E. Limitations

Getimage *cannot* handle capture of areas containing multiple visuals, or areas containing multiple colormaps. The following examples give scenarios where these limitations may affect the performance of getimage.

Example 1

On a display that supports multiple visuals within the same application (eg, 24-bit SGI Irix 5.2), suppose you have displayed a 24-bit image using editimage. You then attempt to capture the entire editimage application using "getimage -o outfile". The captured image not "look right", since the graphical user interface of editimage will be using 8 bits, while the image itself will be using 24 bits, and getimage cannot support capture of multiple visuals. In the captured image, the graphical user interface of editimage will look correct, while the image that was being displayed by editimage will have a corrupted colormap.

Example 2

On a 24-bit display (eg, a SparcStation 20 SX), suppose you have displayed an image. Using "getimage -o outfile -e", you attempt to capture a portion of the screen containing both the displayed image and the text from an xterm that is partially obscured by the image. The displayed image uses one colormap, while the xterm in the background uses another colormap. In a similar effect as that described in example 1, the portion of the captured image that contains the displayed image will have a corrupted colormap, while the portion of the captured image containing the text of the xterm will appear correctly.

Example 3

On an 8-bit display (eg, a DecStation 5000 with an HX display), suppose you have displayed two images side by side using two separate executions of putimage. You then use "getimage -o outfile -e", and rubberband an area that includes the left half of one image and the right half of the other image. In the captured image, neither of the images captured will have the correct colormap (the images will look black or "scrambled"), since getimage cannot handle areas containing multiple colormaps.

Table of Contents

This page left intentionally blank