NAME

imkey - keys a foreground image over a background image

SYNOPSIS

imkey [*options*] *foreground-filename background-filename outfilename*

DESCRIPTION

imkey imkey places a foreground image over a background image, letting the background show through wherever the foreground's pixels match a key color specifified on the command-line. The resulting image is stored to an output file. Input and output files may have different image file formats.

OPTIONS

imkey has a variety of options in the following categories:

File Selection	What input and output files to use
Format Selection	What image file format to use
Format Control	What variant of a file format to generate
Standard	Standard generic options on all SDSC tools
Field(s)	Key color (or other field) selection
Positioning	Position foreground onto background
Subregion	Select a subregion of the foreground

File Selection, *Format Selection*, *Format Control*, and *Standard* options are common to all SDSC image tools and are discussed in depth in the man page for imconv(1IM).

All options can be abbreviated to the first few unique characters.

Keying

Keying is a technique that combines pixels of two input images together to create a third *composite* image. *Keying* is commonly used to take two separately computed images and combine them, with portions of the foreground made transparent so that the background can show through. In such cases the first input image would act as a *foreground* image, and the second as a *background* image.

Foreground and background images can be composited together to create a variety of special effects, or as an aid to speed up the rendering of complex scenery. For instance, a complex computer-generated robot must walk across the screen in front of a computer-generated library filled with books. The library could be modeled, book by book, and the robot modeled, joint for joint. An image showing the robot in the library could be rendered by combining both models and sending the entire mass of geometry off to the renderer all at once. A single image would result.

To render an animation of the robot moving in front of the library one could send both masses of geometry to the renderer for each frame. However, the library is static in this scene and the interaction between the robot and its books nil. The time spent rendering, and rerendering, each library book, frame after frame is a waste and dramatically increases the rendering time of the animation.

A more efficient approach would be to render the static library just once, without the robot. The robot would then be rendered for each frame, but without the library behind it (just a solid black background). Once rendering is complete, the separate foreground (robot) and background (library) images can be keyed together to create a series of frames showing the robot moving in front of the library. The total rendering time for such an approach is considerably less.

Such compositing tricks are regularly used in Hollywood in order to place actors in front of locations they aren't in, or to make spaceships fly all about in front of swirling planets or sparkley star fields. In video this same technique is used to place the weather man in front of a computer-generated weather map.

In each of these cases, the keying operation must know what portions of the foreground image to paste over the background image.

The net effect is that the foreground image is pasted over the background image letting the background show through where the foreground's pixels are satisfied by the keying information.

Field(s)

Users may choose key off of specific foreground fields using one of the following options.

Option	Key foreground off of...
-mono	the monochrome field
-index	the color index field
-red	the red field
-green	the green field
-blue	the blue field
-hue	on the virtual hue field
-saturation	on the virtual saturation field
-intensity	on the virtual intensity field

In general, a key field named by any of the above options must exist in both input images. For instance, keying using -index for color index fields doesn't make sense on RGB images. A certain amount of automatic image type conversion is performed by imkey in order to make things a little bit more flexible:

-mono Monochrome keying requires that both input images have monochrome image fields.

-index Color index keying requires that both input images have color index image fields.

-red, -green, -blue

Keying on red, green, or blue fields works for any image, whether monochrome, grayscale, color indexed, or RGB. Such images are promoted internally to RGB images in order to accomplish the keying.

-hue, -saturation, -intensity

Keying on the virtual image fields of hue, saturation, or intensity works for any image type. Input image color field values are converted internally to the HSI color space in order to accomplish the compositing.

Positioning

There also exist provisions for positioning and sizing the foreground image. This is useful since sometimes only a part of the foreground is desired in the composite. The positioning of the foreground in respect to the background is useful if the foreground needs to be somewhere other than the upper left hand corner of the image, which is the default.

-xposition x Specify left edge of foreground into background
-yposition y Specify top edge of foreground into background

-xposition -yposition

Allows you to position the foreground in the background. The foreground can be positioned at a new X and Y position other than the default, which is 0 and 0. Negative values for for X and Y are allowed. This will simply move the foreground off the left edge and/or the top edge of the background. Those areas of the foreground that are wide enough and/or long enough will extend back into the final image. Quantities are specified in pixels.

Subregion

A subregion of the foreground image may be selected using -xstart, -ystart, -xsize, and -ysize. Start values are in pixels measured from (0,0) in the upper left-hand corner of the foreground image. Size values are in pixels. Start values must be positive and size values must result in a subregion wholely contained within the foreground image.

-xstart x	Specify left edge in foreground data
-ystart y	Specify top edge in foreground data
-xsize width	Specify width of foreground data
-ysize height	Specify height of foreground data

-xstart -ystart

-xstart and -ystart each default to (0,0) in the foreground image. -xsize and -ysize each default to the full image width and height from the starting location to the edge of the foreground image.

-xsize -ysize

A subregion of the foreground image may be selected using -xstart, -ystart, -xsize, and -ysize. Start values are in pixels measured from (0,0) in the upper left-hand corner of the foreground image. Size values are in pixels. Start values must be positive and size values must result in a subregion wholely contained within the foreground image.

NOTES

See the NOTES section of imcomp for a description of typical uses of compositing and notes reguarding the composite operation itself. Since imkey uses the same compositing function as imcomp, the text written for it will be helpful.

For notes regarding file format conversion and standard image tool options, see the man page on imconv(1IM).

Error messages are reported to stderr.

EXAMPLES

Key one Sun Rasterfile image onto another, only key where the pixels are red from 20 to 128:

```
imkey inone.ras intwo.ras -red 20-128 out.ras
```

Key using green in the range of 30 to 200, and move the foreground image right by 20 pixels and down by 50:

```
imkey inone.ras intwo.ras -red 30-200 -xposition 20 -yposition 50 out.ras
```

Key only using a rectangular sub-region of the foreground. The left-hand edge of the foreground will start at 10 pixels in from the left edge of foreground and will be 30 pixels wide and 40 pixels high against the background. Key on blue in the range of 0 to 128:

```
imkey inone.ras intwo.ras -blue 0-128 -xstart 10 -xsize 30 -ysize 40 out.ras
```

SEE ALSO

imadjust (1IM), imcomp (1IM), imdissolve (1IM), imfill (1IM), ImVfbAdjust (3IM) ImVfbComp (3IM)

For information on SDSC's image library, see imintro(3IM).

AUTHORS

Shane Cooper and Dave Nadeau
San Diego Supercomputer Center

See the individual file format man pages for the authors of the underlying format read and write code. The names of these man pages begin with the letters "im" followed by the format name. For example, the name of the TIFF man page is imtiff. To display it, enter man imtiff.

CONTACT

SDSC consultants, (619)534-5100, consult@y1.sdsc.edu