I Image Tools (iTools)

I INTRODUCTION

Before Side Effects Software introduced Houdini, which included the Image Compositor (COP Editor) as an integral part of the package, a wide range of image processing tools were created to manipulate the 2D images within a project. These tools – the precursors to COPs – are still available and, collectively, are called the "Image Toolkit".

The Image Toolkit consists of a conglomeration of UNIX programmes designed for the purpose of compositing and processing 2D images in various file formats. Although the COP Editor now performs many of these functions, it is sometimes convenient to use them in custom scripts, or as stand alone command line functions within a c-shell or *spy*. For this reason they are still available.

All Image Toolkit programs have on-line help. You need only to type the command name followed by a "–" within a c-shell to obtain a summary of the command usage with a brief explanation of the options. For example:

```
iblur -
```

will produce:

```
usage:iblur [-v][-C rgba][-H][-V][-W weight][-n passes]
                [sub_frame] <inframe> <outframe>
where:sub_frame := [-w width] [-h height] [-x xoff] [-y yoff]
```

options:

`-v`	Verbose.
`-C`	Any combination of RGB and alpha may be specified, restricting the blurring to those channels only. Default is RGB.
`-H`	Horizontal pass only.
`-V`	Vertical pass only (default is to do both)
`-W`	Weight of the current pixel, in range 0-1. Default is 0.5 weight of 1 leaves picture unchanged.
`-n`	Number of passes that blur operation to be performed on image.

All Image Toolkit programs read and write Houdini *(.pic)*, TIFF *(.tif)*, Targa/Vista and Wavefront run-length encoded files *(.rla)*, Alias™ RGB and mask pictures, as well as Abekas RGB and YUV images and Video Framer. Format management is invisible to the user, and is controlled by the filename suffix.

2 FAMILIES OF IMAGE TOOLS

2.1 ITOOLS

iblur – Blur Image p. 292 Blur an Image. Blurs an image and/or mask with options for horizontal and vertical low-pass filter sizes.

ibulge – Bump Image p. 294
Generate a Bumpmap Image. Generates a bumpmap image by scanning for changes in the value channel (from HSV), or from the alpha channel of the input image.

ibumpmap – Create BumpMap p. 295
Create Bumpmap from RGB Image. Creates a bump map by detecting changes in intensity.

icineon – Convert Cineon p. 296
Convert Images between PRISMS Formats. Converts images between any prisms formats including supported image types with up to 16 bits per color channel.

iclear – Clear Range p. 297
Clear a Region of an Image.

icomposite p. 298 Composite Multiple Images. A full language interpreter for compositing functions. “Duff-Porter” type operations like “A over B”, inside, dissolve, cross, multiply, plus, etc. Other operators include monochrome, difference, channel (rgba swapping/copying), and even/odd (scanline interlacing).

icp – Copy / Crop Image p. 306
Copy/Crop/Convert Images. A powerful general-purpose copying and conversion program with cropping and simple overlay features.

icurve – Lens Curvature p. 308
Simulate Lens Curvature. Distorts the input image by simulating lens curvature.

idof – Depth of Field p. 309
Depth of Field Blurring. Generates an image with depth of field blurring using a depth map image.

iexabyte – Exabyte Tape p. 310
Format Multiple Images to Exabyte Tape. Formats a series of images for output to Exabyte tape.

ifilter – Filter / Blur p. 311 Apply a Digital Filter to an Image. Filters can blur, sharpen or try to detect edges.

iflip – Flip Image p. 313 Flip an Image.

iflop – Reflect Image p. 314
Reflect an Image about Diagonal Axis.

ifractal – Fractal Image p. 315
Creates a fractal picture.

ihot – Hot Video p. 316 Find Illegal Video. Scans an image for pixels with rgb values that will produce "unsafe" values of chrominance signal or composite signal amplitude when encoded into an ntsc (or pal) colour signal.

iinfo – Image Info p. 317 Display Image Information.

ijpeg – Convert jpeg p. 319
Compress Images. Converts images between jpeg rgb format and any known prisms format.

ilookup – Gamma Correct p. 320
Apply Gamma correction or Lookup Tables. Applies colour-maps with independent RGB control.

inewtoold – Convert Old .pic p. 322
Convert Old .pic Files. Converts new .pic files to the old format (before 1993).

ipaint – Mini-Paint p. 323 Mini-Paint. A useful mini-paint tool.

iplay - View Images p. 324 View Images as Flipbook. Loads one or more images into memory and flips between images in an iris window. This command also displays image pixel values.

iprint – ASCII Values p. 333
Display Values for Image in ASCII. Produces image pixel values in various formats.

iquantize – Minimize Colors p. 334
Minimize Colour Set. This program takes a 32 bit raster image and quantizes the colour space to use a given maximum number of colours.

iramp – Convert Ramp p. 335
Convert Ramp File to Image File.

iscroll – Scroll Image p. 336
Scroll an Image. Scrolls pixel rows or columns along either the X or Y axes.

ishear – Skew Image p. 337

Skew an Image. Slants (skews) an image in either the X or Y direction. This command is also able to rotate the image to a maximum of 45˚ by invoking three shear operations in a row.

isi – Convert to Softimage p. 338

Coverts to SoftImage Format. Converts images in any Houdini format to and from Microsoft SoftImage format.

isixpack – Make Reflection Map p. 339

Pack Six Images into One Reflection Map. Creates a polar reflection map from six images.

istipple – Apply Noise p. 340

Apply Random Noise to an image. Adds controllable amounts of noise to rgb or A.

itim – Convert to Sony PSX p. 342

Convert to Sony PSX Format. Converts Houdini pictures to .tim files used by the Sony .psx format.

izmatte – Composite RGBA/Z p. 344

Composite a Series of RGBA and Z Pairs. Composites pairs of images where each pair is an rgba and Z-depth rendering of each scene.

2.2 MOVIE TOOLS

mcp – Movie Conversion p. 351

Movie Conversion. Converts a sequence of image files to a movie file.

minfo – Display Movie Info p. 353

Display Movie information. Displays information about a Houdini movie (.hmv) file.

hmvplay – Play Houdini Movies p. 345

Play Houdini Movies. Plays a Houdini Movie (.hmv) file at near to real-time speeds.

2.3 RAMP TOOL

redit – Color Ramp Editor p. 356

Colour Ramp Editor. Interactive ramp editing tool. Can be used to create ramp files usable by *iramp* and can also be used in texture applications.

3 IBLUR – BLUR IMAGE

3.1 SYNOPSIS

```
iblur  [ -v ] [-rgba] [-H] [-V] [-W weight]
  [-n passes] [sub] inimage outimage
```

where sub:
is `[-x xoff][-y yoff][-w width][-h height]`
or `[-d divs section]`

3.2 DESCRIPTION

iblur blurs an image by smearing in four directions, using a "partial sum" technique: each pixel is replaced by some combination of that pixel and the value calculated for a horizontally or vertically adjacent pixel. The algorithm runs in constant time, regardless of the weight chosen.

Four passes are made on the image. The horizontal pass consists of a left-to-right run, and then a right-to-left run on each scanline. These runs are averaged, and placed in the image. The vertical pass is similar. Hence, each pixel is replaced by some proportion of its neighbours to the right, left, top and bottom.

Example:

```
iblur -W .4 -n 3 marble.pic ip
```

3.3 OPTIONS

`-W`

Specifies the weight of the current pixel to be used when averaging with the previous pixel. The argument must be in the range [0,1] where a weight of 1 means that the current pixel will be replaced with (1 * current) + (0 * previous) or, simply put, the image will be unchanged. The default weight of 0.5 replaces the current pixel with half of the current value, plus one quarter of the previous pixel, plus one eighth of the pixel before that (and so on), resulting in a light blurring of the image.

`-n`

Lets you specify the number of times an image is to be re-blurred. This option can be used where a high amount of blur is required. Without multi-passes, the image will start to contain noticeable horizontal and vertical streaks where low weighting factors are used (<.02). The latter may be a desirable effect. If not, you can specify higher weighting factors and several passes. Each pass takes the same time as the first pass after the image is read into memory.

`-H` option restricts the blurring to the horizontal pass only. The *-V* option restricts the blurring to the vertical pass only. Any combination of *rgba* may be specified. The default *-rgb* blurs the red, green and blue channels; the alpha channel is left unmodified.

`-v` Outputs a status message to the standard error file indicating the source, resolution and nature of the operation.

4 IBULGE – BUMP IMAGE

4.1 SYNOPSIS

```
ibulge [-g] [-a] [-b #passes] <inimage> <outimage>
```

`-g`	Generate a grey level image instead of bumpmap.
`-a`	Use the alpha channel instead of value channel.
`-b`	Specify number of blur passes (default 1).

4.2 DESCRIPTION

ibulge generates a Bumpmap image by scanning the value channel (or alpha if -a is specified) of the input image, in areas where alpha coverage exists, looking for local minimum values (or dark lines). It then attempts to generate a bump surface by pulling these low values down and leaving other areas high. A circular rolloff is used between high and low regions. Because of the raster scanning algorithm used, the resulting surface can be very rough.

The *-b* option allows you to increase or decrease the amount of blurring performed. The *-g* option is mainly for visual testing of the results.

5 IBUMPMAP – CREATE BUMPMAP

5.1 SYNOPSIS

```
ibumpmap [-v] <inimage> <outimage>
```

5.2 DESCRIPTION

The output of *ibumpmap* is suitable for use with Houdini's bump mapping capability. Images are often blurred prior to being run with *ibumpmap* in order to get smooth bumps.

ibumpmap reads a standard input image and writes a bump map out to the specified output image. A bump map is an image containing 16-bit X normal vector components in the frame's red and green channels and a 16-bit Y normal component in the frame's blue and alpha channels. The output bump map is created by interpreting the input picture's luminance at each pixel to be the altitude of a surface. The resulting normals at a pixel are computed by comparing a pixel's altitude value with those at adjacent pixels.

The *-v* option outputs the information about the source and resolution of the image to the screen.

5.3 EXAMPLES

```
iblur mono.pic blur.pic
ibumpmap blur.pic bumpmap.pic
iblur image.pic stout | ibumpmap stdin bumpmap.pic
```

6 ICINEON – CONVERT CINEON

6.1 SYNOPSIS

```
icineon [-F] [-g <gamma>] <inimage> <outimage>
```

6.2 DESCRIPTION

icineon will convert images between 10 bit CINEON format and any known 8 bit Houdini format (i.e..*pic, .rla, .tga*). *icineon* will work automatically since it is included in the FBio files.

The *icineon* suffix *.cin* and *.kdk* is installed and the *$HFS/houdini/FBio* table recognizes *.cin* and *.kdk* images file names.

6.3 OPTIONS

`-F` Flip the image in Y during conversion.

`-g` Used to specify a particular gamma correction to compensate for the varying brightness. If a CINEON image is displayed by Houdini, Houdini will darken the image to correct for the gamma that is inherent in the original CINEON file.
The default gamma correction is 0.6 when converting from CINEON to Houdini and the default is set to 1.67 when converting from Houdini to CINEON.

Since CINEON files do not include alpha, *icineon* fills in the alpha channel with 1 when converting from CINEON to Houdini. All alpha is lost when converting from Houdini to CINEON.

6.4 EXAMPLES

1. To convert and flip a Houdini image file to a CINEON image file:

```
icineon -F marble.pic marble.cin
```

2. To convert and gamma correct a CINEON image file to Houdini format and output to screen:

```
icineon -g 0.8 block.kdk ip
```

3. To *iplay* an image of CINEON format:

```
iplay image.cin
```

7 ICLEAR – CLEAR RANGE

7.1 SYNOPSIS

```
iclear [-v] [-c r g b [t]] [res] [sub_frame] <image>
```

where:

`-c` is the color to clear with (values 0.0 - 1.0)
`res` is `[-w` *`width`*`][-h` *`height`*`]`
`sub_frame` is `[-x` *`xoff`*`][-y` *`yoff`*`]` or `[-d` *`divs image`*`]`

7.2 DESCRIPTION

iclear clears a region of an image. Without any options *iclear* clears the entire image with black. The *-v* option will output a status message to the standard error file indicating the source, resolution and nature of the operation.

7.3 EXAMPLES

To create a red image of resolution 400×300:

```
iclear -400 -h 300 -c 1 0 0 red.pic
```

To put blue in a small portion near the centre of a previously saved file:

```
iclear -c .1 .1 .7 .9 -w 64 -h 60 -x 200 -y 200 savefile.pic
```

8 ICOMPOSITE

8.1 SYNOPSIS

```
icomposite [-v] [-w width][-h height][-f][-c][command]
```

8.2 DESCRIPTION

icomposite composites layers of images, provides some simple image processing, and resizes images. Arbitrary filters can be specified when scaling images.

8.3 OPTIONS

`-v` — Operate verbosely by outputting the current scanline and the current composite sub-operation as they are executed.

`-w width` — The output horizontal resolution is set to *width*.

`-h height` — The output vertical resolution is set to *height*. Normally, all frames are scaled in X and Y to match the resolution of the largest input image. Occurrence of either the width or the height options, cause the input and output frames to be scaled to the specified values, while the vertical resolution is set to preserve the width/height ratio of the largest input image.

`-f` — The input images are filtered (using a box filter) to the desired output resolution. Normally, the input images are sampled. Filtering takes longer than sampling, but gives better results when scaling images.

`-c` — Normally, *ic* will terminate the current composite operation if an error occurs when opening or reading an image file. The *-c* option indicates that the composite operation should continue even if such an error occurs. If this happens, a blank image is loaded in place of the one which caused the error. Note that this may cause the resulting composite to be incorrect.

`-B` — Backwards compatible (Houdini 2.5 and prior) image scaling. No extra filter support.

`-F` — Specify filter type for reading of images (entering “icomposite -” lists the available filters).

command — Any single input line which can be supplied to *ic* in interactive mode can also be supplied here.

8.4 COMPOSITE OPERATORS

A unary operator is applied to one image such as:

```
dissolve 0.7 back.pic
```

A binary operator is applied to two images, such as:

```
claw.pic over back.pic
```

The *0.7* is a *Constant***,** a positive real number. Constants, except those used in a constant image, cannot exceed one.

A constant image is an image containing one solid color. It may be included as a composite element (typically for use as a background element) and is specified as red, green, blue and, optionally, alpha values enclosed within square brackets:

```
[1 0.7 0.5]
[0.8 0.5 0.4 0.8]
```

The second example includes alpha. In the first example, alpha is not present and defaults to one. The constants should be real values in the range zero to one.

The compositing operators are defined as follows:

`dissolve <value> image`
: The subject image is dissolved by the amount given as the constant with the operator. This means that the red, green, blue and alpha channels are all multiplied by the supplied value.

`darken <value> image`
: The subject image is darkened by the amount given as the constant with the operator. This means that only the red, green and blue channels are multiplied by the supplied value. The alpha channel is left unmodified by the darken operator.

`opaque <value> image`
: Only the alpha channel of the subject image is multiplied by the amount given as the constant with the operator. This has the effect of creating a luminescent element which adds color information without obscuring the background elements.

`mono <value> image`
: The subject image is made more monochrome by the amount given, where one is fully monochrome, zero leaves the colors unchanged, and values between result in partially monochrome images.

`gamma <value> image`
: Gamma correction is applied to each of R, G and B of the subject image. A value more than one makes the picture brighter, and vice versa.

`channel chans image`
: The subject image has its channels copied, swapped or replaced according to the combination and order of the four characters in the supplied *chan* string. The first character represents what is placed in the output red channel, the second character the output green channel, etc. The character *r* represents the input red channel, *z* represents a channel containing zero (black), and *o* represents a channel containing one (full value). For example, *bbba* copies blue into the red and green chan-

nels, while blue and alpha remain unchanged. *rgba* leaves the picture unchanged. *aaaz* copies alpha into the RGB channels and zero in the alpha channel.

`alpha <value> image`
The subject image has its red, green and blue channels multiplied by its alpha channel.

`setup <value> image`
The subject image is boosted by *value* in pixels where alpha is greater than zero, and is untouched where alpha is zero. The *setup* operator will boost low luminance levels of an image (based on alpha) to be above the given lower limit, usually to accommodate zero-black keying. For example:

```
icomposite abekas:34.rgb = setup .075 claw.pic
```

If there is non-zero alpha in a pixel, the luminance would be boosted to be a minimum of .075. Any colors in the image over .075 luminance would remain unchanged.

`even image`
Maintain only the even scanlines of the subject image (i.e. erase all odd scanlines to black). Since NTSC frames result in the bottom scanline being odd, this operator assumes that the first scanline of an image file is odd.

`odd image`
Maintain only the odd scanlines of the subject image (i.e. erase all even scanlines to black). Since NTSC frames result in the bottom scanline being odd, this operator assumes that the first scanline of an image file is odd.

`image cross image`
The cross operator is the only binary operator which requires a constant operator value. This operator performs a cross-dissolve of two elements. The operation "A cross 0.4 B" is equivalent to dissolve 0.6 A plus dissolve 0.4 B. It is provided as a separate operator for efficiency and convenience.

`image over image`
The over operator is likely the most commonly used composite operation. It places its left operand over top of its right operand. The foreground element survives everywhere, while the background element survives only where the foreground does not have full coverage.

`image plus image`
The plus operator results in the simple addition of each channel of the left and right operands. The sums are clipped to the maximum channel value.

`image minus image`
The minus operator results in the simple subtraction of each channel of the right operand from the left operand. The results are clipped to the minimum channel value of zero.

`image diff image`
The diff operator results in the absolute value of the difference between the right operand from the left operand.

`image multiply image`
The multiply operator results in the simple multiplication of each channel of the left and right operands. Channel values are treated as fixed point values between zero and one for the purposes of this operator.

`image inside image`
The inside operator results in the survival of the left operand anywhere that the right operand has coverage.

`image outside image`
The outside operator results in the survival of the left operand anywhere that the right operand does not have coverage.

`image atop image`
The atop operator places its left operand atop its right operand. The result is that the left operand survives wherever the right operand has coverage and elsewhere only the right operand survives. Thus, A atop B is the union of A inside B and B outside A.

`image xor image`
The *xor* (exclusive or) operator provides the union of A outside B and B outside A. In other words, the result is everything that does not overlap in the two elements.

8.5 GRAMMAR

icomposite has a formal grammar. The allowed syntax is defined by the following.

The unary operators *dissolve, darken, opaque, mono, gamma, alpha, even* and *odd* have equal precedence which is higher than the binary operators.

Binary operators are left associative (i.e. *a over b over c* is equivalent to *(a over b) over c).*

```
Composite: FILENAME = Expression

Expression: FILENAME
  | Constant_Frame
  | dissolve Constant Expression
  | darken Constant Expression
  | opaque Constant Expression
  | mono Constant Expression
  | gamma Constant Expression
  | channel Channels Expression
  | alpha Expression
  | even Expression
  | odd Expression
  | Expression cross Constant Expression
  | Expression plus Expression
  | Expression minus Expression
  | Expression diff Expression
  | Expression multiply Expression
```

```
  | Expression over Expression
  | Expression atop Expression
  | Expression outside Expression
  | Expression inside Expression
  | Expression xor Expression
  | (Expression)

Constant_Frame: [Constant Constant Constant Constant] |
                        [Constant Constant Constant]
```

icomposite performs compositing operations as described in the paper "Compositing Digital Images" (*Computer Graphics*, Volume 18, Number 3, Thomas Porter and Tom Duff, July 1984). All elements used in composite operations (except the alpha operator, described below) assume that the red, green and blue channels have been pre-multiplied by their corresponding alpha channel as do most renderers. This means that frames without pre-multiplied r, g, b channels cannot be used directly in composite operations, except possibly as background elements.

The *alpha* operator converts a non-pre-multiplied image into the required pre-multiplied format. Thus, any frames without pre-multiplied color channels should first have the *alpha* operator applied to them (see examples below).

8.6 EXAMPLE

```
csh-> icomposite ip = mono .5 $HH/pic/Mandril.pic
```

icomposite Either gets one command from the command line or it reads image composite sentences from its standard input and performs the operations required to generate the requested composite image.

When reading commands interactively, each line of input is a single composite operation or sentence. Environment variables denoted with a leading $ are first expanded.

more examples

The following are some examples of composite sentences which may be provided as input to *ic*. Some samples are fairly complex. The use of `(...)` signals to *ic* to do the operations between parentheses first. The use of `[...]` means it is an image with a constant color given by the three RGB or four RGBA numbers inside:

```
cd $HFS/demo/ImageTools
ip = claw.pic over back.pic
junk.pic = claw.pic outside back.pic
ip = dissolve .5 claw.pic over ([0.3 0.1 0.1 0.25]
atop back.pic)
ip = marble.pic multiply [.5.7 1]
ip = back.pic minus [0.25 0 0] over [0.3 0.2 0.1]
fields.pic = even claw.pic plus odd back.pic over [.2.1.1]
ip = (darken 0.7 (object.pic inside shadow.matte) atop
object.pic) over .bg.pic
ip = (marble.pic cross 0.7 air.pic) outside claw.pic over back.pic
ip = gamma 1.3 back.pic
```

filtering examples

```
icomposite -F gauss -w 640 -h 486 ip = $HH/pic/Mandril.pic

icomposite -F box -w 640 ip = $HH/pic/Mandril.pic
```

8.7 INTERACTIVE COMPOSITE MODE

If no composite *command* is given on the command line, *icomposite* goes into its interactive mode where the user can type many commands. If input is from a terminal then *icomposite* prompts for a composite sentence with the string `=>`.

Interrupt signals (Ctrl C) will terminate the command unless a composite operation is under way, in which case, only the in-progress composite is interrupted. If it is not being run interactively, then interrupt signals will always terminate the command.

Each input line is individually interpreted until the end of file is reached.

An input line starting with the character '!' is interpreted as a UNIX command and is executed accordingly.

8.8 NOTES

icomposite does not accept file names containing any of the following characters: `[] () = ; !` or filenames that look like a real number (e.g. 123).

Errors are handled in a simplisticly, so you will typically get something like, "Syntax error on line 1".

9 ICONVERT – CONVERT IMAGES

9.1 SYNOPSIS

```
iconvert [-d depth] infile outfile [tag tagvalue] ...
```

9.2 DESCRIPTION

Converts an image of one type to another type. The type is determined by the extension on the file.

The -d option can specify a channel depth of 8 or 16 bits per channel to override the input file depth.

9.3 SUPPORTED FORMATS

• PRISMS	PRISMS/Houdini Image Format: 8 bit RGBA channels
• TIFF	Version 3.4. Allows the tags: compression none LZW
• SGI	SGI classic image format
• .pic .Z	Compression is done using the compress utility
• RAT	Houdini Random Access Texture
• iplay	Display image in an *iplay* window
• mdisplay	Display image in an mdisplay window. Additional images are displayed in the same window
• Jpeg	Allows the following tags: quality integer value
• Cineon	Cineon Image Format
• RLA	RLA Wavefront RLA/RLB Format
• Alias	Alias Image Format
• Bitmap	Windows BMP Bitmap format
• Softimage	Softimage Picture Format
• Abekas	Abekas RGB/YUV
• Targa	Targa/Vista
• Vertigo	Vertigo RLE
• IOTable	The IOTable supports the following image extensions:

.si	Read/Write
.jpg	Read/Write
.JPG	Read/Write
.jpeg	Read/Write
.JPEG	Read/Write
.ice	Read Only
.qtl	Read Only
.cin	Read/Write
.kdk	Read/Write
.tif16	Read/Write
.fit	Read/Write
.gif	Read/Write

.gif89	Read/Write
.GIF	Read/Write
.GIF89	Read/Write
.tx	Read/Write
.acc	Read/Write
.accom	Read/Write
.xwd	Read/Write
.qtl	Read Only
.fit	Read/Write
.gif	Read/Write
.gif89	Read/Write
.GIF	Read/Write
.GIF89	Read/Write
.tx	Read/Write
.acc	Read/Write
.accom	Read/Write

For a list of extensions, please check the file: *$HH/dso_fb/index* .

10 ICP – COPY / CROP IMAGE

10.1 SYNOPSIS

```
icp [-v] [-u] [-m] [-b] [res] [sub] <inimage> [sub] <outimage>
iconvert [-d depth]
```

where:

```
res  is [-w width][-h height]
sub  is [-x xoff][-y yoff][-w width][-h height]
or [-d divs section]
```

10.2 DESCRIPTION

icp copies an image, or a portion thereof, from the *inimage* image to the *outimage* .

Without any arguments, *icp* copies an entire image from the specified source to the destination. At any time, if the resolutions of the source image and the destination image differ, *icp* will copy the largest portion of the source which will fit within the destination image.

If *outimage* is of a different file type than *inimage*, then the file is converted to the new filetype automatically. For example:

```
icp myimage.sgi myimage.tif
```

Converts *myimage.sgi* to a *.tif* format file.

If the destination image is an existing file the *-m* option indicates to *icp* that the file is to be modified rather than re-created. This allows sub-frames to be copied into existing frames stored on disk without destroying the portion of the destination image outside the sub-image area.

To avoid the internal allocation of memory for image storage, frames are copied on a per scanline basis, but may be overridden by specifying the *-b* block option, which reads/writes entire images in one operation.

To keep images from being compressed, you can use the -u option (uncompressed). This avoids run-length encoding (RLE) for output formats that support it. This includes, Prisms/Houdini (.pic), Wavefront (.rla/.rlb) and Targa (.tga/.vst).

10.3 RESOLUTION + OFFSET (RES)

An arbitrary portion of an image may specified by defining its resolution and its offset from the origin of the image. The width and height of the sub-area are specified with the *-w* and *-h* options respectively. If one is omitted, its value will be the full width or height of the image. These options must be specified before the source image on the command line.

Either the source or the destination image may have offsets specified for the sub-area. These are given with the *-x* and *-y* options. The offset options appearing in front of the source image apply to the source image, while those specified immediately before the destination apply to the destination. If one or both are omitted, their values default to zero.

If offsets are specified without a resolution, then *icp* adjusts the resolution of the image appropriately for the given offset values.

10.4 DIVISIONS + IMAGE (SUB)

A sub-image may be specified for either the source or the destination by specifying the *-d* option before the source and/or the destination. This option takes two values representing the number of sub-frames into which the image is to be divided and the sub-image number to be used for this copy operation.

The number of sub-divisions must be the square of an integer. The sub-image number must be between zero and the number of sub-divisions -1 where sub-image 0 is at the top left corner and sub-image sub-divisions -1 is at the bottom right.

As with the offset specifications, the option appearing in front of the source image applies to the source image, while that specified immediately before the destination applies to the destination.

The two methods of specifying sub-areas of an image may be mixed so that one method is given for the source while the other is used for the destination. As mentioned above, any mis-match of resolutions will result in copying the largest portion of the source which will fit within the destination image area.

The *-v* option will output a status message to the standard error file indicating the source, destination and resolution of the copy operation.

10.5 EXAMPLES

To copy the top-right corner of the IRIS into the top-right corner of a previously saved file with 512×384 resolution:

```
iclear -w 800 -h 600 temp.pic
icp -m -d 4 1 marble.pic -d 4 1 ip
```

To display the top half of an image:

```
icp -y 242 back.pic ip
```

11 ICURVE – LENS CURVATURE

11.1 SYNOPSIS

```
icurve [options] inimage outimage
```

11.2 DESCRIPTION

icurve will distort the input image by simulating lens curvature.

11.3 OPTIONS

`-p#`	1-Ordinary projection r'= f tan(angle) 2-Stereographic projection r'= 2f tan(angle/2) 3-Orthoscopic projection r'= f sin(angle) 4-Equisolid angle projection r'= 2f sin(angle/2) (default is: 3-Orthoscopic projection)
`-f` *length*	The focal length in mm (default 10 mm).
`-h` *angle*	The hemispherical field between 0 and 180 degrees (default 180).
`-n`	Don't automatically fit the image to the lens.
`-s` *size*	Size of a pixel in mm (default 0.26 mm).

12 IDOF – DEPTH OF FIELD

12.1 SYNOPSIS

```
idof [options] inimage zdepthmap outimage
```

12.2 DESCRIPTION

idof will generate an image with depth of field blurring using a Depth map image which can be made with *mantra*.

12.3 OPTIONS

`-l length`	Focal length of the lens in mm (default 55 mm).
`-a aperture`	Aperture f-stop value. (default 5.6).
`-f distance`	Point of focus in picture in mm (default 1000 mm).
`-s size`	Size of a pixel in mm (default 0.28 mm).
`-d scale`	Scale factor for Z depth images (default 1.0).

Note: The Z-Depth map units are assumed to be in millimeters. Use the -d option to scale the depths if necessary.

There are two test images for *idof* in: *$HFS/houdini/pics* called *DepthScene.pic* and *DepthMap.pic*.

12.4 RENDERING DEPTH PICTURES FROM MANTRA FOR IDOF

When rendering from *mantra*, you can generate two types of Z-depth images:

- Closest Z-Depth
- Average Z-Depth

These are covered in the *User Guide* > *Rendering* section.

When rendering images for *idof*, you should ensure that you're rendering using Closest Z-Depth. Using Average Z-Depth may result in any single surface not rendering unless there is another surface behind it, and it will set the value of all the pixels that want blurring to 0.

13 IEXABYTE – EXABYTE TAPE

13.1 SYNOPSIS

```
iexabyte [-f start end] [-i inc] [-v] filepattern
```

13.2 DESCRIPTION

The *iexabyte* program reads images and outputs them to a file format compatible with the Abekas save/restore 8mm streaming tape format. The output of *iexabyte* can be sent to the 8mm streaming tape, inserted in an Abekas tape drive and read off.

All images which are read into *iexabyte* are filter-scaled to the Abekas resolution (720×486). A PAL version of *iexabyte* will be implemented on demand.

If the *-f* option is used, the *filepattern* should contain a *$F* which represents the frame number (i.e. between *start* and *end* by *inc*).

For example:

```
iexabyte -f 1 50 -i 2 \$F.pic.Z
```

will create a tape file of images from

```
1.pic.Z, 3.pic.Z, ... , 49.pic.Z, 50.pic.Z
```

iexabyte looks for an environment variable *EXABYTE* which is the resolution of the images written to tape. Usage:

```
setenv EXABYTE 720 575
iexabyte *.pic >/dev/tape
```

iexabyte does not work on PAL systems.

14 IFILTER – FILTER / BLUR

14.1 SYNOPSIS

```
ifilter [-5] [-v] [-C rgba] [-t type | -m matrix]
  [sub] inimage outimage
```

where *sub*

is: `[-x xoff][-y yoff][-w width][-h height]`
or: `[-d divs section]`

14.2 DESCRIPTION

ifilter applies a digital filter to an image. Digital filters may be used to blur, sharpen or manipulate images in other ways. Several filters are pre-defined. However, it is possible to define your own filter by specifying the 3×3 or 5×5 matrix.

One of the primary uses of filtering is the reduction of high frequency noise (i.e. aliasing). The bartlet, hamming and blackman filters are digital filters suitable for this purpose. The filtering does not scale the image down, so the resulting image will appear to be slightly blurred.

Sharpening the image is done by increasing the contrast on adjacent pixels. Edge detection attempts to find edges by finding areas of high contrast.

14.3 OPTIONS

Any combination of *rgba* may be specified. The default *-rgb* filters the red, green and blue channels; the alpha channel is left unmodified.

The *-v* option outputs a status message to the standard error file indicating the source, resolution and nature of the operation.

The *-5* option sets up *ifilter* to use 5×5 filter matrices. The default is 3×3. The 5×5 matrices cover a larger image area so more pixel values are evaluated on each application of the filter. The effects of this can be seen with any of the bartlet, hamming or blackman filters where the apparent blur increases.

The -C option specifies the channels to be filtered. Any combination of r, g, b and a can be used. Only the channels specified will be filtered.

The type of filter can be specified by the *-t* option. The available filter types are:

- *rectangular* standard box filter
- *bartlet* triangular filter
- *hamming* bell curve shaped filter
- *blackman* sine curve shaped filter
- *sharpen* increases contrast in image
- *edge* try to detect edges

A user-specified matrix can be supplied by using the *-m* option. Following the *-m*, there should be a list of integers specifying the filter coefficients.

This option overrides the *-t* option. To specify a 5×5 user matrix, the *-5* option must appear on the command line before the *-m* option. For example:

```
ifilter -m -1 1 -1 1 4 1 -1 1 -1 in.pic out.pic
```

or

```
ifilter -5 -m c00 c01 c02 c03 c04 c10 c11...c44 in.pic out.pic
```

14.4 NOTES

- All matrix coefficients should be integers.
- Floating point coefficients are not supported.

15 IFLIP – FLIP IMAGE

15.1 SYNOPSIS

```
iflip [-v] [-X] [-Y] [sub] inimage outimage
```

where *sub*

is `[-x xoff][-y yoff][-w width][-h height]`
or `[-d divs section]`

See also *icp – Copy / Crop Image* p. 306.

15.2 DESCRIPTION

iflip exchanges the rows, columns, or both rows and columns of an image or sub-frame within a image.

The *-X* option interchanges rows, *-Y* interchanges columns, and specifying both will interchange both rows and columns.

If the *-v* option is specified, *iflip* will display the source, resolution and nature of the operation.

15.3 EXAMPLES

```
iflip -X -Y back.pic ip
iflip -Y -d 9 4 back.pic ip
```

16 IFLOP – REFLECT IMAGE

16.1 SYNOPSIS

```
iflop [-v] [-m] [sub] inimage [sub] outimage
```

where *sub*

is `[-x xoff][-y yoff][-w width][-h height]`
or `[-d divs section]`

16.2 DESCRIPTION

iflop reflects an image, or a portion thereof, of the *inimage* and copies it to the *outimage*. Columns of the input image become rows of the output, and the output resolution is the opposite of the input resolution.

Without any arguments, *iflop* copies an entire reflected image from the specified source to the destination. At any time, if the resolutions of the source image and the destination image differ, *iflop* will copy the largest portion of the reflected source which will fit within the destination image.

If the destination image is an existing file, the *-m* option indicates to *iflop* that the file is to be modified rather than re-created. This allows reflected sub-frames to be copied into existing frames stored on disk without destroying the portion of the destination image outside the sub-image area.

Refer to *icp – Copy / Crop Image* p. 306 for details on *res* and *sub.*

The *-v* option will output a status message to the standard error file indicating the source, destination and resolution of the reflect/copy operation.

16.3 EXAMPLES

```
iflop back.pic tall.pic
icp tall.pic ip
```

To reflect/copy a small portion near the center of a previously saved file to a subimage of the IRIS:

```
iflop -w 64 -h 60 -x 200 -y 200 back.pic ip
```

17 IFRACTAL – FRACTAL IMAGE

17.1 SYNOPSIS

```
  ifractal [-v][-n][-S seed][-s scale][-r roughness]
  [-o offset] [sub_frame] <image>
<seed> is an integer used as a random number generator seed (1234)
<scale> is between 0 and 1, default .27
<roughness> is between 0 and 1, default .77
<offset> is between 0 and 1, default .5
<sub_frame> := [-w width] [-h height] [-x xoff] [-y yoff]
                      | [-d divs frame]
<sub_frame> is ineffective now: ifractal makes a 512x512 pattern

-n instructs ifractal to omit the creation of noise in the alpha
  channel as it is already present from a previous run of ifractal.
  Noise creation is the first step.
-v is the verbose option
```

17.2 DESCRIPTION

ifractal creates a 2D fractal pattern in image using a recursive subdivision method. The image is output in monochrome at 512×512 resolution only.

The seed initializes the random number generator, and defaults to 1234. A random number pattern is left in the alpha channel, and is used to create the visible pattern. The regeneration of the noise pattern in the alpha channel may be avoided by giving the *-n* option, which reuses the alpha channel if the image already exists.

scale and *roughness* are similar to brightness and contrast, and default to 0.27 and 0.77 respectively. *offset* is similar to a minimum brightness and defaults to 0.5. All three parameters must be between 0 and 1.

18 IHOT – HOT VIDEO

18.1 SYNOPSIS

`ihot [-b|-w] [-p]` *`inimage outimage`*

where:

`-b`	Flag hot pixels with black.
`-w`	Flag hot pixels with white.
`-p`	Use PAL video standards.

18.2 DESCRIPTION

This program scans an image for pixels with RGB values that will give "unsafe" values of chrominance or composite signal amplitude when encoded into an NTSC or PAL. This happens for certain high-intensity and high-saturation colours that are rare in real scenes, but can be easily present in synthetic images.

By default the offending pixels are made "safe" by reducing their intensity (luminance) while leaving hue and saturation the same. If one of the *-b* or *-w* options is specified, hot pixels are flagged so you can choose other colours.

19 IINFO – IMAGE INFO

19.1 SYNOPSIS

```
iinfo [-i] [-s] imagefile [imagefile*]
```

where:

`-i` Checks the image file's integrity. If the file is corrupted in any way, the line: *Integrity: File bad* will appear in the output. Otherwise, *Integrity: File OK* will appear. If any of the read files read are bad, *iinfo* will exit with error status 1; if they are all good, it returns status 0.

`-s` Suppresses output. When used in conjunction with the *-i* option, the exit status can be used in shell scripts to test the integrity of image files.

19.2 DESCRIPTION

Displays information about the selected files to standard output.

For Houdini, TARGA/VISTA or Wavefront picture files, the information displayed includes:

- Width and Height (Resolution)
- X and Y offsets
- Bytes Per Pixel
- Format (R8, G8, B8, A8, R8G8B8, or R8G8B8A8)

For ALIAS picture files, the information displayed includes:

- Width and Height (Resolution)
- X and Y offsets
- Number of Bit Planes

19.3 EXAMPLE

```
!iinfo HeadRender1b.tif Bouguereau.jpeg WebvertImage.gif
File:          HeadRender1b.tif (TIFF format)
Image Count:   1
Resolution:    640 x 395
Data:          byte
Color Model:   rgba
Channel Depth: 8

File:          Bouguereau.jpeg (JPEG format)
Image Count:   1
Resolution:    320 x 197
Data:          byte
```

```
Color Model:   rgb
Channel Depth: 8

File:          WebvertImage.gif (IOTable format)
Image Count:   1
Resolution:    468 x 60
Data:          byte
Color Model:   abgr
Channel Depth: 8
```

20 IJPEG – CONVERT JPEG

20.1 SYNOPSIS

`ijpeg [-d depth] inimage outimage [quality value]`

where:

inimage / outimage	Filenames of source and destination image files.
-d *depth*	Spceify a channel depth of 8 or 16 bits to override the input file depth.
quality *value*	Specify the JPEG quality (1-100).

20.2 EXAMPLE

```
ijpeg myImageFile.tif JpegFile.jpeg quality 70
```

20.3 DESCRIPTION

Converts an image to/from JPEG RGB format and any other Houdini image format (i.e. .pic, .rla, .tga). One image must have the *.jpeg* or *.jpg* extension. If you specify a JPEG (.jpg or .jpeg extension) image as the *inimage*, then it will convert from JPEG format, if you specify JPEG as the *outimage*, then it will convert to JPEG format. Any alpha channel data is discarded.

The JPEG algorithm is 'lossy' which means you will lose some of the exact pixel-by-pixel data contained in your image, with the benefit that you can obtain a very substantial savings in stored file size. Compression achieved can be up to 50:1, meaning a 2Mb file will compress to about 40Kb. The quality setting allows to specify the amount of compression-to-quality trade-off.

Note: stdin.jpeg and stdout.jpeg are recognized filenames which represent standard input/output.

21 ILOOKUP – GAMMA CORRECT

21.1 SYNOPSIS

```
ilookup [-g|G <gamma>] [-a|A <cmap>] [-v] [-m] [<res>] [<sub>]
  [inimage] <sub> [outimage]
-g <gamma>    floating point gamma correction on rgb
-G <r g b a>  floating point gamma corrections on rgb and alpha
-a <cmap>     reads color map file and applies to rgb
-A <cmap>     reads color map file and applies to rgba
        res:= [-w <width>] [-h <height>]
```

where *sub*

is `[-x <xoff>] [-y <yoff>]`

or `| [-d <divs> <frame>]`

21.2 DESCRIPTION

ilookup copies an image from *inimage* to *outimage* and modifies the pixel colors according to either the supplied gamma-correction number or an RGB and Alpha color lookup table.

A color lookup table (LUT), which is input by *ilookup*, is a three or four-column array of numbers.

Under the *-a* option, columns 0, 1 and 2 of the LUT, which is read from any channel file *lut*, map the red/green/blue channels respectively to new color values. The array may have any number of rows, and is internally re-sized to be 256 rows, each row corresponding to one of the 256 possible pixel values. Array values are expected to be in the range 0 to 1, and are clipped outside that range.

The option *-A* operates the same as *-a* except it also applies the last column of the maximum four-column lookup table to the frame's alpha channel.

The *-g* option applies gamma correction to an image. Gamma is a floating point number used in a power function, *(pixel_value/255)**gamma*. A value of gamma greater than 1 makes the picture brighter, and vice versa. Pixels at values 0 and 255 are unchanged. *-g* does not affect alpha.

The *-G* option is similar to *-g* but accepts four gamma values, one for each of red, green, blue and alpha.

21.3 SEE ALSO

- Refer to *icp – Copy / Crop Image* p. 306 for details on *sub.*

22 IMDISPLAY – VIEW SEQUENTIAL IMAGES

22.1 SYNOPSIS

```
imdisplay image
```

or:

```
imdisplay width height format
```

22.2 DESCRIPTION

imdisplay copies an image to an *mdisplay* window (i.e. like *mantra* rendering to *iplay*). If an *mdisplay* window is already open, the same window will be used to display the image (unless the server has been disconnected).

This allows things like COP render sequences render to a single iplay-like window. To use this, specify “md” instead of “ip” when specifying the output image name.

The imdisplay program can also be used to allow other renderers (or other applications) to take advantage of the *mdisplay* capabilities.

23 INEWTOOLD – CONVERT OLD .PIC

23.1 SYNOPSIS

```
inewtoold new_image_input old_image_output
```

23.2 DESCRIPTION

The magic number of a file is a special code which is used to determine what type and format of data is contained in the file. This means that old programs will not be able to read images created with the new programs (shipped with Prisms 5.3.2). However, the new images can be converted to the old format by using the *inewtoold* program.

The old magic number (before 1993) used to identify *.pic* files was exactly the same as RenderMan's magic number leading to a variety of problems. The new magic number for *.pic* files has since been changed. All image tools now create images using the new magic number. Any images with the old magic numbers will still work with any Side Effects image tool.

Old magic number (in hexadecimal format): `80e8 0000`

New magic number: `ebe8 0000`

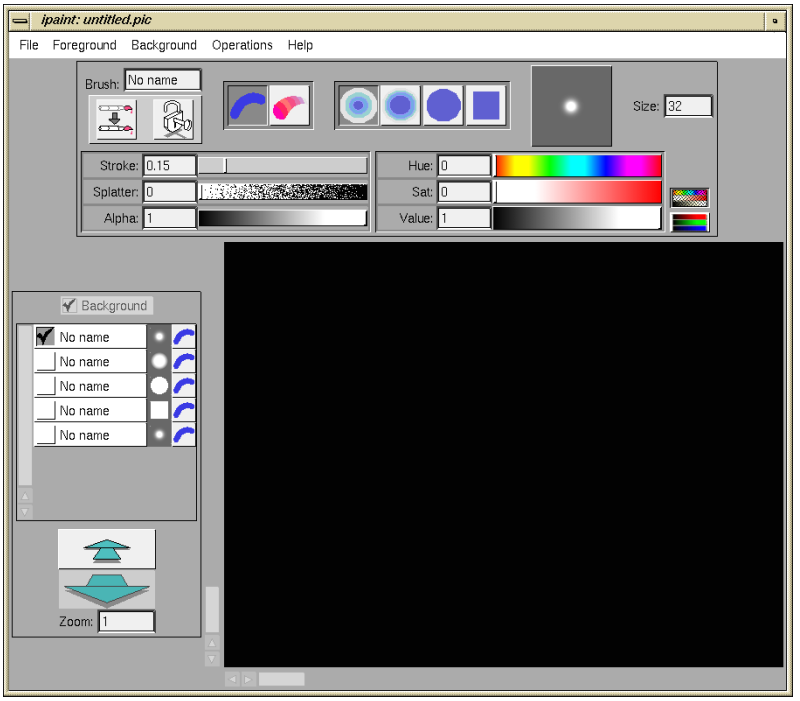
If necessary, the program *inewtoold* will convert a new *.pic* to an old *.pic* file. If necessary, old *.pic* files can be updated to the new magic number using *icomposite* by inputting the old sequence and outputting the new sequence.

24 IPAINT – MINI-PAINT

24.1 DESCRIPTION

ipaint, is the Houdini Paint tool. It is designed for use in preparing:

- images to be used as repeatable textures
- reflection maps, bump maps and backgrounds
- mattes from RGB images which do not contain alpha
- painting existing images

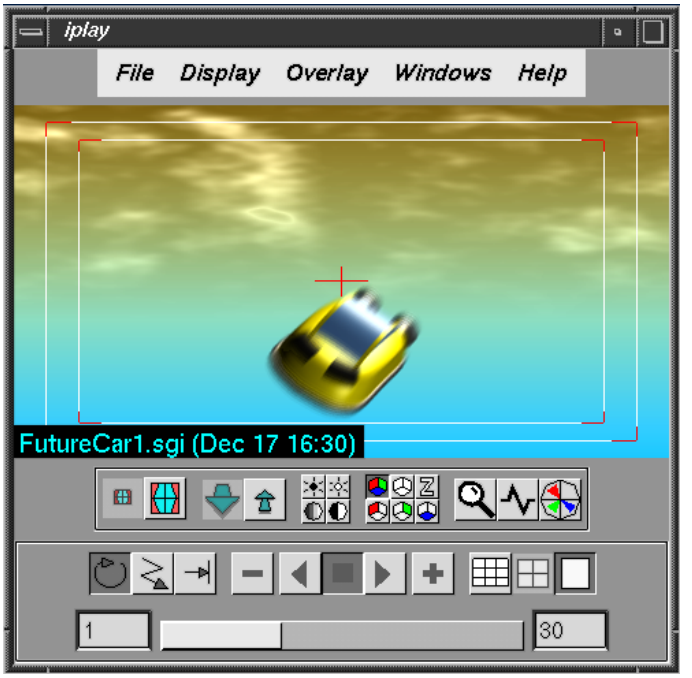


24.2 FEATURES

ipaint features an air brush, and smear, splatter and erase brushes. It can store a library of named brushes. It zooms/pans images of any resolution. It scrolls 50% to allow work on edges of images which must match up. You can paint a foreground image over a locked background and swap foreground/background or fuse foreground into back. *ipaint* can also paint-through and pull-up background images to the foreground.

Tip: In general, set your image into the background, and paint on the foreground. This way you can use as a magic eraser to reveal the background image.

25 IPLAY - VIEW IMAGES



Note! As of Houdini 5, mPlay replaces iPlay – you should use mPlay instead!
See: *COPs > mPlay* ***p. 491 for complete info.***

25.1 DESCRIPTION

iplay displays one or more images in sequence. It is the standard program that Houdini uses for displaying images, or a sequence of images.

While *iplay* is loading images you can scroll through frames and make adjustments. You can also *Clone* the *iplay* window by selecting from the *Windows* menu. This enables you to view image(s) at different settings simultaneously.

You can use *iplay* indirectly from any of the renderers or Image toolkit by specifying the device *ip*. This will use *iplay* for reading and displaying an image from the output of an Image toolkit program.

HOUDINI MOVIE PLAYER

If you need to play Houdini Movie (.hmv) files, use the *hmvplay* program, described in *hmvplay – Play Houdini Movies* p. 345.

25.2 IPLAY MENUS

If no menubar is displayed in the *iplay* window, click with the to toggle various options on/off, including the menubar.

FILE

Image Save...	Save the current frame to disk.
Image Insert...	Insert images before the current frame.
Image Append...	Add images after the last frame.
Audio Load...	Load an *.aiff* audio file from disk.
Audio On	If audio is loaded, turn it on/off
Preferences	Brings up the Preferences dialog
Quit	Quits *iplay.*

DISPLAY

Full Interface	Turn on/off all tools.
Menus	Turn iplay's menubar on/off.
Scrollbars	Toggles scrollbars on/off. Use for large images.
Toolbar	Toggles the Toolbar on/off. The Toolbar includes: size, zoom, magnify, brightness, contrast, RGB, alpha and Z-depth display controls.
Frame Control	Toggle Frame Controls on/off. They include: Loop, Zig-zag, Stop, Reverse, Play, Stop, Forward, Filmstrip, Current, and Rate controls.
Range Control	Toggles Range Controls on/off. They allow you to select a subset of frames to playback.
Audio	Turn audio on/off. Also allows you to set the audio off-set.
Inspect	Toggle information about overall image and pixel currently under cursor on/off.
Border	Toggle window borders on/off. To get the image only, and nothing else, turn off all tools.
DoubleBuffer	The double buffer option is useful to avoid flicker when large images sequences are being played.

OVERLAY

Field Guide	Toggle field guide overlay on/off
Safe Area	Toggle safe area overlay on/off.
Black Overlays	Make overlays black instead of white

Frame Number	Display the current frame number in the bottom-left corner of the image.
Filename	Display the name of the current file in the bottom-left corner.
Time Stamp	Displays the file's time of creation (if possible). If the image is loaded from stdin, or rendered into *iplay*, the time in which it was loaded into *iplay* is displayed.

WINDOW

Magnify	Opens a magnifying-glass window which displays a close-up of the area currently under the cursor.
Clone	Clones the current *iplay* window so you can view it with different settings.

25.3 BUTTONS & CONTROLS

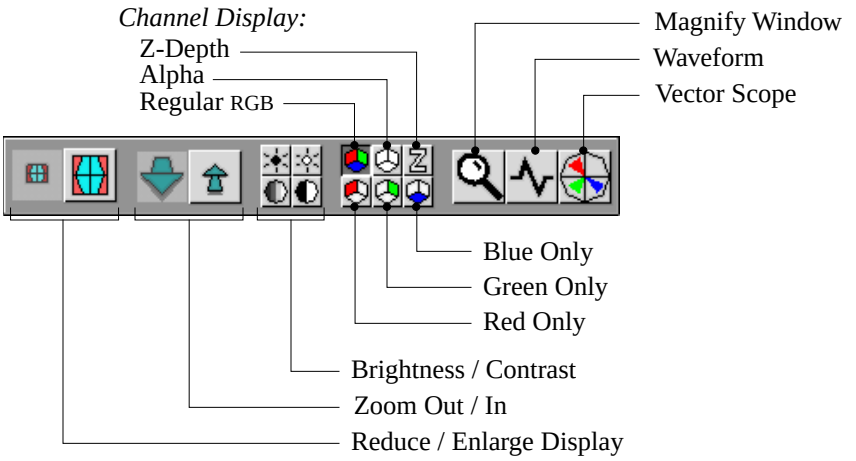
- You can control the playback direction, real-time speed, zoom-factor, and display of alpha mattes. You can magnify a section, get a display of RGBA HSV and XY values, and can write the image to a disk file.
- Use the middle mouse ( ) to pan large images.
- Click within the image with the right mouse ( ) to display a pop-up menu.
- Change the image magnification by clicking on one of the *Zoom* buttons. The images can be played at close-to-realtime rates by clicking on the Speed control to select *Realtime* and making the *Images/sec* the same as the sequence that was loaded into *iplay.*

25.4 KEYBOARD SHORT-CUTS

Images can be flipped through by clicking the *Forward*, *Reverse* and *Stop* buttons. You can click and drag the frame slider, increment one frame at a time, or use the following keys:

↑	Play forward
↓	Play reverse
→	Ahead one frame
←	Back one frame
Space	Stop playback

25.5 DISPLAY CONTROLS



REDUCE / ENLARGE DISPLAY

Reduces the *iplay* display size by half, or
Scales-up the *iplay* display size by a factor of two.

ZOOM OUT / IN

Magnifies images by using multiple pixels for each pixel within the image.
Reduces magnification using a single pixel for every several pixels in the image.

CONTRAST / BRIGHTNESS

The two brightness buttons (upper) control the brightness levels. Clicking and holding on these buttons changes the brightness levels. The button remains highlighted as long as the brightness level is changed.

The two contrast buttons (lower) control the contrast of the image.
They operate in the same manner as the brightness buttons.

CHANNEL DISPLAY

regular rgb display

Displays the regular RGB channels of images.

alpha display

Displays the alpha (transparency) channel of images instead of RGB.

z-depth display

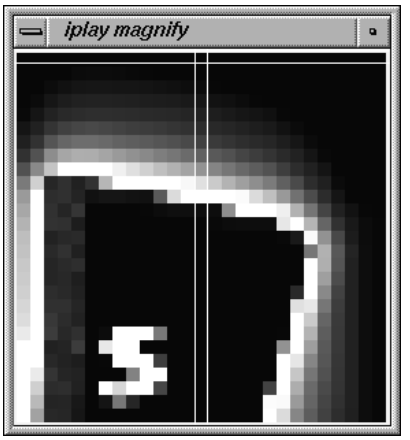
Displays the image's Z-depth values which are mapped to grey levels.
Closer values are whiter than distant values. Invalid depth values appear in red.

red / green / blue only

Displays the only the red, green, or blue channel of an image.

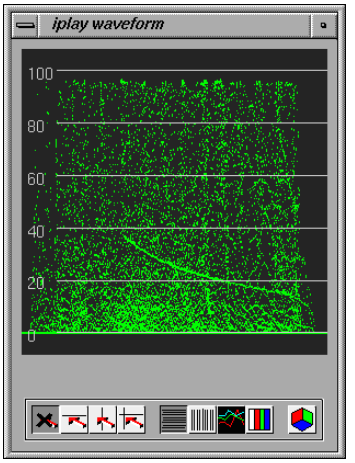
MAGNIFY WINDOW

Opens a window to display a magnified view of wherever the cursor is pointing.



WAVEFORM

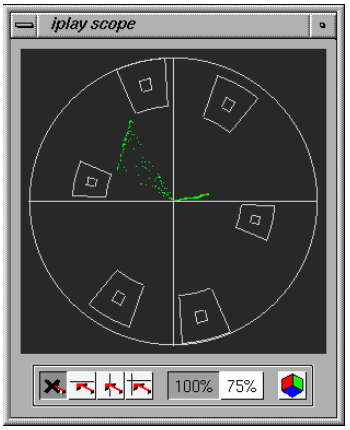
Opens a window which displays the image's waveform.The waveform monitor can be used to analyse the pixel colors within a source image. You can examine the luminance, red, green and blue values of an image by horizontal line, vertical line or point. The display can show values either in a generic color or in the colors of the source image (RGB).



Use this feature to spot potential roll-off problems with your images before they are transferred to video tape or film.

VECTORSCOPE

Opens a vectorscope window. The vectorscope performs a function similar to that of waveform, allowing you to analyse the red, green and blue values of a source image by horizontal line, vertical line or point. It displays the values in green, or in the colors of the source image (RGB).



25.6 COMMAND LINE OPTIONS

Many options can be set from the command line. For a complete listing, type:

```
iplay -
```

IPLAY OPTIONS

```
iplay [options] frameNames
```

Where *options* are any of:

`-a`	Set audio file name.
`-A`	Turn audio on.
`-o`	Set the frame corresponding to audio offset in seconds.
`-w`	Force image width.
`-h`	Force image height.
`-q`	Quick load. Assumes all images are same size
`-Q`	Assume that all frames are different sizes
`-S`	Standard input is a sequence of images in Side Effects Software's RGBA format.

`-M` *balance*	Fit images into available workstation memory. Use *balance* to control how images are reduced. Use *-M 0.0* to fit by skipping frames. Use *-M 1.0* to fit by scaling image size. Use values between 0 and 1 to fit by a mixture of frame skipping and image scaling.
`-m` *balance*	Identical to -M except that images are fit into the amount of free memory currently available.
`-C n`	Set maximum number of frames to keep in memory at any one time.
`-t`	Icon title.
`-s`	Show images without sorting by name.
`-p` *x y*	Window position (in pixels; lower left corner).
`-r fps`	Set frames-per-second. Also sets real-time display on.
`-G`	Stops *iplay* from going into forward mode at start-up.
`-n`	Turn off window border.
`-d`	Turn on double buffering
`-W scale`	Set window scale factor (1 to 4)
`-F on`	Flip images top to bottom
`-F off`	Do not flip images top to bottom
`-c`	Hide all default controls and control panels
`-D tools`	Display toolbar panel
`-D menu`	Display menu strip
`-D frame`	Display frame control panel
`-D range`	Display range control panel
`-D audio`	Display audio control panel
`-D scroll`	Display image scrollbars
`-D inspect`	Display image inspect display
`-D slide`	Display images in slide mode
`-l`	This option enables filtered scaling of images when forcing a width/height. It doesn't take any parameters. This option is also available from the *iplay* startup window (i.e., the window that appears when no files are specified). It is called *Filtered scale*, and is only available when *Force image size* is turned on.

frameNames is one of the two following lines:

```
file1 file2 file3...
-f min max [-i inc] prefix\$Fsuffix
```

where:

`min`	is minimum frame number
`max`	is maximum frame number
`inc`	is frame increment
`$F`	is placeholder for frame number within the file name.

If you load a sequence into *iplay* with -f or using sequential filenames, the frame slider will show the actual start and end frame numbers that were loaded. Some previous versions of *iplay* used to always reset the frame range to start at one.

COMMAND LINE DETAILS

iplay displays one or more images in sequence in a display window. The you can give a list of file names:

```
iplay sky.pic hill.pic wall3.pic
iplay -w 400 -h 300 *.pic
```

The above also scales the images to 400×300 resolution, which occupies four bytes per pixel per image. If every second image *1.pic* to *60.pic* is desired among, say, one hundred images that are in the *Pic* directory, the command should be typed as follows:

```
iplay -f 1 60 -i 2 Pic/\$F.pic
```

If the keyword *stdin* is given, then *iplay* reads the input picture file from its standard input, thus images can be sent through UNIX pipes to *iplay.*

```
ihot sky.pic stdout | iplay stdin
```

The verbose option, *-v* lists the images as they are being read.

The double buffer option divides the bitplanes into two groups, reducing the color range, but avoids the flashing when large images are being flipped through.

Any Image Tool can indirectly use *iplay:* the device *ip* is a simple IRIS window occasionally used to display an image. Using the device *ip*, however, causes *iplay* to be used for reading and displaying an image from the output of an Image Toolkit program. Thus, you can run the command:

```
ihot sky.pic ip
```

which is equivalent to the following command:

```
ihot sky.pic stdout | iplay stdin
```

CACHE SIZE

This sets the maximum number of frames to keep in memory at any point in time.

Usage:

`-C [`*`number of frames`*`]`

25.7 USE OF ENVIRONMENT VARIABLES

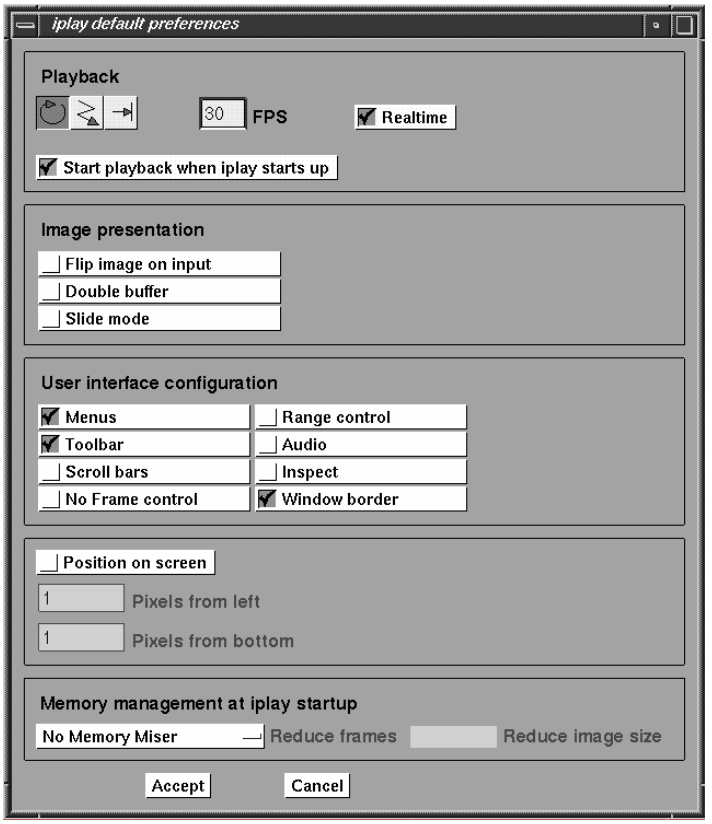
iplay uses the environment variable *$FPS* to determine the default or desired frame rate. Add: *setenv FPS 30* to your *.login* file to default to thirty frames per second.

Note: *setenv* does not work directly from *spy*; you could however do:
*!setenv FPS 30; iplay *.pic* from *spy.*

If all the images in your environment are top-to-bottom (and are displayed upside-down in *iplay*); add: *setenv FLIP* to your *.login* and login again to flip the images every time. To flip the images only once, use the *-F* option to *iplay.*

25.8 IPLAY PREFERENCES

The Preference option in the *File* menu calls up a preferences dialog. You can adjust the *iplay* options here.



25.9 SEE ALSO

- *imdisplay – View Sequential Images* p. 321

26 IPRINT – ASCII VALUES

26.1 SYNOPSIS

```
iprint [-H] [-C] [-F] [-x xoff] [-y yoff]
  [-w width] [-h height] ImageFile
        Displays list of pixel values in RGBA order
        -H      in hexadecimal
        -F      in floating point 0. -> 1.
        -C      in a compact form
```

26.2 DESCRIPTION

iprint displays the R, G, B and Alpha values for an image as a long list of ASCII text. By default, all pixel values are displayed – one pixel per line, with the X and Y coordinates followed by the R, G, B and Alpha values. The range is 0 to 255.

26.3 OPTIONS

`-x` and `-y` — Specify x and y offsets to be applied to the picture, and *-w* and *-h* options specify width and height of a subframe, limiting what is displayed.

`-H` — Indicates that the pixel values should be displayed in hexadecimal rather than the default decimal values.

`-F` — Indicates that the pixel values should be displayed in floating point format in the range 0 to 1 rather than the default decimal values.

`-C` — Indicates that the pixel values should be displayed in a compact form. Instead of each pixel on a separate line, each pixel is separated by two spaces, and each scanline starting a new line. This form is usable by other Houdini tools which utilise arrays.

`-A` — Indicates that the pixel values should be displayed in the form of a Houdini ASCII channel file with no headers and one row per scanline. Any combination of *r*, *g*, *b* and *a* may be given to select which of the red, green, blue or alpha pixel values should be output.

26.4 EXAMPLE

```
!iprint RenderedFrame.pic
    0    1: 255 255 255 255
    1    2: 255 204   0 255 [...638 more lines follow...]
```

27 IQUANTIZE – MINIMIZE COLORS

27.1 SYNOPSIS

```
iquantize [-v] [-G] [-d type] [-n max] [-m method] inimage outimage
```

27.2 DESCRIPTION

This program takes a 32 bit raster image and quantizes the color space to use a maximum given number of colors. The program can also choose the "best" colors for the specified image.

27.3 OPTIONS

`-v`	Verbose output.
`-G`	Force output to be GIF format.
`-d` *type*	Specify dithering method 0 = no dithering 1 = Floyd-Steinberg (default)
`-n` *max*	Specify the maximum number of colors to use (256).
`-m` *method*	Specify color map selection algorithm. 0 = quick method 1 = median cut algorithm (default) or specify a map file

If a map file is specified for the method, then only colors from the mapfile will be used. The mapfile is a list of colors ranging from 0 to 255 in R, G, and B. Each color should be specified on a different line. For example:

```
0 0 0
85 85 85
170 170 170
255 255 255
```

This mapfile would result in a monochrome image using only four colors.

28 IRAMP – CONVERT RAMP

28.1 SYNOPSIS

```
iramp [-H|-V] -w <width> -h <height> <rampfile> <imagefile>
   -H: generate horizontal ramp
   -V: generate vertical ramp (this is the default)
```

28.2 DESCRIPTION

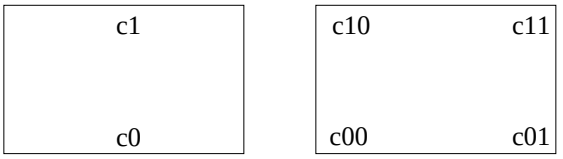
Converts ramp files created by “redit” to image files.

For *mantra*, the minimum Ray Bounce can be specified. If the minimum ray bounce is set to 1, no primary rays would hit the background, the ramp will only be picked up by secondary rays (i.e. reflection/transparent).

Warning: The ramp will refract and reflect in an unexpected manner. Instead of a true reflection or refraction the ray will simply pick up the color of the ramp immediately behind the pixel.

Using the techniques described above you can create a ramp with alpha equal to 1.0. With *mantra* you can also get an alpha less than 1.0 but you must edit the background string directly. The dialog box will not help you:

To the right of the background label is the Options ▷ popup menu. Choose *Edit string*. You will specify the color at the bottom (-c0 option), top (-c1 option) and/or the four corners as shown in the following diagram:



Set the text string using the following syntax:

```
ramp [-c0 r g b [a]] [-c1 r g b [a]] [-c00 r g b [a]]
  [-c01 r g b [a]] [-c10 r g b [a]] [-c11 r g b [a]]
  [-C] [-s scale] [-l level]
```

Then, click on the *Accept* button.

mantra offers an alternate way to handle the background. With the -C option the colors will not be mapped to the screen corners but rather onto a large plane. -s provides a scale which determines how large the plane is.

28.3 OPTIONS

`-H`	Generate horizontal ramp.
`-V`	Generate vertical ramp (this is the default).

29 ISCROLL – SCROLL IMAGE

29.1 SYNOPSIS

```
usage: iscroll [-v] [-X <xcount>] [-Y <ycount>] [sub_frame] <inim-
  age> <outimage>

where:   sub_frame := [-w width] [-h heigth] [-x xoff]
  [-y yoff] | [-d divs frame]

notes:   -X scrolls horizontally.
         -Y scrolls vertically.
```

See also: *icp – Copy / Crop Image* p. 306.

29.2 DESCRIPTION

iscroll scrolls an image, or part of an image, horizontally, vertically or both.

29.3 OPTIONS

`-X`	Scrolls *xcount* pixels horizontally,
`-Y`	Scrolls *ycount* pixels vertically (specifying both will scroll both horizontally and vertically).
`-v`	Causes *iscroll* to be verbose.

30 ISHEAR – SKEW IMAGE

30.1 SYNOPSIS

USAGE1

```
ishear [-s shear] [-y] [-p shear] <inimage> <outimage>
```

-s	Specify the amount of the shear [default 1.0].
-y	Shear in the y direction instead of x.
-p	Previous shear in a rotation sequence.

USAGE2

```
ishear -r <angle> [-n] <in_image> <out_image>
```

-r	Specify angle of rotation [+ or -45 degrees].
-n	No execution, output shear commands.

30.2 DESCRIPTION

Slants (skews) an image in either the X or Y direction. Also rotates the image to a maximum of ±45˚.

31 ISI – CONVERT TO SOFTIMAGE

31.1 SYNOPSIS

```
isi [-F] <inimage> <outimage>
```

31.2 DESCRIPTION

isi will convert images of any Houdini formats to and from Microsoft SoftImage format. One of *inimage* or *outimage* must have a ".si" extension identifying it as a SoftImage image file.

31.3 OPTIONS

`-F` Flip the image in Y.

32 ISIXPACK – MAKE REFLECTION MAP

32.1 SYNOPSIS

```
isixpack [-u usize] [-v vsize] pic1 pic2 pic3 pic4
pic5 pic6 outimage
```

32.2 DESCRIPTION

isixpack generates a reflection map from the six input pictures which have been rendered with viewing angles of 90 degrees, a square resolution, typically 256 × 256, and a square pixel aspect ratio (1.).

pic1 through *pic6* should be images on the six faces of a cube:

pic1 is for front

pic2 is for right

pic3 is for back

pic4 is for left

pic5 is for top

pic6 is for bottom

Default output file resolution is 1024 × 512.

32.3 OPTIONS

`-u` *usize*	Horizontal output resolution.
`-v` *vsize*	Vertical output resolution.
`-a`	Anti-alias the output with 3 × 3 Super sampling.

33 ISTIPPLE – APPLY NOISE

33.1 SYNOPSIS

```
istipple [-v][-i][-D delta][-C r|g|b|rgb][-r min max]
[-c min max][-s seed][res][sub] inimage outimage
```

where:

res is `[-w width][-h height]`

sub is `[-x xoff][-y yoff][-w width][-h height]`
or `[-d divs section]`

33.2 DESCRIPTION

istipple adds random noise to an image.

33.3 OPTIONS

`-D delta`	Set the range of random change to be *delta*. The default is 50.
`-d divs section`	Divide the image into *divs* pieces and choose section number *section*. *divs* is a square: 1, 4, 9, 16 ... Only alter pixels in the requested *section*, numbered from 0 at the top left.
`-C r|g|b|rg|rb|gb|rgb`	Selectively alter the red, green or blue buffer planes. By default, all the color planes are selected.
`-r min max`	Only change pixels that have a color value in the range *min* to *max*., where *min* and *max* are in the range 1-255. By default, *min* is 0 and *max* is 255.
`-c min max`	Clip the new random color value to lie within the range *min* to *max*, where *min* and *max* are in the range 0-255. By default, *min* is 0 and *max* is 255.
`-w width`	Define the width of a window on the image. The default is the full width.
`-h height`	Define the height of a window on the image. The default is the full height.
`-x xoff`	Define the x value of the lower left corner of a window on the image. The default is zero.

`-y` *yoff*	Define the y value of the lower left corner of a window on the image. The default is zero.
`-s` *seed*	Seed the random number generator.
`-v`	Turn on verbose.
`-i`	Enter interactive mode.

33.4 EXAMPLES

To stipple the top-right quarter of an image:

```
istipple -d 4 1 back.pic ip
```

To stipple a small portion near the center of an image:

```
istipple -w 64 -h 60 -x 200 -y 200 marble.pic ip
```

To stipple only the red plane of the file, with noise of 10 out of 255:

```
istipple -C r -d 10 marble.pic ip
```

To stipple the colors between 40 and 50 and clip them if the new random number is less than 30 or greater than 60:

```
istipple -r 40 50 -c 30 60 marble.pic ip
```

34 ITIM – CONVERT TO SONY PSX

34.1 SYNOPSIS

```
itim [-f] [-4] [-8] [-24] [-c colormap.file] [-a] inimage
  outimage [-w width] [-h height]
```

34.2 DESCRIPTION

This program converts Houdini pictures to *.tim* files used by the Sony PSX format. This is used in conjunction with *gpsx* p. 362.

34.3 OPTIONS

`-f`	Generate a fast color map instead of high quality.
`-4`	Generate a 4 bit *.tim* file.
`-8`	Generate a 8 bit *.tim* file.
`-24`	Generate a 24 bit *.tim* file.
`-c`	Use the colormap specified by the file given.
`-w`	Specify the width of the *.tim* file.
`-h`	Specify the height of the *.tim* file.
`-a`	Generate alpha in *.tim* file.

All the options are only valid for creating *.tim* files. By default, a 16 bit *.tim* file is created.

The colormap file has the format:

MapSize

R G B...

Where MapSize should be one of 16 or 256, and the RGB values should be between 0 and 255. If the color map file is not specified, then a default colormap will be used. The color map is only necessary for 4 or 8 bit *.tim* files.

35 IZG – CONVERT Z-DEPTH IMAGES

35.1 SYNOPSIS

```
izg [-s] [-n near] [-f far] <inimage> <outimage>
```

35.2 DESCRIPTION

Converts Houdini Z-depth images to 8-bit greyscale images.

35.3 OPTIONS

`-n`	Set minimum Z value (default 0.01).
`-f`	Set maximum Z value (default 100.0).
`-s`	Operate silently.

36 IZMATTE – COMPOSITE RGBA/Z

36.1 SYNOPSIS

```
izmatte [-v] <rgbpic1> <zpic1> <rgbpic2> <zpic2>...
  <outrgbpic> <outzpic>
```

36.2 DESCRIPTION

izmatte composites pairs of images. Each pair is an RGBA image and a Z image rendered from the same camera of the same scene. To calculate each pixel in the output image, the corresponding pixels of the input Z images are compared. The RGBA of the nearest image is used. *izmatte* uses alpha coverage of the nearer images to determine how much of the farther images to contribute to the output pixel.

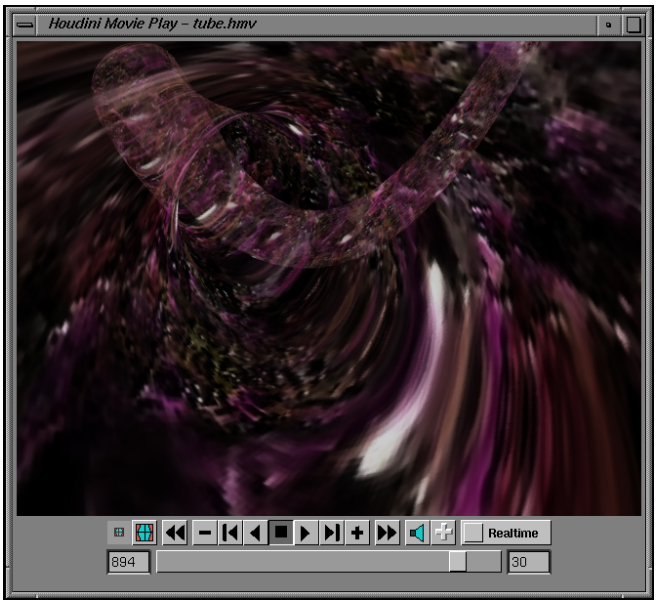
The Z-images are expected to be in the 4-byte floating point format as output by the -Z option of mantra.

izmatte is useful when a scene cannot be rendered fast enough with the available memory. The scenes are rendered separately and the user does not need to care about depth sorting objects for compositing. The animation can be decomposed into any convenient groups of objects.

An inevitable property of the algorithm is that some subtle errors occur when mixing pixels with partial alpha coverage, but it appears barely noticeable.

Both the rgb image and Z image need to be rendered in two calls to mantra. This will be simplified to one call soon, so the polygon data can be loaded only once.

37 HMVPLAY – PLAY HOUDINI MOVIES

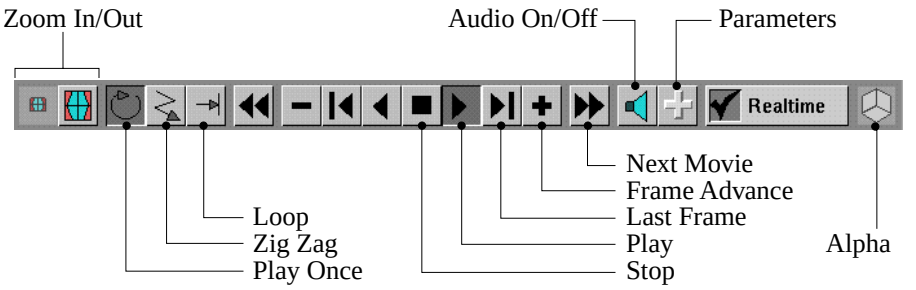


37.1 DESCRIPTION

The *hmvplay* program is a realtime movie player that can playback Houdini Movie (HMV) format files (.hmv suffix). HMV files contain uncompressed sequences of images.

HMV files can be generated from within Houdini by using the Houdini Movie Output OP (found in Output Editor). They can also be read from within Houdini by using the Houdini Movie COP. These files are very efficient for playback of large numbers of animation frames. On an Octane, it can reach speeds of 35 FPS at full video resolution (640×480 pixels) from a disk array running the XFS file system.

37.2 CONTROLS



VCR style controls provide control over movie playback (play, forwards, backwards, step, stop). Controls are also provided for skipping between movie files (next movie file, previous movie file), and for other options.

MULTIPLE MOVIE FILES

If more than one movie file has been supplied, each one will be represented by a small icon. Clicking on an icon makes that movie the current movie for playback.

REALTIME PLAYBACK

A *Realtime* button causes *hmvplay* to playback in realtime (skipping frames or waiting if necessary) or to playback frames as fast as possible (without skipping frames).

FRAME CONTROL

A frame scrollbar allows manual control and scrubbing over specific movie frames. The edit field to the right of the scrollbar should be set to the frames rate required for real-time playback.

ZOOM

The two window icons allow the image to be zoomed in or out.

37.3 KEYBOARD SHORT-CUTS

🖱	Click with 🖱 in the display to get a pop-up menu.
↑ and ↓	The ↑ key starts regular forward playback, and the ↓ key starts backwards playback.
← and →	The → key advances the movie one frame forward; and the ← key moves one frame backwards.
Shift > and Shift <	Skips to the next and previous movie.
Space	Stops movie playback.

37.4 POP-UP MENU

Clicking with 🖱 over the image causes a pop-up menu to be displayed. It provides the following options:

- Show/hide: *Controls, Frame Slider, Movie Icons*
- Update Frame Counter
- Enable Realtime Playback
- Save frame as an Image
- Add to or Modify list of Movie files to play
- Zoom
- Quit *hmvplay.*

There is also an indication of the number of frames per second (FPS) throughput.

37.5 COMMAND LINE OPTIONS

```
hmvplay [-h] [-p x y] [-W scale] movie_file [movie_file ...]
```

`-h`	Displays Help text.
`-p x y`	This command allows the playback window to be positioned to a specific screen position. x and y are the pixel coordinates of the lower left corner.
`-W scale`	This command sets the window Zoom amount.
`-L l, s, z`	Initial Loop type: l = loop continuously s = loop once and stop z = loop in zigzag mode
`-V`	Do not show playback controls
`-S`	Do not show frame slider control
`-r`	Do not startup in realtime mode
`-O`	Send frames to the video out port
`-T imageFile`	When video output is enabled, the imageFile will be displayed for 2 seconds, followed by 2 seconds of black, followed by the movie. This allows titling of edits.
`-c`	To collect multiple movie files and playback as a single movie, use -c before the movie_file. For example: movie_file = *-c file1.hmv file2.hmv ...* *movie_file* = .hmv *sound_file* = .aiff or .aifc

37.6 SEE ALSO

- *Outputs > Houdini Movie (HMV) Output OP* p. 716
- *Formats > Format of Houdini Movie (.hmv) Files* p. 221
- *iplay - View Images* p. 324
- *minfo – Display Movie Info* p. 353

38 HMVRECORD – RECORD HOUDINI MOVIES

38.1 DESCRIPTION (SGI ONLY)

The *hmvrecord* program allows you to send a series of frames in a Houdini movie through the SGI Video Output ports. The video signal can be recorded on external devices ranging from a simple VCR to more complex editing machines.

SYSTEM REQUIREMENTS

HMV Record only is supported on IRIX 6.5 and higher. Video hardware must be available (O2, Octane, Galileo, Indigo, Indy Video). The video daemon must be running.

Playing back video without dropping frames requires very fast disk reads. The most compact format, YUYV NSTC 640 × 480, requires a minimum disk transfer time of 18Mb/s to maintain real time playback.

Since this is a difficult rate to achieve on a normal drive, *hmvrecord* uses frame prequeue buffer to queue as many frames as possible into memory before starting the actual video transfer. This head start allows slower drives to send longer video clips. However, unless the drive can transfer data faster than the minimum video transfer rate, it cannot continue sending uninterrupted frames indefinitely.

38.2 HOW TO USE HMV RECORD

Before running HMV Record, you should create a properly formated HMV movie file. Most video boards support four standard output formats:

- Square NTSC (525) 640 × 480, 640 × 486
- CCIR NTSC (525) 720 × 486
- Square PAL (625) 768 × 576
- CCIR PAL (625) 720 × 576

Use the SGI Video Panel to configure the video output to the format you want (vcp or videopanel). Some video board may have more options.

The HMV movie file you create should be exactly the same resolution as the output resolution. For Square NTSC, both 640x480 and 640x486 resolutions are accepted.

Secondly, the movie file pixel format should be either RGBA or YUV. YUV is the more compact format of the two, requiring 2 bytes per pixel instead of 4. Consequently, twice as much video data can be sent as compared to RGBA. It is recommended that you use this format wherever possible.

All the images in the movie file should be rendered upside down. The SGI Video system sends frames in this format.

OUTPUTTING THE MOVIE

Once you have the sequence rendered, you can now use HMV Record to output the movie.

```
hmvrecord movie.hmv
```

This outputs the full frame range of images using a 20Mb frame prequeue buffer. Once all the frames have been prequeued, it waits for you to press Enter to begin recording. Other options are available:

`-h`	Display the help message.
`-m buffer_size`	The size of the frame prequeue buffer, in Mb. More buffer memory allows more frames to be stored in memory before video output begins.
`-s slate_file`	The name of the slate image file. The slate image is normally text describing the shot. If specified, HMV Record will output 4 seconds of the slate image, followed by 1 second of black, followed by the movie frames. The slate image can be in any image format that Houdini supports, but it must be in the same resolution as the movie and it also must be rendered upside down.
`-f start end`	The frame range of the movie to record. HMV Record will output frames starting at frame 'start' and ending with frame 'end'.
`-l`	Loop the movie until Enter is pressed. Normally, HMV Record will stop outputing frames when the last frame is sent. You can force it to loop back to the first frame and continue by specifying this option. Press Enter to stop outputing frames.
`-w`	Hold the final frame at the end of frame recording and wait until the Enter key is pressed.
`-e`	Hold the last frame at the end of the movie rather than a black frame (the default). A frame is 'held' by sending it continuously.
`-n`	Non-interactive mode. Recording begins automatically, instead of waiting for Enter to be pressed.

38.3 EXAMPLES

```
hmvrecord -m 40 -s info.pic -w -f 10 40 movie.hmv
```

Allocates 40Mb of prequeue buffer. Queues 40Mb worth of frames and then waits for Enter to be pressed. Sends 4 seconds of 'info.pic', 1 second of black, frames 10-40 of movie.hmv, and holds a black frame until Enter is pressed.

```
hmvrecord -n -e movie.hmv
```

Allocates the default 20Mb of prequeue buffer. Queues 20Mb of frames and immediately begins outputing all the frames in movie.hmv. When finished, the last frame is held.

38.4 RETURN MESSAGES

HMV Record returns a list of timing results upon success or failure:

```
Disk read rate is            7.333 Mb/sec
Average frame rate is        8.560 FPS
Frame rate with prequeue is  0.000 FPS
Average frame time is        83.295 ms
31 out of 31 frames successfully sent.
```

- The *Disk read rate* reports how fast data was read from the disk.
- The *Average frame rate* is the maximum rate the disk could acheive using the above disk read rate.
- The *Frame rate with prequeue* is similar to the 'average frame rate', except the prequeued frames pull up the average significantly (prequeued frames are sent at 30 FPS). If result is 0, all the frames were prequeued.
- The *Average frame time* is the average amount of time taken to process one frame. This combines CPU processing with disk read rates. For sustained 30 FPS, this should be less than 33.333 ms.

In this case, the disk is far too slow to support real time playback, but the pre-queue allowed all the frames to be sent in real time.

38.5 SPEED TIPS

We have found the best speed performance depends on:

- Best results with the IRIX XFS file system.
- IRIX is faster than NT.
- Use of multi-disk drives and striping the disks
  (two 10,000 RPM striped disks will give realtime).
- It depends on what you will be viewing the file on.
  The Byte order should be the same as the display.
  On an O2 machine, it is RGB or RGBA.
- If it's going to be played or recorded out its video-out port with *hmvrecord*, then the internal format within the .hmv file should be set to: YUV.
- Be sure to create the .hmv file from the output driver of Houdini.
- Don't use .jpeg – hmv is for uncompressed movies.

39 MCP – MOVIE CONVERSION

39.1 SYNOPSIS

```
mcp [-v] input_movie -o output_movie
```

39.2 DESCRIPTION

mcp converts a sequence of images to a single movie file. For example, you can change a sequence of .pic images into a Quicktime movie. You can then use the *movieplayer* application to play back the resulting movie file. All the output movie files are composed of 32-bit images.

- When converting a sequence of image files to a movie, it is important to note that jpeg compression is slow to write and playback.
- Converting to the Houdini Movie Format (.hmv) offers very efficient for playback of large numbers of uncompressed animation frames.

39.3 OPTIONS

`-v`	Enables verbose output.
`input_movie`	is one of: *filename.mv* (SGI movie) *filename.qt* (Quicktime movie) *filename.hmv* (Houdini movie) *[filename] ...* -f min max [-i inc] xxx$Fxxx -w width -h height -f 1 max stdin
min	is the minimum frame number
max	is maximum frame number
inc	is the frame increment
$F	is a placeholder for the frame number within filenames.
`output_movie`	is: [-w width] [-h height] [-f min max] [-i inc] [-c compression] [-r framerate] output_file ***Note:*** After a movie is read in, its frame numbers are renumbered to the range 1,2,3,... the -f and -i options of the output_movie act on the new range. The default frame rate is 15 frames per second (the maximum).
output_file	is one of: *filename.mv* (SGI movie) *filename.qt* (Quicktime movie) *filename.hmv* (Houdini movie) *xxx$Fxxx* *stdout*

`-c`

The compression of *xxx$Fxxx* is determined by its suffix. The compression, as follows:

For SGI movies, compression is one of:
none : lossless - the default, no compression
rle24 : lossless - compresses areas of the same colour
jpeg : lossy - slow, 20:1 compression
cosmo : special case of jpeg for use with cosmo board.

width & height - must fit within video-res. Width must be a multiple of 16; height must be a multiple of 8.

mvc1 : lossy - Good for video; problems with contrast
mvc2 : lossy - asymmetric version of mvc1

For QuickTime movies, compression is one of:
none : lossless - the default, no compression
jpeg : lossy - slow, 20:1 compression
qtvideo : lossy - (QuickTime only) similar to mvc1
qtanim : lossy - (QuickTime only) similar to rle24

For Houdini movies, compression is one of:
rgba : Red, green, blue, and alpha channels are stored
abgr : Same as rgba but channels are stored in reverse order (for playback on non-MIPS machines)
rgb : Color channels, but no alpha
bgr : Same as rgb but channels are reverse order.
yuv : Image stored in YUV format. Width of image must be an even number of bytes.

The -c option is ignored in other cases.

39.4 EXAMPLES

```
mcp -v -f 1 40 CorkScrew\$F.sgi -o -c rle24 CorkScrew.mv
```

This will convert a sequence of images from 1-40 named *CorkScrew1.pic*... *CorkScrew40.pic* to a lossless compression using rle24, and will call the resulting movie *CorkScrew.mv.* Use this if there is a lot of black in your images, or pixels with constantly the same value.

If you want a lot of compression overall and don't mind some loss, try:

```
mcp -v -f 1 150 -i 1 convertme\$F.pic -o -c jpeg filename.mv
```

39.5 SEE ALSO

- *icp – Copy / Crop Image* p. 306
- *iplay - View Images* p. 324
- SGI documentation on "movieplayer"

40 MINFO – DISPLAY MOVIE INFO

40.1 SYNOPSIS

```
minfo filename [filename ...]
```

40.2 DESCRIPTION

minfo is a command that displays information about movie files.
The supported formats are:

- AVI (.avi)
- MPEG (.mpeg)
- Houdini HMV format (.hmv)
- Quicktime (.qt)
- SGI Movie (.mv)

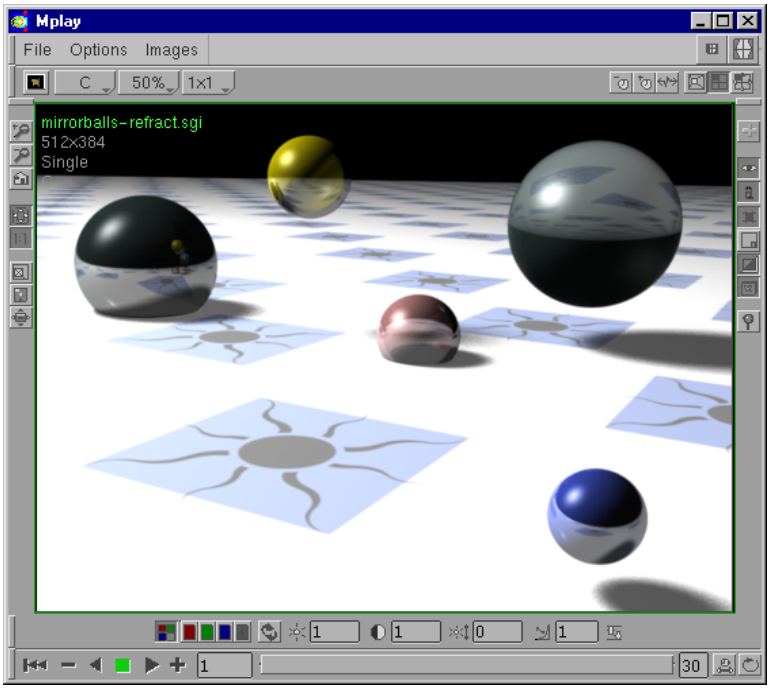
40.3 OPTIONS

For each filename supplied, *minfo* will display the resolution, length and other movie related parameters as appropriate for the type of movie (for example, compression, image pixel format, blocking, etc.).

40.4 SEE ALSO

- *Outputs > Houdini Movie (HMV) Output OP* p. 716
- *Formats > Format of Houdini Movie (.hmv) Files* p. 221
- *iinfo – Image Info* p. 317
- *iplay - View Images* p. 324
- *mcp – Movie Conversion* p. 351

41 MPLAY



mplay replaces iplay, and is as similar as possible to the Houdini COP Viewer.

Please see: *COPs > 2D Viewport* ***p. 471*** **for complete info.**

41.1 COMMANDLINE USAGE

```
mplay [options] image_files...
```

IMAGE OPTIONS

`-g`	Group the command line images into separate sequences, based on base name and extension.
`-u`	Leave the images in the command line unsorted (i.e. in the order they appear).
`-z [level]`	Load images at the specified zoom level (in %): 12, 25, 33, 50, 66, 75 or 100.
`-v`	Flip all images vertically.
`-c`	Load RGB color components only (no alpha).
`-U`	Use an unlimited amount of memory for images.
`-m [mem]`	Limit the memory usage to 'mem' Mb.

`-w [width]`	Load images at width specified (used only by stdin)
`-h [height]`	Load images at height specified (used only by stdin)
`-S [nframes]`	Load nframes images from stdin in RGBA format (used only by stdin).

PLAYBACK OPTIONS

`-f [s] [e] [st]`	Sets the frame range (s,e) and frame step (st).
`-p`	Startup in playback mode.
`-P [mode]`	Sets the playback mode to 'loop', 'once' or 'zigzag'.
`-r [fps]`	Set the frames per second for realtime play.

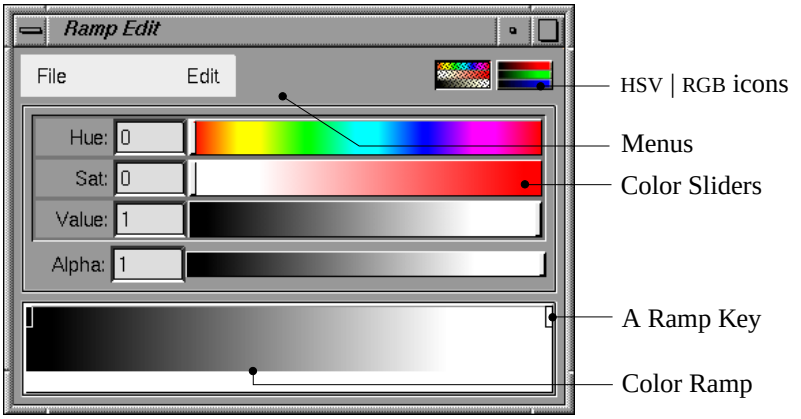
UI OPTIONS

`-b`	Startup in single buffer mode.
`-F`	Full image view; do not show any extra UI.
`-K`	Tell mPlay to listen on a socket for input images. Images rendered to the ip device will be displayed.
`-T`	mPlay will always remain a topmost window (NT only).
`-V [x] [y]`	Sets the viewport layout to 'x' cols by 'y' rows.

AUDIO OPTIONS

`-a [file]`	Set audio filename.
`-A [volume]`	Sets the volume level (default 1).
`-o [fr] [sec]`	Set the animation frame 'fr' corresponding to 'sec' audio seconds.
`-s on|off`	Enable or disable the audio scrub sustain.

42 REDIT – COLOR RAMP EDITOR



42.1 DESCRIPTION

Invoking *redit* by typing the command in a UNIX shell displays a dialog in which you create smooth color ramps with alpha.

For information on how to edit the ramp, see *Color Ramp* p. 185 of the *Interface* section.

Add new keys by clicking within the ramp area.

Delete unwanted keys by selecting the key, and selecting *Delete Key* from the *Edit* menu, or by selecting a key and dragging it outside the ramp area.

Save or *Load* ramps by choosing the option in the *File* menu.

2 Geometry Tools (gTools)

I INTRODUCTION

Before Side Effects Software introduced Houdini, which included the SOP Editor as an integral part of the package, a wide range of geometry processing tools were created to manipulate geometry within a project. These tools – the precursors to SOPs – are still available and, collectively, are called "Geometry Tools".

The Geometry Tools consists of a conglomeration of UNIX programmes designed for the purpose of processing geometry in various file formats. Although the SOP Editor now performs many of these functions, it is sometimes convenient to use them in custom scripts, or as stand alone command line functions within a c-shell or *spy*. For this reason they are still available, and you can still access them via the Unix SOP.

All Geometry Toolkit programs have on-line help. You need only to type the command name followed by a "–" within a c-shell to obtain a summary of the command usage with a brief explanation of the options.

2 POLYGON TOOLS

2.1 INTRODUCTION

There are a variety of stand-alone UNIX tools for creating, converting, modifying, and viewing polygonal data. These include those many of those which are commonly referred to as Geometry Tools. Most of the original Houdini polygon tools are now implemented as SOPs. Some of the basic primitives and operations are still maintained as separate polygon tools.

Most of the programs have a built-in help message which can be displayed by giving the - option to the command. For example:

```
ginfo -
```

2.2 COMMANDS

GCONVERT - CONVERT BETWEEN FILE FORMATS

Polygon files can be stored in three formats: *.poly, .bpoly* and *.d*. *pconvert* converts between these formats. See the on-line manual page.

GDXF - DXF TO HOUDINI

gdxf is a standalone "polygon tool" that translates an AutoCad DXF format file into a *HOUDINI* *.hip* file. The translator fully supports all geometric entities except for the use of SHAPE files. Entities such as *arcs, circles* and *polyarcs* are converted to polygons. The "resolution angle" used for this conversion may be changed using the *-r* option as described below. Due to differences between DOS and UNIX file systems, "external references" are not supported.

If the *-t* option is used, the program will determine the layers in the DXF file and output them. Options:

`-h`	Display program help information (this is it).
`-b`	Output block insert information.
`-k`	Retain the end elements in extrusions of closed polygons.
`-t`	This option will determine the layers that are found in the input DXF file.
`-a`	Angle resolution. This indicates the maximum angle used between vertices in circles and arcs. For instance, an angle resolution of 10 would create a circle with 36 vertices. The default is 10 degrees.
`-v`	An AutoCAD version number.

GEPS – EPS TO HOUDINI

The command *geps* allows Encapsulated Postscript (EPS) files to be converted to *HOUDINI* polygon format. Usage:

```
geps [-l Curve Segment Length]
  [-m Minimum number of Segments]
  [-M Maximum number of Segments]
  [-n Number of Segments]
  [-u]
  [infile.eps outfile.[b]*geo
```

GINFO - DISPLAYS POLYGON FILE STATISTICS

ginfo displays statistics about a group of polygon files. See the on-line manual. This tool is also implemented in SOPs when the you click on the *INFO* button in the SOP tile.

3 GEO FORMAT TOOLS

GCONVERT

This program converts geometry files from one format to another. If a level of detail is specified, all geometry will be converted to polygons at the detail specified

`gconvert [-l lod] infile outfile`

`-l lod`	Level of detail for conversion.

GDXF

usage 1

`gdxf [-k] [-b] [-a angle] infile.dxf outfile.[b]geo`

Converts AutoCAD *.dxf* files into HOUDINI *.[b]geo* files. All existing layers will be converted to groups.

usage 2

`gdxf -t [-i] infile.dxf`

Outputs the layers of the DXF file

usage 3

`gdxf [-v version] infile.[b]geo outfile.dxf`

Converts Houdini *.[b]geo* files to DXF files. All existing groups are converted to layers. Only polygons are handled.

options

`-t`	Display the layer table for the dxf file.
`-i`	Display block insert information from the dxf file.
`-b`	Create a group per dxf block.
`-a`	Angle resolution - the maximum angle used between vertices in circles and arcs. i.e. an angle resolution of 10.0 would create a circle with 36 vertices.
`-k`	Retain the end elements in extrusions of closed polygons.
`-v`	A version number of the AutoCAD release.

GEPS

`geps infile.eps outfile.[b]*geo`

GFONT

```
gfont [-s] [-r] font_file1 [font_file2...]
   -s   Work on the system fonts instead of private fonts
   -i   Interactive installation/removal
   -r   Remove specified fonts (batch mode only)
```

gfont allows you to install, remove, or edit the fonts contained in the *$HFS/houdini/fonts* directory.

GIGES

Converts IGES geometry files – Import Only. It handles trimmed NURBS patches and curves. This is enough to load most files exported from a 3D graphics package.

usage

```
giges [-r] [-c] [-v] infile.{igs,iges} outfile.[b]geo
```

options

`-r`	Reverse direction of trim loops. Sometimes trim loops in IGES files will be specified in a direction opposite to the Houdini convention. This option reverses all of the trim loops so that they will agree with Houdini's convention (i.e. CCW means "keep inside").
`-c`	Read the primitive colours defined in the IGES file.
`-v`	Output verbose status messages to the standard output.

GINVENTOR

Converts Inventor file format to or from *.bgeo format.*
One of either the infile/outfile must be an inventor (.iv) file.

```
ginventor infile outfile -a
```

The *-a* option causes *ginventor* to generate ASCII inventor (.iv) files. Binary inventor files are generated by default.

If an indexed primitive type references an out of bounds index, the following message is displayed on the command line:

```
Error loading (nodetype): Index out of bounds
```

where *(nodetype)* is replaced by the Inventor node type.
The object in which the error occurred is skipped over in the Inventor file.

GLIGHTWAVE

```
glightwave [-s] inFile outFile
```

Where inFile has the extension .lw and is in Lightwave file format. outFile should be a format understood by the geometry library.

Options:

`-s`	Do not convert surface smoothing attribute.

The *-s* option forces the converter to not perform any cusping of polygons regardless of surface settings in the original file.

GMED

This program converts MetaCorp *.med* files into *HOUDINI .bgeo* format. Eccentric metaballs are not supported.

```
gmed infile.med outfile.[b]geo
```

GPSX

This program converts Houdini geometry into Sony PlayStation GPSX geometry files. The *infile* is a Houdini *.poly, .bpoly, .geo,* or *.bgeo* file. The *outfile* will be used as the prefix and will be given *.ply* and *.mat* file extensions.

This command will not do anything unless you are keyed for the Playstation option and you have installed the PlayStation tools. This requires a Sony license from Side Effects. Usage:

```
gpsx [-f] [-s] [-b] [-c transparency] [-t texture (TIM file)]
  [-m material_file] [-R red] [-G green] [-B blue] infile
  outfile
```

`-f`	Fixed color (don't take into account lit sources).
`-s`	Smooth shaded.
`-b`	Backfaces removed.
`-c`	Transparency type: 0 : 50% background (BG) + 50% polygon (PG) 1 : 100% BG + 100% PLY 2 : 100% BG + 50% PLY 3 : 100% BG - 100% PLY
`-m`	Houdini material file.
`-RGB`	Diffuse color for the material.
`-t`	TIM image format.

For textured objects, there can be only one texture map. This map must be a *.tim* file (see the *itim* program to generate these files). This texture can be built by unwrapping a material.

GSDL

Reads Alias v6.0 *.sdl* files.

```
gsdl [-p precision] [-t] [-a] [-n uorig,vorig | -n noshift] [-v][-o
  "patchname... "] infile.sdl
  outfile.{[b]geo,[b]poly,rib}
```

`-p precision`	Uniform level of detail for conversion to polygons.
`-t`	Save with texture coordinates.
`-a`	Save name and shader as attributes.
`-n uorig,vorig`	Normalize the domain and shift its origin.
`-n noshift`	Normalize the domain but do not shift its origin.
`-v`	Verbose conversion or just list contents.
`-o patchnames`	Convert objects (i.e. patches) by name.

GWAVEFRONT

Converts an *.obj* file to the other format or convert the other format to an *.obj* file.

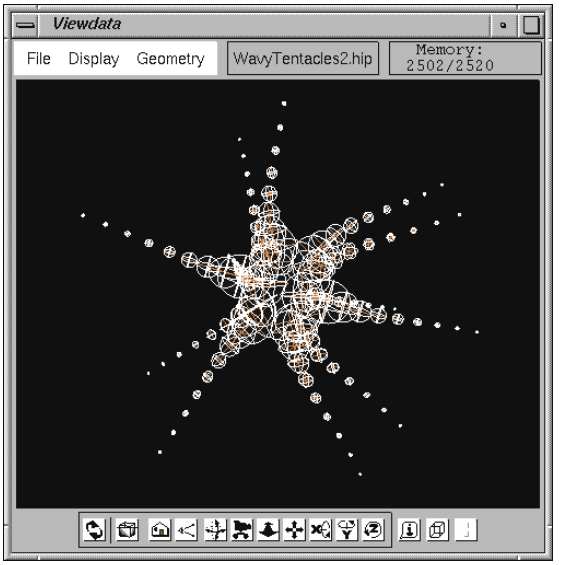
```
gwavefront [-l lod] [-u] [-t] inFile outFile
```

`inFile outFile`	One of *inFile/outFile* has the extension *.obj* and is in Wavefront file format, the other file should be a format understood by the geometry library.
`-l lod`	Uniform level of detail for conversion to polygons. The default is 1.0
`-t`	Force texture attribute to be attached to points.
`-u`	Force points to be unique. This helps with multiple texture coordinates per point.

The *-t* option is basically equivalent to the *-u* option that uniquing in that it will only be done if texture coordinates exist. This ensures that the attribute will be correctly mapped

Note: When converting to .poly or .bpoly format, the -t option should be used. As well, normals are always attached per point, which means that the -u option should be used if there are different normals per position vertex in the .obj.

4 GPLAY – VIEW 3D GEOMETRY



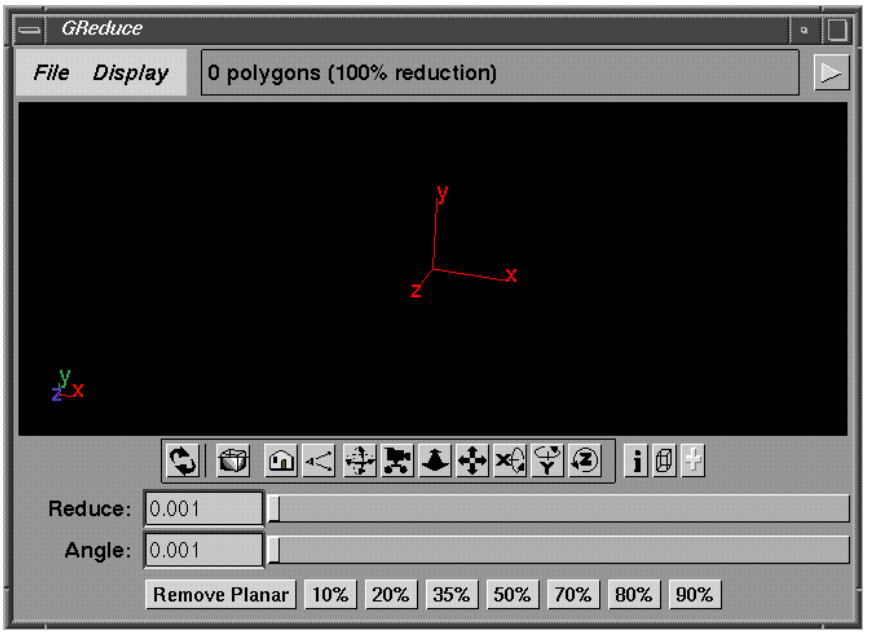
4.1 DESCRIPTION

The *gplay* application runs on IRIX 5.2 and later. It allows you to view 3D geometry files in a Viewport similar to the one in Houdini. You can rotate, tumble, zoom, and even render the viewed geometry.

You can also display the Viewport Options dialog, which allows you to optionally display hulls, point and vertex numbers, and normals, etc.

For details on the *gplay* controls, see the *Interface* section.

5 GREDUCE – POLYGON REDUCTION TOOL



5.1 DESCRIPTION

greduce reduces the polygon count of an object while trying to keep the general shape of the object. *greduce* can also be run in batch mode, where no graphical interface appears and the conversion is performed from the command line only.

The *Open Triangulated* and *Open 4 sided* File menu options allows you to open the geometry with triangles or quadrilaterals as the polygon type respectively.

Note: It is recommended that you use the percent reduction buttons for speedier operations with this tool. Files with many polygons may take a long time to reduce using the reduction factor and angle parameters.

SYNTAX – INTERACTIVE MODE

```
greduce filename
```

EXAMPLE

```
greduce teapot.bgeo
```

This will start the tools interactive mode, where you can reduce the object's polygon count using the sliders and controls provided.

There are two controls for determining the polygon reduction. These are represented as sliders at the bottom of the window. The reduction factor is the primary control. However, this should be used in conjunction with a small increase in the angle factor (e.g. 5).

To simply remove planar polygons, set the reduction factor to a small number and the angle factor to a small number (i.e. 0.001, 0.001). Beneath the sliders are a set of buttons which can be used to reduce the polygon count by differing degrees. For example, remove 10% of the polygons. These controls are approximate and are usually good to plus or minus 5%. In some cases, it will not be possible for the program to remove a large number of polygons. In these cases, hitting 90% reduction will result in a smaller percentage actually removed.

SYNTAX – BATCH MODE

```
greduce [-v] [-t|-q] -r reduction_factor -a angle infile
  outfile
```

This reduces the geometry using the reduction factor and angle factor specified.

```
  greduce [-v] [-t|-q] -p percentage [-a angle] infile outfile
```

This reduces the polygon count by the percentage specified. The angle does not have to be specified, though it can make a small difference in the resulting geometry.

EXAMPLE

```
  greduce -v -p 40 -a 10 teapot.bgeo reduced.bgeo
```

This example will run the polygon reduction program in "batch" mode. With the -v specified, the resulting statistics will be displayed. Sample output from the above command might look like:

```
Used reduce = 0.151 and angle = 10 Reduced to 170 polygons. 40.9722%
  reduction
```

EXAMPLE

```
  greduce -v -r 0.08 -a 10 teapot.bgeo reduced.bgeo
```

The above command will reduce the polygons in the teapot file using the parameters specified. Again, the verbose option (-v) will cause some statistics to be displayed when the program is finished. This output will look something like:

```
Reduced to 193 polygons. 32.9861 % reduction
```

options

`-v`	Verbose option for batch mode versions
`-t|-q`	Triangulate or make quadrilaterals out of the input file
`-r`	Reduction factor (corresponds to upper slider in interactive version)
`-a`	Angle of reduction (corresponds to lower slider in interactive version)
`-p`	The amount, expressed as a percentage, of the original geometry to reduce.

3 Channel Tools (chTools)

1 CHANNEL TOOLS

These tools allow you to perform Channel commands normally performed in the CHOP Editor from a regular shell window:

1.1 CHCHAN

Copy channel collection to/from action channel format.

```
chchan infile outfile
```

Where *infile* is a raw channel (.chan/.bchan) file and *outfile* is a channel file (.chn), or *infile* is a channel file and *outfile* is a raw channel file.

1.2 CHCP

Copy channel collection file to another format.

```
chcp infile outfile
```

Where *infile* and *outfile* are one of: .chan/.bchan, .chn/.bchn, or .clip .
For a summary of channel file types, see *Channel File Formats* p. 199 in the *Formats* section.

1.3 CHINFO

Provides information on a channel collection.

```
chinfo channelFile
```

Where *channelFile* are one of the formats listed in *Channel File Formats*.

1.4 CLAUDIO

Copy CHOP data (clip) to/from audio format.

```
claudio [-v] [-f format_int] infile outfile
```

Where:

`-v` Specifies verbose mode.

`-f` Specifies a format integer when saving to an audio format when the suffix is unknown or updated since this version.

Exactly one of *infile/outfile* must be an audio file currently supported by the Silicon Graphics Audio File Library. Known suffixes include:

```
.aiff(-f 2)  .wav(-f 4)  .aifc(-f 1)
.snd(-f 3)   .au(-f 3)    .sf(-f 5)
```

The other must be a .bclip or .clip file.

1.5 CLCHAN

Copy CHOP data (clip) to/from action channel format.

```
clchan [-r rate] [-m] infile [outfile]
```

`-r` Specifies a sample rate when writing a clip.

`-m` Writes each channel to a separate file.

Exactly one of infile or outfile is a raw channel (.chan/.bchan) and the other is a clip file (.clip/.bclip).

1.6 CLCHN

Copy CHOP data (clip) to/from channel collection.

```
clchn [-r rate] infile outfile
```

`-r` Specifies a sample rate when writing a clip.

Exactly one of *infile* or *outfile* is a channel collection (.chn/.bchn) and the other is a clip file (.clip/.bclip).

1.7 CLCP

Copy CHOP data to another format.

```
clcp infile outfile
```

Where *channelFile* are one of the formats listed in *Channel File Formats*.

1.8 CLINFO

Provides information on the specified CHOP format file.

```
clinfo clipFile.clip
```

1.9 FILE FORMATS USED

There are three file formats being used here:

- `chan (.chan, .bchan)`
  The .chan is a simple ASCII format, arranged as rows of numbers, one row per frame, one column per channel. The .bchan is the old-style *PRISMS* raw binary format. These files can be read into the Channel Editor.

  `chn (.chn, .bchn)`
  Houdini-format files for a group of jive channels. (bezier(), ease(), etc.).

- `clip (.clip, .bclip)`
  Format used by CHOP nodes. (Raw values like .chan, but with more information attached).

MORE INFORMATION ON CHANNEL FORMATS

For more Information on CHOP Channel Formats, see Formats > *Channel File Formats* p. 199.

4 hCommands

1 HCOMMANDS

hCommands are programs that can be run from a shell and perform some of the functions of Houdini's component parts and without a GUI. They are an efficient means of performing specific tasks without having to load Houdini in its entirety.

1.1 HSCRIPT

SYNTAX

```
hscript [-h] [-q] [-u] [-v] [file.hip ...]
```

Provides a Houdini shell. This is te non-graphical interface to a motion database (i.e. .hip file). Any number of motion files or .cmd scripts may be specified on the commandline. Multiple motion files are merged together (see the mread command).

OPTIONS

-h	Displays this help message.
`-q`	Quiet: supresses the display of version information.
`-u`	Enables use of the Ultimatte COP.
`-v`	Specifies verbose handling of renders.

hscript runs as a stand-alone application that allows access to all of Houdini's Scripting commands that are usually available in Houdini's Textport. Please see: *Ref > Scripting* for the complete scripting language reference.

- To get a list of valid keywords, type *help* at the prompt.
- To get syntax help on any command, type *help commandName* at the prompt.
- Type *quit* to end a session and return to shell.

1.2 HRENDER

hrender renders a composite network, contained in a .hip file, using a command line interface. *hrender* is typically run from *spy*, *csh*, *sh* or other UNIX shell. It can also be part of a batch script file (e.g. csh or sh) for execution at a later time or as part of a sequence of rendering steps.

SYNTAX

Single frame: `hrender [options] driver|cop file.hip [imagefile]`

Frame range: `hrender -e [options] driver|cop file.hip`

COMMAND LINE PARAMETERS

`driver	cop`	-c comp/net_name -c net_name -c net_name/cop_name -d output_driver (without /out prefix)
`options`	-w pixels Output width -h pixels Output height -f frame Single frame -b fraction Image processing fraction (0.01 to 1.0) -o output Output name specification
`with -e`	-f start end Frame range start and end -i increment Frame increment

NOTES

- For output name use $F to specify frame number (e.g. -o $F.pic).
- If only one of width (-w) or height (-h) is specified, aspect ratio will be maintained based upon aspect ratio of Output OP.

1.3 HSOP

hsop allows you to edit specific objects and their component Surface operations in a .hip file.

SYNTAX

`hsop [-o object_name] [-t object_type] [filename ...]`

COMMAND LINE OPTIONS

`-o`	Specify the name of the object to edit (no default)
`-t`	If the specified object does not exist create one of this type.

2 DSPARSE – DIALOG SCRIPT PARSE

2.1 SYNOPSIS

```
dsparse dialog_script_name [initial dialog string]
```

`dialog_script_name` the name of a dialog script to test.

2.2 DESCRIPTION

This application parses a dialog script and displays the dialog without having to run Houdini. This can be used to quickly test dialog scripts without the overhead of re-loading Houdini.

This program generates .ui files which are cached for future runs of the application. When changing parameters, it's a good idea to remove these .ui files to make sure that the dialogs are up to date. These *.ui* files are built in */usr/tmp/Dialogs/HDSParse*, so for testing, it's usually a good idea to run:

```
rm -rf /usr/tmp/Dialogs/HDSParse
dsparse dialog_script_name
```