Book design and layout by Todd Brunhoff

Software design and implementation by Todd Brunhoff, Al Tabayoyon, Chris Jacobson and most recently, Russ Gorby. Many thanks to Russ, Chris and Al for creativeness, perseverance and long, long hours. Many thanks to Don for endless (but useful) criticisms.

15262 NW Greenbrier Parkway
Beaverton, OR 97006
(503) 531-5707
support@nvr.com

These changes should be added to your `.cshrc`. And if you already have a value for XFILESEARCHPATH you should append it instead of replacing it as we have shown here.

One very important part of our software is the color fidelity. The best results can only be achieved if you install data for the X Color Management System, a standard in the X Consortium. For the Sun 19” Trinitron monitors, all that is necessary is to type:

```
% xcmsdb $NVRHOME/Xdata/xcms/Trini19.dcc
```

There are several device color calibration files (DCC) in this directory. You may have to try others to get the best results.

Installing the demonstration key

If we made arrangements with you ahead of time, then your license keys are in `$NVRHOME/adm/keys/productkeys`. If not, you can start working now by installing the demo key (but its only good for a short time). In either case, simply type:

```
% installkeys
```

Now you are ready to run `recorder`, `player` and `compressor`, whose manual pages you will find in Appendix D. We suggest you read the tutorial on **How To Play** first.

When you have time, be sure to read the **Installation Guide** in Appendix C to find out about our licensing server and other fine points of installation. If you have any trouble, please call us at (503) 531-5707 or send us email at `support@nvr.com`.

Note for SUNOS 4.1.x users!

When you start the `player`, `recorder` or `compressor`, you may get lots of message that start with

```
Warning: translation table syntax error...
```

This is because your system’s key symbol database file, named `XKeysymDB` is either

- not where you said it is (with the $XKEYSYMDB variable)
- not where the library expects it to be (only a problem with SunOS 4.1.x)

If you have a value for $XKEYSYMDB, make sure it points to a valid keysymdb. If you *do not* have a value for $XKEYSYMDB and you are running SunOS 4.1.x, you can use the database we provide with the software. To use it, simply type

```
setenv XKEYSYMDB $NVRHOME/Xdata/XKeysymDB
```

ments about our software. You can reach us at (503)645-6345 or with email to `support@nvr.com`.

The First Step: Analog to Digital

As with any change in representation from analog to digital, a high-speed sampler chip called an analog to digital converter, or DAC, converts the video signal into digital samples. Let's make a few assumptions about the original analog video signal[1]:

- it has 30 frames per second (fps)
- it has 640 active picture lines
- it has enough resolution to support 8 bit samples for each of red, green and blue
- we want "square" pixels, implying a picture that is 640x480

Video Bandwidth

Some simple arithmetic suggests that the digital version of the video will require about 27.5 megabytes per second[2] (MB/sec). That's a lot.

Now that (we think) we have digital video available on the workstation, we must be able to store it, retrieve it and display it. Consider each of the subsystems involved: memory, display (or frame buffer) and disk.

The Disk

Most inexpensive disks available on workstations can read and write at about 1 MB/sec. Newer, larger disks improve that up to about 2-4 MB/sec, and more expensive disk subsystems can be configured up well past 27.5 MB/sec. However, these transfer rates assume a raw disk at peak transfer rates. Saddle it with a file system, and throughput is greatly reduced. Most UNIX variants today use some form of the Berkeley Unix file system designed by Kirk McKusick, and it is better than most. Even so, you can expect the throughput to drop 50%-75% below raw disk performance[3]. A few new file system designs consider multimedia demands and recover much of the lost performance.

Disk Bandwidth

The bottom line is that disk throughput of 250 kilobytes per second (KB/sec) is a safe bet on most computers, and 2 MB/sec is the ceiling for those of us not in the broadcast industry or a national research laboratory.

The Frame Buffer

On most computers, reads and writes to and from the raw frame buffer is a little to a lot slower than main memory. Few programmers write their own file systems, and as you might expect, few programmers write their own display system (or window system). Using the X

1. This ignores many aspects, such as whether this is NTSC or PAL, frames vs. fields, and how many active lines the signal really has.
2. 640x480x(3 bytes per sample)x(30fps) = 27,648,000 bytes per second.
3. The tutorial on How to Go Faster discusses using a raw disk.

How can you reduce the video bandwidth?

There are two ways to reduce the bandwidth requirement with hardware:

- Slow down the video
- Use some form of compression, such as JPEG or MPEG

The first option can be done by using a frame-accurate source[1] that can "step" through each frame; for example a video disc player (about $1,000) or a video disc recorder (about $15,000). We don't think this is interesting because chances are the video material you want to use is not available on video disc, and a video disc recorder may not fit your budget.

With hardware compression...

Compression seems to be the ticket, although only JPEG hardware compression is available today at any reasonable price. For now, this works. Generally, the results of 11:1 compression with JPEG is considered very good. This brings our example input down from 27.5MB/sec to about 2.5 MB/sec. Fortunately, the color representation[2] used by the Parallax Xvideo board gives another 1.5:1 compression down to 1.7 MB/sec[3]. This is a rate that will work with a raw disk.

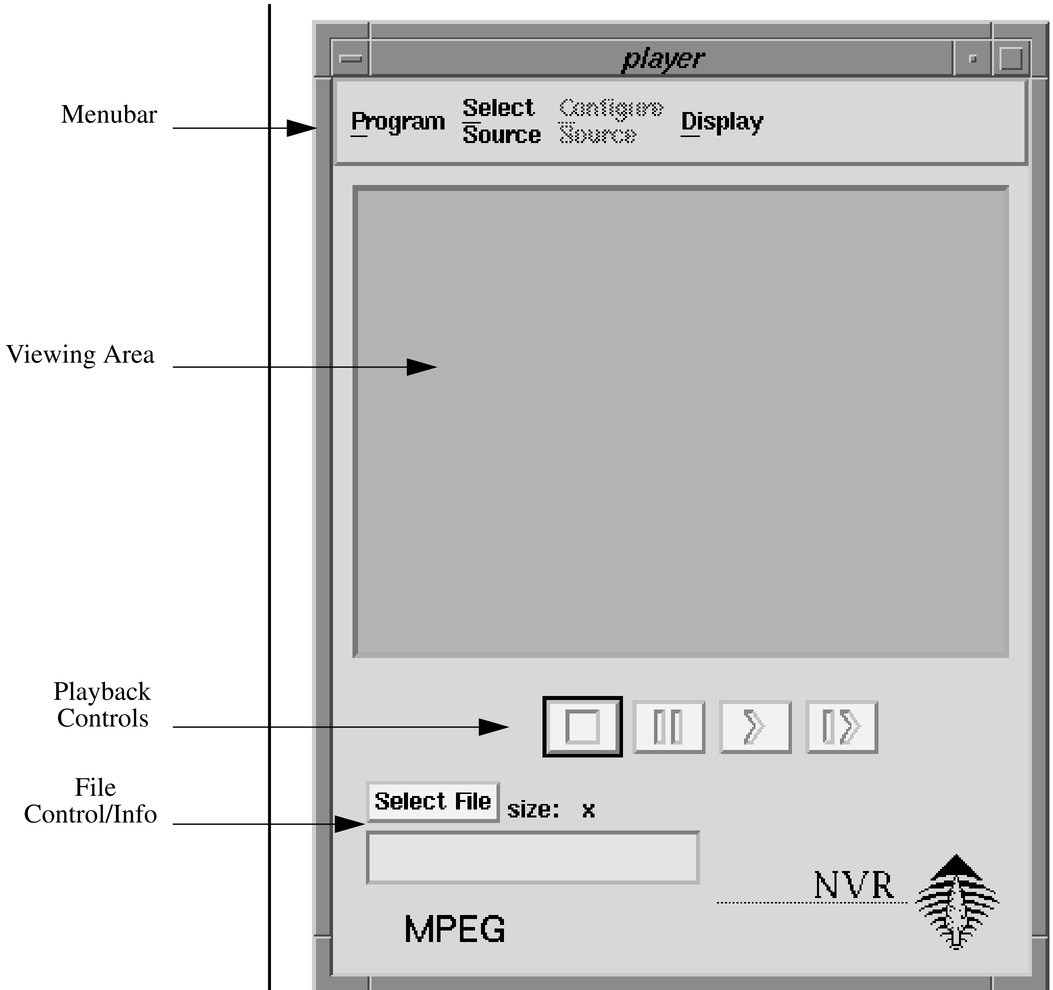
Further compression with JPEG means lesser quality, which may actually be acceptable for your application. We have found that true in many cases. However, MPEG can achieve better compression at no less quality because it allows the use of motion compensation. Since most pictures in a sequence share more in common than not, motion estimation can be used to remove those redundant parts.

Without hardware compression...

Without hardware assistance there isn't time to do real-time compression while the video flies by, so a simple reduction must take place at once. The only possibility is to sub-sample the video picture as it is digitized. It must be reduced to the smallest size, and the table below shows the kind of picture size and frame rate possible assuming the 1.7 MB/sec we achieved in the JPEG compression example.

---

1. Frame-accurate tape players don't count because even though they can step through frames, the video picture in pause mode is not stable enough to digitize.
2. The picture is compressed not as red, green, blue, but as one luminance channel and 2 chrominance channels. Because the human eye perceives less color information, the chrominance channels are half-sized.
3. Parallax software does not yet support this bandwidth. NVR is working with them to produce our own software that will support it.



As you would expect, the player comes up with a blank “screen”. So the first thing we’ll need to do is select an MPEG file for viewing.

File Selection

File selection is performed with the Motif file selection dialog. To bring up the file selection dialog for MPEG files, press and release the **Select File** button. Now you can peruse through the file system and select your desired file, however, for the purpose of this tutorial select the file “hockey1.sif.variablepn”.

Playback Controls

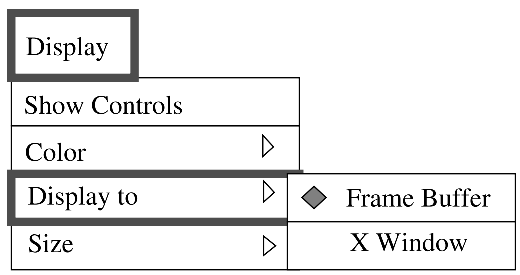
By now, the first frame of the MPEG sequences should be displayed. To play, simply press the **Play** button

Rendering Method

If you haven't already noticed, the MPEG Player does not require special hardware since MPEG decompression is totally implemented in software.The MPEG Player renders frames in one of two methods:

- Direct to Frame Buffer
  This is the fastest of the two modes since it "draws" directly into the frame buffer. There is a drawback to this method however — occluding windows will be drawn over. Therefore, when viewing in this mode, keep the player's "screen" free from occluding windows. This mode is the default rendering method.
- Via the X Protocol
  Since this method uses the X Protocol, the X server will not draw over occluding windows. The drawback to this method, however, is a lack in performance.

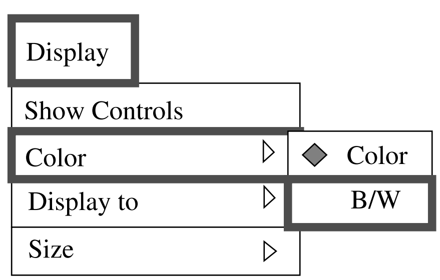
To change to either method, pulldown the Display Menu, press the **Display to** button, then select your method of choice by pressing the appropriate button from the resulting cascade menu.



Black & White or Color

The MPEG Player can playback MPEG sequences in either color or black & white. The suggested mode for playing back sequences with audio is black & white. Let's do one right now.

Pulldown the Display Menu, press the **Color** button, then press the **B/W** button as shown below.



Next, bring up the file selection dialog, then select the file "calendar .sif.variable". Finally, play the MPEG sequence by pressing the

Now lets concentrate on the audio related sliders and toggles, but first bring up the file selection dialog select the file "`black_magic_woman.Level2_32kHz_10to1.ampeg`". This file contains an MPEG audio stream compressed with the level 2 MPEG audio compression technique that has resulted in a 10 to 1 data reduction.

Now, play the audio sequence by pressing the **Play** button. While in play, experiment with the **Volume** and **Audio Quality** sliders; and finally with the Audio Output toggles if you have connected external speakers to the SPARCstation audio port. Moving the Audio Quality slider to the left reduces audio quality by not decompressing the higher frequency audio signals. This feature enables you to continue listening to the MPEG audio sequences while running moderately compute intensive processes.

Switching to Analog Video

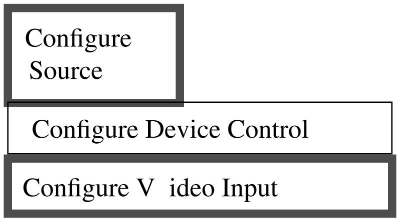
The Player will know if your display hardware has been configured to play Analog Video. If so, the **Analog Device** button from the Select Source Menu will be enabled. If disabled, the button will be grayed out and not selectable. At this time, bring up the Select Source Menu by pressing the **Select Source** button from the menubar. Then, if enabled, press the **Analog Device** button. If not enabled, you'll need to install the required hardware as described in **Device Connections** before proceeding with this tutorial.

Analog Video Player

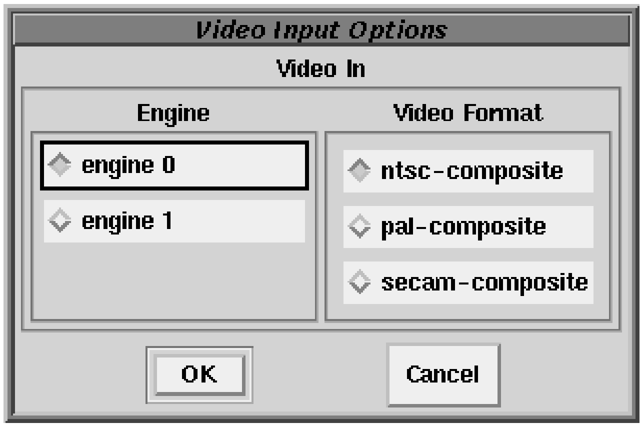
At this point, we assume that you have connected a video source device to one of the Parallax XVideo input ports. If not, you can connect the source device now or skip this section of the tutorial to view JPEG sequences. If you the latter, proceed to the "Switching to JPEG"

Configuring Video Input

No picture is being displayed? You may need to tell the Player which of the frame buffer inputs your video has been connected to. To do this, go to the menu bar at the top of the Player application, pull down the **Configure Source** menu item, and select the **Configure Video Input** item as shown below.



The dialog box containing the necessary selections should appear as shown below.

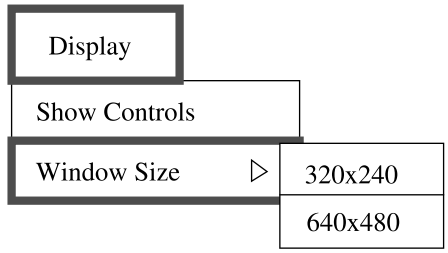


The first item is the engine. For the Parallax XVideo, the engines are analogous to the video input connectors (**engine 0** is Parallax input #1 and **engine 1** is Parallax input #2). If the engine that you have chosen is capable of accepting multiple types of video signals, you will need to make this selection as well. The Parallax XVideo, for example, provides "multi-headed" cables capable of accepting several types of video signals into a single engine. The choices you have made will not take effect until you press the **OK** button. If you have correctly specified your video input configuration, you should now see a video image from your source device.

Display Controls

OK, you have a video image displaying in the video display. Once again, make sure that your video is active. Now pull down the **Dis-**

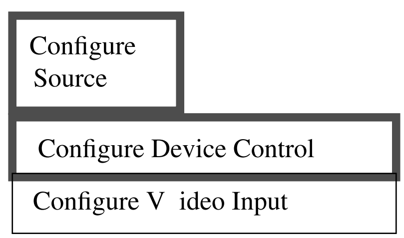
640x480 size windows. To change the size, press on the **Display** button in the menubar. This will “pulldown” the Display Menu. Now press on the **Window Size** button, then press the button from the resulting cascade menu for the size desired.



The no-controls mode can be quite cumbersome. Some video devices now support computer control of their behavior. Some, such as the Pioneer LD-V4400 laser disk player, allow direct control via an RS-232 connection. A host of others with LANC (formerly Ctrl-L) ports may be controlled using a Sony CI-1000 ViSCA Vbox between the workstation and the device. Assuming you’ve connected your source device for remote control (refer to the Device Connections section for more information), let’s configure our Analog Video Player to remotely control that source device.

Configuring for Remote Control

To engage remote device control, pull down the **Configure Source** menu bar item and press the **Configure Device Control** button.

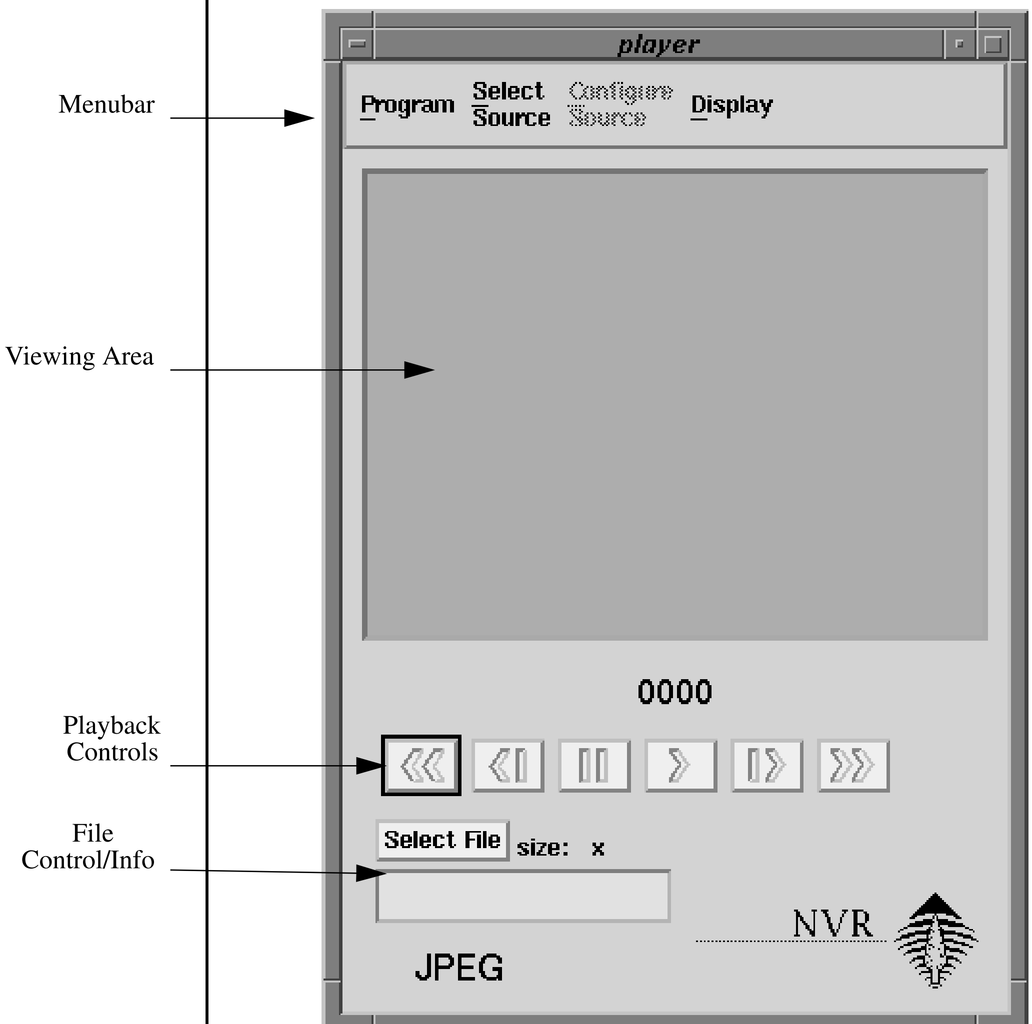


This will open a dialog box allowing you to make the necessary selections.

Switching to JPEG

The Player will also know if your display hardware has been configured to play JPEG sequences. If so, the **JPEG File** button in the Select Source Menu will be enabled. If disabled, the button will be grayed out and not selectable. At this time, bring up the Select Source Menu by pressing the **Select Source** button from the menubar. Then, if enabled, press the **JPEG File** button. If not enabled, you'll need to install the required hardware as described in section **Device Connections** before proceeding with this tutorial.

JPEG Player



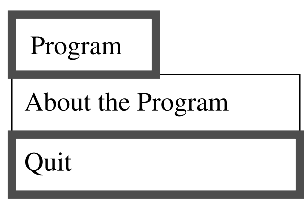
The JPEG Player comes up with a blank "screen" as with the MPEG Player. Likewise, the first thing we'll need to do is select a JPEG file for viewing.

The Audio/Video Controls panel as shown below should appear.

Unlike our MPEG and Analog Video Players, there are no video controls — only audio controls. Go ahead manipulate these controls while playing a JPEG sequence.

Quit

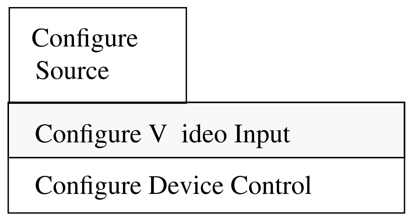
When you're done experimenting with all the players, end your session by pressing the Quit button from the Program pulldown menu as shown below.



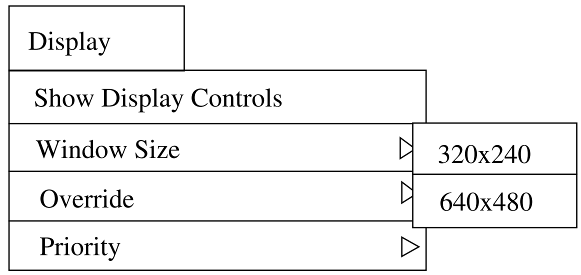
The Video Input

By default the recorder application comes up in its simplest configuration. In this mode the recorder has a video display pane acting as a passive video monitor. To activate the video display press the play (>) button. To freeze the picture in the display, press the pause (||) button.

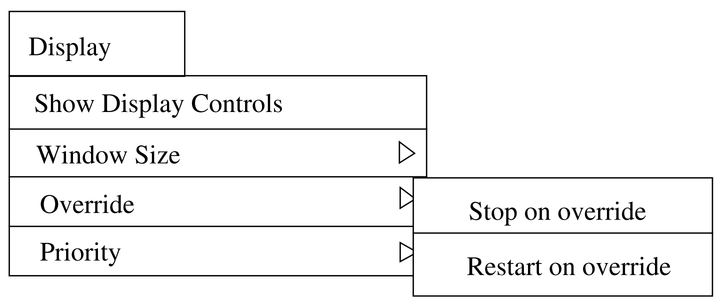
If you have the video display active and your video source device is properly connected to the Parallax, you should be able to manually control your source device to display an image in the video display.

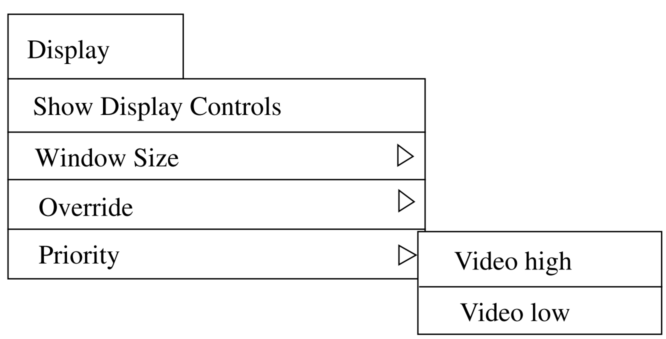


No picture is being displayed? You may need to tell the recorder which of the frame buffer inputs your video has been connected to. To do this, go to the menu bar at the top of the recorder application, pull down the **Configure Source** menu item, and select the **Configure Video Input** item. A dialog box will open containing the necessary selections. The first item is the engine. For the Parallax XVideo, the engines are analogous to the video input connectors (**engine 0** is Parallax input #1 and **engine 1** is Parallax input #2). If the engine that you have chosen is capable of accepting multiple types of video signals, you will need to make this selection as well. The Parallax XVideo, for example, provides "multi-headed" cables capable of accepting several types of video signals into a single engine. The choices you have made will not take effect until you press the **OK** button. If you have correctly specified your video input configuration, you should now see a video image from your source device.

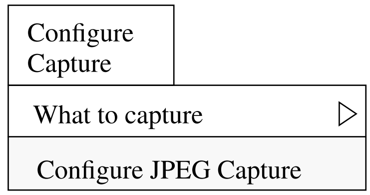


You may also adjust the size of the video display by again pulling down the **Display** item from the menu bar, then moving down to the **Window Size** item and sliding right to *cascade* to the menu of available sizes. Note that when you record, you capture the video image in the size of the video display. A larger window size will result in a larger data set for the same length of video material.

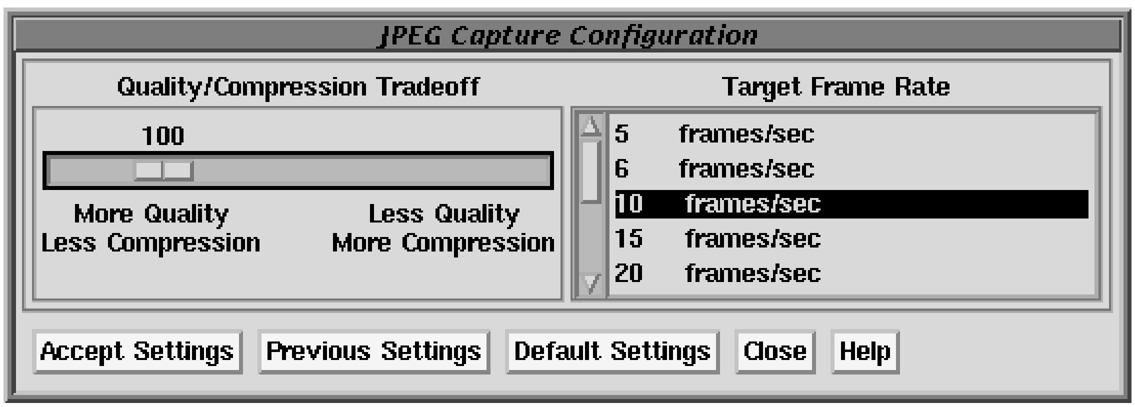




The remaining items in the **Display** menu are **Override** and **Priority**. With some configurations, multiple video displays can compete for scarce video resources. When another process starts that is in need



When recording with the NVR recorder application, real-time JPEG compression is performed. When you select the **Configure JPEG Capture** menu item, a dialog box will be opened with controls which allow you to tailor the compression process.



With the JPEG compression used, there is a trade-off between the image quality of the output and the amount of data used to represent the images. The more data used to represent the images, the higher potential quality . A control is included that allows you to specify a trade-off along a spectrum of possible trade-offs.

You may also choose from among a set of frame rates. The frame rate specifies the number of frames to be digitized and written to the destination file during each second of the recording activity . If the JPEG compression hardware is unable to keep up with the selected frame rate, some frames may be dropped, that is not written to the file, along the way . At the end of the recording operation, you will be informed as to how many frames were recorded and how many were dropped.

You may append video to existing JPEG movie files only if the compression parameters that you have chosen match those of the existing file.

Two areas of the interface are changed when device control is successfully engaged. The section of buttons below the video display region is expanded from two buttons to as many as seven. The buttons now correspond to the transport controls of the media device (play, pause, fast forward, ...). These allow you to remotely control the position of the video media in the source device. You will also notice that as the media is in motion, a counter just above the transport controls is updated reflecting the position of the media.

Precision Recording

The second region of change is an additional section of buttons and displays to the right of the video display region. This section is the segment record section. The segment record section allows you to specify a starting position on the media, the *in point*, and a stopping position, the *out point*. You can then preview the specified *segment*, adjusting the in and out points as desired, and ask to have the segment of video between the two points recorded. This capability gives the user as much accuracy as the source device offers in specifying what to record. This region should look familiar to users of video editing equipment.

The first row of buttons in the segment record section is the in

This chapter will describe all three processes of producing an MPEG stream: the video, the audio and the multiplexed system stream. The bulk of the chapter is devoted to video compression because there are so many ways to compress a good sequence; that begins below. The methods for compressing audio and creating a system stream begin on page 45.

The basic setup for compressing a movie file to MPEG video can be done in a series of very simple steps:

- acquire a movie file; currently this must be either a Parallax JMOVIE file or a nvridx file.
- specify the movie file to compress (the **Source**)
- specify the desired filtering parameters
- specify the desired compression parameters
- specify an output file in which to write the MPEG output (the **Destination**)
- press the **Start** button

A Word About Movie Files...

If you want to use Parallax JMOVIE files as your movie input files, you can use the NVR Recorder or Parallax's RTV software. Otherwise you can create a movie file by constructing an nvridx file. This particular file type, described on page 79, can describe a group of individual graphic files such as TIFF, JPEG or SGI rgb files, as a single movie.

Setting up the MPEG Video Compression

Once you have a movie file to compress, press the **Source** button (see the section titled **Movie Files** following this one). A Motif file selection dialog[1] will pop up allowing you to navigate through the file system to locate a file. When you have successfully selected a movie file, some information about the file will be displayed next to the **Source** button on the main application window. The path name will be displayed in a Motif text widget below the Source button.

Specifying a destination file is equally simple. Press the **Destination** button to get another Motif file selection dialog. The MPEG video compression process will extract the video information from the movie file and compress it into an MPEG video stream. To the destination file name that you specify will be appended the extension *.vmpeg* for the video MPEG file and *.ampeg* for the audio MPEG file.

Pressing the Start button will begin the compression process. As frames are compressed, information is displayed into the Compression Statistics pane to keep you informed as to the progress of the process. You may temporarily stop the compression activity by pressing the

1. The conventions of the Motif user interface and its associated widgets are well documented elsewhere and will not be repeated here.

There are four sections to the filtering:

- Deinterlacing
- Clipping
- Resizing
- Smoothing

Deinterlacing

If the frames in your input material are comprised of two fields, it is said to be “interlaced”. MPEG 1 does not allow interlaced frames, so one field must be dropped, and the NVR Compressor always does this. However, with some extra effort you can ensure the highest quality. The two pictures below illustrate the kind of problem. Note the low angle sharp line of the gutter. When a field is simply dropped, as in the left hand picture, a jagged line is evident. With our deinterlace filter, the continuity is restored. When choosing the width you can use the following guidlines:

- The lower the angle, the wider the filter that must be used to restore the continuity.
- The wider the filter, the more time it takes to process each picture.

Without deinterlace filtering

With width 13 deinterlace filter

Resizing

This step allows you to specify the final size compressed by the MPEG compression engine. The size must be a multiple of 8 in the height and a multiple of 16 in the width. There are three methods you can choose for doing the resizing; each has its benefit and its cost:

- Nearest Neighbor. This is the fastest method because it requires almost no computation. Unfortunately, the results can be pretty disappointing. Use this only if you are trying to produce a quick sample sequence.
- Bi-linear. This method takes a weighted sample of a 2x2 square for each pixel, so it works very well for changing sizes small amounts. Consider the clipping example above: if the clipped picture size is 344x240 and you want to get back to 352x240, the bi-linear method is perfect. However, if you are resampling the width or the height by a factor of two or more, then the final picture will be fuzzy.
- 7-tap FIR. This is a high-brow method you can use to impress your friends (FIR stands for finite impulse response). It is the best method for changing picture size by a factor of two. You can't go wrong with this selection although it takes a little longer than the bi-linear.

If you simply leave the setting on 7-tap FIR, the compressor will always do the right thing. That is, the 7-tap FIR will be used whereever factor-of-two size changes are necessary, and bi-linear will be used everywhere else.

Smoothing

Rarely, you will need to smooth your input data. This is only necessary in two situation:

- The input material is very noisy. Using the smoothing filter will reduce some of the noise making the result more pleasing and easier for the MPEG compression algorithm to work[1].
- You want to achieve a greater compression ratio. By smoothing good quality input material, the picture quality goes down, but it is easier for the MPEG compression algorithm to do a good job. This is useful if you want to produce a very low bitrate stream.

When you have finished making your filtering selections, press the **Accept** button to make them ready for use in the compression process. The dialog will be automatically closed if the values have been accepted. Should you decide to abandon your current selections prior

1. Compression algorithms depend on the fact that the material is predictable in some way. Noise, by its nature, is random. Hence, the noisier your input picture, the poorer the job the compressor will do. As the saying goes "garbage in, garbage out".

process known as *Motion Estimation*. The NVR compressor application provides a number of algorithms for performing this important task. The methods differ primarily in terms of how much effort they expend to determine the components of the one video image that will be different in the next frame of a video sequence (pictures in a video sequence often have large portions remaining constant from frame to frame, the exceptions occurring when picture elements are in motion). While one algorithm may expend more effort than another to detect motion, it is not guaranteed to be successful; greater effort therefore guarantees only increased potential for higher quality in the resulting MPEG sequence. There are three algorithms available:

- Exhaustive. This is the best method because it does a complete search for every motion vector.
- Multi-processor Exhaustive. This is the same method is above, except two or more processors are used to do the motion estimation (this is only available on some platforms). Be sure to select the number of processors to use.
- Logarithmic. This is a fast method because it only looks a 1/16 of the motion vectors, then tries to zero in on the best. This will work very well if the input sequence has little or no motion.

The **Exhaustive** search methods can be further tailored in the amount of effort they expend by adjusting the **Search Radius**. A larger search radius requires more effort in the search and, again, guarantees only an increased potential for a higher quality compressed image. The **Logarithmic** algorithm ignores the search radius adjustment.

You may specify a **Target Bit Rate** for the MPEG file. This specifies the number of bits used in the file to describe one second of the video sequence. The more bits of data used, the greater potential exists for a higher quality image but less compression is achieved. When a JPEG movie source file is selected, a default target bit rate is calculated as well as the range of bit rates from which you may select.

If you are compressing a file to be used with a fixed bandwidth delivery/playback system, you may wish to select a **Fixed Rate**. This assures that the number of bits in the MPEG file meet the target on a second-by-second basis. The default **Variable Rate** algorithm may vary but, over the course of the sequence, will average out to the target rate. Both types will result in a similarly sized file for a given source.

And lastly, you may specify the group of pictures layout for the MPEG file. You may specify the number of P-frames in each group and the number of B-frames for each P-frame. The compression in MPEG files comes largely from the reduced size of the P and B-frames as comtiplexer is not available; only a command line version is available. Please see the `smpegcomp` man page at the end of this manual.

- Increase the throughput to the disk

Without JPEG compression, image size would be fixed at 320x240 or 76,800 pixels. With a quality factor of 100, JPEG compression reduces this to between 2,000 and 15,000 bytes. A normal, well-lit picture with moderate detail yields about 11,000 bytes. A flat blue picture compresses to about 2,000 bytes and a bright, highly detailed picture compresses to about 15,000 bytes.

How to minimize compressed data

You can expect slightly better throughput if your original video is very dark. We found that most of the scenes from the recent Dick Tracy film released on video disc by Touchstone Home Video compressed to about 7,000 bytes, while video shot in our offices with the Cannon A1 Digital camcorder compressed to about 11,000 bytes.

All of these numbers can be reduced by increasing the quality factor (remember, the higher the number, the lower the quality).

Since JPEG recording to memory quickly overflows system memory limitations, this is not as interesting as recording to disk. But you can increase your disk throughput in two ways:

- Use a faster disk
- Bypass the file system and use a raw disk

Throughput to a file in a UNIX file system is limited to between one-third and one-half of the raw disk throughput, so the faster your disk is, the faster the UNIX file throughput can be.

Recording to a raw disk partition

NVR software supports recording to, and playback from a raw disk partition. Mind you, taking this route is not for the feint of heart, but requires a knowledgable UNIX systems expert to make sure you don't destroy your system. In fact, the risk of software damage to your entire system is so high, that we insist that you implement these suggestions at your own risk!

That being said, find your local system administrator and show him the procedure below, he can help you decide if you have access to someone with enough expertise. We give only a brief description here, although an expert will understand what must be done.

First, select a disk drive on the system with the Parallax board. It should be the fastest. Now select a partition that you will reduce in size; the free space on this partition must be larger than the size of your intended raw partition, otherwise, you won't be able to restore all your files. You should allocate at least 50 megabytes to the raw partition.

Second, run full dumps on all your system disks. This will help

Parallax Xvideo board, you must use OpenWindows 3.0.

Our software user interfaces were build under Motif 1.2, and they have been tested running under the Motif window manager , and under the OpenLook window manager . In fact, even if you are using some other ICCCM-compliant[1] window manager , our software should work just fine.

SGI System Requirements

Silicon Graphics Hardware

NVR software has been tested on SGI Indy and Indigo systems. The compression engines (vmpegcomp, ampegcomp) have been tested on Challenge series machines.

Display

All standard SGI workstation displays will work.

Operating System And Window System Software

NVR software tested under IRIX 5.2.

Our software user interfaces were build under Motif 1.2, and they have been tested running under the IRIX window manager

1. Inter-Client Communication Convention. This is a document published by the X Consortium which details the proper behavior of clients and window managers. Both the Motif and OpenLook window manager are ICCC-compliant.

The attenuator that we use has stereo-mini and separate 2-RCA inputs and outputs to RCA or mono-mini connectors. Replacing 5k resistors for the 10k in the diagram you will get a -34 db attenuation which may be more appropriate with some devices[1].

Video Input Connections

The ability to digitize a video signal and write it to disk in real-time requires the use of the Parallax XVideo product. This product provides two video input connectors and several cables to use in connecting video source devices. The specific connections are well documented in the Parallax literature and will not be repeated here.

Be sure to note the port/engine of the Parallax board that you are feeding video into and the specific type of connection to this port. The video input you provide may be a “composite” signal, such as NTSC or PAL, or may be a “component” signal, such as RGB or YUV. The Parallax video cables have connectors for each of the video signal types supported. You will need this information to tell the NVR recorder application where to look for your video information.

Device Control Connections

The task of the NVR recorder application is to capture analog video and audio into digital form. The video and audio source material can come from a wide variety of video source devices. With any source device that is capable of putting out a video signal compatible with the frame buffer’s inputs you can digitize the incoming signal. The mechanism for doing simply requires hitting a record button to start recording whatever video signal comes into the selected video input and hitting a stop button to complete the operation. This is useful, simple, and allows a broad range of source devices to provide the video input.

A growing number of video devices are externally controllable. The recorder leverages off of this capability to provide the ability to record a precisely specified segment of video from a video disk or tape (to the extent of the individual device’s capabilities).

For some such devices, a direct RS-232 connection is possible[2]. The specific RS-232 cable connections required for these devices are well documented in the manuals for the workstation products and in the manuals for the controllable video devices. This information will not be repeated here. It is important to note, however, that the NVR

1. If you are in need of an attenuator but don’t have access to resources for the construction of such a device, contact us.
2. The Pioneer LD-V4400 Laser Video Disk is such a device and is supported by NVR software. Other Pioneer Video Disk devices may also work with NVR software but have not been tested.

`usr/local/bin/nvr`; we'll refer to that directory from now on as $NVRHOME. The distribution tape is set up with the same hierarchy as shown above.

Each of the subsections below will talk about details for some subset of this directory tree. We'll cover:

- Basic installation
- Environment variables
- X applications defaults
- License installation and maintenance
- Known problems and solutions

Basic Installation

First select an appropriate file system for the NVR Development kit; it must have at least 33 megabytes of free space. This is how much space is taken by the programs, the compressed sequences, the data files, etc.

Find a tape drive appropriate for the distribution media you received, typically an 8mm tape, a 4mm DAT tape or a standard quarter -inch cartridge (QIC). The tape drive must be connected to the machine making the installation[1].

Installing the software

Most of the commands we use in the installation are shown preceded by a `%`, which is usually the prompt issued by the UNIX™ C-shell. In a few examples, we show commands preceded by a `$`, the usual Bourne shell prompt. We assume that you are familiar with the C-shell (and provide instructions about the Bourne shell for completeness). Insert the tape into the tape drive and enter the following commands:

```
% mkdir nvr
% cd nvr
% tar xvf /dev/rst0
```

The /dev/rst0 is a possible device name for the Sun workstation. On SGI, it may be /dev/tape. In any case, you should contact your system administrator if you are unsure.

This extraction from tape shouldn't take too long. The `v` flag in the tar command will show you file by file as the product is extracted (so you may omit it if you like). You should now have a directory tree in your current working directory like that illustrated earlier.

1. Because we use tar to create the tapes, it is also possible to install this distribution when the tape drive is not local, but we would prefer that you call us if you don't know how to do this. Your system administrator probably knows.

```
$ XFILESEARCHPATH=$NVRHOME/Xdata/app-defaults/%N%S
$ export XFILESEARCHPATH
```

You may already have a value for `$XFILESEARCHPATH`, especially for OpenW indow application defaults. In this case, simply prepend the value with the path shown:

```
% setenv XFILESEARCHPATH \
$NVRHOME/Xdata/app-defaults/%N%S
% setenv XFILESEARCHPATH \
$XFILESEARCHPATH\:/usr/openwin/lib/%T/%N%S
```

Note that the : is preceded by a back-slash (\) to prevent the shell from interpreting it. This is very important.

If you would like more information on application defaults environment variables, send us email at `support@nvr.com`.

You may need to set `XKEYSYMDB`, too!

You may also need to set the XKEYSYMDB environment variable. Read the note at the end of the Quick Install on page 4.

X Application Defaults

The `player`, `recorder` and `compressor` must each have access to the proper application defaults file to function correctly, because many of the aspects of the program layout are specified there. If you do not choose to modify your `XFILESEARCHPATH` variable, then you must place the three application defaults files whereever you keep these files for other X applications. If neither of these are done, the programs will remind you that they are not installed properly.

License Installation and Maintenance

Distributed license management is provided by the Élan License Manager. Your shipping invoice will tell you what kind of license was enabled for your product. If you provided your host id and internet number with your purchase order, the software will already be configured for the licensing you purchased. Otherwise, it will be set up for a single-use demonstration license with an expiration date. This subsection will tell you how to manage the licensing.

What if I don't have the right software license?

If your product was not shipped with a full license, please call us at (503) 690-2355 and we will give you a key over the phone and you can have your software running in a few minutes. Even easier, send electronic mail to support@nvr.com and we will reply with the key file. This will save typing in a very long license key.

If this software will only be used on a single system, then all NVR software knows how to look for the license manager and to start it up if it is not running. You can also start it by hand or arrange for the license host to start up the license manager at system boot time.

This is apparently a bug with the X11R5 Parallax server.

MPEG display on an X server configured to use a 24-bit default visual seems to have the wrong colormap installed.

The Parallax server seems to install the colormap correctly when you select a new sequence, but when you move out, a different one gets installed and never gets reinstalled. The xprop program shows that the correct properties are on the application window. We suggest that you bring up the Parallax server in the default mode: an 8-bit pseudo color visual as default.

OpenWindows X server patch level requirement

The patch level should be at least Patch #3000-104. Otherwise server crashes may be experienced.

NAME

ampegcomp - NVR’s MPEG audio compressor/encoder

SYNOPSIS

```
ampegcomp options
```

OPTIONS

-in *filename*	Specifies the input file. Default is standard input (stdin).
-out *filename*	Specifies the output file. Default is standard output (stdout).
-layer *n*	Specifies the MPEG layer to be used for encoding. Currently of the three layers, only layers 1 and 2 are supported. Default is Layer 2.
-mode *name*	Where name is one of *stereo*, *dual_channel*, *joint_stereo*. Specifies the 1) the mode (mono or otherwise) used to capture the raw audio data contained in the input file, and 2) the mode to encode the data (i.e., when data is captured in two channels you’ll need to specify if the data should be treated as two disjoint channels (“dual channel”) or in stereo. If the latter, you have the choice of encoding the data in regular “stereo” or “joint stereo” mode. Joint stereo encoding exploits stereophonic irrelevance or stereophonic redundancy. Default mode is mono.
-ext_mode *n*	Where *n* is 0, 1 2 or 3. When mode is joint_stereo, this option indicates which subbands are encoded as intensity_stereo: 0: subbands 4-31 in intensity_stereo 1: subbands 8-31 in intensity_stereo 2: subbands 12-31 in intensity_stereo 3: subbands 16-31 in intensity_stereo Default is 0.
-sampling *s*	Where *s* is one of 32000, 44100, or 48000. Specifies the sampling frequency. Default is 32000 kHz. Valid MPEG audio sampling frequencies 48000, 44100, and 32000 Hz.
-bitrate *b*	Specifies the target bitrate in kilobits. Valid bitrates depend on the layer selected. Valid bit rates are Layer 1: 32, 64, 96, 128, 160, 192, 224, 256, 288, 320, 352, 384, 4416, 448 Layer 2: 32, 48, 56, 64, 80, 96, 112, 128, 160, 192, 224, 256, 320, 384 The default for Layer 1 is 448 kb/sec.; the default for Layer 2 is 384 kb/sec.
-psy *m*	Where *m* is 1 or 2. Specifies the psychoacoustic model to use. Default is 2.
-emphasis *e*	Where e is *5* or *c*. Specifies the type of de-emphasis to use: *5* for 50/15, *c* for ccitt j.17. The default is none.
-copyright	Indicates that the input material is NOT copyrighted.
+copyright	Indicates that the input material is IS copyrighted.
-original	Indicates that the input material is NOT the original.
+original	Indicates that the input material is IS the original.

DESCRIPTION

audiocomp is an MPEG audio encoder/compressor that takes a variety of audio files and writes out an MPEG audio data stream to the specified output file. Input files may be any of

NAME

Compressor - User interface application for compressing MPEG sequences

SYNOPSIS

`compressor [options]`

OPTIONS

-destFile <file name>

Specifies a file relative to the directory in the -destPath option as the output file for the MPEG compression.

-destPath <path>

Specifies a directory . To be used in conjunction with the -destFile option.

-sourceFile <file name>

Specifies a JPEG movie file relative to the directory in the -sourcePath option.

-sourcePath <path>

Specifies a directory . To be used in conjunction with the -sourceFile option.

DESCRIPTION

This application is a user interface to control the compression of JPEG movie files into MPEG files. The application provides the ability to specify source (JPEG) and destination(MPEG) files, to tailor the compression process to create an MPEG file that meets your needs, and to monitor the progress of the compression process as it is underway .

RESOURCES

NVRCompressor*destinationPath : <path>

Specifies a default directory into which to write MPEG files.

NVRCompressor*motionEstimation : <index>

Specifies an index into a table of motion estimation methods.
0 --> Exhaustive 1/2 Pixel
1 --> Exhaustive Full Pixel
2 --> Jain and Jain
If you do not specify this option, 2 (Jain and Jain) will be the default.

NVRCompressor*numberBperP : <b>

Specifies the number of B pictures per P picture in a group of pictures. If you do not specify this option, the default will be 3. The range is {0..6}.

NVRCompressor*numberPperI : <p>

Specifies the number of P pictures in a group of pictures. If you do not specify this option the default will be 4. The range is {0..6}.

NVRCompressor*searchRadius : <radius>

Specifies the search radius used by the Exhaustive motion estimation methods. If this option is not specified, the default will be 16. The range is {8..64}.

SEE ALSO

recorder , player

NAME

fileinfo - provide summary information about video and audio files

SYNOPSIS

`fileinfo <file> ...`

DESCRIPTION

This is a very simple program that tries to pick up where the standard unix command, `file`, and its generalized database fall short. In particular, many movie file formats are too complex to be parsed by the file command.

This program will try to print out relevant video and audio information. For example, given a system file, the command might produce the output shown below:

```
% fileinfo horse.sys
Copyright 1992-1994 North Valley Research.
All Rights Reserved. Product release 2.0.2.
horse.sys    : mpeg system (video), 352x240, 25.000fps, 1.186Mbits/sec
             : mpeg system, mpeg system (audio), 16bit/44.1khz/Stereo
```

BUGS

Doesn' t fully understand non-movie file formats, such a TIFF.

Doesn' t understand audio bitrates.

SEE ALSO

`$NVRHOME/etc/magic`

COPYRIGHT

This low-level application is a basic image sequence processor used to transform a sequence from one form into another. Input may be a set of individual pictures, such as TIFF or JPEG, or a movie sequence, such as raw rgb frames or Parallax's JMOVIE.

Imageop requires `io` (read) as the first operation and `io` (write) as the last. Any number of other operations may appear in between, except `io`. During initialization, each operator forms its own copy of the image from specified attributes or inherits the attributes from the previous pipeline operation. For example, with the command

```
imageop \
io read=somefile size=320x240 \
io write=anotherfile
```

the first operator specifies the size, and the second operator inherits the size from the first.

ARGUMENT AND VALUES

`decimate` This specifies that the filter is a vertical decimation filter (`decimate=vertical`), a horizontal decimation filter (`decimate=horizontal`), or both (`decimate`).

`inclip, outclip`
This specifies the clipping applied to the input and output image. For example, `inclip=320x240+4+4` resamples only the specified pixels. Outclip limits the pixels touched in the output image (this isn't interesting just now because, there is no way to specify a value for those untouched pixels.

`interpolate` This specifies that the filter is a vertical interpolation filter (`interpolate=vertical`) or a horizontal interpolation filter (`interpolate=horizontal`).

`method` (`deinterlace` operator)
Currently there are four values for the deinterlace method: `copy`, meaning to create the second field by making a copy of the first, field 1 (this typically produces alot of jagged edges on sharp lines); `copyfield2`, meaning to use the second field (this introduces motion artifacts); `black`, meaning to fill every other line with black (usefull for testing); `spatial`, meaning to predict spatially the value of the missing lines (this has wonderful results with `width=13`).

`method` (`resample` operator)
Currently there are two values for the resample method: `nearest`, meaning nearest-neighbor; and `bilinear`.

`prefix, suffix, ndigits, start, stop`
For implicit file lists, the values of these arguments are used to construct file names. Essentially:

```
for (i = start; i <= stop; i++)
   sprintf(file, "%s%*d%s",
      prefix, ndigits, i, suffix);
```

`read, write` Whenever a file is read or written, its name may imply compression or decompression. Three suffixes are recognized: .Z, .z and .gz. The .Z suffix implies

rawrgb	Raw RGB data; each pixel is three bytes: R followed by G followed by B. (must be specified)
rawycbcr	Raw Y/Cb/Cr data; entire Y component, followed by Cr component, followed by Cb component. (must be specified for read operations)
rawycrcb	Raw Y/Cr/Cb data; same as rawycbcr, except that the Cb component comes after the Cr component. (must be specified for read operations)
sgirgb	SGI rgb data. This is not a standard file type, but a convenient one. The source code for a library that read and writes these files is typically found on any SGI system in `~4Dgifts/iristools`.
sun	SUN raster
tiff	Tagged Image File Format. When writing tiff files, LZW compression is always used. When reading, the tiff libraries insist on being able to perform seeks on the file. This can affect the performace if the files being read are not internally compressed with LZW but are externally compressed with unix compressor or gzip. In these cases, imageop must make a temporary disk copy of each uncompressed image. We suggest tiff files only use internal LZW

BUGS

This (and vmpegcomp) are extremely difficult to use because of the complexity of the command line. Use at your own risk.

SEE ALSO

vmpegcomp

ding). Defaults to 0.3 seconds for normal operation. This option is ignored for compact disc operation.

headertime= *time*
: Will for a system header to be emitted at least every “time” seconds. This rate also controls the frequency of sending buffer sizes in each elementary stream. Defaults to 0.5 seconds for normal operation. This option is ignored for compact disc operation.

maxbehind= *sec*
: Maximum number of seconds between input and output (defaults to 2 maximum sized packets). This option is ignored for compact disc operation.

audiolock=0/1 Sets the audio lock flag in system header (default=1)

videolock=0/1 Sets the video lock flag in system header (default=1)

format= *type* Sets the stream encoder mode. The type may be any of `mpeg`, `cdi`, videocd, `vcdseg`. Without this flag the default is standard mpeg (ISO/IEC 11172-1). The type `cdi` means CI-Interactive or “Green Book”; `videocd` means Video CD 2.0 or “White Book”); `vcdseg` mean Video CD 2.0 segment-play. For any of the types `cdr`, `videocd` or `vcdseg`, the options described under GLOBAL COMPACT DISC OPTIONS may be used.

GLOBAL COMPACT DISC OPTIONS

filenbr= *file* Specifies CD subheader file number. If not specified, then only the system multiplexed stream with 2324 byte video “packs” and 2304 byte audio “packs”. If specified, the output will be the “packs” described above inside of 2336 byte CD sectors. Essentially the sectors begin with a subheader, followed by data, followed by EDC code. More detail can be found the the appropriate standards.

channbr= *chan* Specifies CD subheader channel number. If not specified, a default is chosen appropriate to the standard. This is only meaningful if `filenbr=` is specified.

pause= *nsect* Specifies the number of sectors in pause region of CD track [def=150]. This is only meaningful if `filenbr=` is specified.

fmsect= *nsect* Specifies the number of sectors in front margin of CD track [def=15]. This is only meaningful if `filenbr=` is specified.

rmsect= *nsect* Specifies the number of sectors in rear margin of CD track [def=15]. This is only meaningful if `filenbr=` is specified.

stills= *file* Specifies a file containing a list of still frames and timecodes. The file format is described under STILL PICTURE FILE FORMAT

tcfps= *fps* This specifies the frames/second for still picture timecode parsing. If unspecified, it defaults to 30fps.

endtime= *time* Specifies the last packet time for the finished, multiplexed stream. When producing sectors for a compact disk, the sectors must continue to be delivered until the last presentation time stamp has been delivered. This parameter allows you to specify an end time beyond the last presentation time stamp.

'starttime'. Note that this option is ignored if the elementary streams are specified with `videos=` or `audios=` because the sequence files contain their own start times.

id=*hex* Specify stream ID (lower 8 bits of packet startcode). The valid ranges are 0xC0-0xDF for audio, 0xE0-0xEF for video, and 0xF0-0xFF for data. Default allocation is the lowest unused ID for this stream kind. This option is ignored for CD, because the standards specify the correct value for each of the stream types.

DESCRIPTION

Combine elementary video, audio and data stream into a single MPEG 1 system stream. All of the elementary streams are assumed to be constant bit rate (CBR) and all I/O is from files [for now].

Timestamps (PTS and optionally DTS) are pre-appended to each presentation unit.

ABOUT STILL PICTURES...

Still pictures are a nice addition to CD-i and Video CD 2.0 formats (Note that). CD-i and Video CD have different limitations for still pictures summarized in the table below. Prior to version 2.0, the Video CD specification did not allow still pictures, and now they are only allowed in the segment play area, which is the CDROM track 1. Both allow mixing video, audio and still pictures.

	CD-i (`format=cdi`)	Video CD Segment Play (`format=vcdseg`)
NTSC picture size	maximum area of 221,184 pixels. For example 384x576	normal (352x240) and high (704x480) resolution
PAL picture size	same as NTSC	normal (352x88) and high (704x576) resolution
Number of still streams	16	two: one normal and one high resoltion stream
Data size	47,104 bytes	47,104 bytes for normal resolution; 229,376 bytes for high resolution

About the only other constraints on still pictures is that you must leave enough between pictures to load the data. You may be surprised to know that although quadruple speed cdrom drives are available, CD-i and Video CD players cannot take advantage of them, because the sector rate is defined to be 75 sectors per second! Each sector on the cdrom is 2352 bytes, but after you subtract the various overhead information, it can contain only 2294 bytes of pure video data. For example, if you have nothing but video on your cdrom, the mximum data rate is 2,294 X 75 = 172,050 bytes per second. If your still pictures are 25,000 bytes each, then you cannot display them faster than about 7 frames per second; less if you include audio; even less

ABOUT VIDEO SEQUENCES...

For a generic system stream (i.e., not CD-i and not Video CD), there is only one limitation: it must be encoded as a fixed bitrate stream. This is not a requirement of the MPEG standard, but it is a practical limitation for this software.

CD-i and Video CD 2.0 formats have a number of limitations for video sequences; some are the same, some slightly different. The important ones are summarized in the table below. There are a few other limitations, but these are taken care of by specifying `format=cdi`, `vcdseg` or `videocd`. Consult the standards documents for details.

	CD-i (`format=cdi`)	Video CD (`format=vcdseg` or `videocd`)
picture size and frame rate	Maximum rate of 9,900 macroblocks (16x16 pixels) per second. For example: 352x240 @ 30 Hz 352x240 @ 29.97 Hz 352x288 @ 25 Hz 352x288 @ 25 Hz 352x288 @ 23.976 Hz	352x240 @ 29.97 Hz 352x240 @ 23.976 Hz 352x288 @ 25 Hz
Bit rate	Max. of approx. 1,400,000 bits per second from disk; 5,000,000 bits/sec from memory.	Segment Play (track 1): 1,377,600 bits/sec. max. All other tracks: 1,151,930 bits/sec. max.
Buffer size	Max of 40,960 bytes	Same
Motion vectors	+/- 128 pixels max. (full-pixel) +/- 64 pixels max. (half-pixel)	Same
Number of streams	16 max.	1 max.

The buffer size and motion vectors are complex, and fall under the nomenclature of "Constrained Parameter Streams". For simplicity, NVR's video compressor includes support for creating such streams. Simply specify the `-cpf` flag to `vmpegcomp`, or by selecting CD-i or Video CD in the Compressor GUI.

CD and CD-i include nformation about a system stream at the sector level and these support the concept of a pause or “chapter mark”. You can cause a chapter mark by including an EOR string at the end of each line.

SEQUENCE LIST FILE FORMAT

The format consist of one field and two optional fields separated by white space on a single line. The fields from left-to-right are filename, timecode, EOR (end of record) marker. The timecode is specified in hour:min:sec:fraction. The # character introduces a comment and the remaining part of the line is ignored. The EOR marker is meaningful only for CDROM applications where an EOR bit may be set in the sector header.

For most applications, it is sufficient to simply list the files, and the effect is that all sequences will run together. If a timecode is provided, then that sequence is delayed until then. For example::

```
# video list
sequence1.vmpeg
sequence2.vmpeg
```

Another possiblility:

```
# timed audio list
sequence1.ampeg          00:00:00:00
sequence2.ampeg          00:00:45:00
```

SEE ALSO

player, recorder, compressor, vmpegcomp, ampegcomp

EXIT CODES

For the benefit of those doing shell programming with mpegsmux, the following exit codes are used by mpegsmux:

Exit value	Meaning
0	No error
1	Miscellaneous errors
2	Internal problem
3	A video or audio file is encoded incorrectly
4	Write failed because disk is full
5	Command line parameter error, or syntax error in sequence list or still files

COPYRIGHT

BUGS

No data streams are allowed on compact discs.

Only `videocd`, `vcdseg` and `cdi` formats can contain still pictures. Generic system streams cannot.

dant servers mode but there are too few servers running currently.) An **S** in column one indicates that this client has a shared license for the indicated feature. Shared license groups share the same LID.

The next field, CID, is the client ID. Each client application will be assigned a unique ID. The CID may be used with the **-b** option described above. The LID is the license ID. These are unique license numbers. Each client will normally have unique LID's for each licensed feature, except when the license is shared. If clients share a license, they share the LID.

The user is displayed as *user@host,display* where *display* is the value of the X DISPLAY when the application was invoked. The *display* is absent for non-X applications. The **-l** option scans all servers of the network, as well as the one defined by SYNC_ELMHOST, if any.

-n
: Generates a numeric-only server code. This option should be used in conjunction with the **-c** option above.

-q
: Query. Print various environment information, including the name and value of the SYNC_ELMHOST environment variable, the current port, and license service name.

-r *#*
: Ready-key mode, for use with **-c** option. Depending on the parameter *#*, do not print the code(s) or prompt for the key:

If the number *#* is positive, then do not print the server code(s); only prompt for the server key. The *#* in this case is the number of servers that will be running, usually 1. (It will only be greater than 1 when backup servers are used.) Thus, "-r 1" prompts for a key for a single server configuration. If the number # is negative, then instead no key prompt will be issued, instead only that number of codes indicated will be printed. Thus, "-r -1" prints the current server code but does not prompt for a key.

-v *#* *
: Change the license manager log file verbose level to *#*. This may be used to change the level after the daemon has started. See nvrlmd(1) for the list of levels and more details.

-z *
: Zero log file. One method to clean up the log file every evening on is:

```
cp nvrlm.log nvrlm.log.old
SYNC ELMHOST=somehost nvradmin -z
```

where somehost is the host name where the nvrlmd is running. Alternately, the -m option of nvrlmd(1) may be used to automatically keep the log file under control. Note that truncating the log file by other means would be ineffectual since nvrlmd keeps the log file open at all times. The next nvrlmd log file write would simply use the old file pointer, filling the file to the original length. Thus **-z** of *nvradmin* or **-m** of *nvrlmd* must be used to truncate the file.

FILES

/usr/lib/elm - default key file directory.

SEE ALSO

nvrlmd(1), xnvrusage(1).

data files without informational headers (like a raw *Abekas* files). In the first case the attributes *prefix, suffix, ndigits, start,* and *stop* are used to describe the file location and in the latter case the attribute *read=<filename>* is used.

EXAMPLES

An example nvridx file which describes a series of 10 *tiff* files might look something like this:

```
nvr_idx           # the magic number. It must be the first 7 bytes
                  # of the file

type=tiff         # this describes the pixel organization
colorspace=rgb    # ... the color space
size=640x480      # ... dimensions
sample=444        # sample rate of the colorspace, 420
                  # or 4:2:0 works here
interlace=false   # this file is non-interlaced
framerate=10.0    # frame rate of the material
nframes=10        # number of frames for the movie

#
# This next section will construct an implicit list of file names that
# looks like
#        /tiff-files/NLM.Anim.01.tiff
#        /tiff-files/NLM.Anim.02.tiff
#        ...
#        /tiff-files/NLM.Anim.10.tiff
#
read              # all by itself 'read' means an implicit
                  # list of files
prefix=/tiff-files/NLM.Anim.
suffix=.tiff
ndigits=2         # with this the numbers are 01, 02, 03, ..., 10
                  # without ndigits, the numbers
                  # are 1, 2, 3, ..., 10, 11
start=1
stop=10
```

And, and example nvridx file which describes an *abekas* file might look something like this:

```
nvr_idx
type=abekas       # This descibes the pixel organization
#
# single file means no need for prefix, suffix, ndigits, start, stop
#
read=/abekas-files/flowerGarden.abekas
colorspace=yc     # ... the colorspace
```

NAME

nvrlmd - NVR's license server daemon

SYNOPSIS

`nvrlmd [options]`

DESCRIPTION

The *nvrlmd* command starts the NVR *license server daemo* n (based on the Elan License Manager from Elan Computer Group, Inc.). The preferred method for executing this command is during system boot via the system startup script, usually **/etc/rc.local** or **/etc/inittab**. If *nvrlmd* is not running, an NVR application will attempt to launch the daemon. When running, the daemon will process client requests, which include issuing and returning licenses, and interfacing with nvradmin to list the current users of licensed applications.

For each NVR licensed application, there are one or more key files. These files are expected to be in the key directory, specified with the **-e** option or specified in the users NVRKEYDIR environment variable. If not specified in either manner, the key directory defaults to /usr/lib/elm. The key file has the same name as the feature and is created by nvradmin(1), for example the NVR Digital Media Development Kit feature name is "000", so the command:

```
nvradmin –c –n –e $NVRHOME/adm/keys NVRDevelKit
```

would look for that feature key in `$NVRHOME/adm/keys/000`. Refer to *nvradmin* (1) for other options. Each key file contains the encrypted key which encodes the feature name, the maximum number of licenses available for the feature, and the feature's expiration date (if applicable). The manager *nvrlmd* (1) reads this key and will not issue more licenses than indicated or past the expiration date (if applicable).

OPTIONS

-e *path* — Specifies the directory, or set of directories that contain keys for loading by nvrlmd. Directories are specified as a colon (:) separated list of directory names.

-i — Print the ELM release number, the vendor ID, and exit. The vendor ID provides a way to verify that the same version is being used among all the NVR license management utilities. The vendor ID should be the same for *nvrlmd*, *nvradmin*, and *xnvrusage*.

-l *file* — Creates a log file named file and writes all relevant information to that file. If the file already exists, *nvrlmd* will append to the end of the file. Use of a log file is highly recommended as it provides a way of tracking problems if they occur.

-m *size* — Limit the log file to the size specified. With this option, the log file is "self maintained" by *nvrlmd* such that *nvrlmd* limits the size that the log file may grow to. When this size is reached, the log file named file is moved to file.old and the current log file is truncated.

The size is an integer or floating point number. It may also be specified in kilobytes by suffixing a **k** or megabytes by suffixing an **m** to the size. For efficiency, the log file size is actually checked only every 100 lines, so it might

NAME

Oneplay - simple playback interface for MPEG

SYNOPSIS

`oneplay [options] videofile [ audiofile ]`

OPTIONS

-24bit	display in 24 bit color
-12bit	display in 12 bit color
-232	display in 7-bit color
-332	display in 8-bit color
-gray7	display in 7-bit grayscale
-gray	display in same as gray7
-gray8	display in 8-bit grayscale
-555	display in 125-colors; faster than -232, but more memory
-666	display in 216-colors; faster than -332, but more memory

DESCRIPTION

This simple program is intended as a prototype for incorporating MPEG playback into your application. Source can be found in $NVRHOME/src.

SEE ALSO

player

media (i.e., analog video, JPEG, and MPEG). The MPEG Player interface comes up as the default, but may be changed upon invocation via the *-jpeg* or *-analog* options. Transitioning between Player interfaces can also be done while the application is running. This is done via the **Select Source** pulldown menu.

RESOURCES

NVRPlayer*engine : <e>

[for analog video]
Specifies the default engine; the connector of the frame buffer to which the video signal to be played is connected. The engine <e> is in the range <0..1>.

NVRPlayer*imagePath : <path>

Specifies a directory containing .miff files used by the player application. The default is $NVRHOME/images.

NVRPlayer*sourceProtocol : <p>

Specifies the default type of source device control.
<p> = 0 specifies nocontrols/camera mode
<p> = 1 specifies Pioneer LD-V
<p> = 2 specifies ViSCA/LANC

NVRPlayer*tty : <n>

Specifies the default tty used for control of video source device.
<p> = 1 specifies /dev/ttya
<p> = 2 specifies /dev/ttyb

SEE ALSO

compressor, recorder

NAME

Recorder

SYNOPSIS

```
recorder [options]
```

OPTIONS

-camera
: Specifies no device control desired; simply displays in the application’s video display the video signal present on the specified video input. Equivalent to the -nocontrols option.

-destFile <file name>
: Specifies a file relative to the destPath to which you will record (the destination file).

-destPath <path>
: Specifies the directory in which to place/locate a file to which you will record.

-engine <e>
: Specifies the connector <e> in the range of <0..1> of the frame buffer to which the video signal of interest is connected.

-lanc
: Make an attempt to connect to a LANC-controllable device connected through a Sony Vbox using the ViSCA protocol. Equivalent to the -visca switch. This should be used in conjunction with the -ttya or -ttyb switch.

-ldv
: Make an attempt to connect to the Pioneer LDV family of video disc players. This should be used in conjunction with the -ttya or -ttyb switch.

-nocontrols
: Specifies no device control desired; simply displays in the application’s video display the video signal present on the specified video input. Equivalent to the -camera option.

-ttya
: Specifies that the Pioneer LDV or ViSCA/LANC device is connected to the workstation on tty port A. This should be used in conjunction with the -ldv or -lanc switch.

-ttyb
: Specifies that the Pioneer LDV or ViSCA/LANC device is connected to the workstation on tty port B. This should be used in conjunction with the -ldv or -lanc switch.

-visca
: Make an attempt to connect to a LANC-controllable device connected through a Sony Vbox using the ViSCA protocol. Equivalent to the -lanc switch. This should be used in conjunction with the -ttya or -ttyb switch.

DESCRIPTION

This application program is primarily used for digitizing analog video and audio. The recording process performs real-time JPEG compression and writing to disk at a variety of frame rates.

RESOURCES

NVRRecorder*destinationPath : <path>
: Specifies a default directory in which to place/locate files to which you will

NAME

Vmpegcomp - MPEG video compression engine

SYNOPSIS

`vmpegcomp [pipeline options] [compression options]`

PIPELINE OPTIONS

See `imageop` for a description of the pipeline options. All operators are valid except `io write`. The final pipeline must be:

```
io write=filename type=mpeg
```

Followed by the compression options described below.

COMPRESSION OPTIONS

-b*r[,s[,d[,i]]]* — Specify the bitrate, *r* (bits/second), buffer size, *s* (bits), danger margin, *d* (float fraction of full buffer size) and the initial buffer fullness, *i* (bits or float buffer fraction).

-cpf — Force the “constrained parameters” flag as described in the MPEG 1 standard. This option makes sure that the other selected options fit the constraints and will print an error if they done. This also ensures that the Video Buffering Verifier size is set correctly, and that no motion vectors exceed a magnitude of 64. This flag is necessary for some hardware MPEG players. CD-I authoring systems typically insist on this.

-C*c* — Center the quantizer matrices around the specified center, *c*. For example, the default quantizer matrices assume an average quantizer value of 16. If the combination of pictures and bitrate yields an average quantizer of, say 3, then more dynamic range could be realized by specifying a new center of between 3 and 16.

-g*p,b* — Specify The Number Of P Pictures, *P*, Per Gop And The Number Of B Pictures *B*, Per I/p Picture In The Gop. If unspecified, the default is -g4,2 which will produce an I frame every 15 frames.

-h*n* — Specify the frequency of sequence headers for each I frame. For example, -h4 means to put out a sequence header every four I frames. If the frame rate is 30fps (-p30) and the default GOP is used (-g4,2), this will produce a sequence header every two seconds (a desirable thing for Video CD and CD-i). This helps hardware decoders to do random access.

-M*a[,r[,p]]* — Select a motion estimation algorithm, *a*, motion estimation radius, *r*, and motion estimation parameter, *p*. The algorithms are 0 for exhaustive MAD, 1 for Jain & Jain, 2 for a user-provided motion estimator; 3 for multi-processor exhaustive MAD. All motion estimation does a final half-pel refinement search.

-p*r* — Specify a picture rate, *r*. It must be a standard rate, one of: 23.976, 24.0, 25.0, 29.97, 30.0, 50.0, 59.94, or 60.0. It may also be one of a few non-standard rates: 5, 6, 10, 15 or 20 (but these will soon be obsolete). The rate *r* is used to define the MPEG pictures per second field and to implement the time code pictures per second.