21. Tips

In this section, you will find ways to solve common problems using Cinelerra. This section is arranged in order of the problems and what tools are used to solve them. Following sections are arranged in order of the tools and their uses.

21.1 Encoding into Dolby Pro Logic

Dolby pro logic is an easy way to output 6 channel audio from a 2-channel soundcard with degraded but useful results. Rudimentary Dolby pro logic encoding can be achieved with clever usage of the effects.

First, create the front left and right channels. Create 2 audio tracks, each carrying either the left or right channel. Pan the left channel to the left and the right channel to the right with **pan**.

Next, create the rear left and right channels. Create another 2 audio tracks as above: the left channel panned left and the right channel panned right. Then apply **invert audio** to both new channels and the signals will come out of the rear speakers.

Next, create the center channel by creating a single audio track with monaural audio from a different source. Center it with the **pan** control and the signal will come out of the center speaker.

If a copy of the signal in the back speakers is desired in any single front speaker, the signal in the back speakers must be delayed by at least 0.05 seconds and a single new track should be created. Pan the new track to orient the signal in the front speakers.

If the same signal is desired in all the speakers except the center speaker, delay the back speakers by 0.5 seconds and delay either the front left or front right by 0.2 seconds.

If you want to hear something from the subwoofer, create a new track, select a range, drop a synthesizer effect, and set the frequency below 60 Hz. The subwoofer merely plays anything below 60Hz or so.

Other tricks you can perform to separate the speakers are parametric equalization to play only selected ranges of frequencies through different speakers and lowpass filtering to play signals through the subwoofer.

21.2 Cleaning analog TV

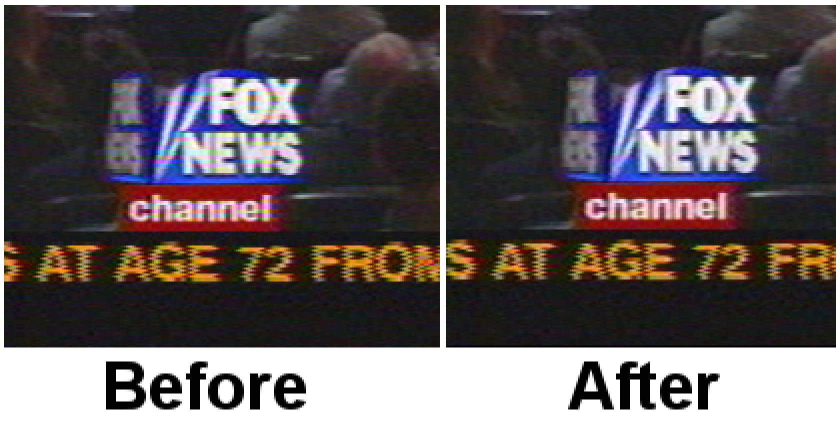
Unless you live in a rich nation like China or are a terrorist, you probably record analog TV more than you record digital TV. The picture quality on analog TV is horrible but you can do things in Cinelerra to make it look more like it did in the studio.

First, when capturing the video, capture it in the highest resolution possible. For Europeans it is 720x576 and for North Americans it is 720x480. Do not bother adjusting the brightness or contrast in the recording monitor, although maxing out the color is useful. Capture it using MJPEG or uncompressed Component Video if possible. If those are too demanding, then capture it using JPEG. RGB should be a last resort.

Now on the timeline use **Settings->Format** to set a YUV colorspace. Drop a **Downsample** effect on the footage. Set it for

```
Horizontal:        2
Horizontal offset: 0
Vertical:          2
Vertical offset:   0
      red
  x   green
  x   blue
      alpha
```

Use the camera tool to shift the picture up or down a line to remove the most color interference from the image. This is the difference we are looking for:



If you have vertical blanking information or crawls which constantly change in each frame, block them out with the **Mask** tool. This improves compression ratios.

This is about all you can do without destroying more data than you would naturally lose in compression. The more invasive cleaning techniques involve deinterlacing.

21.3 Defeating interlacing

Interlacing is done on most video sources because it costs too much to build progressive scanning cameras and progressive scanning CRT's. Many a consumer has been disappointed to spend 5 paychecks on a camcorder and discover what horrible jagged images it produces on a computer monitor.

As for progressive scanning camcorders, forget it. Cost factors are probably going to keep progressive scanning cameras from ever equaling the spatial resolution of interlaced cameras. Interlacing is here to stay. That is why they made deinterlacing effects in Cinelerra.

We do not believe there has ever been a perfect deinterlacing effect. They are either irreversible or do not work. Cinelerra cuts down the middle by providing deinterlacing tools that are irreversible sometimes and do not work sometimes but are neither one or the other.

- **Line Doubling** This one is done by the **Deinterlace** effect when set to **Odd lines** or **Even lines**. When applied to a track it reduces the vertical resolution by 1/2 and gives you progressive frames with stairstepping. This is only useful when followed by a scale effect which reduces the image to half its size.
- **Line averaging** The **Deinterlace** effect when set to **Average even lines** or **Average odd lines** does exactly what line doubling does except instead of making straight copies of the lines it makes averages of the lines. This is actually useful for all scaling.
  There is an option for adaptive line averaging which selects which lines to line average and which lines to leave interlaced based on the difference between the lines. It does not work.
- **Inverse Telecine** This is the most effective deinterlacing tool when the footage is an NTSC TV broadcast of a film. See section Inverse telecine.
- **Time base correction** The first three tools either destroy footage irreversibly or do not work at times. **Time base correction** is last because it is the perfect deinterlacing tool. It leaves the footage intact. It does not reduce resolution, perceptually at least. It does not cause jittery

timing.

- The **Frames to Fields** effect converts each frame to two frames, so it must be used on a timeline whose project frame rate is twice the footage's frame rate. In the first frame it puts a line-averaged copy of the even lines. In the second frame it puts a line-averaged copy of the odd lines. When played back at full framerates it gives the illusion of progressive video with no loss of detail.
  Best of all, this effect can be reversed with the **Fields to frames** effect. That one combines two frames of footage back into the one original interlaced frame of half the framerate.
  Be aware that frames to fields inputs frames at half the framerate as the project. Effects before frames to fields process at the reduced framerate.
  Unfortunately, the output of **Frames to Fields** can not be compressed as efficiently as the original because it introduces vertical twitter and a super high framerate.
  Interlaced 29.97 fps footage can be made to look like film by applying **Frames to Fields** and then reducing the project frame rate of the resulting 59.94 fps footage to 23.97 fps. This produces no timing jitter and the occasional odd field gives the illusion of more detail than there would be if you just line averaged the original.

HDTV exceptions
1920x1080 HDTV is encoded a special way. If it is a broadcast of original HDTV film, an inverse telecine works fine. If it is a rebroadcast of a 720x480 source, you need to use a time base and line doubling algorithm to deinterlace it, See section 1080 to 480.

21.4 Making video look like film

Video sweetening is constantly getting better. Lately the best thing you can do for dirt cheap consumer camcorder video is to turn it into progressive 24 fps output. While you can not really do that, you can get pretty close for the money. Mind you, since this procedure can degrade high quality video just as easily as it improves low quality video, it should only be used for low quality video.

1. Set project framerate to twice the video framerate.
2. Apply a **Sharpen** effect. Set it to sharpness: 25, no interlacing, and horizontal only.
3. Drop a **Frame to Fields** effect on the same track. Set Average Empty Rows to on and play through the video a few times to figure out which field is first. If the wrong field is first, the motion is shaky. Secondly, any

editing in the doubled frame rate may now screw up the field order. We are still figuring out the easiest way to support warnings for field glitches but for now you need to go back to the normal framerate to do editing or play test to make sure the fields are right.
4. Render just the video to the highest quality file possible.
5. Import the video back to a new track. Set the project framerate to 24. The new track should now display more filmish and sharper images than the original footage.

This entire procedure could be implemented in one non-realtime effect, but the biggest problem with that is you will most often want to keep the field based output and the 24 fps output for posterity. A non-realtime effect would require all that processing just for the 24 fps copy. Still debating that one.

21.5 Clearing out haze

You probably photograph a lot of haze and never see blue sky. Even if you can afford to briefly go somewhere where there is blue sky, horizon shots usually can stand for more depth. This is what the **gradient effect** is for.

Drop the gradient effect on hazy tracks. Set the following parameters:

- Angle: 0
- Inner radius: 0
- Outer radius: 40
- Inner color: blue 100% alpha
- Outer color: blue 0% alpha

It is important to set the 0% alpha color to blue even though it is 0% alpha. The color of the outer alpha is still interpolated with the inner color. This is a generally applicable setting for the gradient. Some scenes may work better with orange or brown for an evening feel.

21.6 Making a ringtone

This is how we made ringtones for the low end Motorola V180's and it will probably work with any new phone. Go to **File->Load files...** and load a sound file with Insertion strategy: **Replace current project**. Go to **Settings->Format** change **Channels** to 1 and **Samplerate** to 16000 or 22050.

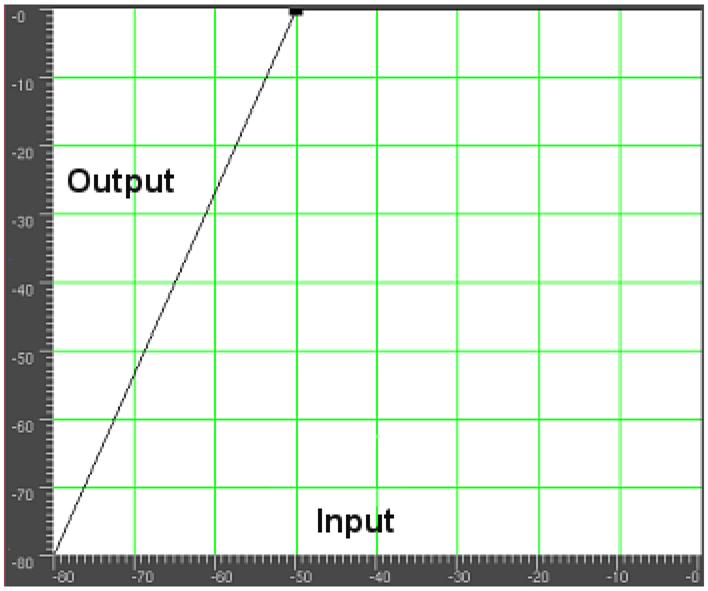
Either highlight a region of the timeline or set in/out points to use for the ringtone. To improve sound quality on the cell phone, you need the maximum amplitude in as many parts of the sound as possible. Right click on track Audio 1 and select **Attach effect...**. Highlight the **Compressor** effect and hit **Attach** in the attachment popup.

Make sure the insertion point or highlighted area is in the region with the Compressor effect. Right click on track Audio 2 and select **Attach effect...**. Highlight **Audio 1: Compressor** and hit **Attach**. Click the Audio1 Compressor's magnifying glass 🔍 to bring up the compressor GUI.

Set the following parameters:

- Reaction secs: **-0.1**
- Decay secs: **0.1**
- Trigger Type: **Total**
- Trigger: **0**
- Smooth only: **No**

Click **Clear** to clear the graph. Click anywhere in the grid area and drag a new point to 0 Output and -50 Input. The graph should look like this.



Go to **File->Render**. Specify the name of an mp3 file to output to. Set the file format to **MPEG Audio**. Click the wrench 🔧 for Audio and set **Layer** to **III** and **Kbits per second** to **24** or **32**. Check **Render audio tracks** and uncheck **Render video tracks**. Hit OK to render the file.

The resulting `.mp3' file must be uploaded to a web server. Then, the phone's web browser must download the `.mp3' file directly from the URL. There also may be a size limit on the file.

21.7 Time stretching audio

It may appear that time stretching audio is a matter of selecting a region of the audio tracks, enabling recording for the desired tracks, going to **Audio->Render Effect**, and applying **Time Stretch**. In actuality there are 3 audio effects for time stretching: **Time Stretch**, **Resample**, and **Asset info dialog**.

Time Stretch applies a fast Fourier transform to try to change the duration without changing the pitch, but this introduces windowing artifacts to the audio. It is only useful for large changes in time because obvious changes in duration make windowing artifacts less obtrusive.

For smaller changes in duration, in the range of 5%, **Resample** should be used. This changes the pitch of the audio but small enough changes are not noticeable. Resample does not introduce any windowing artifacts, so this is most useful for slight duration changes where the listener is not supposed to know what is going on.

Another way to change duration slightly is to go to the **Resources** window, highlight the **media** folder, right click on an audio file, click on **Info**. Adjust the sample rate in the **Info** dialog to adjust the duration. This method also requires left clicking on the right boundary of the audio tracks and dragging left or right to correspond to the length changes.

21.8 Video screen captures

We explain here how to record video screen captures and edit them in Cinelerra.

First, you have to record the video with xvidcap. You can find that utility in most distributions' repositories, or download it here: http://xvidcap.sourceforge.net

First, capture the screen:
```
xvidcap --fps 10 --cap_geometry 1280x1024+0+0 --file "file1.mpeg" --gui no --audio no
```

Do not forget to change the geometry option according to your screen size. Then, convert the `` `file1.mpeg' `` file you obtained into an mpeg file suitable for Cinelerra:

```
ffmpeg -r 10 -i file1.mpeg -s 1280x1024 -b 3000 -aspect 1.33 -r 25 file2.mpeg
```

You can now load that file into Cinelerra. Make sure you properly set the video format of your project (size, frame-rate, aspect-ratio)

When you have finished editing your video, you have to render it. Render it as a jpeg-sequence. It is recommended to write the jpeg files in a new folder, since there probably will have a lot of files created.

Then, open a shell window, and cd into that folder. Encode the jpeg files using those commands:

First pass:

```
mencoder "mf://*.jpg" -mf fps=25 -oac pcm -sws 2 -vf scale=\
1280:1024,hqdn3d=2:1:2 -ovc lavc -lavcopts vcodec=mpeg4:\
vbitrate=800:aspect=4/3:vpass=1 -ofps 10 -of avi -o /dev/null \
-ffourcc DIVX
```

Second pass:

```
mencoder "mf://*.jpg" -mf fps=25 -oac pcm -sws 2 -vf \
scale=1280:1024,hqdn3d=2:1:2 -ovc lavc -lavcopts \
vcodec=mpeg4:vbitrate=800:aspect=4/3:vpass=2 -ofps 10 -of avi \
-o ../rendered_file.avi -ffourcc DIVX
```

You can also render the video to mpeg4 directly from Cinelerra if you wish to.

21.9 Improving performance

For the moment GNU/Linux is not an excellent desktop. It is more of a server. Most of what you will find on modern GNU/Linux distributions are faceless, network-only programs strategically designed to counteract one Microsoft server feature or another and not to perform very well at user interaction. There are a number of parameters on GNU/Linux, which ordinary people can adjust to make it behave more like a thoroughbred in desktop usage.

21.9.1 Disabling swap space

On systems with lots of memory, Cinelerra sometimes runs better without a swap space. If you have 4 GB of RAM, you are probably better off without a swap space. If you have 512MB of RAM, you should keep the swap. If you want to do recording, you should probably disable swap space in any case. There is a reason for this. GNU/Linux only allows half the available memory to be used.

Beyond that, it starts searching for free pages to swap, in order to cache more disk access. In a 4 GB system, you start waiting for page swaps after using only 2 GB.

The question then is how to make GNU/Linux run without a swap space. Theoretically it should be a matter of running

```
swapoff -a
```

Unfortunately, without a swap space the kswapd tasklet normally spins at 100%. To eliminate this problem, edit `` `linux/mm/vmscan.c' ``. In this file, put a line saying `return 0;` before it says

```
    /*
     * Kswapd main loop.
     */
```

Then recompile the kernel.

21.9.2 Enlarging sound buffers

In order to improve realtime performance, the audio buffers for all the GNU/Linux sound drivers were limited from 128k to 64k. For recording audio and video simultaneously and for most audio recording this causes dropouts. Application of low latency and preemptible kernel patches make it possible to record more audio recordings but it does not improve recording video with audio. This is where you need to hack the kernel.

To see if your sound buffers are suitable, run the included soundtest program with nothing playing or recording. This allocates the largest possible buffers and displays them. If the **Total bytes available** is under 131072, you need to see about getting the buffers enlarged in the driver. While many drivers differ, we have a hack for at least one driver.

This only applies to the OSS version of the Soundblaster Live driver. Since every sound card and every sound driver derivative has a different implementation you will need to do some searching for other sound cards. Edit
`` `linux/drivers/sound/emu10k1/audio.c' ``

Where it says

```
if (bufsize >= 0x10000)
```

change it to:

```
if (bufsize > 0x40000)
```

Where it says

```
    for (i = 0; i < 8; i++)
        for (j = 0; j < 4; j++)
```

change it to:

```
    for (i = 0; i < 16; i++)
        for (j = 0; j < 4; j++)
```

In `` `linux/drivers/sound/emu10k1/hwaccess.h' ``, change

```
#define MAXBUFSIZE 65536
```

to

```
#define MAXBUFSIZE 262144
```

Finally, in `` `linux/drivers/sound/emu10k1/cardwi.h' ``, change

```
#define WAVEIN_MAXBUFSIZE         65536
```

to

```
#define WAVEIN_MAXBUFSIZE         262144
```

Then recompile the kernel modules.

21.9.3 Freeing more shared memory

The GNU/Linux kernel only allows 32MB of shared memory to be allocated by default. This needs to be increased to do anything useful. When launched, Cinelerra may remind you that with the following error message:

```
The following errors occurred:
void MWindow::init_shm0: WARNING:/proc/sys/kernel/shmmax is 0x2000000, which is too low.
Before running Cinelerra do the following as root:
echo "0x7ffffff">/proc/sys/kernel/shmmax
```

For a permanent change, add to the `` `/etc/sysctl.conf' `` file the following line:

```
kernel/shmmax=0x7fffffff
```

or if you prefer:

```
kernel.shmmax = 2147483647
```

For the first time, to avoid restarting your computer, use the following command as root:

```
sysctl -p
```

21.9.4 Speeding up the hard drive

This is a very popular command sequence among GNU/Linux gurus, which is not done by default on GNU/Linux distributions.

```
hdparm -c3 -d1 -u1 -k1 /dev/hda
```

- `` `-c3' `` puts the hard drive into 32 bit I/O with sync. This normally does not work due to inept kernel support for most IDE controllers. If you get lost interrupt or SeekComplete errors, quickly use `` `-c0' `` instead of `` `-c3' `` in your command.
- `` `-d1' `` enables DMA of course. This frees up the CPU partially during data transfers.
- `` `-u1' `` allows multiple interrupts to be handled during hard drive transactions. This frees up even more CPU time.
- `` `-k1' `` prevents GNU/Linux from resetting your settings in case of a glitch.

21.9.5 Disabling cron

GNU/Linux runs some daily operations like compressing man pages. These may be acceptable background tasks while compiling or word processing but not while playing video. Disable these operations by editing `` `/etc/rc.d/init.d/anacron' ``.

Put `exit` before the first line not beginning in `#`.

In `` `/etc/rc.d/init.d/crond' `` put `exit` before the first line not beginning in `#`. Then reboot.

You can not use the `at` command anymore, but who uses that command anyways?

21.9.6 Reducing USB mouse sensitivity

Gamers like high resolution mice, but this can be painful for precisely positioning the mouse on a timeline or video screen. XFree86 once allowed you to reduce PS/2 mouse sensitivity using commands like `xset m 1 1` but you are out of luck with USB mice or KVM's.

We have a way to reduce USB mouse sensitivity but it requires editing the kernel source code. Even though USB mice have been supported for years, the kernel source code for USB mice is constantly being rewritten. These instructions were relevant for 2.6.12.3. Edit `` `/usr/src/linux/drivers/input/mousedev.c' ``.

After the line saying

```
struct mousedev_hw_data {
```

put

```
#define DOWNSAMPLE_N 100
#define DOWNSAMPLE_D 350
int x_accum, y_accum;}
```

Next, the section which says something like:

```
switch (code) {
    case REL_X: mousedev->packet.dx += value; break;
    case REL_Y: mousedev->packet.dy -= value; break;
    case REL_WHEEL:     mousedev->packet.dz -= value; break;
}
```

must be replaced by

```
switch (code) {
    case REL_X:
    mousedev->packet.x_accum += value * DOWNSAMPLE_N;
    mousedev->packet.dx += (int)mousedev->packet.x_accum
    / (int)DOWNSAMPLE_D;
    mousedev->packet.x_accum -=
    ((int)mousedev->packet.x_accum / (int)DOWNSAMPLE_D)
    * (int)DOWNSAMPLE_D;
    break;
    case REL_Y:
    mousedev->packet.y_accum += value * DOWNSAMPLE_N;
    mousedev->packet.dy -= (int)mousedev->packet.y_accum
    / (int)DOWNSAMPLE_D;
    mousedev->packet.y_accum -=
    ((int)mousedev->packet.y_accum
    / (int)DOWNSAMPLE_D) * (int)DOWNSAMPLE_D;
    break;
    case REL_WHEEL: mousedev->packet.dz -= value; break;
}
```

Change the value of **DOWNSAMPLE_N** to change the mouse sensitivity.

21.9.7 Assorted X tweeks

XFree86 by default can not display Cinelerra's advanced pixmap rendering very fast. The X server stalls during list box drawing. Fix this by adding a line to your XF86Config* files.

In the **Section "Device"** area, add a line saying:

```
Option "XaaNoOffscreenPixmaps"
```

and restart the X server.

Screen blanking is really annoying, unless you are fabulously rich and can afford to leave your monitor on 24 hours a day without power saving mode. In `/etc/X11/xinit/xinitrc' put

```
xset s off
xset s noblank
```

before the first `if` statement.

How about those windows keys which no GNU/Linux distribution even thinks to use. You can make the window keys provide ALT functionality by editing `/etc/X11/Xmodmap'. Append the following to it.

```
keycode 115 = Hyper_L
keycode 116 = Hyper_R
add mod4 = Hyper_L
add mod5 = Hyper_R
```

The actual changes to a window manager to make it recognize window keys for `ALT` are complex. In **FVWM** at least, you can edit `/etc/X11/fvwm/system.fvwm2rc' and put

```
Mouse 0 T A move-and-raise-or-raiselower
#Mouse 0 W M move
Mouse 0 W 4 move
Mouse 0 W 5 move
Mouse 0 F A resize-or-raiselower
Mouse 0 S A resize-or-raiselower
```

in place of the default section for moving and resizing. Your best performance is going to be on FVWM. Other window managers seem to slow down video with extra event trapping and are not as efficient in layout.

21.9.8 Speeding up the file system

You will often store video on an expensive, gigantic disk array separate from your boot disk. You will thus have to manually install an EXT filesystem on this disk array, using the `mke2fs` command. By far the fastest file system is

```
mke2fs -i 65536 -b 4096 my_device
tune2fs -r0 -c10000 my_device
```

This has no journaling, reserves as few blocks as possible for filenames, and accesses the largest amount of data per block possible. A slightly slower file system, which is easier to recover after power failures is

```
mke2fs -j -i 65536 -b 4096 my_device
tune2fs -r0 -c10000 my_device
```

This adds a journal which slows down the writes but makes filesystem checks faster.

21.9.9 Improving Zoran video

Video recorded from the ZORAN inputs is normally unaligned or not completely encoded on the right. This can be slightly compensated by adjusting parameters in the driver sourcecode.

In `/usr/src/linux/drivers/media/video/zr36067.c' the structures defined near line 623 affect alignment. At least for NTSC, the 2.4.20 version of the driver could be improved by changing

```
    static struct tvnorm f60ccir601 = { 858, 720, 57, 788, 525, 480, 16 };
to
    static struct tvnorm f60ccir601 = { 858, 720, 57, 788, 525, 480, 17 };
```

In `/usr/src/linux/drivers/media/video/bt819.c' more structures near line 76 affect alignment and encoding.
For NTSC

```
    {858 - 24, 2, 523, 1, 0x00f8, 0x0000},
could be changed to
    {868 - 24, 2, 523, 1, 0x00f8, 0x0000},
```

Adjusting these parameters may or may not move your picture closer to the center. More of the time, they will cause the driver to lock up before capturing the first frame.

New in 2.6.5:
In the 2.6 kernels, the video subsystem was rewritten again from scratch. To adjust the Zoran parameters go to `drivers/media/video/zoran_card.c' and look for a group of lines like

```
    static struct tvnorm f50sqpixel = { 944, 768, 83, 880, 625, 576, 16 };
    static struct tvnorm f60sqpixel = { 780, 640, 51, 716, 525, 480, 12 };
    static struct tvnorm f50ccir601 = { 864, 720, 75, 804, 625, 576, 18 };
    static struct tvnorm f60ccir601 = { 858, 720, 57, 788, 525, 480, 16 };

    static struct tvnorm f50ccir601_lml33 = { 864, 720, 75+34, 804, 625, 576, 18 };
    static struct tvnorm f60ccir601_lml33 = { 858, 720, 57+34, 788, 525, 480, 16 };
```

```
/* The DC10 (57/16/50) uses VActive as HSync, so HStart must be 0 */
static struct tvnorm f50sqpixel_dc10 = { 944, 768, 0, 880, 625, 576, 0 };
static struct tvnorm f60sqpixel_dc10 = { 780, 640, 0, 716, 525, 480, 12 };

/* FIXME: I cannot swap U and V in saa7114, so i do one
 * pixel left shift in zoran (75 -> 74)
 * (Maxim Yevtyushkin <max@linuxmedialabs.com>) */
static struct tvnorm f50ccir601_lm33r10 = { 864, 720, 74+54, 804, 625, 576, 18 };
static struct tvnorm f60ccir601_lm33r10 = { 858, 720, 56+54, 788, 525, 480, 16 };
```

These seem to control the image position. At least for the LML33 the following definition for **f60ccir601_lml33** does the trick.

```
static struct tvnorm f60ccir601_lml33 = { 858, 720, 67+34, 788, 525, 480, 13 };
```

21.10 Translating Cinelerra

This information is needed if you wish to partipate in translating Cinelerra. See section Environment variables, for running Cinelerra in your own language.

21.10.1 Available localizations

There are some existing localizations for cinelerra:

- **DE** - German
- **ES** - Spanish
- **EU** - Basque
- **FR** - French
- **IT** - Italian
- **PT_BR** - Brazilian Portuguese
- **SL** - Slovenian

If your distribution has only UTF-8 support (like Ubuntu), you must create the language charset first. See section Environment variables.

21.10.2 Updating an existing translation

To generate an updated `` `*.po' `` file with the newer strings of Cinelerra source code not yet present in the `` `.po' `` file, run after `./configure`:
`cd po && make`

Then, edit the `` `.po' `` file located in `` `po/' `` directory of your target language and submit the diff file to the Cinelerra-CV team.

21.10.3 Creating a new translation

To create a new translation, run after `./configure`:

```
cd po && make
```

Then, edit the `` `cinelerra.pot' `` file located in `` `po/' `` and add the appropriate translated strings. Rename the file to `` `(lang_prefix).po' `` and add the language prefix to `` `po/LINGUAS' ``. Finally, submit the diff file to the cinelerra-CV team.

21.11 Panning and zooming still images

Cinelerra's powerful keyframe features allow you to use pan and zoom effects on still pictures.

1. Load and create a clip from a still image as described above. Make the clip 10 seconds long.
2. Activate the **automatic generation of keyframes**
3. Using the **transport controls**, go to the beginning of the clip
4. Using the **compositing camera control** set the clip's initial position
5. Using the **transport controls**, move forward a couple of seconds on the clip
6. Dragging on the **compositing camera** move the camera center to a new position further
7. Now, rewind to the beginning of the clip and play it.

You can see that the camera smoothly flows from keyframe point to next keyframe point, as Cinelerra automatically adjusts the camera movement in straight lines from point to point.

21.12 HDV 1080i editing using proxy files

Working with high definition video, which typically comes from HDV camcorders, requires a lot of processing power. Even if the system is able to play a single track at full framerate, it is usually not able to play several tracks simultaneously. Thus simple dissolve transition is slowed down to unacceptable level. Moreover, HDV is in GOP based format, and simple cut requires decoding the whole GOP in less then 1/25s. Thus, one of the possibilities is to perform all edits on low resolution files, and use HDV material only for the final rendering. The workflow presented below was first proposed by Hermann VOSSELER.

21.12.1 Overview

- For each HDV file a proxy is created with a scale of 0.5.
- The project is created with HDV resolution e.g. 1440x1080 and 16/9 aspect.
- New resources are created with both proxies as well as HDV files.
- Each video track must have Camera Automation set to 2.0.
- Editing is performed with the proxy files.
- For HDV rendering, exit Cinelerra and convert the project file with proxychange.py and reopen the project.
- After rendering, if further editing is required, the project file can be back-transformed into a proxy version.

21.12.2 Grabbing HDV from camcorder

There is no perfect solution so far. One possibility is to run the `test-mpeg2` command available with the sources of **libiec61883**. Use this syntax:

```
test-mpeg2 > hdv_tape.mpeg
```

and press **Play** on the camcorder. You should not run any heavy ressources consuming task on your computer since the lack of caching in test-mpeg2 causes framedrops.

New version of dvgrab seems to support HDV. Minimal example:
Syntax:

```
dvgrab -format mpeg2 myclip
```

21.12.3 Using TOC and WAV files

Try using WAV files for sound, and load HDV MPEG-2 files via their generated toc. To create toc files, use the following command:

```
for i in *.mpeg; do mpeg3toc $i `basename $i mpeg`toc; done
```

21.12.4 Making the proxy files

Proxy files can be converted in many ways and can use any format. However, Cinelerra works better when editing non-GOP based formats. To convert your HDV files into I-frame based mjpeg files with 50% scaling, use the following

command:
```
for i in *.mpeg;do mencoder -mc 0 -noskip $i -ovc lavc -lavcopts vcodec=mjpeg -vf scale=720:540
-oac pcm -o `basename $i mpeg`avi; done
```

21.12.5 Converting HDV and proxy files

The **proxychange.py** python script converts HDV to/from proxies. You can download that script here:
http://cvs.cinelerra.org/docs/proxychange.py

It overwrites the existing project files, and creates copy of the original in `projectfile.xml.bak'.

- Proxy -> HDV (e.g. for rendering):
  ```
  ./proxychange.py projectfile.xml -from `proxyfiles/(\w+)\.avi` -to `hdv/\1.toc` -scale 0.5
  ```
- HDV -> Proxy (e.g. after rendering if you want go back to editing):
  ```
  ./proxychange.py projectfile.xml -from `hdv/(\w+)\.toc` -to `proxyfiles/\1.avi` -scale 2.0
  ```

The project XML file is not a perfectly valid XML file. Thus after each Cinelerra "Save", some problem can occur. Sometimes the tags are not closed, <TAG> is not followed by </TAG>. This must be corrected manually.

ACODEC contains some \001 characters. Edit the file manually or use the following command:
```
cat temp001.xml| tr -d `\001` > /tmp/1 ; mv /tmp/1 temp001.xml
```

Update: Recent version of Cinelerra seems to produce valid XML.

21.12.6 Rendering the HDV project

HDV files can be rendered to an YUV4MPEG stream and then encoded to MPEG2 using a modified Mjpegtools binary. `mpeg2enc -verbose 0 -aspect 3 -format 3 -frame-rate 3 -video-bitrate 25000 -nonvideo-bitrate 384 -force-b-b-p -video-buffer 448 -video-norm n -keep-hf -no-constraints -sequence-header-every-gop -min-gop-size 6 -max-gop-size 6 -o %`

Render the sound as an AC3 file, and multiplex both the video and the audio with mplex.

21.12.7 Other issues

When playing MJPEG files, the dissolve transistion does not work properly in

RGBA or YUVA modes but it works fine in RGB or YUV.

21.13 Adding subtitles

There are two methods available for adding subtitles in a video:

- Use Cinelerra's Titler effect. That task is long and fastidious. Moreover, the subtitles are actually incrusted into the image. It is not be possible to display the rendered video without subtitles. If you want your video to be available with subtitles in several languages, you have to render it several times. See section Title, for information about Cinelerra's titler.
- Add the subtitles with a subtitles editor after having rendered the video.

The second method is the one to use if you want your video to be available with subtitles in multiple languages. If you want to produce a DVD, that method is also the only one which is compatible with dvdauthor subtitles feature. If you plan to distribute your video over the internet, one video file and several subtitles files is smaller than several video files, one for each language.

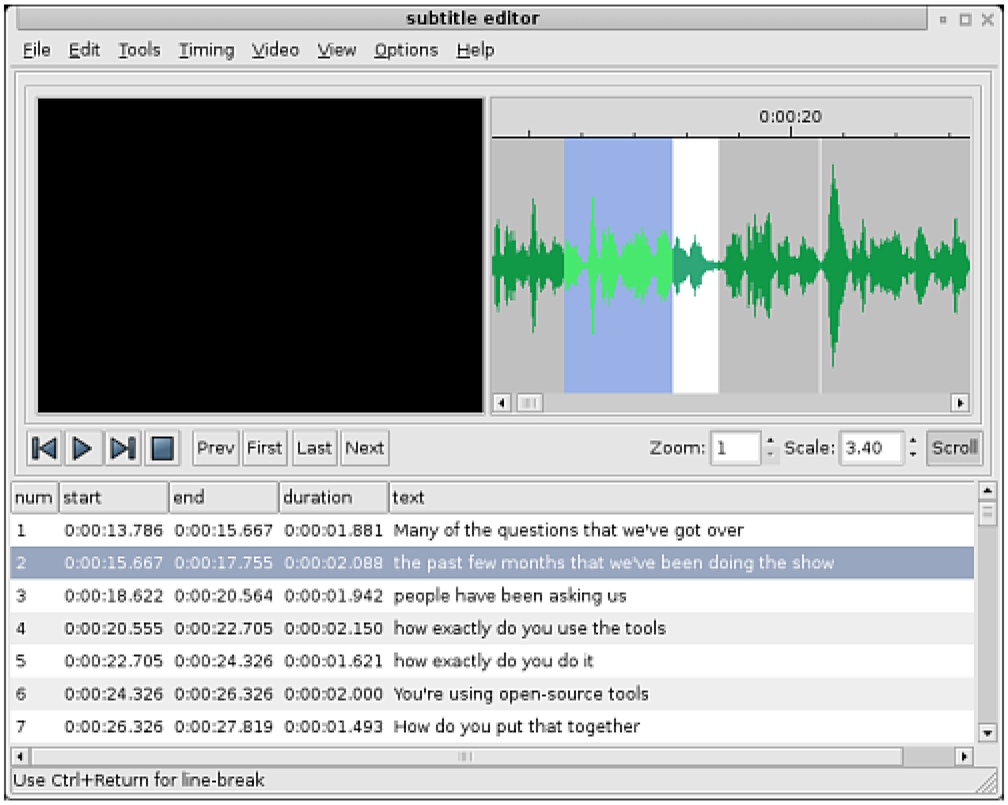
Subtitles text files can be displayed by any decent video player. With mplayer one can use the following syntax:

```
mplayer -sub <the_subtitles_text_file> <the_video_file>
```

A subtitle file is a simple plain text file, which contains the text and the time or frame number where each subtitle should be displayed on the screen.

There are a lot of subtitles editor available on Linux. However, most of them are fine for easing **translation** of subtitles, but are not appropriate to actually **add** and **synchronize** new subtitles on a video. Since video creation is what most of us focus on, the task we are mostly interested in is **creating** subtitles for a video.

We highly recommend you **Subtitleeditor**, which is available here: http://kitone.free.fr/subtitleeditor


subtitle editor
File Edit Tools Timing Video View Options Help
0:00:20
Prev First Last Next
Zoom: 1 Scale: 3.40 Scroll

num	start	end	duration	text
1	0:00:13.786	0:00:15.667	0:00:01.881	Many of the questions that we've got over
2	0:00:15.667	0:00:17.755	0:00:02.088	the past few months that we've been doing the show
3	0:00:18.622	0:00:20.564	0:00:01.942	people have been asking us
4	0:00:20.555	0:00:22.705	0:00:02.150	how exactly do you use the tools
5	0:00:22.705	0:00:24.326	0:00:01.621	how exactly do you do it
6	0:00:24.326	0:00:26.326	0:00:02.000	You're using open-source tools
7	0:00:26.326	0:00:27.819	0:00:01.493	How do you put that together

Use Ctrl+Return for line-break

Subtitleeditor

Subtitleeditor has the huge advantage of displaying the audio waveform. This feature is really important to precisely synchronize subtitles and talks. Keep in mind the synchronization would be lost if you edit your video after having added the subtitles. Adding subtitles should be done after the video edition is finished.

Once the subtitle text file is created, you can:

- Distribute it with your video. People will have to load the appropriate subtitle file in their video player to actually see the subtitles.
- Use it with dvdauthor, to add the subtitles in a DVD. Read dvdauthor's documentation for more information.
- Incrust the subtitles into the video using mencoder. This command line is an example. Adapt the options to your needs:
  ```
  mencoder -sub <your_subtitle_file> <video_file_without_subtitles> -ovc lavc -lavcopts
  vcodec=mpeg4:vhq:vbitrate=1000 -oac mp3lame -lameopts br=256:vol=1 -ffourcc DIVX -o
  <converted_video.avi>
  ```

21.14 Creating DVD video from Lower Quality Footage

This section is for those who want to create PAL or NTSC DVD-format videos, using lower quality source footage such as that sourced on the net, or from cheap cameras such as SD-cams.

Typically, SD cameras produce progressive footage at framerates of 10-30fps at frame sizes like 640x480. Footage sourced online can have framerates as slow as 8fps and frame sizes as small as 320x240.

This section outlines a recipe for making the most of this limited quality footage, and minimising any further quality losses.
The steps we will follow are:

1. Up-sample the frame size
2. Convert to yuv4mpeg format
3. Up-sample the framerate with motion interpolation
4. Import into Cinelerra without loss
5. Interlace appropriately before DVD export

This technique requires that you have ffmpeg, mjpegtools and yuvmotionfps installed. You will likely already have ffmpeg and mjpegtools installed on your system. If not, you can get them from your distro feeds. But will need to get yourself a copy of yuvmotionfps from: http://jcornet.free.fr/linux/yuvmotionfps.html. yuvmotionfps is a great "poor man's" free/opensource equivalent of the 'Twixtor' plugin on Adobe Premiere.

We will perform steps 1-3 with 2 shell commands, and create temporary .yuv and .wav files ready to import into Cinelerra. Note too that we're assuming you want to create a PAL-DVD project, which is 25fps interlaced, 720x576 framesize. Convert these figures below to the NTSC framerate and framesize if you are creating an NTSC-DVD project.

Assume you have your source footage in the file `` `myfootage.avi' ``

The shell command to separate out the audio is:
```
ffmpeg -i myfootage.avi -f wav myfootage.wav
```

The shell command to separate out the video, upsample the framesize and framerate is:
```
ffmpeg -i myfootage.avi -s 720x576 -f yuv4mpegpipe -vcodec pgmyuv - | yuvmotionfps -r 50:1 >
myfootage.yuv
```

After executing both these commands, you should have separated audio and video files ready for Cinelerra.

Now, make sure that in your Cinelerra project options, you have set the framerate to 50fps. This is crucial, otherwise you will get a quality loss and jerky motion after rendering. (The sad thing is that this quality loss may not even show until you have mastered your DVD and are playing it to others - embarrassing). But with your project framerate at 50fps, you should be able to avoid this.

Now, import your new separated and converted video and audio files into Cinelerra. Apply effects as needed, such as colour corrections, zooms etc. Step through the frames, and verify that you see motion change with every frame. If you've got this, then you're on track.

Now, when you're ready to render, add one last effect to your video, and make sure this effect sits at the bottom of your effect stack. Select the entire duration of your video, and add the **Fields to frames** effect. Your footage is sitting within Cinelerra as 50fps progressive, and this effect will correctly convert it to 25fps interlaced. I'd suggest setting **Bottom fields first** initially, and changing this later if it doesn't play properly on your DVD player.

To render, I'd suggest you use the recipe on the Crazed Mule Productions website: http://crazedmuleproductions.blogspot.com/2007/06/beginners-guide-to-exporting-video-from.html#dvd But, contrary to this recipe, leave in the `-ilme -ildct` options. Depending on your ffmpeg version, you may need to change this to `-flags +ilme+ildct`

After this, you should end up with video that plays on a wide variety of consumer DVD players with good, flicker-free motion.

Tweaking

Here are some ideas for tweaks, in case you're getting less than perfect results:

- When initially separating audio/video with the ffmpeg command, cut out the frame size upsampling - remove the `-s 720x576` option - perform the size upsampling within Cinelerra using the camera/projector settings.
- In the final **Fields to frames** effect, switch between **Top fields first** and **Bottom fields first**
- Disable the **Fields to frames** effect in Cinelerra, and delay the reinterlacing to the render stage. Get the alternative version of yuvdeinterlace from http://silicontrip.net/~mark/lavtools/, and when rendering, put: `yuvdeinterlace -i -t |` before the `ffmpeg...` part of the yuv4mpegpipe shell command
- If the motion looks weird, try different settings in the yuvmotionfps command - do `yuvmotionfps -h` and look at the various options.

Warning

Before you release your DVD to anyone important, make sure to try it on as many consumer DVD appliances and TVs as possible. You don't want to release a DVD into the wild - for sale etc - and have it look like crap on your customer's TV. Even if it looks good on your DVD player and TV, and those of your friends, there could be some appliances out there that handle it badly.

Conclusion

With a small amount of experimentation, you should be able to import lower-quality video into Cinelerra, process it and render out to DVD-quality video and end up with the best overall quality that will look as good as possible on the greatest possible range of DVD players and TVs. Good luck!

[<<] [>>]   [Top] [Contents] [Index] [?]

This document was generated by *root* on *March, 8 2008* using *texi2html 1.76*.