Creating DVD videos on Linux and Unix

Ben Hutchings, July 2006

Introduction

The DVD video format is in large part proprietary and not at all widely understood; even decoding and playing it has proven to be legally problematic. I hope to provide a basic explanation of the format and to demonstrate free software that supports creation of widely compatible DVD videos.

Overview of the format

The DVD physical format is similar to the CD format, but with finer tracks read using a shorter wavelength of laser light, providing its higher capacity of 4.7 GB (where GB is 1 billion bytes). There's also the option for a second recording layer, raising the capacity to 8.5 GB, and double−sided discs.

As you may be aware, there are many types of DVD recordable media. The original DVD industry group, the DVD Forum, specified 3 of them. DVD−R and DVD−RW are the counterparts to CD−R and CD−RW. DVD−RAM is a more robust but incompatible rewritable format which cannot be used for standard DVD videos. Some manufacturers wanted to produce a more flexible rewritable format than DVD−RW while retaining compatibility with DVD players; they defined the DVD+RW format and for the sake of completeness also defined a rival one−time recordable format, DVD+R.

Unlike a CD, which is divided into tracks defined by a Table of Contents, a DVD is always expected to have a filesystem; this is normally UDF or its predecessor ISO 9660. Aside from decryption keys on DVD−ROMs and a small amount of low−level metadata on all DVD formats, the content is stored entirely within this filesystem.

DVD videos are logically divided into a first−play section, a set of top−level menus (called *VMGM* for Video Manager Menus) and one or more *video title−sets* (VTSs). Each title−set is logically divided into menus and *titles*, meaning the main content of the disc. Menus and titles each have MPEG−2 program streams that combine video, audio and *sub−pictures* − that is, subtitles and button images. Each set of menus is divided into language units and the player should automatically select the right one. Titles are not organised this way, but they may have multiple audio streams in different languages and codecs, and multiple sub−picture streams in different languages. They may also have sections with multiple video streams showing different "angles". In addition, the first−play section and each menu and title may have program routines to handle certain events.

Producing program streams

Video encoding and transcoding

The video streams must use the MPEG−1 or MPEG−2 video codec. MPEG−2 is more advanced and generally to be preferred, but at bit rates below 1 Mbit/s (which would allow you to fit about 10 hours on a single−layer disc) MPEG−1 may give better results.

There are two free encoders for both these codecs, found in *libavcodec* and *mjpegtools*. **`ffmpeg`** and **`mencoder`** are front−ends to libavcodec and mjpegtools provides **`mpeg2enc`**. **`transcode`** provides a front−end to both, and the *gstreamer* framework has a plugin that appears to use the mjpegtools encoder. Some video editors such as **`kino`** and **`avidemux`** can also use one of these encoders to export MPEG streams. Note that MPEG−2 at least is patent−encumbered and this can be an obstacle to distribution and use of encoders.

The encoding parameters are relatively constrained compared to those you might use for video streams intended for display on a computer. The frame rates and sizes must match one of the following supported options, depending on the codec used and the video standard that the target DVD players and TVs support, which in turn depends on the region of the world they were intended to be used in:

Analogue signal parameters	Region of world	Codec	Digital frame parameters			
			Rate (Hz)	Width	Height	Aspect ratio
59.94 fields per second, interlaced; 525 lines per frame (commonly called NTSC)	Americas except Argentina, Paraguay, Uruguay; Japan; Laos; Myanmar; Philippines; South Korea; Taiwan; some Pacific islands	MPEG−2	29.97 or 23.976 with 3−2 pulldown	720	480	16:9
				720	480	4:3
				704	480	4:3
				352	480	4:3
				352	240	4:3
		MPEG−1	29.97	352	240	4:3
50 fields per second, interlaced; 625 lines per frame (commonly but misleadingly called PAL)	Rest of world	MPEG−2	25	720	576	16:9
				720	576	4:3
				704	576	4:3
				352	576	4:3
				352	288	4:3
		MPEG−1	25	352	288	4:3

Frame aspect ratios other than 4:3 and 16:9 must be simulated by including black borders in the frame.

The MPEG video codecs rely on similarities between consecutive frames for much of their effectiveness, and represent most frames as the result of applying changes to earlier frames (or to later frames, which may appear ahead of time in the stream). Only some frames − *key* or *I* frames − are represented independently. When a viewer skips to a particular position, the player must find a key frame to start at; similarly in visual rewind or fast forward mode the player can generally only afford to read and show key frames. For this reason you must ensure that the interval between key frames, sometimes called the *GOP size* (for Group Of Pictures), is at most 0.6 seconds − 15 or 18 frames depending on the frame rate.

The bandwidth of the entire multiplexed stream is limited to 9.8 Mbit/s and the largest and most widely variable part of this is likely to be the video stream, so you must be sure to limit the video bit rate to avoid exceeding that.

mjpegtools appears to have some support for multi−angle video, but I do not know how to use this or whether it is supported by dvdauthor.

Example commands

1. Capture from a DV camera then convert to MPEG−2:

   ```
   dvgrab capture_20060604.avi
   ffmpeg −i capture_20060604.avi −target pal−dvd capture_20060604.mpeg
   ```

   The `−target pal−dvd` option to ffmpeg selects the MPEG−2 codec with "PAL" parameters for the maximum frame size and aspect of 4:3. You can override any of those if needed. The `−target ntsc−dvd` option would select the corresponding "NTSC" frame parameters. These also set the correct key−frame interval, a target average video bit rate of 6 Mbit/s and a maximum of 9 Mbit/s. They also set what should be appropriate audio and multiplexing settings, more about which later.

2. Convert phone−camera video to low bit−rate MPEG−1:

   ```
   ffmpeg −i phonecam.3gp −target pal−dvd −vcodec mpeg1video −s 352x288 −b 1000 \
       phonecam.mpeg
   ```

   Here we override the video codec, size and bit rate; ffmpeg will automatically resize as necessary. The input file will likely have a frame rate lower than 25 Hz but ffmpeg will duplicate frames as necessary. The file will probably use the AMR audio codec, which doesn't appear to have a free implementation yet, though source code is available and can optionally be compiled into a local version of ffmpeg. If you don't need audio you can discard it with the `−an` option.

3. Convert a static picture to MPEG−2 (using ffmpeg):

   ```
   ffmpeg −f image2 −vcodec png −i menu.png \
       −target pal−dvd −vcodec mpeg2video −an menu.mpeg
   ```

4. Convert a static picture to MPEG−2 (using mjpegtools):

   ```
   jpegtopnm menu.jpeg \
   | ppmtoy4m −n 1 −F 25:1 −A 59:54 −I p −S 420mpeg2 \
   | mpeg2enc −f 8 −a 2 −o menu.mp2v
   ```

   mpeg2enc and the other programs in *mjpegtools* are relatively difficult to use, but possibly more flexible. There is no resizing done here; the input picture must have the right size. If we wanted to use "NTSC" frame parameters we would replace the options `−F 25:1 −A 59:54` with `−F 30000:1001 −A 10:11`.

   Note that this produces an MPEG−2 video stream which needs to be multiplexed (described later) to produce a program stream.

Audio encoding and transcoding

DVD players are required to support AC−3 (Dolby Digital) and 16− and 24−bit PCM (uncompressed) codecs; those sold in European markets must also support MPEG−1 audio layer 2 (MP2). Audio streams must have a sample rate of 48 kHz or 96 kHz (for PCM only) and between 2 and 6 channels inclusive. The allowed bit rates depend on the codec. PCM obviously provides the best audio quality, but normally you should use either AC−3 or MP2 depending on whether you will have viewers outside Europe. I have tried using MP2 for speech at 96 kbit/s, which should be adequate, but found this to be incompatible with at least one DVD player, so I recommend using a minimum of 128 kbit/s.

Example commands

1. Convert to MPEG−2 with MP2 audio:

   ```
   ffmpeg −i capture_20060604.avi −target pal−dvd −acodec mp2 −ab 192 \
       capture_20060604.mpeg
   ```

   The `−target {ntsc,pal}−dvd` options select AC−3 as the audio codec, which we override with the `−acodec` option. We also override the audio bit−rate to 192 kbit/s.

2. Extract and compand audio before encoding:

```
ffmpeg −i capture_20060604.avi −f wav −vn −acodec pcm_s16le − \
| sox −t wav − −t wav capture_20060604.wav \
    compand 0.5,3 −70,−70,−50,−20,0,0 0 0 3
ffmpeg −i capture_20060604.avi −i capture_20060604.wav −target pal−dvd \
    −map 0:0 −map 1:0 capture_20060604.mpeg
```

   Here we try to compensate for variable sound levels from a camera−mounted mic by *companding* (compressing and expanding). First use ffmpeg to decode audio to a PCM (lossless) wave−file and pipe that into **sox**, which does the companding. Next we use ffmpeg to encode the video together with the modified audio. The `−map` options specify which streams it should use; the original audio, stream 0:1, is discarded.

Subtitles

Subtitles are encoded as streams of *sub−pictures*, which are simple 4−colour bitmaps. The colours are defined with an alpha channel so that the sub−pictures can be blended with the video behind them; typically subtitles are rendered as white text on a dark but semi−transparent background, with the remaining 2 colours used for anti−aliasing.

There are various simple text−based formats for subtitles, most of which can be converted to DVD sub−pictures by the program **spumux**. They can be created using **gaupol**, **ksubtile** and other programs.

I have not yet used subtitles, so I can only recommend that you read the documentation for these programs.

Multiplexing

The video, audio and sub−picture streams must be multiplexed together so that playback is not interrupted by the need for the player to seek between separate streams. Normally this will be done at the same time as encoding video and audio, but encoding and multiplexing may be done as separate steps. The multiplexer needs to be configured to produce a DVD−compatible program stream with space reserved for *NAV packets*; these provide information to support visual fast forward and rewind and are filled in at the next step by the DVD authoring software. In many MPEG multiplexers this is "mode 8". Most MPEG multiplexers do not support sub−pictures, so if you need them you must first multiplex video and audio streams and then pass the results through **spumux**.

Example commands

Multiplex video, audio and sub−pictures:

```
mplex −f 8 −o menu.mpeg menu.mp2v menu.mp2 \
| spumux −m dvd menu.spumux
```

Here `menu.spumux` is the control file for spumux specifying timings and sources for subtitles and sub−picture bitmaps.

Cells, program chains and chapters

The streams on disc are divided into cells, each of which is contiguous on the disc and in playback. The order in which cells are played back does not have to match the order on disc, and after each cell the player will run a program routine that may arbitrarily select the next cell. The normal ordering of cells for a menu or title is defined by one or more *program chains* (PGCs).

A PGC may also define *programs* as starting points within the order, and programs can be designated as *parts of title* (PTTs), more commonly known as chapters. The chapter divisions are meant to be useful to the viewer: most players can show the current chapter number, and title−set menus may contain commands to jump to a specific chapter within a title, but not to a specific cell. Each cell and therefore each chapter must begin with a key frame. You can ensure this by encoding the material for each chapter to a separate file.

Authoring

Authoring is an awkward term used to cover the compilation of the DVD file structure from streams and program code.

Programming model

DVD players implement a 16−bit virtual machine with 16 general−purpose registers, some system registers holding player status and viewer preferences, and no other memory. It has arithmetic and logic instructions, conditional and unconditional branches and register moves. There is no hardware stack or indirect jump instruction, therefore no support for subroutines so far as I can see! There are various specialised instructions for directing the player to some other logical division of the disc, though these are unfortunately restricted in non−obvious ways.

Program code is divided into routines of limited size placed in the various logical divisions of the DVD. General register contents are preserved between execution of these routines, and players that memorise the state of discs between uses also preserve general register contents.

Each PGC may have a pre−display routine that runs before the stream is displayed and/or a post−display routine that runs after the stream finishes. Each cell has a single instruction (possibly a no−op) that runs after it finishes.

Menus and buttons

As previously mentioned, menus, like titles, have MPEG−2 program streams. Menus can have up to 36 buttons at any one time, rendered as sub−pictures i.e. 4−colour bitmaps. The colours can change between the normal, *highlighted* (focussed) and *selected* (pressed) states of a button, but the same bitmaps are used.

These limitations may come as a surprise those who have previously seen highly coloured and animated buttons in commercial releases. These can be produced by careful coordination of the video stream and sub−pictures. Given that the colours of sub−pictures include an alpha channel, it is possible to put the basic button image in the video stream and to use the sub−picture merely as an overlay of lighting and shading on this. It is also possible to make a sub−picture entirely opaque in one state and entirely transparent, revealing the video stream, in another. Also, the sub−picture stream can of course have different sub−pictures for different points in time.

For each button, the menu has a single instruction to handle each of the Enter, Up, Down, Left and Right buttons being pressed while it is highlighted. Usually the directional buttons cause the highlight to move to another button in the given direction but this is not required. For example some discs provide extensive information in the menu system that can be "paged through" simply by pressing directional buttons.

As with subtitles, you should use **`spumux`** to include button images in the MPEG stream. spumux requires them to be rendered into PNG files in the positions they are to appear in the frame. It uses a separate file for the bitmaps for each button state, plus a control file specifying where the buttons are within these bitmaps; if you use transparent backgrounds then you can also tell it to find the edges of the buttons automatically, but this is somewhat less reliable. spumux works out the colour mapping automatically, though I'm not convinced it does a good job of this. If the buttons cannot be represented within the limitations explained above, it will report an error or possibly crash.

Putting it all together: dvdauthor and its front-ends

dvdauthor

This is a free command−line program for DVD video authoring, and to my knowledge the only such free program. With just a few commands it can create simple DVD videos with no menus, directing the player to start with the first title. For anything more complex, it requires the logical structure and program routines to be defined in a XML−based format. This format is extremely tiresome to write, but provides control over most of the features of the DVD video format. dvdauthor implements a language with syntax similar to C that it compiles to virtual machine instructions. Please refer to the dvdauthor documentation for the details.

There are many programs that provide an easier interface to dvdauthor and to the process of encoding menus and titles.

'Q' DVD−Author

This is a GUI for creating menus and titles, transcoding video and audio as necessary. It can create slideshows from static images and can include previews of titles in the menus that link to them. It provides dialogs for most of the advanced features of dvdauthor.

DVDStyler

This is a basic GUI for creating menus and defining chapters. The titles must be encoded beforehand. By default it runs dvdauthor, mkisofs and **`growisofs`** to produce a finished DVD, but can be set to stop at any stage in this process.

KDE DVD Authoring Wizard

Given encoded titles, this "wizard" guides the user to create menus according to a template with labels, previews of the titles, and optional background music. It then runs dvdauthor and **`mkisofs`** to create a finished "ISO" filesystem image.

kmediafactory

This GUI supports menu creation following various templates, transcoding of titles from any of many video formats and creation of slideshows from static pictures and various document formats. It can either run dvdauthor or else save a dvdauthor XML file for later use.

tovid

This is a set of command−line programs that call mencoder, mjpegtools and dvdauthor to produce DVD videos (or video CDs). They appear to simplify title encoding somewhat, and provide simple menu creation and authoring functionality.

WebDVD

I wanted to produce a DVD authoring tool that could be used from the command line while still providing a graphical interface where necessary. I considered DVDs to be a form of hypermedia, and felt that it ought to be possible to design them using the most popular hypermedia format available today − that is, HTML − since there are already a huge range of tools available for that. WebDVD uses the Mozilla browser to do all the layout and rendering work (guided by some important style rules). It currently only does menu creation and authoring, but I intend to add the capability to detect and transcode video files in an unsuitable format, and an option to create "ISO" images.

Example commands

1. Preview HTML pages as they will appear on DVD:

```
webdvd --preview main.html
```

   Here `main.html` is the main menu for the DVD. This displays pages in an undecorated browser window with the same size and stylesheet as will be used for the DVD. By default it uses "PAL" frame parameters; this can be overridden using the `--video-std` option.

2. Convert HTML pages and video to DVD:

```
webdvd main.html dvd
```

   Here `dvd` is the directory in which the DVD filesystem is to be created.

Creating and recording the filesystem

DVD videos use a UDF filesystem, but it must be laid out in such a way that a dumb player can read them without implementing much of the filesystem, and the metadata files include the offsets on disc of other files. Jörg Schilling's version of **`mkisofs`**, which is probably the most commonly used, has an option `-dvd-video` which causes it to lay out the filesystem appropriately and to fix up the offsets in the metadata files as they appear in its output. You **must** use this option to produce a working DVD video.

Dual-layer recordable discs are unfortunately relatively expensive and I do not have any experience of using them. I do know that a change of layers should be taken into account when authoring, to avoid an obvious pause in playback, but I don't know of free software to help with that. So for our purposes we need to use single-layer discs.

Recent DVD recorders can generally use all recordable formats, and recent DVD players can play all of them. Unfortunately, the various formats are distinguished by a so-called *book type* code included in the low-level metadata and some players and drives for computers will reject discs with an unrecognised book type. This particularly affects the DVD+R and +RW formats since they were not approved by the DVD Forum and their book types are unknown to older players. However, since the book type is in the recordable part of the disc, some drives allow you to record the book type as DVD-ROM. The *dvd+rw-tools* package includes a program to select this behaviour, and despite its name this package now supports all DVD recordable formats. I recommend you use this before recording DVD videos if you can; if your drive doesn't have this capability then it is probably best to avoid the "+" formats.

There are many programs for recording DVDs; I stick to the command line and use **`growisofs`** from the *dvd+rw-tools* package. This command can either record an "ISO" filesystem image or pass arguments through to **`mkisofs`** to create the filesystem on-the-fly.

Example commands

1. Make an "ISO" filesystem image of a DVD filesystem:

```
mkisofs -dvd-video dvd >dvd.iso
```

   `dvd` is the directory containing the filesystem produced by dvdauthor.

2. Record "ISO" filesystem image to disc:

```
growisofs -dvd-compat -Z /dev/dvd=dvd.iso
```

   The `-Z` option selects one-time recording or overwriting. growisofs can also record a new session (hence its name) but DVD videos must be recorded in a single session. The `-dvd-compat` option enables various recording features to improve compatibility.

3. Record DVD filesystem directly to disc:

```
growisofs -dvd-compat -Z /dev/dvd -dvd-video dvd
```

Index of software

Package	Program	Function	Home page
avidemux		video and audio editing and encoding	http://fixounet.free.fr/avidemux/
cdrtools	mkisofs	filesystem creation	http://cdrecord.berlios.de/
dvd+rw−tools	dvd+rw−booktype	improving disc compatibility	http://fy.chalmers.se/~appro/linux/DVD+RW/
	growisofs	filesystem creation and recording	
dvdauthor	dvdauthor	authoring	http://dvdauthor.sourceforge.net/
	spumux	sub−picture encoding and multiplexing	
DVDStyler	dvdstyler	front−end for menu creation and authoring	http://dvdstyler.sourceforge.net/
ffmpeg		video and audio encoding and multiplexing	http://ffmpeg.mplayerhq.hu/
gaupol		subtitle editing	http://home.gna.org/gaupol/
KDE DVD Authoring Wizard	DVDAuthorWizard.kmdr	front−end for menu creation and authoring	http://dvdauthorwizard.sourceforge.net/
kino		DV editing and encoding to various formats	http://www.kinodv.org/
kmediafactory		front−end for encoding, multiplexing and authoring	http://susku.pyhaselka.fi/damu/software/kmediafactory/
ksubtile		subtitle editing	http://ksubtile.sourceforge.net/
mjpegtools	mpeg2enc, ppmtoy4m	encoding to MPEG−1 and −2 video	http://mjpeg.sourceforge.net/
	mplex	multiplexing	
mplayer	mencoder	video and audio encoding and multiplexing	http://www.mplayerhq.hu/
'Q' DVD−Author	qdvdauthor	front−end for encoding, multiplexing and authoring	http://qdvdauthor.sourceforge.net/
sox		audio processing	http://sox.sourceforge.net
toolame		encoding to MPEG−1 audio layer 2	http://www.eftel.com/~mikecheng/planckenergy/
tovid	makedvd, makemenu, makexml	front−ends for menu creation and authoring	http://tovid.wikia.com/
	tovid	front−end for encoding and multiplexing	
WebDVD	webdvd	front−end for menu creation and authoring	http://womble.decadent.org.uk/software/webdvd/

Further reading

various authors, DVD authoring, Wikipedia, http://en.wikipedia.org/wiki/DVD_authoring

Dao, The Unofficial DVD Specification Guide, DVD−Replica Media, http://www.dvd−replica.com/DVD/orderdvd.php