DVDs rippen

Autor: Moritz Bunkus (*moritz@bunkus.org*)
Layout: Matthias Hagedorn (*matthias.hagedorn@selflinux.org*)

DVD-Rippen und -Wandlung mit Linux

Inhaltsverzeichnis

6.5 Quarter Pixel (QP)

1 Einführung

Da hast Du nun also die schöne neue DVD gekauft. Du findest sie absolut Klasse und möchtest sicher sein, dass, was immer auch passiert, Du auf jeden Fall eine Kopie dieses Films besitzt. Aber, zum Kuckuck, wie bekommst Du ihn von der großen DVD auf so eine kleine CD? Und wohlgemerkt in guter Qualität? Lies einfach weiter.

WARNUNG! Diese Anleitung ist nichts für das mutlose Gemüt. Sie umfasst Dinge wie das Kompilieren von Code, Installieren von Software und die Benutzung der Kommandozeile. Wenn Du nicht weißt, wie `configure`; `make`; `make install` auszuführen ist, dann lese zunächst andere Dokumentationen und lerne es. Wenn Du das nicht willst, dann installiere Windows und benutze eines jener guten DVD-Ripping-Programme, die es dafür gibt. Ich möchte keinen Flame-Krieg **Linux gegen Windows**. Es ist einfach so, dass dieser Vorgang unter Linux etwas schwieriger ist.

Du bist immer noch dabei? Wunderbar! Du bist dabei, eine Reise durch ein wunderschönes Land anzutreten...

1.1 Copyright

1.2 Notwendige Software

Zunächst möchte ich Dir einige Werkzeuge vorstellen, die Du benutzen wirst:

* `MPlayer` ist ein Movie-Player, der eine große Zahl von Dateiformaten und Media-Codecs unterstützt: MPEG1, AVIs (einschließlich aller Windows-Codecs und alle DivX-Codecs), DVDs, VCDs, SVCDs usw. Ich benutze ihn zum Betrachten aller Arten von Dateien.
* `MEncoder` ist Teil des MPlayer-Pakets und wird zum Kodieren von Video und Sound verwendet. MEncoder ist in der Lage, DVDs zu wandeln.
* `transcode` ist ein Satz von Werkzeugen, die es erlauben, eine lange Liste von Dateiformaten in eine lange Liste von anderen Dateiformaten zu wandeln, einschliesslich von DVD zu AVI oder (S)VCD
* Wie du weißt, sind die meisten DVDs verschlüsselt.libdvdcss ist eine Library, die DVD-Inhalte entschlüsselt.
* DVDs enthalten Informationen über Titel, Kapitel, Angles und Sprachen, die in den Dateien enthalten sind. Diese Informationen werden in .IFO-Dateien gespeichert. libdvdread wird häufig verwendet, um diese Dateien zu analysieren und die nötigen Informationen zu extrahieren.
* `lame` (Lame Ain't an MP3 Encoder) kodiert MP2/WAV-Audio in das MP3-Format.
* `vobcopy` kann dazu benutzt werden, VOBs von der DVD zu kopieren. Die Entschlüsselung geschieht dabei 'on the fly'.
* `dvd::rip` ist die grafische Benutzeroberfläche für `transcode`.

Ich werde Dir zwei Methoden zur DVD-Wandlung zeigen - eine unter Verwendung von `MEncoder` und eine unter Verwendung von `transcode`. Nichtsdestotrotz benötigst du für beide Methoden einige Dateien.

1.3 Ein Wort zu Codecs

In einer idealen Welt würde es nur ein Codec geben - das beste. Aber diese Welt ist weit vom Ideal entfernt. Das Ergebnis ist, dass es eine große Anzahl von Video- und Audio-Codecs zur Auswahl gibt. Zunächst werde ich die verschiedenen Arten von Codecs erklären:

* Native Codecs sind normale Linux-Binärdateien (meistens shared objects, libCODECNAME.so). Ihre

Unterstützung ist trivial.
* Windows-Codecs sind die originalen oder leicht modifizierten Windows dynamic libraries (DLLs, AX u.ä.) Diese Codecs funktionieren nicht von Haus aus unter Linux - Du benötigst eine spezielle Library, um diese Codecs verwenden zu können. Diese Library nennt sich avifile. Noch vor einigen Monaten war dies die einzige Möglichkeit, AVIs unter Linux anzusehen.

Heute erscheinen die meisten Codecs mit einer speziellen Linux-Version (wie z.B. XviD, DivX 4/5/) und andere.

1.3.1 Video Codecs

Hier ist eine Liste der unterstützten Video-Codecs

* **MPEG4** ist ein offizieller Video-Kompressions-Standard. Es gibt keinen Codec, dessen Name einfach MPEG4 ist. Wenn Du über MPEG4 sprichst, sprichst Du nicht über einen spezifischen Codec - Du sprichst über einen Sammlung von Techniken, Videos zu komprimieren. MPEG4-kompatibel meint, dass ein Codec Dateien erzeugt, die mit anderen MPEG4-kompatiblen abgespielt werden können.
* **DivX** ;-) ist der **original** gehackte Microsoft MPEG4-Codec, mit dem all diese **Ripperei** anfing. Er unterstützt nicht alle MPEG4-Features wie z.B. B-Frames oder global motion compensation GMC). Einige benutzen diesen Codec noch immer unter Windows mit dem exzellenten Werkzeug `Nandub`. Er wird unter Linux seltener benutzt, da es Codecs gibt, die eine bessere Qualität zur Verfügung stellen. Der Codec kommt als Windows dynamic library daher (divx.dll und einige andere) und benötigt unter Linux avifile.
* **DivX 4** und **5** sind die offiziellen Nachfolger. DivX 5 ersetzt DivX 4. Das ist der Grund, warum Du nicht beide Codecs zur gleichen Zeit installieren kannst (nun - technisch ist das möglich, aber Du solltest das aus ersichtlichen Gründen nicht tun). Dies sind native Linux shared objects - genannt libdivxdecore.so und libdivxencore.so. Die Sourcen dazu liegen nicht offen. Für Windows gibt es andere Versionen, und nur die kommerzuelle Version unterstützt sämtliche Optionen wie B-Frames, GMC oder Quarter PEL. Die kostenlose Version kann aber Dateien abspielen, auch solche, die mit der kommerziellen Version erstellt wurden. Leider gibt es nur eine kostenlose Version für Linux, die all diese netten Möglichkeiten nicht zur Verfügung stellt. DivXNetworks denkt darüber nach, auch für Linux eine kommerzielle Lösung zu Verfügung zu stellen. Aber erwarte nicht, dass dieser Codec in Naher Zukunft erscheinen wird.
* **XviD** (das ist DivX rückwärts gelesen) ist eine OpenSource MPEG4 Implementierung, die im Hinblick auf Kompression und Bildqualität wirklich gut ist. An der Unterstützung erweiterter MPEG4-Funktionen (b-frames , GMC) wird gearbeitet oder sind bereits implementiert.
* **libavcodec** oder kurz **lavc** ist ein weiterer Open Source MPEG4-kompatibler Video-Codec, der in Performance und Qualität DivX 5 und auch XviD überlegen ist. Dieser Codec unterstützt B-Frames. Er ist Teil des `ffmpeg`-Projekts.
* Natürlich gibt es MPEG1-kompatible Codecs, die Du für VCDs benötigst und MPEG2-kompatible Codecs für SVCDs oder DVDs. Im Moment zielt diese Anleitung auf die Erzeugung von AVIs, daher werde ich zu keinem dieser Codecs ins Detail gehen.
* Die meisten anderen Codecs sind entweder veraltet (wie Intel Indeo 5) oder andere gecrackte Version von Microsofts MPEG4-Codec wie AngelPotion). Es gibt weitere Codecs (wie VP4), die aber noch in der Entwicklungsphase sind und noch keine funktionierenden Lösungen für uns hier bieten.

Die Anleitung richtet ihr Hauptaugenmerk auf zwei Codecs: **XviD** und **lavc**. Die Gründe dafür sind, dass beide eine exzellente Qualität bieten, beide schnell sind und Du nur einen MPEG4-kompatiblen Decoder (wie DivX 5 oder XviD) auf einem Windowssystem zum Abspielen brauchst. Die Wiedergabe unter Linux ist überhaupt kein Problem - **MPlayer** oder **Xine** spielen fröhlich DivX 4/5, XviD und lavc-kodierte Filme.

1.3.2 Audio Codecs

Wieder eine Liste, dieses mal für Audio-Codecs:

* **MP3** ist die Kurzform für MPEG1 layer 3 und ist ein offizieller Kompressions-Standard. Wenn Du über MP3 sprichst, sprichst Du ber eine Kompressions-Technik, nicht über einen speziellen Codec. Es gibt mehr Codecs für

MP3 als ich an einem Tag aufzählen könnte.
* **lame** ist eine Abkürzung für **Lame Ain't an MP3 Encoder** (**Lame ist kein MP3-Encoder** - auch wenn es einer ist ;-)) `lame` bietet einen Encoder der der MP3-Dateien in sehr guter Qualität produziert. Sowohl `transcode` als auch `MEncoder` nutzen `lame` zur Audio-Kompression.
* **AC3** ist wieder ein offizieller Audio-Kompressions-Standard. Nahezu alle DVDs enthalten AC3-kodierte Audiospuren. Es gibt heute Dekoder sowohl für Windows als auch für Linux, die mit AC3-Ton innerhalb von AVIs arbeiten. Der Vorteil ist, dass keine erneute Komprimierung nötig ist (eine erneute Komprimierung ist immer mit einem Verlust an Qualität verbunden) und das Mehrkanal-Ton (Dolby Surround und ähnliches) erhalten bleibt. Der Nachteil ist, dass AC3-Ton mehr Speicherplatz als MP3-kodierter Ton benötigt.
* **Vorbis** ist ein neuer Open Source Audio-Kompressions-Codec. Er ist sowohl für Windows als auch Linux erhältlich. Die Vorteile sind ein besseres Qualität-zu-Größe-Verhältnis verglichen mit MP3 und Mehr-Kanal-Unterstützung. Leider kann man Vorbis-Audio nicht vernünftig in AVIs einbetten - aber ich arbeite an einem Programm, das Video und Vorbis-Audio zusammen in einer OGG-Datei speichern kann. Sobald dieses fertig ist werde ich es in diese Anleitung aufnehmen.

Ich hoffe, dies hat die Dinge ein bischen geklärt.

1.4 Allgemeine Libraries / Codecs

Ich werde in diesem Kapitel nicht all zu sehr ins Detail gehen. Es gibt in jedem Paket weitere Dokumentation zur Installation. Siehe dort, falls Du Probleme haben solltest. Beachte, dass `MPlayer` ziemlich umfangreiche Anforderungen bezüglich der Versionen von `gcc`, `binuntils` und anderen zentralen System-Komponenten hat. Siehe Die MPlayer-Installations-Dokumentation.

Beachte: Prüfe vor dem Download der Software, ob Deine Distribution bereits diese Software enthält. Ich verwende Debian Woody (testing) und habe festgestellt, dass sehr viel Software bereits als fertiges Paket vorhanden ist.

* libdvdcss - Gehe zu the Ogle project page und lade libdvdcss-0.0.3.ogle3.tar.gz herunter (die Versionsnummer könnte abweichen, da die Sofware sich in der Entwicklung befindet). Un-tar-gz sie, compiliere sie, and installiere sie unter /usr/local.
  Beachte das verschiedene Versionen dieser Library im Umlauf sind 0.0.3, 1.0.1). `MPlayer` empfiehlt immer noch **Version 0.0.3**.
* libdvdread - erhälst Du von der gleichen Website wie libdvdcss. Beachte das Debian Woody libdvdread bereits enthält. Ein einfaches `apt-get install libdvdread2 libdvdread2-dev` sollte genügen.
* **XviD-Codec** - wir gebraucht, wenn Du `transcode` benutzen willst.
  Eine kurze Anleitung aus der `MPlayer`-Dokumentation:
  * `cvs -z3 -d:pserver:anonymous@cvs.xvid.org:/xvid login` Wenn Du nach einem Passwort gefragt wirst, drücke einfach Enter
  * `cvs -z3 -d:pserver:anonymous@cvs.xvid.org:/xvid co xvidcore`
  * `cd xvidcore/build/generic`
  * Passe Makefile.linux Deinem System an.
  * `make -f Makefile.linux`
  * Kopiere die Datei xvidcore/src/divx4.h nach /usr/local/include/decore.h und nach /usr/local/include/encore2.h. Erstelle ein Backup der Dateien, die Du überschreibst!
  * Kopiere libxvidcore.so und libcore.a nach /usr/local/lib/

  Zum Compilieren brauchst du möglicherweise eine spezielle Version von `nasm`, die Du unter http://homepage1.nifty.com/herumi/soft/petit/nasmsse2.tgz bekommst.
* `vobcopy` - Erhälst Du von der Homepage http://linux-programming-newbie.org/.
* `avifile` - wie bereits erklärt wird es gebraucht, um Windows-Codecs zu verwenden, für die es keine linuxeigenen Binärdateien gibt. Gehe zur Homepage http://avifile.sourceforge.net/. Bevor Du das tust, prüfe, ob Deine Linux-Distribution nicht bereits `avifile` enthält (Debian Woody enthält `avifile`). Außerdem benötigst Du die Windows-Codecs von der `avifile`-Homepage oder direkt von der MPlayer's Homepage http://www2.mplayerhq.hu/MPlayer/releases/w32codec-0.60.tar.bz2.
* **DivX 5** - obwohl diese Anleitung sich nicht mit DivX 5 beschäftigt, kannst Du das Codec bei

http://www.divx.com/ erhalten. Falls Du ein wenig Zeit übrig hast, schlage ich vor, dass Du selbst einige Vergleiche zwischen DivX 5 und XviD oder lavc vornimmst und selbst siehst, dass es schlechter als die letzen beiden Codecs ist.

1.5 MPlayer

Du brauchst MPlayer. Es ist egal, ob Du später `MEncoder` oder `transcode` verwenden willst - Du brauchst in jedem Fall MPlayer. Also besorge MPlayer von http://www.mplayerhq.hu/. Ich bevorzuge die CVS-Version, da sie häufig Features besitzt, die dem offiziellen Release fehlen. Tatsächlich basiert diese Anleitung auf Features seit wenigen Tagen in der CVS-Version existieren (heute ist der 30. April 2002). Tu einfach das Gleiche - sei tapfer und nimm die CVS-Version.

Wenn Du `MEncoder` zum Kodieren verwenden möchtest, empfehle ich außerdem die Verwendung von libavcodec. Bei Verwendung der CVS-Version von MPlayer musst Du libavcode herunterladen, die Release-Version von MPlayer beinhaltet bereits libavcodec. Die Anweisungen sind aus der MPlayer-Dokumentation entnommen:

* Lade das **ffmpeg**-Project über CVS: `cvs -d:pserver:anonymous@cvs.ffmpeg.sourceforge.net:/cvsroot/ffmpeg login`. Drücke einfach ENTER, wenn Du nach einem Login oder Passwort gefragt wirst.
* `cvs -d:pserver:anonymous@cvs.ffmpeg.sourceforge.net:/cvsroot/ffmpeg co ffmpeg`
* Verschiebe das libavcodec-Verzeichnis von den FFmpeg-Sourcen in das Wurzelverzeichnis des MPlayer-CVS-Pfads. Symlink reicht nicht aus, Du musst das Verzeichnis bewegen oder kopieren.

Eine Bemerkung zum Kompilieren: Stelle sicher, nicht XviD support einzubinden. Stattdessen binde die Unterstützung für den libavcodec ein. Installations-Anweisungen stehen in der MPlayer-Dokumentation zur Verfügung. Ja, Du hast richtig gelesen: Kompiliere `transcode` mit XviD und MPlayer ohne XviD aber mit libavcodec. Der Grund dafür ist, dass MPlayer nicht für beide Codecs zur gleichen Zeit Unterstützung enthalten kann, da beide Codecs Variablen mit dem gleichen Namen verwenden.

1.6 Transcode

Besorge `transcode` von der Homepage. Stelle sicher, dass es mit XviD-Unterstützung und Unterstützung für MPlayer's post processing-Funnktionen compiliert wird. Nochmals: Ich empfehle die Verwendung der CVS-Version.

Außerdem willst Du `dvd::rip` herunterladen, eine exzellente, Gtk+ basierte Benutzerobefläche für `transcode`. Du bekommst sie hier. Installations-Anweisungen sind enthalten.

1.7 RPMs erhalten

Ich kenne zumindest zwei Adressen, an denen du einige RPMs für die oben erwähnte Software erhälst.

* Penguin Liberation Front lair http://plf.zarb.org/ - RPMs für Mandrake
* http://home.elka.pw.edu.pl/~dmierzej/mplayer.html - RPMs für Redhat 7.2

2 Rippen der VOBs

Dieser Teil ist ziemlich einfach. Alles was Du brauchst ist etwas freier Speicherplatz (eigentlich viel freien Speicherplatz). In Abhängigkeit davon, was Du wandeln willst, brauchst Du etwa 10GB freien Speicher. Solltest Du den nicht haben, investiere etwas Geld in ein neues Laufwerk, sie sind wirklich günstig zur Zeit (ca. 100 ? für ein 80GB-Laufwerk).

Bevor Du fortsetzt, mache Dir Gedanken über den Regional-Code. Alle heute erhältlichen Laufwerke haben den sogenannten RPC-mode-2-Schutz. Das bedeutet, dass Du nur fünf mal den Regional-Code ändern kannst, bevor er gesperrt wird. Du erhältst Informationen über das Entfernen des Regional-Codes Deines Laufwerkes bei Digital Digest. Warum also erwähne ich dies? Die Ripping-Methoden, die hier erläutert werden, beruhen auf einer korrekten Wahl des Regional-Codes (lies: der gleiche Code, den auch die eingelegte DVD beinhaltet) oder gar keinen Regional-Code. Alle diesbezüglichen Fehlermeldungen beim Rippen sind eher kryptisch, und oftmals ist es nicht sehr klar, dass es sich um ein Regional-Code-Problem handelt. Behalte dies also im Hinterkopf.

Teste zunächst die DVD. Starte MPlayer und schau Dir den ersten Titel an: `mplayer -dvd 1`. Gefällt es Dir? Cool. Stelle fest, ob das Video interlaced ist oder nicht (siehe unten). Schließe den MPlayer.

2.1 Interlaced Video

Was Du auf einem Fernseher siehst, ist nicht 25 Bilder pro Sekunde. In Wirklichkeit siehst Du 50 Halb-Bilder pro Sekunde. Ein Bild enthält die ungeraden Zeilen, ein Bild die geraden Zeilen. Auf diese Art und Weise siehst Du 25 Bilder pro Sekunde (frames per second=fps). Das Problem ist, das diese Halb-Bilder zu unterschiedlichen Zeiten aufgenommen sind. Wenn Du schnelle horizontale Bewegung in Deinem Film hast, wirst Du ein Objekt oder eine Person in den ungeraden Zeilen an einer Position und in den geraden Zeilen an einer anderen Position sehen. Das ist für das endgültige Video nicht erwünscht. Schau Dir dieses **interlaced** Bild

interlaced

an, es stammt von meiner Ally McBeal DVD. Mit den geeigneten Filtern kannst Du den diesen Interlacing-Effekt abstellen (genannt de-interlacing... was für eine überraschung ;)) Hier ist das gleiche Bild mit aktiviertem de-interlacing.

de-interlacing

Da Du nun eine Vorstellung davon besitzt, wie ein interlaced-Bild aussieht, solltest Du in der Lage sein, selbst zu entscheiden, ob Deine DVD Dateien mit oder ohne Interlacing enthält.

Bis zum Ende dieser Anleitung nehme ich folgendes an:

* Das DVD-Laufwerk ist verfügbar unter /dvd. Es gibt einen Eintrag in /etc/fstab für /dvd.
* Du besitzt genügend freien Speicher unter /space.
* Der DVD-Titel 1 ist derjenige, den Du kopieren möchtest (Robos, Autor von vobcopy, sagte mir, dass die meisten DVDs den Haupttitel unter 2 beinhalten. Dein Ergebnis kann anders sein.).

Bitte ersetze diese Pfade mit Pfaden, die zu Deinem System passen.

Es gibt viele Wege, die VOBs zu rippen. Ich werde zwei vorstellen: Benutzen des Kommandozeilen-Werkzeugs (`vobcopy`) und die Benutzung der `transcode`-GUI `dvd::rip`. Du musst nicht beides vornehmen :-)

2.2 Benutzung des Kommando-Zeilen-Interfaces

Starte `vobcopy`. Es wird automagisch den Inhalt der DVD auf Deine Festplatte kopieren.

* Als erstes binde Dein DVD ein: `mount /dvd`
* Dann lass `vobcopy` seine Arbeit tun: `vobcopy -i /dvd -m`

Dies wird die VOBs von der DVD kopieren, sie entschlüsseln (das ist die Stelle, an der libdvdcss benötigt wird) und sie in das aktuelle Verzeichnis schreiben (was Du z.B. mit `-o /space` ändern kannst). Die Dateinamen werden nach dem DVD-Titel benannt (z.B. habe ich ALLY_MCBEAL_DISC21-1.vob, ALLY_MCBEAL_DISC21-2.vob etc.). Die Option -m veranlasst `vobcopy`, die komplette DVD zu kopieren, inklusive der .IFO-Datein. Dies ist nützlich, da sowohl MPlayer/MEncoder als auch `transcode` die kopierten Dateien genauso wie eine echte DVD behandeln können. Das dauert seine Zeit. Sei einfach geduldig.

Bemerkung des Autors (Robos):

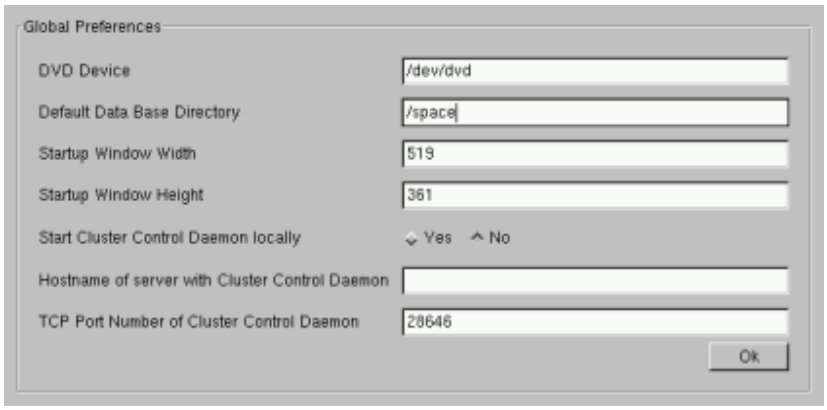
`vobcopy` hat einige Probleme beim Folgen von Angles. Möglicherweise erhälsts Du doppelte Szenen z.B. in **The Matrix**. Ich arbeite daran.

Noch eine Bemerkung von Robos:

Es gibt inzwischen ein neues Programm namens dvdbackup, das ebenfalls DVDs kopieren kann.

2.3 Benutzen von dvd::rip

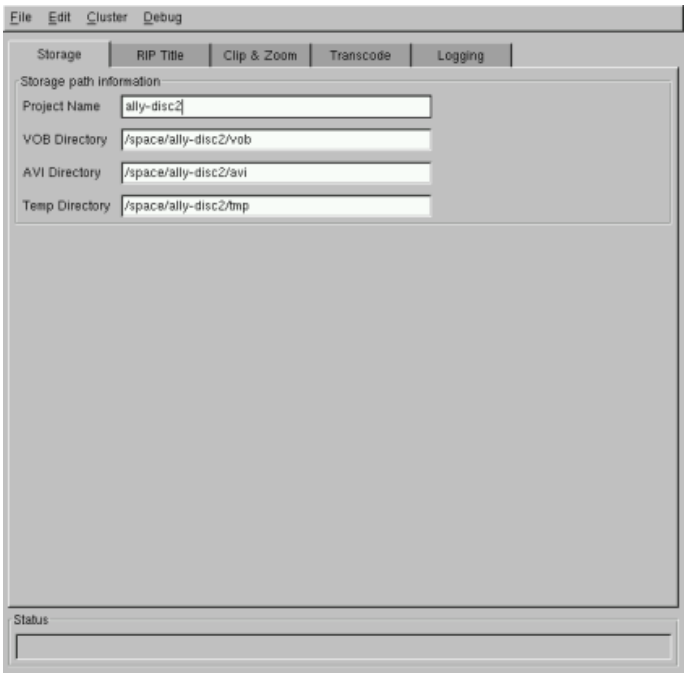
dvd::rip kann fast das gleiche für Dich tun. Starte dvd::rip durch Eingabe von `dvdrip`. Du siehst dann das Haupt-Fenster. Wähle Edit / Preferences und dvd::rip zeigt Dir den Voreinstellungen-Dialog.



Voreinstellungen-Dialog

Hier musst Du Deine Pfadangaben eingeben. Das erste ist der Pfad zum DVD-Gerät (device) und nicht zum mount point. Häufig ist dies /dev/dvd, welches ein symbolischer Link zu dem tatsächlichen Laufwerk sein kann, z.B. /dev/hdc.

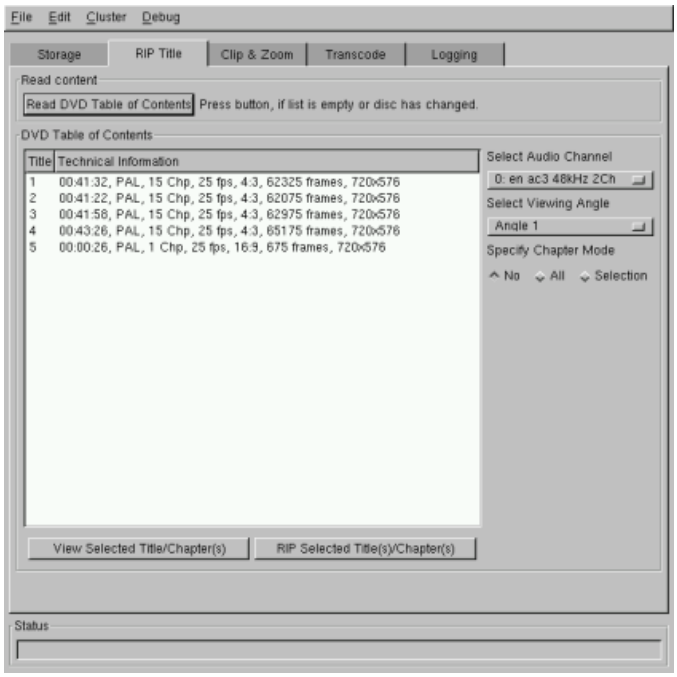
Schliesse den Dialog. Nun starte ein neues Projekt (File / New Project). Es beginnt mit der Speicher-Tabelle



Speicher-Tabelle

Gib wieder die korrekten Pfade ein. Beachte wie die anderen Namensfelder sich ändern, wenn Du den Projekt-Titel änderst.

Wechsel zur Rip Title tab



Rip Title tab

und klicke auf den Read DVD Table of Contents (lies DVD-Inhaltsverzeichnis)-Knopf. Nach ein oder zwei Sekunden wird die Liste darunter mit Titeln, die auf der DVD gespeichert sind, gefüllt, Wähle nun einfach die Titel, die Du rippen möchtest (Du kannst mehrere Titel durch Drücken von `Strg + klicken auf die Titel` auswählen) Wähle Sprache und Angle. Lasse Specify Chapter Mode auf No (Nein).
Letzter Schritt: Drücke Rip selected Title(s)/Chapter(s). Wiederum: Sei geduldig, trink etwas Milch, habe eine nette Unterhaltung mit Deiner Freundin.

Der Rest von dvd::rip wird später behandelt.

3 Wandlung der DVD in eine AVI-Datei

Hier hast Du die Wahl zwischen zwei Möglihckeiten - entweder `transcode` oder `MEncoder`. Beide haben Pro und Contras. Hier ist eine kurze Liste, die zumindest für die CVS-Versionen vom 28. April 2002 zutrifft.

* `MEncoder` kann keine Audio-Verzögerungen korrigieren. Falls Audio und Video nicht beim Abspielen mit MPlayer synchron laufen, solltest Du `transcode` stattdessen verwenden.
* `transcode` kann keine VBR- MP3, nur CBR-MP3 . Solltest Du wirklich VBR MP3 audio benötigen, dann bist Du an `MEncoder` gebunden. Beachte, dass die AVI tools, die mit `transcode` zusammen kommen, nicht mit AVIs, die VBR MP3 enthalten, arbeiten. Daher kannst Du keine sync-Probleme in von `MEncoder` erzeugten AVIs mit dem `avisync`-utility beheben. Gemäß der Dokumentation sollte `MEncoder` nur mit VBR/ABR MP3 Kodierung verwendet werden. CBR kodierte Dateien werden nicht unter Windows-Systemen laufen (während MPlayer sie problemlos abspielt)
* Es gibt im Moment noch keine GUI für `MEncoder`. Das macht Schneiden oder Größe verändern ziemlich mühselig.
* Siehe auch Glossar

3.1 Seitenverhältnisse

Bevor wir anfangen, möchte ich Dich in die meist gebräuchlichen Seitenverhältnisse, die Dir begegnen könnten, einführen. DVDs sind immer kodiert als 720x576 (5:4), obgleich das Bild beim Abspielen skaliert wird. übliche Video-Formate sind 4:3 = 1.33 für TV, 16:9 = 1.78 für normale Kinofilme und 2.35:1 für Cinemascope.

Hier eine Liste der üblichen Szenarien:

* **4:3 Filme**, die das gesamte Bild ausfüllen (kein letterboxing) : scale 5:4 to 4:3 16:9 Filme, die das gesamte Bild ausfüllen (wiedrum kein letterboxing): scale 5:4 to 16:9
* **16:9 Filme**, die "letterboxed" sind : scale 5:4 to 4:3 und schneide einen Teil der schwarzen Balken ab
* **2.35 Filme**, die "letterboxed" sind : scale 5:4 to 16:9 und schneidet einen Teil der schwarzen Balken ab

Keine Sorge, falls Du das nicht auf Anhieb verstehst. Schau dir dvd::rip's `clipping` und `scaling` Optionen für einen eher intuitiven überblick an.

3.2 Benutzung von dvd::rip

Die Benutzung von dvd::rip ist mit Abstand die beste und einfachste Methode Dein AVI-File zu erstellen. Wie bereits zuvor erwähnt, beruht es stark auf `transcode`, dass die eigentliche Aufgabe erledigt.

Nach dem Start von dvd::rip ripst Du die Titel, die Du wandeln möchtest auf Deine Festplatte.

3.2.1 Schneiden und Zoomen

Nun wirf einen Blick auf die Clip & Zoom

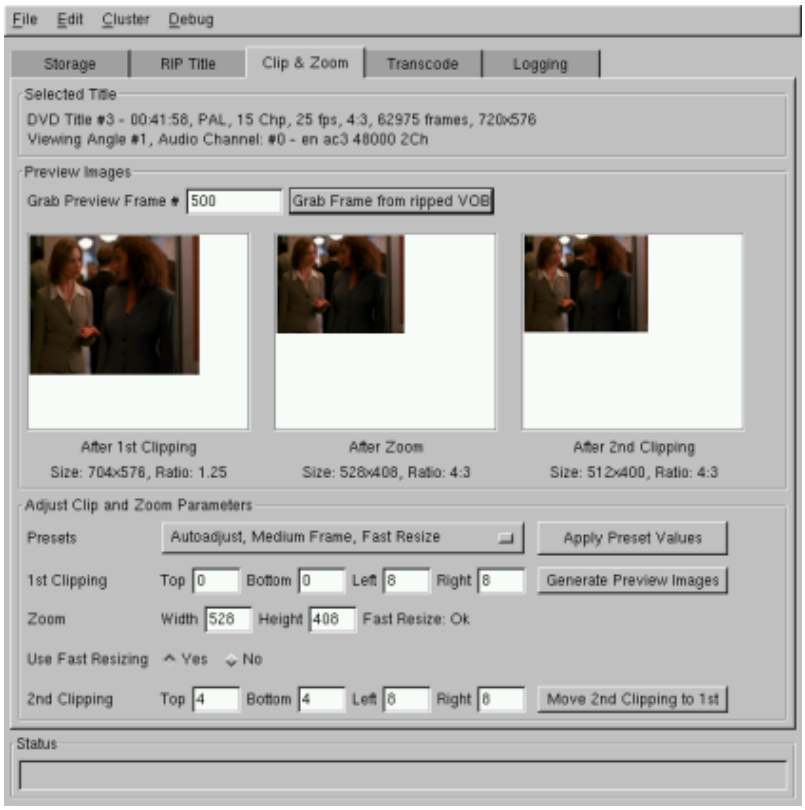


Tabelle. Hier kannst Du alle Parameter bezüglich Bildgröße und Ausschnitt einstellen. Zunächst hole ein Bild zum damit Arbeiten: Gib einfach eine Frame-Zahl (wie z.B. 200) in die Grab Preview Frame Eingabezeile und drücke auf Grab Frame from ripped VOB. Nach wenigen Sekunden zeigen die 3 Anzeigen darunter den angeforderten Frame.

`transcode` verwendet drei Schritte um das Bild zu seiner endgültigen Größe zu bringen: Erst schneidet es das nichtskalierte Bild, dann ändert es die Größe und schliessliche schneidet es nochmals. Jeder dieser Schritte kann ausgelassen werden; durch Auswahl von 0 für Clipping und durch die Original-Breite und Höhe für das ändern der Größe. Jedes Fenster zeigt das Ergebnis eines dieser drei Schritte. Wenn Du sehen willst, wie groß die Bilder tatsächlich sind, klicke einfach auf eines von ihnen und ein PopUp-Fenster öffnet sich mit dem Ergebnis.

Möglicherweise hast Du die drop-down-listbox unter den drei Vorschau-Fenstern bemerkt. dvd::rip biete eine große Zahl an Voreinstellungen zum Arbeiten an. Die einfache Auswahl eine dieser Voreinstellungen gibt Dir eine gute Startposition. Alle Voreinstellungen die **autoadjust...** heißen, versuchen die korrekte Schnitt-Region automatisch zu ermitteln. Nach der Auswahl einer Voreinstellung drücke den Apply Preset Values Knopf. Nun verstelle die Werte bis du mit dem Ergebnis zufrieden bist.

Die big frame, medium frame und small frame Teile beziehen sich auf die endgültige Bildgröße, die erreicht werden soll. big erhält fast die volle DVD-Auflösung, medium liegt irgendwo zwischen 500 und 600 für die Breite und small liegt irgendwo um 350.

Eine Anmerkung zum Verändern der Größe: Use fast resizing (benutze schnelle Größenänderung) hat großen Einfuß auf die Bildverarbeitung aber erfordert, dass die Ziel- Breite und Höhe durch 32 teilbar ist. Keine Sorge dvd::rip wird Dir sagen, wenn Du kein **Fast Resizing** nutzen kannst.
Beachte: Wenngleich es möglich ist, tatsächlich die Bilder zu vergrößern, rate ich stark davon ab. Es gibt mehrere Nachteile: Zum einen braucht der Encoder viel mehr Bits um die gleiche Qualität zu erreichen, die Du auch erreichst,

wenn Du ein kleineres Bild kodierst und dieses dann beim Abspielen vergrößerst, und zudem kann es auch zu Bildverzerrungen kommen (Peter Schuller berichtete mir von einem solchen Fall). Skaliere immer nach unten (=kleiner).

3.2.2 Wahl der Kodierungs-Parameters

Wir sind fertig mit dieser Tabelle. öffne die `Transcode`

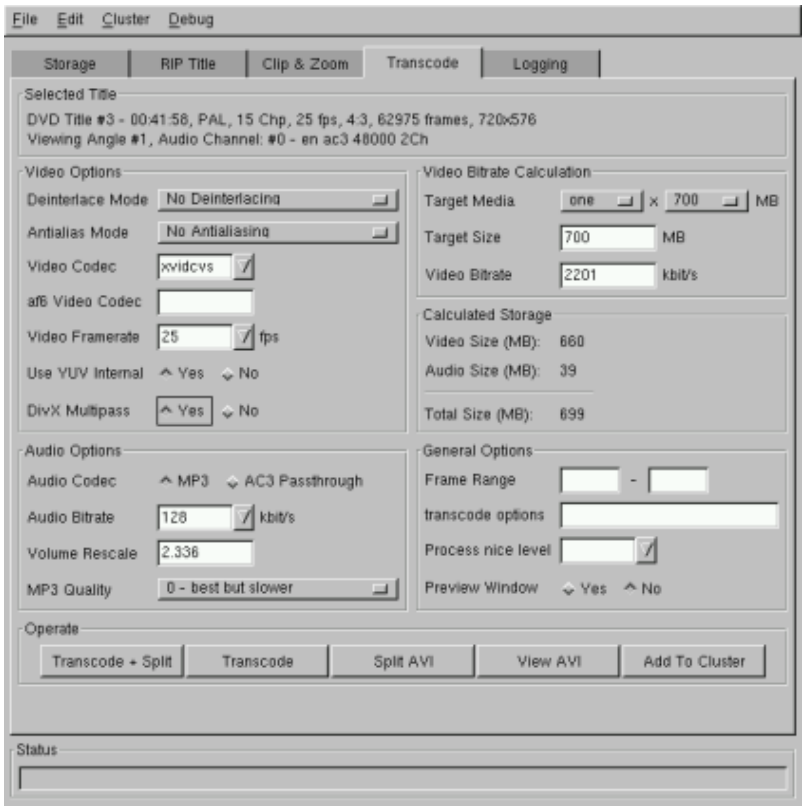


Tabelle. Entscheide zuerst welches Video-Codec Du verwenden möchtest. Wenn Du DivX5 für Linux installiert hast, kannst Du es verwenden, aber ich würde die Verwendung von XviD empfehlen. Wenn Du die CVS-Version von XviD heruntergeladen hast, dann ist dazu die Option `xvidcvs` (offensichtlich) zu wählen. Sollte `transcode` später mit einigen obskuren Fehlermeldungen abbrechen, kannst Du `xvid` ausprobieren.

Erinnerst Du Dich, dass ich Dich bat, zu prüfen, ob das Video interlaced ist? Jetzt brauchen wir diese Information. Falls Dein Video nicht interlaced ist, lasse einfach deinterlacing und antialias auf off. Andernfalls wähle 3 - Zoom To Full Frame. Dies ist der langsamste aber am besten aussehende Deinterlacer. Hier ist nochmal das Interlaced

Bild-Beispiel aus Ally McBeal. Du kannst das Ergebnis aus 3 - Zoom To Full Frame unter first deinterlaced picture

betrachten.

Ein weiterer De-Interlacing-Filter steht durch ein externes Plugin zur Verfügung (keine Sorge, Du hast es bereits beim Installieren von `transcode` selbst mitinstalliert). Schau nach unten rechts. Dort ist ein Eingabefeld mit der Bezeichnung `transcode` options. Alle hier eingegebenen Optionen werden einfach an `transcode` weitergegeben. Das können wir für das De-Interlacing nutzen. Hier ist das gleiche Bild wie oben deinterlaced mit einem anderen Deinterlacer

Dieser Deinterlacer ist schneller als die 3 - Zoom To Full Frame-Option. Wenn Du diesen Deinterlacer verwenden willst, dann setze deinterlace auf off und gebe `-J dilyuvmmx` in die Eingabezeile ein.

dvd::rip versucht die Bildrate automatisch zu ermitteln. Falls dies fehlschlägt kannst Du es hier korrigieren. Use YUV internal sollte immer auf yes stehen, es sei denn der Ausgabe-Codec unterstützt keine YUV-Modes. XviD unterstützt aber YUV. Es angeschaltet zu lassen gibt einen großen Geschwindigkeitsschub. DivX multipass sollte auch auf yes stehen, es sei denn, Du möchtest lediglich eine Vorschau. Für das endgültige AVI-File solltest Du immer 2-Pass-Kodierung verwenden. Auch wenn diese Option DivX multipass bezeichnet ist, arbeitet sie auch prima mit XviD

Die Audio-Optionen sollten selbsterkärend sein. Du wählst entweder die Tonspur in MP3 umzukodieren (unter Verwendung der angegebenen Bitrate) oder die Original AC3-Tonspur ohne Veränderung beizubehalten. AC3-Audio stellt Surround-Sound zur Vefügung und all diese Vorteile, aber benötigt mehr Platz als MP3-Sound. Es ist wirklich Deine Wahl. Wenn Du MP3 wählst, stelle sicher, dass die MP3 Quality auf 2 steht. Gemäß der `lame`-Dokumentation (`lame` wird zur MP3-Kompression verwendet) höhere Qualitätseinstellungen wie 1 oder 0 - beste aber langsamste sind viel langsamer aber erzeugen keine bezeichnend bessere Qualität. Volume rescale erlaubt die Normalisierung des Tons, die kein separates (externes) Programm benötigt. dvd::rip scant automatisch die Tonspur und stellt einen rescale-Wert zur Verfügung

Oben rechts befindet sich die Bitraten-Option. Du kannst einfach die Rip-Größe durch Wahl der Anzahl und Größe der CDs, die Du verwenden willst einstellen. Verändern der Target Size (Zielgröße) oder der Video Bitrate funktioniert auch. dvd::rip aktualisiert automatisch die errechneten Werte darunter, sodass es einfach ist, die optimale Bitrate zu ermitteln.

3.2.3 Erstellen einer schnellen Vorschau

In den meisten Fällen solltest Du dvd::rip eine kleine Vorabansicht erstellen lassen. Das tust Du duch Angabe eine Frame Range. Ich verwende meist ein 30 Sekunden-Sample, das entspricht 750 Bildern bei 25 fps (Anzahl der Sekunden * Bildrate = Anzahl der zu kodierenden Frames) und starte irgendwo im Film (so bei 200 oder 300 frames). Gute Werte könnten z.B. 200 - 950 sein. Du solltest außerdem `transcode` einen hohen nice-Wert (was einer niedrigen Prozess-Priorität entspricht) geben, sodass es nur freie CPU-Zeit verwendet. Wenn Du wirklich dein Preview jetzt und sofort haben möchtest, kannst Du auch vorübergehend DivX Multipass abschalten. Vergiss nicht, es später wieder einzuschalten!

Wenn Du fertig bist klicke auf `Transcode`. Nach einigen Minuten wird Dein AVI/MPEG fertig sein. Nun schau es

Dir an (z.B. mit `mplayer my-new-file.avi`) Dies ist ein guter Moment um zu sehen, ob deinterlacing wie erwartet arbeitet und um zu prüfen ob Audio und Video synchron laufen. Wenn ja, bist Du bereit. Andernfalls lies weiter:

3.2.4 Verhindern von Audio/Video Desynchronisation

Wenn Deine Audio- und Videospur bei der Vorschau nicht synchron zueinander abgespielt werden (oder auch beim Abspielen der DVD selbst), kannst Du `transcode` einen Frame-Offset für die A/V-Synchronisation geben. Das tust Du durch Angabe des Parameters `-D nr` in der `transcode` options Eingabezeile in der Transcode Tabelle. `nr` ist die Anzahl der Frames die die Audiospur verzögert ist. Diese Zahl kann auch negativ sein. Ein Frame ist 40ms lang bei 25 fps. Im Allgemeinen ist die Frame-Länge (1000 / fps) ms lang. Für meine Ally McBeal-DVD benötige ich eine Korrektur von -2, was -80ms entspricht: -D -2. Nun erstelle erneut eine Vorschau (die anderen Parameter sollten die gleichen bleiben) und prüfe die Tonspur erneut. Falls nötig wiederhole diesen Schritt bis Du mit dem Ergebnis zufrieden bist.

3.2.5 Das Ganze ausführen

Wenn die Vorschau gut ist, kannst Du mit der Wandlung beginnen. Stelle sicher, dass DivX Multipass wieder eingeschaltet ist, falls Du es für die Vorschau abgestellt hast. Klicke dann entweder auf `Transcode` oder auf `Transcode + Split` falls Du Deine Dateien automagisch nach den Angaben unter Video Bitrate Calculation geteilt haben möchtest.

Jetzt ist eine gute Zeit für soziale Kontakte :-)

3.2.6 Für fortgeschrittene Benutzer: transcode Kommandozeilen-Argumente

Dieses Kapitel erklärt die `transcode` Kommandozeilen-Optionen, die von dvd::rip genutzt werden. Dies ist nicht entscheidend für den Wandlungs-Prozess, Du kannst diesen Teil also auch überspringen. Ich schreibe ihn für diejenigen, die einen tieferen Einblick haben möchten, was dvd::rip und `transcode` tun.

In dvd::rip drücke `Strg + t`, um die Kommandos, die dvd::rip ausführt, zu sehen. Im Moment wollen wir uns auf die letzten Kommandos konzentrieren, `Transcode` command.
Hier ein Beispiel für meine Star Trek: The Next Generation DVD ohne die ganzen `mkdir` und `cd` Kommandos:

```
user@linux ~/ # transcode -i /space/tng-biggoodbye/vob/004 -w 4357,250,100 -a
1 -b 128,0,0 -s 3.311 -V -f 25 -B 12,10,8 -R 1 -x vob,null -o /dev/null -y
xvidcvs,null
user@linux / # transcode -i /space/tng-biggoodbye/vob/004 -w 4357,250,100 -a 1
-b 128,0,0 -s 3.311 -V -f 25 -B 12,10,8 -R 2 -x vob -o
/space/tng-biggoodbye/avi/004/tng-biggoodbye-004.avi -y xvidcvs
```

Ein Blick auf das erste Kommando und seine Parameter:

* `-i /space/tng-biggoodbye/vob/004` teilt offensichtlich `transcode` mit, wo es die Dateien findet. Das kann eine Datei, ein Gerät oder ein Verzeichnis mit Dateien sein.
* `-w 4357,250,100` setzt die Video-Kodierungs-Parameter: Bitrate, maximales Keyframe-Interval und Crispness.
* `-a 1` wählt Audio-Track Nummer 1 (beginnend mit 0).
* `-b 128,0,0` setzt die Audio-Kodierungs-Parameter für lame: bit rate, vbr und Qualität. Siehe auch lame's man page für eine Erklärung der Parameter -V and -q.
* `-s 3.311` veranlasst lame die Tonspur zu verstärken und damit **on the fly** zu normalisieren
* `-V` veranlasst `transcode` die Bildverarbeitung im YUV Farbraum vorzunehmen. Ohne `-V` würden die Bilder in den RGB Farbraum konvertiert werden. Beachte, dass einige externe Filter einen der beiden Farbräume voraussetzen. YUV processing erhöht die Geschwindigkeit stark.

* `-f 25` setzt die Bildrate.
* `-B 12,10,8` setzt die fast scaling (schnelle Skalierung) Optionen: das Bild wird herunterskalier auf Höhe - 12 * 8 Zeilen und auf Breite - 10 * 8 Spalten.
* `-R 1` ist der Marker für den ersten (von zwei) Kodierungs- Durchgang
* `-x vob,null` - Die Video Eingabe kommt aus VOB-Dateien und die Audio-Eingabe wird übersprungen (sie ist im ersten Durchgang ohnehin unnötig)
* `-o /dev/null` - Wir brauchen die Ausgabe des ersten Durchgangs nicht.
* `-y xvidcvs,null` - Ausgabe Video unter Verwendung von XviD und ohne audio.

Die zweite Kommandozeile weicht nicht so sehr davon ab. Sie überspringt alle Optionen die die Ausgabe abschalten würden (wie `-o real-file-name` und `-y xvidcvs`). Für eine vollständige Kommandoreferenz siehe die `transcode` und `lame` man pages.

Eine letzte Anmerkung: Du wirst sehen, dass `transcode` seit Version 0.6.0pre6-20020529 Unterstützung für das libavcodec über ein von mir geschriebenes noch expermentelles Export-Modul enthält. Du kannst dies aktivieren durch Verwendung von `-y ffmpeg4` (zur Erinnerung: libavcodec ist Teil von ffmpeg) anstelle des vorherigen Codecs. dvd::rip unterstützt dies zur Zeit noch nicht (lies: der Codec ist nicht in der Liste der auswählbaren Codecs).

3.3 Benutzung von MEncoder

Wie zuvor erwähnt, gibt es im Moment noch keine GUI für den MEncoder. Daher werde ich keine Screenshots zeigen und Du musst alles von Hand machen. Es ist ein Prozess der **3-Pass-Kodierung** genannt wird.

3.3.1 Kodierung der Tonspur

Der erste Schritt ist die Kodierung der Tonspur. Stelle sicher, dass du weisst, welche Sprache Du verwenden willst - Du benötigst entweder die Audio ID (d.h. 128 für den ersten Stream, 129 für den zweiten, usw.) oder den zwei-Buchstaben-Länder-Code (wie **en** für Englisch oder **de** für Deutsch). Du kannst diese Codes durch Ausführen von MPlayer im Verbose-Mode ermitteln: `mplayer -dvd 1 -v`. Das Programm sollte eine Menge Zeilen ausgeben. Suche nach Zeilen die diesen ähneln:
[open] audio stream: 0 audio format: ac3 language: en aid: 128
[open] audio stream: 1 audio format: ac3 language: de aid: 129
[open] audio stream: 2 audio format: ac3 language: es aid: 130
Hier habe ich drei Audio-Streams: Englisch, Deutsch, Spanisch und ihre IDs

Rufe nun MEncoder auf, um die Tonspur für dich zu kodieren

```
user@linux / # nice -+19 cat /space/*vob | nice -+19 mencoder -ovc frameno -o
frameno.avi -oac mp3lame -lameopts abr:br=128 -alang de -
```

Hier ist eine Erklärung der verwendeten Kommandozeilen-Argumente:

* `nice -+19`
  * Gibt MEncoder die niedrigste Prozess-Priorität, sodass er die normale Arbeit nicht stört
* `-ovc frameno`
  * Ausgabe-Video-Codec ist frameno was bedeutet, dass gar keine Video-Daten geschrieben oder verarbeitet werden.
* `-o frameno.avi`
  * Schreibe die Ausgabe in eine Datei mit Namen frameno.avi. Die Tonspur muss in eine Datei mit diesem Namen geschrieben werden, da MEncoder die Tonspur genau aus dieser Datei im nächsten Schritt lesen wird.
* `-oac mp3lame`

    * Ausgabe-Audio-Codec ist die lame encoder library.
* `-lameopts abr:br=128`
    * Optionen für `lame`. Veranlasst `lame` eine Datei unter Verwendung von ABR (average bitrate (durchschnittliche Bitrate, eine variable Bitrate mit dem vom Benutzer angegeben Durchschnitt) mit einer durchschnittlichen Bitrate von 128kbits/s zu erzeugen.
* `-alang de`
    * verwendet die deutsche Tonspur. Alternativ könnte `-aid 129` verwendet werden.

Warte eine Weile und Du hast Deine Tonspur. MEncoder gibt auch einige Vorschläge für die Video-Bitrate an:
Recommended video bitrate for 650MB CD: 1845
Recommended video bitrate for 700MB CD: 1992
Recommended video bitrate for 800MB CD: 2287
Wie Du erkennst, sind diese Werte ziemlich hoch - aber das ist darauf zurückzuführen, dass eine Ally-McBeal-Folge nur 41 Minuten lang ist. Also würde ich zwei Episoden pro CD erzeugen und dem Video eine Bitrate von etwa 1000 geben.

3.3.2 Erstellen einer Vorabansicht (Erster Durchgang)

Nun ist es Zeit sich zu entscheiden, welcher Video-Codec verwendet werden soll. Ich nehme an, dass eine AVI-Datei mit einem der verschiedenen DivX-Encoder erstellt werden soll. MEncoder unterstützt DivX 4 oder DivX 5 (das gilt auch für Windows - Du hast entweder v4 oder v5 installiert) genauso wie XviD oder lavc. Obwohl die meisten von Euch nichts von libavcodec oder dem ffmpeg-projekt zuvor gehört haben, sei Dir angeraten dass `lavc` den Codecs DivX 4 or 5 weit überlegen ist und mindestens so gut ist wie XviD. Daher will ich nur Beispiele für `lavc` zur Verfügung stellen, zumal Du dafür keine speziellen Codecs zum späteren Abspielen benötigst. MPlayer spielt sie (natürlich) prima ab, und für Windows benötigst Du einfach DivX 5 (die kostenlose Ausgabe ist absolut ausreichend). Auf geht's.

Wir wollen nun die Kommandozeilen-Optionen zusammenstellen:

* `-oac copy`
    * MEncoder braucht die Informationen des ersten Durchgangs, um Audio & Video synchron zu halten. Du darfst auf keinen fall `-nosound` benutzen, auch wenn ich das in früheren Versionen dieser Anleitung empfohlen habe. Wenn dich die technischen Details dazu interessieren, dann schau bei der MPlayer-Mailliste nach. `-o /dev/null` - Wir brauchen auch die Ausgabe des ersten Durchgangs nicht.
* `-ovc lavc`
    * wählt `lavc` als Ausgabe-Video-Codec
* `-lavcopts vcodec=mpeg4:vbitrate=1000:vhq:vqmin=1:vqmax=31:vpass=1`
    * libavcodec unterstützt tatsächlic eine handvoll verschiedener Video-Codecs. Mit `vcodec=mpeg4` wählen wir den MPEG4-kompatiblen Encoder. `vbitrate=1000` ist die gewünschte Bitrate. `vhq` (very high quality / sehr hohe Qualität) veranlasst den Kodierer eine Menge Zeit mit der Optimierung des Ergebnisses zu verbringen. Es ist den Aufwand wert. `vpass=1` teilt schliesslich dem Encoder mit, dass dies nur der erste Durchgang ist. Die Benutzung von `vqmin` und `vqmax` teilt dem Codec mit, wie stark das Video mindestens und maximal komprimiert werden soll. Diese Werte auf ihren Voreinstellungen zu lassen (siehe (see man mencoder) beschränkt die erreichbare Ausgabe-Bitrate sehr, daher empfehle ich eine größere Reichweite.
* `-vop scale=640:480`
    * Skaliert das Bild herunter auf 640x480 Pixel. Ich habe in der Mailing-Liste gelesen dass es einen neuen Parameter gibt, der automatisch die Höhe von der angegebenen Breite und umgekehrt errechnet, sodass das Seitenverhältnis erhalten bleibt. Ich habe es selbst nicht probiert, aber das sollte so aussehen: `-vop scale -xy 640` . Beachte wenngleich es möglich ist, tatsächlich die Bilder zu vergrößern, rate ich stark davon ab. Es

gibt mehrere Nachteile: Zum einen braucht der Encoder viel mehr Bits um die gleiche Qualität zu erreichen, die Du auch erreichst, wenn Du ein kleineres Bild kodierst und dieses dann beim Abspielen vergrößerst, und zudem kann es auch zu Bildverzerrungen kommen (Peter Schuller berichtete mir von einem solchen Fall). Skaliere immer nach unten (=kleiner).

* Falls Dein Bild interlaced ist, kannst Du es durch Angabe des Parameters `-npp lb` de-interlaced ausgeben.
* Wir möchten nur eine Vorabansicht, daher geben wir eine Startposition und die gewünschte Länge mit an: `-ss 0:20` bezeichnet 20 Sekunden nach Filmbeginn als Anfang, und `-endpos 0:30` sagt, dass wir 30 Sekunden bearbeiten wollen. Dieser Parameter wurde ungünstig benannt, da es sich nicht um die End-Position, sondern um die Länge des zu kodierenden Ausschnitts handelt.

Füge die gesamte Kommandozeile zusammen:

```
user@linux ~/ # nice -+19 mencoder -nosound -o /dev/null -ovc lavc -lavcopts
vcodec=mpeg4:vbitrate=1000:vhq:vpass=1 -vop scale=640:480 -npp lb -ss 0:20
-endpos 0:30 /space/first.vob
```

3.3.3 Erstellen einer Vorabansicht (zweiter Durchgang)

Für dieses Schritt können wir den größten Teil der vorhergehenden Kommandozeile beibehalten. Natürlich ändern wir `vpass=1` in `vpass=2` um dem Encoder mitzuteilen, dass es diesmal der zweite Durchgang ist. Natürlich sollten wir diesmal die endgültige Ausgabe speichern und ersetzen daher `-o /dev/null` mit etwas Brauchbarem, z.B. `-o ally-preview.avi`.

Hier die Kommandozeile:

```
user@linux ~/ # nice -+19 mencoder -oac copy -o ally-preview.avi -ovc lavc
-lavcopts vcodec=mpeg4:vbitrate=1000:vhq:vpass=2 -vop scale=640:480 -npp lb
-ss 0:20 -endpos 0:30 /space/first.vob
```

Nachdem MEncoder fertig ist, sehen wir uns die Vorabansicht an. Berichtige, sofern nötig, die Einstellungen. Wiederhole die Erstellung der Vorabansicht, bis Du zufrieden damit bist.

3.3.4 Erstellen der endgültigen AVI-Datei (beide Durchgänge)

Die Aufrufe der endgültigen Optionen lassen einfach die Optionen `-ss` und `-endpos` weg. Unglücklicherweise kann MEncoder nicht mehrere Eingabe-Dateien von der Kommandozeile lesen. Daher verwenden wir das `cat`-Kommando. Gib einfach

```
user@linux ~/ # cat /space/*vob | nice -+19 mencoder -oac copy -o /dev/null
-ovc lavc -lavcopts vcodec=mpeg4:vbitrate=1000:vhq:vpass=1:vqmin=1:vqmax=31
-vop scale=640:480 -npp lb -
user@linux / # cat /space/*vob | nice -+19 mencoder -oac copy -o ally.avi -ovc
lavc -lavcopts vcodec=mpeg4:vbitrate=1000:vhq:vpass=2:vqmin=1:vqmax=31 -vop
scale=640:480 -npp lb -
```

ein.
Beachte, dass `-ss` scheinbar nicht zusammen mit der `cat *vob mencoder...`-Variante funktioniert. Also muß für die Vorschau einfach die erste VOB-Datei als letztes Argument angegeben werden (siehe oben)

Nun geh und hol Dir ein Bier und ein gutes Buch.

4 Zusätzliche Tricks

Es gibt einige weitere Themen bezüglich der Video-Produktion.

4.1 Reparieren einer nichtsynchronen Audio-Spur

Dies funktioniert nicht mit AVIs die VBR/ABR MP3s verwenden. Dieser Schritt benutzt `avisync`, welches Teil des `transcode` Pakets ist.

`avisync` ist leicht zu benutzen: gib einfach eine Eingabe-Datei (`-i name.avi`), eine Ausgabe-Datei (`-o newname.avi`) und den Sync-Offset in Frames (`-n count`) an. Aus der `avisync`-Hilfe: `count>0`: Audio startet mit frame 'count'. count<0: stelle 'count' audio frames voran. Ein Beispiel:

```
user@linux ~/ # avisync -i ally-not-synched.avi -o ally-synched.avi -n -2
```

4.2 Zerteilen von AVIs

Dies funktioniert nicht mit AVIs, die VBR/ABR MP3s benutzen. Dieser Schritt benutzt `avisplit`, welches Teil des `transcode`-Paketes ist. Benutze es nur mit AVIs, die von `transcode` erzeugt wurden.

`avisplit` ist leicht zu benutzen: Gib einfach eine Eingabe-Datei (`-i name.avi`) an und an welcher Stelle die Datei aufgeteilt werden soll. Du kannst die Datei nach einer festen Größe spalten: (`-s Größe_in_Megabyte`), nach einer Anzahl von Frames (`-f f1-f2`) oder nach einer Anzahl von Sekunden `-t s1-s2`). Ein Beispiel:

```
user@linux ~/ # avisplit -i ally-big-file.avi -o ally -s 100
```

Dies erzeugt Dateien mit den Namen ally-0000.avi, ally-0001.avi etc., die jeweils maximal eine Größe von 100 MByte haben.

4.3 Extrahieren des Tons aus VOBs MPEGs AVIs

MPlayer kann leicht dazu verwendet werden, den Ton zu extrahieren. Die Option -vo null veranlasst MPlayer, die Videospur nicht auszugeben.

* `VOB -> AC3: mplayer -vo null -nogui -alang de -dumpaudio -dumpfile lang-de.ac3 *vob`
  Dies extrahiert die deutsche Tonspur in eine Datei namens lang-de.ac3.
* `VOB -> WAV: mplayer -vo null -nogui -aid 128 -ao pcm -aofile lang-de.wav *vob`
  Dies extrahiert die Tonspur Nr. 128, mischt den AC3-Ton herunter in PCM und schreibt das Ergebnis in lang-de.wav.
* `MPG -> MP2: mplayer -vo null -nogui -dumpaudio -dumpfile audio.mp2 myvideo.mpg`
  Dies extrahiert die Tonspur in eine Datei namens audio.mp2.
* `AVI -> WAV: mplayer -vo null -nogui -ao pcm -aofile audio.wav myvideo.avi`
  Die extrahiert die Tonspur, konvertiert sie in PCM und schreibt die entstehende .WAV in audio.wav.

Ich liste nicht alle möglichen Optionen auf. Im Allgemeinen verwende `-dumpaudio -dumpfile newaudio.extension` wenn du die Tonspur ohne Bearbeitung extrahieren willst und `-ao pcm -aofile newaudio.wav` Wenn Du die Datei als .WAV benötigst.

4.4 Benutzung separater Audio-Dateien

Manchmal willst Du eine andere Ton-Datei zusammen mit Deinem Video abspielen. MPlayer macht dies ziemlich einfach: `mplayer -audiofile anotherlang.mp3 myvideo.avi` spielt das Video aus myvideo.avi und den Ton aus anotherlang.mp3. Eine Menge Windows media player unterstützen ebenfalls externe Audio-Dateien. Auf diese Art und Weise kannst Du mehrere Sprach-Versionen als separate Audio-Dateien verfügbar machen. Eine häufig verwendete Kombination ist die Verwendung des englischen Audio-Tracks im AVI-File und die zusätzliche Erzeugung anderer Audio-Tracks als separate Files.

4.5 Ein grafischer AVI-Editor: avidemux

Wenn Du schon einmal unter Windows Videobearbeitung gemacht hast, dann kennst Du wahrscheinlich auch den exzellente OpenSource-Videoeditor VirtualDub. Ein OpenSource-Werkzeug für Linux, das darauf abzielt, dem Benutzer eine ähnliche Funktionsvielfalt wie `VirtualDub` zu bieten, ist avidemux. Damit kannst du ganz einfach bestimmte Teile deines Films herausschneiden, verschiedene Filter anwenden, Filme erneut enkodieren und noch eine ganze menge mehr machen. Ich empfehle Dir, dass Du es Dir einfach einmal anschaust, auch wenn die Benutzung der Kommandozeile für Dich so natürlich ist wie sie es für mich ist.

5 Links

5.1 Libraries, codecs

* libdvdread and libdvdcss:
  http://www.dtek.chalmers.se/groups/dvd/downloads.shtml
* `DivX 5`:
  http://www.divx.com/
* `XviD`:
  http://www.xvid.org/
* `ffmpeg` (the home of libavcodec):
  http://ffmpeg.sourceforge.net/

5.2 Media players

* **MPlayer**:
  http://www.mplayerhq.hu/
* **Xine**:
  http://xine.sourceforge.net/
* **Ogle**:
  http://www.dtek.chalmers.se/groups/dvd/index.shtml

5.3 Media encoders

* `MEncoder` (part of MPlayer):
  http://www.mplayerhq.hu/
* `transcode`:
  http://www.theorie.physik.uni-goettingen.de/~ostreich/transcode/
* `dvd::rip` (GUI for `transcode`):
  http://www.exit1.org/dvdrip/
* some Perl scripts for `transcode`:
  http://www9.informatik.uni-erlangen.de/~Vogelgsang/bp/tctools.html

5.4 Andere Dokumentation

Sowohl `MPlayer` als auch `transcode` bringen normale Unix-man-pages mit: `man mplayer`, `man mencoder` und `man transcode` zeigen diese Seiten. Zusätzlich liefert die Angabe von `-h` oder `--help` als einzigem Parameter zu einem dieser Programme eine Liste der Kommandozeilenoptionen (die allerdings evtl. nicht komplett ist).

MPlayer-Dokumentation:
http://www.mplayerhq.hu/DOCS/ transcode-Dokumentation und Beispiele:
http://www.theorie.physik.uni-goettingen.de/~ostreich/transcode/examples.html#top

* Der beste Ort im Netz für DVD ripping und Wandlung ist doom9. Obwohl sich die Seite auf Windows konzentriert, liefert sie eine Menge Einblicke zu DVD und Codecs im Allgemeinen. Auch das Forum ist exzellent. Wenn du wirklich an Videoverarbeitung interssiert bist, dann ist dies eine Seite für dich.

5.5 Mail Listen

Ich empfehle sehr, dass Du die Mailing-Listen nach Informationen durchsuchst/durchstöberst und sie Dir selbst abonnierst. Eine Menge Informationen sind dort verfügbar, die nicht in der Haupt-Dokumentation oder der

Kommandozeilen-Referenz auftauchen.

* `MPlayer's` user mailing list:
  http://mplayerhq.hu/mailman/listinfo/mplayer-users
  `MPlayer's`developer mailing list:
  http://mplayerhq.hu/mailman/listinfo/mplayer-dev-eng
* `transcode`'s user mailing list:
  http://www.theorie.physik.uni-goettingen.de/pipermail/transcode-users/
* `dvd::rip's` mailing list:
  http://www.exit1.org/dvdrip/list.cipp

6 Glossar

6.1 CBR, VBR und ABR

Diese drei Abkürzungen stehen für Constant Bit Rate, Variable Bit Rate und Average (=durchschnittliche) Bit Rate.

Mit **CBR** verwendet der Encoder die gleiche Anzahl von Bits in jedem Frame, unabhängig davon, wieviele Bits er tatsächlich bräuchte. Dies hat den Nachteil, das z.B. ein komplett schwarzes Bild oder absolute Stille in einem Audio-Frame eine Menge Bits veschwendet, während Bilder oder Audio-Frames mit einer Menge Elementen von diesen Bits profitieren könnten.

VBR nimmt nun soviele Bits, wie tatsächlich gebraucht werden. So bekommt ein komplett schwarzes Bild nur einige wenige Bits und ein komplexes Bild bekommt soviele Bits wie es benötigt. Während dies die beste Qualität erzeugt, ist die endgültige Dateigröße nicht mehr vorhersagbar.

Das ist die Stelle, an der **ABR** in Spiel kommt. Du gibst eine durchschnittliche Bitrate an,die Du erreichen möchtest, und der `Encoder` verteilt die Bits, die ihm zur Verfügung stehen. So spart der `Encoder` bei schwarzen Bildern Bits ein, die dann komplexeren Bildern zur Verfügung stehen. Es ist somit ein Kompromiss zwischen Qualität und Vorhersagbarkeit der Dateilänge.

6.2 Global Motion Compensation (GMC)

Global motion compensation (GMC) hilft beim Zoomen und Schwenken der Kamera, wenn sich Objekte nur in ihrer Größe oder Position verändern, ihren Typ dabei aber beibehalten. Die Aktivierung von GMC ist für Naturfilme oder Landschafts-Dokumentationen zu empfehlen, die komprimiert werden sollen.

6.3 Letterboxing

Vielleicht ist Dir aufgefallen, dass Videos manchmal vor dem Kodieren mit schwarzen Balken eingerahmt sind. Dieser Vorgang nennt sich letterboxing. Er kann bei einigen Abspielern dabei helfen, das korrekte Seitenverhältnis zu erhalten. Andere nutzen die schwarzen Balken zum Anzeigen von Untertiteln. Der Nachteil ist, dass diese schwarzen Balken das Bild vergrößern und daher mehr Bandbreite zum Kodieren benötigen, obgleich komplett schwarze Teile nicht sehr viel Platz beanspruchen - aber gerade die Übergänge zwischen dem Bild und den schwarzen Streifen brauchen sehr viele Bits.

6.4 P/I/B-Frames, GOPs

Dies stammt aus doom9's Forum: Dies ist eine kurze Erklärung einer MPEG "GOP", oder "Group of Pictures".

GOP:

* beginnt mit einem "I"-Frame, gefolgt normalerweise von einer Anzahl von "P" und "B"-Frames
* jede GOP ist unabhängig: alle Frames, die für Hochrechnungen benötigt werden, sind in jedem GOP vorhanden.
* GOPs können so klein sein wie ein einzelner I-Frame oder so groß wie gewünscht, aber normalerweise nicht mehr als 15 Frames für MPEG2. MPEG4 GOPs können so groß wie das maximale Keyframe-Intervall sein (üblicherweise 200-300 frames). Die meisten Codecs erlauben GOPs beliebiger Länge.
* Je länger eine GOP, desto effizienter aber unrobuster die Kodierung.

I-frame:

* "Intra-coded" frames: Durchschnittliche 7:1-Reduzierung.
* wie JPEG, jedes Video-Bild ist unterteilt in Blöcke zu 8x8 Pixeln aus Y, R-Y, and B-Y

* Blöcke sind in "macroblocks" zu 16x16 gruppiert.
* Die Standards besagen, dass Macroblocks horizontal in Scheiben gruppiert werden, die eine ähnliche durchschnittlichen Blockhöhe besitzen, obgleich tatsächlich vorhandene Codecs dieser Regel nicht notwendigerweisefolgen.
* Außerdem besagen die Standards, dass mehrere Scheiben einen Frame formen, und dass diese Frames die sich ergebenden "I"-Frames sind, obgleich Codecs auch dieser Vorgabe nicht folgen.

P-frame:

* P-Frames sind auf der Basis vorhergehender I- oder P-Frames zuzüglich der Daten veränderter Macroblocks vorhergesagte Frames.
* Durchschnittlich etwa 20:1 Reduzierung, oder etwa die Hälfte der Größe eine I-frames

B-frame:

* Bidirektional vorhersagbare Frames basierend auf dem Aussehen und der Position vergangener und folgender Frame-Macroblöcke.
* B-Frames benötigen weniger Daten als P-Frames, durchschnittlich etwa 50:1 Reduzierung.
* BFrames benötigen mehr decoder-buffer-Speicher da zwei Frames während der Rekonstruktion miteinander verglichen werden.
* B frames benötigen außerdem einer änderung der Kodierungsreihenfolge: Frames, die sich vom Encoder zum Decoder bewegen, sind NICHT in der Reihenfolge der Präsentation.

Für jeden Macroblock in einem P-Frame entscheidet der Encoder, ob er diesen Block bereits aus dem vorhergeheneden Frame kennt, oder ob es sich um einen komplett neuen Frame handelt. Im ersten Fall codiert er nur die Unterschiede (INTER mode). Im letzen Fall codiert er den gesamten Macroblock (INTRA mode).

Die Situation für B-Frames ist wie die folgende: "Ich kennen diesen Block nicht" (INTRA mode), "Ich kenne diesen Block vom vorhergehenden I- oder P-Frame (backward mode)", "dieser Block sieht aus wie ein Block im nächsten Frame (forward mode)", oder "dieser Block sieht aus wie eine Mischung des vorhergehenden und des nachfolgenden Frames (bidirectional mode".

MPEG1 GOPs sind immer "IBBPBBPBBPBB" (dies ist die Reihenfolge in der die Frames angezeigt werden, nicht notwendigerweise die Reihenfolge in der sie kodiert/gespeichert werden). MPEG2 GOPs sehen genauso aus. Sie können auch drei B-frames enthalten. Der DivX5 MPEG4 codec benutzt nur "IBPBPBPBPB", da es leichter zu implementieren ist, und weil B-Frame-Support for MPEG4-Codecs ziemlich neu ist. Du kannst in Zukunft mehr B-Frames erwarten.

6.5 Quarter Pixel (QP)

Quarter pel oder Quarter Pixel wirkt auf die Präzision beim Filtern von Macroblöcken. DivX 4 arbeitet mit Half Pel (1.5, 1.5); 1.25, 1.75, etc. sind beginnend mit DivX 5 möglich. Die konventionelle Teilung eines Bildes in Macroblöcke wird auf der Basis von Integer-Zahlen - 16x16 oder 8x8 - unter Verwendung der ergänzenden Information von dem, was als Virtual Block bekannt ist, verfeinert. Dies erlaubt es, Bewegungen von Objekten in Bildern realistischer wiederzugeben.