

Was soll das Ganze?

Ich schreibe dieses Dokument um interessierten Usern eine kleine Hilfe analog zu einer 'manpage' für den Umgang mit dem x264 CLI zur Verfügung zu stellen.

Für wen ist es gedacht?

Dieses Dokument ist für Leute gedacht, die das CommandLineInterface, kurz CLI, von x264 benutzen. Es ist definitiv nicht für Anfänger gedacht, das heißt es wird eine gewisse Menge an Vorwissen vorausgesetzt.

1 Grundstruktur eines x264 Aufrufes

Ein Aufruf der x264.exe folgt folgender Syntax: x264 [options] -o outfile infile [widthxheight] Dinge in eckigen Klammern [] sind hierbei als optional zu betrachten. Lässt man sie weg so werden Standardwerte verwendet. Lässt man die Option *options* weg so würde ein 1pass Encode mit konstantem Quantizer 26 und maximal einem B-Frame in Folge durchgeführt. Andernfalls kann man Einstellungen zum *Frame-type*, zur *Ratecontrol*, zur *Analyse* und zu den *Video Usability Info* vornehmen, welche in den folgenden Abschnitten behandelt werden. Die optionale Option *widthxheight* ist nur dann nötig wenn man x264 mit .raw Material füttert. x264 kann ansonsten nicht wissen welche Auflösung das Material hat.

Die zwingend zu setzende Option -o outfile infile legt fest wo x264 die Inputdatei finden kann und wo es was für eine Outputdatei erstellen soll. Als input erwartet x264 einen Pfad zu einer Datei mit der Endung .raw, .avi, .y4m (YUV4MPEG) oder .avs, wobei es bei .raw einen Rohdatenstream im YUV 4:2:0 Farbformat erwartet, .avs ein Avisynthscript (www.avisynth.org) sein muß und ein .avi mit vfw Decodern decodiert werden kann. (x264 nimmt auch Inputs über den STDIN an.) Um zu checken ob dies gegeben ist kann man einfach gucken ob man die Input Datei z.B. Mit www.VirtualDub.org ohne Probleme öffnen kann. Am besten schreibt man den Pfad zur Input/Output Datei jeweils immer in Anführungszeichen, damit es keine Probleme mit etweligen Leerzeichen im Pfad oder Dateinamen gibt. Neben der Ausgabe von Rohvideodaten (.264) ermöglicht x264 auch direkt das speichern in einen Matroska (.mkv) oder Mpeg4 (.mp4) Container, welche beide ihre Vor- und Nachteile haben. Was natürlich nur gilt wenn der entsprechende Support mit reincompiliert wurde, was bei den binaries von Sharktooth normalerweise der Fall ist.

Beispiel: x264.exe -o "g:\testclip.mp4" "e:\x264\testing\testclip.avs"

So nachdem wir nun den Aufbau der Grundstruktur eines Aufrufes der x264.exe geklärt haben geht es weiter mit den Optionen die man wählen kann.

2 Options

Bei x264 unterteilt man die *options* in vier Unterkategorien:

1. Frame-Type options
2. Ratecontrol
3. Analysis
4. Video Usability Info

Allgemein gilt: Wenn man nicht verstanden hat was eine Option macht und wann es sinnig ist sie anzuwenden, sollte man sie nicht aktivieren bzw. die Defaulteinstellung ändern.

Neben diesem Dokument besitzt auch x264 selbst eine kleine Hilfe, in der kurze Anmerkungen zu den einzelnen Optionen existieren. Die kurze Version dieser Hilfe kann man mit dem Parameter **-h** bzw. **--help** aufrufen und die etwas längere Version ist mit **--longhelp** erreichbar. Um herauszubekommen welche x264 Version man gerade verwendet benutzt man den **--version** Parameter.

2.1 Frame-type options

Die hier aufgeführten Optionen beschäftigen sich vor allem damit wann IDR-, I- und B-Frames gesetzt und wie sie gespeichert werden.

2.1.1 **-I <integer> bzw. --keyint <integer> (0 bis ...)**

Der hier festzulegende Wert gibt an, nach wie vielen Frames spätestens ein IDR-Frame gesetzt werden soll. Einerseits ist es aus kompressionstechnischen Gründen sinnvoll, diese nur an Szenenwechseln zu setzen, andererseits kann man nur von einem zum nächsten IDR-Frame springen und bei diesen schneiden, da man sonst nicht sicher wüsste, dass man keine Frames entfernt, auf die vielleicht später noch referenziert wird. Man will also nicht zu wenige, da man sonst nur in groben Sprüngen im Film umherspulen bzw. -springen oder nur sehr 'ungenau' schneiden kann. Als guter Richtwert erscheint mir ein Wert, der etwa dem zehnfachen der Framerate entspricht. Für PAL-Filme wäre dies ein Wert von 250. (NTSC = 240 bzw. 300) Der Standardwert den x264 nimmt wenn man den Parameter weglässt ist 250.

*Sollte normalerweise weglassen wenn man PAL-Material verwendet, bei NTSC-Material sollte **-I 300** gewählt werden.*

2.1.2 **-i <integer> bzw. --min-keyint <integer> (1 bis keyint/2)**

Hier legt man eine Grenze für den minimalen Abstand zwischen zwei IDR-Frames fest. Da man nur an IDR-Frames schneiden kann, will man keine zu große Distanz. Ein Wert von 1 sorgt dafür, dass der Codec jedes I-Frame automatisch zu einem IDR-Frame machen. Dieses vom Programmierer festgelegte Verhalten würde x264 um die Möglichkeit multipler Referenzframes über mehrere I-Frames hinweg berauben würde. Allgemein wird von den Programmierern ein Intervall von $0.4 * \text{Max IDR-keyframe interval}$ empfohlen. Unter einen Mindestabstand, der der Framerate entspricht sollte man möglichst nicht gehen, da die Heuristik zur Entscheidung ob ein IDR-Frame gesetzt wird oder nicht anscheinend noch nicht ganz ausgereift ist. Persönlich wähle ich immer den Mindestabstand. Der Standardwert den x264 nimmt wenn man den Parameter weglässt ist 25.

*Sollte normalerweise weglassen werden wenn man PAL-Material verwendet, bei NTSC-Material sollte **-i 30** gewählt werden.*

2.1.3 --scenecut <integer> (-1 bis 100)

Dieser Schwellenwert legt fest ab wie viel Prozent an Bildänderung ein neues Bild als Szenenwechsel erkannt wird, was direkt beeinflusst ob ein I-Frame gesetzt wird oder nicht. (-1 deaktiviert die Szenenwechselerkennung.) Das Problem ist, dass I-Frames idealerweise immer bei stärkeren Änderungen als qualitativ hochwertige Referenzframes gebraucht werden, sie jedoch einiges mehr an Datenrate benötigen als P- und B-Frames und deshalb nicht unnötig oft verwendet werden sollten. Der Standardwert den x264 nimmt wenn man den Parameter weglässt ist 40.

Sollte normalerweise weglassen werden.

2.1.4 -b <integer> bzw. --bframes <integer> (0 bis 16)

Dieser Parameter legt fest, wie viele B-Frames (Bidirektionale Frames) hintereinander erlaubt sind. Durch die Möglichkeit der Bidirektionalreferenz können B-Frames vor allem im mittleren und unteren Datenratenbereich (bei DVD Sicherheitskopien so kleiner 1500kBit/s) einiges an Datenrateneinsparungen bringen, ohne zu sichtbaren Qualitätsverlusten zu führen. Der Standardwert den x264 nimmt wenn man den Parameter weglässt ist 0, was die Nutzung von B-Frames deaktiviert. Zu beachten ist, dass man hier nur eine obere Schranke festlegt, bei normalem Videomaterial werden aber nur seltenst mehr als fünf B-Frames in Folge benötigt. Würden mehr verwendet wäre ein zu großer Detailverlust zu befürchten.

Sollte normalerweise zwischen -b 3 und -b 5 gewählt werden.

2.1.5 --no-b-adapt

Dieser Parameter ermöglicht es einem die an das Quellmaterial angepasste B-Frame-Auswahl zu deaktivieren, was zwar etwas an Performance bringt, jedoch in Hinsicht auf die Qualität in der Regel nicht zu empfehlen ist, besonders wenn man Werte größer Eins bei der maximalen Anzahl von aufeinander folgenden B-Frames gewählt hat.

Sollte normalerweise weglassen werden.

2.1.6 --b-bias <integer> (-100 bis 100)

Mit diesem Parameter kann der User die prozentuale Wahrscheinlichkeit erhöhen/senken, dass anstatt einem P- ein B-Frame verwendet wird. Das mit **-b <integer>** festgelegte Maximum wird hierdurch jedoch nicht überschritten. Es ist vielmehr die Frage, wie schnell bzw. ob es erreicht werden wird. Erhöht man die Wahrscheinlichkeit wird mehr Datenrate gespart, jedoch kann es zu einem verstärkten glätten des Bildes kommen. Senkt man die Wahrscheinlichkeit werden seltener B-Frames verwendet, jedoch kann dies dazu führen, dass das man etwaiges Deblocking verstärken muss damit keine Makroblöcke sichtbar werden. Der Standardwert den x264 nimmt wenn man den Parameter weglässt ist 0. Meiner Erfahrung nach macht es nur bei extrem niedrigen Auflösungen und in Vor- und Abspannen Sinn dies Feature zu benutzen und den Standardwert zu verändern.

Sollte normalerweise weglassen werden.

2.1.7 --b-pyramid

Erst wenn man dieses Feature aktiviert können B-Frames auch wieder auf B-Frames referenzieren, was nochmal einen kleinen Qualitätsgewinn bringt, jedoch auch die CPU Anforderung leicht steigert. Außer vielleicht wenn man für Handys&Co encoded würde ich dieses Feature immer aktivieren.

Sollte normalerweise mit --b-pyramid aktiviert werden.

2.1.8 --no-cabac

x264 unterstützt dabei neben CAVLC (Context Adaptive Variable Length Coding), einer Huffman-artigen, auch eine ebenfalls verlustlose leistungsfähigere arithmetische Codierung namens CABAC (Context Adaptive Binary Arithmetic Coding). Da eine Beschreibung von CABAC bzw. arithmetischer Codierung die meisten Leser eh nur langweilen würde, möchte ich hier auf einen netten und kurzen 'Artikel' hinweisen, den man unter http://www.computerbase.de/lexikon/Arithmetisches_Kodieren finden und lesen kann. Ich persönlich finde, CABAC ist das eindrucksvollste Feature bei x264, was die dahinter stehende Technik/Idee betrifft. Interessant ist, dass CABAC relativ langsam ist, aber ca. 10% an Datenrateneinsparungen bringt. Nur wenn man x264 zum Capturen benutzt, sollte man eventuell mit der Option --no-cabac von CABAC Abstand nehmen. Erwähnenswert ist auch noch, dass CABAC langsamer wird je höher die angestrebte Datenrate ist.

Sollte normalerweise weglassen werden.

2.1.9 -r <integer> bzw. --ref <integer> (1 bis 16)

Anders als bei anderen MPEG-Varianten können, sind bei H.264 auch multiple Referenzframes, zu einem Frame möglich. Mit diesem Parameter legt fest wie viele andere Bilder auf der Suche nach Ähnlichkeiten durchsucht werden und als Referenzen dienen können. Bei Auflösungen die größer oder gleich der Auflösung einer DVD sind sollte man hier nicht mehr als 5 nehmen, da man sonst nur seltenst noch eine Verbesserung erzielt jedoch einiges mehr an Zeit beim Komprimieren benötigt. Bei sehr niedrigen Auflösungen, etwa 320x240 und kleiner, kann man hier aber durchaus auch Werte bis 16 nehmen, was auch gleichzeitig das Maximum an möglichen Referenzen ist. Der Standardwert den x264 nimmt wenn man den Parameter weglässt ist 1.

Sollte normalerweise -r 6 sein.

2.1.10 -nf, --no-deblock

Mit dieser Option kann der Loop-Filter von x264 deaktiviert werden. Die Aufgabe des Loopfilters ist es an den Grenzen der Blöcke in die ein Bild eingeteilt wird zu glätten, so dass auch bei hohen Quantizern das Bild eher verschwommen als artefaktartig wirkt. Meiner Erfahrung nach sollte man den Loopfilter nur dann deaktivieren, wenn man definitiv nicht genug Prozessorpower hat.

Sollte normalerweise weglassen werden.

2.1.11 -f <integer:integer>, --deblock <integer:integer> (-6 bis 6)

Mit diesen Parametern kann man den Inloop-Deblocking-Filter von x264 beeinflussen, der dafür sorgt, dass auch bei starker Kompression möglichst keine Makroblockübergänge zu sehen sind, wie auch bei anderen MPEG-Varianten. Grob gesagt wird einfach an den Makroblockübergängen geglättet. Bei einem Quantizer < 16 wird in der Regeln der Loopfilter automatisch deaktiviert. Die Stärke des Filterns wird automatisch erhöht je höher der Quantizer ist. Obwohl man die Funktionen der beiden Regler an sich nicht getrennt betrachten sollte, kann man zum Verständnis sagen, **Strength** (erster Wert) legt die Stärke der Glättung fest und **Threshold** (zweiter Wert) legt fest, ab wann geglättet wird. Je nach Zielgröße sollte man die Stärke und den Schwellenwert etwas verändern. Persönlich lasse ich bei 1-CD-Sicherheitskopien einer DVD beide Werte auf 0 und stelle bei 2 CDs oder mehr beide auf -2. Gerade bei kleinen Clips oder verrauschtem Inputmaterial sollte man aber am Besten erst etwas herumprobieren, um nicht zu viel oder zu wenig zu glätten. Beim Encoden von Animes finde ich z.B. Werte wie 1:1 oder 2:2 recht ansprechend. Der Standardwert den x264 nimmt wenn man den Parameter weglässt ist 0:0.

Sollte man je nach Qualitätsempfinden leicht anpassen.

2.1.12 --interlaced

Ist diese Option aktiviert erwartet x264 interlaced Inputmaterial (kein Hybrid-Material!) und encoded auch wieder interlaced. Normalerweise sollte dieses Feature bei interlactem Inputmaterial immer aktiviert werden, aber Achtung: Nicht alle aktuellen AVC Decoder kommen mit interlactem AVC Material klar.

Sollte man bei interlactem Inputmaterial aktivieren.

2.2 Ratecontrol

Bei den folgenden Optionen geht es darum die Datenratenverteilung von x264 zu beeinflussen und zu entscheiden mit welcher generellen Vorgehensweise das Material encoded wird.

2.2.1 -q <integer> bzw. --qp <integer>(0 bis 51)

Dieser Parameter ist nur interessant wenn man einen Single Pass Encode machen will mit einem festen Quantizer. Ein Quantizer von 0 würde hierbei einer verlustfreien Kompression entsprechen.

Sollte nur verwenden wenn man einen single pass constant quantizer encode machen will.

2.2.2 -B <integer> bzw. --bitrate <integer> (1 – ...)

Dieser Parameter ist interessant wenn man einen Single oder Multi Pass Encode mit einer angestrebten durchschnittlichen Datenrate machen will. Durchschnittlich ist hierbei der wichtige Punkt. Wie bei vielen anderen Encodern auch wird hier die Bitrat in Kilobit (k=1000) pro Sekunde angegeben.

Sollte nur verwenden wenn man eine Durchschnittliche Datenrate anstrebt.

2.2.3 --crf <integer> (1 – 51)

Dieser Parameter ist nur interessant wenn man einen Single Pass Encode machen will mit einer angestrebten durchschnittlichen Quantizer. Durchschnittlich ist hierbei der wichtige Punkt. CRF ist eine gute Möglichkeit nur einen Durchgang anstatt zwei Durchgängen beim Encoden zu machen und eine dem 2pass Verfahren sehr ähnliche Bitratenverteilung zu haben, wobei man jedoch nicht die Zielgröße angeben kann.

Sollte nur verwenden wenn man einen single pass average quantizer encode machen will.

2.2.4 --qpmin <integer> (1 bis 51)

Dieser Parameter legt eine untere Schranke für den kleinsten Quantizer fest der beim single pass abr oder multipass Encoden benutzt werden darf. Persönlich setze ich diesen Wert meist auf 0, beim Encoden von normalem DVD/DVB Material mit einer Datenrate von 1000 - 2000kBit/s kann man den Wert jedoch beruhigt auf 10 setzen, da niedrigere Quantizer eh nicht vorkommen werden und man so dem Encoder etwas unter die Arme greift. Der Standardwert den x264 nimmt wenn man den Parameter weglässt ist 10.

Sollte normalerweise weglassen werden.

2.2.5 --qpmax <integer> (1 bis 51)

Dieser Parameter legt eine obere Schranke für den größten Quantizer fest der beim single pass abr oder multipass Encoden benutzt werden darf. Der Standardwert den x264 nimmt wenn man den Parameter weglässt ist 51. Bei normalen DVD/DVB&Co Encodes kann man ihn aber ruhig auf 42 senken.

Kann man ruhig mit --qpmax 42 aufrufen.

2.2.6 --qpstep <integer> (1 bis 50)

Dieser Parameter legt fest wie stark die Quantizerschwankung zwischen zwei benachbarten Frames sein darf die beim single pass abr oder multipass Encoden auftreten darf.. Ein zu niedriger Wert führt dazu, dass die Ratecontrol keinen Spielraum hat und man einen Constant Quantizer Encode erhält und ein zu hoher Wert führt eventuell dazu, dass man enorme Qualitätsschwankungen erhält. Für DVD Auflösungen ist vier normalerweise okay, für sehr kleine Auflösungen ist acht aber öfters besser. Der Standardwert den x264 nimmt wenn man den Parameter weglässt ist 4.

Sollte normalerweise weglassen werden.

2.2.7 --ratetol <float> (0.1 – 100.0)

Dieser Parameter ist nur interessant wenn man einen Single Pass Encode machen will mit einer angestrebten durchschnittlichen Datenrate. ratetol beschreibt wie viel die Datenrate eines Frames von der angestrebten Durchschnittlichen Datenrate abweichen darf (etwa +/- 0.5% pro Punkt). Hierbei wird bedeutet ratetol=inf nicht, dass jedes Frame mit minQP encoded wird, sondern dass nach der initialen Quantizerverteilung-/schätzung keine Anpassung mehr anfällt. Der Standardwert den x264 nimmt wenn man den Parameter weglässt ist 1.0.

Sollte normalerweise weglassen werden.

2.2.8 --vbr-maxrate <integer> (0 bis ...)

Mit diesem Wert kann man die maximale Datenrate festlegen. VBV steht für 'Video Buffering Verifier'. Interessant ist dieser Parameter normalerweise nur wenn man für bestimmte Geräte encoded die nur Clips bis zu einer gewissen Datenrate encoden können. Der Standardwert den x264 nimmt wenn man den Parameter weglässt ist 0, was eine beliebig hohe Datenrate zulässt.

Sollte normalerweise weglassen werden.

2.2.9 --vbr-bufsize <integer> (0 bis ...)

Mit diesem Wert kann man die maximale Buffergröße festlegen, die ein Geräte zur Verfügung stehen muß, das den Stream später einwandfrei decoden kann. Interessant ist dieser Parameter normalerweise nur wenn man für bestimmte Geräte encoded die nur eine forgegebene Buffergröße besitzen. Der Standardwert den x264 nimmt wenn man den Parameter weglässt ist 0, was besagt, dass x264 nicht garantiert, dass eine bestimmte Buffergröße ausreichend ist um den encodeten Stream abspielen zu können.

Sollte normalerweise weglassen werden.

2.2.10 --vbr-init <float> (0.0 bis 1.0)

Dieser Wert legt fest wieviel des Video Buffers eines Endgerätes initial belegt werden ehe die Wiedergabe gestartet wird. Der Standardwert den x264 nimmt wenn man den Parameter weglässt ist 0.0, d.h. es würde einfach mit dem Playback gestartet und davon ausgegangen, daß der Video Buffer nicht nötig bzw. immer ausreichend gefüllt ist.

Sollte normalerweise weglassen werden.

2.2.11 --ipratio <float> (1.0 bis 10.0)

Mit diesem Parameter legt man fest wieviel höher die Quantizer eines P-Frames im Vergleich zu den I-Frames sein soll auf die er referenziert. Der Standardwert den x264 nimmt wenn man den Parameter weglässt ist 1.4, d.h. P-Frames werden immer mit mindestens einem 40% größeren Quantizer encoded, als das I-Frame auf das sie verweisen.

Sollte normalerweise weglassen werden.

2.2.12 --pbratio <float> (1.0 bis 10.0)

Mit diesem Parameter legt man fest wieviel höher die Quantizer eines B-Frames im Vergleich zu den P-Frames sein soll auf die es referenziert. Der Standardwert den x264 nimmt wenn man den Parameter weglässt ist 1.3, d.h. B-Frames werden immer mit mindestens einem 30% größeren Quantizer encoded, als dass die P-Frames auf das sie verweisen.

Sollte normalerweise weglassen werden.

2.2.13 --chroma-qp-offset <integer> (-12 bis 12)

Mit diesem Parameter legt man fest um wieviel Quantizerstufen Farbinformationen stärker als Helligkeitsinformationen quantisiert werden sollen. Der Standardwert den x264 nimmt wenn man den Parameter weglässt ist 0. Je nach Film kann hierdurch zwar bis zu 10% der Bitrate einsparen, ich würde jedoch normalerweise davon abraten einen der Bereiche stärker zu quantisieren.

Sollte normalerweise weglassen werden.

2.2.14 -p <1|2|3> bzw. --pass <1|2|3> (1 bis 3)

Hat man sich entschieden anstatt einem one pass ein multi pass encoding durchzuführen, so teilt man mit diesem Parameter dem Encoder mit in welchem pass er sich befindet. Normalerweise sollte man hier nur --pass 1 und --pass 3 verwenden, da --pass 2 das gleiche wie --pass 3 macht, nur das kein Analysefile mehr erstellt wird. Außer bei sehr kurzen Clips lohnt es sich aber nur seltenst mehr als zwei Durchgänge (pass 1 + pass 3) durchzuführen.

Sollte normalerweise erst einmal mit -p 1 und dann beliebig oft mit -p 3 aufgerufen werden..

2.2.15 --stats <string>

Hiermit kann man festlegen wo in jedem Encodingdurchlauf die etwelige Statistikdatei erstellt werden soll. Achtung, die Datei wird nach jedem Durchlauf durch eine neue ersetzt. Lässt man den Parameter weg, so legt x264 automatisch eine Datei namens x264.stats im Hauptverzeichnis des Datenträgers des Inputfiles an.

Sollte normalerweise weglassen werden.

2.2.16 --rceq <string>

Über diesen Parameter kann der User die Formel ändern über die x264 die Datenratenflußkontrolle regelt. Der Standardwert den x264 nimmt, wenn man den Parameter weglässt ist "blurCplx^(1-qComp)". Eventuell gibt es hier besser geeignete Formeln, jedoch bis jemand eine findet würde ich raten die Finger von diesem Parameter zu lassen.

Sollte normalerweise weglassen werden.

2.2.17 --qcomp <float> (0.0 bis 1.0)

Mit diesem Wert kann man grob festlegen was im jedem Multipass eher angestrebt wird. Eine konstante Qualität, d.h. jedes Bild verliert gleichmäßig viele Informationen oder eher eine konstante Datenrate, d.h. jedes Bild wird mit der gleichen Menge an Bits kodiert egal wie komplex es ist. Ein Wert von 1 würde hierbei zu einem konstantem Quantizer encode und ein Wert von 0 zu einem konstanten Bitraten Encode führen. Der Standardwert den x264 nimmt, wenn man den Parameter weglässt ist 0.6, da die Entwickler diesen Wert am Optisch ansprechensten in den Ergebnissen fanden. Metriken, wie PSNR liefern jedoch mit Werten im Bereich 0.7 bis 0.8 meist bessere Werte.

Sollte normalerweise weglassen werden.

2.2.18 --cplxblur <float> (0.0 bis 999.0)

So wie cplxblur momentan in der Formel für die Datenratenflußkontrolle verwendet wird, verringert es Schwankungen in der Quantizervergabe vor der Curve Compression. Verändert man die Formel für die Datenratenflußkontrolle sollte man eventuell auch hier etwas ändern. Der Standardwert den x264 nimmt, wenn man den Parameter weglässt ist 20.0.

Sollte normalerweise weglassen werden.

2.2.19 --qblur <float> (0.0 bis 99.0)

Dieser Parameter ist für eine Reduzierung der Schwankungen in der Quantizervergabe nach der Curve Compression zuständig. Der Standardwert den x264 nimmt, wenn man den Parameter weglässt ist 0.5.

Sollte normalerweise weglassen werden.

Anmerkung: Wer einen groben Überblick darüber braucht wie die Ratecontrol in x264 funktioniert sollte sich mal dieses (<https://trac.videolan.org/x264/changeset/350?format=diff>) Dokument angucken.

2.2.20 --zones <zone0>/<zone1>/...

Mit dieser Option kann man mehrere Abschnitte bestimmen, in denen man festlegen kann, dass entweder ein bestimmter Quantizer, oder das xy-fache der durchschnittlichen Datenrate in dieser Zone verwendet werden soll. Zu beachten ist, dass man immer noch an die Grenzen gebunden ist die durch qpmin und qpmax festgelegt wurden. Eine Zone besteht also aus: <start frame>, <end frame> und <option>, wobei die Option hier q=<integer> (force QP, aus 0 bis 51) oder b=<float> (bitrate multiplier, aus 0.01 bis 100.0) beinhalten muß. Hier mal ein kleines Beispiel, der Parameter --zones 0,10231,b=0.25/156034,183123,q=42 würde zwei Zonen erstellen, eine von Frame 0 bis Frame 10231 in dem nur ¼ der angestrebten durchschnittlichen Datenrate genommen wird und eine zweite Zone von Frame 156034 bis Frame 183123 in der alle Frames mit einem festen Quantizer von 42, bzw. qpmax falls 42 > qpmax, encoded werden.

Sollte nur verwenden, wenn man sie braucht.

2.2.21 --qpfile <string>

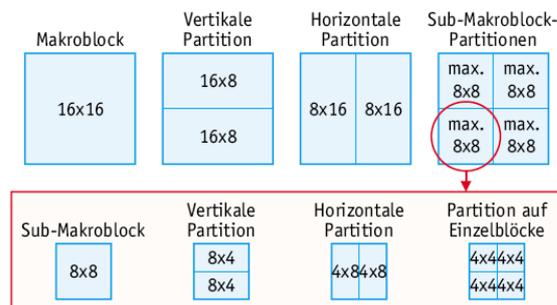
Mit diesem Parameter kann man x264 eine Datei zur Hand geben in der man zu einzelnen Frames den Frametyp und auch den zu verwendenden Quantizer festlegen kann ohne das auf die normale Ratecontrol geachtet wird. Für jedes Frame ist im Input-QPfile eine einzelne Zeile nötig. Die Syntax die man verwendet ist: *Framenummer(int) Frametyp(char) FrameQP(int)*, wobei Frametyp entweder *I* (IDR-Frame), *i* (normales I-Frame), *P* (P-Frame), *B* (B-Frame auf das referenziert wird) oder *b* (normales B-Frame) sein kann. Alle Frames für die mit durch --qpfile "Pfad zum QP-File" keine vorgaben gegeben werden wird die normale Ratecontrol benutzt.

Sollte nur verwenden, wenn man sie braucht.

2.3 Analysis

MPEG-Komprimierungsverfahren basieren stark darauf Ähnlichkeiten in anderen Bildern zu finden und dadurch Platz zu sparen, so dass man nur noch Referenzen auf Teile in anderen Bildern speichert. In diesem Bereich hat der User nun die Möglichkeit festzulegen wie genau die Unterteilung der Bilder/Frames geschehen soll. Je genauer die Einteilung ist desto höher ist die Wahrscheinlichkeit Ähnlichkeiten zu finden, jedoch steigt auch der Arbeitsaufwand.

Beim Komprimieren wird ein einzelnes Bild in Makroblöcke unterteilt, anschließend wird nach Ähnlichkeiten und Bewegungsvektoren zwischen den Makroblöcken des Bildes und denen der Referenzframes gesucht. Da sich schon bei älteren Verfahren gezeigt hat, dass eine genauere Bewegungssuche zwar enorm bei der Kompression helfen kann, jedoch auch extrem viel Zeit in Anspruch nehmen kann, hat man hier die Möglichkeit Einfluß auf die Berechnung und Sucher der Bewegungsvektoren zu nehmen. Hier kann nun man festlegen wie genau ein Makroblock in den einzelnen Framearten zerlegt wird. Normalerweise sollte man hier alles aktivieren. Zur leichteren Vorstellung was 4x4, 8x16 usw. Bedeutet hier eine kleines Bild, in dem gezeigt wird wie ein einzelnes Frame unterteilt wird:



2.3.1 -A, --partitions <string>, ehem. --analyse <string>

Dieser Parameter legt fest wie genau die Unterteilung der Makroblöcke geschehen soll. Es stehen die Partitionsmöglichkeiten none, p8x8, p4x4, b8x8, i8x8, i4x4, all zur Verfügung, wobei ein i, b oder p vor einer Zahl jeweils festlegt, dass die Partitionierung sich auf einen bestimmten Frametyp bezieht. Der Standardwert den x264 nimmt, wenn man den Parameter weglässt ist "p8x8,b8x8,i8x8,i4x4". Wobei zu beachten ist, dass i8x8 nur im High Profile von H.264 erlaubt ist und den Parameter --8x8dct voraussetzt. Das High Profile wird zwar von den meisten besseren H.264 Decoder, aber wahrscheinlich (zumindest anfangs) nicht von allen StandAlonePlayern unterstützt.

Sollte mit -A "all" oder -A "p8x8, p4x4, b8x8" aufgerufen werden

2.3.2 --direct-8x8 < -1 | 0 | 1 >

Dieser Parameter beeinflusst die Bewegungsvorhersage in soweit, als das es festlegt mit welcher Blockgröße angefangen werden soll. Ein Wert von 0 entspricht hier 4x4, 1 entspricht 8x8 und bei -1 wird immer der kleinste im Profil erlaubte ausgewählt. Sollte aus Kompatibilitätsgründen aktiviert sein, da der Defaultwert -1 ist kann man die Option aber weglassen.

Sollte normalerweise weglassen werden.

2.3.3 --direct <string>

Hier kann man festlegen ob und wie die Motionvektorendifferenzen der Makroblöcke eines B-Frames mit benachbarten Böcken im Frame selber („spatial“) oder mit den Motionvektoren der Nachbarframes („temporal“) gebildet werden sollen. „Spatial“ sieht bei langsamen und „temporal“ bei schnellen Szenen etwas besser aus. Wenn man ein multipass encoding durchführt sollte man idealerweise dem Codec die Wahl überlassen („auto“) ob temporal oder spartial gefiltert wird. Der Standardwert den x264 nimmt, wenn man den Parameter weglässt ist „spatial“.

Sollte normalerweise mit --direct auto aufgerufen werden.

2.3.4 -w bzw. --weightb

Mit diesem Parameter kann man die weighted biprediction aktivieren, tut man dies so werden Bildinhalte aus mehreren Referenz-Frames gemischt und können in der Quellen beliebig gewichtet in die Mischung eingehen. Dies ist vor allem hilfreich um Aus- und Überblendungen effizienter zu speichern.

Sollte normalerweise mit -w aufgerufen werden.

2.3.5 --me <string>

Mit diesem Parameter kann man zwischen den eigentlichen Suchmethoden wählen mit denen die Bewegungsvektoren ermittelt werden. Zur Auswahl stehen diamond search („dia“) welche man normalerweise nicht nehmen sollte, da sie zwar schnell aber meist doch recht ungenau ist. hexagonal search („hex“), was auch der Standard ist und normalerweise reichen sollte. uneven multi-hexagon search („umh“) was eine erweiterte hex search ist und exhaustive search („esa“) bei der alles abgesucht wird im festgelegten Umkreis.

Sollte normalerweise weglassen werden.

2.3.6 -m <integer> bzw. --subme <integer> (1 bis 7)

Mit diesem Parameter legt man fest wie detailliert die Sub-Makroblockverfeinerung durchgeführt wird. Bei Level 1 wird erst eine Bewegungsanalyse auf jedem ganzen Pixel gemacht. Auf den besten Kandidaten wird dann eine schnelle Bewegungsanalyse auf jedem ¼ Pixel durchgeführt. Bei Level 2 wird das Gleiche gemacht wie bei Level 1, jedoch mit etwas genaueren und damit auch langsameren Suchalgorithmen. Bei Level 3 wird das Gleiche wie bei Level 2 gemacht, nur nicht auf ganzen, sondern auf halben Pixeln und später wieder auf ¼ Pixeln gesucht. In Level 4 wird direkt mit einer schnellen Suche auf ¼ Pixel angefangen, deren beste Kandidaten dann mittels einer genauen/langsamen Suche noch verfeinert werden. In Level 5 wird direkt eine genaue / langsame Suche auf jedem ¼ Pixel gemacht und der beste Kandidat gewählt. Bei Level 6 (RDO (Rate Distortion Optimization)) werden immer die Bewegungsvektoren genommen, die das möglichst beste Verhältnis zwischen Qualität und der angestrebten Datenrateversprechen. Level 7 ist Level 6 und zusätzlich wird noch eine weitere sub8x8 Aufteilungsvorgenommen. Der Standardwert den x264 nimmt, wenn man den Parameter weglässt ist 5, was auch sinnig ist, da Level 6 und Level 7 nur etwas für Qualitätsfanatiker ist.

Sollte normalerweise weglassen oder mit -m 6 oder -m 7 aufgerufen werden.

2.3.7 --mvrage-thread <int>

Hier kann man die minimale Größe des Motionvektor-Buffers angeben die x264 zwischen zwei threads benutzt. Der Standardwert ist hier -1, was einer automatischen Auswahl entspricht.

Sollte normalerweise weglassen werden.

2.3.8 --merange <integer> (aus 4 bis 1024)

Mit diesem Parameter legt man die Größe der Suchumgebung für eine uneven multi-hexagon search oder eine exhaustive search fest, falls diese ausgewählt wurden. Der Standardwert den x264 nimmt, wenn man den Parameter weglässt ist 16, was auch normalerweise der zu empfehlende Wert ist. Zwar kann man den Wert bis 1024 hoch setzen, jedoch tut sich ab 32 eigentlich nichts mehr und zu große Werte können zu großen Bewegungsvektoren führen die aber beim Partitionieren eher kontraproduktiv sind. Angemerkt sei auch, das bei 'Brute-Force' Suchen i.d.R. hier auch 'nur' 32 benutzt wird.

Sollte normalerweise weglassen oder mit --merange 32 aufgerufen werden.

2.3.9 --b-rdo

Mit diesem Parameter kann man eine Rate Distortion basierte Entscheidung für B-Frames aktivieren, wenn man vorher Level 6 oder höher gewählt hat. Da B-Frames einen Großteil der Frames stellen wenn man sie nutzt würde ich empfehlen dieses Feature zu aktivieren, wenn man wirklich auf Qualität aus ist.

Sollte man mit --b-rdo aktivieren, wenn man vorher Level 6 oder 7 gewählt hat.

2.3.10 --mixed-refs

Mit dieser Option legt man fest, dass in einem Makroblock auch Referenzen in unterschiedlichen Genauigkeitsstufen gewählt werden können und nicht nur aus einer Stufe. Dieses Feature kostet einen etwa 10% an Zeit beim Komprimieren und liefert nur relativ wenig Qualitätsgewinn im Ausgleich. Persönlich würde ich das Feature aktivieren, da es bei mir nicht mehr darauf ankommt ob es 10% schneller oder langsamer ist.

Sollte man mit --mixed-refs aktivieren.

2.3.11 --no-chroma-me

Ist dieses Feature aktiviert verwendet x264 nur die Luminanz-(Helligkeits-)Informationen, um Bewegungsrichtungen festzustellen. In seltenen Fällen (die gleiche Helligkeit, aber wechselnde Farbigkeit) mag dies jedoch vielleicht nicht genau genug sein. Wird das Feature nicht aktiviert so wird zur Helligkeits- auch die Farb-Komponenten zur Ermittlung der Bewegungsrichtung verwendet.

Sollte normalerweise weglassen werden.

2.3.12 --bime

Bidirectional ME bedeutet, dass bei der Suche nach Referenzen für B-Frames eine zusätzliche Suche sowohl in vorherigen, als auch in folgenden Frames durchgeführt wird. Ist diese Option nicht aktiviert, wird nur eine Suche in den vorherigen und eine in den nachfolgenden Bildern gemacht. In den meisten Fällen ist dies auch ausreichend. öfters findet man jedoch mit einer weiteren Suche noch bessere Referenzen. Dieses Feature sollte normalerweise aktiviert werden, da es die Effizienz von B-Frames nochmal etwas erhöht.

Sollte mit --bime aktiviert werde um die Effizienz der B-Frames zu erhöhen.

2.3.13 -8, --8x8dct

Aktiviert eine 8x8 DCT-Transformation ohne die eine i8x8 und i4x4 Analyse nicht möglich ist. Genau wie i8x8 und i4x4 handelt es sich hierbei um ein High profile Feature.

Sollte man mit -8 aktiviere, wenn man das High Profile nutzen darf.

2.3.14 -t <integer> bzw. --trellis <integer> (0 bis 2)

Der der Trellisquantisierung handelt es sich um eine Erweiterung der normalen Quantisierung bzw. ein Art 2ten quantization pass in dem die DCT Verteilung nochmal überdacht wird. Es werden einige Koeffizienten fallen gelassen (Details entfernt) und andere Koeffizienten die sonst wegfallen würden, gerettet um ein besseres Bild bzw. eine bessere Quantisierung zu erreichen. Da x264 sich beim Einsparen geschickt anstellt kann im Endeffekt oft ein etwas kleinerer Quantizer verwendet werden und man gewinnt sogar Qualität. Hier hat man nun die Optionen die Trellisquantisation zu deaktivieren (0), nur für den entgeltigen Motionblock durchzuführen (1) oder bei jeder partitionierungsentscheidung durch zu führen (2). Der Standardwert den x264 nimmt, wenn man den Parameter weglässt ist 0. Da Trellis je nachdem aber doch einiges machen kann bevorzuge ich trellis 1 oder 2.

Sollte man wenn man Zeit hat mit -t 1, als Qualitätsfanatiker mit-t 2 aktivieren und als normaler User eher weglassen

2.3.15 --aq-strength <float> (0.0 bis 1.1) (nicht im Standard SVN)

Adaptive Quantization betrachtet die absolute Helligkeit einer Szene. Aktiv wird es in sehr hellen oder dunklen Bildbereichen. In solchen Bildbereichen weist Adaptive Quantization Texturen, die nicht leicht mit den Augen zu erkennen sind, einen höheren Quantizer zu und versucht so Datenrate zu sparen. Es wird nicht mehr nur ein, sondern mehrere Quantizer für einen Block verwendet, die ein wenig vom durchschnittlichen Quantizer abweichen. Dieses Abweichen sollte es normalerweise ermöglichen, Datenrate einzusparen. Zusätzlich zum oben genannten wird auch versucht größere Flächen einheitlicher zu Quantisieren um so auffällige Blockbildungen zu verringern. Mit dieser Option kann man festlegen wie stark adaptive quantization sich auswirkt. Ein Wert von 0.0 deaktiviert adaptive quantization und ein Wert von 1.1 verwendet sehr starke adaptive quantization. Welchen Wert man hier wählt ist rein subjektiv, da ich persönlich 0.4 bevorzuge ist dies auch meine Empfehlung. Der Standardwert den x264 nimmt, wenn man den Parameter weglässt ist 0.0.

Aktiviere ich persönlich gerne mit --aq-strength 0.4, je nach Geschmack kann man diesen Parameter aber auch einfach weglassen.

2.3.16 --aq-sensitivity <float> (nicht im Standard SVN)

Mit diesem Parameter wird festgelegt wie detailarm ein Makroblock sein muß, ehe adaptive quantization auf ihn angewendet werden darf. Ein Wert von 5 würde auf fast jeden Block adaptive quantization anwenden, wohingegen ein wert von 22 nur auf komplett einheitliche Blöcke adaptive quantization anwendet. Der Standardwert den x264 nimmt, wenn man den Parameter weglässt ist 15.

Sollte normalerweise weglassen werden.

2.3.18 --pre-scenecut

Diese Option aktiviert eine für die neue Threadverwaltung nötige schnelle und etwas ungenauere Szenenwechselerkennung. Es werden nicht mehr Teilbilder sondern nur ganze Bilder auf einzelne Threads verteilt. Wenn mit mehr als einem Thread encoded wird, wird diese Option automatisch aktiviert.

Sollte normalerweise weglassen werden.

2.3.19 --nr <integer> (0- 100000)

Dieser Option ermöglicht es dem User einer Noise Reduction, Rausch unterdrückung in x264 zu nutzen um so vielleicht etweilliges Preprocessing des Inputs sich zu sparen. Sinnige Werte bie normalen Quellen liegen so zwischen 0 und 600, höhere Werte sind nur bei sehr verrauschtem Material sinnig. Hier sollte man ersteinmal ein bißchen rumtesten ehe man sich hier auf einen Wert festlagt. Der Standardwert den x264 nimmt, wenn man den Parameter weglässt ist 0.

Sollte normalerweise weglassen werden.

2.3.20 --no-fast-pskip

In x264 sind ein P-Frame skip patch enthalten, der zwar das Encoden um etwa 5% beschleunigt, jedoch bei manchen Clips die Qualität etwas senkt, mit dem --no-fast-pskip Parameter deaktiviert man diesen Patch.

Sollte mit --no-fast-pskip aktiviert werden, wenn man es nicht eilig hat

2.3.21 --no-dct-decimate

Mit diesem Parameter kann man verhindern, dass bei der DCT-Umwandlung die Rundung kleiner Koeffizient auf Null vorgenommen wird um bits zu sparen. Standardmäßig ist bei B-Frames diese Dezimierung immer deaktiviert und bei Trellis immer an. Da die Dezimierung aber nicht immer optimal ist kann man mit dieser Option die Dezimierung für alle Frames deaktivieren, was bei Verwendung von Trellis sinnig ist.

Sollte normalerweise mit --no-dct-decimate aufgerufen werden, falls man trellis verwendet.

2.3.22 Custom Matrix Options: --cqm <sting> bzw. --cqmfile <string>

Anders als bei Mpeg4 ASP kann man bei x264 nicht nur zwei eigene Quantisierungsmatrizen angeben, sondern man hat vielmehr die Möglichkeit für alles mögliche Matrizen anzugeben. Da man über Custom Matrizen an sich schon ein ganzes Dokument verfassen könnte werde ich hier nicht auf sie eingehen. Ein generelles Verständnis um was es bei ihnen geht kann man z.B. im Anhang des 'Wissenswertes rund um Xvid' erhalten. Anzumerken ist, dass Custom Matrizen erst im High Profile von H.264 unterstützt werden. Sharktooth hat im englischen Doom9 Forum mal eine selbst geschriebene Matrizenfolge gepostet (<http://forum.doom9.org/showthread.php?t=96298>), welche eventuell einen gewissen Anstoß für User geben mag. Persönlich würde ich momentan noch keine Custom Matrizen verwenden, da noch wenig Erfahrungen mit x264 und den Custom Matrizen gesammelt wurden und sie vom Decoder das High profile fordern.

Sollte normalerweise weglassen werden.

2.3.23 --deadzone-intra <0 bis 32>

Dieser Parameter sollte helfen die feinen Details und Filmrauschen in I- und IDR-Frames besser zu erhalten. Der Standardwert ist hier 21, je höher desto mehr wird Rauschen&Co erhalten.

Sollte normalerweise weglassen werden.

2.3.24 --deadzone-inter <0 bis 32>

Dieser Parameter sollte helfen die feinen Details und Filmrauschen in P- und B-Frames besser zu erhalten. Der Standardwert ist hier 11, je höher desto mehr wird Rauschen&Co erhalten.

Sollte normalerweise weglassen werden.

2.4 Video Usability Info

Durch die Video Usability Info Optionen werden keine Parameter gesetzt die das Encoden als solches nicht beeinflussen, sondern vielmehr Hinweise für den Decoder hinterlegen. Soweit mir bekannt beachtet momentan aber noch kein einziger Decoder diese Einstellungen.

2.4.1 --sar width:height

Das Sample Aspect Ratio (PAR) gibt das Verhältnis von Breite zu Höhe eines Pixels (16x16 Bildpunkte) in einem Bild an. Bei 'normalem', nicht anamorphem Material sollte man diesen Parameter immer auf 1:1 setzen, anstatt den Standardwert von undefiniert zu belassen. Eine kleine Einführung was es mit anamorphem Encoding auf sich hat findet man bei http://home.arcor.de/brotherjohn/anamorph_intro.html .

Sollte normalerweise mit --sar 1:1 aufrufen.

2.4.2 --overscan <string>

Overscan bezeichnet den Bereich eines Bildes, der keine wesentlichen Informationen enthält, sondern vielmehr dazu benutzt wird das Bild an den Rändern aufzufüllen um gewissen Auflösungs oder Aspekt Ratio Anforderungen gerecht zu werden. Setzt man overscan = show, so wird immer das ganze Bild, wählt man overscan=crop wird ein Decoder der das Flags akzeptiert versuchen das Bild vor der Wiedergabe von schwarzen Rändern zu befreien.

Sollte normalerweise weglassen werden.

2.4.3 -videoformat <string|>

Mit diesem Parameter kann man dem Decoder sagen um was für Material es sich handelt, zur Auswahl stehen hier:component, pal, ntsc, secam, mac und undef.

Sollte normalerweise mit -videoformat pal oder -videoformat ntsc angerufen werden.

2.4.4 --fullrange <string>

Mit diesem Parameter kann man festlegen ob die Farben um die es geht das volle Spektrum 1 bis 255, wie es bei PC basiertem Material normal ist oder nur einen eingeschränkten Bereich 16 bis 240 darstellen. Der Standardwert den x264 nimmt wenn man den Parameter weglässt ist "off".

Sollte normalerweise mit --fullrange "on" aufrufen.

2.4.5 --colorprim <string>, --transfer <string>, --colormatrix <string>

Die hier aufgeführten Optionen sind nur für Videophile User, die genau wissen was für Hardware sie haben, welche Farbdarstellungen verwendet werden usw. Wenn man nicht 100%ig weiß, was man hier einstellen soll, sollte man bei den Standardwerten bleiben.

Sollte normalerweise weglassen werden.

2.4.6 --chromaloc <integer>

Auch bei Chroma Sample Location handelt es sich ein Option die für Videophile gedacht ist. Sie ermöglicht es dem User fest zu legen wie genau die Chromainformation des H.264 Material beim Playback wieder hergestellt werden soll. Interessant sind hier die Werte 1 wenn das Inputmaterial MPEG1 und 0 wenn das Inputmaterial MPEG2 oder MPEG4 war.

Sollte normalerweise weglassen werden.

2.5 Input/Output:

In diesem Bereich kann man einige Einstellungen bezüglich des Inputs, des Outputs und des Encodings an sich feststellen.

2.5.1 --level <string>

Mit diesem Parameter kann dem Decoder anhand von gewissen Leveln mitteilen welche Voraussetzungen er erfüllen muss um den Videostream richtig wiedergeben zu können. (Welche Level es genau gibt und was sie machen steht in Anhang A des H.264 Standards.)

Sollte normalerweise weglassen werden.

2.5.2 --fps <float|rational>

Mit diesem Parameter kann man dem Decoder mitteilen mit welcher Framerate das encodierte Material abgespielt werden soll. Die gängigsten Parameter sind 24000/1001, 24, 25, 30000/1001, 30, 50, 60000/1001 und 60. Der Standardwert den x264 hier nimmt wenn man den Parameter weglässt ist 25.

Sollte normalerweise weglassen werden oder bei NTSC Material mit --fps 24000/1001 aufgerufen werden.

2.5.3 --seek <integer> (1 bis ...)

Mit diesem Parameter kann man dem Encoder mitteilen, das er den Inputstream erst ab einem gewissen Frame encoded und die Frames davor ignoriert. Der Standardwert den x264 hier nimmt, wenn man den Parameter weglässt ist 1.

Sollte normalerweise weglassen werden.

2.5.4 --frames <integer> (1 bis ...)

Mit diesem Parameter kann man x264 sagen wie viele Frames insgesamt encoded werden sollen.

Sollte normalerweise weglassen werden.

2.5.5 -o <string> <string> bzw. --output <string> <string>

Die zwingend zu setzende Option -o outfile infile legt fest wo x264 die Inputdatei finden kann und wo es was für eine Outputdatei erstellen soll. Als input erwartet x264 einen Pfad zu einer Datei mit der Endung .raw, .avi oder .avs, wobei es bei .raw einen Rohdatenstream im YUV 4:2:0 Farbformat erwartet, .avs ein Avisynthscript (www.avisynth.org) sein muss und ein .avi mit vfw Decodern decodiert werden kann. Um zu prüfen ob dies gegeben ist kann man einfach gucken ob man die Input Datei z.B. Mit www.VirtualDub.org ohne Probleme öffnen kann. Am besten schreibt man den Pfad zur Input/Output Datei jeweils immer in Anführungszeichen, damit es keine Probleme mit Leerzeichen im Pfad oder Dateinamen gibt. Neben der Ausgabe von Rohvideodaten (.264) ermöglicht x264 auch direkt das speichern in einen Matroska (.mkv) oder Mpeg4 (.mp4) Container, welche beide ihre Vor- und Nachteile haben. Was natürlich nur gilt wenn der entsprechende Support mit rein compiliert wurde.

Muss verwendet werden!

2.5.6 --threads <integer>|auto (1 bis 16)

Hinter dieser Option verbirgt sich die Möglichkeit dem Encoder zu sagen, dass einige der internen Berechnungen so ausgeführt werden, dass sie separat berechnet werden können. Dies bringt nur etwas was wenn man mehrere (eventuell simulierte) Prozessoren im System hat und diese vorher nicht schon 100%ig ausgelastet sind. Momentan werden max. 16 Threads unterstützt. Der Standardwert den x264 nimmt, wenn man den Parameter weglässt ist 1. Wenn man mehrere CPU-Kerne im System hat sollte man hier auto wählen,

Sollte man mit --threads auto nutzen wenn man mehrere CPUs hat.

2.5.7 --non-deterministic

Diese Option führt zu einer nicht deterministischen Verteilung der Frames auf die zur Verfügung stehenden Threads. Dies führt zu einem kleinen Geschwindigkeitsgewinn bei Multiprozessorsystemen, hat jedoch den Nachteil, dass Fehler bei der Threadzuweisung wesentlich schwerer aufzufspüren sind, da die Zuweisung in jedem Durchlauf anders läuft.

Sollte man mit --non-deterministic nutzen wenn man mehrere CPUs hat.

2.5.8 --thread-input

Sorgt dafür, dass ein zusätzlicher Thread für das Einlesen des Inputs verwendet wird, dies bringt nur etwas wenn die CPU nicht voll ausgelastet wird, da die Daten nicht schnell genug gelesen werden können. Bei einem Single Core System kommt dies nur selten vor, wenn z.B. die Festplatte nicht nachkommt oder z.B. manche Daten über die GPU geschleift werden und sich deshalb verzögern. Stellt man die Threads > 1 wird die Option automatisch verwendet.

Sollte man aktivieren wenn die CPU nicht voll ausgelastet werden.

2.5.9 --no-asm

Diese Option ist für den normalen user eher uninteressant, da sie es dem User ermöglicht etwelige CPU Optimierungen zu deaktivieren, was man nur machen sollte, falls man einen Fehler in diesen Optimierungen vermutet.

Sollte normalerweise weglassen werden.

2.5.10 --no-psnr

Standardmäßig speichert x264 immer noch einen Stapel PSNR Werte um einen Vergleich mit dem Referenceencoder zu ermöglichen, da man dies aber normalerweise nicht macht, kann man dieses Feature ruhig aktivieren.

Sollte normalerweise mit --no-psnr aufgerufen werden.

2.5.11 --no-ssim

Analog zu 2.5.9. nur für SSIM.

Sollte normalerweise mit --no-ssim aufgerufen werden.

2.5.12 --quiet

Dieser Parameter sagt dem Encoder, dass er keine unnötigen Ausgaben werden des Encodens machen sollte.

Sollte normalerweise mit --quiet aufgerufen werden.

2.5.13 -v, --verbose

Aktiviert man diese Option wird zu jedem einzelnen Frame, das encoded wird, ein Stapel an Informationen gepostet, welche man normalerweise nicht braucht. Nur wenn man statistische Informationen über die Frames braucht sollte man dieses Feature aktivieren.

Sollte normalerweise weglassen werden.

2.5.14 --progress

Aktiviert man dieses Feature, so gibt x264 in der Kommandozeile einen Report ab wie weit es schon mit dem Encoden ist. Dieses Feature ist nur interessant, wenn man nicht irgendein Frontend verwendet, welches diese Anzeige automatisch macht.

Muss man selber wissen ob man wissen will wie weit der Encodingprozess schon gelaufen ist..

2.5.15 --visualize

Falls x264 mit den entsprechenden Sourcen compiliert ist und das Betriebssystem dies unterstützt (geht unter Windows ?noch? Nicht, jedoch unter X11) wird sobald man diese Option aktiviert beim Encoden ein weiteres Fenster geöffnet in dem eine Übersicht gepostet wird wie jedes einzelne Frame encoded wird. Rot/rosa werden hier intra Blöcke, blau werden inter Blöcke, grün werden übersprungene Blöcke und gelb werden b-Blöcke visualisiert.

Sollte normalerweise weglassen werden.

2.5.16 --sps-id

Diese Option soll ermöglichen es dem User die 'sequence parameter set id' und die 'picture parameter set id' eines Streams auf Null zu setzen. Dies soll es ermöglichen Clips die mit unterschiedlichen Parametern encoded worden sind zu verbinden.

Sollte normalerweise weglassen werden.

2.5.17 --aud

Aktiviert die Nutzung von 'Access unit delimiters', hab aber noch keinen Plan wofür die gut sind.

Sollte normalerweise weglassen werden.

3. Zu diesem Dokument

Dieses Dokument wurde grundsätzlich nachts bzw. früh morgens nach längeren, Gedächtnislücken hervorrufenden, Kneipenbesuchen unter dem Einfluss von (etwa 42) Aufputzmitteln, deren Verfallsdatum (1994) abgelaufen war, und einigen weiteren 'Hilfsmitteln', die ich hier nicht aufführen darf (Strafverfolgung), dem Meer des Chaos entwendet. Auch wegen oben erläuteter Entstehungsgeschichte übernehme ich weder implizit noch explizit eine Garantie dafür, dass dieses Dokument einen wie auch immer gearteten (sinnvollen) Zweck erfüllt. Auch lehne ich jede Verantwortung oder Haftung bezogen auf etwaige durch dieses Dokument entstandene Schäden ab.

=> Wenn ihr glaubt, was ich schreibe, selber schuld. Hiermit solltet ihr gewarnt sein.