Wissenswertes rund um x264 - 0.1.6

(by Selur)

Da mit SVN Version 581 offiziell die VFW Schnittstelle für x264 nicht weiterentwickelt wird bis sich ein neuer Entwickler findet, der sich darum kümmern will, ist dies vorerst das letzte Wissenswertes rund um x264.

Hoffe dem ein oder anderen war/ist es hilfreich.

Cu Selur

1 Vorwort:

x264 ist eine Bibliothek für das Komprimieren von Videostreams in das H.264/AVC Videoformat. Anders als bei Xvid, DivX handelt es sich bei x264 nicht um eine Implementierung eines MPEG-4-ASP-Codecs (advanced simple profile codecs), sondern um eine Implementierung des MPEG-4 AVC Zweiges des MPEG-4 ISO-Standards (ISO/IEC 14496 Part 10). Mit x264 erstellte Dateien kann man also nicht mit Standalone-Playern abspielen, die nur DivX bzw. MPEG-4 -ASP (advanced simple profile) unterstützen. (MPEG = Motion Picture Expert Group)

Anzumerken ist, dass ich in diesem Dokument nicht über die unter der GPL (Gnu Public Licence) veröffentlichte Bibliothek, sondern über den auf ihr aufbauenden VFW-Codec (Video for Windows) schreiben werde. Sicher wäre auch eine gute Dokumentation der Bibliothek als solche interessant, dies überlasse ich aber lieber anderen.

1.1 Was spricht gegen/für H.264 aka. AVC?

Obwohl MPEG-4-AVC seine Vorteile hat, ist es nicht für jederman wirklich geeignet. Um einen kleinen Anstoß zu geben, ob man H.264 anstelle von MPEG-4-ASP verwenden sollte, folgen ein paar Gedanken dazu. (Falls jemand noch andere Pro/Contra Argumente hat, immer her damit.)

1.1.1 Was spricht gegen AVC aka. H.264?

1. Es wird einiges mehr an Prozessorleistung zum Abspielen eines Clips benötig, der mit H.264 anstatt mit ASP encodet wurde. (H.264 etwa ca.1000Mhz+, ASP ca. 450Mhz+; Aufösung: 720*576; bei idealer Konfiguration)
2. Es gibt (bis dato) nur wenige Tools, die mit H.264 und der .mp4-Hülle (die man verwenden sollte) umgehen können,
3. (Noch) kein Support in Standalone-Playern; (HTPC = Home Theater Pcs mal nicht betrachtet)
4. Das Encoden, Erstellen einer Datei, dauert einiges länger bei gleicher CPU.

=> Nur wenn man genug Prozessorleistung zur Verfügung hat, sich notfalls zum Editieren mit Tools für die Eingabeaufforderung rumschlagen will und (momentan) keinen Support für Standalone-Player braucht, ist H.264 zu empfehlen.

1.1.2 Was spricht für AVC aka. H.264?

1. Ca. 30% kleinere Dateigröße bei gleicher Qualität, durch Loopfiltering, adaptive Blockgrößen ...
2. Bei H.264s Integer Transformation treten im Vergleich zu DCT keine Rundungsfehler auf,
3. Bessere Qualitätsabstufung, da mehr Quantizerschritte.
4. Keine Probleme mit unterschiedlichen IDCT-Implementationen wie es sie bei Mpeg4 gibt.

Anmerkung:
Pluspunkte wie die Möglichkeit Menüs, Kapitelpunkte usw. zu erstellen, führe ich nicht auf, da man dies auch für MPEG-4-ASP-Material erstellen kann, wenn man die entsprechenden Tools und die mp4-Dateihülle verwendet.

1.2 Warum x264 und nicht andere H.264-Implementierungen?

Da es neben dem (VFW-Codec) x264 auch noch andere Möglichkeiten gibt, H.264 Videostreams zu erzeugen, kommt eventuell der Gedanke auf, dass man auch ein anderes 'Programm' verwenden könnte. Ein gut Zusammenstellung von Alternativen zu x264 findet man z.B. im englischen Doom9 Forum[1].Angemerkt sei hier, dass momentan Qualitativ x264 und Nero hier die Spitzenreiter sind.

Für x264 spricht vor allem, dass es ein für den User kostenloses Open Source Projekt unter der GPL ist und die Qualität des Codecs sich immer mehr verbessert.

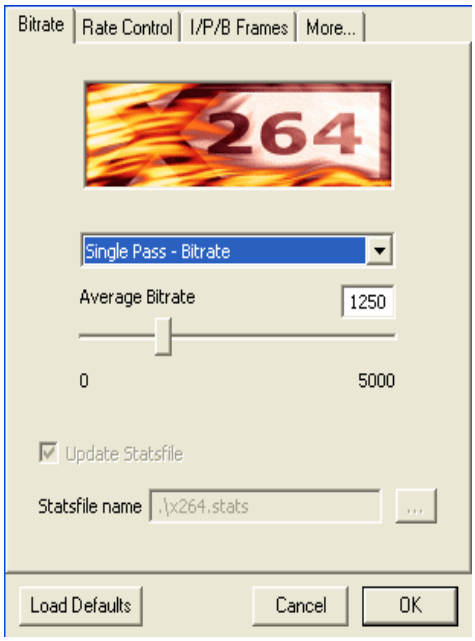
1.3 Wo kriegt man aktuelle Versionen von x264s VFW-Codec her?

Die momentan verlässlichsten Quellen für aktuelle builds des VFW-Codec von x264 sind bei im englischen Doom9 Forum (http://forum.doom9.org/showthread.php?s=&threadid=89979) und auf Jarods Hompage (http://x264.nl/) zu finden. Ich werde versuchen mich in diesem Dokument immer auf den neusten build aus zu beziehen. Falls also die Abbildungen und der Feature Umfang in diesem Dokument nicht mit der Version übereinstimmt, die man selber hat sollte man eventuell checken ob es ich neuere Versionen gibt.

So, nachdem jetzt nur noch die Leute weiterlesen sollten, die sich sicher sind, dass sie mit etwas schlechterer Qualität als Ateme leben können, werde ich auf den nächsten paar Seiten kurze Erklärungen zu den Einstellungsmöglichkeiten des VFW- x264-Codecs geben.

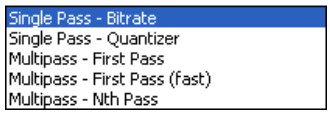
1 Unter http://forum.doom9.org/showthread.php?p=673420#post673420

2 Bitrate:



Auf der ersten Seite der graphischen Benutzeroberfläche des VFW x264 Codecs kann man zwischen mehreren 'Encoding'-Typen, kleine Einstellungen zur Statistikdatei machen, zu erweiterten Einstellungen über die Reiter More..., I/P/B Frames und Rate Control gelangen. Was der OK- und der Cancel- Buttonmachen, ist hoffentlich klar. Der Load Defaults-Button ist dazu da, wieder die Standardeinstellungen des Codecs zu laden, falls man sie mal verstellt hat.

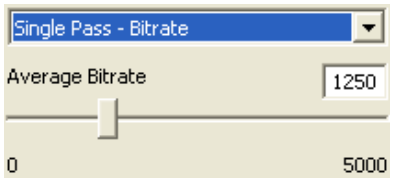
2.1 Encoding-Typen:



Die Encoding Typen werden in 'Single Pass – Bitrate', 'Single Pass – Quantizer', 'Multipass – First Pass', 'Multipass – First Pass (fast)' und 'Multipass – Nth Pass' unterteilt.

"Single Pass" -Verfahren sind vor allem für Live-Aufnahmen (z.B. beim Capturen von analogen Quellen) gedacht, auch wenn man momentan vor allem auf Grund der CPU-Anforderung H.264 nicht zum Capturen verwenden sollte.

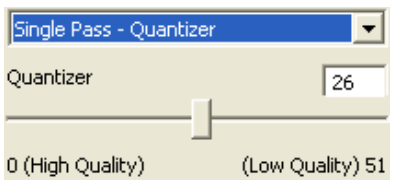
2.1.1 Single pass – Bitrate:



Beim "Single Pass – Bitrate"-Verfahren versucht x264 möglichst die vorgegebene *durchschnittliche* Datenrate zu erreichen. Zu empfehlen ist der Single Pass-Bitrate Mode vor allem wenn man nur einen Pass durchführen möchte, jedoch eine in etwa konstante Datenrate einhalten will.
x264 erlaubt dem User hierbei Werte im Bereich von 0 bis 5000 Kbit/s. (K = 1000)
Zu bedenken ist, dass i.d.R. eine konstante Datenrate nicht auch eine konstante Qualität zur Folge hat. "Single Pass – Bitrate"-Encodes tendieren dazu, (meist stark) schwankende Qualität bei sich stark ändernden Szenen zu liefern.

Sollte man verwenden wenn es schnell gehen soll und die Datenrate wichtig ist.

2.1.2 Single pass – Quantizer:

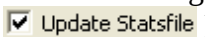


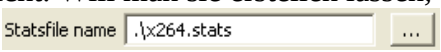
Quantizer geben eine Art Qualitätsmaß an. Ein bestimmter Quantizer legt fest, wieviel Daten-/Informationverluste im Vergleich zum Orginal auftreten dürfen. Je niedriger der Quantizer, desto weniger und je höher der Quantizer, desto mehr Informationen können verloren gehen.
Hierbei wird aber nur eine Schranke für den maximalen Verlust festgelegt, ein höherer Quantizer muss also nicht zwangsweise zu mehr Verlusten führen.

Beim **Single pass – Quantizer** Verfahren encodet x264 alle Daten mit einem festen Quantizer und garantiert so eine Schranke für den Qualitätsverlust. Vor allem wenn man eine Quelle umwandelt, die man später weiter verarbeiten will, ist dieser Mode zu empfehlen. Anders als bei "Single pass – Bitrate"- Mode ist jedoch keine durchschnittliche Datenrate garantiert, ein normaler "Single pass – Quantizer"-Encoding wird (meist) eine stark schwankende Datenrate aufweisen und so keine Vorhersagen über die Zielgröße zulassen.

Anmerkung: Um (etwa) den äquivalenten H.264-Quantizer zu einem MPEG-Quantizer zu finden hilft folgende Formel: H.264QP = 12 + 6*log2(MPEGQP). Ein Quantizer von 2 bei Xvid oder DivX würde also etwa einem H.264-Quantizer von 18 entsprechen.

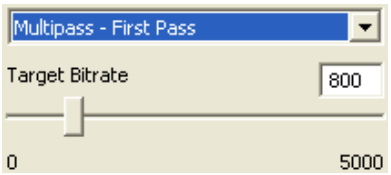
Sollte man verwenden wenn es schnell gehen soll und eine konstante Verlustgrenze gewünscht ist.

Anders als beim den *'Single Pass'*-Verfahren wird bei einem *Multipass*-Encode versucht die subjektive Qualität möglichst konstant zu halten und gleichzeitig die gewünschte durchschnittliche Zieldatenrate zu erreichen. Bevor wir nun zu den einzelnen *Multipass*-Varianten kommen noch etwas zur Statistikdatei, x264 kann während jedes *Multipass*-Durchlaufes eine Statistikdatei erstellen, auf welche ein etweiliger folgdender 'Multipass – Nth Pass' zugreifen kann um besser zu encoden. Mittels 

kann man entscheidenob eine solche Statistikdatei erstellt wird oder nicht. Will man sie erstellen lassen, so wird der Ort, an dem diese Statistikdatei erstellt wird, durch 

festlegt. Das erzeugen einer Statistikdatei ist nur von nöten, wenn geplant ist, dass noch weitere Durchläufe geplant sind.

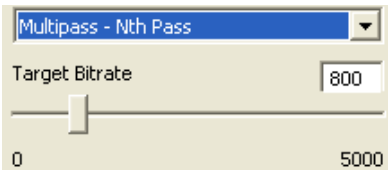
2.1.3 Multipass - First pass (fast):



Wählt man den *First pass (fast)* anstatt dem normalen *First pass* werden wird bei der Analyse des Quellmaterial ungenauer vorgegangen. Es werden keine P-Frame - Makroaufteilungen der Größe 8x8, keine B-Frame-Makroblöcke der Größe 16x16 betrachtet und auch wird die Anzahl der **Max reference Frames** während der Analyse **wird** halbiert (abgerundet) und die **Subpixel refinement precision** wird auf 3 reduziert falls der eigentliche Wert größer als 3 ist oder nur um eins verringert, falls der eigentliche Wert kleiner oder gleich 3 ist.
Die *First pass (fast)*-Option liefert zwar eine etwas ungenauere Analyse des Quellmaterials, jedoch beschleunigt sie den *First pass* ungemein. Vor allem wenn man 3 oder mehr Durchläufe macht, sieht man keinen Unterschied ob am Anfang ein normaler oder ein *First Pass (fast)* oder ein normaler *First pass* verwendet wurde. Führt man nur zwei Encodingdurchgänge durch, vermindert sich die Qualität etwas, jedoch nicht stark, so dass ich normalerweise empfehlen würde die Option zu aktivieren.

Muss immer vor etwaige weiteren "Nth pass" - Durchgängen durchgeführt werden und die Option 'Update statsfile' muß aktiviert sein damit der First pass etwas bringt.

2.1.4 Nth pass:

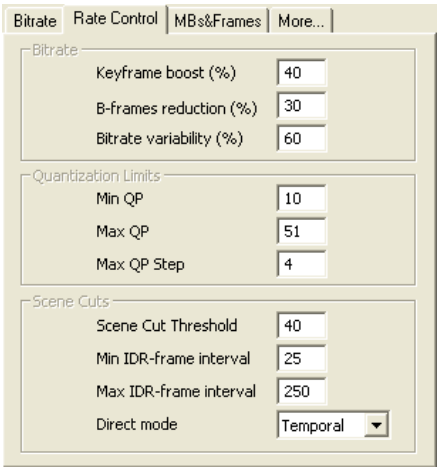


Anders als im ersten Durchgang, in dem nur eine Analyse der Daten stattfand, wird in allen weiteren Durchgängen nicht nur eine Analyse durchgeführt, sondern auch eine Ausgabe erzeugt. Will man mehr als zwei Durchgänge machen, sollte man auch die Option **Update statsfile** aktivieren, damit eine neue Statistikdatei erstellt wird auf die der folgende Durchlauf zugreifen kann.
Damit bei einem Computerabsturz nur der letzte Durchlauf wiederholt werden muss, hängt x264 an das Ende des unter **Statsfile name** festgelegten Namens immer ein "-Nummer des Durchlaufes" an. Die Statistikdatei des 3ten Durchlaufes würde also *x264-3.stats* heißen.
Allgemein sind mehr als 2 Durchläufe nur zu empfehlen, wenn man eine sehr schwere oder nur wenige Sekunden lange Sequenz encodet, da x264 in diesen Fällen meist 3 oder 4 Passes braucht um die gewünschte durschnittliche Datenrate hinreichend genau zu treffen.
Wichtig ist, dass man keine der Einstellungen zwischen den einzelnen Durchläufen ändert, da x264 die Datenrate ansonsten nicht optimal skalieren kann.

Ein bis zwei solcher Durchgänge, nach dem First Pass, sollte der normale Standard sein und die 'Update statsfile' Option sollte in allen, außer dem letzen Durchgangn, aktiviert sein.

Nachdem wir uns die grundlegenden Encodingtypen von x264 angeguckt haben wird es nun Zeit sich über die *Rate Control* gedanken zu machen.

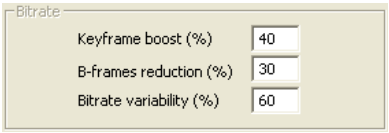
3 Rate Control:



In diesem Bereich geht es darum direkt die Bitraten-, Quantizerverteilung und die Szenenwechselerkennung des Codecs zu manipulieren. Normalerweise sollte man diese Werte nicht ändern müßen. Damit man wenn es aber doch mal nötig seien sollte, hier eine Änderung vorzunehmen, weiß was die Optionen machen:

3.1 Bitrate:

Mit den drei folgenden Einstellungen wird direkt die Datenratenverteilung beeinflußt.



Keyframe boost(%) legt fest, um wieviel Prozent bei der Datenratenverteilung Keyframes bevorzugt werden. Die Standardwerte zwischen 30% und 45% haben bis dato bei mir die besten Ergebnisse geliefert.

B-Frame reduction(%) legt fest, um wieviel die Datenrate von B-Frames unter der von anderen Frames liegen sollte. Zu große Werte führen hier zu hohen Quantizern in den B-Frames und glätten das Bild zu stark, wohingegen zu niedrige Werte unnötig Datenrate verschwenden. Allgemein sollte man zwischen 15% bei sehr hohen Datenraten und 50% bei niedrigen Datenraten bleiben.

Bitrate variability(%) gibt an, wie stark die Bitrate über dem ganzen Film schwanken darf. Ein Wert von 0 würde zu einem Encode mit konstanter Datenrate und ein Wert von 100 zu einem Encode mit konstantem Quantizer führen. Da meist eher eine konstante 'Qualität' gewünscht ist, würde ich Werte zwischen 50% und 70% empfehlen.

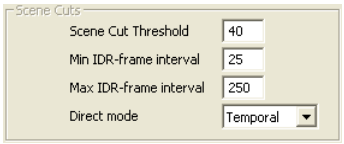
Die Standardwerte sollten normalerweise nicht verändert werden.

3.2 Quality:

Die Einstellungen hier wirken sich direkt auf die Quantizerverteilung des Codecs aus. **Min QP** legt eine untere Schranke für den kleinsten Quantizer fest der bein 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. Auch **Max QP** kann man durchaus auf 42 senken, da die höheren Quantizer meist nur bei relativ niedrigen Datenraten benötigt werden. **Max QP Step** legt fest wie stark die Quantizerschwankung zwischen zwei benachbarten Frames seien 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.

Die Standardwerte sollten normalerweise nicht verändert werden.

3.3 Scene Cuts:



Anders als bei anderen MPEG-Varianten können sich auf Grund der möglichen multiplen Referenzframes Frames auch auf andere Frames beziehen, die noch vor dem ihnen am nächsten gelegenen I-Frame liegen. D.h. also, dass I-Frames nicht mehr automatisch 'echte' Keyframes sind. 'Echte' Keyframes sind jedoch zum Spulen/Springen im einem Videostream notwendig, da sie einem garantieren, dass alle Frames hinter ihnen sich auf kein Frame vor ihnen beziehen, was die
Anzahl der im Speicher zu bewahrenden Frames einschränkt. Diese 'echten' Keyframes werden im H.264-Standard IDR -(Key-)Frames (IDR = instantaneous decoding refresh) genannt. Bei diesen sollte der Decoder seinen Speicher leeren.

3.3.1 Scene Cut Threshold:

Dieser Schwellenwert legt fest ab wieviel Prozent an Bildänderung ein neues Bild als Szenenwechsel erkannt wird, was direkt beeinflusst ob ein I-Frame gesetzt wird oder nicht. Das Problem ist, dass I-Frames idealerweise immer bei stärkeren Änderungen als qualitativhochwertige Referenzframes gebraucht werden, sie jedoch einiges mehr an Datenrate benötigen als P- und B-Frames.

Sollte man auf dem Standardwert lassen.

3.3.2 Min IDR-keyframe interval:

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 sort dafür, dass der Codec jedes I-Frame automatisch zu einem IDR-Frame machen. Dieses vom Programmierer festgelegte verhalten würde H.264 um die Möglichkeit multipler Referenzframes über mehrere I-Frames hinweg berauben würde. Allgemein wird von den Programmierern ein Interval von 0.4***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.

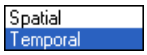
Ich nehme normalerweise die Framerate als Wert.

3.3.3 Max IDR-keyframe interval:

Das hier festzulegende Intervall gibt an, nach wievielen 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)

Sollte normalerweise 250 sein.

3.3.4 Direct Mode:

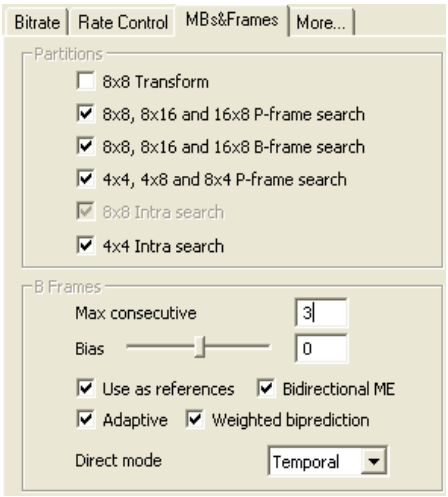


Hierbei handelt es sich um die Art zu ermittelen, ob die Motionvektorendifferenzen der Makroblöcke eines B-Frames mit benachbarten Böcken im Frame selber (**spatial**) oder mit den Motionvektoren der Nachbarframes (**temporal**) gebildet werden sollen. **Spatial** liefert (meist) bessere PSNR (peak signal-to-noise ratio) Werte und **temporal** sieht meist, meiner Meinung nach, etwas besser aus.

Ich würde eher temporal empfehlen.

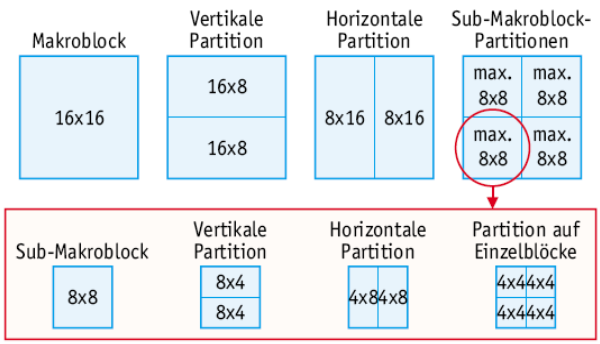
So, nachdem wir nun auch etwas zu den Möglichkeiten zur Manipulation der Rate Control im x264 Codec betrachtet haben geht es direkt weiter zur Analyse der I- P- und B-Frames.

4 MBs&Frames:



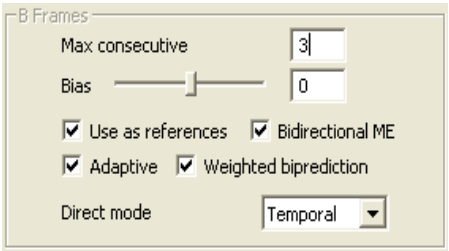
MPEG-Komprimierungsverfahren basieren stark darauf Ähnlichkeiten in anderen Bildern zu finden und dadurch Platz zu sparen, dass man nur noch Referenzen auf Teile in anderen Bildern speichert. In diesem Bereich hat der User nun die Möglichkeit festzulegen wie genau die Unterteilung der Bilder/Frames geschehen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

4.2 Partitions:



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. Die Optionen **8x8 Transform** und **8x8 Intra search** gehören zum High Profile des Mpeg4 AVC Standard und bringen zwar einen kleinen Qualitätsgewinn, sollten aber nur akitviert werden, wenn man sicher ist, dass etwaige Decoder/Player auch mit dem High Profile umgehen können.Mit der Option **8x8 Transform** festlegen ob eine 8x8 DCT Transformation aktiviert wird, so steht auch imdie Option **8x8 Intra search** zur Verfügung.

4.3 B-Frames:



B-Frames betrachtet bei der Berechnung nicht nur vorangegangene, sondern auch nachfolgende Frames, was meist zu beachtlichen Datenrateneinsparungen führt. Die Standardeinstellungen betreffend B-Frames sollten nur geändert werden, wenn man weiß was man tut. Sie sind so gewählt, dass sie in den meisten (normalen) Szenarien gute Ergebnisse liefern.

4.3.1 Use as references:

Erst wenn man dieses Feature aktiviert können Frames auch wieder auf B-Frames referenzieren, was nochmals einen kleinen Qualitätsgewinn bringen, jedoch auch die CPU Anforderung steigern sollte. Allgemein würde ich empfehlen dieses Feature zu aktivieren wenn man B-Frames verwendet und der Decoder den man zur Wiedergabe verwendet dieses Feature unterstützt.

Sollte man aktivieren, da es hilft und von besseren Decodern unterstützt wird.

4.3.2 Bidirectional ME:

Bidirectionnal 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. Diese Option sollte normalerweise aktiviert werden, da es die Effizienz von B-Frames nochmal etwas erhöht.

Sollte man aktivieren, da es die Effizienz der B-Frames erhöht..

4.3.3 Adaptive:

Ist die adaptive B-Frame-Vergabe aktiviert bekommen langsamere Szenen mehr B-Frames als schnellere/aktionhaltigere Szenen, was auch sinnig ist, da schnellere Szenen mit einem höheren Quantizer encoded werden als langsamere. Deaktiviert man das adaptive Verteilen der B-Frames werden die B-Frames unabhäniger davon gesetzt ob eine schnelle oder Langsame Szene vorliegt. Vorallem wenn man mehr als drei B-Frames verwendet kann dies jedoch einen unangenehmen Einfluß auf die Qualität haben.

Standardmäßig sollte man Adaptive aktiviert lassen.

4.3.4 Weighted biprediction:

Wird die weighted biprediction aktiviert, so werden Bildinhalte aus mehreren Referenz-Frames gemischt und können in der Quellen beliebig gewichtet in die Mischung eingehen. Dies ist vorallem hilfreich um Aus- und Überblendungen effizienter zu speichern.

Sollte aktiviert werden.

4.3.5 Max consecutive:

Dieses Feature legt fest, wie viele B-Frames (Bidirektionale Frames) hintereinander erlaubt sind. Durch die Möglichkeit der Bidirektionalenreferenz 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.

Ich würde zwei bis drei B-Frames für normale Encodes empfehlen.

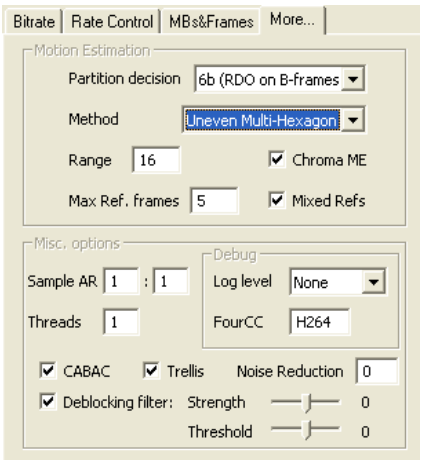
4.3.5 Bias:

Bei diesem Feature kann der User die prozentuale Wahrscheinlichkeit erhöhen/senken, dass anstatt einem P- ein B-Frame verwendet wird. Das unter **Max consecutive** festgelegte Maximum wird hierdurch jedoch nicht überschritten. Es ist vielmehr die Frage, wie schnell 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.

Sollte man normalerweise auf 0 lassen.

So nachdem wir nun den MBs&Frame Bereich durchgegangen sind, geht es nun weiter zum | More... | - Reiter.

5 More ...

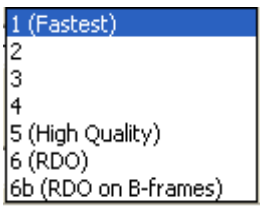


In diesem Bereich geht es um die Bewegungsanalyse, Multithreading, Deblocking und Debugging.

5.1 Motion Estimation:

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.

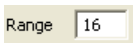
5.1.1 Partition decision:



Diese Optionen, legt fest wie detailiert die Sub-Makroblockverfeinerung benutzt wird. Bei **Level 1** wird erst eine Bewegungsanalyse auf jedem ganzen Pixel gemacht. Auf den besten Kanidaten 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. Bei **Level 6b** wird die Rate Distortion Optimization auch für B-Frames angewandt.

Als Standard würde ich 4 oder 5 empfehen und wenn man die Rechenleistung hat liefert 6b die beste Qualität.

5.1.2 Method:

Hier kann man zwischen den eigentlichen Suchmethoden wählen mit denen die Bewegundsvektoren ermittelt werden. Ohne genau darauf eingehen zu wollen wie die einzelnen Methoden genau funktionieren.
Die einzelnen Suchmethoden von oben nach unten immer genauer und langsamer werden und die als Standard gewählte **Hexagonal Search** normalerweise das Beste Geschwindigkeits-/ Qualitätsverhältnis liefert. Aktiviert man die **Uneven Multi-Hexagon** Suche, kann man den Radius (Range) ändern über den sie geht und wird sicher bessere Ergebnisse erziehen, jedoch macht dies meist nur etwas 0.1 PSNR-Punkte aus, was den zusätzlichen Rechenaufwand weder hier noch im Fall einer kompletten Suche (**Exhaustive Search**) rechtfertigt. Wählt man die **Uneven Multi-Hexagon** Suche oder die **Exhaustive Search**, so hat man bei 

auch die Möglichkeit den Pixelradius vom Standardwert von 16 zu ändern, was aber meiner Erfahrugn anch wenig bis keinen Sinn macht. **Diamond Search** sollte man nur benutzen wenn es schnell gehen muß.

Sollte man auf Hexagonal Search stehen lassen.

5.1.4 Chroma ME:

Ist dieses Feature nicht 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. Dann kann man mit Hilfe dieser Option zusätzlich zur Helligkeits- auch die Farb-Komponenten zur Ermittlung der Bewegungsrichtung verwenden.

Sollte normalerweise zu Gunsten genauerer Bewegungsabschätzungen aktiviert sein.

5.1.5 Max Ref. Frames:

Die Option Max Ref. Frames legt fest wieviele andere Bilder auf der Suche nach Ähnlichkeiten durchsucht werden. 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 erziehlt 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.

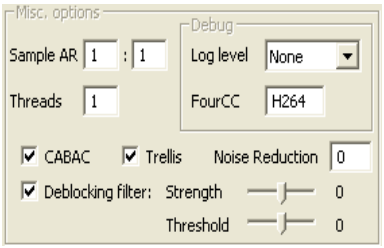
Sollte normalerweise nur bei niedrigen Auflösungen mit einem Wert von >5 belegt werden.

5.1.6 Mixed Refs:

Die Option Mixed Refs legt 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 auf 10% mehr oder weniger auch nicht mehr ankommt.

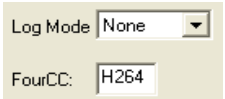
Sollte normalerweise aktiviert sein.

5.2 Misc options:



5.2.1 Debug:

Im Debugbereich kann man den **Log Level** und den **FourCC** einstellen.



5.2.1.1 Log Level:

Durch den **Log Level** legt man fest wieviele Informationen zum allgemeinen Ablauf von x264 mit protokoliert werden, dies ist vorallem dafür gedacht um den Entwicklern zu helfen etwaige Fehler in x264 zu finden und zu beheben.
Mit **None** deaktiviert man den Log Mode, was die Standardeinstellung seien sollte, falls man keine Probleme hat oder die Entwickler ausdrücklich darum gebeten haben diesen Mode nicht zu nutzen. Im **Error**-Mode werden neben kritischen Fehlern vor allem Fehler aufgezeichnet, die zwar nicht zu abstürzen führen, aber durchaus auf ein Fehlverhalten von x264 hinweisen. Im **Warning**-Mode werden nur Warnungen archiviert, die auf kleine Rechenfehler und dergleichen hinweisen, i.d.R. aber nicht schlimm seien sollten. Im **Info**-Mode werden detailierte diagnostische Informationen gesammelt. Im **Debug**-Mode werden alle x264 möglichen Informationen für eine Fehlersucher gespeichert. Zu beachten ist, dass je nach Einstellung der Encodingprozess um einiges gebremst werden kann.

Sollte man normalerweise auf None stehen haben.

5.2.1.2 FourCC:

FourCC steht für Four Character Code und weist dem Videostream (in .avi-Dateien) einen Decoder zu. Standardmäßig wird momentan H.264 als FourCC verwendet, da sowohl der Nero-/ Ateme-Decoder als auch ffdshow diesen unterstützen.

Sollte auf dem Standardwert H264 belassen werden.

5.2.2 Sample AR (Aspect Ratio):

Das **Sample AR (SAR)** gibt das Verhältnis von Breite zu Höhe **eines Pixels** (16x16 Bildpunkte) in einem Bild an. Wenn man nicht anamorph encoden will sollte man hier auf 1:1 bleiben.[2]

Sollte auf dem Standardwert 1:1 belassen werden.

5.2.3 Threads:

Hinter der Option 

verbirgt sich die Möglichkeit dem Codec zu sagen, dass einige der internen Berechnungen so ausgeführt werden, dass sie separat berechnet werden können. Die 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. 4 Threads unterstützt.

Anzahl der (simulierten) Prozessoren, sollte hier der Standard sein.

2 Weiter Informationen zu anamorphen Encoding findet man z.B. hier: http://encodingwissen.brother-john.net/anamorph_intro.html

5.2.4 CABAC:

H.264 unterstützt dabei neben CAVLC (Context Adaptive Variable Length Coding), einer Huffman-artigen, auch eine 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 beieindruckendste Feature bei x264, was die dahinterstehende Technik/Idee betrifft. Was für alle Leser aber interessant ist, ist, dass CABAC relativ langsam ist, aber ca. 10% an Datenrateneinsparungen bringt, und deshalb definitv aktiviert werden sollte. Nur wenn man x264 zum Capturen benutzt, sollte man eventuell von CABAC Abstand nehmen.

Sollte aktiviert werden auch wenn es Prozessorpower fordert.

5.2.5 Trellis:

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. Aktiviert man diese Option wird eine Trellisquantisation für den entgültigen Motionblock durchgeführt.

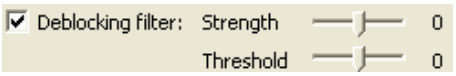
Bringt nicht viel, aber sollte wenn es auf die Qualität ankommt aktiviert sein.

5.2.6 Noise reduction:

Dieser Option ermöglicht es dem User eine Noise Reduction/Rauschunterdrückung in x264 zu nutzen um so vielleicht etweilliges Postprocessing des Inputs sich zu sparen. Sinnige Werte bei normalen Quellen liegen so zwischen 0 und 600, höhere Werte (bis 10 000) sind nur bei sehr verrauschtem Material sinnig. Hier sollte man ersteinmal ein bißchen rumtesten ehe man sich hier auf einen Wert festlagt. Persönlich mag ich es nicht zusätlich zum Loopfilter noch ein weiteres glättendes Element zu habne was ich nicht genau vorhersagen kann.

Sollte normalerweise auf 0 belassen werden.

5.2.7 Deblocking Filter:



Aktiviert man diese Option wird, eine der Stärken von H.264 angewandt: der Inloop-Deblocking-Filter, der dafür sorgt, dass auch bei starker Kompression möglichst keine Makroblockübergänge zu sehen sind, wie es bei anderen MPEG-Varianten der Fall ist. 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 zumVerständnis sagen, **Strength** legt die Stärke der Glättung fest und **Threshold** legt fest, ab wann geglättet wird. Würde man Strength z.B. Auf 5 setzen, so würde ein Quantizer 16 Makroblock so stark geglättet wie normalerweise ein Quantizer 21 Block. Stellt man den Threshold auf -5 so würde schon ab Quantizer 11 geglättet,.. Je nach Zielgröße sollte man die Stärke und den Schwellenwert etwas verändern. Persönlich lasse ich bei 1-CD-Sicherheitskopien einer DVD beideWerte auf 0 und stelle bei 2 CDs oder mehr Strength(B) 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.

Sollte man je nach Qualitätsempfinden leicht anpassen.

6 Changelog:

0.1.5 nach 0.1.6
- Kapitel 1.2 überarbeitet
- vorerst letzte Version da VFW Schnittstelle nicht mehr gepflegt wird

0.1.4 nach 0.1.5
- Neue Features: trellis, rdo für b-frames, bidirektionale ME für B-Frames

0.1.3 nach 0.1.4
- Frontend Änderung, Restrukturierung

0.1.2 nach 0.1.3
- Frontend Änderung
- Typos&Co

0.1.1 nach 0.1.2
- Aufbau des Frontends hat sich geändert
- weighted prediction ist wieder da

0.1.0 nach 0.1.1
- Partition decision quality wurde um RDO erweitert.

0.0.9 nach 0.1.0
- Typos, kleine Umformulierungen
- Link zu Doom9 letztem Codecvergleich eingefügt
- komplettes Redesign des Advanced Bereichs, mit einigen neuen Features

0.0.8 nach 0.0.9
- Pixel Aspect Ratio, 8x8 DCT und 8x8 Intra Search wurde in die GUI aufgenommen

0.0.7 nach 0.0.8
- max reference Frames max = 16 nicht mehr 15Referenzen
- Jarods Homepage als weiter x264 Quelle angegeben
- kurze Beschreibung zum Thread Support

0.0.6 nach 0.0.7
- GUI im Advanced Featurebereich wurde umgestaltet
- Weighted Prediction
- Deblocking Threshold wurde entfernt
- Meine Empfehlungen farblich in Grau eingefügt

0.0.5 nach 0.0.6
- Deblocking Filter Strength(A) und Strength(B) heißen nun auch Strength und Threshold
- B-Frame Pyramide

0.0.4 nach 0.0.5
- Grammatik- und Rechtschreibfehler minimieren
- Doom9 Forum als alternative x264 Quelle eingetragen
- multiple B-Frames scheinen nun zu fluppen

0.0.3 nach 0.0.4

- Grammatik- und Rechtschreibfehler minimieren
- min IDR umgeschrieben; min nicht zu klein wegen Heuristik
- B-Frame prediction Mode umgeschrieben

0.0.2 nach 0.0.3
- Grammatik- und Rechtschreibfehler minimieren; Dank an Nexus.
- Grafiken vergrößert

0.0.1 nach 0.0.2:
- B-Frame prediction verbessert
- subpixel refinement Erläuterung erweitert
- max reference frames auf 15 verbessert
- min IDR-keyframe interval Verwirrung
- einige Verbesserungen aus dem deutschen Gleitz/Doom9 Forum
- da noch keine adaptive B-Frame decision unterstützt wird => max b-frames = 1

0.0.0 nach 0.0.1:
- erste Grundversion, nicht sehr ausführlich aber hoffentlich hilfreich

7 Abschließende Worte:

Dieses Dokument wurde grundsätzlich nachts bzw. früh morgens nach längeren, Gedächtnislücken hervorrufenden, Kneipenbesuchen unter dem Einfluss von (etwa 42) Aufputschmitteln, deren Verfallsdatum (1994) abgelaufen war, und einigen weiteren 'Hilfsmitteln', die ich hier nicht aufführen darf (Strafverfolgung), dem Meer des Chaos entwendet. Auch wegen oben erläuterter 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.