

Bruno George de Moraes

*Uma métrica para Taxa de Distorção voltada para
codificação de vídeo perceptiva*

Florianópolis, Santa Catarina

2010.1

Bruno George de Moraes

*Uma métrica para Taxa de Distorção voltada para
codificação de vídeo perceptiva*

Trabalho de Conclusão de Curso apresentado
como parte dos requisitos para obtenção do grau
de Bacharel em Ciências da Computação.

Orientador:

Prof. Dr. José Luís Almada Güntzel

UNIVERSIDADE FEDERAL DE SANTA CATARINA
CENTRO TECNOLÓGICO
DEPARTAMENTO DE INFORMÁTICA E ESTATÍSTICA

Florianópolis, Santa Catarina

2010.1

Trabalho de conclusão de curso sob o título “*Uma métrica para Taxa de Distorção voltada para codificação de vídeo perceptiva*”, defendido por Bruno George de Moraes e aprovado em 31 de Maio de 2010, em Florianópolis, Santa Catarina, pela banca examinadora constituída pelos professores:

Prof. Dr. José Luís Almada Güntzel
Universidade Federal de Santa Catarina
Orientador

Prof. Dr. Luiz Cláudio Villar dos Santos
Universidade Federal de Santa Catarina
Membro da Banca

Prof. Dr. Olinto José Varela Furtado
Universidade Federal de Santa Catarina
Membro da Banca

Sumário

| | |
|-------------------------------------------------------------------|-------|
| Lista de Acrônimos | p. 5 |
| Lista de Figuras | p. 6 |
| Lista de Tabelas | p. 7 |
| Keywords: | p. 1 |
| Resumo | p. 2 |
| Palavras-chave: | p. 2 |
| 1 Introdução | p. 1 |
| 1.1 Justificativa | p. 1 |
| 2 Conceitos de Compressão de Vídeo | p. 2 |
| 2.1 Padrão H.264/AVC | p. 5 |
| 3 Métricas de Custo | p. 11 |
| 3.1 Otimização Taxa-Distorção | p. 11 |
| 3.2 Métricas de Distorção | p. 13 |
| 4 Métrica Proposta | p. 16 |
| 4.1 Compressor H.264/AVC base da implementação proposta | p. 18 |
| 5 Metodologia de Experimentação | p. 19 |
| 6 Resultados Experimentais | p. 21 |
| 6.1 Experimento A | p. 21 |
| 6.2 Experimento B | p. 24 |
| 6.3 Experimento C | p. 26 |
| 7 Conclusões | p. 29 |

| | |
|-----------------------------------|-------|
| 7.1 Trabalhos Futuros | p. 29 |
| Referências Bibliográficas | p. 30 |
| Apêndice | p. 31 |

Lista de Acrônimos

| | |
|--------------|----------------------------------------------------------------------|
| CABAC | Codificação Aritmética Binária Adaptativa ao Contexto |
| CAVLC | Codificação Adaptativa de Comprimento de Palavra Variável |
| CRF | Fator de Taxa Constante (Constant Rate Factor) |
| HD | Alta Definição (High Definition) |
| HVS | Sistema Visual Humano (Human Visual System) |
| ISO | International Standard Organization |
| ITU | International Telecommunication Union |
| MB | Macrobloco |
| MSE | Erro Médio Quadrático (Mean Squared Error) |
| MPEG | Motion Picture Experts Group |
| PSNR | Razão entre Pico de Sinal e Ruído (Peak Signal to Noise Ratio) |
| QDE | Energia de Distorção da Quantização (Quantization Distortion Energy) |
| QP | Parâmetro de Quantização (Quantization Parameter) |
| RD | Taxa-Distorção (Rate-Distortion) |
| RDO | Otimização Taxa-Distorção (Rate Distortion Optimization) |
| SAD | Somas das Diferenças Absolutas (Sum of Absolute Differences) |
| SATD | Soma das Diferenças Tranformadas Absolutas |
| SSD | Soma das Diferenças Quadráticas (Sum of Squared Differences) |
| SSIM | Índice de Similaridade Estrutural (structural similarity index) |
| DSSIM | Dissimilaridade Estrutural (structural dissimilarity) |
| VCEG | Video Coding Experts Group |

Lista de Figuras

| | | |
|------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------|-------|
| 2.1 | Amostragem de um vídeo em macrobloco 16x16 pixels. Fonte: [Manoel 2007] | p. 2 |
| 2.2 | Organização hierárquica da camada de codificação de vídeo, subdividindo o vídeo em quadros, quadros em fatias e fatias em macroblocos. Fonte: [Manoel 2007] | p. 3 |
| 2.3 | Linha do tempo das principais técnicas de compressão de vídeo. Fonte: [Reader 2002] | p. 4 |
| 2.4 | Histórico dos padrões apresentados pelos órgãos ITU-T e ISO/IEC. Fonte: [Manoel 2007] | p. 5 |
| 2.5 | Esquema de compressor H.264/AVC. Fonte: [Manoel 2007] | p. 6 |
| 2.6 | Exemplo de múltiplas partições com predição inter-quadro (verde) e intra-quadro (rosa). | p. 7 |
| 2.7 | Exemplo de compensação de movimento usando vetor 3/4 à direita e 1/2 acima. Fonte: [Manoel 2007] | p. 8 |
| 2.8 | Exemplo de múltiplas referências temporais. Fonte: [VIJAYAKUMAR 2008] | p. 8 |
| 2.9 | Exemplo de predição intra-quadro 4x4 com modo 7 selecionado. Fonte: [VIJAYAKUMAR 2008] | p. 9 |
| 2.10 | Esquema dos modos de predição Intra 4x4 e Intra 16x16. Fonte: [VIJAYAKUMAR 2008] | p. 10 |
| 3.1 | Exemplo de decisão final com imagem original (F_n), predita (F'_{n-1}), resíduo e reconstruída (F'_n). | p. 12 |
| 3.2 | Espaço de soluções possíveis e otimização Lambda. Fonte: [Terribery]. | p. 12 |
| 3.3 | As duas imagens possuem o mesmo <i>PSNR</i> , mas o ruído está distribuído em regiões diferentes [WINKLER 2005]. | p. 15 |
| 4.1 | tabela com os <i>thresholds</i> para as <i>dct</i> 4x4 e 8x8 do padrão <i>H.264/AVC</i> | p. 17 |
| 4.2 | Comparação entre diversos compressores em modo de qualidade máxima com a métrica DSSIM. Fonte: [Merritt et al.] | p. 18 |
| 5.1 | Vídeos de teste utilizados nos experimentos. Fonte: [Xiph.org] | p. 20 |
| 6.1 | Gráfico ilustrando desempenho relativo, taxa de bit e novo CRF obtido. | p. 25 |
| 6.2 | Gráfico ilustrando tamanho relativo, taxa de bit e novo CRF obtido. | p. 25 |
| 6.3 | Gráfico com curva Taxa-Distorção usando a métrica DSSIM e uma curva do ganho de desempenho. | p. 27 |
| 6.4 | Gráfico com curva Taxa-Distorção usando os CRF originais e uma curva do ganho de desempenho. | p. 28 |

Lista de Tabelas

- 4.1 Tabela de constantes MF da quantização para cada posição 4x4 [Richardson
2003] p. 17

Resumo

Current applications such as terrestrial digital TV, Blu-ray and IPTV, demand a more efficient use of available bandwidth even with the latest H.264/AVC standard currently adopted. The distortion metrics play an important role in efficiency, which are critical in decision making, due to feedback of the current image distortion during encoding. So we present their classification, acceptance criteria and finally a new distortion metric proposal. The proposed metric makes a prediction of the quantization effects, such approach improves the approximation of the real rate-distortion. There been experienced gains up to 60 % in bit rate, with a proportional gain in performance.

Keywords:

H.264/AVC, compression, video coding, x264, SSIM, RDO, distortion, perceptive coding, metric, psychovisual, subjective.

Resumo

Aplicações atuais como TV digital terrestre, Blu-ray e IPTV, demandam um uso mais eficiente da largura de banda disponível mesmo com o mais novo padrão H.264/AVC. As métricas de distorção desempenham um importante papel nessa eficiência, porque são fundamentais na tomada de decisão, devido à realimentação da distorção da imagem durante a codificação. Portanto apresentamos suas classificações, critérios de aceitação e finalmente a proposta de uma nova métrica de distorção. A métrica proposta faz uma predição dos efeitos da quantização melhorando a aproximação da taxa-distorção real. Com isso obtemos ganhos de até 60% em taxa de bit, com um proporcional ganho em desempenho.

Palavras-chave:

H.264/AVC, compressão, codificação de vídeo, x264, SSIM, RDO, distorção, codificação perceptiva, métricas, psicovisual, subjetiva.

1 Introdução

Este trabalho visa atender às necessidades de aplicações atuais como TV digital terrestre, Blu-ray e IPTV, que demandam o uso mais eficiente da largura de banda disponível. Propõe-se uma métrica de distorção (qualidade) para compressão de vídeo. Inicialmente, serão discutidas diferentes métricas de distorção, sua importância no processo de codificação e classificação, a grosso modo, em duas abordagens:

- Métricas objetivas baseadas em modelos matemáticos de processamento de sinais;
- Métricas psicovisuais ou subjetivas que usam características do sistema visual humano (HVS - Human Visual System);

Como plataforma de teste para a métrica será usado o compressor do projeto x264 [Merritt et al.], compatível com o padrão H.264/AVC. O x264 apresenta várias métricas e ajustes psicovisuais usados para obter ganhos em qualidade de imagem e desempenho [Merritt e Vanam 2007], sendo reconhecido atualmente como a implementação em software mais eficiente do padrão H.264/AVC.

1.1 Justificativa

O uso do padrão H.264/AVC tem aumentado significativamente devido principalmente ao ganho de 50% em eficiência em relação ao MPEG-2, o que garantiu sua adoção como padrão de TV digital (SBTVD , DVB-S2) e filmes como Blu-Ray, HD-DVD, AVCHD. Apesar do H.264/AVC constiuir o estado-da-arte pela sua eficiência, sua adoção em algumas aplicações ainda é inviável devido ao custo e ao limite da largura de banda disponível. Explorando diversas técnicas e métricas é possível melhorar a eficiência de compressão no codificador para o nível de banda necessário às aplicações.

2 *Conceitos de Compressão de Vídeo*

Compressão de vídeo é uma área dentro da computação que busca eliminar a redundância das imagens em movimento para viabilizar a sua transmissão e armazenamento [Agostini 2007]. Segundo Wu [Wu 2004], os principais motivos para se processar uma imagens são:

- Os dados visuais apresentam, em geral grande densidade de informações.
- Enorme quantidade de imagens nos cercam como fotografias, desenhos artísticos, de engenharia, médicas e astronômicas.
- Fontes de vídeo como TV, vigilância e filmes caseiros.

Um vídeo digital corresponde a amostras espaciais e temporais de um sinal elétrico representando a luz captada por uma câmera. Estas informações são guardadas em matrizes com um tamanho definido, onde os pixels armazenam as amostras de intensidades de brilho e cor. Em outras palavras, um vídeo pode ser imaginado como uma sequência de quadros, onde cada quadro é uma imagem estática i.e., uma fotografia.

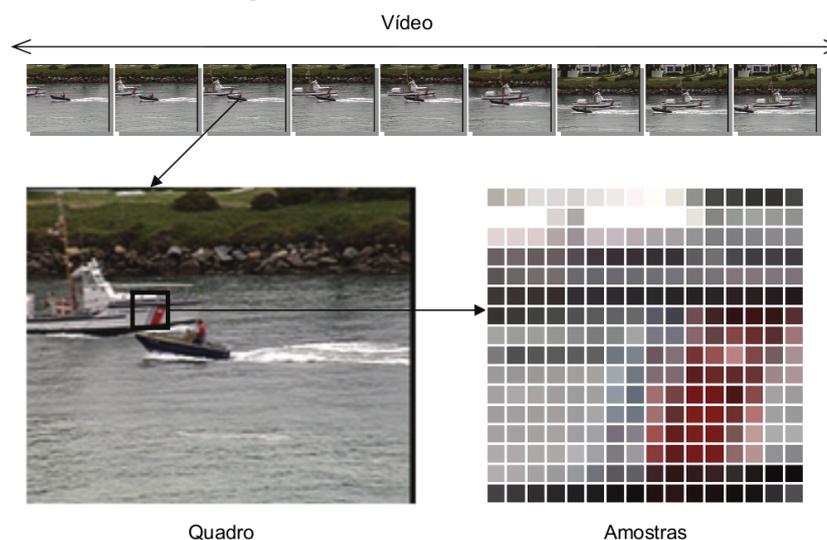


Figura 2.1: Amostragem de um vídeo em macrobloco 16x16 pixels. Fonte: [Manoel 2007]

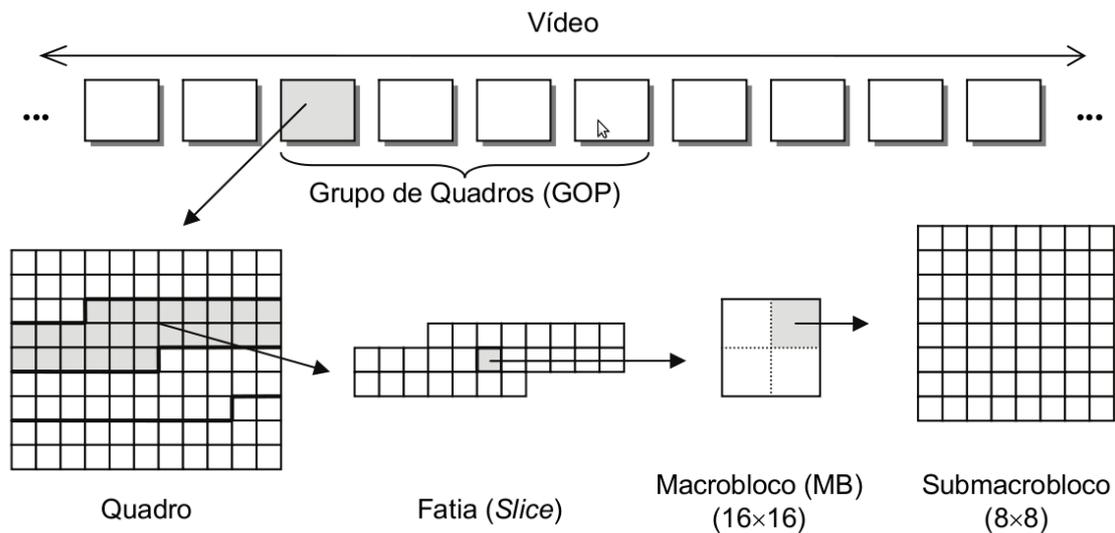


Figura 2.2: Organização hierárquica da camada de codificação de vídeo, subdividindo o vídeo em quadros, quadros em fatias e fatias em macroblocos. Fonte: [Manoel 2007]

Não é prático usar um vídeo em sua forma descomprimida ou *raw* em função do tamanho. Por exemplo, o DVD, que muitas pessoas usam diariamente, tem o tamanho do quadro definido por 704 colunas com 480 linhas de resolução, ou simplesmente 704x480 pontos coloridos (pixels). Como cada pixel precisa de 24 bits, uma imagem necessita de 990 Kbytes. Para obtermos um vídeo são necessários pelo menos 24 fps (frames per second ou quadros por segundo) para que o olho humano tenha a ilusão de movimento [VIJAYAKUMAR 2008]. Assim, a taxa de dados necessária por segundo é :

$$(704 \times 480) \text{ pixels/quadro} \times 24 \text{ bits/pixel} \times 24 \text{ quadro/s} \cong 195 \text{ Mbits/s}$$

Essa taxa é maior do que a mais alta taxa de transmissão de dados disponibilizada por internet de banda larga para a maioria das pessoas. Porém, como se não bastasse, deve-se multiplicá-la pela duração do filme para saber o quanto de espaço é necessário para armazenamento de um filme. Considerando um filme comum de 90 minutos, chegamos ao valor de **128GB**, um absurdo mesmo com os discos rígidos de hoje em dia. Cabe observar que atualmente, o armazenamento de um filme em mídia ótica tipo DVD e o acesso a este filme somente são possíveis graças à taxa de compressão de 32:1 proporcionada pelo padrão de compressão H.262/MPEG-2, utilizado nos sistemas reprodutores de vídeo. Logo, sem a compressão, seria impossível armazenar um filme convencional em uma mídia DVD cuja capacidade é de apenas 4.7GB e cuja taxa de transferência é de cerca de 6Mbits/s. Atualmente, a TV digital de alta definição exige resoluções maiores como 1920x1080 pixels, que ainda podem ser disponibilizadas pela internet, restringindo a largura de banda em torno de 4Mbits/s. O problema do armazenamento e da taxa de transmissão começa na captura da imagem e segue pelo armazenamento e transmissão dos quadros.

O nível de compressão dos dados que representam um vídeo é o resultado da aplicação de diversas técnicas as quais retiram informações redundantes. A redundância de informação existente nos vídeos é classificada em:

- Redundância Temporal: Relativa às informações repetidas entre quadros próximos;
- Redundância Espacial: Relativa às informações repetidas entre os pixels próximos, dentro de um quadro;
- Redundância Freqüencial: Relativa às freqüências do quadro que o sistema visual humano (HVS) não é sensível, portanto podem ser retiradas;
- Redundância Entrópica: Relativa à repetição estatística dos símbolos e bits no fluxo de saída;

Na figura 2.3 podemos observar um histórico das principais técnicas de compressão de vídeo e uma linha de tempo de sua criação. [Reader 2002]

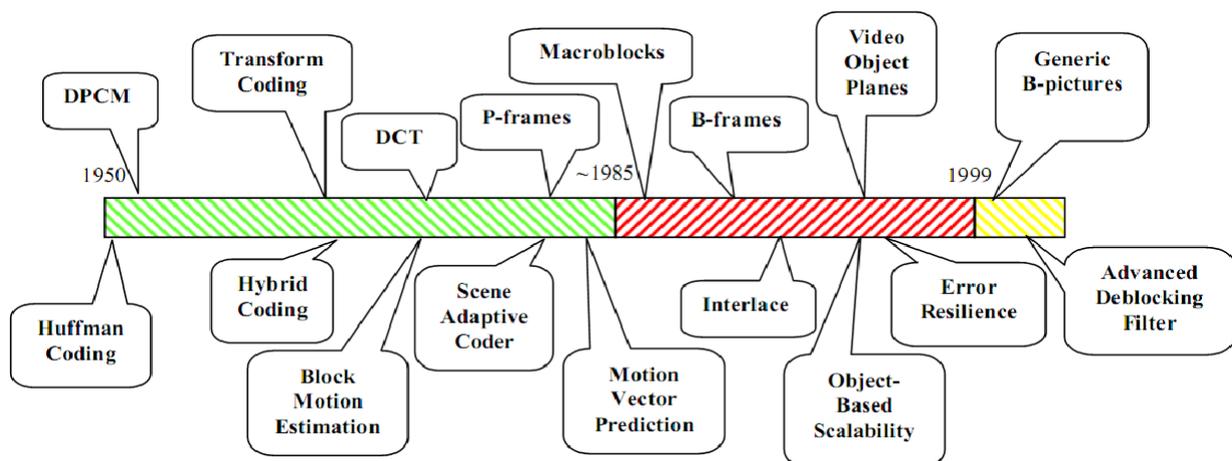


Figura 2.3: Linha do tempo das principais técnicas de compressão de vídeo. Fonte: [Reader 2002]

Como o desafio da compressão de vídeo acompanha o crescimento da digitalização e distribuição de conteúdos pela internet, TV digital e outros meios, é importante buscarmos soluções que garantam os requisitos de desempenho, qualidade, disponibilidade e manutenção dessas aplicações. Para facilitar a tarefa, a partir dos anos 80 e 90 as organizações ITU-T VCEG (International Telecommunication Union - Vídeo Coding Experts Group) e ISO/IEC MPEG (International Organization for Standardization e International Electrotechnical Commission - Moving Picture Experts Group) definiram padrões de compressão de vídeo, os quais reúnem diversas técnicas de maneira organizada para a interoperabilidade entre fabricantes de dispositivos, produtores de conteúdo, transmissores e usuários finais. Sistemas de TV digital (incluindo o brasileiro), novos formatos de filmes e sistemas de transmissão por internet adotaram o mais novo padrão de vídeo chamado de H.264/AVC.

2.1 Padrão H.264/AVC

As organizações ITU-T VCEG (International Telecommunication Union - Vídeo Coding Experts Group) e ISO/IEC MPEG (International Organization for Standardization e International Electrotechnical Commission - Moving Picture Experts Group) definiram diversos padrões de compressão. A figura 2.4 observa-se uma linha de tempo com os padrões criados pelas referidas organizações.

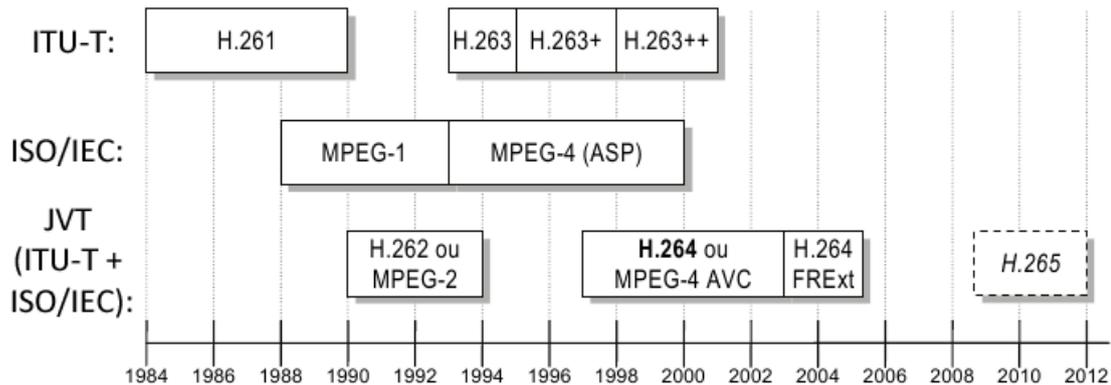


Figura 2.4: Histórico dos padrões apresentados pelos órgãos ITU-T e ISO/IEC. Fonte: [Manoel 2007]

O padrão H.264/AVC melhora em mais de 2 vezes a eficiência do MPEG-2, exigindo para tal, maior quantidade de processamento e novas técnicas, o que garante o status atual de estado-da-arte em compressão de vídeo pelo seu aumento em taxa de compressão. Utiliza, para tanto, as seguintes técnicas [Manoel 2007]:

- Predição Intra-quadro;
- Transformada DCT inteira com inversa exata;
- Transformada Hadamard na predição Intra-quadro;
- Aritmética com 16 bits;
- Precisão de $\frac{1}{4}$ pixel na estimação de movimento;
- Múltiplas referências temporais;
- Predição de movimento direta e skip;
- Tamanhos possíveis para as partições ou blocos: 16x16, 16x8, 8x16, 8x8, 8x4, 4x8, 4x4.
- Filtro anti-bloco adaptativo;
- Predição usando pesos e offsets nas referências.
- Novas codificações entrópicas:
 - Codificação Adaptativa de Comprimento de Palavra Variável - CAVLC;
 - Codificação Aritmética Binária Adaptativa ao Contexto - CABAC;

A figura 2.5 apresenta um esquema de codificador segundo o padrão H.264/AVC.

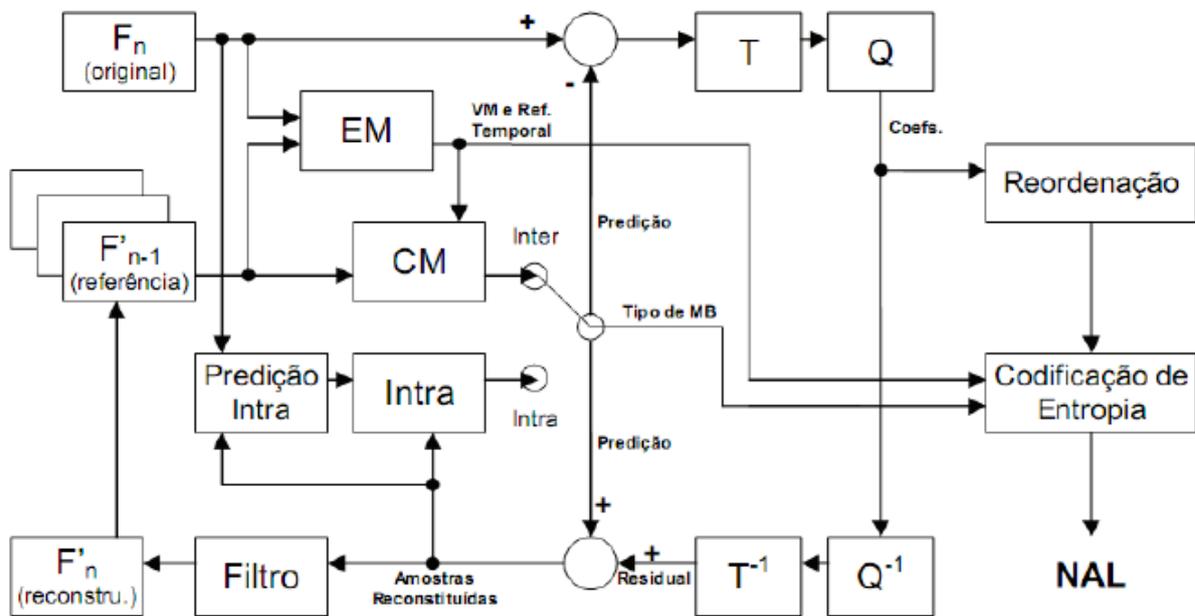


Figura 2.5: Esquema de compressor H.264/AVC. Fonte: [Manoel 2007]

O quadro original (F_n) é submetido à transformação do valor das amostras do domínio espacial para o domínio das frequências (módulo T) para a concentração da energia em coeficientes representando frequências distintas. O primeiro elemento denominado DC (corrente contínua) não representa uma frequência como os outros coeficientes denominados AC (corrente alternada), os quais à medida que se percorre a matriz indicam maiores frequências. Para tal tarefa é usada a DCT inteira com inversa exata. Esta é uma inovação do H.264/AVC em relação aos outros padrões, os quais apresentavam problemas de arredondamento. Além da DCT, o H.264/AVC também prevê o uso de uma transformada Hadamard 2×2 , 2×4 e 4×4 de forma hierárquica nos coeficientes [Manoel 2007].

A etapa seguinte é a quantização (módulo Q), que elimina redundância frequencial, maior responsável pela perda de informação ou distorção devido ao arredondamento e eliminação de coeficientes [Manoel 2007]. Filtra principalmente as altas frequências, para as quais o HVS apresenta menor sensibilidade. A quantização equivale a uma divisão com arredondamento nos coeficientes transformados, mas no H.264/AVC opera somente com multiplicação por fatores constantes (MF - Multiplication Factors) e deslocamento de bits [Richardson 2003]. Após a quantização, os coeficientes são reordenados agrupando símbolos repetidos, para então realizar a codificação entrópica, que elimina as redundância estatísticas. Então o quadro original é reconstruído (F'_n) para ser usado como referência (F'_{n-1}).

O próximo quadro (F_{n+1}) usa previsões (módulos EM, CM e Intra) para reaproveitar valores codificados anteriormente (referências). O uso da informação de múltiplos quadros já reconstruídos, sendo estes passados e futuros é denominada previsão inter-quadro. O uso da informação já reconstruído do quadro atual é chamada previsão intra-quadro [Manoel 2007]. A figura 2.6 exemplifica um quadro cujas partições possuem diferentes tamanhos e modos de previsão.

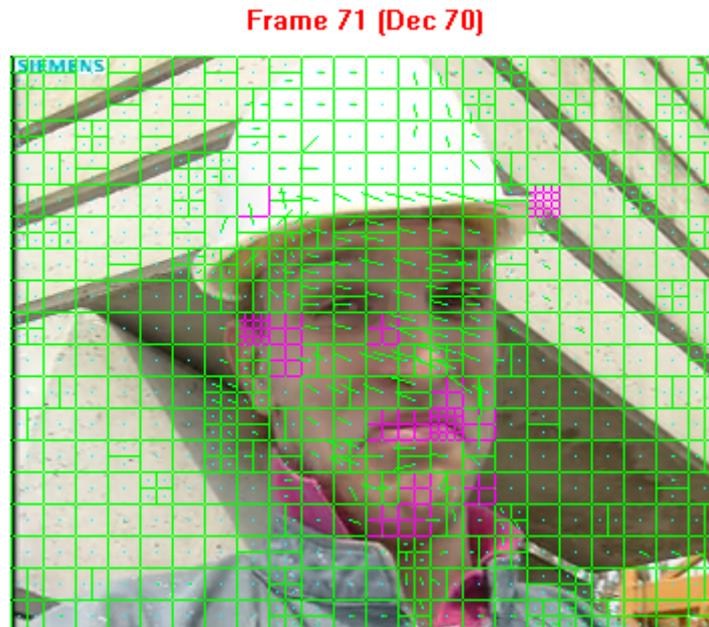


Figura 2.6: Exemplo de múltiplas partições com previsão inter-quadro (verde) e intra-quadro (rosa).

A previsão inter explora a redundância temporal, ou seja, o fato de que em quadros próximos apenas alguns objetos da cena se deslocam em relação ao resto. Usando a compensação de movimento, é possível medir o deslocamento e reaproveitar as partes que não sofreram alteração, codificando apenas os vetores de movimento e o resíduo. Para essa técnica, no H.264/AVC é usada precisão $\frac{1}{4}$ de pixel no deslocamento dos vetores de movimento, favorecendo pequenas alterações de movimento, como mostrado na figura 2.7. É importante observar ainda que essa previsão é realizada em tamanhos variáveis de partição, gerando dois modos de previsão inter-quadro, dependendo do uso das listas de referências passadas e futuras. O modo P somente aponta para quadros passados e o modo B aponta para passados e futuros. No H.264/AVC, a quantidade de quadros de referência nas duas listas foi generalizada. Além disso, os vetores de movimento podem selecionar qualquer quadro da lista, como na figura 2.8, evitando casos de oclusão e deformação de objetos. Outra possibilidade são as previsões “modo skip” e “direta”, em que as informações codificadas são reduzidas pelo uso de convenções pré-estabelecidas, como a co-localização com o quadro de referência e uso de um vetor de movimento predito. Se a referência for a primeira da lista passada, o modo é skip, ou então a referência é derivada, constituindo o modo direto [Manoel 2007].

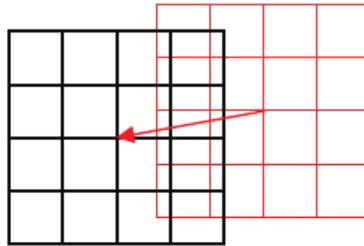


Figura 2.7: Exemplo de compensação de movimento usando vetor 3/4 à direita e 1/2 acima. Fonte: [Manoel 2007]

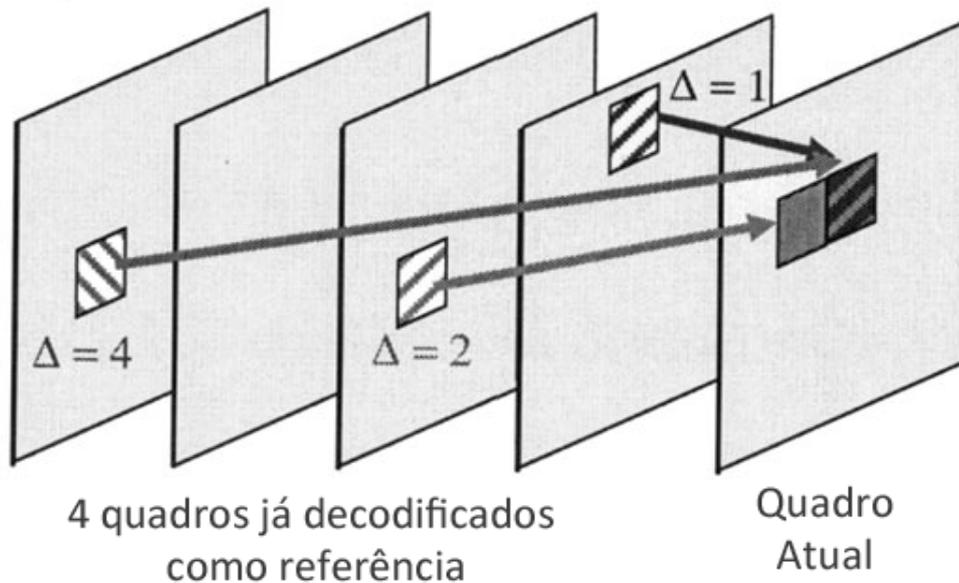


Figura 2.8: Exemplo de múltiplas referências temporais. Fonte: [VIJAYAKUMAR 2008]

O quadro reconstruído pela compensação de movimento passa por um filtro anti-bloco (módulo Filtro) para reduzir os efeitos de borda e melhorar a qualidade da predição. O H.264/AVC foi o primeiro padrão a incorporar esse filtro como parte do processo de codificação, já que anteriormente era usado opcionalmente como pós-processamento na decodificação [Manoel 2007].

A predição intra-quadro é uma das principais inovações do H.264/AVC, em que elimina-se a redundância espacial aproveitando as bordas de blocos vizinhos já reconstruídos, replicando-as em alguma das direções ou modos de predição. Um exemplo pode ser encontrado na figura 2.9. A figura 2.10 possui um diagrama dos 9 modos intra 4x4 (análogos em 8x8), além dos 4 modos intra 16x16 [VIJAYAKUMAR 2008].

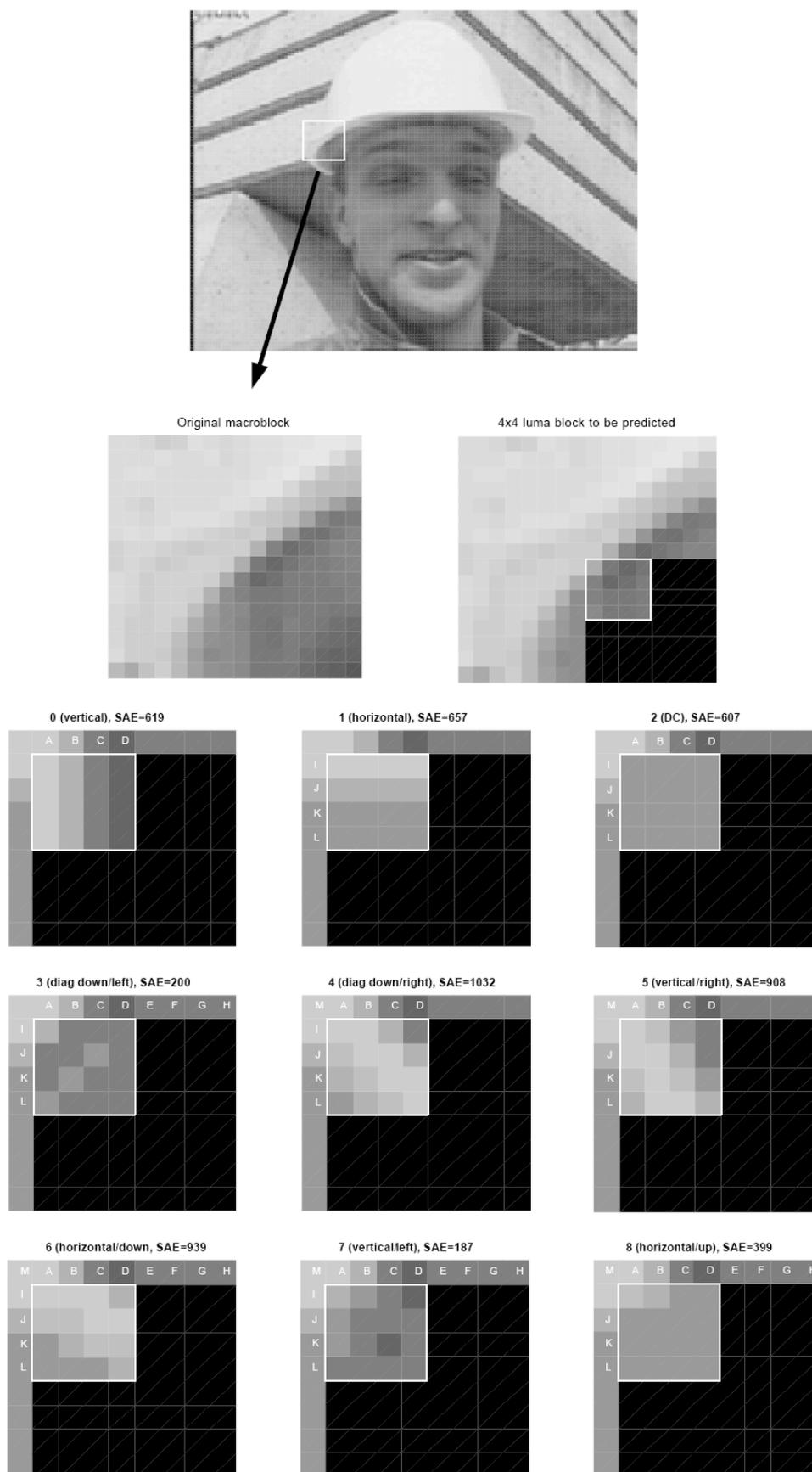


Figura 2.9: Exemplo de predição intra-quadro 4x4 com modo 7 selecionado. Fonte: [VIJAYAKUMAR 2008]

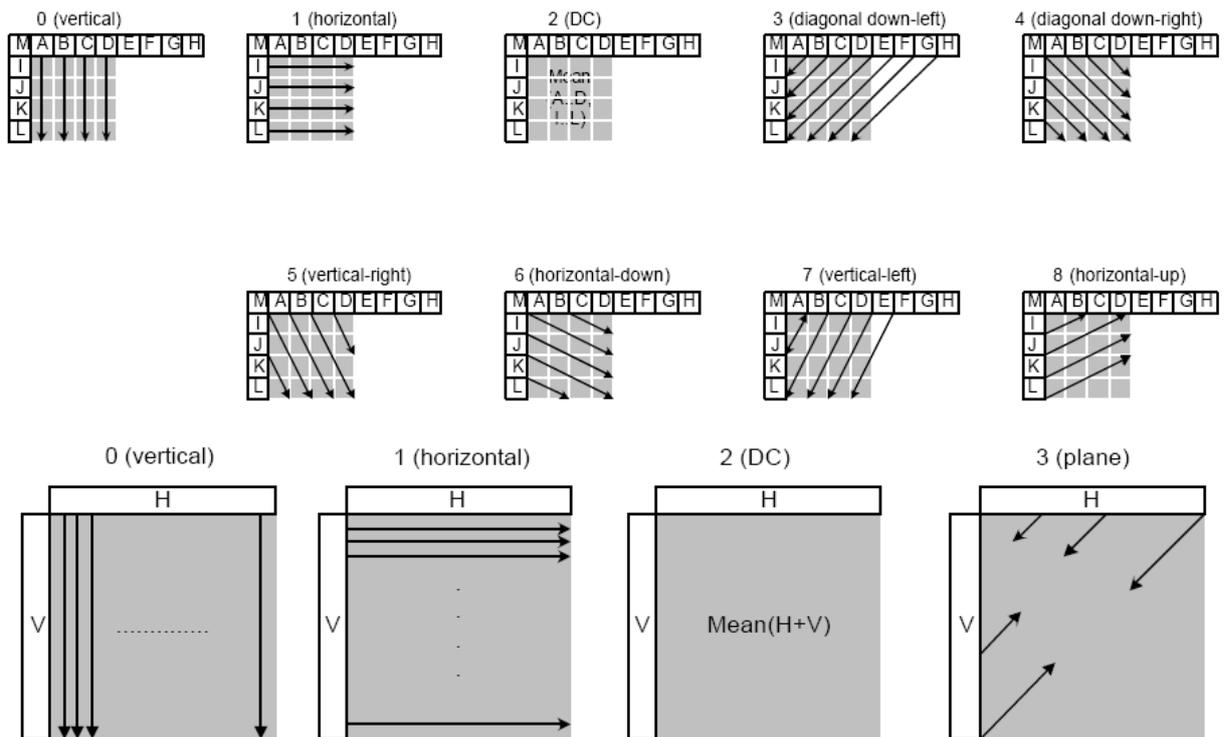


Figura 2.10: Esquema dos modos de predição Intra 4x4 e Intra 16x16. Fonte: [VIJAYAKUMAR 2008]

Em todos os tipos de predição, é necessário fazer o cálculo da distorção do candidato predito com o atual usando alguma métrica para encontrar o melhor candidato na predição. Cabe observar que a métrica adotada tem impacto direto e significativo no tempo de codificação e na qualidade final do processo de compressão. Portanto a investigação de métricas que proporcionem uma boa relação entre qualidade da compressão e complexidade do cálculo (i.e. custo de implementação) tem sido um assunto de grande relevância.

3 Métricas de Custo

No padrão *H.264/AVC* é permitida uma vasta escolha de partições para um macrobloco em blocos menores como 16x8, 8x16, 8x8, 8x4, 4x8, 4x4 na predição *inter* ou 16x16, 8x8, 4x4 na predição *intra*. As decisões sobre qual tipo de predição e qual o tamanho da partição a ser usada são realizadas pelo “controle de taxa“ baseadas em métricas de custo e parâmetros do usuário. Para cada macrobloco, é analisado seu custo usando todas as possibilidades de particionamento e predição para todos os quadros de referência disponíveis. Os dados resultantes desta análise fazem parte da tomada de decisão final, como na figura 3.1, pela Otimização Taxa-Distorção [Merritt e Vanam 2007].

3.1 Otimização Taxa-Distorção

Comprimir vídeo significa retirar as redundâncias, usando como base várias métricas, através da técnica de Otimização Taxa-Distorção (*RDO*), medindo o custo de bit e qualidade para cada possibilidade de decisão, de acordo com a escolha do usuário, i.e., procurando minimizar a taxa de bits para um fator de qualidade especificado ou buscando minimizar a distorção para uma taxa de bits especificada [Manoel 2007]. Para tal, utiliza-se a seguinte métrica de taxa-distorção:

$$J = D + \lambda T$$

Onde D é uma medida de distorção, T é a taxa de bits obtida da codificação de entropia e λ é o multiplicador de *Lagrange* que informa a relação entre a taxa de bit e distorção [CBLOOM], calculada para cada nível de qualidade definido pelo parâmetro de quantização. Qualquer taxa de bit com ganho de distorção inferior a λ não será codificada (*lagrange rate control*). Na figura 3.2 observamos um espaço de soluções. Nesse espaço existe uma curva de soluções ótimas obtidas pela intersecção com a tangente de λ formando uma borda convexa.



Figura 3.1: Exemplo de decisão final com imagem original (F_n), predita (F'_{n-1}), resíduo e reconstruída (F'_n).

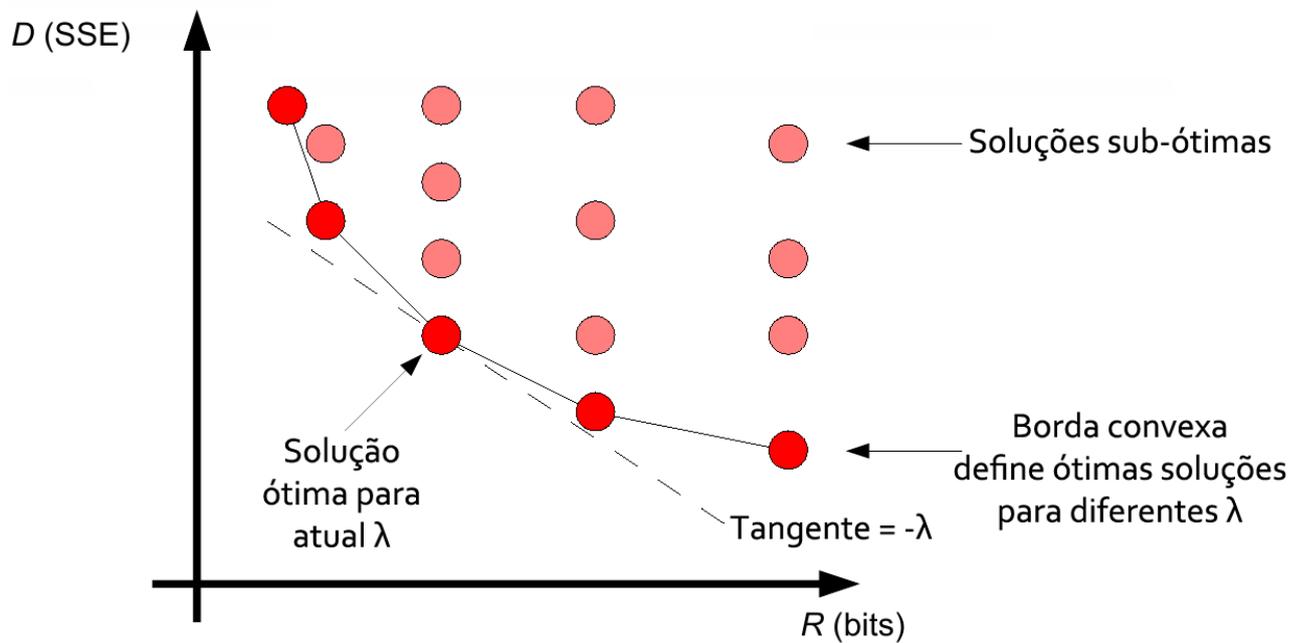


Figura 3.2: Espaço de soluções possíveis e otimização Lambda. Fonte: [Terribery].

3.2 Métricas de Distorção

Como as métricas de distorção desempenham um papel importante na tomada de decisão e no desenvolvimento de um compressor, apresenta-se a seguir uma revisão sobre as métricas mais usadas e os respectivos critérios usados em seu desenvolvimento.

À medida que o conhecimento sobre o sistema visual humano (*HVS*) aumenta, métricas mais apropriadas podem ser desenvolvidas, as quais geram ganhos no sistema de codificação de vídeo [Wu e Rao 2005]. Tais métricas devem satisfazer a critérios de aceitação como aqueles propostos pelo Dr. K.R. Rao, criador da transformada discreta do cosseno:

- O modelo matemático deve ser facilmente calculável;
- Deve correlacionar o sistema visual humano *HVS*;
- Deve correlacionar os artefatos de compressão;
- Precisa permitir operação de um modo de qualidade constante;
- Deve modelar requisitos e expectativas individuais;

Podemos então dividir as métricas entre dois grupos:

- **Subjetivas:** São avaliações que usam pessoas divididas em grupos, as quais assistem a vídeos de teste em um ambiente controlado. Após a exibição de cada vídeo, cada pessoa completa um formulário-padrão, o qual serve para coletar dados sobre os limiares de percepção observados e quais tipo de artefatos geram incômodo. As condições de teste e algumas métricas são padronizadas pela ITU-R BT.500-11.
- **Objetivas:** Criadas a partir da extração e análise de artefatos e características visuais dos testes subjetivos, que em conjunto com modelos matemáticos do HVS e de processamento de sinais, geram modelos de cálculo específicos.

As métricas de distorção objetivas mais usadas são apresentadas a seguir:

SAD - Soma das Diferenças Absolutas: Métrica de distorção constituída pela soma do valor absoluto das diferenças entre cada amostra original e a amostra correspondente, após o processo de reconstrução. Devido a sua baixa complexidade computacional, é adequada para implementação direta em *hardware* [Manoel 2007] .

$$SAD = \sum_{i=0}^M \sum_{j=0}^N |Ori_{i,j} - Dec_{i,j}|$$

SSD - Soma das Diferenças Quadráticas: Métrica de distorção correspondente à soma do quadrado das diferenças entre cada amostra original e a amostra após o processo de reconstrução. Equivalente à norma L2 e sua característica principal é evidenciar os erros significativos [Manoel 2007].

$$SSD = \sum_{i=0}^M \sum_{j=0}^N (Ori_{i,j} - Dec_{i,j})^2$$

SATD - Soma das Diferenças Tranformadas Absolutas: Baseada na *SAD*, mas usa a transformada *Hadamard* (H) numa partição 4x4 de diferenças antes de efetuar a soma absoluta dos coeficientes. Apresenta maior custo que a *SAD*, com melhor qualidade por operar no domínio das frequências [Manoel 2007].

$$SATD = \frac{1}{2} \sum_{i=0}^3 \sum_{j=0}^3 |H(Ori_{i,j} - Dec_{i,j})H^T|$$

$$Hadamard H = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \\ 1 & -1 & 1 & -1 \end{bmatrix}$$

MSE - Erro Quadrático Médio: Corresponde à média do quadrado das diferenças entre cada amostra original e a amostra correspondente, após o processo de codificação/decodificação [Manoel 2007] calculada em um bloco com tamanho M por N.

$$MSE = \frac{SSD}{M \times N}$$

PSNR - Razão Sinal-Ruído de Pico: Métrica de distorção objetiva de vídeo mais amplamente usada. Seu valor é dado em decibéis para amostras de n bits [Manoel 2007].

$$PSNR = 10 \log_{10} \frac{(2^n - 1)^2}{MSE}$$

SSIM - Índice de Similaridade Estrutural: Valor máximo 1 que corresponde a duas imagens X e Y, exatamente iguais, estruturalmente [Wang et al. 2004] .

$$SSIM(x,y) = \frac{(2\mu_x\mu_y+c_1)(2\sigma_{xy}+c_2)}{(\mu_x^2+\mu_y^2+c_1)(\sigma_x^2+\sigma_y^2+c_2)}$$

- μ_x a média de x;
- μ_y a média de y;
- σ_x^2 a variância de x;
- σ_y^2 a variância de y;
- σ_{xy} a covariância de x e y;
- $c_1 = (k_1L)^2$, $c_2 = (k_2L)^2$ variáveis para estabilizar a divisão por denominador fraco;
- L a gama dinâmica do pixel, $2^{\#bits \text{ pixel}} - 1$);
- $k_1 = 0.01$ e $k_2 = 0.03$ por padrão.

Conforme [Wu e Rao 2005], as métricas mais utilizadas, tais como *MSE*, *SAD*, *PSNR* e *SSIM*, não são capazes de correlacionar o grau de distorção de uma imagem tão bem quanto o esperado pelo olho humano (HVS). Por exemplo, as duas imagens da figura 3.3 possuem o mesmo valor de *PSNR*. Apesar disto, é possível perceber-se claramente que existe uma diferença na distribuição espacial do ruído entre áreas de alta complexidade (rochedo) e áreas de baixa complexidade (céu claro). Como o *HVS* tem dificuldade em distinguir ruído dentro de áreas de alta complexidade, a qualidade percebida (subjéitiva) resultante é maior na imagem da esquerda.



(a)

(b)

Figura 3.3: As duas imagens possuem o mesmo *PSNR*, mas o ruído está distribuído em regiões diferentes [WINKLER 2005].

4 Métrica Proposta

Face ao exposto no capítulo anterior, conclui-se que há necessidade de criação de novas métricas que sejam mais adaptadas às limitações de HSV. Desta forma este trabalho propõe e avalia uma nova métrica para a taxa de distorção. A métrica proposta trabalha com a quantização e os conceitos do teorema de *Parseval* [WOLFRAM]. Este teorema postula a conservação de energia de um sinal no domínio espacial e no domínio frequencial. Para exemplificar essa conservação, tomemos a *SSD* de uma imagem e comparemo-la com a *SSD* da mesma imagem anteriormente transformada pela *DCT*. Conclui-se que a *SSTD* (parecida com a *SATD* só que nesse caso usa a *DCT*) é equivalente a *SSD* devido à ortonormalidade da transformada. Considerando que a *SATD* utiliza a transformada *Hadamard*, que também aproxima a *DCT* inteira do *H.264/AVC*, tal métrica portanto opera no espaço da frequência, o qual também é utilizado pela quantização. Desse modo, a *SATD*, diferentemente da *SAD*, mede as frequências que serão posteriormente quantizadas, obtendo assim uma melhor aproximação da taxa-distorção real.

$$Hadamard_{4 \times 4} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \\ 1 & -1 & 1 & -1 \end{bmatrix} \quad DCT_{4 \times 4} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 2 & 1 & -1 & -2 \\ 1 & -1 & -1 & 1 \\ 1 & -2 & 2 & -1 \end{bmatrix}$$

Grande parte das métricas não considera a quantização, que é a maior causa de distorção gerada, seja por erros de arredondamento, seja principalmente pela eliminação de coeficientes. Neste caso, propõe-se criar uma classificação dos coeficientes transformados como forma de obter ganhos em desempenho e aproximação da taxa-distorção. Para tal, a partir das fórmulas da quantização, os valores limite (*thresholds*) dos coeficientes *DCT* que serão eliminados são pré-calculados para cada um dos 52 parâmetros de quantização (*QP*) utilizados no controle, também de acordo com cada posição da tabela de fatores multiplicativos *MF* do padrão *H264/AVC* [Richardson 2003].

Usando esses *thresholds*, podemos classificar os coeficientes de resíduo transformados pela *DCT* em dois grupos: codificados e eliminados. Agora, cada grupo receberá um tratamento diferente:

Tabela 4.1: Tabela de constantes MF da quantização para cada posição 4x4 [Richardson 2003]

| QP % 6 | (0,0),(0,2),(2,0),(2,2) | (1,1),(1,3),(3,1),(3,3) | Restante |
|--------|-------------------------|-------------------------|----------|
| 0 | 13107 | 5243 | 8066 |
| 1 | 11916 | 4660 | 7490 |
| 2 | 10082 | 4194 | 6554 |
| 3 | 9362 | 3647 | 5825 |
| 4 | 8192 | 3355 | 5243 |
| 5 | 7282 | 2893 | 4559 |

- Os coeficientes codificados serão somados absolutamente como na *SATD*. A transformada usada no caso é a *DCT* para melhorar a correlação em comparação com a Hadamard.
- Os coeficientes eliminados serão somados quadraticamente como na *SSD*.

Isto garante a aproximação dos codificados para a taxa de bits, e dos eliminados para a distorção causada pela quantização. Os *thresholds* são calculados para as transformadas *DCT* de tamanho 4x4 e 8x8 do padrão *H.264/AVC*, sendo apenas armazenados para o maior parâmetro de quantização (51), que com uma operação de deslocamento à direita (\gg) por $qf = 8 - \text{floor}(QP/6)$, extrai-se para todos os outros *QPs*. Equação de quantização, usada para geração dos *thresholds* [Richardson 2003]:

$$0 = \text{round}[\text{Coef}f * MF[QP\%6, n] / 2^{\text{floor}(QP/6)}] * \frac{1}{2^{15}}$$

$$qp51zero4x4 = x \begin{bmatrix} 640, & 1039, & 1599, \\ 703, & 1119, & 1800, \\ 832, & 1279, & 2000, \\ 896, & 1440, & 2300, \\ 1023, & 1599, & 2500, \\ 1151, & 1840, & 2899, \end{bmatrix} qp51zero8x8 = \begin{bmatrix} 640, & 734, & 399, & 686, & 500, & 541, \\ 703, & 774, & 437, & 758, & 559, & 58, \\ 832, & 938, & 525, & 867, & 660, & 699, \\ 896, & 1019, & 562, & 939, & 699, & 745, \\ 1023, & 1141, & 637, & 1083, & 799, & 857, \\ 1151, & 1305, & 725, & 1228, & 920, & 970, \end{bmatrix}$$

Figura 4.1: tabela com os *thresholds* para as *dct* 4x4 e 8x8 do padrão *H.264/AVC*.

Como o resultado da métrica é a soma dos valores obtidos criando uma função definida em trechos descontínuos, é necessário corrigir a descontinuidade, multiplicando por uma constante C, como no exemplo. Completando denominamos a métrica proposta de *Quantization Distortion Energy* (QDE), já que medimos a energia perdida na distorção pela quantização.

```

If (residuo[i] > zeroqp51_4x4[i])

    sum_bits += abs(residuo[i]);
else
    sum_distortion += residuo[i] * residuo[i];
result = sum_bits + (sum_distortion*(1/zeroqp51_4x4[i])

```

Inclui correção da descontinuidade entre as parcelas usando constante 1/threshold.

4.1 Compressor H.264/AVC base da implementação proposta

A primeira implementação do padrão H.264/AVC é chamada Joint model [JVT]. Esta implementação de referência nasceu de versões do Test Model Long-Term desenvolvido pelo MPEG em Agosto de 1999. O JVT foi criado quando o VCEG uniu esforços com o MPEG, trocando o nome do software para JM. Uma outra implementação que merece destaque é um projeto open source chamado x264 [Merritt et al.], projeto este que nasceu em 2003. Mesmo sendo *open source*, existe a necessidade das licenças associadas ao padrão H.264/AVC. O x264 atinge velocidade 50 vezes maior que a JM [Merritt e Vanam 2007]. No teste disponível em <http://x264dev.multimedia.cx/?p=102> o x264 demonstra melhor qualidade que versões comerciais de diversas empresas. A fig. 4.2 mostra esse resultado, com um teste onde um vídeo complexo foi codificado em qualidade máxima, salientando que por se tratar do estado da arte melhorias de 2% já são significativas. Além disso, é constantemente atualizado, constituindo assim uma ótima plataforma para estudo de inovações. Para a implementação da métrica proposta, foi escolhido então o compressor x264.

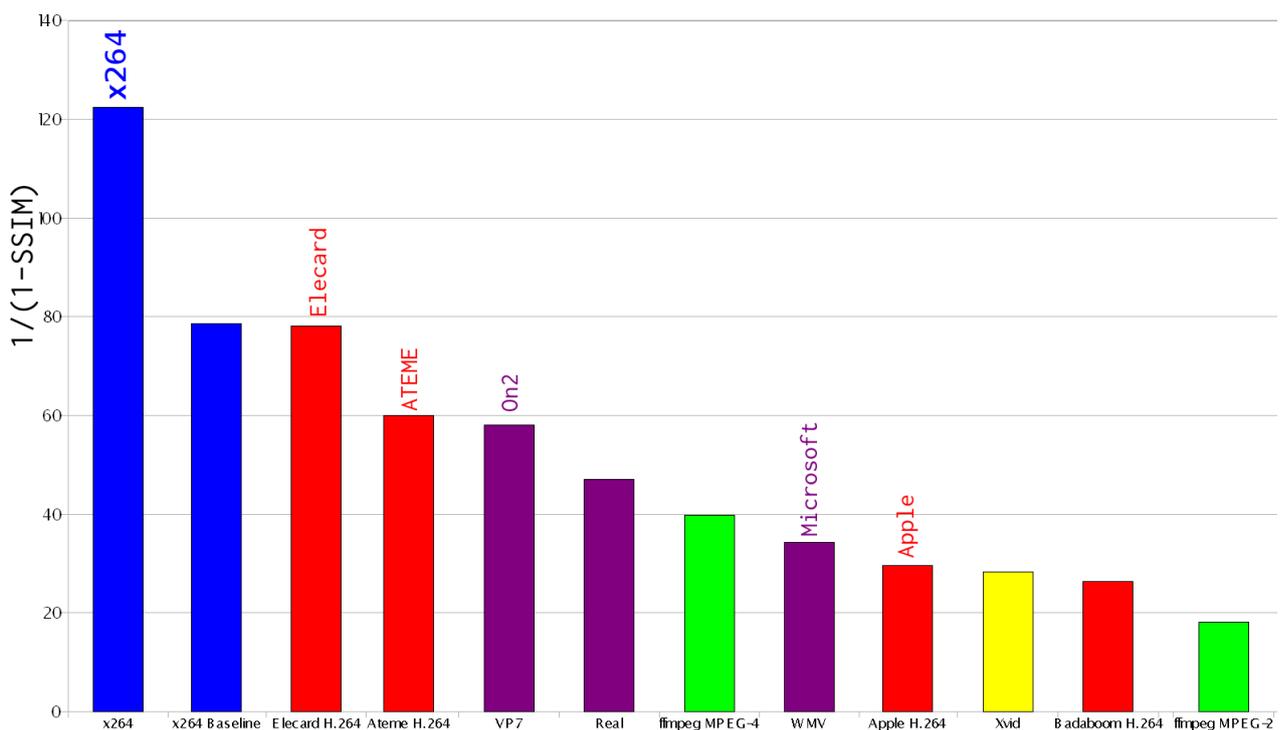


Figura 4.2: Comparação entre diversos compressores em modo de qualidade máxima com a métrica DSSIM. Fonte: [Merritt et al.]

5 *Metodologia de Experimentação*

A métrica proposta foi incorporada no compressor x264 revisão 1564. Originalmente o x264 usa a SATD como decisão final, após descarte prévio de alguns candidatos utilizando como métrica a *SAD* [Merritt e Vanam 2007]. Foram realizados três tipos de experimentos segundo as possibilidades do RDO. São eles:

- Experimento A: restrição na taxa de bit variando em 400, 700, 1000, 2000, 4000, 8000 (kbit/s), usando controle multi-passos que garante o tamanho especificado. Os testes usam um script automático que itera entre os arquivos de entrada, gerando as versões x264 original (SATD) e x264 modificado com a métrica proposta (QDE). A seguinte linha de comando para o compressor foi utilizada:

```
x264 --bitrate $j --direct auto --psy-rd 1.0:1.0 --aq-strength 1.0 --aq-mode 2
-t 2 --level 4.1 --ssim --psnr --tune film $i
```

- Experimento B: restrição na qualidade (SSIM) usando o original com o fator de qualidade constante (CRF) fixo em valor 20, valor este considerado como excelente. A busca pelo novo CRF na métrica proposta que atinge a mesma qualidade (SSIM) necessita realização manual. A seguinte linha de comando para o compressor foi utilizada:

```
x264 --CRF 20 --direct auto --psy-rd 1.0:1.0 --aq-strength 1.0 --aq-mode 2
-t 2 --level 4.1 --ssim --psnr --tune film $i
```

- Experimento C: restrição igual ao experimento B mas variando o CRF (1, 10, 17 até 30, 40 e 50), somente para o vídeo *sunflower*, o qual obteve melhores resultados no experimento B.

A configuração da máquina usada para executar os experimentos foi um Intel Core 2 Quad Q9550 12 M Cache @ 2.83GHz com 4GB de memória RAM, rodando Ubuntu Linux 10.04 SMP 64bit 2.6.32-23-generic.

Os vídeos de teste utilizados nos experimentos, foram obtidos no repositório da Xiph Foundation [Xiph.org], e correspondem a um mix de resolução e complexidade escolhidos aleatoriamente. Os vídeos escolhidos estão listados na fig. 5.1 .



sunflower, 1080p



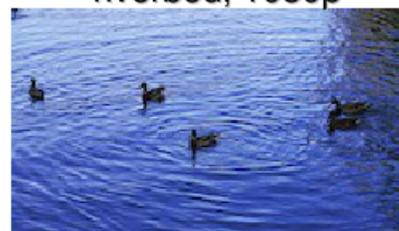
riverbed, 1080p



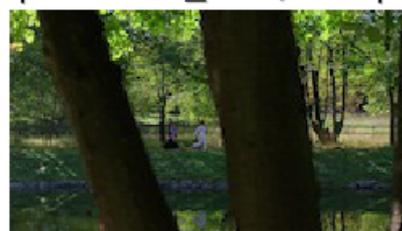
pedestrian_area, 1080p



in_to_the_tree, 720p



ducks_take_off, 720p



park_joy, 720p



soccer, 288p



foreman, 288p



big_buck_bunny, 480p

Figura 5.1: Vídeos de teste utilizados nos experimentos. Fonte: [Xiph.org]

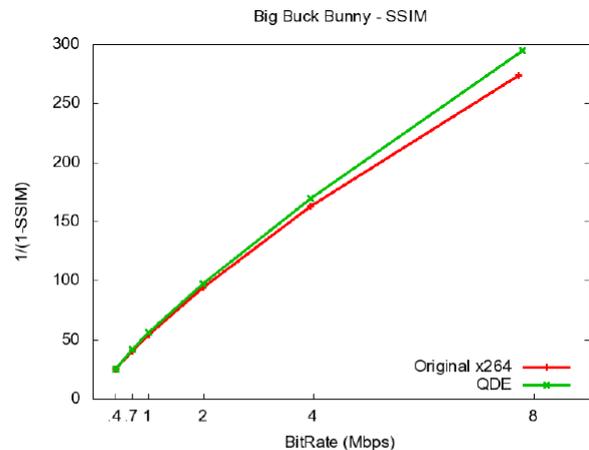
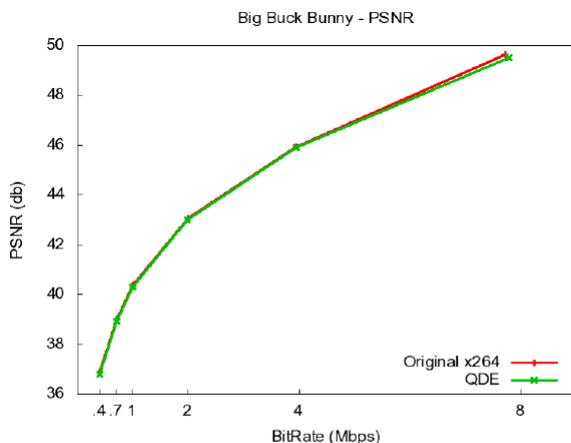
6 Resultados Experimentais

Os resultados dos experimentos serão apresentados separadamente nas próximas seções.

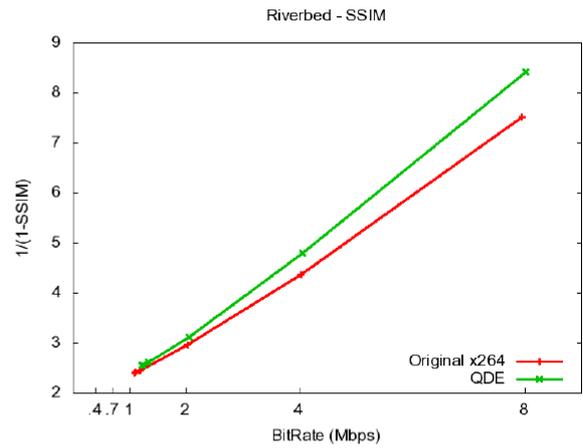
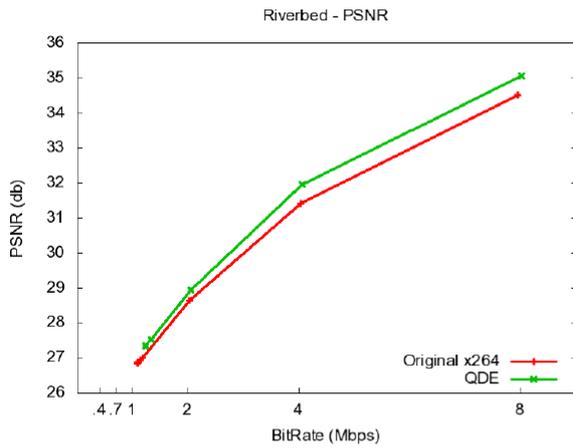
6.1 Experimento A

Cada um dos experimentos mostrará: tabelas com o quantizador médio dos quadros tipo I do vídeo codificado com a métrica original (inferior) e a métrica proposta (superior). Também apresentaremos gráficos distorção/bitrate (RD - RateDistortion) que correlacionam qualidade e tamanho atingidos para cada vídeo, usando duas métricas de distorção PSNR e DSSIM. A métrica dissimilaridade estrutural (DSSIM) é uma métrica de distância derivada da SSIM para comparação entre os casos [Wang et al. 2004].

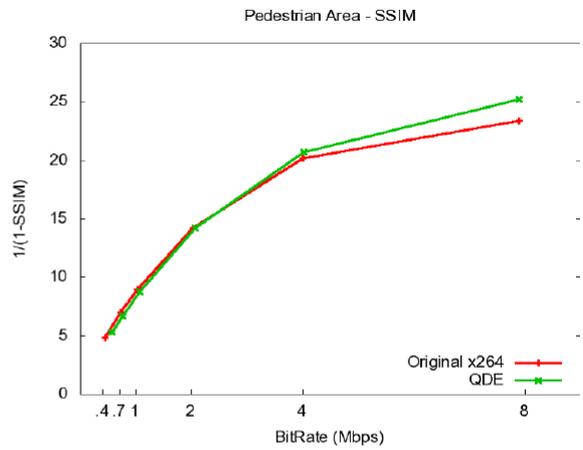
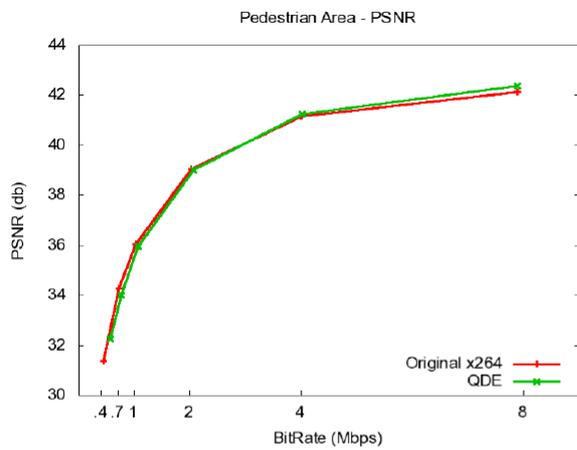
$$DSSIM(x, y) = \frac{1}{1 - SSIM(x, y)}$$



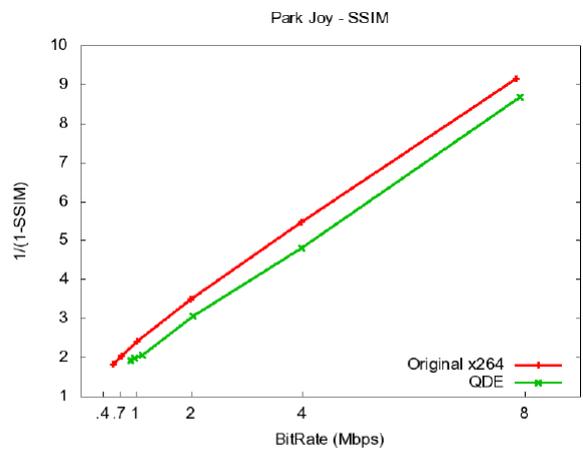
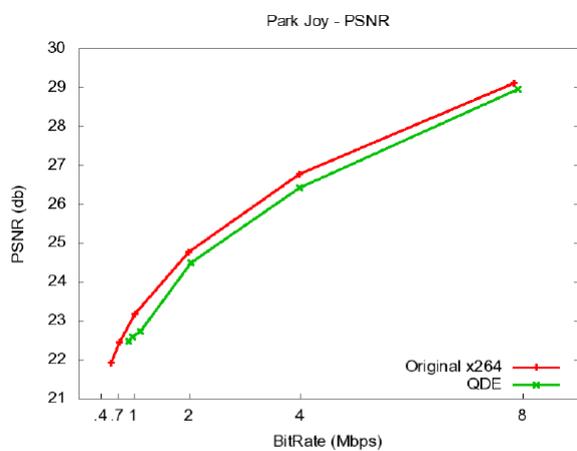
| | | | | | | |
|------------|-------|-------|-------|-------|-------|-------|
| I-QP média | 25,67 | 22,29 | 20,32 | 16,59 | 12,94 | 10,29 |
| | 24,85 | 21,53 | 19,60 | 15,80 | 12,28 | 10,29 |



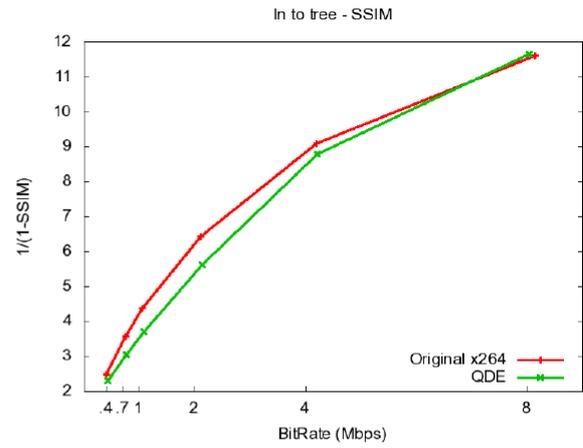
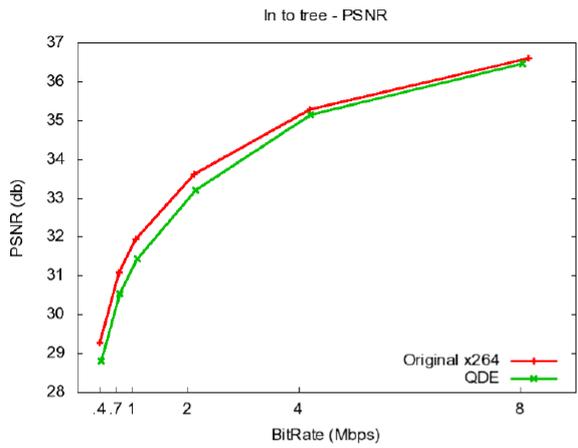
| | | | | | | |
|------------|-------|-------|-------|-------|-------|-------|
| I-QP média | 50,17 | 49,97 | 48,88 | 46,69 | 40,84 | 35,46 |
| | 50,14 | 49,93 | 48,85 | 46,07 | 40,27 | 34,92 |



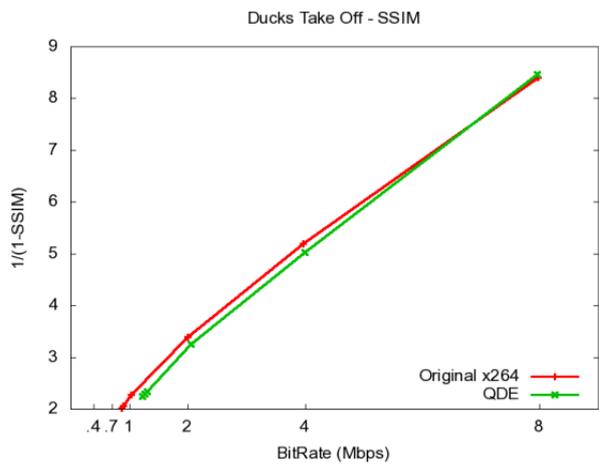
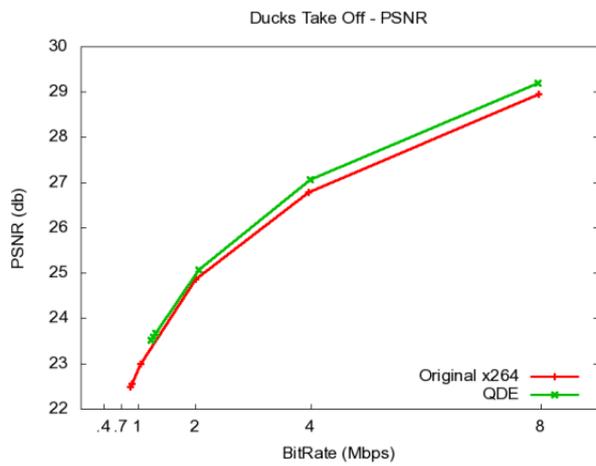
| | | | | | | |
|------------|-------|-------|-------|-------|-------|-------|
| I-QP média | 46,19 | 40,57 | 36,73 | 30,21 | 24,93 | 21,50 |
| | 46,20 | 39,07 | 35,28 | 29,11 | 24,15 | 20,89 |



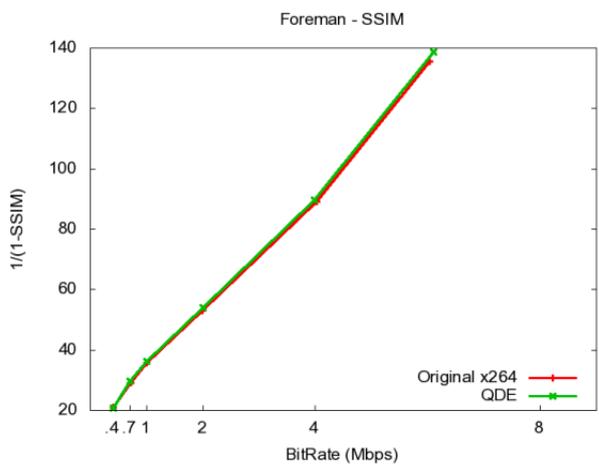
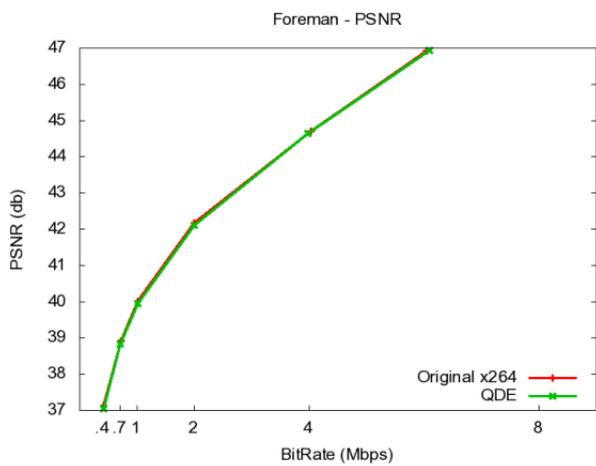
| | | | | | | |
|------------|-------|-------|-------|-------|-------|-------|
| I-QP média | 48,95 | 46,48 | 44,70 | 38,95 | 35,00 | 30,80 |
| | 49,00 | 45,56 | 42,29 | 38,02 | 34,10 | 30,13 |



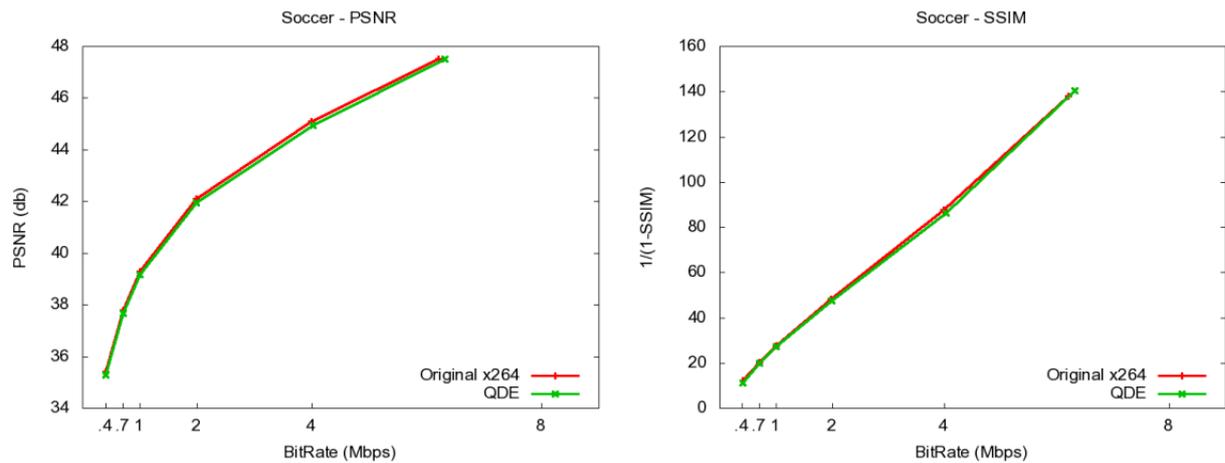
| | | | | | | |
|------------|-------|-------|-------|-------|-------|-------|
| | 41,55 | 36,18 | 33,57 | 29,72 | 26,19 | 23,27 |
| I-QP média | 39,96 | 34,54 | 32,18 | 28,72 | 25,48 | 22,76 |



| | | | | | | |
|------------|-------|-------|-------|-------|-------|-------|
| | 50,18 | 48,39 | 47,02 | 41,49 | 37,52 | 33,62 |
| I-QP média | 50,18 | 48,39 | 45,54 | 40,69 | 36,71 | 32,78 |

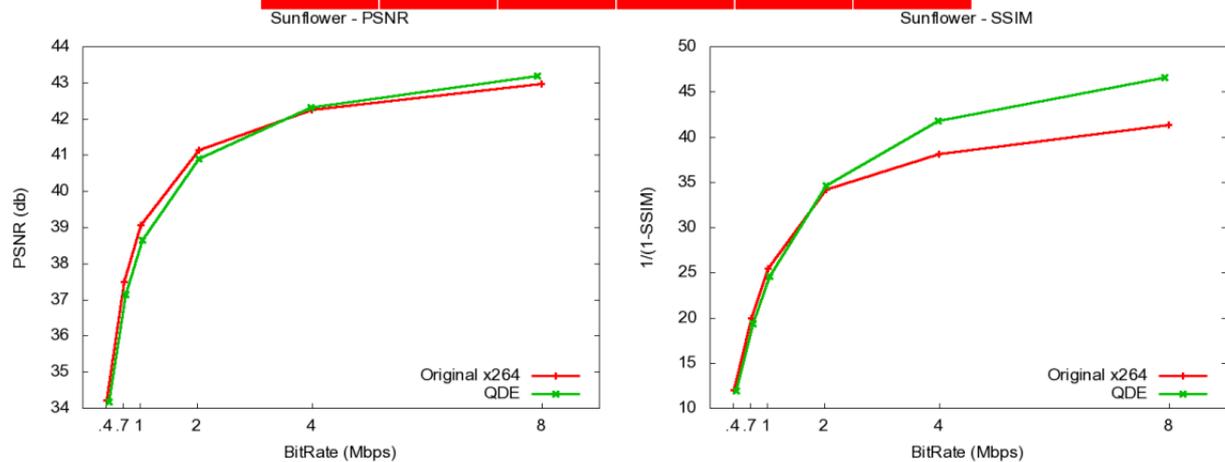


| | | | | | | |
|------------|-------|-------|-------|-------|-------|-------|
| | 25,03 | 21,89 | 19,90 | 16,04 | 12,13 | 10,77 |
| I-QP média | 24,58 | 21,42 | 19,48 | 15,54 | 11,84 | 10,60 |



| | | | | | |
|-------|-------|-------|-------|-------|-------|
| 28,41 | 24,46 | 22,03 | 17,70 | 13,42 | 12,04 |
| 27,53 | 23,70 | 21,50 | 17,24 | 13,09 | 11,80 |

I-QP média



| | | | | | |
|-------|-------|-------|-------|-------|-------|
| 41,76 | 32,58 | 28,81 | 23,37 | 19,64 | 17,05 |
| 40,12 | 31,03 | 27,47 | 22,39 | 18,95 | 16,46 |

I-QP média

Pela análise dos resultados apresentados nos gráficos podemos verificar uma relação entre a complexidade do vídeo e o ganho em qualidade. O uso da métrica proposta (QDE) proporcionou bons ganhos de qualidade para taxas de bit fixas moderadas principalmente em resolução alta.

6.2 Experimento B

Os testes fixam a qualidade SSIM obtida para o CRF 20 no compressor original. Efetua-se a iteração manual com a métrica proposta para atingir o mesmo SSIM obtido anteriormente, atingindo um novo CRF que proporciona a mesma qualidade SSIM do original.

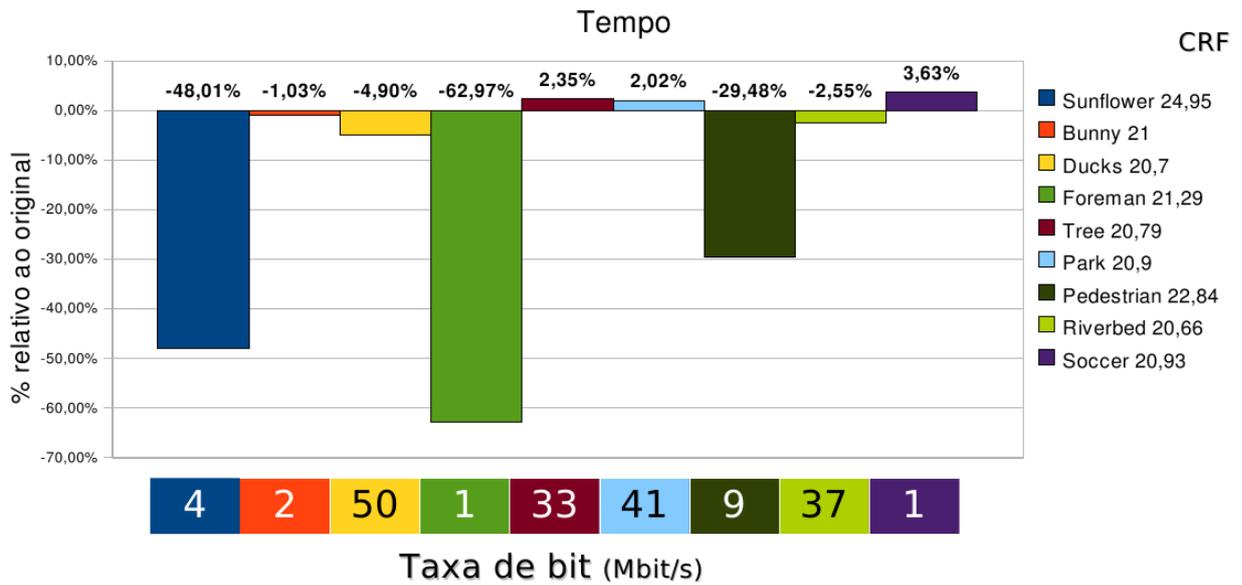


Figura 6.1: Gráfico ilustrando desempenho relativo, taxa de bit e novo CRF obtido.

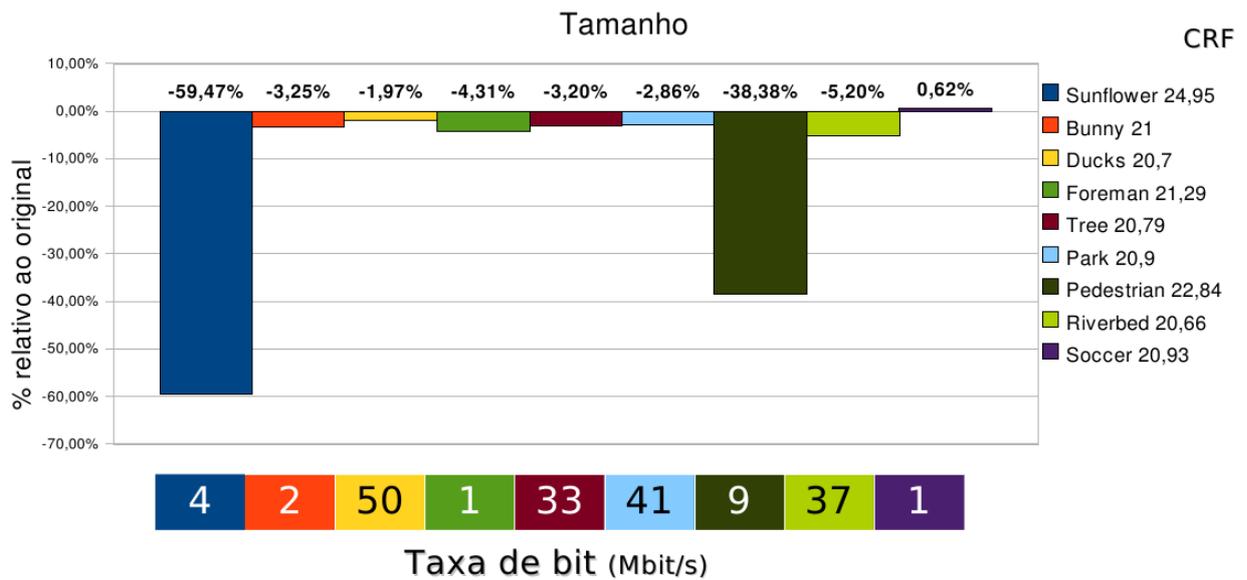


Figura 6.2: Gráfico ilustrando tamanho relativo, taxa de bit e novo CRF obtido.

De acordo com os resultados deste experimento, apresentados nos gráficos acima. A métrica proposta (QDE) conduz a excelentes ganhos em taxa de bit de até 60%, salientando que nos casos em que houve perda, a mesma não foi significativa. Destaca-se também o ganho em desempenho (quadros codificados por segundo) de até 63% no caso do vídeo *foreman* .

6.3 Experimento C

Estes testes obtém os diversos SSIM do vídeo *sunflower* numa faixa variável de CRF no compressor original. Efetua-se a iteração manual com a métrica proposta para atingir o mesmo SSIM nos casos originais. O objetivo deste experimento é investigar mais profundamente os efeitos da métrica proposta em um amplo intervalo de CRF no vídeo que demonstrou melhor resultado no experimento B. As curvas mostram o comportamento da métrica proposta (QDE) para uma faixa ampla de qualidades em um caso favorável. Observa-se, no geral, uma melhora no custo-benefício do processo de codificação. Destacando-se a melhora de qualidade no CRF 51 onde o SSIM original é 0.8765415 (9,0848 dB), e o proposto atinge 0.8981785 (9,9216 dB). Igualmente, o CRF 1 original tem SSIM 0.9971280 (25,4181 dB) e o proposto atinge 0.9972084 (25,5414 dB), lembrando-que a escala do SSIM é logarítmica.

Assim temos como conclusao que neste caso a faixa com melhores ganhos corresponde a faixa de CRF mais recomendada pela comunidade do x264 entre 15 e 30.

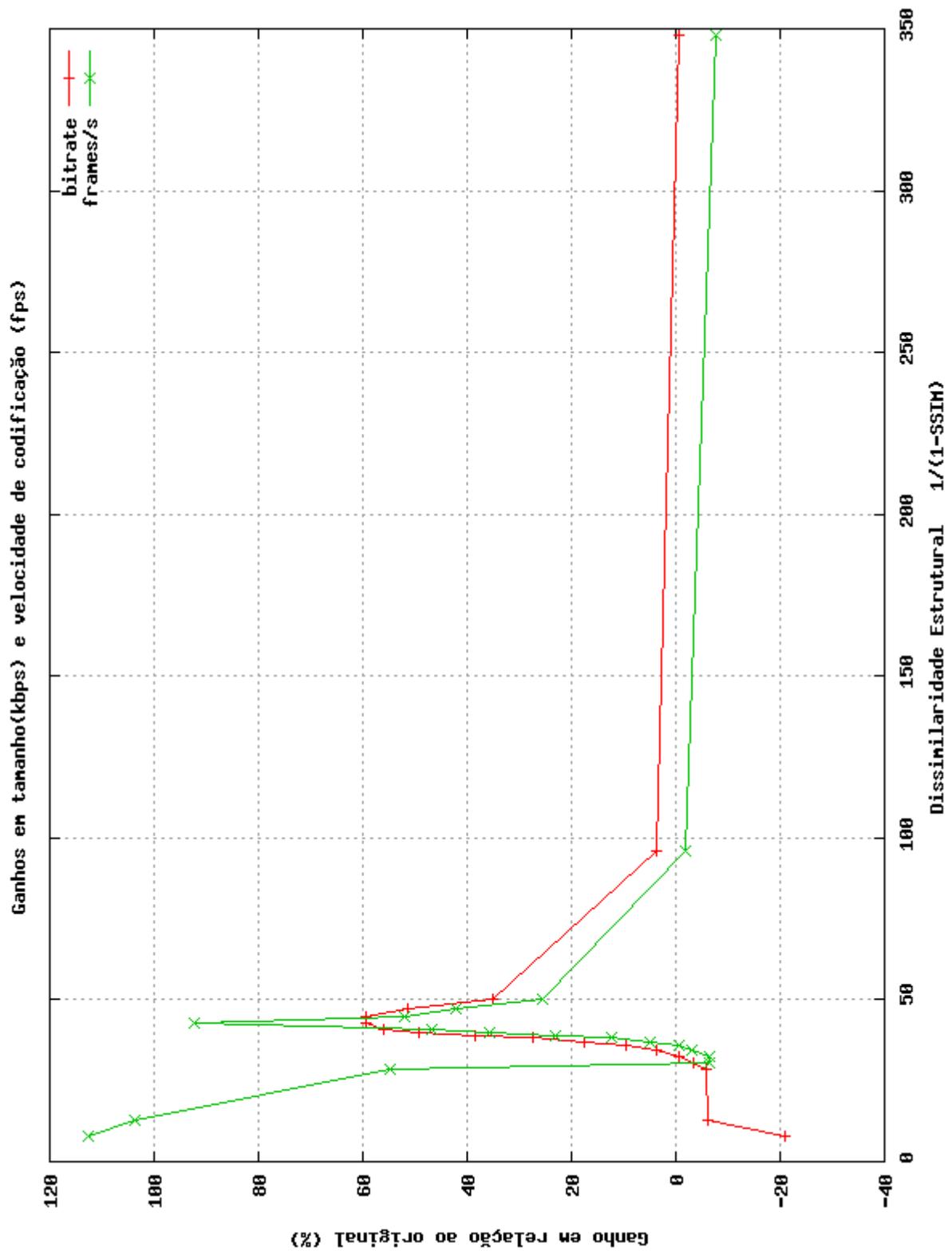


Figura 6.3: Gráfico com curva Taxa-Distorção usando a métrica DSSIM e uma curva do ganho de desempenho.

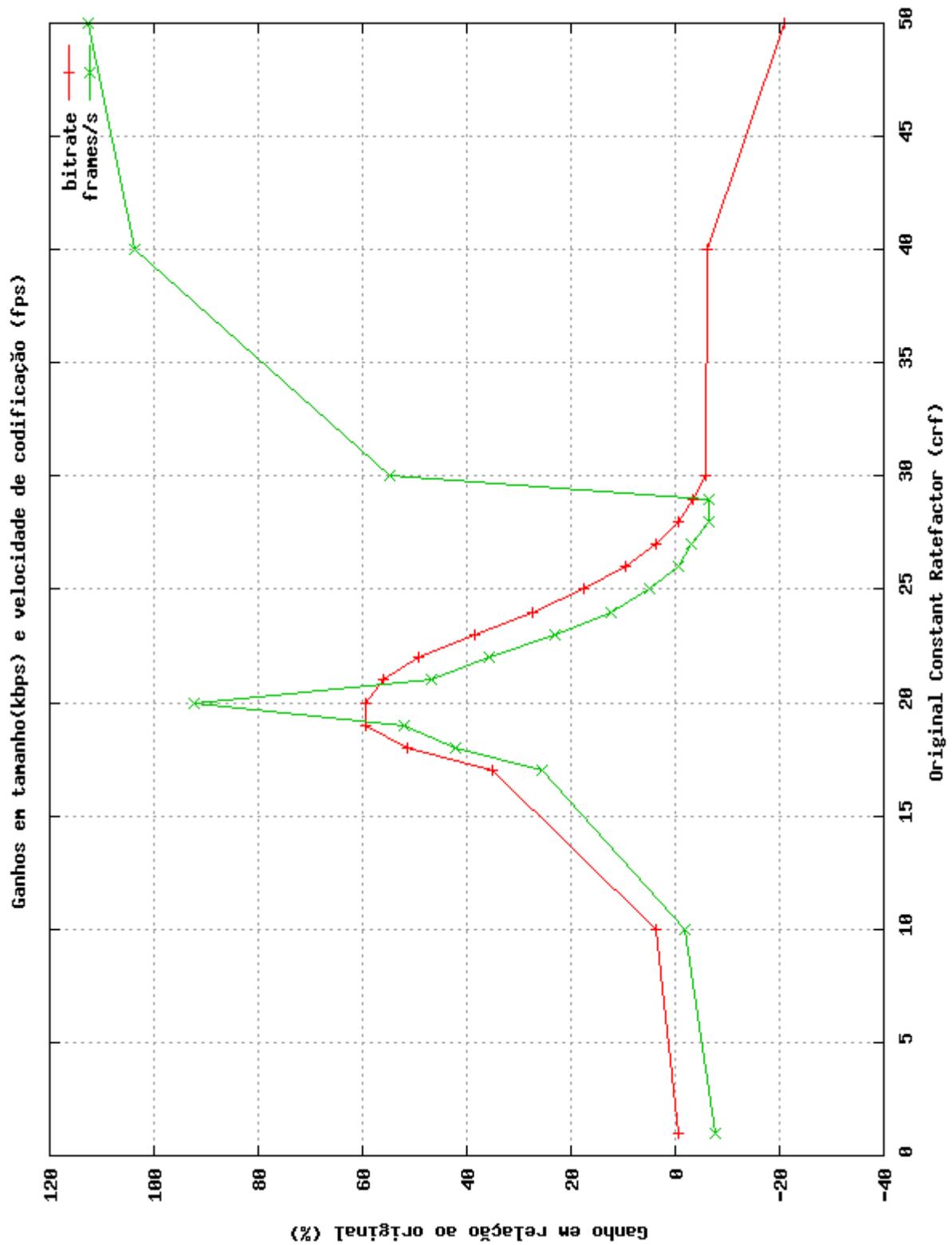


Figura 6.4: Gráfico com curva Taxa-Distorção usando os CRF originais e uma curva do ganho de desempenho.

7 *Conclusões*

A métrica proposta aproveita as características subjetivas e demonstra nos ensaios ganhos em taxa de bit entre 2% até 60%, dependendo do conteúdo do vídeo, para a mesma qualidade. Em outros ensaios foi percebido a melhora no custo-benefício da codificação em todos os parâmetros de quantização. Atinge também um aumento em desempenho em até 63%, proporcional à redução em taxa de bit na maioria desses casos. Foram realizados ensaios com a taxa de bit constante para verificar o impacto na qualidade. Notou-se uma melhora da qualidade pelas métricas PSNR e SSIM na maioria dos casos. Ganhos obtidos já são significativos mesmos sem uma sintonia fina no sistema de controle, mas esta tarefa está descrita nos trabalhos futuros. A complexidade de cálculo da métrica QDE proposta está entre a SATD e a SSD, de acordo com o número de elementos eliminados somado aos desvios condicionais necessários.

7.1 **Trabalhos Futuros**

Como trabalhos futuros está a realização de estudos subjetivos segundo o padrão ITU-R BT.500-11, com o objetivo de avaliar a correlação psicovisual da métrica, com uso dos resultados no processo de ajuste da métrica proposta. Quanto à métrica original (SATD), a comunidade do x264 realizou ajustes no sistema de controle, baseados em testes objetivos e subjetivos. Tais testes são realizados no decorrer de alguns meses, com o apoio de centenas de voluntários, e mostram melhorias na ordem de 50%. Para o uso efetivo da métrica de distorção proposta, e descobrir o seu real potencial, um procedimento semelhante ao feito com a SATD deve ser executado para a mesma. Além disso, existe a possibilidade da geração de arquiteturas dedicadas em hardware, com a obtenção de um IP visando integração em SoCs.

Referências Bibliográficas

- AGOSTINI, L. V. *Desenvolvimento de Arquiteturas de Alto Desempenho Dedicadas à Compressão de Vídeo Segundo o Padrão H.264/AVC*. Dissertação (Monografia) — UFRGS, 2007.
- CBLOOM. *Rants on Lagrange rate control*. Disponível em: <<http://cbloomrants.blogspot.com/2010/01/01-12-10-lagrange-rate-control-part-1.html>>.
- JVT. *JM JOINT VIDEO TEAM Reference Software*. Disponível em: <<http://iphome.hhi.de/suehring/tml/>>.
- MANOEL, E. T. M. *Codificação de Vídeo H.264 - Estudo de Codificação Mista de Macroblocos*. Dissertação (Monografia) — UFSC, 2007.
- MERRITT, L. et al. *Projeto x264*. Disponível em: <<http://developers.videolan.org/x264.html>>.
- MERRITT, L.; VANAM, R. Improved rate control and motion estimation for h.264 encoder. *ICIP*, p. 4, 2007.
- READER, C. *Video Coding IPR Issues*. 2002. [Http://www.avs.org.cn/avsdoc/2003-7-30/Cliff.pdf](http://www.avs.org.cn/avsdoc/2003-7-30/Cliff.pdf).
- RICHARDSON, I. *H. 264 and MPEG-4 video compression: video coding for next-generation multimedia*. [S.l.]: John Wiley & Sons Inc, 2003.
- TERRIBERRY, T. B. *Thusnelda: Modernizing Theora*. [Www.lca2010.org.nz/slides/50119.pdf](http://www.lca2010.org.nz/slides/50119.pdf).
- VIJAYAKUMAR. *A peek @ H.264*. 2008. Material de curso com Dr. K. R. Rao, em maio 2009.
- WANG, Z. et al. Image quality assessment: From error visibility to structural similarity. *IEEE TRANSACTIONS ON IMAGE PROCESSING VOL. 13 NO. 4 APRIL 2004*, p. 13, 2004.
- WINKLER. *Digital Video Quality: Vision Models and Metrics*. [S.l.: s.n.], 2005. ISBN 978-0470024041.
- WOLFRAM. *Parseval Theorem*. Disponível em: <<http://mathworld.wolfram.com/ParsevalsTheorem.html>>.
- WU, H.; RAO, K. *Digital Image Video Quality and Perceptual Coding*. [S.l.]: Boca Raton, FL: CRC, 2005.
- WU, M. *Digital Image and Video Processing – An Introduction*. 2004. Disponível em: <<http://www-ee.uta.edu/Dip/Courses/EE5356/631pub04,ec1intro.ppt>>.
- XIPH.ORG. *Xiph.org Test Media Repository*. Disponível em: <<http://media.xiph.org/>>.

Apêndice

Patch com as modificações realizadas no compressor x264 revisão 1564.

```

1 From 2075f4f2788be959ed6630d33dd17e47c0ee9520 Mon Sep 17 00:00:00 ↵
   2001
2 From: Bruno George de Moraes <brunogm0@gmail.com>
3 Date: Wed, 9 Jun 2010 19:58:17 -0300
4 Subject: [PATCH] Inclusion of unoptimized QDE metric in rdo.c/ssd_mb ↵
   ;
5 38% size gain in pedestrian.y4m;
6 59% size gain in sunflower.y4m;
7 Preliminary tests
8
9 ---
10 encoder/rdo.c | 172 ↵
   +++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++--
11 1 files changed, 167 insertions(+), 5 deletions(-)
12
13 diff --git a/encoder/rdo.c b/encoder/rdo.c
14 index 574a484..ea98c07 100644
15 --- a/encoder/rdo.c
16 +++ b/encoder/rdo.c
17 @@ -145,7 +145,7 @@ static inline int ssd_mb( x264_t *h )
18  {
19     int chromassd = ssd_plane(h, PIXEL_8x8, 1, 0, 0) + ssd_plane(h, ↵
   PIXEL_8x8, 2, 0, 0);
20     chromassd = ((uint64_t)chromassd * h->mb.i_chroma_lambda2_offset↵
   + 128) >> 8;
21 - return ssd_plane(h, PIXEL_16x16, 0, 0, 0) + chromassd;
22 + return qde_plane(h, PIXEL_16x16, 0, 0, 0) + chromassd;
23 }

```

```

24
25 static int x264_rd_cost_mb( x264_t *h, int i_lambda2 )
26 @@ -182,6 +182,168 @@ static int x264_rd_cost_mb( x264_t *h, int ↵
    i_lambda2 )
27     return i_ssd + i_bits;
28 }
29
30 +
31 /* Psy RD distortion metric: SSD plus "Quantization Distortion ↵
    Energy" */
32 /* Estimates the distortion and bit cost, using a table of threshold↵
    DCT coeff that are zero after quant. */
33 /*The table has QP51 values, so a right shift is needed to obtain ↵
    for the others (>> 8-floor( h->mb.i_qp /6)) .*/
34 +static const int quant4_zero_51[6][3] =
35 +{
36 + { 640, 1039, 1599 },
37 + { 703, 1119, 1800 },
38 + { 832, 1279, 2000 },
39 + { 896, 1440, 2300 },
40 + { 1023, 1599, 2500 },
41 + { 1151, 1840, 2899 },
42 +};
43 +static const uint16_t quant8_zero_51[6][6] =
44 +{
45 + { 640, 734, 399, 686, 500, 541 },
46 + { 703, 774, 437, 758, 559, 58 },
47 + { 832, 938, 525, 867, 660, 699 },
48 + { 896, 1019, 562, 939, 699, 745 },
49 + { 1023, 1141, 637, 1083, 799, 857 },
50 + { 1151, 1305, 725, 1228, 920, 970 },
51 +};
52 +static const int scan[16] = {0,3,4,3, 3,1,5,1, 4,5,2,5, 3,1,5,1};
53 +
54 +inline int qde_plane( x264_t *h, int size, int p, int x, int y )
55 +{
56 + uint64_t cost=0,cost8 = 0;
57 + int i = 0 ;
58 +

```

```
59 + switch( size )
60 +     {
61 +         case PIXEL_16x16:
62 +             for (i=(x + y*FENC_STRIDE)%4; i<4; ++i)
63 +                 cost8 += qde_plane_internal_8x8(h,size, p, x, y,i)←
64 +             ;
65 +             for (i=(x + y*FENC_STRIDE)%16; i<16; ++i)
66 +                 cost += qde_plane_internal_4x4(h,size, p, x, y,i);
67 +
68 +             break;
69 +         case PIXEL_16x8:
70 +             i=(x + y*FENC_STRIDE)%4;
71 +             cost8 += qde_plane_internal_8x8(h,size, p, x, y,i);
72 +             cost8 += qde_plane_internal_8x8(h,size, p, x, y,++i);
73 +
74 +             for (i=(x + y*FENC_STRIDE)%16; i<8; ++i)
75 +                 cost += qde_plane_internal_4x4(h,size, p, x, y,i);
76 +
77 +             break;
78 +         case PIXEL_8x16:
79 +             i=(x + y*FENC_STRIDE)%4;
80 +             cost8 += qde_plane_internal_8x8(h,size, p, x, y,i);
81 +             cost8 += qde_plane_internal_8x8(h,size, p, x, y,i+2);
82 +
83 +             for (i =(x + y*FENC_STRIDE)%16; i<16; i+=4 )
84 +                 {
85 +                     cost += qde_plane_internal_4x4(h,size, p, x, y,i);
86 +                     cost += qde_plane_internal_4x4(h,size, p, x, y,++i);
87 +                 }
88 +
89 +             break;
90 +         case PIXEL_8x8:
91 +             i = (x + y*FENC_STRIDE)%4;
92 +             cost8 += qde_plane_internal_8x8(h,size, p, x, y,i);
93 +
94 +
95 +
96 +             for ( i=(x + y*FENC_STRIDE)%16; i<i+4; ++i)
```

```

97 +             cost += qde_plane_internal_4x4(h,size, p, x, y,i);
98 +
99 +             break;
100 +         case PIXEL_8x4:
101 +     if(p == 0){
102 +         i=(x + y*FENC_STRIDE)%16;
103 +         cost += qde_plane_internal_4x4(h,size, p, x, y,i);
104 +         cost += qde_plane_internal_4x4(h,size, p, x, y,++i);
105 +     }
106 +         break;
107 +     case PIXEL_4x8:
108 +
109 +         i=(x + y*FENC_STRIDE)%16;
110 +         cost += qde_plane_internal_4x4(h,size, p, x, y,i);
111 +         cost += qde_plane_internal_4x4(h,size, p, x, y,i+4);
112 +
113 +     break;
114 +     case PIXEL_4x4:
115 +         i = (x + y*FENC_STRIDE)%16;
116 +         cost += qde_plane_internal_4x4(h,size, p, x, y,i);
117 +         break;
118 +     default:
119 +         x264_log( h, X264_LOG_ERROR, "qde_plane in rdo.c error\↵
120 +         n" );
121 +         break;
122 +     }
123 + return    ( ((cost - cost8) + ((cost>>32) - (cost8>>32))) );
124 + }
125 +
126 + inline int qde_plane_internal_4x4( x264_t *h, int size, int p, int x,↵
127 +     int y, int m )
128 + {
129 +     uint64_t qde = 0;
130 +     int q = h->mb.i_qp;
131 +     int qmod = q % 6;
132 +     int qf = 8-floor( q /6 );
133 +

```

```

134 +     uint8_t *fenc = h->mb.pic.p_fenc[p] + x + y*FENC_STRIDE;
135 +     uint8_t *fdec = h->mb.pic.p_fdec[p] + x + y*FDEC_STRIDE;
136 +
137 +     if(p == 0 && h->mb.i_psy_rd )
138 +     {
139 +         for( int i = 0; i < 16; i++ )
140 +         {
141 +             int qi = h->dct.luma4x4[m][i];
142 +             if (qi == 0) continue;
143 +             int j = (i&1) + ((i>>2)&1);
144 +             int qz = quant4_zero_51[qmod][j] >> qf;
145 +             if (qi >= qz)
146 +                 qde += abs( qi);
147 +             else
148 +                 {
149 +                     float k = 1/qz;
150 +                     qde += (uint32_t) (qi * qi) * k;
151 +                 }
152 +             }
153 +             qde = (qde* h->mb.i_psy_rd_lambda )>>1;
154 +         }
155 +     return h->pixf.ssd[size](fenc, FENC_STRIDE, fdec, FDEC_STRIDE) ←
        + qde;
156 + }
157 +
158 + inline int qde_plane_internal_8x8( x264_t *h, int size, int p, int x, ←
        int y, int m )
159 + {
160 +     uint64_t q8 = 0;
161 +
162 +     int q = h->mb.i_qp;
163 +     int qmod = q % 6;
164 +     int qf = 8-floor( q /6 );
165 +
166 +     uint8_t *fenc = h->mb.pic.p_fenc[p] + x + y*FENC_STRIDE;
167 +     uint8_t *fdec = h->mb.pic.p_fdec[p] + x + y*FDEC_STRIDE;
168 +
169 +     if( p == 0 && h->mb.i_psy_rd )
170 +     {

```

```

171 +         for( int i = 0; i < 64; i++ )
172 +             {
173 +                 int qi = h->dct.luma8x8[m][i];
174 +                 if(qi == 0)continue;
175 +                 int j = scan[((i>>1)&12) | (i&3)];
176 +                 int qz = quant8_zero_51[qmod][j] >> qf;
177 +                 if ( qi >= qz )
178 +                     q8 += abs( qi );
179 +                 else
180 +                     {
181 +                         double k = (qz ? 1/qz : 1);
182 +                         q8 += (uint32_t)(qi*qi) * k;
183 +                     }
184 +                 }
185 +                 q8 = (q8* h->mb.i_psy_rd_lambda )>>1 ;
186 +             }
187 +         return h->pixf.ssd[size](fenc, FENC_STRIDE, fdec, ←
FDEC_STRIDE) + q8;
188 +}
189 +
190 +
191 +
192 /* For small partitions (i.e. those using at most one DCT category s←
worth of CABAC states),
193 * it s faster to copy the individual parts than to perform a whole ←
CABAC_COPY. */
194 static ALWAYS_INLINE void x264_copy_cabac_part( x264_t *h, ←
x264_cabac_t *cb, int cat, int intra )
195 @@ -217,7 +379,7 @@ static uint64_t x264_rd_cost_subpart( x264_t *h, ←
int i_lambda2, int i4, int i_pi
196     if( i_pixel == PIXEL_4x8 )
197         x264_macroblock_encode_p4x4( h, i4+2 );
198
199 -     i_ssd = ssd_plane( h, i_pixel, 0, block_idx_x[i4]*4, block_idx_y←
[i4]*4 );
200 +     i_ssd = qde_plane( h, i_pixel, 0, block_idx_x[i4]*4, block_idx_y←
[i4]*4 );
201
202     if( h->param.b_cabac )

```

```

203     {
204 @@ -258,7 +420,7 @@ uint64_t x264_rd_cost_part( x264_t *h, int ←
        i_lambda2, int i4, int i_pixel )
205     chromassd = ssd_plane( h, i_pixel+3, 1, (i8&1)*4, (i8>>1)*4 )
206         + ssd_plane( h, i_pixel+3, 2, (i8&1)*4, (i8>>1)*4 );
207     chromassd = ((uint64_t)chromassd * h->mb.i_chroma_lambda2_offset←
        + 128) >> 8;
208 -     i_ssd = ssd_plane( h, i_pixel, 0, (i8&1)*8, (i8>>1)*8 ) + ←
        chromassd;
209 +     i_ssd = qde_plane( h, i_pixel, 0, (i8&1)*8, (i8>>1)*8 ) + ←
        chromassd;
210
211     if( h->param.b_cabac )
212     {
213 @@ -280,7 +442,7 @@ static uint64_t x264_rd_cost_i8x8( x264_t *h, int←
        i_lambda2, int i8, int i_mode
214     h->mb.b_transform_8x8 = 1;
215
216     x264_mb_encode_i8x8( h, i8, h->mb.i_qp );
217 -     i_ssd = ssd_plane( h, PIXEL_8x8, 0, (i8&1)*8, (i8>>1)*8 );
218 +     i_ssd = qde_plane( h, PIXEL_8x8, 0, (i8&1)*8, (i8>>1)*8 );
219
220     if( h->param.b_cabac )
221     {
222 @@ -300,7 +462,7 @@ static uint64_t x264_rd_cost_i4x4( x264_t *h, int←
        i_lambda2, int i4, int i_mode
223     uint64_t i_ssd, i_bits;
224
225     x264_mb_encode_i4x4( h, i4, h->mb.i_qp );
226 -     i_ssd = ssd_plane( h, PIXEL_4x4, 0, block_idx_x[i4]*4, ←
        block_idx_y[i4]*4 );
227 +     i_ssd = qde_plane( h, PIXEL_4x4, 0, block_idx_x[i4]*4, ←
        block_idx_y[i4]*4 );
228
229     if( h->param.b_cabac )
230     {
231 —
232 1.7.1.1

```