UNIVERSIDADE DO EXTREMO SUL CATARINENSE - UNESC

CURSO DE CIÊNCIA DA COMPUTAÇÃO

TIAGO PIZZETTI MEDEIROS

SEGMENTAÇÃO QUADTREE PARA SELEÇÃO DE BLOCOS NA CODIFICAÇÃO DE VÍDEO H.264/AVC

CRICIÚMA, JULHO DE 2011

TIAGO PIZZETTI MEDEIROS

SEGMENTAÇÃO QUADTREE PARA SELEÇÃO DE BLOCOS NA CODIFICAÇÃO DE VÍDEO H.264/AVC

Trabalho de Conclusão de Curso apresentado para obtenção do Grau de bacharel em Ciência da Computação da Universidade do Extremo Sul Catarinense.

Orientador: Prof. MEng. Evânio Ramos Nicoleit

CRICIÚMA, JULHO DE 2011

TIAGO PIZZETTI MEDEIROS

Segmentação Quadtree para Seleção de Blocos na Codificação de Vídeo H.264/AVC

Submetido ao corpo docente do Curso de Ciência da Computação da Universidade do Extremo Sul Catarinense como um dos requisitos para obtenção do grau de Bacharel em Ciência da Computação.

Profa. MSc. Ana Claudia Garcia Barbosa
Coordenadora do Curso de Ciência da Computação

Banca Examinadora:

Prof. MEng. Evânio Ramos Nicoleit (UNESC)
Orientador

Prof. MSc. Paulo João Martins (UNESC)

Profa. MSc. Leila Laís Gonçalves (UNESC)

Dedico este trabalho primeiramente a Deus; aos meus pais Ênio e Vitória; aos meus irmãos Jardel e Jaqueline; e a minha noiva Aline; pelo esforço, dedicação e compreensão, em todos os momentos desta e de outras caminhadas.

AGRADECIMENTOS

Agradeço a Deus, que me deu saúde e persistência nesta caminhada. Aos meus pais, por me ensinarem a trilhar sempre pelo caminho correto. Aos mestres, que com sua paciência, antes de me ensinarem, fizeram-me aprender!

Ao meu orientador Prof. MEng. Evânio Ramos Nicoleit que me ajudou nas pesquisas! Aos meus colegas de classe, por compartilharem informações e pelo convívio fraternal e familiar.

A Todos, o meu, MUITO OBRIGADO!

“Grandes realizações são possíveis quando se dá importância aos pequenos começos.” (Lao-Tsé)

RESUMO

Com o crescente aumento das capacidades de processamento dos dispositivos de computação e de transmissão de dados das redes, a demanda por aplicações de *streaming* de vídeo também aumenta. Entretanto, estas aplicações necessitam cada vez mais de compressão de vídeo para reduzir a largura de banda necessária para a transmissão, o que exige muito em termos de recursos computacionais. Por meio de técnicas de compressão na codificação de vídeo, é possível aprimorar o processo em termos de qualidade do vídeo resultante, de complexidade computacional, de taxa de bits, evitando atrasos na origem do *stream* e possibilitando atender aos requerimentos de uma aplicação para codificação de vídeo em tempo real. O padrão para codificação de vídeo H.264/AVC é o estado da arte em codificação de vídeo. Este projeto discute a utilização de uma estrutura de segmentação baseada em *quadtree* para seleção dos macroblocos a serem codificados pelo padrão H.264/AVC. Nessa estrutura, os macroblocos das imagens de diferença para a estimação e a compensação de movimento são segmentados em estruturas regulares de 16x16, 8x8 e 4x4 pixels. Os resultados indicam que o esquema de codificação descrito apresenta complexidade computacional reduzida em relação à estrutura do Padrão H.264/AVC convencional, mantendo-se equivalente a relação taxa-qualidade.

Palavras-chave: H.264. *Quadtree*. Codificação de vídeo. Complexidade Computacional.

ABSTRACT

With the increasing processing capabilities of computing devices and data transmission networks, the demand for video streaming applications also increases. However, these applications increasingly require video compression to reduce bandwidth required for transmission, which requires much in terms of computational resources. With compression techniques in video encoding, this process can be improved in terms of quality of resulting video, computational complexity, bitrate, avoiding delays at origin of the stream and meeting the requirements of an application for real time video coding. The H.264/AVC Advanced Video Coding standard is the state of the art. This project discusses the use of a structure based on *quadtree* segmentation for selection of macroblocks to be encoded by the H.264/AVC standard. In this structure, the macroblocks of difference images of the estimation and motion compensation are segmented into regular structures of 16x16, 8x8 and 4x4 pixels. The results indicate that the encoding scheme described has reduced computational complexity compared to the conventional structure of the H.264/AVC standard, keeping the equivalent bitrate-quality ratio.

Keywords: H.264. *Quadtree*. Video Coding. Computational Complexity.

LISTA DE ILUSTRAÇÕES

Figura 1. Divisão de um *frame* em macroblocos. 10

Figura 2. Exemplo de transição de quadros. 11

Figura 3. Comparação entre sinais analógico e digital. 23

Figura 4. Comparação entre TV analógica e digital. 25

Figura 5. Sequência Foreman: (a) Quadro 1; (b) Quadro 2; (c) Diferença entre Quadros. 27

Figura 6. Tipos de Predição: (a) Tipo P; (b) Tipos P e B; (c) Tipos P e B codificação..... 28

Figura 7. Determinação do vetor de movimento de um bloco na ME. 30

Figura 8. Modos de predição Intra 4x4. 34

Figura 9. Modos de predição Intra 16x16. 35

Figura 10. Modos de predição em blocos 8x8 de crominância. 35

Figura 11. Representação tridimensional da DCT. Antes e depois da transformação. 37

Figura 12. Quantização uniforme. 38

Figura 13. Varredura em zig-zag. 40

Figura 14. Estruturas em Árvore para Partição de Macrobloco: (a) Quad-Tree; (b) H.264. ... 46

Figura 15. Imagem da diferença entre dois frames (FD). 49

Figura 16. Imagem da diferença entre dois frames com compensação de movimento. 49

Figura 17. Método *up-bottom*. 51

Figura 18. Método *bottom-up*. 52

Figura 19. Estrutura da árvore resultante. 52

Figura 20. Cálculo para média e variância de blocos. 53

Figura 21. Composição real em *quadtree*. 54

Figura 22. Estrutura em bits da *quadtree*. 54

Figura 23. Saída de dados em tempo de codificação. 60

Figura 24. Resultados apresentados ao final da codificação. 61

Figura 25. Estrutura da função *Main.* 62

Figura 26. *Bitstream* de vídeo..... 63

Figura 27. Sequência de funções, *encode_sequence* à *code_a_picture.*..... 64

Figura 28. Alteração no método de segmentação. 68

Figura 29. Informações das sequências utilizadas. 69

Figura 30. Fórmula para comparar a qualidade objetiva de vídeos. 71

Figura 31. Fórmula para encontrar o MSE. 71

Figura 32. Fórmula para calcular o SAD. 72

Figura 33. Alocação de bits da sequência Carphone. 74

Figura 34. Alocação de bits da sequência Foreman. 74

Figura 35. Alocação de bits da sequência Salesman..... 75

Figura 36. PSNR da sequência Carphone. 76

Figura 37. PSNR da sequência Salesman. 76

Figura 38. PSNR da sequência Foreman. 77

Figura 39. Tempo de codificação dos quadros da sequência Carphone. 77

Figura 40. Tempo de codificação dos quadros da sequência Foreman. 78

Figura 41. Tempo de codificação dos quadros da sequência Salesman..... 78

Figura 42. Foreman, (a) original (b) H.264 (c) *quadtree.* 79

Figura 43. Salesman, (a) original (b) H.264 (c) *quadtree.* 79

Figura 44. Carphone, (a) original (b) H.264 (c) *quadtree*. 80

Figura 45. Total de bits gerado ao final da codificação. 81

Figura 46. Taxa bit rate gerada ao final de cada codificação. 81

Figura 47. PSNR resultante de cada sequência. 82

Figura 48. Tempo de codificação de cada sequência. 83

Figura 49. Resultado de PSNR para cada taxa de bits para a sequência Foreman. 84

Figura 50. Tempo de codificação para cada bit rate para a sequência Foreman. 85

Figura 51. Total de bit para as cinco segmentações..... 86

Figura 52. Taxa de bit rate para as cinco segmentações. 86

Figura 53. PSNR resultante para as cinco segmentações. 87

Figura 54. Tempo para a codificação das cinco segmentações. 87

LISTA DE TABELAS

Tabela 1. Formatos de transmissão da TV digital 23

Tabela 2. Bloco 8x8 de resíduos. 39

Tabela 3. Tabela de quantização do JPEG. 39

Tabela 4. Bloco transformado. 39

Tabela 5. Bloco transformado e quantizado. 40

Tabela 6. Níveis do H.264/AVC. 43

Tabela 7. Parâmetros de entrada do arquivo "*encoder_baseline.cfg*". 58

Tabela 8. Cabeçalho da saída de dados. 60

LISTA DE ABREVIATURAS E SIGLAS

AVC	Advanced Video Coding
CODEC	Codificador e Decodificador
DCT	Discrete Cosine Transform
DPCM	Differential pulse-code modulation
DVB-T	Digital Video Broadcasting — Terrestrial
EM	Estimação de movimento
H.263	Padrão para compressão de vídeo
H.263+	Padrão para compressão de vídeo (versão melhorada do H.263)
H.264/AVC	Padrão de compressão de vídeo (Advanced Video Coding)
ISO	International Organization for Standardization
ISDB-T	Integrated Services Digital Broadcasting Terrestrial
ITU	International Telecommunication Union
JPEG	Joint Photographic Experts Group
KBPS	kilobits por segundo
LZ77	Algoritmos de compressão de dados
LZ78	Algoritmos de compressão de dados
MCD	Motion Compensated Difference
MPEG	Motion Picture Experts Group
MPEG-1	Motion Picture Experts Group - One
MPEG-2	Motion Picture Experts Group - Two
MPEG-4	Motion Picture Experts Group - Tree
MV	Motion Vector
PSNR	Peak Signal-to-Noise Ratio
QT	Quadtree

SAD	Sum of Absolute Differences
SAMVIQ	Subjective Assessment Methodology for Video Quality
SVC	Scalable Video Coder
VCEG	Video Coding Experts Group
VQEG	Video Quality Experts Group
WMV	Windows Media Video

SUMÁRIO

1 INTRODUÇÃO 1
1.1 OBJETIVO GERAL 3
1.2 OBJETIVOS ESPECÍFICOS 4
1.3 JUSTIFICATIVA 4
1.4 ESTRUTURA DO TRABALHO 5
2 COMPRESSÃO DE DADOS 6
2.1 ESPAÇO E SUBAMOSTRAGEM DE CORES 6
2.2 COMPRESSÃO DE VÍDEOS DIGITAIS 9
2.3 COMPRESSÃO SEM PERDA DE DADOS 12
2.4 COMPRESSÃO COM PERDA DE DADOS 14
3 O PADRÃO DE CODIFICAÇÃO DE VÍDEO H.264/AVC 15
3.1 VÍDEOS DIGITAIS 16
3.1.1 História 17
3.1.2 Formatos de Vídeos Digitais 18
3.1.3 TV Digital no Brasil 22
3.2 MODELO TEMPORAL 25
3.2.1 Predição *Inter* 26
3.2.2 Predição de Quadros Predito (P) e Bipredito (B) 27
3.2.3 Estimação de Movimento (*Motion Estimation* - ME) 29
3.2.4 Compensação de Movimento (MC – *Motion Compensation*) 31
3.3 MODELO ESPACIAL 32
3.3.1 Predição *Intra* 32
3.3.2 Transformada 36
3.3.3 Quantização 37
3.3.4 Reordenamento 40
3.4 PERFIS E NÍVEIS DO PADRÃO H.264/AVC 41
4 ESTRUTURAS DE SEGMENTAÇÃO EM CODIFICAÇÃO DE VÍDEO 45
4.1 SEGMENTAÇÃO NO PADRÃO H.264/AVC 45
4.2 CODIFICANDO COM A QUADTREE 47
4.3 SEGMENTAÇÃO QUADTREE 50
5 TRABALHOS CORRELATOS 55

5.1 A dual quad-tree based variable block-size coding method 55
5.2 CODIFICADOR H.264/AVC COM COMPENSAÇÃO DE MOVIMENTO BASEADA EM PARTIÇÕES ALTERNATIVAS DE MACROBLOCO 55
5.3 UMA MÉTRICA PARA TAXA DE DISTORÇÃO VOLTADA PARA CODIFICAÇÃO DE VÍDEO PERCEPTIVA 56
6 ESTRUTURA EM QUAD-TREE NA SEGMENTAÇÃO DOS MACROBLOCOS.... 57
6.1 H.264/AVC JM REFERENCE 17.2 57
6.1.1 Parâmetros de Entrada 59
6.1.2 Formatos de Saída 59
6.2 MÓDULOS DO LENCOD ESPECIFIVADOS NO PADRÃO H.264 61
6.2.1 Função *Main* - main 61
6.2.2 Função *Encode One Frame* - encode_one_frame 63
6.2.3 Funções *Code a Picture* e *Code a Plane* - code_a_picture e code_a_plane 64
6.2.4 Função *Encode One Slice* - encode_one_slice 65
6.2.5 Funções *Start Macroblock e Encode One Macroblock* 66
6.3 ABORDAGEM POR MEIO DA ESTRUTURA QUADTREE 67
6.4 AMBIENTE DE TESTES E DE ANÁLISE 68
6.4.1 Sequências de Testes 68
6.4.2 Métricas Objetivas de Qualidade 70
6.4.3 Dados Utilizados para Análise 72
7 RESULTADOS E ANÁLISES 73
7.1 ANÁLISE QUADRO A QUADRO 73
7.1.1 Alocação de bits por quadro codificado 74
7.1.2 PSNR resultante de cada quadro 76
7.1.3 Tempo de codificação por quadro 77
7.1.4 Comparação, Frames Original e Codificado 79
7.2 MÉDIAS RESULTANTES 80
7.3 TESTES DE CODIFICAÇÀO EM DIFERENTES TAXAS DE BITS 83
7.3.1 Análise de Codificação com Diferentes Taxas no Bits 84
7.3.2 Segmentação com Blocos de Tamanho Único 85
8 CONCLUSÃO 89
REFERÊNCIAS 91
APÊNDICE A – ARTIGO CIENTÍFICO 96

1 INTRODUÇÃO

Na última década, a tecnologia tem apresentado grandes evoluções, tendo em vista as altas qualidades alcançadas em vídeo e imagem. Esse grande aumento na qualidade tem tido um custo altíssimo aos hardwares na parte de transmissão de dados. Organizações formadas por grandes empresas do setor tecnológico vêm constantemente aprimorando os padrões de mídia e de comunicação de dados para que os mesmos atendam à demanda exigida hoje pelos arquivos digitais.

Uma das últimas soluções veio com o lançamento do padrão *Blu-ray*, um disco óptico de alta densidade que entrou no mercado para suprir a demanda da indústria do cinema com seus vídeos de alta definição, e também para armazenamento de dados, tendo discos com capacidade de 25 GB e 50 GB, esse tipo de disco tem o mesmo formato físico de um DVD e tem como principal desenvolvedora a Sony.

Com a implantação da TV com sinal digital no Brasil, tornou-se necessário o estudo e implementação de melhorias nos modelos, sistemas e padrões de TV Digital existentes para se adaptarem ao meio brasileiro. O padrão de televisão digital adotado no Brasil é o ISDB-TB, uma adaptação do *Integrated Services Digital Broadcasting Terrestrial* (ISDB-T), padrão japonês acrescida de tecnologias desenvolvidas em pesquisas das universidades brasileiras (WIKIPÉDIA, 2010).

O padrão de referência japonês foi escolhido por atender melhor as necessidades de energia nos receptores e por ter uma mobilidade e portabilidade sem custo para o consumidor, diferente do padrão europeu (DVB-T), onde esta operação é tarifada pelas empresas telefônicas, a principal diferença adotada no padrão brasileiro em comparação com o japonês foi a substituição do formato de compressão MPEG-2

para o MPEG-4 que comporta o codec H.264 (ITU-T, 2005).

H.264 (ITU-T, 2005) é um padrão para compressão de vídeo, baseado no MPEG-4 Part 10/AVC (*Advanced Video Coding*). Uma das principais metas era fazer com que o padrão fosse capaz de fornecer uma boa qualidade de vídeo com uma taxa de bits muito baixa em relação aos padrões já existentes.

O MPEG-4 é utilizado primeiramente para compressão de dados digitais de áudio e vídeo e engloba um grande grupo de padrões para a codificação desses tipos de informações digitais. Ele é um padrão em constante desenvolvimento e divide-se em várias partes, as principais são: parte 2 (incluindo ASP, usado por *codecs* como DivX e Xvid) e a parte 10 (AVC/H.264, usado pelos *codecs* x264 e também no padrão de codificação para TVs Digitais no Brasil) (RICHARDSON, 2003).

O quadro de um vídeo digital[1] no padrão H.264/AVC, é dividido em macroblocos com tamanhos de 16x16, 16x8, 8x16, 8x8 (PURI, 2004). A escolha do tamanho da partição depende de alguns fatores, porém nas áreas do quadro do vídeo digital onde há mais movimento o tamanho de partição escolhido será menor que aqueles escolhidos para áreas onde se tem movimento reduzido.

Entretanto, o esforço computacional envolvido para seleção das áreas a serem codificadas a partir das imagens de diferença a serem codificadas e da seleção dos vetores de deslocamento no processo de estimação de movimento é significativo.

Uma técnica recente que exibe grande potencial para a representação de imagens em diferentes níveis de resolução é a decomposição em *quadtree* (QT). A QT é

[1] Um vídeo digital é formado por uma sequência de imagens que são formadas por vários pontos (pixels), cada imagem (quadro) do vídeo é dividida em *slices*, que por sua vez é dividido em macroblocos que podem ser divididos em outros blocos.

uma estrutura de dados hierárquica que possibilita a segmentação de uma imagem em regiões bidimensionais homogêneas. A homogeneidade é definida com respeito a uma dada propriedade de interesse em blocos com diferentes níveis de resolução (dimensões). A QT origina uma árvore, cujos nós dão origem a quatro blocos filho (sub-blocos). A cada particionamento, o tamanho dos sub-blocos resultantes é a quarta parte do seu predecessor. Cada nó, por sua posição na árvore, corresponde a um sub-bloco que é único em tamanho e posição dentro da imagem. O nó raiz, que está associado à imagem completa, deve ter dimensões de potência de 2 para possibilitar a aplicação recursiva do algoritmo de decomposição.

A proposta deste projeto de TCC visa o uso de uma estrutura de seleção de regiões a serem codificadas e seleção dos vetores de deslocamento baseando-se em uma estrutura de decomposição QT para reduzir a complexidade computacional nos onerosos processos de codificação.

Os caminhos que serão seguidos por este trabalho estão voltados à descrever e documentar o funcionamento do padrão de codificação de vídeo digital H.264/AVC como também a estrutura de segmentação *quadtree* e trazer a conhecimento as principais métricas aplicadas aos vídeos digitais.

1.1 OBJETIVO GERAL

Aplicar uma estrutura de segmentação em *quadtree* no processo de divisão dos macroblocos codificados pelo padrão H.264/AVC buscando uma redução do custo computacional com baixa perda de qualidade visual.

1.2 OBJETIVOS ESPECÍFICOS:

a) descrever e documentar o funcionamento do padrão de codificação de vídeo H.264/AVC;
b) descrever o funcionamento de uma estrutura de segmentação *quadtree*;
c) propor a implementação da estrutura de segmentação *quadtree* dentro do padrão de codificação H.264/AVC;
d) aplicar uma maneira de selecionar os macroblocos que serão codificados;
e) implementar a proposta e documentar os resultados para averiguação de possíveis ganhos.

1.3 JUSTIFICATIVA

Os arquivos de vídeo e imagem estão se tornando cada vês maiores e exigindo dos computadores, grandes capacidades de armazenamento. Esse grande aumento no tamanho dos arquivos não trouxe problemas apenas na área de armazenamento, mas também na parte de transferência dos dados pelas redes de computadores, que acaba se tornando muito demorado e inapropriado.

Nota-se que foi deixado de lado a implementação de novas técnicas de compressão de dados e dado prioridade ao desenvolvimento de novas formas de armazenamento, é de grande importância que seja estudado a possibilidade de se implementar novos algoritmos de compressão ou aprimorar os já existentes afim de

amenizar a demanda por espaço exigida hoje pelos arquivos.

Aplicando a estrutura de segmentação *quadtree* no padrão de codificação H.264 podem-se obter resultados positivos sobre a compressão de vídeos digitais difundidos hoje nos sistemas de televisão digital e principalmente em aplicações de *streaming*. Podemos ser mais otimistas, analisando a idéia fundamental de uma *quadtree*, de que qualquer imagem pode ser dividida em quatro quadrantes, sendo que cada quadrante pode ser dividido novamente em quatro subquadrantes e assim sucessivamente (FREDERICK, 1999).

Dessa forma estará sendo aplicado um novo método de segmentação dos macroblocos que serão codificados pelo codificador, um método mais simples que poderá reduzir o custo computacional.

1.4 ESTRUTURA DO TRABALHO

Este trabalho divide-se em oito capítulos apresentando no primeiro capítulo o porquê da proposta e os principais objetivos que se busca alcançar ao final deste projeto. Na sequência tem-se uma breve explicação de como funciona a compressão de dados e quais os métodos mais utilizados. Nos capítulos 3 e 4 constam informações sobre os modelos aos quais são aplicadas as principais técnicas de redução de dados na codificação de vídeos e detalhado o funcionamento da estrutura de segmentação especificada no padrão H.264 e a estrutura *quadtree* proposta. Em seguida serão apresentados alguns trabalhos relacionados a mesma área deste projeto, seguindo nos capítulos 6, 7 e 8 com a descrição do trabalho desenvolvido e apresentação dos resultados obtidos.

2 COMPRESSÃO DE DADOS

A compressão de dados consiste em diminuir o tamanho físico dos blocos de informações, um compressor utiliza um algoritmo que serve para otimizar os dados, por conseguinte torna-se necessário a utilização de um descompressor para reconstruir os dados originais. Dependendo do tipo de dados que se irá comprimir, são utilizados diferentes métodos de compressão, diferenciam-se os dados de imagem, som, vídeo e entre outros.

A compressão de dados também objetiva a retirar redundâncias, levando em consideração que muitos dados contêm informações repetidas que podem ou devem ser eliminadas de alguma forma. A forma utilizada para remover as redundâncias é uma regra especificada no codificador, que, quando seguida, elimina os bits redundantes de informações, de modo a diminuir o tamanho dos arquivos. Por exemplo, a sequência "AAAAAA" que ocupa 6 bytes, poderia ser representada pela sequência "6A", que ocupa 2 bytes, economizando 67% de espaço (RIBEIRO; TORRES, 2009).

2.1 ESPAÇO E SUBAMOSTRAGEM DE CORES

A forma como são representadas as cores em um vídeo digital está associada à maneira de como essas cores são interpretadas pelo sistema visual humano. O sistema humano possui os bastonetes e cones, que são altamente sensíveis à luz. Os bastonetes são utilizados para visão noturna, captando imagens de baixa resolução com poucos detalhes. Já os cones, são sensíveis as cores vermelha, verde, azul e amarela e identificam detalhes de alta resolução, más precisam de uma maior luminosidade para

funcionarem (POLLACK, 2006). Segundo autor as saídas dos cones são pré-processadas por neurônios presentes na retina, acima dos cones, estes neurônios codificam a imagem em um canal de luminância e dois canais de crominância, um canal gerência as o vermelho e o verde e outro canal o azul e o amarelo. Para Gonzalez (2003) o sistema visual humano, utilizando combinações de intensidades diferentes das cores captadas pelos cones é capaz de discernir milhares de cores distintas, por outro lado, consegui distinguir mais do que cinco dúzias de tons de cinza, que indicam a intensidade luminosa da imagem (luminância).

Um sistema para representar cores de forma digital é chamado de espaço de cores e a definição do espaço de cor a ser utilizado para representar um vídeo é essencial para a eficiência da codificação deste vídeo.

Existem vários espaços de cores para representar imagens digitais, tais como: HSI, RGB, YCbCr. O RGB é um dos mais conhecidos, sendo que é este o espaço de cores utilizado nos monitores coloridos, o RGB representa as três cores primárias captadas pelo sistema visual humano em três matrizes diferentes: vermelho, verde e azul. No espaço de cores YCbCr, as três componentes utilizadas são luminância (Y), que defina o brilho ou a intensidade luminosa, a crominância azul (Cb) e a crominância vermelha (Cr) (BHASKARAN, 1997).

Segundo Richardson (2002) os componentes R, G e B são muito correlacionados, não sendo propícios para a compressão de vídeos. Por esse motivo a compressão de vídeo e aplicada nos espaços de cores do tipo luminância e crominância, que é o caso do espaço YCbCr. Outra vantagem do YCbCr em relação ao RGB, é que no YCbCr as informações sobre as cores são completamente separadas da informação de brilho, deste modo, estas informações podem ser tratadas de forma diferenciada pelos

codificadores de imagens estáticas e de vídeos.

O sistema visual humano possui uma maior sensibilidade às informações de luminância do que das de crominância, os padrões de compressão de imagem e vídeo exploram esta característica humana onde através da redução da taxa de amostragem dos componentes de crominância em relação aos de luminância, conseguem uma maior eficiência na codificação (RICHARDSON, 2002). Esta operação é chamada de subamostragem de cores e é realizada sob o espaço de cores YCbCr nos padrões de compressão de vídeo atuais.

Existem várias maneiras de relacionar os componentes de crominância e luminância para realizar a subamostragem. Os formatos mais comuns são o 4:4:4, o 4:2:2 e o 4:2:0. No formato 4:4:4, para cada quatro amostras de luminância (Y), existem quatro amostras de crominância azul (Cb) e quatro amostras de crominância vermelha (Cr). Os três componentes de cor possuem a mesma resolução e existe uma amostra de cada elemento de cor para cada pixel da imagem. No formato 4:2:2, para cada quatro amostras de Y na direção horizontal, existem apenas duas amostras de Cb e duas amostras de Cr. Neste caso, as amostras de crominância e luminância possuem a mesma resolução vertical, más metade da resolução horizontal. No formato 4:2:0, para cada quatro amostra de Y, existe apenas uma amostra de Cb e uma amostra de Cr. Neste caso, as amostras de luminância possuem o dobro da resolução vertical e horizontal das amostras de crominância (RICHARDSON, 2003).

A subamostragem de cor aumenta muito a eficiência da codificação, uma vez que parte da informação da imagem é simplesmente descartada, sem causar impacto visual perceptível. O formato 4:2:0 por exemplo, onde cada componente de crominância possui um quarto das amostras existentes no componente de luminância, então um vídeo

YCbCr no formato 4:2:0 irá utilizar metade das amostras necessárias à uma vídeo RGB ou YCbCr no formato 4:4:4, implicando em uma taxa de compressão de 50% considerando apenas a subamostragem.

Segundo Bhaskaran (1997) o espaço de cores YCbCr é utilizado pelo padrão de vídeo H.264/AVC onde o mesmo considera que os dados de entrada do vídeo trabalham neste espaço de cores. Apesar de permitir os formatos 4:2:0, 4:2:2 e o 4:4:4 o formato mais utilizado é o 4:2:0, que é o mais adequado para a percepção do sistema visual humano.

2.2 COMPRESSÃO DE VÍDEOS DIGITAIS

Um vídeo é composto por várias imagens projetadas em sequência, cada imagem desta sequência é chamada de quadro e a quantidade de imagens que são projetadas por segundo chamada-se cadência, a medida utilizada na cadência é o fps (*frames* por segundo). Quanto mais *frames*/quadros por segundo tiver um vídeos mais realista e maior será a qualidade da imagem. Os vídeos digitais normalmente trabalham com o mesmo fps das TV, que é de 30 frames por segundo (SALOMON, 2004).

Para trabalhar os vídeos e obter um resultado final satisfatório, os codificadores de vídeos costumam dividir os quadros em macroblocos, principal unidade de codificação. Esses MBs possuem 16 amostras de largura e 16 amostras de altura (16 x 16) e subdividem-se conforme a movimentação entre os quadros em blocos de 16 x 8, 8 x 8, 8 x 4 e 4 x 4. Podemos observar na Figura 1 a divisão de um *frame* em vários blocos:

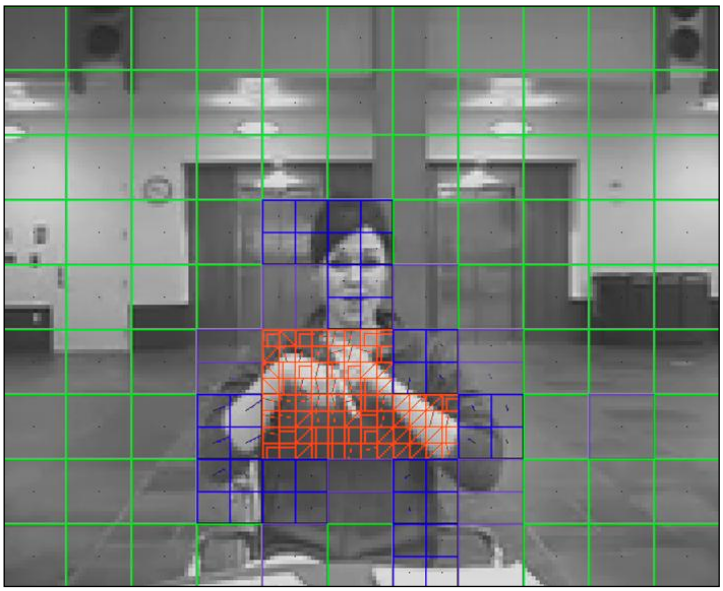


Figura 1. Divisão de um *frame* em macroblocos.
Fonte: MOBILEASL. (2010).

Uma dos métodos utilizados para diminuir o tamanho do vídeo é justamente reduzindo a quantidade de quadros por segundo que além de reduzir o tamanho do vídeo, consequentemente reduzirá a qualidade quanto ao realismo do vídeo, pois haverá quebra de quadro, isto é, os movimentos do vídeo ficarão "truncados".

Utiliza-se também para reduzir o tamanho dos vídeos, uma técnica de compressão de imagem, que através da analise dos macroblocos possibilita remover das imagens informações que já foram projetadas. Um exemplo seria o de um vídeo onde se tem uma pessoa parada apenas falando, no quadro inicial a imagem é projetada por completa, mas nos quadros seguintes os pedaços da imagem que são idênticos ao quadro anterior são removidos atualizando apenas a área em que há movimento, se só a boca esta se movimentando, somente a área da boca será desenhada nos quadros seguintes. Pelo fato de mostrar apenas o primeiro quadro por completo, esta técnica economiza uma quantidade enorme de espaço, os demais quadros só mostram o que muda em relação ao quadro anterior. Esses quadros incompletos são chamados quadros delta (delta *frames*) (RIBEIRO e TORRES, 2009).

A Figura 4 nos dá um ótimo exemplo da transição de quadros onde o sistema citado acima funciona perfeitamente. Observa-se que o Buzz (personagem criado pela ©Disney/Pixar) move-se para a direita e mexe a cabeça, mas o restante do corpo se mexe muito pouco. Utilizando o *codec* de compressão H.264/MPEG-4 são trocadas apenas as partes que se movem. Além do Woody (personagem criado pela ©Disney/Pixar) e do fundo, as partes do Buzz que se moveram, mas não sofreram modificação entre os quadros também são mantidas (MORIMOTO, 2009):

Figura 2. Exemplo de transição de quadros.
Fonte: MORIMOTO, C. E. (2009).

No padrão MPEG os quadros delta podem ser classificados em quadros P (preditivos) ou quadros B (bidirecionais). Os quadros P funcionam da maneira descrita, enquanto os quadros B podem ainda ter a diferença não só para o quadro anterior, mas também a diferença para o quadro seguinte na sequência, daí o nome “bidirecional”.

O problema é que por conta desta técnica, não haveria como você usar os recursos de avanço e retrocesso do seu tocador de mídia, pois ele precisaria tocar o filme desde o início para poder construir uma imagem que esteja no meio do filme, já que no meio do filme só haverá a informação do que é diferente para o quadro anterior e não uma imagem completa (RIBEIRO; TORRES, 2009).

Por isso, de tempos em tempos é necessário inserir um quadro completo (como o primeiro quadro do filme) para que os recursos de avanço e retrocesso possam ser usados. Esses quadros completos são chamados quadros-chave (*key frames*) ou quadros I (*I-frames*). Quanto mais quadros-chave seu vídeo tiver, maior ele será (pois mais imagens completas, que ocupam mais espaço, serão inseridas), mas em compensação mais pontos de avanço e retrocesso existirão. Você precisa esperar o tocador chegar a um quadro-chave para que ele consiga mostrar o vídeo e quanto menos quadros-chave o vídeo tiver mais frequente será este problema em seu vídeo.

Conforme Ribeiro e Torres (2009) além da técnica descrita acima, cada quadro é comprimido usando um algoritmo baseado em perda de dados, da mesma forma que ocorre com imagens no formato JPEG e áudio no formato MP3, por exemplo. Isso significa que o vídeo comprimido não tem a mesma qualidade do vídeo original.

A compressão de dados classifica-se em compressão sem perdas e compressão com perdas. Na compressão sem perdas, após o processo de compressão/descompressão, os dados reconstituídos ficam idênticos ao original, enquanto na compressão com perdas há uma certa alteração desses dados (KIOSKEA, 2009).

2.3 COMPRESSÃO SEM PERDA DE DADOS

A compressão sem perdas consegue, geralmente, compactar dados de duas a três vezes e é aplicada em áreas que exigem que o processo de compressão e descompressão seja livre de qualquer perda de informação, como nas imagens médicas digitais, transmissão de textos, programas executáveis, banco de dados e entre outros.

Os seus algoritmos são divididos em duas categorias: ou são baseados em dicionários ou em métodos estatísticos. Os métodos baseados em dicionário geram um arquivo comprimido contendo códigos de comprimento fixo, de 12 a 16 bits, onde cada código representa uma sequência particular de valores dos dados originais; não precisa do conhecimento da frequência com que os símbolos fontes ocorrem na informação original, caso que acontece no método de conhecimento estatístico (MORIMOTO, 2009).

No método estatístico utilizam-se códigos de comprimento variável, os dados da informação original que aparecem com maior frequência são representados por palavras-código de menor tamanho, enquanto os dados de menor incidência são representados por palavras-código maiores.

Alguns dos principais algoritmos de compressão sem perda de dados são os de Huffman e Lembel - Ziv. O código de Huffman é um método para codificar um texto de forma a obter uma compactação que seja ótima dentro de certos critérios. A construção desse código foi desenvolvida por David Huffman, que utilizou a estrutura de árvore binária, de forma a gerar um código binário (SALOMON, 2004).

No final da década de 70 Jacob Ziv e Abraham Lempel desenvolveram dois algoritmos para compressão de dados, o LZ77 e LZ78. Eles são compressores que, sempre que uma frase é repetida, substituem a ocorrência original da frase por uma referência.

Conforme o autor a compressão sem perdas limita-se a um pequeno nicho de aplicações que não toleram qualquer distorção, como vídeos médicos ou sistemas de arquivamento. Por sua vez, a compressão com perdas é a mais usada e difundida, já que certas distorções podem ser imperceptíveis ao olho humano, ou mesmo toleradas – é

este o tipo de compressão que trata o padrão H.264 aqui apresentado.

2.4 COMPRESSÃO COM PERDA DE DADOS

A compressão com perdas, ao contrário da compressão sem perdas, permite que sejam eliminadas algumas informações para se ter uma melhor taxa de compressão, conservando ao mesmo tempo um resultado que seja o mais próximo possível dos dados originais. É o caso, por exemplo, de certas compressões de imagens, vídeos ou sons, como o MP3 e o AVI.

Os arquivos executáveis, por exemplo, não podem ser comprimidos por este tipo de método porque os mesmos têm a necessidade de serem íntegros para funcionarem. Já os dados multimídia (áudio, vídeos) toleram um certo nível de degradação sem que os captores sensoriais (olho, tímpano e entre outros.) distingam uma degradação significativa, sendo mais vantajosa a utilização da compressão com perdas nesse tipo de arquivo.

Afirma Morimoto (2009) que a taxa resultante da compressão com perdas depende apenas da distorção admitida pelos arquivos, as tecnologias de compressão mais recentes podem comprimir a uma taxa de até 100 vezes menor do que a taxa original, com uma distorção ainda aceitável.

Este método de compressão é utilizado pelo padrão H.264 de compressão de vídeo, também é utilizado na compressão de imagem e som gerando arquivos do tipo JPEG, GIF, MP3, AAc e entre outros (MEGRICH, 2009).

3 O PADRÃO DE CODIFICAÇÃO DE VÍDEO H.264/AVC

O H.264/AVC é um padrão desenvolvido pelo *Joint Video Team* (JVT) para a codificação de vídeo, o JVT é formado conjuntamente pelo *Video Coding Experts Group* (VCEG), um grupo da ITU-T (grupo ITU-T SG16 Q.6) e o MPEG, grupo da ISO/IEC (grupo ISO/IEC JTC 1/SC 29/WG 11).

A União Internacional de Telecomunicações (ITU) é uma agência da ONU que trabalha apenas no campo das telecomunicações. O Setor de Padronização em Telecomunicações da ITU (ITU-T) é responsável por estudos de questões técnicas, operacionais e de tarifação que visam padronizar a telecomunicação no mundo. A Organização Internacional para Padronização (ISO), organização não governamental, é composta pelos institutos nacionais de padronização de 157 países e funciona como uma ponte entre os setores público e privado (RIBEIRO e TORRES, 2009).

O Padrão H.264 é conhecido também como MPEG-4 PART 10, ele é uma evolução de outros padrões de codificação de vídeo, os quais são: ITU-T H.261 , H.262 (MPEG-2), H.263 e seus sucessores H.263+ e H.263++. Seu desenvolvimento se iniciou pela necessidade de melhorar a qualidade das compressões de vídeo, com o intuito de sua utilização em vídeo conferências, armazenagem em meio digital, radio-difusão de sinal de TV, vigilância remota por vídeo e entre outros. As primeiras propostas surgiram em 1998 sugeridas pelo VCEG, que visavam duplicar a eficiência de codificação. Em 1999 surgiu a primeira versão.

Em qualquer tipo de sinal, o processo de compressão sempre visa a trabalhar sobre a redundância de informações, no caso de vídeos não é diferente, entretanto, existem dois tipos de redundância nos vídeos: redundância temporal e espacial. A

redundância espacial é feita pela correlação entre pixels vizinhos num mesmo quadro, já a redundância temporal é feita pela semelhança entre quadros consecutivos. Em mudanças de cenas os codificadores não são tão eficientes, pois a correlação existente entre os quadros e muito baixa.

Atualmente os padrões de codificação que atuam nos sistemas de TV digital baseiam-se no modelo modulação diferencial de código de pulso (DPCM/DCT). O termo DCT é por causa da transformada de cossenos discreta que busca a concentração de energia em uma quantidade menor de amostras.

3.1 VÍDEOS DIGITAIS

Um vídeo nada mais é do que uma sucessão de várias imagens apresentadas por um determinado tempo. Pelo fato de o olho humano com suas limitações ser capaz de distinguir apenas 20 imagens por segundo, a afixação de mais de 20 imagens gera ao cérebro a sensação de se estar observando uma imagem animada que parece haver movimento (KIOSKEA, 2009).

Segundo Kioskea (2009) no vídeo digital há a sucessão de imagens digitais afixadas a certo ritmo, essas imagens, dependendo de cada formato de vídeo, possuem características padrões que devem ser respeitadas para que o vídeo apresente a qualidade desejada como resolução e qualidade de compressão. Os vídeos multimídia que são utilizados em grande escala, são vídeos digitais acompanhados de dados de áudio.

São inúmeras as aplicações para os vídeos digitais, podemos destacar o vídeo doméstico, onde é utilizada uma filmadora digital para a gravação do vídeo e em

seguida o mesmo é reproduzido em um aparelho de DVD, o vídeo trabalhado no computador (gravado em filmadora e editado no PC), o vídeo para cinema ou televisão e o vídeo para Internet (webcam, teleconferências e entre outros.) (SANADA; SANADA, 2004).

3.1.1 História

O formato de vídeo digital foi lançado em 1996 primeiramente em fitas cassetes, onde os vídeos eram registrados através de uma leve compressão da imagem nas fitas, o que facilitava depois, a transferência do vídeo para um computador onde eram feitas as edições (WIKIPÉDIA, 2004).

As fitas existem ainda hoje nos seguintes formatos: DV, MiniDV, DVCAM, Digital8, DVCPRO, DVCPRO50 e DVCPRO HD. Elas registram um vídeo digital comprimido graças a um método DCT. A qualidade do vídeo digital é superior aos formatos analógicos atuais, como o video8, VHS-C ou o Hi-8.

Um amplo consórcio de grandes empresas foi quem desenvolveu o formato DV, empresas como Matsushita (dona da Panasonic), Philips, Sony, Thomson, juntas com a Hitachi, JVC, Mitsubishi, Sanyo, Sharp e Toshiba, mais as empresas de informática Apple e IBM formaram esta aliança para definir as especificações da nova geração de fitas (MUSEUDOCOMPUTADOR, 2004).

A fita cassete se manteve no mercado por um longo tempo, até meados de 2005 ainda encontrava-se fitas nas prateleiras das locadoras. O seu fim era previsto já no inicio de 1990, quando dois tipos de discos-ópticos de alta capacidade já se encontravam em desenvolvimento, um era o MultiMedia *Compact Disc* (MMCD),

liderado pela Philips e Sony, e o outro era o Super *Density Disc* (SD), patrocinado pela Toshiba, Time-Warner, Matsushita Electric (Panasonic), Hitachi, Mitsubishi, Pioneer, Thomson e JVC (WIKIPÉDIA, 2004).

Philips e Sony abandonaram o formato MMCD e seguiram com o formato da Toshiba, más com duas modificações referentes à tecnologia implicada. A primeira foi a aplicação de uma geometria que permitisse o "push-pull" (pular faixas como num CD), a segunda foi a adoção do sistema Philips EFMPlus. O resultado final apareceu em 1996 com o surgimento do DVD 1.5 de 4.7 GB (TORRES, 1998).

Os *DVD Players* apareceram primeiramente no Japão em 1997, em seguida nos Estados Unidos da América em 1998, em 1999 na Europa, 2000 na Austrália e só começou a ganhar força no Brasil em 2002 e 2003, o primeiro filme foi lançado em 1996.

3.1.2 Formatos de Vídeos Digitais

Existe uma grande variedade de formatos de vídeo digital, apesar de todos terem a mesma finalidade cada um possui algumas características particulares, o que possibilita ao usuário dependendo de suas necessidades, optar entre o uso de um ou outro formato. Nos formatos mais simples podemos encontrar diferentes tipos de *codecs* de áudio enquanto nos formatos mais avançados é comum a presença de múltiplas transmissões de áudio e vídeo, legendas, informações sobre capítulos e metadados (LIMA, 2010).

Segundo Lima (2010) entre os formatos mais populares podemos encontrar o mov, mpeg, avi, wmv entre outros. Ambos os formatos visam conseguir a melhor

sincronia entre o tamanho e a qualidade do vídeo, quanto maior a qualidade por consequência maior será o tamanho do vídeo para ser armazenado em uma mídia digital.

3.1.2.1 Mov

Formato multimídia utilizado pelo software QuickTime para o armazenamento de vídeos digitais, este software foi lançado pela Apple Computer em 1991 e atualmente encontra-se na versão 7.6.8.75.0. Das principais características do formato mov destacam-se, a capacidade de manipular formatos de vídeo digital, mídia clips, som, texto, animação, música e vários tipos de imagens panorâmicas interativas (FELITTI, 2007).

O formato mov utiliza o codec de compressão H.264/MPEG-4 AVC, o mais recomendado codec de compressão de vídeo e mais atual, amplamente utilizado para comprimir vídeos com transmissão em tempo real.

3.1.2.2 Mpeg

O *Moving Picture Experts Group* (MPEG) foi formado pela ISO para definir padrões para a compressão e transmissão de áudio e vídeo, incluir aproximadamente 350 membros oriundos de várias indústrias, universidades e instituições de pesquisa. Seus primeiros padrões para o desenvolvimento de vídeos digitais foram:

a) vídeo e áudio associados a uma taxa de 1.5 Mbps (mais tarde chamado de MPEG-1);

b) imagens em movimento e áudio associados a uma taxa de 10 Mbps (mais tarde chamado de MPEG-2).

O MPEG-1 foi orientado como imagem digital armazenada em Mídia de armazenagem digital (DSM - *Digital Storage Media*). MPEG-2 foi orientado como broadcast[2] e para televisão de alta-definição (HDTV) e o MPEG-4, que é um padrão utilizado principalmente para a compressão de dados de áudio e vídeo, possui muitas das funcionalidades do MPEG-1 e MPEG-2 além de apresentar novas funcionalidades (FELITTI, 2007).

O MPEG-4 é um padrão ainda em desenvolvimento, está dividido em várias partes onde as principais são a parte 2 que inclui ASP, usado por *codecs* como DivX, Xvid, Nero Digital e 3ivx e pelo Quicktime 6 e a parte 10 mais conhecida como AVC/H.264 que será estudado neste trabalho de conclusão de curso, usado pelos *codecs* x264, Nero Digital AVC, QuickTime 7 e pelos formatos de DVD da nova geração como HD DVD e *Blu-ray*.

3.1.2.3 Avi

Formato criado pela Microsoft que encapsula áudio e vídeo num mesmo arquivo, por ser nativamente suportado pela maioria das versões do Windows tornou-se

[2] Broadcast: processo pelo qual se transmite ou difunde determinada informação para muitos receptores ao mesmo tempo.

um dos formatos mais populares, também é compatível com todos os leitores de DVD que utilizam o codec DivX.

Conforme Felitti (2007) em seu funcionamento um arquivo avi assume a forma de um único bloco, que é dividido obrigatoriamente em outros dois blocos e opcionalmente em mais um bloco. O primeiro sub-bloco obrigatório é o cabeçalho do arquivo, nele estão contidas informações sobre o vídeo, tais como sua largura, altura e taxa de quadros, o segundo sub-bloco obrigatório contém o áudio real e os dados visuais que compõe o vídeo e por último o sub-bloco opcional que indexa os deslocamentos dos blocos de dados dentro do arquivo.

3.1.2.4 Wmv

Windows Media Vídeo, é um forma de compressão de vídeo desenvolvida pela Microsoft que possui diversos *codecs* proprietários, foi projetado originalmente para concorrer com o RealVideo em aplicações de streaming na Internet, recentemente o formato WMV 9 foi aprovado para ser utilizado em mídias de alta densidade como o HD DVD e o *Blu-ray*.

Na grande maioria dos casos o wmv é encapsulado com o formato *Advanced Systems Format* (ASF) usando o *Windows Media Video* para codificar o vídeo e *codec Windows Media Audio* para o áudio. Pelo uso do ASF, torna-se possível a gestão de direitos autorais através de uma combinação de criptografia de curva elíptica com a troca de chaves DES. Seus principais concorrentes são o MPEG-4 AVC , AVS , RealVideo , DivX e Xvid (LIMA, 2010).

3.1.3 TV Digital no Brasil

A primeira transmissão oficial de sinal de TV digital no Brasil ocorreu em dezembro de 2007 e a partir de maio de 2008, teve início uma campanha de popularização da televisão digital brasileira, que incluía demonstrações em pontos de grande circulação. Antes da primeira transmissão houve muita polêmica em torno da tecnologia de transmissão que seria adotada: européia, americana ou japonesa. Em junho de 2006, foi adotado o padrão japonês para a TV digital brasileira, o que quer dizer que a TV digital no país é compatível com a tecnologia atualmente utilizada no Japão (CRUZ, 2008).

O padrão adotado no Brasil é o ISDB-TB, uma adaptação do padrão japonês *Integrated Services Digital Broadcasting Terrestrial* (ISDB-T), acrescida de tecnologias desenvolvidas nas pesquisas das universidades brasileiras. O padrão japonês foi escolhido, por que atender melhor as necessidades de energia nos receptores, mobilidade e portabilidade sem custo para o consumidor, diferente do padrão europeu (DVB-T), onde esta operação é tarifada pelas empresas telefônicas. A principal diferença que houve após a adoção do padrão japonês, foi a substituição do formato de compressão MPEG-2 para o MPEG-4 (CROCOMO, 2007).

Segundo Torres (2005) além da transmissão em alta definição, o padrão ISDB-TB permite a transmissão de multiprogramações, onde ao invés de transmitir um único programa em alta definição é possível transmitir oito em uma definição padrão (720 x 480 pixels, a mesma do DVD). Para uma breve comparação, a televisão analógica pelo fato de ter perdas na transmissão pelo ar, chega a uma resolução máxima de 333 x 480. Com o codec H.264 do formato MPEG-4 que será estudado neste projeto,

é possível transmitir até 2 canais HD (1080i), 4 Canais HD (720p) e/ou 8 SD (480p) pela mesma transmissora. A Figura 1 abaixo mostra a comparação entre a transmissão de um sinal analógico e um sinal digital, onde temos à direita a transmissão feita pelo sinal digital e à esquerda pelo sinal analógico:

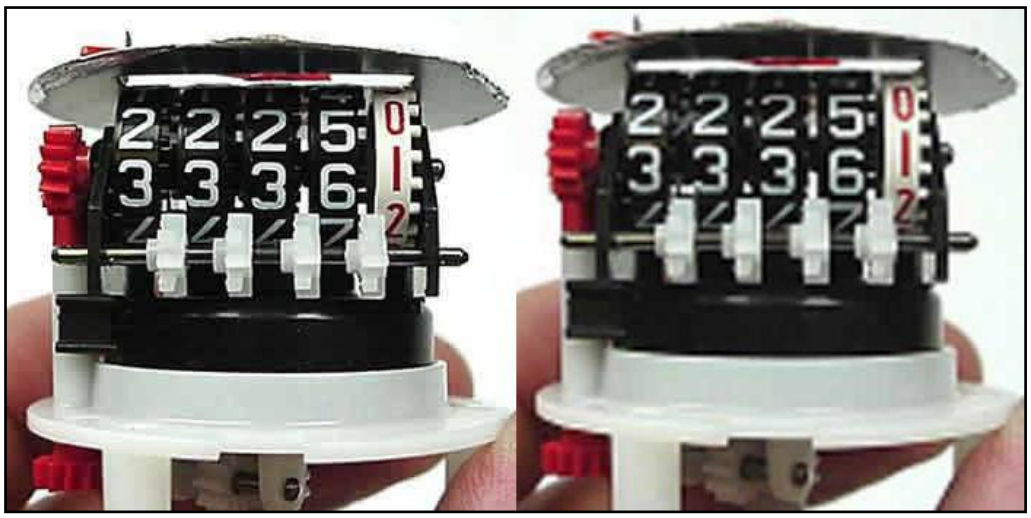


Figura 3. Comparação entre sinais analógico e digital.
Fonte: BRAIN, M. (2006).

Por padrão a TV digital pode transmitir a vários formatos diferentes, demonstrados na Tabela 1.

Tabela 1 Formatos de transmissão da TV digital

Formato	**Detalhes**
480i	A imagem possui 704 x 480 pixels e é enviada a 60 quadros entrelaçados por segundo (30 quadros completos por segundo).
480p	A imagem possui 704 x 480 pixels e é enviada a 60 quadros completos por segundo.

720p	A imagem possui 1280 x 720 pixels e é enviada a 60 quadros completos por segundo.
1080i	A imagem possui 1920 x 1080 pixels e é enviada a 60 quadros entrelaçados por segundo (30 quadros completos por segundo).
1080p	A imagem possui 1920 x 1080 pixels e é enviada a 60 quadros completos por segundo.

Fonte: TORRES, G. (2005).

A designação "p" no final do formato significa "progressivo" e o “i” significa "entrelaçado" do inglês, *interlaced*. No formato progressivo, há a cada sexagésimo de segundo uma atualização completa da imagem, no formato entrelaçado, metade da imagem se atualiza a cada sexagésimo de segundo (TORRES, 2005).

Conforme Cruz (2008) os formatos 480 são chamados de formatos *standard definition*, definição padrão (SD), o 480i equivale-se a uma imagem de TV analógica normal. Quando os programas de TV analógica são convertidos e transmitidos em estações de TV digital, são irradiados em 480p ou 480i. Os formatos 720p, 1080i e 1080p são os formatos HD, ou seja, de alta definição, quando fala-se em "HDTV", está se falando em um sinal digital no formato 720p, 1080i ou 1080p (BRAIN, 2006).

Conforme Brain (2006) outra grande característica do sinal digital esta na proporção de aspecto, os formatos de alta definição da TV digital possuem uma proporção de aspecto diferente da apresentada nas TVs analógicas. Uma TV analógica possui uma proporção de aspecto de 4:3 (4 unidades de largura e 3 unidades de altura), neste caso uma TV analógica com "diagonal de 25 polegadas" possui 38 cm de altura e

51 cm de largura, já o formato de alta definição da TV digital tem uma proporção de aspecto de 16:9, conforme mostrado na Figura 2.

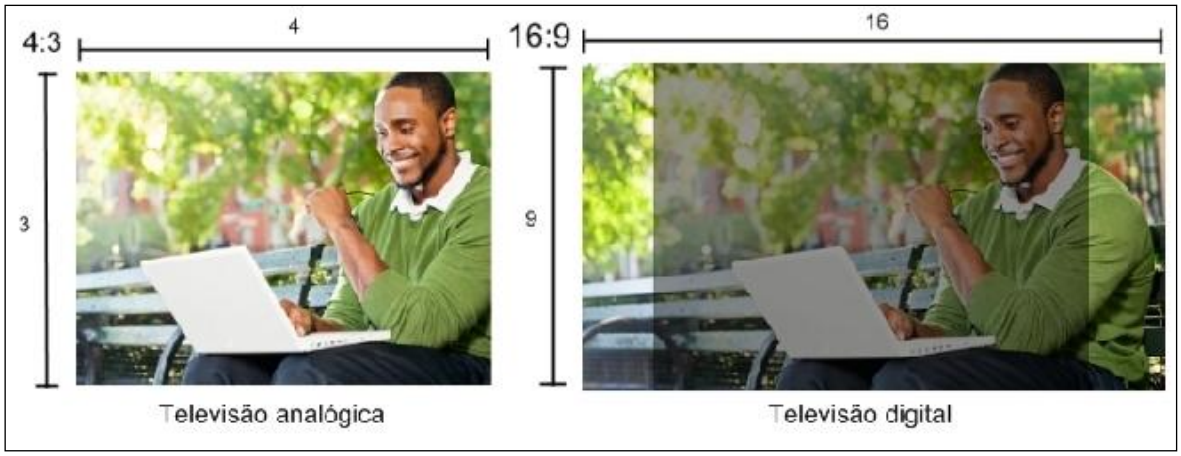


Figura 4. Comparação entre TV analógica e digital.
Fonte: BRAIN, M. (2006).

3.2 MODELO TEMPORAL

O objetivo do modelo temporal é explorar a redundância entre os quadros prevendo um quadro através de outro quadro anteriormente codificado (quadro de referência). Encontra-se a diferença entre ambos os quadros, resultando em um resíduo. O resíduo é codificado e enviado ao decodificador, que então adiciona o resíduo do quadro de referência. Na maior parte dos casos utiliza-se apenas o quadro anterior como referência, mas também pode-se usar um ou mais quadros passados ou futuros para se prever os quadros seguintes. Todo este processo de exploração da redundância existente entre os quadros chama-se predição Inter-quadros.

3.2.1 Predição *Inter*

Na predição *Inter*, os quadros são codificados um de cada vez e a exploração da redundância é feita pela diferença do quadro atual com uma versão predita do quadro atual. A diferença encontrada é chamada de resíduo ou erro de predição. A predição do quadro atual consiste em buscar em outros quadros anteriormente codificados e reconstruídos (decodificados localmente), informações que permitam reduzir a energia do resíduo. Os quadros usados para a geração do quadro predito são denominados quadros de referência. Do lado do decodificador, o quadro predito é recriado e somado ao resíduo gerando o quadro atual decodificado (RICHARDSON, 2003).

Segundo Wiegand (2003) a principal maneira de se fazer a predição é utilizando o quadro imediatamente anterior como predito do quadro atual e obtendo o resíduo através da diferença entre esses dois quadros. A Figura 5 mostra os dois primeiros quadros da sequência “Foreman” e a diferença gerada pelos dois. No entanto, mesmo os dois quadros pertencendo à mesma cena do vídeo (sem nenhuma mudança bruta de cena), o resíduo gerado apresenta uma quantidade de energia elevada nas regiões de movimento, seja movimento de câmera ou de objetos. Uma boa forma de predição pode ser feita apenas compensando-se o movimento existente na cena. O quadro predito não precisa obrigatoriamente ser um dos quadros anteriores por inteiro, ele pode ser uma composição dos mesmos. Esta composição é chamada justamente de Compensação de Movimento (MC). Porém, é necessário que se determine os movimentos no quadro atual e isto é feito pela Estimação de Movimento (ME).

O módulo de ME, foi implantado apenas no codificador do padrão H.264/AVC, que junto com a MC, formam a predição inter-quadros. É na predição *inter* onde se encontra a maior complexidade computacional de um codificador H.264/AVC (PURI, 2004). Este custo computacional é muito elevado devido à várias ferramentas novas e inovadoras que foram implantadas nesta etapa de codificação. Devido a estas diversas ferramentas, é neste módulo onde se encontra o maior potencial de ganhos do H.264/AVC, em termos de compressão, em relação aos padrões anteriores de compressão de vídeos (WIEGAND, 2003).

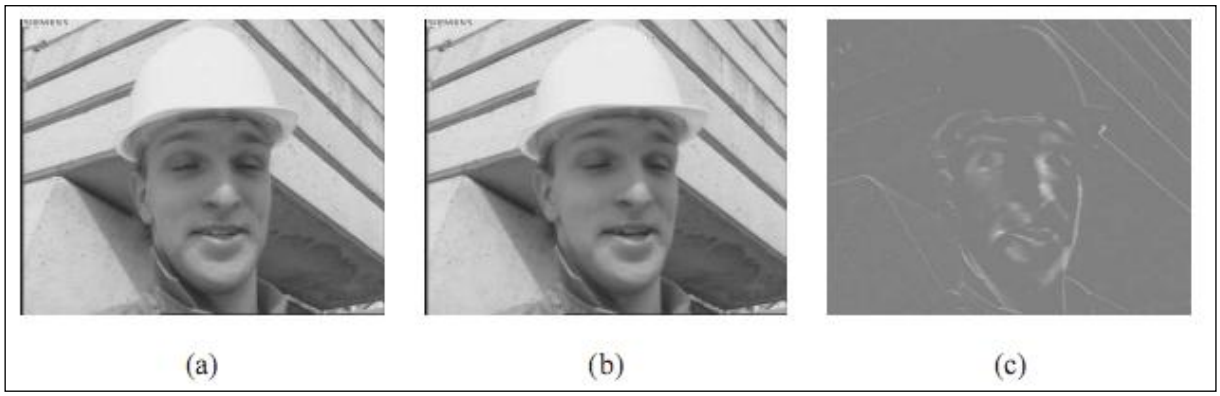


Figura 5. Sequência Foreman: (a) Quadro 1; (b) Quadro 2; (c) Diferença entre Quadros 1 e 2.
Fonte: RIBEIRO, N; TORRES, J. (2009).

3.2.2 Predição de Quadros Predito (P) e Bipredito (B)

No padrão H.264 a predição *inter* pode ser feita de duas maneiras: predição de quadros do tipo P e do tipo B . Na predição do tipo P, os quadros são preditos baseando-se apenas em quadros passados, sejam eles do tipo I ou até mesmo do tipo P e também são utilizados para a predição de outros quadros futuros do tipo P ou B. Já a predição do tipo B, é feita utilizando-se quadros I e P passados e ainda quadros futuros, oferecendo uma maior compressão. Neste tipo de predição existem duas listas de

quadros de referência que podem conter tanto quadros passados quanto quadros futuros. A primeira lista chama-se Lista 0, é formada por quadros passados e depois por quadros futuros. A segunda chama-se Lista 1, que é formada por quadros futuros e depois por quadros passados. O macrobloco será escolhido entre os quadros da Lista 0 e 1 ou ambas. No caso de ambas, o macrobloco predito resultante será a média de cada um dos blocos escolhidos de cada lista (MEGRICH, 2009).

As Figuras 6(a) e 6(b) mostram exemplos com quadros com predições dos tipos P e B, respectivamente, enquanto a Figura 6 (c) mostra a sequência de codificação. As setas indicam de qual quadro é feita a predição. Existe a predição do tipo I (Intra-quadro), onde o macrobloco predito provém do próprio quadro atual, não existindo estimação e compensação de movimento.

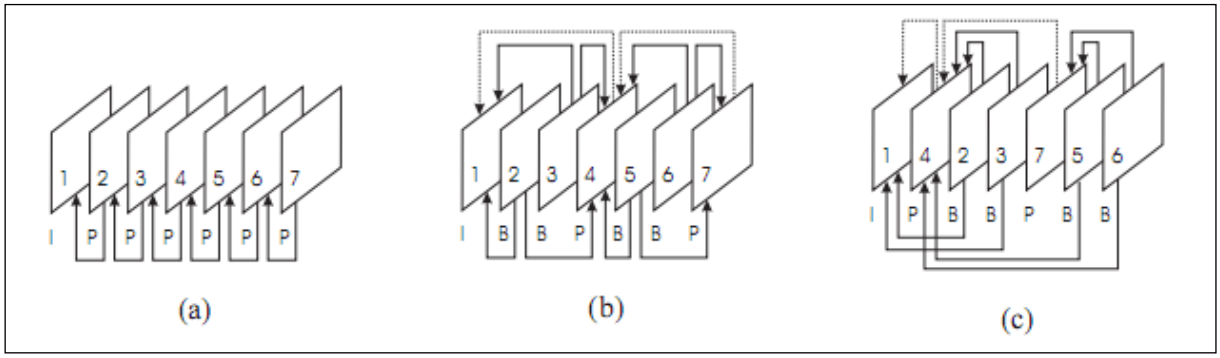


Figura 6. Tipos de Predição: (a) Tipo P; (b) Tipos P e B - Ordem de apresentação; (c) Tipos P e B – Ordem de codificação.
Fonte: MEGRICH, A.(2009).

Em um mesmo quadro pode haver macroblocos preditos pelos diferentes tipos apresentados. Isto acontece porque o quadro pode ser separado em *slices* (fatias), os *slices* são agrupamentos de macroblocos em ordem *raster* (da esquerda para a direita, de cima para baixo), e não necessariamente contíguos. Os *slices* classificam-se em I, P e

B. Entretanto, as *slices* dos tipos P e B também podem ter macroblocos do tipo I. o que aumenta a flexibilidade da predição.

O padrão H.264 inovou com o uso de uma resolução de um quarto de *pixel* para os vetores de movimento, ao contrário dos padrões anteriores que usam meio *pixel*. Apesar de isto já estar presente no padrão MPEG-4 Visual (ou MPEG-4 PART 2), a complexidade do processo de interpolação teve uma redução considerável (MEGRICH, 2009).

3.2.3 Estimação de Movimento (*Motion Estimation* - ME)

O objetivo da ME é encontrar, nos quadros de referência, qual o bloco se assemelha mais com o bloco do quadro atual. Esta busca é realizada em uma determinada área ao redor do bloco que esteja em análise, esta área denomina-se área de pesquisa. Assim que o melhor casamento é encontrado, a ME gera um vetor de movimento (MV – *Motion Vector*) que indica qual é o deslocamento da posição bloco do quadro atual para o bloco do quadro de referência. O valor do MV é enviado junto à codificação do macrobloco no *bitstream*. A Figura 7 demonstra o processo de ME (PORTO, 2008).

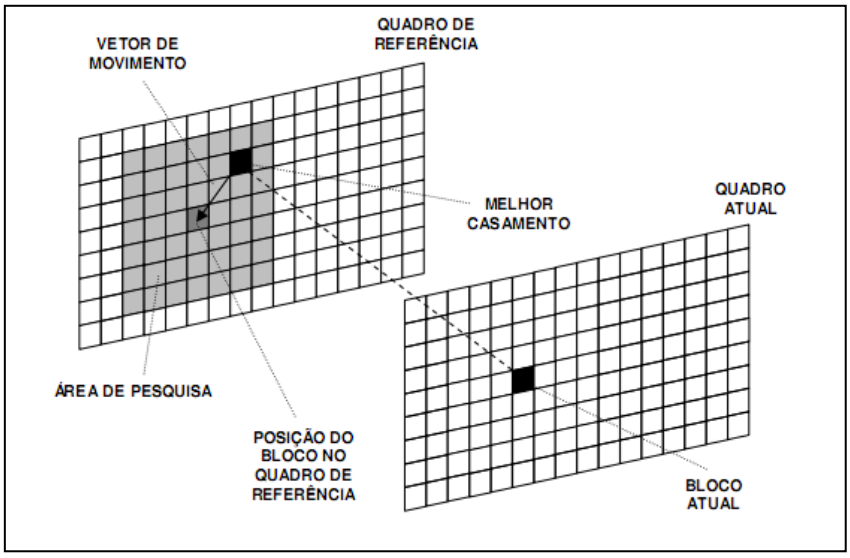


Figura 7. Determinação do vetor de movimento de um bloco na ME.
Fonte: PORTO, R. (2008).

Geralmente os deslocamentos ocorrem pela movimentação de algum objeto ou pela movimentação da câmera em relação à cena observada. Desta maneira, no caso da Figura 5, de dois quadros subsequentes, determinada região de um quadro, se deslocou para outra região em outro quadro.

No entanto, há a possibilidade de que a percepção de um objeto, por exemplo, nos dois quadros seja diferente, seja por mudança na iluminação do objeto, deslocamento do próprio objeto, deslocamento da câmera ou até mesmo deslocamento de um outro objeto sobre o outro. Usando o primeiro quadro como referência, pode-se fazer uma busca do objeto presente no segundo quadro e determinar o quanto ele se deslocou. Essa diferença entre as posições do objeto nos dois quadros é o MV, e o conjunto dos MVs de um quadro é chamado de fluxo óptico (RIBEIRO; TORRES, 2009).

Durante o desenvolvimento de padrões de codificação, pode-se observar que a ME de objetos inteiros pode em alguns casos, ser bastante complexa, principalmente porque pode ser bastante difícil determinar qual parte do quadro pertence a um objeto

qualquer. Outra ideia seria fazer a busca de cada *pixel* individualmente, o que permitiria um MV exato para cada *pixel*, porém o custo computacional de fazer buscas para todos os *pixels* é extremamente alto. Decidiu-se então por fazer a busca para blocos de *pixels*. Esses blocos são de tamanho 16x16 *pixels* e são chamados de macroblocos (PEIXOTO, 2008).

A ME é feita então bloco a bloco, buscando nos quadros de referência o bloco mais parecido possível com o bloco atual. Para encontrar o bloco de referência que tem uma melhor relação com o atual, usa-se técnicas baseadas na diferença entre os blocos. As técnicas mais comuns são a Soma das Diferenças Absolutas (SAD - *Sum of Absolute Differences*) e a Soma da Diferença Quadrática (SSD - *Sum of Squared Differences*) (MEGRICH, 2009).

Segundo Richardson (2003) a estimação de movimento é aplicada apenas ao componente de luminância do macrobloco. Pelo fato de os componentes de crominância possuírem metade da resolução horizontal e vertical da componente de luminância, basta que seja feito uma divisão por dois nos componentes horizontal e vertical de cada vetor de movimento de luminância para serem aplicadas aos blocos de crominância.

3.2.4 Compensação de Movimento (MC – *Motion Compensation*)

A MC encontra-se no codificador e no decodificador do padrão H.264/AVC, este módulo faz a operação inversa da ME e deve ser compatível com as operações pela mesma. Com as informações do MVs e dos índices dos quadros de referência, a MC encontra o bloco que foi escolhido no quadro de referência pela ME e reconstrói o bloco do quadro atual.

No codificador a MC pode não ser tão complexa, adaptando-se às possíveis simplificações feitas na ME. No entanto, no decodificador ele deve ser completa, sendo totalmente compatível com todas as ferramentas previstas em um perfil do padrão H.264/AVC, sendo possível ser feita a decodificação do vídeo gerado pelos diversos codificadores compatíveis com o mesmo perfil (AZEVEDO, 2006).

3.3 MODELO ESPACIAL

O modelo espacial ou predição intra-quadro, explora a redundância espacial dentro de um quadro da sequência de vídeo, descorrelacionando o resíduo para convertê-lo em uma forma que possa ser ainda mais comprimido por um codificador de entropia. Os modelos espaciais geralmente possuem três componentes principais: a transformada, que descorrelaciona as amostras; a quantização, que diminui a precisão dos dados transformados permitindo diminuir o número de bits para compressão; e a reordenação, que rearranja os dados em grupos de valores significativos (RICHARDSON, 2003).

3.3.1 Predição *Intra*

A predição intra-quadro é aplicada a cada macrobloco do tipo *Intra*, tanto para as amostras de luminância (Y) quanto para as amostras de crominância (Cb e Cr). A predição de um macrobloco é gerada a partir de amostras já reconstruídas (porém não filtradas) localizadas nas fronteiras deste macrobloco e contidas dentro de um mesmo *slice* do quadro. Entende-se por amostras reconstruídas as amostras que já foram

processadas pelo caminho de reconstrução T/Q/TI/QI (Transformadas e Quantização Direta e Inversa), mas que não foram processadas pelo filtro de deblocagem (RICHARDSON, 2003).

Dois tipos de macrobloco I (*Intra*) são definidos no H.264/AVC: Intra 4x4 e Intra 16x16. No primeiro caso, as 16x16 amostras de luminância são divididas em blocos de 4x4 amostras para predição. Os blocos 4x4 são numerados de acordo com sua ordem de processamento, que é a mesma ordem de processamento dos blocos de transformadas e quantização. No segundo caso, o macrobloco é predito por inteiro, sem subdivisão. Somente um tipo de predição é definido para ambos os blocos 8x8 de crominância (Cb e Cr), quando a sub-amostragem 4:2:0 é usada, não havendo subdivisão dos blocos para predição (ITU-T, 2005).

Cada tipo de predição (Intra 4x4 ou Intra 16x16) contém diferentes modos para gerar o macrobloco predito. Estes modos procuram reproduzir diferentes padrões espaciais contidos em um quadro do vídeo. Um macrobloco predito muito similar ao macrobloco original pode reduzir significantemente o valor dos resíduos a serem codificados e, por consequência, o número de bits para codificar o macrobloco em questão. O diferente particionamento (Intra 4x4 ou Intra 16x16) determina a granularidade com que é feita a predição. O modo Intra 4x4 possui granularidade mais fina que o tipo Intra 16x16, porém gera mais bits de informação dos modos de predição a serem enviados para o decodificador, como será detalhado nas próximas seções.

O tipo Intra 4x4 define 9 modos de predição que abrangem padrões horizontais, verticais e diagonais com diferentes ângulos de inclinação. Esta predição é muito adequada para regiões da imagem com maior nível de detalhe. A Figura 8 ilustra a direção dos 9 modos (ITU-T, 2005).

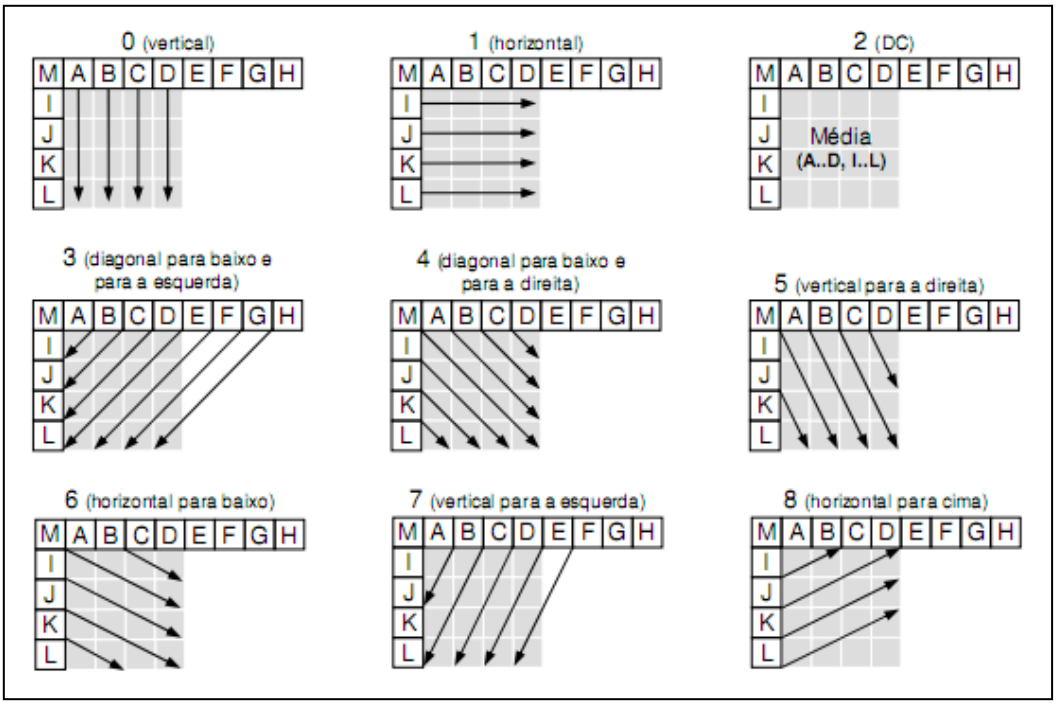


Figura 8. Modos de predição Intra 4x4.
Fonte: AGOSTINI, V. (2007).

Nos modos 0 (vertical) e 1 (horizontal) o bloco predito é construído pela cópia das amostras vizinhas A-D e I-L (Figura 8), respectivamente. O modo 2 (DC) faz uma média das amostras A-D e I-L e copia o resultado para todas amostras do bloco predito. Os modos diagonais (3 a 8 na Figura 8) são calculados por interpolações lineares usando dos valores das amostras vizinhas, dependendo das direções das setas e da posição da amostra no bloco predito (HUANG, 2005).

A predição Intra 16x16 gera um macrobloco predito de tamanho 16x16 amostras e luminância, a partir das 32 amostras de luminância vizinhas contidas na borda dos macroblocos acima e à esquerda do macrobloco a ser codificado (H e V na Figura 9). São definidos 4 modos de predição, como ilustrado na Figura 9.

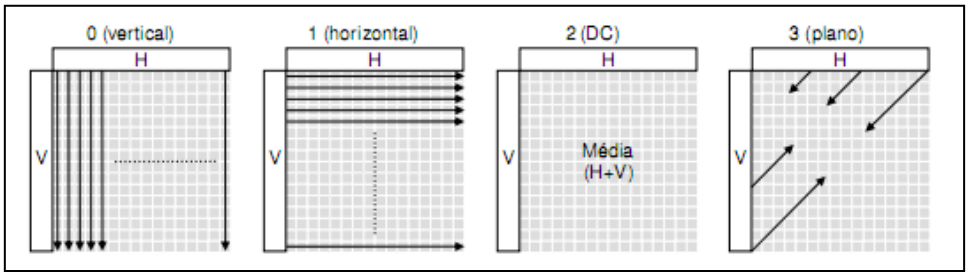


Figura 9. Modos de predição Intra 16x16.
Fonte: AGOSTINI, V. (2007).

No modo 0 (vertical) o macrobloco predito é construído pela cópia das amostras vizinhas contidas em H. O modo 1 (horizontal) é similar ao modo 0, mas copia as amostras em V. O modo 2 (DC) faz uma média das amostras contidas em H e V e copia o resultado para todas amostras do macrobloco predito.

O modo 3 (plano) é o mais complexo de ser calculado, pois necessita que sejam previamente gerados três parâmetros antes do cálculo da predição para cada amostra do macrobloco predito (AGOSTINI, 2007).

Os modos de predição para amostras de crominância são semelhantes aos modos Intra 16x16, porém o número dos modos é alterado, como mostrado na Figura 10. No caso de sub-amostragem 4:2:0, usado neste trabalho, a predição é aplicada a blocos 8x8 de crominância azul (Cb) e vermelha (Cr), a partir de 16 amostras vizinhas (H e V na Figura 10). O mesmo modo é aplicado para ambos os canais de crominância Cb e Cr.

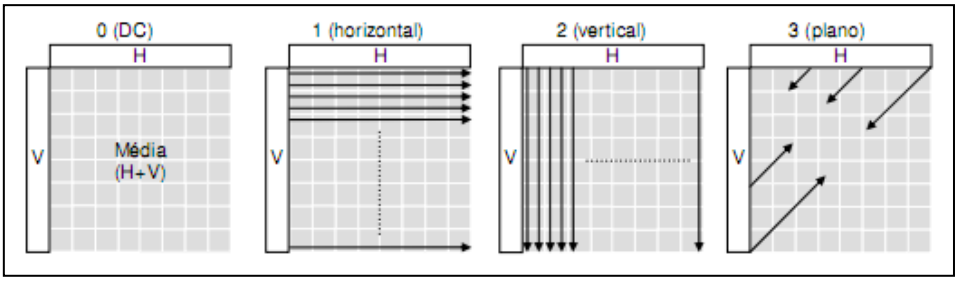


Figura 10. Modos de predição em blocos 8x8 de crominância.
Fonte: AGOSTINI, L. (2007).

Segundo Agostini (2007) os modos DC e planos para crominância são similares aos modos de predição intra 16x16 mas calculados sobre as 16 amostras vizinhas (8 em H, 8 em V), assim, diferenciando também os valores de arredondamento e deslocamento.

3.3.2 Transformada

O processo de transformação que converte os resíduos presentes no domínio espacial para outro domínio de frequência. Pelo fato de que as amostras no domínio espacial variam muito ao longo do tempo, elas não são comprimidas muito bem pelas técnicas de entropia. Já quando as amostras são convertidas para o domínio da frequência, elas se tornam muito mais adequadas para compressão por entropia.

A técnica para transformada mais utilizada em codificação de vídeo é a *Discrete Cosine Transform* (DCT) (ITU-T, 2005). A transformada dos blocos ocorre em todos os componentes, da esquerda para a direita, de cima para baixo, em um esquema chamado de ordenamento não-entrelaçado. Os blocos são compostos de 64 valores que representam a amplitude do sinal amostrado que é função de duas coordenadas espaciais, ou seja, $a = f(x,y)$ onde x e y são as duas dimensões. Após a transformação, obtém-se a função $c = g(Fx,Fy)$ onde c é um coeficiente e Fx e Fy são as frequências espaciais para cada direção. O resultado é outro bloco de 64 valores onde cada valor representa um coeficiente DCT.

O coeficiente $g(0,0)$ correspondente às frequências zero é chamado de coeficiente DC. Ele representa o valor médio das 64 amostras. Como em um bloco representando uma porção da imagem os valores amostrados geralmente variam pouco

de um ponto para outro, os coeficientes de mais baixa frequência serão altos e os de média e alta frequência terão valores baixos ou zero, podendo ser descartados. A energia do sinal é concentrada nas frequências espaciais mais baixas. A Figura 11 é uma representação tridimensional da transformação DCT (WIEGAND, 2003).

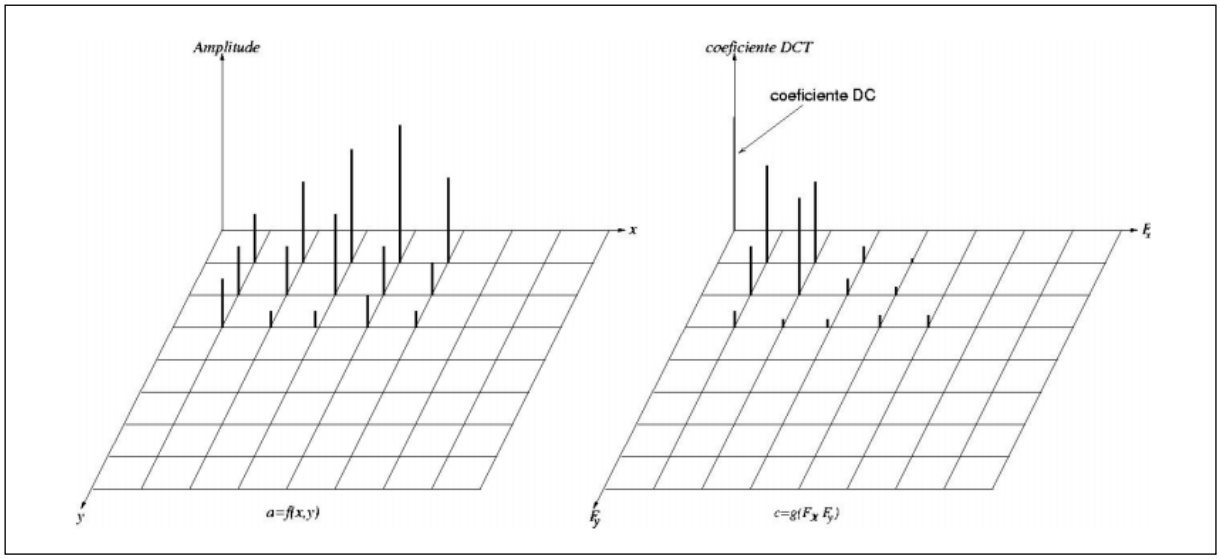


Figura 11. Representação tridimensional da DCT. Antes (esquerda) e depois (direita) da transformação.
Fonte: WIEGAND, T. (2003).

Em uma imagem, quando há uma mudança brusca ocorrerão os coeficientes de média e baixa frequência. Em uma imagem da natureza, por outro lado, as transições entre as zonas da imagem são suaves. A transformada do H.264/AVC introduz duas grandes novidades: a transformada usando apenas aritmética inteira e o operando sobre blocos menores.

3.3.3 Quantização

No processo de quantização, mapeia-se um sinal com um intervalo de valores *X* para um sinal quantizado, com um intervalo reduzido de valores *Xq*. Assim,

pode-se representar um sinal quantizado com uma quantidade menor de bits. A quantização escalar mapeia um valor do sinal de entrada em um valor quantizado. Já a quantização vetorial mapeia um grupo de amostras do sinal de entrada (vetor) em um grupo de valores quantizados (RICHARDSON, 2003).

A maneira mais simples de realizar a quantização escalar de valores reais é por arredondamento, ou seja, um mapeamento de *R* para *Z*. neste processo é impossível recuperar o valor original a partir do valor quantizado, tornando-se um processo com perdas. Outro exemplo mais geral de quantização é a chamada quantização uniforme, representado por na fórmula da figura 12, onde *Qp* é o passo de quantização, *X* é o valor a ser quantizado, *Xq* é o valor quantizado, $\hat{X}$ é o valor recuperado.

$$X_q = round\left(\frac{X}{Q_p}\right)$$
$$\hat{X} = X_q.Q_p$$

Figura 12. Quantização uniforme.
Fonte: RICHARDSON, I. (2003).

Este processo é aplicado após o processo de transformada, onde se eliminam valores insignificantes próximos a zero, mantendo apenas um número reduzido de valor significativo. Do processo de quantização dos coeficientes transformados gera-se uma matriz esparsa, composta em grande parte por entradas iguais a zero.

Abaixo temos um exemplo, onde é usado um bloco de tamanho 8x8 extraído do resíduo de uma sequência de frames e a tabela de quantização do padrão JPEG mostrada na Tabela 3 abaixo. Os valores dos pixels deste bloco são apresentado na

Tabela 2 também abaixo. Os valores dos blocos transformados pela DCT e quantizados são mostrados nas Tabelas 4 e 5, respectivamente.

Tabela 2. Bloco 8x8 de resíduos.

-2	-2	-1	6	18	12	-1	-2
-1	0	-1	-1	1	6	5	-1
0	-1	0	2	-7	-6	1	-3
-1	1	-2	-3	-1	2	3	2
-1	1	2	-1	2	1	-2	-1
0	2	1	0	0	0	1	-1
-1	1	2	1	2	2	1	1
1	1	1	1	-1	-3	-2	-2

Fonte: SAYOOD, K. (2000).

Tabela 3. Tabela de quantização do JPEG.

16	11	10	16	24	40	51	61
12	12	14	19	26	58	60	55
14	13	16	24	40	57	69	56
14	17	22	29	51	87	80	62
18	22	37	56	68	109	103	77
24	35	55	64	81	104	113	92
49	64	78	87	103	121	120	101
72	92	95	98	112	100	103	99

Fonte: SAYOOD, K. (2000).

Tabela 4. Bloco transformado.

57,8750	34,0422	-30,8251	-1,8822	3,6250	-8,4951	-5,4972	4,2405
-38,2364	1,1645	-41,5549	22,1580	-27,7356	8,9246	-7,5870	-1,4612
-9,2204	28,3195	-41,8711	33,5663	-34,4742	6,5632	-0,8436	-5,8853
-1,8892	41,9590	-7,5757	25,4611	-1,9353	-2,6480	0,7227	-3,4208
-31,8750	52,5784	1,1200	-11,4830	24,6250	-7,0802	2,3773	2,3093
-4,9106	6,4469	2,1081	-9,7191	4,1768	4,6382	2,3321	-1,9163
-0,3751	-9,9092	-5,3436	7,4830	-7,7741	-1,0881	4,6211	-2,4698
11,2908	-8,5322	-8,7200	6,9507	-5,5660	-4,8681	7,8368	-0,7637

Fonte: SAYOOD, K. (2000).

Tabela 5. Bloco transformado e quantizado.

-4	3	-3	0	0	0	0	0
-3	0	-3	1	-1	0	0	0
-1	2	-3	1	-1	0	0	0
0	2	0	1	0	0	0	0
-2	2	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0

Fonte: SAYOOD, K. (2000).

3.3.4 Reordenamento

Os coeficientes transformados e quantizados devem ser codificados da forma mais agrupada possível. Para isso, deve ser feito o reordenamento desses coeficientes para que os valores significativos se agrupem dando uma representação mais eficiente dos coeficientes de valor zero. Na região de baixa frequência é onde se concentram os coeficientes mais significativos da DCT. O reordenamento se dá pela varredura em zig-zag, ou zig-zag scan, mostrado na Figura 13 (SAYOOD, 2000).

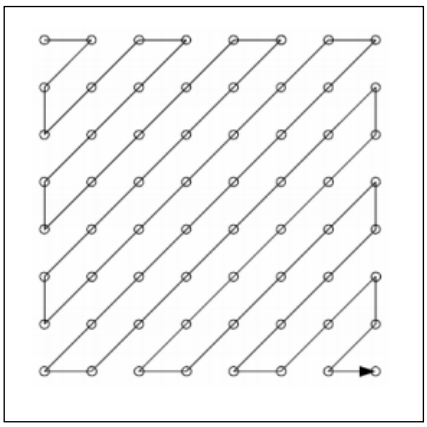


Figura 13. Varredura em zig-zag.
Fonte: SAYOOD, K. (2000).

Como resultado tem-se um vetor composto de alguns coeficientes não-zeros e vários coeficientes de valor zero espalhados. Para diminuir essa informação usa-se a técnica de run-zero. Esta técnica representa o vetor por uma série de pares (*run, level*), em que *run* indica a quantidade de zeros precedendo um coeficiente não-zero e *level* indica a magnitude do coeficiente não-zero. Ao final da sequência, é apresentado um código especial separado que determina o último dos coeficientes não-zeros. Para exemplo, foi feita a varredura em zig-zag no bloco apresentado na Tabela 5 e o agrupamento em pares (*run, level*) onde tem-se os resultados a seguir:

−4, 3,−3,−1, 0,−3, 0,−3, 2, 0,−2, 2,−3, 1, 0, 0,−1, 1, 0, 2, 0, 0, 0, 0, 1,−1, 0 . . .

(0,−4), (0, 3), (0,−3), (0,−1), (1,−3), (1,−3), (1, 2), (1,−2), (0, 2), (0,−3), (0, 1), (2,−1)
(0, 1), (1, 2), (4, 1), (0,−1), (38, 0)

onde o valor de *level* = 0 indica o fim do bloco. Nota-se claramente a redução da quantidade de valores a serem codificados de 64 para 34 (17 pares (*run,level*)).

3.4 PERFIS E NÍVEIS DO PADRÃO H.264/AVC

Os perfis do H.264 especificam quais ferramentas de codificação e algoritmos podem ser usados para transformar um vídeo bruto em um *bitstream*. Todos os decodificadores de um determinado perfil, devem por obrigação suportar todas as

características deste perfil, lhe permitindo transformar o *bitstream* em um vídeo bruto novamente. Os codificadores não precisão utilizar todas as características especificadas pelo perfil, mas devem sempre comprimir gerando um *bitstream* compatível, de modo que um decodificador no padrão H.264 tenha condições de decodificar o *bitstream* (ITU-T, 2003).

Primeiramente o H.264 possuía três perfis (*baseline, main* e *extended*). Mais adiante foram especificados outros 4 perfis, usados principalmente para codificação de alta qualidade (high, high 10, high 4:2:2, high 4:4:4) (SULLIVAN, 2004).

a) ***Baseline*:** Utilizado por aplicações que exigem baixo custo computacional e que não necessitam de alta qualidade.

b) ***Main*:** Busca igualar qualidade e custo computacional, sendo utilizado tanto para armazenagem quanto para transmissões. Atualmente é substituído pelo *High*, que possui melhores resultados;

c) ***Extended*:** Possui uma alta taxa de compressão, mais utilizado para aplicações de streaming (vídeos na Internet);

d) ***High*:** Usado para vídeos em alta definição, é adotado principalmente pelo formato de disco *Blu-Ray* e também para a transmissão de HD TV;

e) ***High* 10:** Perfil baseado no *High*, mas com suporte a codificação de 10 bits por quadro;

f) ***High* 4:2:2:** Utilizado em aplicações profissionais, é baseado no *High* 10 e permite a o uso do formato 4:2:2 para chroma;

g) ***High* 4:4:4:** Desenvolvido para dar suporte a aplicações que exijam uma qualidade muito alta, podendo codificar sequências no formato 4:4:4.

Todos os estudos e testes realizados neste projeto serão feitos utilizando o perfil *baseline*, que suporta as seguintes ferramentas: Apenas s*lices* do tipo P e I, multiplos *frames* de referência, codificador de entropia CAVLC, *Flexible Macroblock Ordering* (FMO), *Arbitrary Slice Ordering* (ASO), *Redundant Slices* (RS), formato 4:2:0 para Chroma e codificação de 8 bits por quadro (ITU-T, 2009).

Os níveis são restrições que são dadas às ferramentas utilizadas pelo perfil definindo algumas características do codificador. Estas restrições estão apresentadas na Tabela 6:

Tabela 6. Níveis do H.264/AVC.

Nível	**Formato da imagem**	**Tamanho máximo do frame em macroblocos**	**Bit rate máximo de compressão**	**Número máximo de frames de referência**
1	QCIF	99	64 kbps	4
1b	QCIF	99	128 kbps	4
1.1	CIF ou QCIF	396	192 kbps	2 ou 9
1.2	CIF	396	384 kbps	6
1.3	CIF	396	768 kbps	6
2	CIF	396	2 Mbps	6
2.1	HHR (480i ou 576i)	792	4 Mbps	6
2.2	SD	1620	4 Mbps	5
3	SD	1620	10 Mbps	5

3.1	1280x720p	3600	14 Mbps	5
3.2	1280x720p	5120	20 Mbps	4
4	720p ou 1080i	8192	20 Mbps	4
4.1	(720p ou 1080i	8192	50 Mbps	4
5	2kx1k	22080	135 Mbps	5
5.1	2kx1k ou 4kx2	36864	240 Mbps	5

Fonte: ITU-T, (2009).

4 ESTRUTURAS DE SEGMENTAÇÃO EM CODIFICAÇÃO DE VÍDEO

Os macroblocos são partes de um *frame*, geralmente de tamanho 16 x 16 *pixels* que são agrupados em *slices* e codificados um a um por codificadores que trabalham com blocos como o H264. Um macrobloco possui três maneiras de ser predito:

a) referenciando o macrobloco atual com um macrobloco já codificado do quadro atual (predição *intra*);
b) referenciando o macrobloco com o macrobloco de um quadro anterior (predição *inter* do tipo P);
c). referenciando o macrobloco atual ao macrobloco de dois ou mais quadros, tanto para quadros passados como futuros (predição *inter* do tipo B).

4.1 SEGMENTAÇÃO NO PADRÃO H.264/AVC

O padrão H.264 trouxe consigo um avanço muito considerável que é a possibilidade de subdividir o macrobloco em blocos menores de pixels. Essa partição chamada de Compensação de Movimento em Estrutura de Árvore. Ela guarda certa semelhança com a estrutura *quadtree* (mostrada na Figura 14(a)). Nas áreas de processamento de imagens e vídeos e computação gráfica, a estrutura *quadtree* consiste na subdivisão de blocos de pixels em quatro quadrantes. Cada quadrante pode ser novamente dividido em quadrantes e assim por diante, daí o nome que se refere a uma

árvore de quadrantes. Esta estrutura pode ser usada para uma série de finalidades, incluindo compressão e decomposição de imagens para classificação de regiões (ITU-T, 2003).

A principal diferença entre a estrutura de *quadtree* e a estrutura usada no H.264 é que, apesar de no H.264 também consistir uma estrutura de árvore, os blocos não precisam ser divididos em quadrantes, podendo ser quebrados em pares de blocos retangulares, o que de certa forma permite maior flexibilidade. Desta forma, um macrobloco pode ser dividido em dois blocos de 16x8 pixels, dois blocos de 8x16 pixels ou quatro blocos de 8x8 pixels. Os blocos de 8x8 pixels podem ser divididos em sub-blocos de 8x4 pixels, 4x8 pixels ou 4x4 pixels, como pode ser visto na Figura 14(b).

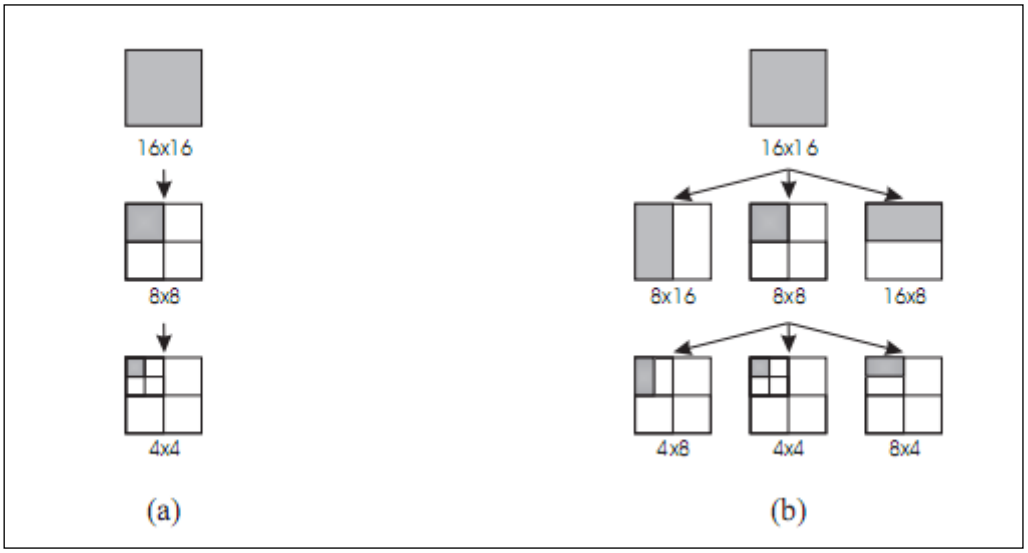


Figura 14. Estruturas em Árvore para Partição de Macrobloco: (a) Quad-Tree; (b) H.264. Fonte: STROBACH, (1991).

Quando o macrobloco é dividido, cada região originada pode ter um MV diferente, seguindo o mesmo princípio que levou o quadro a ser dividido em macroblocos, dentro de um macrobloco, várias regiões podem guardar algumas

semelhanças das partes do quadro de referência que não pertençam a um mesmo macrobloco. Apesar de serem necessários mais bits para esses MVs, diferentemente dos poucos usados para apenas um vetor no caso do macrobloco completo, o ganho que se obtém pela redução do resíduo de cada partição acaba por ser vantajoso.

A decisão se há ou não vantagem em dividir o macrobloco depende da quantidade de bits necessários para armazenar tanto os resíduos quanto os MVs. Somando-se esses bits aos demais resultantes da codificação, tem-se a taxa referente ao macrobloco. Outra avaliação para decidir dividir ou não está na distorção. O custo de um macrobloco é determinado em função dessas duas métricas.

Conforme especificado pela ITU-T (2003) para encontrar a melhor partição, o macrobloco é classificado em modos: 16x16, 16x8, 8x16, 8x8. O codificador verifica quanto será o custo para cada modo. Somente se o modo 8x8 for de menor custo, os blocos serão divididos e classificados em sub-modos: 8x4, 4x8, 4x4.

4.2 CODIFICANDO COM A QUADTREE

Sabe-se que uma imagem possui uma grande quantidade de informações redundantes, chamada de redundância espacial. Ao tratarmos com sequências de imagens, o índice de correlação entre um *frame* e outro se chama redundância temporal que é muito maior. Desta forma, os codificadores procuram retirar primeiramente esta redundância temporal.

O método mais simples utilizado para se explorar a correlação *inter-frames*, é trabalhando sobre a diferença entre dois *frames* consecutivos, o que gera a *frame*-diferença (FD, *Frame Difference*). Nesta diferença, apenas as áreas que apresentam

modificações de um *frame* para outro terão valores elevados. Utilizando técnicas de compensação de movimento para diminuir a energia desta imagem diferença, busca-se explorar características conhecidas dos movimentos a fim de fazer uma predição do *frame* seguinte. À esta imagem diferença dá-se o nome de frame-diferença com compensação de movimento (MCD, *Motion Compensated Difference*). A codificação da MCD é a técnica mais utilizada na compressão de vídeo atualmente (SALOMON, 2004).

Dependendo do tipo de sinal, como nas teleconferências ou vídeo áudio onde os pixels variam pouco de um *frame* para outro, o *frame* MCD possui grande parte de seus elementos nulos ou próximos de zero, isso acontece pelo fato de o frame MCD ser resultado da diferença de dois *frames* altamente correlacionados. Nas regiões com movimentação significativa serão apresentados valores elevados. Estas regiões estão concentradas principalmente nas bordas ou contornos dos objetos, por estes se diferenciarem do fundo fixo.

Como se pode notar, na Figura 15 temos a simples diferença entre dois *frames* consecutivos (FD) e a Figura 16 a diferença entre os *frames* com compensação de movimento, a energia da Figura 16 é bem menor que a energia da Figura 15. De modo que, devido à compensação de movimento, o erro causado pela quantização da Figura 16 resultará em um erro muito menor na imagem reconstruída caso quantizássemos a Figura 15.

Figura 15. Imagem da diferença entre dois frames (FD).
Fonte: SALOMON, (2004).

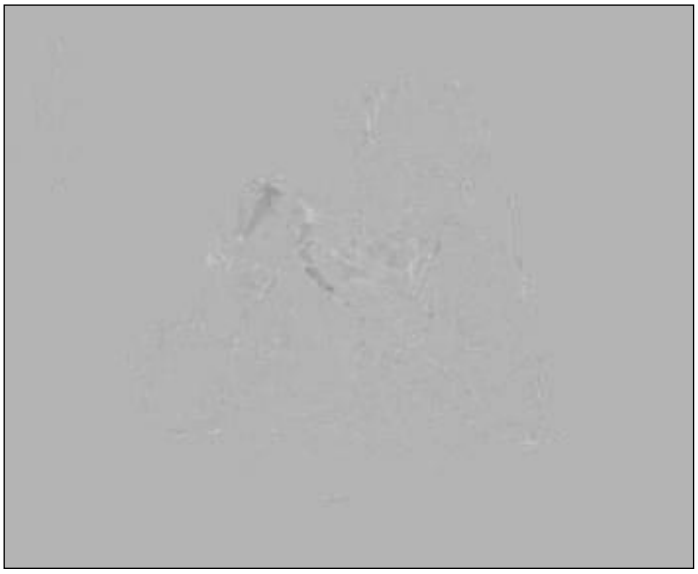


Figura 16. Imagem da diferença entre dois frames com compensação de movimento (MCD).
Fonte: SALOMON, (2004).

Segundo Strobach (1991) a decomposição *quadtree* apresenta um grande potencial para a representação de imagens em diferentes níveis de resolução, esta decomposição pode ser utilizada para reduzir o volume de dados das MCD a serem quantizadas. A decomposição QT utilizada na segmentação das MCD visa a reduzir os

dados a serem quantizados, mantendo-se a necessidade de se determinar os vetores de deslocamento do estimador de movimento para o quadro inteiro.

Com o uso da *quadtree* é possível que seja selecionado eficientemente áreas que sejam de interesse ao codificador. O uso da *quadtree* em codificação de imagens já é bem conhecido, geralmente esta codificação é realizada com quantização escalar da média dos pixels dos blocos que formam as folhas da árvore (seja folha ou não) (STROBACH, 1991).

Na área de codificação de vídeo vários trabalhos foram feitos baseando a construção da *quadtree* na MCD, de forma que a árvore seja um bom indicativo das regiões onde a compensação de movimento não foi eficiente e causou um erro grande que deve ser quantizado. Strobach (1991) propôs um método para otimizar conjuntamente a compensação de movimento e a complexidade da estrutura *quadtree* gerada a partir da MCD de modo que tenha um número mínimo de folhas a quantizar escalarmente. Lu e Pearlman (2003) utilizaram a *quadtree* afim de selecionar as áreas de interesse da MCD para posterior quantização com quantização vetorial com *codebook* estruturado em árvore, possibilitando desta forma a codificação com diferentes taxas. Muitos autores tem se preocupado com a otimização da construção da *quadtree* inserida no contexto da codificação de vídeo (BAKER e SULLIVAN, 1991).

4.3 SEGMENTAÇÃO QUADTREE

A *quadtree* é uma estrutura de dados hierárquica, possui ótima eficiência na descrição de regiões bidimensionais. Não é uma estrutura nova, primeiramente foi utilizada para armazenas figuras “binarizadas” (apenas pixels 0 e 1), depois por

codificadores de imagens com tons de cinza e atualmente codificadores de vídeo que a utilizam em conjunto com a *discrete cosine transform* (DCT) foram propostos (SALOMON, 2004).

Existem duas técnicas para se montar uma *quadtree*: a técnica chamada cima-para-baixo (*Up-bottom*) e sua inversa baixo-para-cima (*Bottom-up*). A *up-bottom* é baseada em um critério de divisão, parte-se de uma região maior e divide-se simetricamente em quatro regiões menores até um limite mínimo ser atingido, ver Figura 17, de modo que a estrutura resultante é completamente regular.

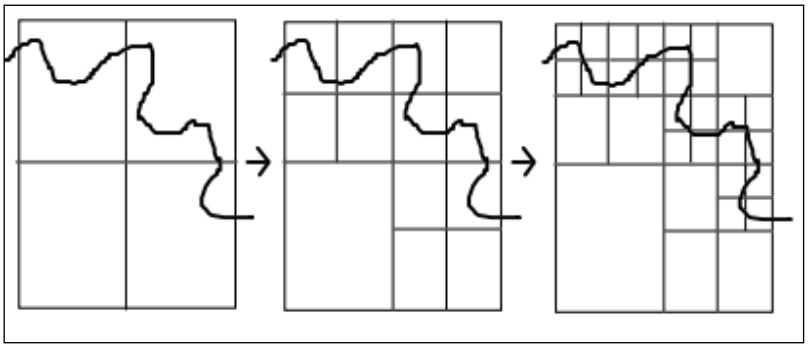


Figura 17. Método *up-bottom.*
Fonte: STROBACH, P. (1991).

A técnica *bottom-up* é baseada em um critério de agrupamento, assim, verifica-se se um determinado conjunto de quatro blocos vizinhos pode ser unido de modo a formar um único bloco, ver Figura 18. É mostrada na figura 19 a estrutura da árvore resultante.

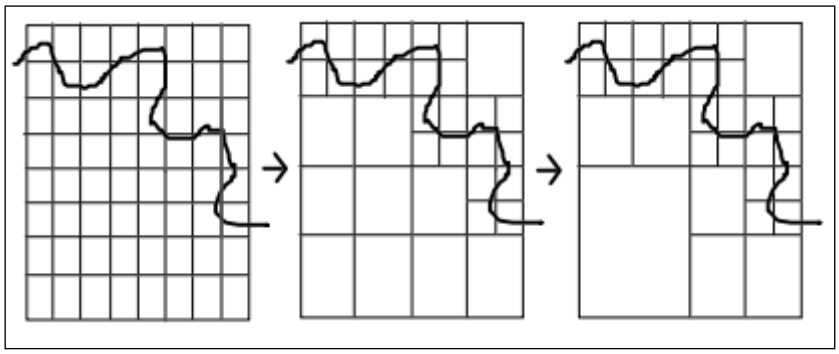


Figura 18. Método *bottom-up*.
Fonte: STROBACH, P. (1991).

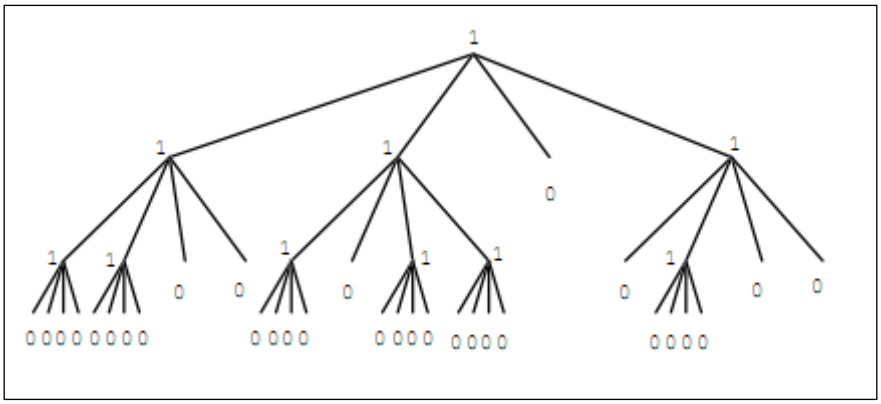


Figura 19. Estrutura da árvore resultante.
Fonte: STROBACH, (1991).

O critério de dividir blocos ou de juntar blocos menores em um bloco maior, é definido como sendo a "atividade" do bloco. Existem várias propostas para a medida da "atividade" de um bloco, a mais comuns é a variância dos pixels pertencentes ao bloco. Se a variância for pequena, quer dizer que os pixels são semelhantes, o que pode resultar na decisão de não dividir (*up-bottom*) o bloco ou de agrupar (*bottom-up*) os blocos caso as médias sejam semelhantes, de modo que um bloco que seja uniforme seja mantido o maior possível (MACLEAN e JERNIGAN, 1991).

As medidas de atividade precisam de muitas operações aritméticas como somas e multiplicações que são utilizadas nos cálculos de médias e variâncias, o que eleva muito a complexidade computacional, principalmente para a técnica *up-bottom*.

Pois se calcula a média e a variância deste bloco grande, se este bloco tiver que ser repartido, para cada um de seus blocos filhos deve-se recalcular suas médias e variâncias para saber se deve ou não prosseguir. O método *bottom-up* é mais eficiente quanto à complexidade computacional, pois se calcula a média e a variância dos blocos filhos, se comparando verifica-se que os mesmos devem ser agrupados e agrupando-os, a média do bloco resultante é a média das médias dos blocos menores, para a variância existe uma relação semelhante, isto é:

$$m_{4x4} = \frac{1}{4}\sum_{k=1}^{4} m_{(2x2)k}$$

$$\sigma^2_{4x4} = \frac{1}{4}\sum_{k=1}^{4} \sigma^2_{(2x2)k} - m^2_{4x4}$$

Figura 20. Cálculo para média e variância de blocos.
Fonte: MACLEAN, F e JERNIGAN, E. (1991).

m são as médias e σ2 as variâncias dos blocos. As informações anteriores dos blocos filhos são utilizadas para o cálculo das características do bloco pai, o que torna esta técnica muito eficiente (MACLEAN e JERNIGAN, 1991).

Na Figura 21 há um exemplo da segmentação de um *frame* em *quadtree* baseado na diferença entre este *frame* e o *frame* anterior compensado. Primeiro a imagem foi dividida em blocos de 32x32 pixels de tal forma que se adequasse as dimensões da imagem, que no caso apresentado é uma imagem no padrão CIF com 352x288 pixels, uma das exigências da decomposição regular da *quadtree*, é que as dimensões do nó raiz sejam potências de 2, de modo a permitir a aplicação recursiva do algoritmo. Desta forma cada bloco de 32x32 pixels da imagem apresentada é um nó raiz

que dará origem a uma *quadtree* cujas folhas foram definidas como sendo de 4x4 pixels. A estrutura de bits está apresentada na Figura 22, convencionando-se que o bit 1 representa que o bloco deve ser repartido e o bit 0 que o bloco não deve ser repartido. Os bits zero que correspondem às folhas são omitidos por não levarem informação alguma à estrutura, é necessário sabermos o tamanho da estrutura de bits da *quadtree* de modo a decodificá-la corretamente no receptor, esta informação é transmitida adicionando-se 11 bits no início da estrutura.

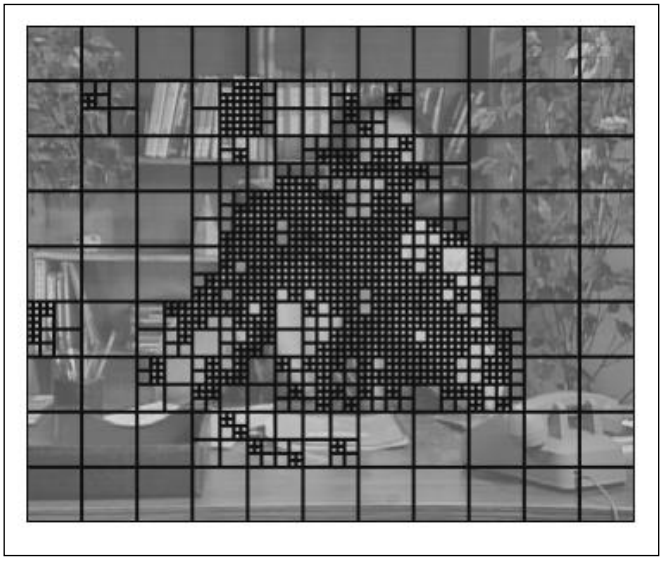


Figura 21. Composição real em *quadtree*.
Fonte: MACLEAN, F e JERNIGAN, E. (1991).

00000000000010111100000001111100000011111000000111111001011111110000111111100000
11100000000000000010000101101001010110010000011111111100100101111111111111100110
11111111111111111011101010100101110110101111111111111101110001110010111111111111011
11101000011000100100111111110101010000101000010001000010011000100110010110000010
11011001111100000010111111111111111101011111101011111111101011011111111110010000111
11001111101111111111111111011111111111101111111111101111011111101011111100110100011
11010001101010011101101000101111110100011010001111101111111111111111111101101111
11101011100101001010101011111001000010011011000010110011011111110001000111110101
10001011010101000101001100000011000

Figura 22. Estrutura em bits da *quadtree*.
Fonte: MACLEAN, F e JERNIGAN, E. (1991).

5 TRABALHOS CORRELATOS

Encontram-se nesta sessão outros trabalhos que assim como o projeto apresentado neste documento buscam a redução do custo computacional na codificação de vídeos digitais utilizando o padrão de codificação H.264/AVC.

5.1 A DUAL QUAD-TREE BASED VARIABLE BLOCK-SIZE CODING METHOD

Artigo científico publicado pela *Academic Press* em Orlando, California Estados Unidos da América no *Journal of Visual Communication and Image Representation* em 8 de novembro de 2010 com autoria de Yunfei Wang, Xuman Mao e Yun He onde os mesmos propõe uma estrutura de codificação com blocos de tamanhos variáveis baseados em uma *quadtree* dupla dando maior flexibilidade e expansibilidade, possibilitando que a predição e transformação do tamanho dos blocos possa ser configurada de acordo com necessário reduzindo a complexidade de implementação.

Os resultados apresentados nas simulações mostraram uma melhora considerável da performance, com uma baixa complexidade de implementação da estrutura proposta e otimização na codificação dos blocos de tamanho variável.

5.2 CODIFICADOR H.264/AVC COM COMPENSAÇÃO DE MOVIMENTO BASEADA EM PARTIÇÕES ALTERNATIVAS DE MACROBLOCO

Dissertação de mestrado em Engenharia Elétrica, apresentada por Renan Utida Ferreira à Universidade de Brasília em Janeiro de 2009, com o objetivo de reduzir

a complexidade computacional no processo de codificação de vídeo utilizando partições alternativas de macroblocos.

A dissertação referida obteve resultados satisfatórios, com uma redução média de 6,5% nas taxas de compressão em oito testes realizados.

5.3 UMA MÉTRICA PARA TAXA DE DISTORÇÃO VOLTADA PARA CODIFICAÇÃO DE VÍDEO PERCEPTIVA

Trabalho de conclusão de curso apresentado por Bruno George de Moraes à Universidade Federal de Santa Catarina em 2010 para a obtenção do grau de Bacharel em Ciência da Computação onde se propõe a aplicação de uma nova métrica de distorção de qualidade para compressão de vídeo. A plataforma de teste utilizada foi a do codificador do projeto x264 que é compatível com o especificado pelo padrão H.264/AVC. Nos testes obteve-se resultados positivos com ganhos na taxa de bit entre 2% e 60% dependendo do conteúdo do vídeo e atingiu também um aumento de desempenho em até 63% proporcional à redução da taxa de bits.

6 ESTRUTURA EM QUAD-TREE NA SEGMENTAÇÃO DOS MACROBLOCOS

Neste capítulo será apresentado o protótipo de estudo utilizado como referência no projeto assim como as principais funções encontradas no código fonte, o método de aplicação da estrutura em *quadtree* para a segmentação dos macroblocos e como foi feita a coleta dos resultados obtidos após cada teste realizado.

6.1 H.264/AVC JM REFERENCE 17.2

O JM é um código de referência do padrão H.264/AVC que engloba em si todas as especificações detalhadas na norma deste padrão. Ele é formado por um codificador chamado *lencod* e por um decodificador chamado *ldecod*. Neste projeto foi trabalhado apenas com o codificador.

Foi utilizada a versão 17.2, lançada em 27 de maio de 2010, atualmente encontra-se na versão 18.0, lançada em 03 de maio de 2011. O codificador *lencod* possui vários parâmetros de entrada, todos especificados em um arquivo texto no mesmo diretório do arquivo *lencod.exe*. Dependendo do perfil desejado, existem arquivos de texto já definidos para codificar um vídeo com as especificações recomendadas pelo padrão h.264, são eles:

a) “encoder_extended.cfg”: perfil *extended*;

b) “encoder_main.cfg”: perfil *main;*

c) “encoder_baseline.cfg”: perfil *baseline*;

d) “encoder.cfg”: Perfil *high*;

e) “encoder_yuv422.cfg”: perfil *High 4:2:2*;

f) “encoder_tonemapping.cfg”: perfil *High 4:4:4*.

Neste projeto foi usado o arquivo pré definido “*encoder_baseline.cfg*” que especifica as ferramentas utilizadas pelo perfil *baseline*, muito recomendado para aplicações de vídeo conferência. A Tabela 7 mostra os principais parâmetros definidos no arquivo “*encoder_baseline.cfg*”.

Tabela 7. Parâmetros de entrada do arquivo “*encoder_baseline.cfg*”.

Parâmetro	**Valor**	**Descrição**
ImputFile	"Salesman.yuv"	Sequencia de entrada
InputHeaderLength	0	Tamanho do cabeçalho da sequência de entrada
StartFrame	0	Frame de inicio da codificação
FramesToBeEncoded	449	Quantidade de frames a codificar
FrameRate	30.0	*Frames* por segundo
SourceWidth	176	Largura da sequência de entrada
SourceHeight	144	Altura da sequência de entrada
OutputWidth	176	Largura da sequência de saída
OutputHeight	144	Largura da sequência de entrada
ReconFile	“teste_rec.yuv”	Arquivo YUV reconstruído
OutputFile	“test.264”	Arquivo de *bitstream*
StatsFile	“Stats.dat”	Arquivo com resultados da codificação

Fonte: Arquivo “*encoder_baseline.cfg*”.

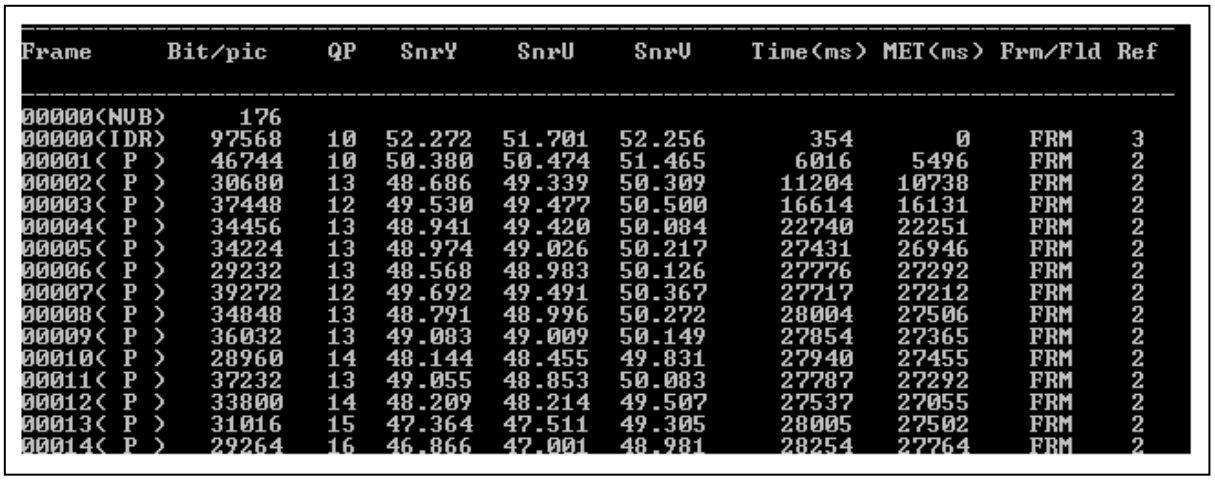
6.1.1 Parâmetros de Entrada

Sabe-se que o codificador é executado pelo arquivo "*lencod.exe*", o codificador necessita de um outro arquivo, onde neste, são definidos os vários parâmetros de entrada para o processo de codificação que vão desde , a quantidade de *frames* a codificar até ao modo de segmentação. A opção "*–d*" permite indicar o nome do arquivo que irá fornecer os parâmetros de entrada, como exemplificado a seguir: *"lencod.exe -d c:\encoder_baseline.cfg"* o arquivo padrão é o *"encoder.cfg"*.

Todos os parâmetros de entrada do codificador estão listados e documentados no manual do programa que se encontra na pasta /*doc,* arquivo de nome "*JM Reference Software Manual.doc*" no diretório raiz do JM.

6.1.2 Formatos de Saída

O *lencod* tem como saída um arquivo reconstruído no formato YUV (test_rec.yuv) e um arquivo em formato H.264 (test.264), assim como as estatística do codificador, que apresentam as taxas, a qualidade das sequências reconstruídas e os tempos necessários para a codificação das mesmas. Quando executado, o codificador ira mostrar na saída padrão (console) os valores de tempo e qualidade para cada quadro codificado. Uma média dos resultados também é gerada no final, além de também registrar estes resultados no arquivo "stats.dat" que pode ser visualizado por qualquer editor de texto. A Figura 23 mostra os dados fornecidos durante a codificação, a Tabela 8 explica o significado de cada nomenclatura e a Figura 24 demonstra a média dos resultados depois de concluída a codificação.



Frame	Bit/pic	QP	SnrY	SnrU	SnrV	Time(ms)	MET(ms)	Frm/Fld	Ref
00000(NVB)	176								
00000(IDR)	97568	10	52.272	51.701	52.256	354	0	FRM	3
00001(P)	46744	10	50.380	50.474	51.465	6016	5496	FRM	2
00002(P)	30680	13	48.686	49.339	50.309	11204	10738	FRM	2
00003(P)	37448	12	49.530	49.477	50.500	16614	16131	FRM	2
00004(P)	34456	13	48.941	49.420	50.084	22740	22251	FRM	2
00005(P)	34224	13	48.974	49.026	50.217	27431	26946	FRM	2
00006(P)	29232	13	48.568	48.983	50.126	27776	27292	FRM	2
00007(P)	39272	12	49.692	49.491	50.367	27717	27212	FRM	2
00008(P)	34848	13	48.791	48.996	50.272	28004	27506	FRM	2
00009(P)	36032	13	49.083	49.009	50.149	27854	27365	FRM	2
00010(P)	28960	14	48.144	48.455	49.831	27940	27455	FRM	2
00011(P)	37232	13	49.055	48.853	50.083	27787	27292	FRM	2
00012(P)	33800	14	48.209	48.214	49.507	27537	27055	FRM	2
00013(P)	31016	15	47.364	47.511	49.305	28005	27502	FRM	2
00014(P)	29264	16	46.866	47.001	48.981	28254	27764	FRM	2

Figura 23. Saída de dados em tempo de codificação.
Fonte: Software JM 17.2 durante a codificação.

Tabela 8. Cabeçalho da saída de dados.

Nome	**Finalidade**
Frame	Numero e tipo do0 quadro
Bit/pic	Quantidade de bits alocado para o quadro
WP	Predição Ponderada
QP	Valor de quantização do quadro
SnrY	PSNR de luminância Y
SnrU	PSNR de Crominância U
SnrV	PSNR de Crominância V
Tempo(ms)	Tempo para codificar o quadro
MET(ms)	Tempo da estimação de movimento do quadro
Frm/Fld	Modo de codificação, frame ou Field

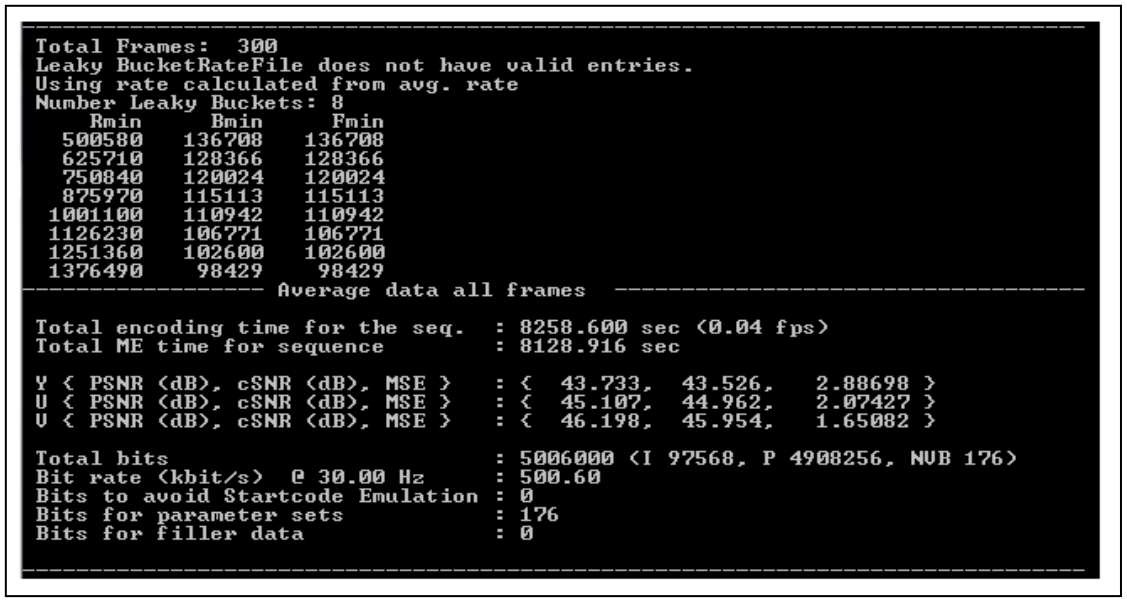


Figura 24. Resultados apresentados ao final da codificação.
Fonte: Software JM 17.2 ao fim da codificação.

6.2 MÓDULOS DO LENCOD ESPECIFIVADOS NO PADRÃO H.264

O estudo do código fonte do JM se deu por meio do ambiente de programação *Microsoft Visual C++ 2008*, sabe-se que o JM é um código de referência do padrão H.264/AVC, apesar de existir um código fonte na linguagem Java, optou-se em usar o JM devido sua melhor organização e principalmente por estar mais bem documentado.

A seguir serão apresentados os principais módulos do padrão H.264 que estão implementados no software de referência JM, versão 17.2.

6.2.1 Função *Main* - main

Como por padrão na maioria dos códigos, ao iniciar a compilação o *encoder* chama a função *main*. A Figura 25 demonstra a estrutura de funções da função *main*.

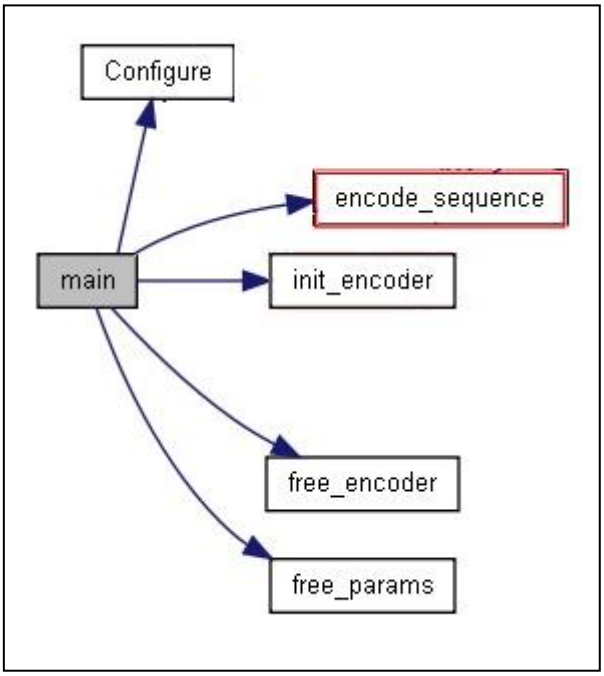


Figura 25. Estrutura da função *Main.*
Fonte: APPLE, C. (2009).

A função *Configure* analise os parâmetros passados na linha de comando e lê o arquivo de configuração, neste caso o arquivo "*encoder_baseline.cfg*".

O codificador é inicializado pela função *init_encoder*, nesta função é aberto o arquivo a que será codificado (o vídeo), inicializa as matrizes *Q* e *QOffSet*, os parâmetros de RDO (técnica que minimiza a função de custo de taxa-distorção), prepara as GOPs (sequência de quadros, geralmente é composto de 12 a 15 quadros), e inicia o módulo de busca por movimento *Init_Motion_Search_Module*. A Figura 26 mostra a estrutura de um *bitstream* de vídeo e nos dá uma melhor visão de como é feito o acesso às estruturas da sequência pelo JM.

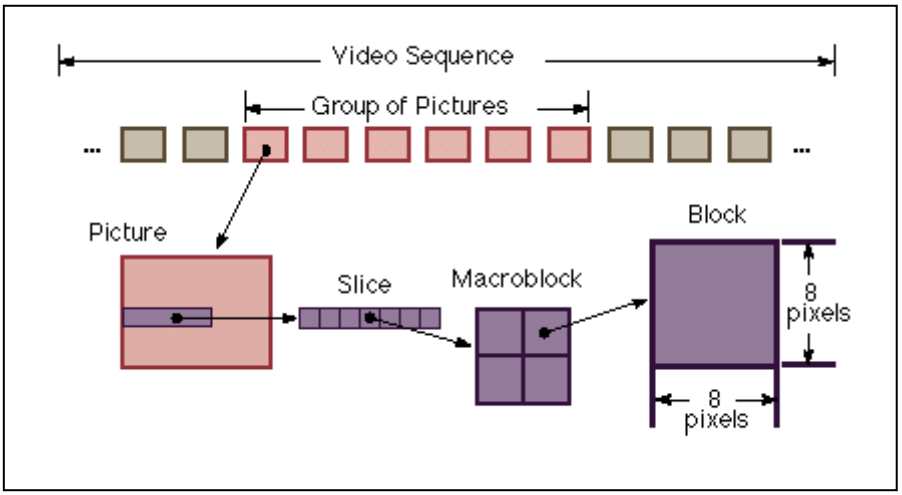


Figura 26. *Bitstream* de vídeo.
Fonte: VICTOR, L. (2011).

Terminada as funções configure e *init_encoder*, chega a vez da função *encode_sequence*, esta é a principal função encontrada no *main*, é ela quem faz a codificação de toda a sequência de vídeo, passando por todos os *frames* da sequência, um por vez, até que todos os *frames* sejam codificados. Esta codificação tem início na função *encode_sequence* e segue a função *encode_one_frame* que será apresentada logo abaixo.

6.2.2 Função *Encode One Frame* - encode_one_frame

Codificando um dos vários *frames* da sequência, a função *encode_one_frame* inicia carregando os parâmetros do *frame* que será codificado e faz a leitura dos dados do *frame* para o arquivo de entrada. O processo de codificação de um *frame* funciona sub-dividindo um *frame* em *slices*.

A codificação pode ser feita de duas maneiras diferentes, por *field* ou *frame*, o processo de codificação é semelhante em ambos os modos, diferenciando apenas em como a imagem é representada, em um *frame* ou em dois *fields (bottom e top)*. Na

codificação por *frames* é chamada a função *perform_encode_frame* inicializam os parâmetros do *frame*, que por sua vez chama a função *frame_picture*, responsável por atualizam os limites do vetor de movimento.

A função *frame_picture* prepara e aloca na memória a estrutura de dados do frame através da função *prepare_enc_frame_picture* e em seguida continua a codificação chamando a função *code_a_picture*. A Figura 27 demonstra a sequência de funções até chegar na função *code_a_picture* que será comentada no tópico seguinte.

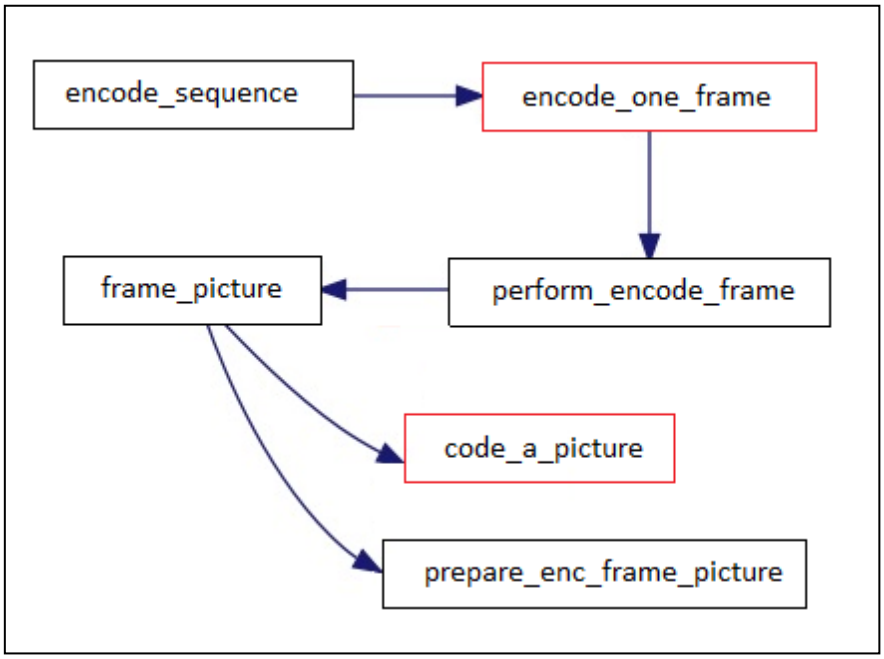


Figura 27. Sequência de funções, *encode_sequence* à *code_a_picture.*
Fonte: APPLE, C. (2009).

6.2.3 Funções *Code a Picture* e *Code a Plane* - code_a_picture e code_a_plane

A função *code_a_picture* é a principal responsável pela codificação da imagem, ela é chamada em todos os *frames* logo após serem inicializados os parâmetros dos mesmos. Primeiramente ela inicia a função R*andomIntraNewPicture* que seleciona

um grupo de macroblocos para forçar a sua predição intra, esta função tem utilidade apenas se a codificação for por *fields*.

Em seguida é chamada a função *code_a_plane*, esta responsável pela codificação de um plano. Primeiro é inicializado o FMO através da função *FmoInit*, em seguida calcula os valores de quantização do frame e entra em um *loop* que percorre todos os *slices*, afim de codificalos. Dentro deste loop é chamada a função *encode_one_slice* que faz a codificação do *slice* e está descrita na próxima sessão .

6.2.4 Função *Encode One Slice* - encode_one_slice

Esta função inicia localizando o primeiro macrobloco não codificado do *slice*, em seguida inicializando o *slice* através da função *init_slice*, também são executadas as funções de quantização baseadas em arredondamento, depois vem a função *start_slice* que gera os cabeçalhos e partições do *slice* e retorna a quantidade de bits usada pelo cabeçalho.

Concluída a inicialização do *slice*, são atualizadas algumas estatísticas e começa então a codificação dos macroblocos. A função entra em um *loop*, percorrendo todos os macroblocos do *slice*. Neste *loop* há duas funções principais chamadas *start_macroblock* e *encode_one_macroblock* responsáveis por iniciar e codificar um macrobloco respectivamente.

Se após codificar um macrobloco for decidido que não é necessário recodificar o mesmo, incrementa-se o contador de macroblocos codificados e chama a função *next_macroblock* que atualiza os valores referentes ao próximo macrobloco.

Para terminar a função *terminate_slice* finaliza o *slice* escrevendo o bit que indica o fim do *slice*.

6.2.5 Funções *Start Macroblock e Encode One Macroblock*

O *lencod* inicia o processo de codificação de um macrobloco chamando a função *start_macroblock*. O objetivo principal desta função é inicializar o macrobloco atual (que será codificado), este processo é feito atualizando-se as coordenadas do próximo macrobloco através da função *set_MB_parameters*, seguindo no *start_macroblock,* a função *FmoGetPreviousMBNr* localiza o macrobloco anterior se houver e a função *CheckAvailabilityOfNeighbors* verifica se os macroblocos vizinhos do macrobloco atual estão disponíveis para a predição, setando todos os vizinhos como indisponíveis e quando confirmada a disponibilidade armazena as coordenadas do macrobloco na respectiva variável.

Saindo da função *start_macroblock* e seguindo para a *encod_one_macroblock.* É nesta função que acontece o processamento pesado do codificador, primeiro é definido o melhore modo de decisão para o macrobloco, em seguida são inicializados os parâmetros de codificação para o macrobloco pela função *init_enc_mb_params*. Carregado os parâmetros, visto que a predição não é do tipo *intra*, mas sim do tipo P, executa-se a função *FindSkipModeMotionVector* que irá buscar o vetor de movimento para o modo *skip*, no modo *skip* as informações do bloco anterior são utilizada para montar o bloco atual, não sendo necessária nenhuma informação extra.

Finalmente é feita a estimação de movimento e decisão de modo, primeiramente no blocos de 16 x 16, 16 x 8 e 8 x16 e depois nas sub-partições dos blocos de 8 x 8. Depois de estimado o movimento é escolhido o melhor modo para o macrobloco e setado os parâmetros finais. Por fim a função *write_macroblock* atualiza o quantidade de macroblocos *intra*, marca o símbolo de término no macrobloco e incrementa o numero total de macroblocos codificados

6.3 ABORDAGEM POR MEIO DA ESTRUTURA QUADTREE

Assim como especificado no padrão H.264/AVC, o JM tem desenvolvido em sua estrutura todas as possibilidades de segmentação de macroblocos utilizando tanto blocos retangulares como blocos regulares. Sabe-se que a segmentação abordada no padrão H.264 possibilita a subdivisão de um macrobloco em blocos e sub-blocos retangulares e a segmentação baseada na abordagem em H.264 com *quadree* possibilita a subdivisão de um macrobloco em blocos e sub-blocos retangulares regulares.

Alterando-se as configurações dos parâmetros de entrada do codificador pode-se modificar a estrutura de referência JM para uma estrutura baseada em *quadtree* para segmentar os macroblocos na predição *inter*. Manipulando-se o arquivo de configuração do *lencod* (“*encoder_baseline.cfg*”) onde são definidos os parâmetros de entrada e de codificação, constatou-se a possibilidade de desativar as opções de segmentação em blocos retangulares (16 x 8, 8 x 16, 8 x 4 e 4 x8) e deixar apenas as segmentações de blocos regulares (16 x 16, 8 x 8 e 4 x 4), estes por sua vez que compõem uma *quatree*. Neste mesmo arquivo é definida a taxa de bits da codificação. A Figura 28 mostra as linhas alteradas do arquivo de configuração do *lencod*.

```
###############################################################################
# PSlice Mode types
###############################################################################
PSliceSkip            = 1   # P-Slice Skip mode consideration  (0=disable, 1=enable)
PSliceSearch16x16     = 1   # P-Slice Inter block search 16x16 (0=disable, 1=enable)
PSliceSearch16x8      = 0   # P-Slice Inter block search 16x8  (0=disable, 1=enable)
PSliceSearch8x16      = 0   # P-Slice Inter block search  8x16 (0=disable, 1=enable)
PSliceSearch8x8       = 1   # P-Slice Inter block search  8x8  (0=disable, 1=enable)
PSliceSearch8x4       = 0   # P-Slice Inter block search  8x4  (0=disable, 1=enable)
PSliceSearch4x8       = 0   # P-Slice Inter block search  4x8  (0=disable, 1=enable)
PSliceSearch4x4       = 1   # P-Slice Inter block search  4x4  (0=disable, 1=enable)
```

Figura 28. Alteração no método de segmentação.
Fonte: Arquivo "*encoder_baseline.cfg*".

6.4 AMBIENTE DE TESTES E DE ANÁLISE

Aqui serão analisadas as sequências utilizadas para a realização dos testes, o método de medição de qualidade, o sistema onde foram aplicados os testes e a maneira como foram colhidos os dados resultantes das codificações.

6.4.1 Sequências de Testes

Os testes foram realizados com três sequências frequentemente utilizadas na realização simulações e testes para mensuração de eficiência e desempenho de codificação de vídeo. A Figura 29 apresenta as informações básicas de cada uma delas:

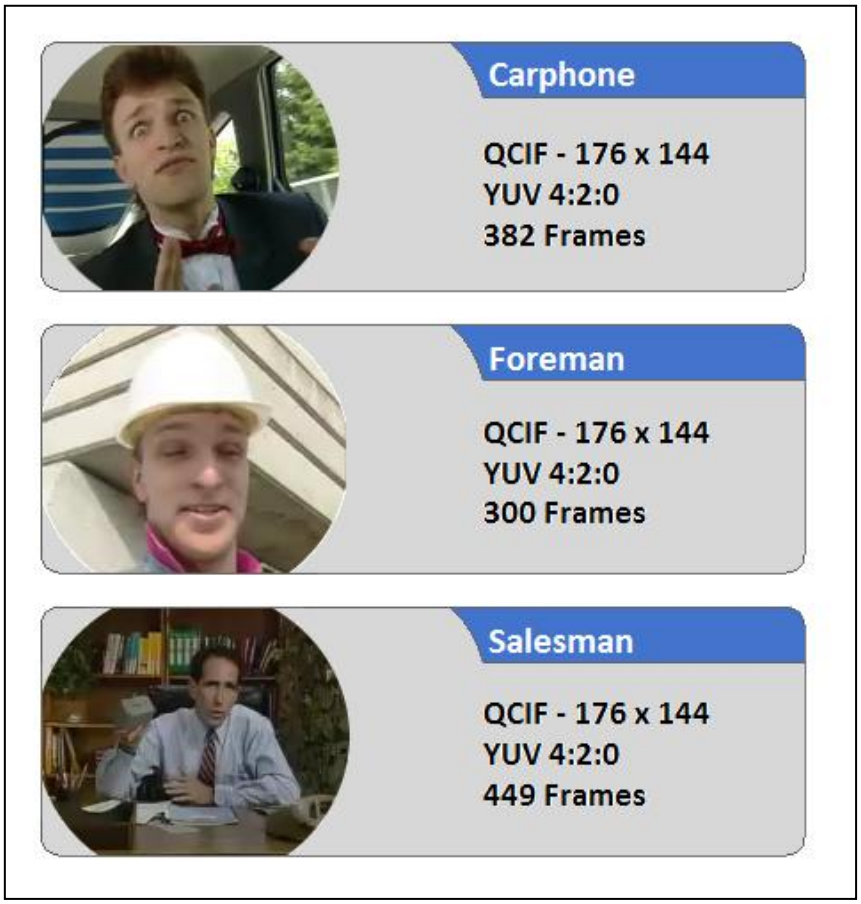


Figura 29. Informações das sequências utilizadas.

As sequências estão no formato QCIF YUV 4:2:0, na componente de luminância possuem 176 x 144 pixels e na de crominância 88 x 72 pixels, em ordem de complexidade computacional entra primeiro a sequência Carphone em seguida a Foreman e por último a Salesman, segue abaixo um detalhamento de cada sequência.

a) Carphone: apresenta grande movimentação, além de ter o sujeito falando e gesticulando, há grande mudança de cenário na janela do carro que está em movimento;

b) Foreman: sequência com média movimentação, conta com um operário falando, e em seguida apresentando a obra;

c) Salesman: esta sequência possui movimentação reduzida em relação às anteriores, o sujeito fica no centro da tela gerando maiores movimentações com as mãos e a boca e o cenário é fixo.

O formato QCIF possui um quarto dos pixels do formato CIF, ele foi utilizado neste projeto à sua grande presença em vídeos para celular e em aplicações de videoconferência, área esta onde uma perda mínima de qualidade se torna irrelevante e imperceptível ao sistema visual humano.

6.4.2 Métricas Objetivas de Qualidade

A qualidade objetiva é medida usando-se duas métricas que comparam o vídeo original com o vídeo codificado. Esta medida é realizada quadro a quadro, comparando-se os pixels dos quadros do vídeo original com os pixels dos quadros do vídeo codificado e decodificado.

A Razão Sinal-Ruído de Pico - *Peak Signal-to-Noise Ratio* (PSNR) - é o parâmetro de avaliação mais utilizado para comparar a qualidade objetiva de vídeos (BHASKARAN, 1997) e está definido na fórmula apresentada na Figura 30. A PSNR é usada para fazer uma comparação entre dois quadros distintos que, neste caso, são o quadro original e o quadro reconstruído. Na Figura 31, *MAX* é o valor máximo que uma amostra de luminância do vídeo pode atingir. A PSNR é medida em uma escala logarítmica, utilizando como base, o critério de similaridade Erro Médio Quadrático - *Mean Squared Error* (MSE) ente o quadro original e o quadro reconstruído.

$$PSNR = 20 \times \log\left(\frac{MAX}{\sqrt{MSE}}\right)$$

Figura 30. Fórmula para comparar a qualidade objetiva de vídeos.
Fonte: KUHN, P. (1999).

A qualidade objetiva está diretamente ligada ao resultado da PSNR obtido. Quanto maior o valor maior a qualidade. É importante saber que o PSNR possui certas limitações como critério de qualidade, sendo que, em alguns casos uma imagem possui uma qualidade subjetiva superior mais com o valor do PSNR inferior à outra imagem com um visual pior (RICHARDSON, 2003). Isto acontece porque o PSNR leva em consideração apenas o valor do MSE para cada pixel da imagem e, desta maneira, não avalia os critérios subjetivos de qualidade (RICHARDSON, 2003).

Uma importante medida para avaliar a diferença ente dois quadros esta no grau de similaridade entre eles. Este grau de similaridade pode ser mensurado a partir de diferentes funções de similaridade. As funções de similaridade que interessam neste trabalho são o MSE, usado para definir o PSNR, e o *Sum of Absolute Differences* (SAD), usado na predição inter-quadros.

O cálculo para encontrar o MSE está definido na fórmula apresentada na Figura 31, em que *m* e *n* são as dimensões do quadro que será comparado e *O* e *R* são as amostras dos quadros originais e reconstruídos respectivamente.

$$MSE = \frac{1}{mn}\sum_{i=0}^{m-1}\sum_{j=0}^{n-1}(R_{i,j} - O_{i,j})^2$$

Figura 31. Fórmula para encontrar o MSE.
Fonte: KUHN, P. (1999).

O SAD é usado para encontrar a distorção entre as regiões comparadas, a partir do somatório das diferenças absolutas, de cada ponto do quadro original e quadro reconstruído (KUHN, 1999). A função para encontrar o SAD está apresentada na Figura 32. Onde *m* e *n* são as dimensões do quadro a ser comparado e *O* e *R* são as amostras dos quadros originais e reconstruídos, respectivamente.

$$SAD = \sum_{i=0}^{m-1} \sum_{j=0}^{n-1} \left| R_{i,j} - O_{i,j} \right|$$

Figura 32. Fórmula para calcular o SAD.
Fonte: KUHN, P. (1999).

6.4.3 Dados Utilizados para Análise

Para analisar o resultado da implementação da proposta, foi registrado as informações da PSNR de Y e tempo de codificação de todos os quadros processados, assim como a taxa de bits, PSNR de Y e tempo de codificação médio das três sequências de vídeo para segmentação em *quadtree* e segmentação especificada no padrão H.264.

Foram realizadas outras codificações para análise de resultados utilizando apenas sequência Foreman, nestes testes foi definido um bit rate de 20kbps, 50kbps, 100kbps, 200kbps, 500kbps e 1Mbps. Em todos os testes foram feitas comparações entre a codificação com segmentação em *quadtree* e segmentação do padrão H.264.

7 RESULTADOS E ANÁLISES

Aqui serão apresentados, comparados e discutidos os resultados obtidos com os testes de codificação realizados. Os resultados serão apresentados por meio de gráficos de taxa distorção, utilizando a PSNR como medida de qualidade de reconstrução, tempo de codificação em segundos, taxa de bits medida em kbps e a quantidade de bits alocada para a codificação. Os resultados têm por base os valores do componente de luminância (Y). Os testes realizados são para os três componentes YUV, mas como os componentes UV dependem do componente Y e o sinal de brilho (Y) possui mais detalhes, a sua análise tornou-se mais importante.

Os testes foram realizados codificandos todos os *frames* de cada sequência (Carphone 382 frames, Foreman 300 frames e Salesman 449 frames) a uma taxa de 30fps com no máximo 5 frames de referência para a codificação.

7.1 ANÁLISE QUADRO A QUADRO

Os gráficos a seguir foram construídos a partir da análise dos resultados de tempo de codificação, PSNR e bits alocados pelo frame de todos os quadros da respectiva sequência durante o processo de codificação. Todos os testes foram realizados com os parâmetros já especificados no arquivo de configuração do JM, utilizando o arquivo "*encoder_baseline.cfg*" e alterando apenas o que foi documentado na Sessão 6.3 possibilitando o uso da estrutura *quadtree*.

7.1.1 Alocação de bits por quadro codificado

Os dados a seguir representam, para as três sequências de teste, a quantidade de bits de cada quadro após os processos de codificação por meio da segmentação convencional do padrão H.264 e por meio da segmentação proposta.

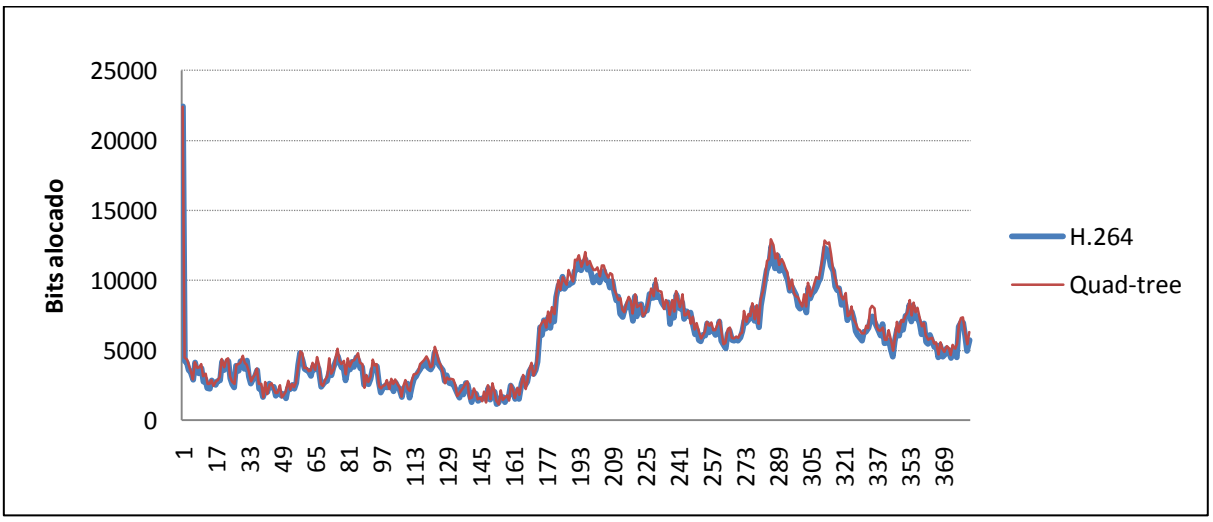


Figura 33. Alocação de bits da sequência Carphone.

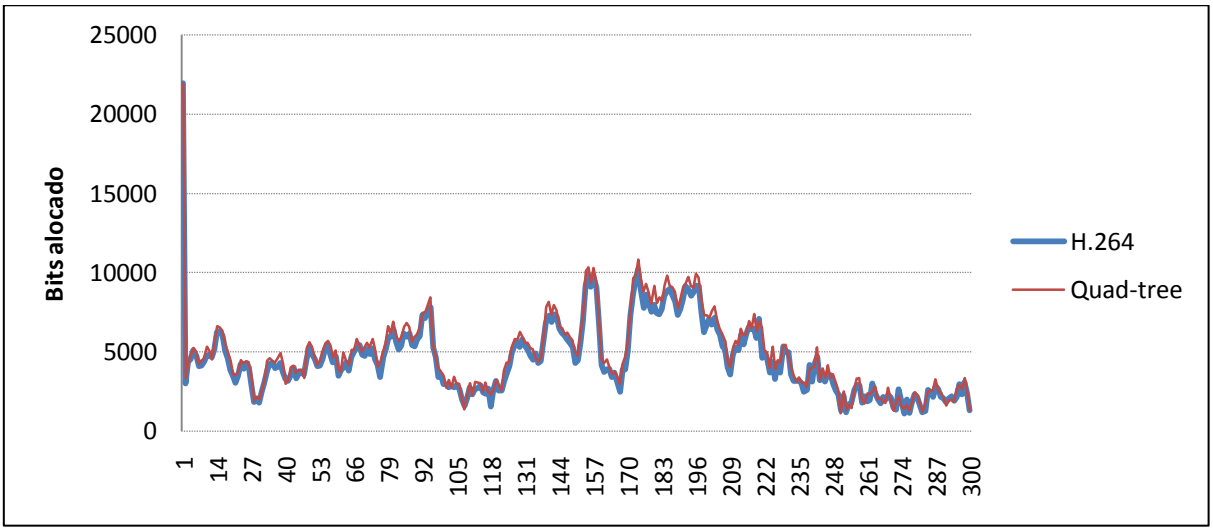


Figura 34. Alocação de bits da sequência Foreman.

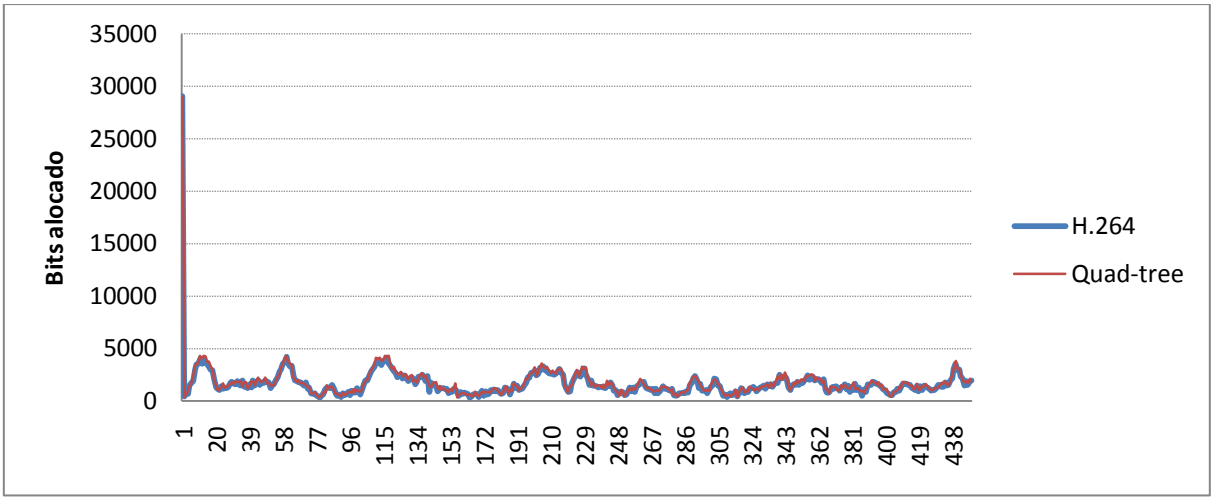


Figura 35. Alocação de bits da sequência Salesman.

Pode-se observar nas Figuras 33, 34 e 35, em suas comparações da quantidade de bits alocada por quadro numa codificação com H.264 e com *quadtree* que, ambas apresentam pouca diferença na quantidade de bits alocados, estando na maior parte da codificação bem próximas como se percebe pelas linhas dos gráficos. Nesta simulação, as sequências foram obtidas com a codificação H.264 a 170,41 kbps (Carphone), 137,3 kbps (Foreman) e 49,1 kbps (Salesman) contra os 178,62 kbps (Carphone), 147,34 kbps (Foreman) e 51,84 kbps (Salesman) apresentados pela codificação abordagem em *quadtree*.

7.1.2 PSNR resultante de cada quadro

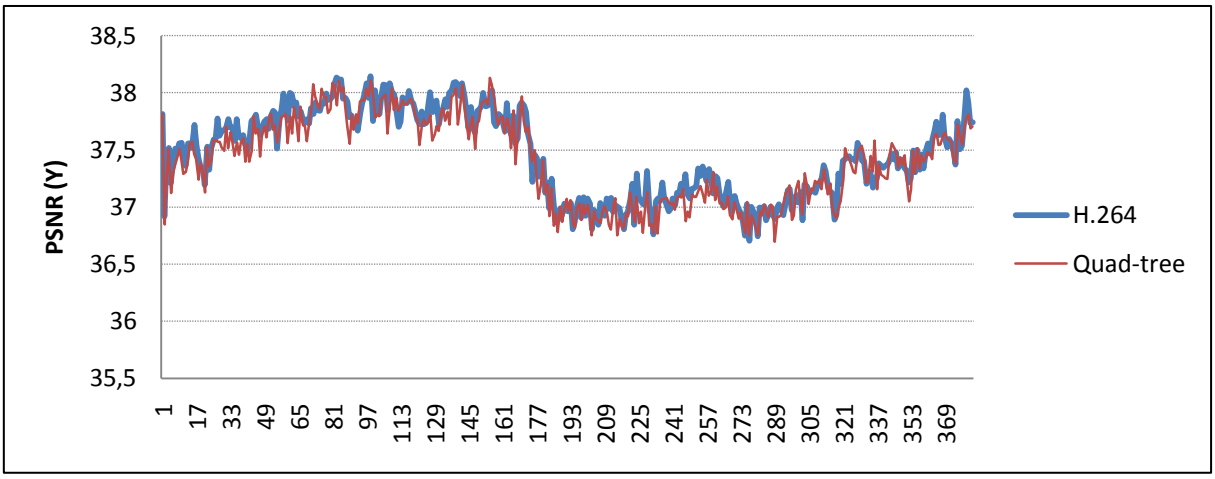


Figura 36. PSNR da sequência Carphone.

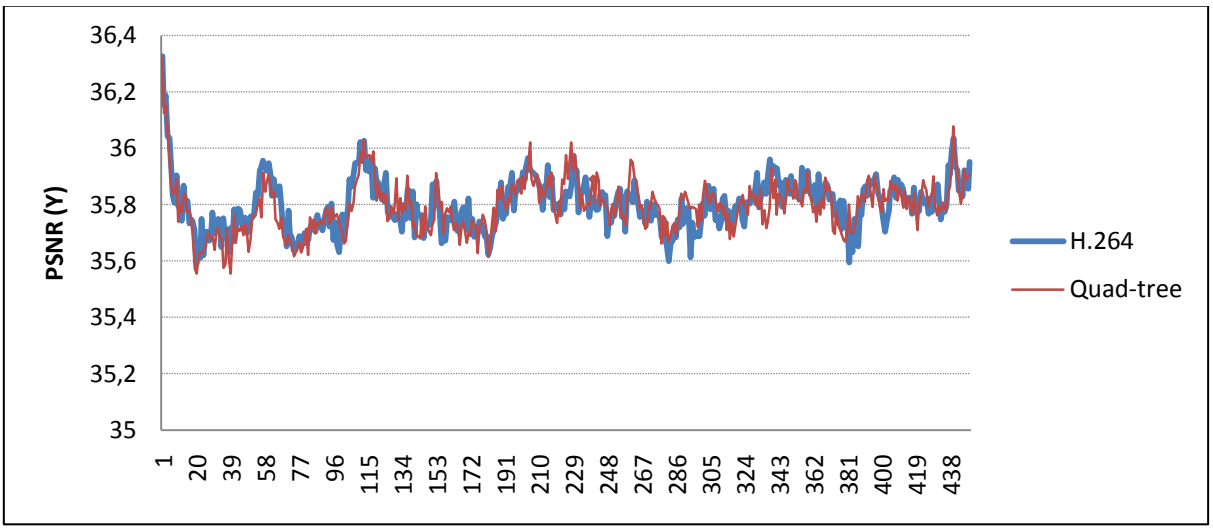


Figura 37. PSNR da sequência Salesman.

Nas Figuras 36 e 37, que demonstram a PSNR resultante da codificação de cada quadro da respectiva sequência, nota-se que a codificação em *quadtree* apresenta na maior parte da codificação um patamar um pouco abaixo da codificação feita pela segmentação do H.264, e em ambas percebe-se a existência de alguns picos pra baixo e também picos pra cima, desfavorecendo e favorecendo respectivamente a segmentação *quadtree*. Estas pequenas diferenças não acarretam grandes perdas da qualidade visual do sequência, tornando-se imperceptíveis à quem simplesmente assiste ao vídeo.

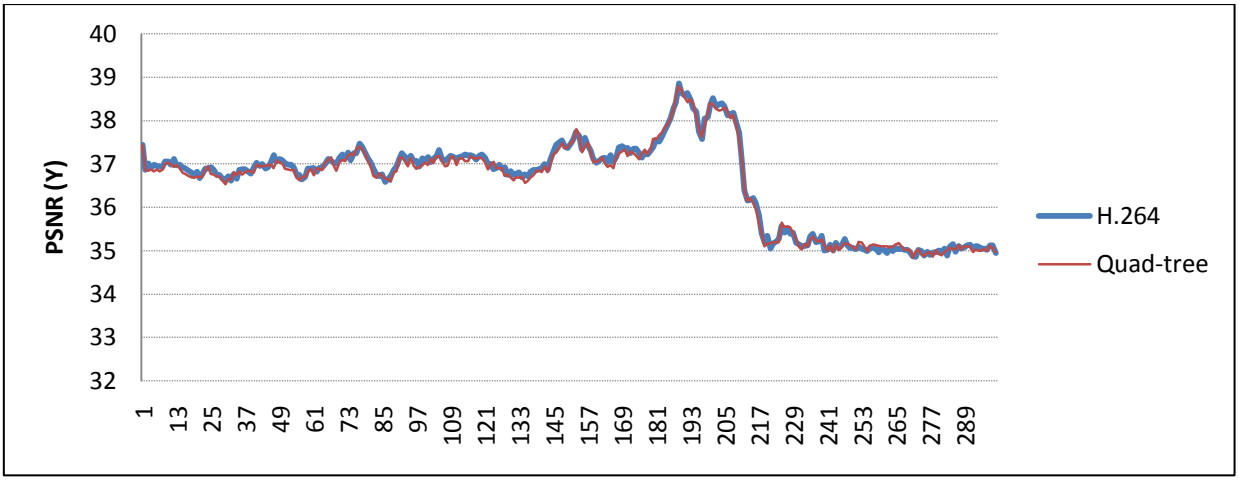


Figura 38. PSNR da sequência Foreman.

Já na Figura 38 que mostra o gráfico de PSNR da codificação da sequência Foreman, observa-se a inexistência de picos nos resultados das PSNR, apresentando um gráfico quase igual tanto para a codificação com segmentação *quadtree* quanto para a segmentação do padrão H.264.

7.1.3 Tempo de codificação por quadro

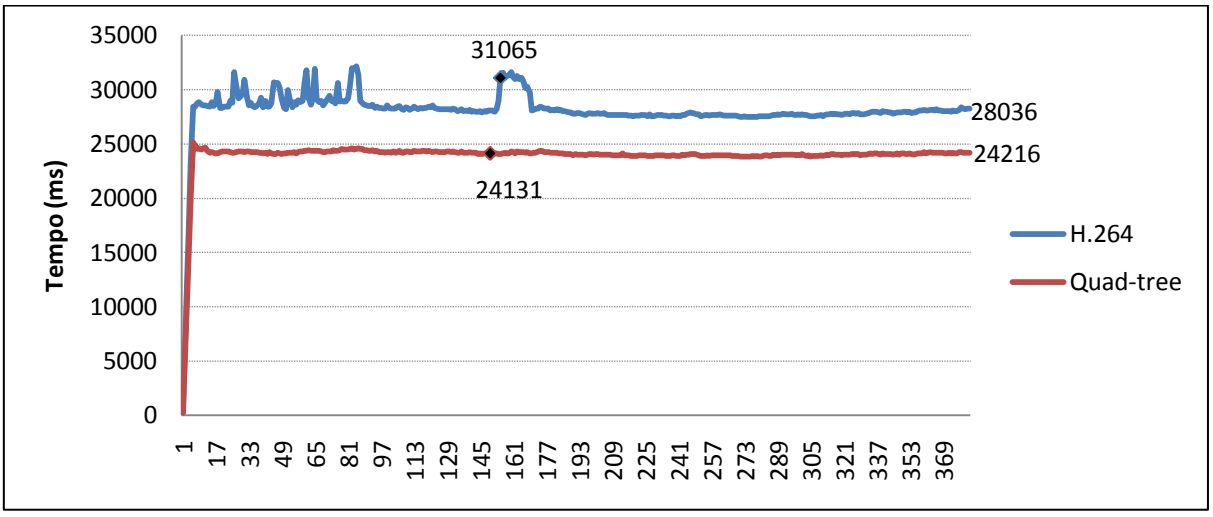


Figura 39. Tempo de codificação dos quadros da sequência Carphone.

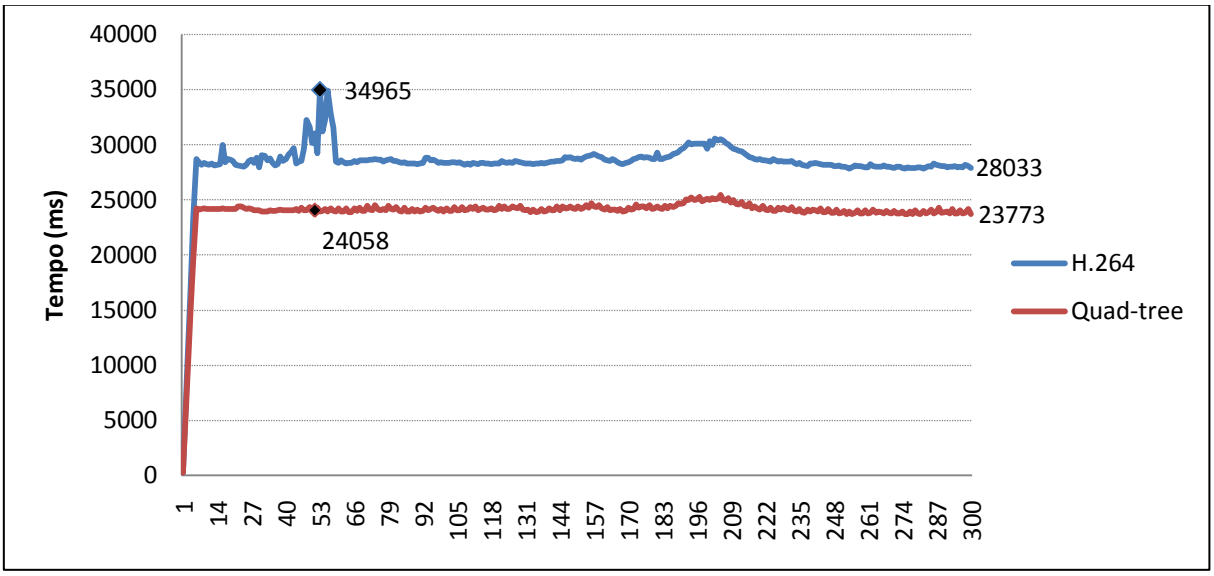


Figura 40. Tempo de codificação dos quadros da sequência Foreman.

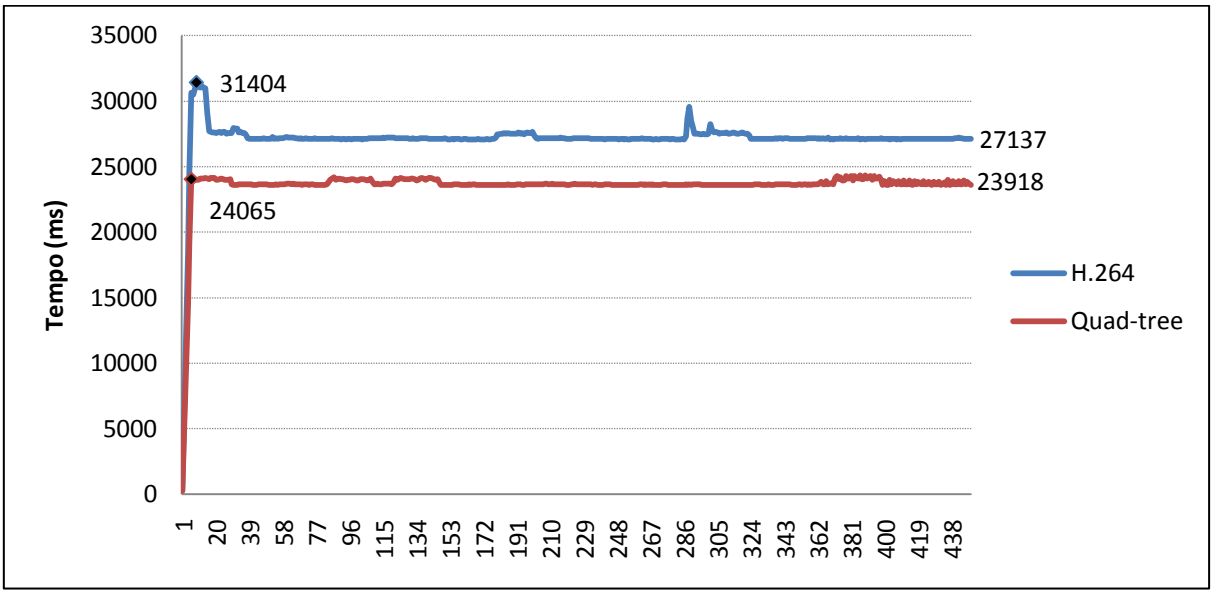


Figura 41**.** Tempo de codificação dos quadros da sequência Salesman.

As Figuras 39, 40 e 41 demonstram o tempo gasto para a codificação de cada quadro das sequências testadas. Pode-se observar que é grande a redução do custo computacional quando se comparam a codificação com a segmentação do padrão H.264 com a segmentação em *quadtree*. Na maior parte da codificação a redução é de 4 segundos chegando a picos de 7 e até 10 segundos.

7.1.4 Comparação, Frames Original e Codificado

Será apresentando a seguir o frame original de cada sequência seguido pelo resultado da sua codificação utilizando a segmentação do padrão H.264 e da segmentação proposta. Observando as Figuras 42, 43 e 44 pode-se notar que a redução da qualidade após a codificação em termos visuais é zero.

Figura 42. Foreman, (a) original (b) H.264 (c) *quadtree*.

Na sequência Foreman a codificação com o H.264 foi feita em 8524 segundos, com uma taxa de bits de 137,30 kbps e PSNR de 36,599 dB, pela codificação com abordagem em *quadtree* a codificação foi feita em 7185 segundos, apresentou uma taxa de bits de 147,34 kbps e uma PSNR de 36,563 dB.

Figura 43. Salesman, (a) original (b) H.264 (c) *quadtree*.

A codificação da sequência Salesman pelo padrão H.264 foi feita em 12185 segundos com uma taxa de bits de 49,1 kbps e PSNR de 35,8 dB e com a abordagem em *quadtree* a codificação foi concluída em 10591 segundos obtendo uma taxa de bits de 51,84 kbps e PSNR de 35,794 dB.

Figura 44. Carphone, (a) original (b) H.264 (c) quadtree.

A sequência Carphone foi codificada pelo padrão H.264 em 10736 segundos, obteve uma taxa de bits de 170,41 kbps e PSNR de 37,457 dB contra os 9150 segundos, 178,62 kbps e PSNR de 37,406 dB obtidos pela codificação com a *quadtree*.

7.2 MÉDIAS RESULTANTES

A seguir serão apresentadas as médias de tempo de codificação, PSNR, alocação de bits e bit rate gerados ao final de cada codificação. Cada imagem constitui o gráfico com o resultado das três sequências, comparando o resultado da codificação feita com a segmentação do padrão H.264 com a segmentação *quadtree*.

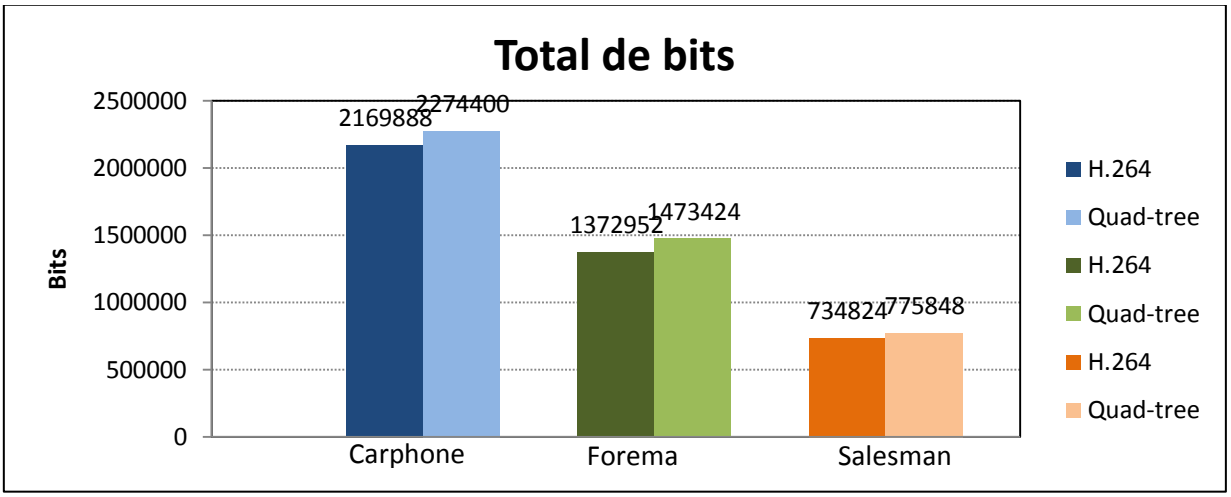


Figura 45. Total de bits gerado ao final da codificação.

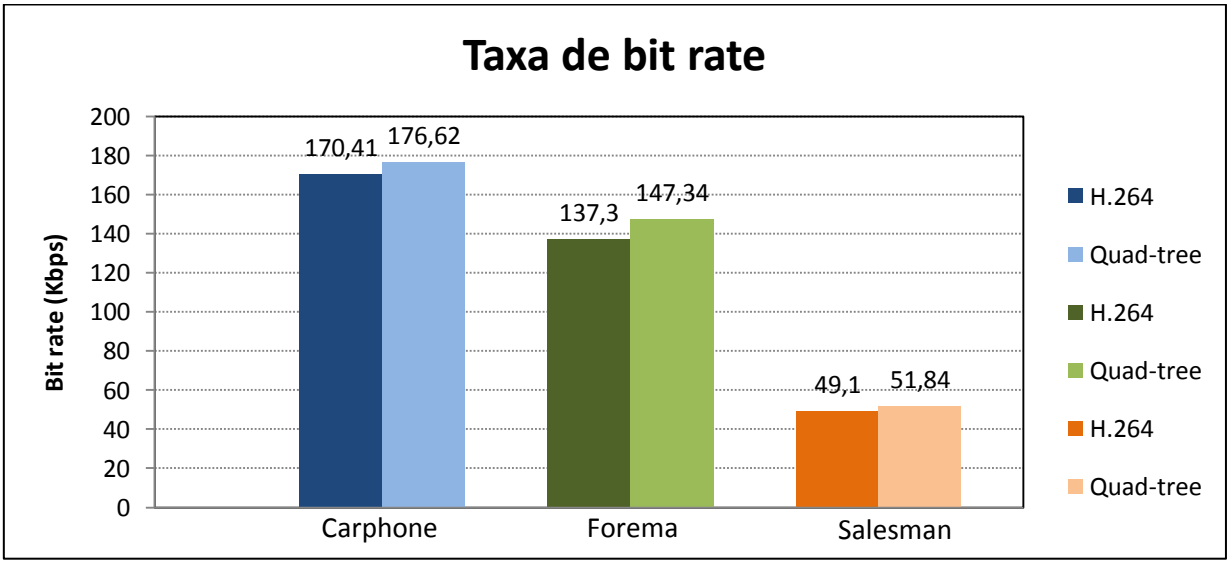


Figura 46. Taxa bit rate gerada ao final de cada codificação.

Observando as Figuras 45 e 46, constata-se que o padrão de segmentação *quadtree* resultou em todas as três sequências de teste, em uma quantidade maior de bits armazenados por segundo (bit rate) gerando uma maior qualidade, o que consequentemente traz a necessidade de mais espaço para o armazenamento do vídeo.

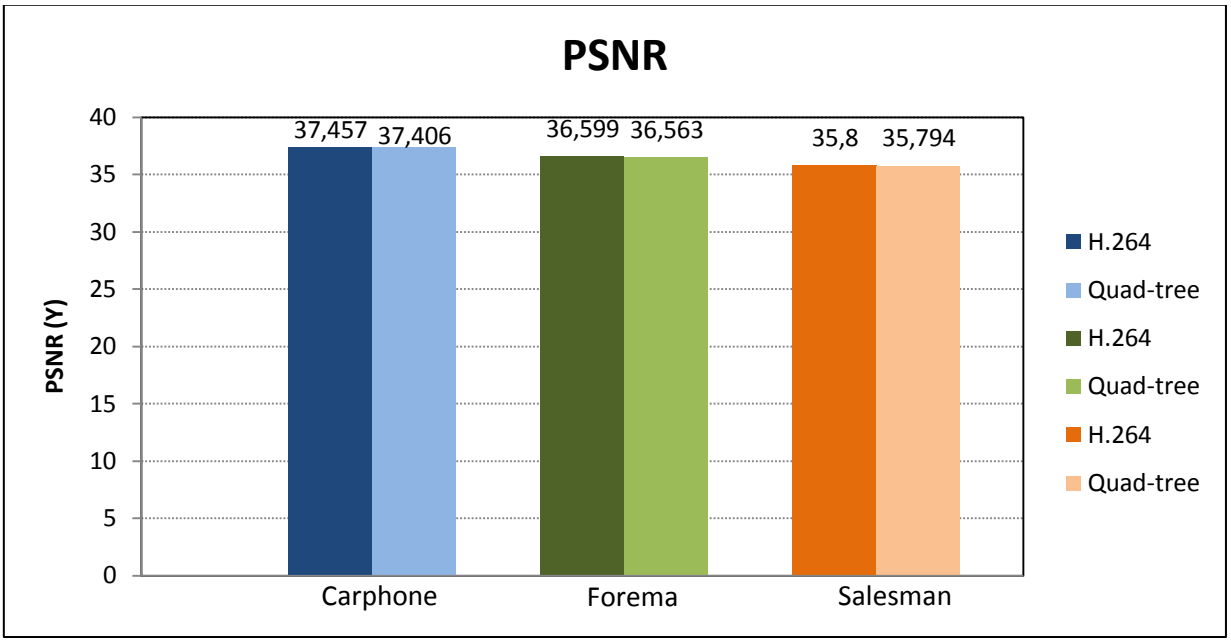


Figura 47. PSNR resultante de cada sequência.

Analisando a Figura 47, na qual são apresentados os valores de PSNR do componente de luminância gerados ao fim de cada codificação, nota-se que a redução na qualidade medida através dessa métrica objetiva é insignificante ficando em alguns casos na casa dos centésimos.

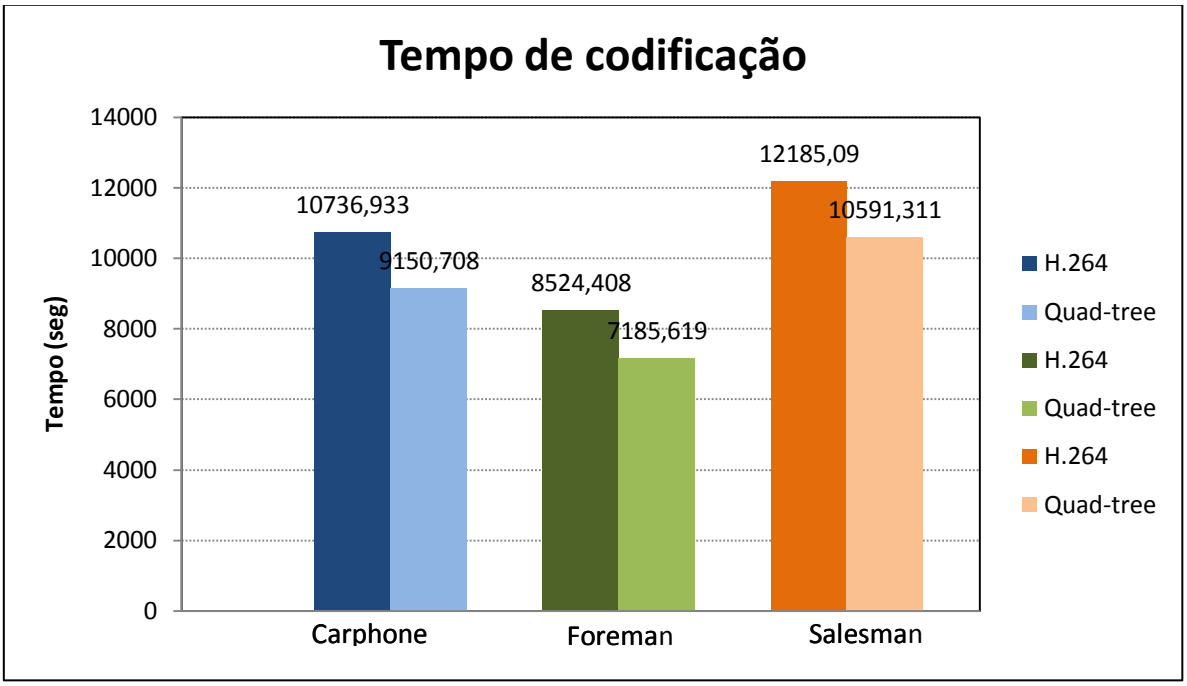


Figura 48. Tempo de codificação de cada sequência.

Dos resultados apresentados na Figura 48 é importante notar a grande redução do custo computacional através do tempo necessário para se codificar cada sequência. Com o uso da segmentação *quadtree* foi possível baixar consideravelmente o tempo de codificação das sequências, reduzindo 26 minutos de uma codificação que levaria 3 horas e 18 minutos (Salesman).

7.3 TESTES DE CODIFICAÇÀO EM DIFERENTES TAXAS DE BITS E DIFERENTES TAMANHOS DE BLOCOS

Neste tópico serão apresentados alguns testes realizados com codificações em diferentes taxas de bits considerando-se a segmentação do padrão H.264 e do padrão H.264 com abordagem em *quadtree* e também uma comparação utilizando uma

segmentação baseada somente em blocos 16 x 16, em blocos de 8 x 8 e também blocos de 4 x 4.

7.3.1 Análise de Codificação com Diferentes Taxas no Bits

As codificações utilizando diferentes taxas de bit rate foram feitas apenas para a sequência Foreman, nelas foram utilizadas as taxas de 20, 50, 100, 200 e 500 kbps, 1, 1,5 e 2Mbps. Foi utilizando tanto a segmentação *quadtree* como a segmentação do padrão H.264 seguindo o mesmo esquema de testes apresentados anteriormente.

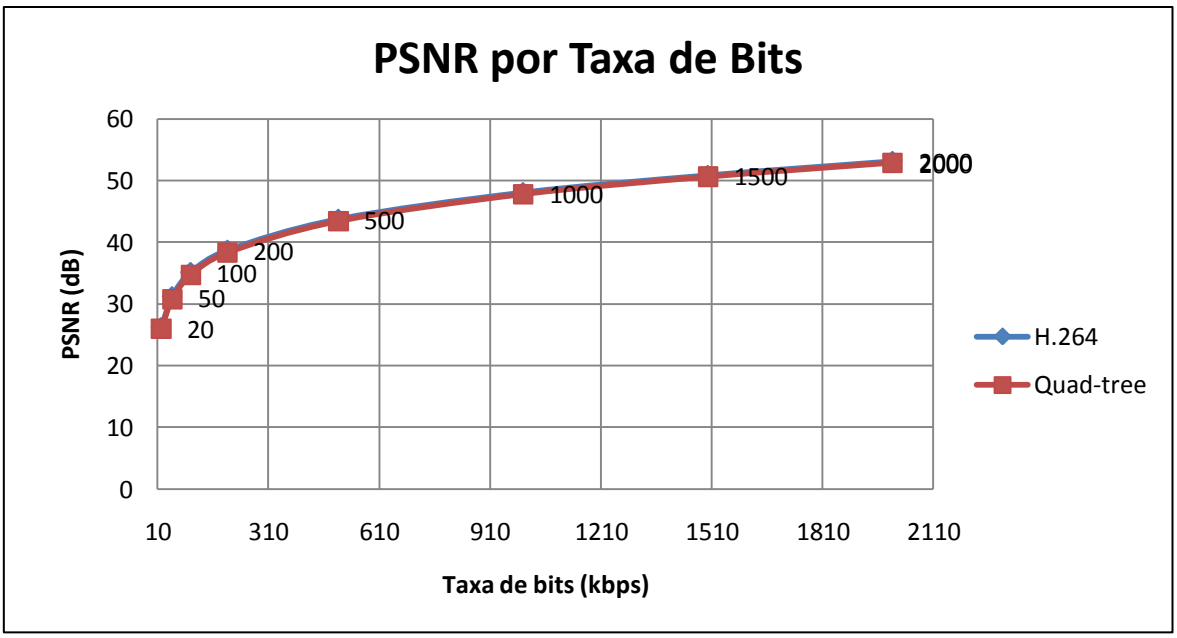


Figura 49. Resultado de PSNR para cada taxa de bits para a sequência Foreman.

Como se pode observar nos testes apresentados Figura 49, independentemente da taxa de bit rate que for definida, a PSNR resultante da codificação utilizando a segmentação *quadtree* será menor que a segmentação do H.264

más sempre quase se igualando a este valor com uma diferença insignificante e imperceptível no caso de a aplicação de uma métrica subjetiva.

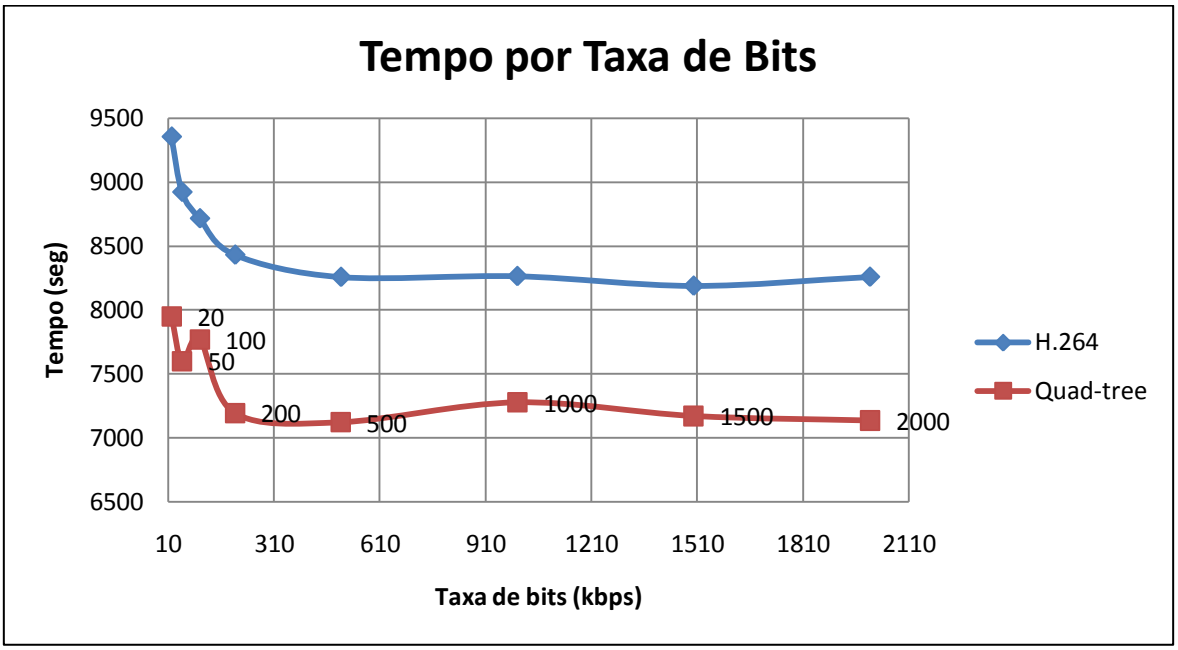


Figura 50. Tempo de codificação para cada bit rate para a sequência Foreman.

Assim como na Figura 49, a Figura 50 apresenta os mesmos resultados positivos demonstrados nos outros testes para o tempo de codificação. A redução do custo computacional é facilmente visualizada se prestarmos atenção na tamanha redução do tempo de codificação para ambas as taxas de bit rate.

7.3.2 Segmentação com Blocos de Tamanho Único

Durante o desenvolvimento do projeto, trabalhando na possibilidade de ganhos com a utilização de uma estrutura *quadtree* para segmentar um macrobloco, surgiu a dúvida de como se sairia o codificador se fosse utilizado apenas um tamanho de

macrobloco no processo de segmentação do mesmo, então optou-se por realizar testes preliminares para constatar como desempenho de uma codificação feita com tamanho único de segmentação. Para este teste foi utilizada a sequência Salesman e os tamanhos 16 x 16, 8 x 8 e 4 x 4.

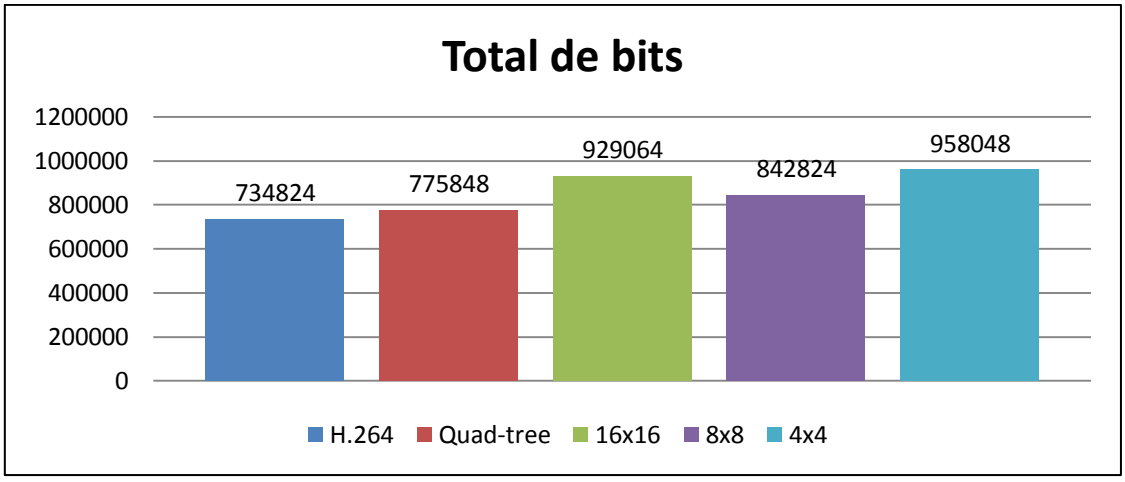


Figura 51. Total de bit para as cinco segmentações.

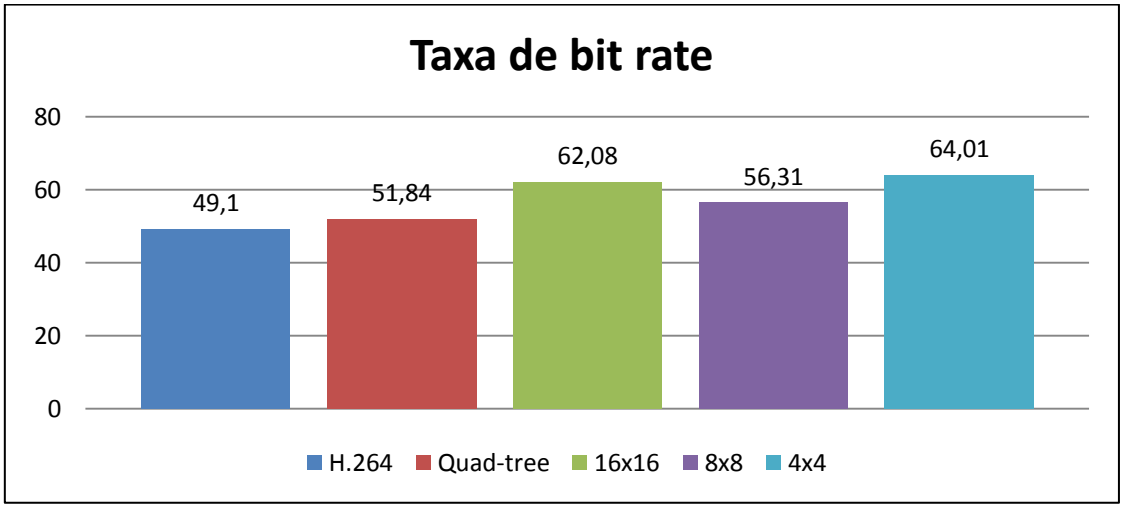


Figura 52. Taxa de bit rate para as cinco segmentações.

Analisando as Figuras 51 e 52, é possível confirmar que com o uso de blocos de tamanho único para a segmentação o codificador apresentou taxas mais

elevadas de bit rate, o que representa uma maior quantidade de dados representados por segundo, logo necessitando de mais espaço, e atrás destas vem as segmentações *quadtree* e a do padrão H.264.

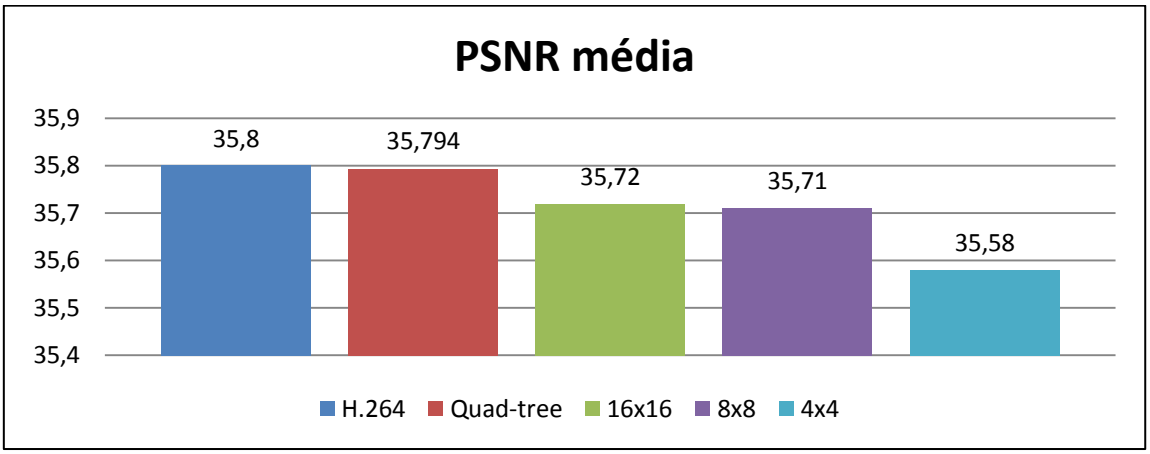


Figura 53. PSNR resultante para as cinco segmentações.

Na Figura 53 dar-se credibilidade às segmentações do padrão H.264 e a *quadtree*, que se encontrar com os valores mais elevados na medição da qualidade objetiva representada pela PSNR, seguidas pelas segmentações em blocos de tamanhos únicos.

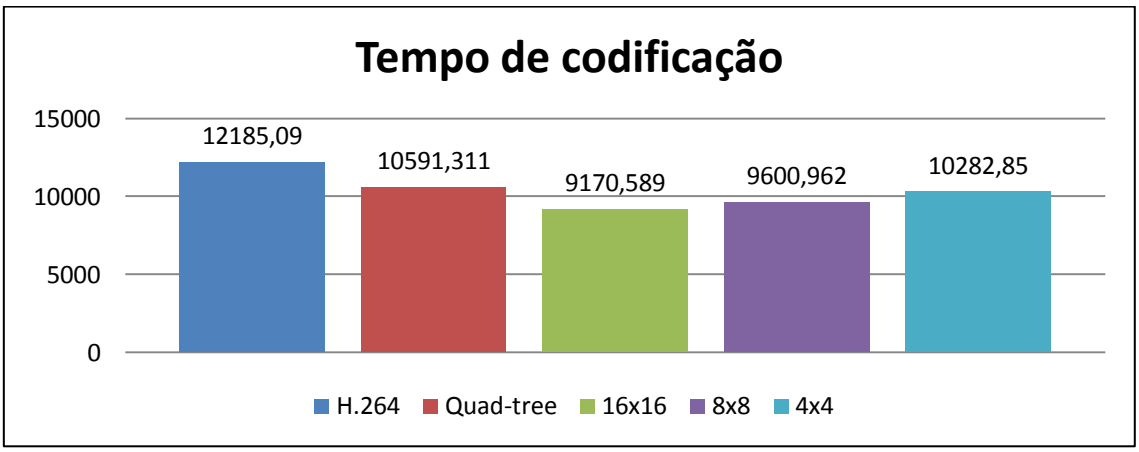


Figura 54. Tempo para a codificação das cinco segmentações.

Observando a Figura 54 pode-se notar a redução da complexidade computacional conforme a menor necessidade de se segmentar um macrobloco em tamanhos distintos. As segmentações em blocos de tamanhos únicos tornam a codificação mais rápida do que a codificação feita pela segmentação em *quadtree* que, por sua vez, é mais rápida que a segmentação do padrão H.264.

8 CONCLUSÃO

Neste trabalho, discutiram-se aspectos de segmentação para redução de complexidade computacional que podem ser utilizados em esquemas de codificação de vídeo digitais. Foi realizada uma abordagem envolvendo um breve histórico sobre vídeos digitais, principais métodos aplicados para compressão de dados e o padrão de codificação de vídeo H.264/AVC e seus modelos temporal e espacial de codificação. Especial atenção foi dada à estrutura de segmentação regular de macroblocos baseada em *quadtree*, com seus princípios e diferenças em relação à estrutura de referência usada pelo padrão H.264.

A proposta de utilizar uma estrutura baseada em *quadtree* para a segmentação dos macroblocos foi motivada pela possibilidade de redução da complexidade computacional. Esta estrutura possui apenas duas possibilidades de segmentação de macrobloco. Já o padrão H.264, possui seis possibilidades de segmentar um macrobloco. Na estrutura baseada em *quadtree*, por possuir menos opções de segmentação, são requeridos menos cálculos e comparações para a escolha da forma de segmentação dos macroblocos.

Analisando-se os resultados, constatou-se que com o uso da estrutura baseada em segmentação com *quadtree* obteve-se uma considerável redução do custo computacional, baseando-se na diminuição de até 15% do tempo de codificação. Contudo, constatou-se reduzida perda de qualidade da sequência ao final do processo de codificação/decodificação, promovendo uma baixa redução de qualidade onde a redução maior foi de 0,051 dB e a menor foi de 0,006 dB nos resultados apresentados pela métrica objetiva, sendo essa redução desconsiderável para a métrica subjetiva.

Como trabalho futuro, a fim de se obter uma redução ainda maior do custo de complexidade computacional, propõe-se um estudo avançado do processo de definição dos macroblocos *skip*, definindo-se os valores que estipulam quando um macrobloco será *skip*. Estes macroblocos *skip* não necessitam de alto custo na etapa de estimação de movimento. Desta forma, se puderem ser mais bem definidos, pode-se reduzir ainda mais o custo computacional. Propõe-se também o uso de diferentes codificadores desenvolvidos conforme o especificado pelo padrão H.264 para comparar o processo codificação/decodificação usando diferentes resoluções espaciais e temporais.

REFERÊNCIAS

AGOSTINI, L. V., **Desenvolvimento de Arquiteturas de Alta Performance Dedicadas à Compressão de Vídeo Segundo o Padrão H.264.** Tese (Doutorado) - Universidade Federal do Rio Grande do Sul. Programa de Pós-Graduação em Computação, Porto Alegre, RS, 2007.

APPLE, Computer et al. **H.264/AVC Reference Software Encoder Documentation**. 2009. Disponível em: <http://iphome.hhi.de/suehring/tml/doc/lenc/html/index.html>. Acessado em: 23 out. 2010.

AZEVEDO, A. P. **MoCHA: Arquitetura Dedicada para a Compensação de Movimento em Decodificadores de Vídeo de Alta Definição, Seguindo o Padrão H.264**. Dissertação (Mestrado em Ciência da Computação) – Instituto de Informática, UFRGS, Porto Alegre, RS, 2006.

BAKER, R; SULLIVAN, G. **Efficient Quadtree Coding of Images and Video**, Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing, 1991, pp.2661-2664.

BHASKARAN, V.; KONSTANTINIDES, K. **Image and Video Compression Standards: Algorithms and Architectures**. 2. ed. Boston: Kluwer Academic Publishers, 1997.

BRAIN, M. **Como funciona a TV digital**, 2006. Disponível em: <http://eletronicos.hsw.uol.com.br/televisao-digital.htm>. Acesso em: 28 set. 2010.

BROTHERTON, M. D. **Subjective Multimedia Quality Assessment**. 2006

CORRIVEAU, P. **All subjective scales are not created equal: the effects of context on different scales.** v.77. 1999.

CROCOMO, F. **TV digital e produção interativa: a comunidade manda notícias**. Florianópolis: UFSC, 2007.

CRUZ, R. **TV Digital no Brasil - Tecnologia Versus Política**. Rio de Janeiro: Senac Editora, 2008.

FELITTI, G. **Saiba quais são os principais formatos de vídeo digital disponíveis na web**, 2007. Disponível em <http://idgnow.uol.com.br/internet/2007/04/13/idgnoticia.2007-04-13.0813873032/paginador/pagina_2>. Acesso em: 27 set. 2010.

FREDERICK, Paula. (1999). **Visualização Eficiente de Objetos Gráficos**. Dissertação (Mestrado em Ciências em Informática), Pontifícia Universida de Católica do Rio de Janeiro, Rio de Janeiro, 1999.

GONZALEZ, R; WOODS, R. **Processamento de Imagens Digitais**. São Paulo: Editora Blucher, 2003.

HUANG, Y. **Analysis, Fast Algorithm, and VLSI Architecture Design for H.264/AVC Intra Frame Coder**. IEEE Transactions on Circuits and Systems for Video Technology,v.15, n.3, p. 378-401, mar. 2005.

HUYNH-THU, Q.; GHANBARI, M. **A Comparison of Subjective Video Quality Assessment Methods for Low-Bit Rate and Low-Resolution Video.** IASTED INTERNATIONAL CONFERENCE ON SIGNAL AND IMAGE PROCESSING, 2005, Honolulu, Hawaii, USA.

ITU-T. Recommentation P.910: **subjective video quality assessment methods for multimedia applications**. 1999.

ITU-R. Recommendation BT.500**: methodology for the subjective assessment of the quality of television pictures**. 2002.

ITU-T Recommendation H.264: **Advanced video coding for generic audiovisual services**. 2003.

ITU-T. **Recommendation H.264 and ISO/IEC 14496-10 (MPEG-4 part 10) Advanced Video Coding**, 2005.

ITU-T; ISO/IEC JVT. **H.264/AVC Reference Software Encoder version JM 14.0**. Disponível em: <http://iphome.hhi.de/suehring/tml/>. Acesso em: 16 out. 2009.

KIOSKEA. **Introdução ao vídeo digital**, 2009. Disponível em: <http://pt.kioskea.net/contents/video/video.php3>. Acesso em 01 out. 2010.

KOZAMERNIK, F. **SAMVIQ - A New EBU Methodology for Video Quality Evaluations in Multimedia.** SMPTE Motion Imaging Journal,v.114, n.4. 2005.

KUHN, P. **Algorithms, Complexity Analysis and VLSI Architectures for MPEG-4 Motion Estimation**. Boston: Kluwer Academic Publishers, 1999.

LIMA, R. **Conheça os diferentes formatos de vídeo digital**, 2010. Disponível em <http://www.superdownloads.com.br/materias/formatos-de-video.html>. Acesso em: 01 out. 2010.

LU, L; PEARLMAN, W. A. **Multi-Rate Video Coding Using Pruned Tree-Structured Vector Quantization**, Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing, 1993.

MACLEAN, G. F. and JERNIGAN, M. E. **Indicator Functions for Adaptive Image Processing**, J.Opt.Soc.Am., Vol 8, N°1, January 1991.

MEGRICH, A. **Televisão digital**. 1. Ed. São Paulo: Érica, 2009.

MOBILEASL. **Mobileasl**, 2010. Disponível em: http://mobileasl.cs.washington.edu/. Acesso em: 03 mar. 2011.

MORIMOTO, C. E. **Os formatos de compressão de vídeo**, 2009. Disponível em: < http://www.guiadohardware.net/tutoriais/formatos-compressao-video/>. Acesso em: 02 out. 2010.

MUSEUDOCOMPUTADOR. **História do dvd**, 2004. Disponível em: <http://www.museudocomputador.com.br/encidvd.php>. Acesso em: 05 out. 2010.

NATIONAL SCIENCE FOUNDATION. **MOBILEASL**, 2010. Disponível em: <http://mobileasl.cs.washington.edu/>. Acesso em: 15 mar. 2011.

POLLACK, J. **Displays of a Different Stripe**. IEEE Spectrum. v. 43. Aug. 2006.

PORTO, R. **Desenvolvimento Arquitetural para Estimação de Movimento de Blocos de Tamanhos Variáveis Segundo o Padrão H.264/AVC de Compressão de Vídeo Digital**. Dissertação (Mestrado em Ciência da Computação) – Instituto de Informática, UFRGS, Porto Alegre, RS, 2008.

PEIXOTO, E. **Transcodificador de vídeo Wyner-Ziv/H.263 para comunicação entre dispositivos móveis**. Dissertação (Mestrado) — Universidade de Brasília, 2008.

PURI, A. **Video Coding Using the H.264/MPEG-4 AVC Compression Standard. Elsevier Signal Processing: Image Communication**, p.793–849, 2004.

RIBEIRO, N.; TORRES, J. **Tecnologias de Compressão Multimédia**. 1. Ed. Lisboa: FCA - Editora de Informática, 2009.

RICHARDSON, I. **H.264/AVC and MPEG-4 Video Compression - Video Coding for Next-Generation Multimedia**. Chichester: John Wiley and Sons, 2003.

RICHARDSON, I. **Video Codec Design - Developing Image and Video Compression Systems**. Chichester: John Wiley and Sons, 2002.

SALOMON, D. **Data compression: the complete reference**. 3rd ed California: Springer, 2004.

SANADA, V.; SANADA, Y. **Vídeo digital: a compra da câmera, edição das imagens e produção de vídeos digitais para DVD, TV e cinema digital**.Rio de Janeiro: Axcel Books, 2004.

SAYOOD, K. **Introduction to Data Compression.** 3. Ed. Morgan Kuffmann Publishers,
2000.

STROBACH, P. **Quadtree Structured Recursive Plane Decomposition Coding of Images.** IEEE Trans. Signal Procesing, vol.39, 1991.

SULLIVAN, G. J. TOPIWALA, P. LUTHRA, A. **The H.264/AVC Advanced Video Coding Standard: Overview and Introduction to the Fidelity Range Extensions**. Agosto, 2004.

TORRES, G. **DVD**, 1998. Disponível em: <http://www.clubedohardware.com.br/artigos/365>. Acesso em 05 out. 2010.

TORRES, G. **Tudo sobre HDTV**, 2005. Disponível em: <http://www.clubedohardware.com.br/artigos/1011>. Acesso em: 05 out. 2010.

VICTOR, L. **A Beginners Guide for MPEG-2 Standardn** 2011. Disponível em: <http://www.fh-friedberg.de/fachbereiche/e2/telekom-labor/zinke/mk/mpeg2beg/beginnzi.htm>. Acessado em: 03 mai. 2011.

WIEGAND, T. **Overview of the H.264/AVC Video Coding Standard**. IEEE Transactions on circuits and systems for video technology, vol. 13, no. 7. 2003.

WIKIPEDIA. **Televisão Digital no Brasil**, 2010. Disponível em: <http://http://pt.wikipedia.org/wiki/Televisao_digital_no_Brasil>. Acesso em: 20 nov. 2010.

WIKIPEDIA. **Vídeo Digital**, 2004. Disponível em: <http://pt.wikipedia.org/wiki/video_digital>. Acesso em: 20 nov. 2010.

APÊNDICE A – ARTIGO CIÊNTÍFICO

Segmentação Quadtree para Seleção de Blocos na Codificação de Vídeo H.264/AVC

Tiago Pizzetti Medeiros

Curso de Ciência da Computação
Universidade do Extremo Sul Catarinense (UNESC) – Criciúma, SC – Brazil

tiagopizzetti@gmail.com

Abstract. *Because the digital videos are showing increasingly better quality, they are requiring more storage and more bandwidth for its transmission. Before this be improving two areas is of utmost importance to enhance the methods of current video encoding, giving greater or equal quality with reduced computational cost. Using a quadtree structure in the segmentation of macroblocks during encoding, the computational cost will be reduced and the quality loss will be close to zero.*

Resumo. *Pelo fato de os vídeos digitais estarem apresentando cada vez mais uma melhor qualidade, eles estão necessitando de mais espaço para seu armazenamento e de uma maior banda para a sua transmissão. Antes de se estar melhorando essa duas áreas, é de suma importância que se aprimore os métodos de codificação de vídeo atuais, dando maior ou igual qualidade com redução de custo computacional. Utilizando uma estrutura quadtree na segmentação dos macroblocos durante a codificação, o custo computacional será reduzido e a perda de qualidade será próxima à zero.*

1. Introdução

Na última década, a tecnologia tem apresentado grandes evoluções, tendo em vista as altas qualidades alcançadas em vídeo e imagem. Esse grande aumento na qualidade tem tido um custo altíssimo aos hardwares na parte de transmissão de dados. Organizações formadas por grandes empresas do setor tecnológico vêm constantemente aprimorando os padrões de mídia e de comunicação de dados para que os mesmos atendam à demanda exigida hoje pelos arquivos digitais.

H.264 [ITU-T, 2005] é um padrão para compressão de vídeo, baseado no MPEG-4 Part 10/AVC (Advanced Video Coding). Uma das principais metas era fazer com que o padrão fosse capaz de fornecer uma boa qualidade de vídeo com uma taxa de bits muito baixa em relação aos padrões já existentes.

O MPEG-4 é utilizado primeiramente para compressão de dados digitais de áudio e vídeo e engloba um grande grupo de padrões para a codificação desses tipos de informações digitais. Ele é um padrão em constante desenvolvimento e divide-se em várias partes, as principais são: parte 2 (incluindo ASP, usado por codecs como DivX e Xvid) e a parte 10 (AVC/H.264, usado pelos codecs x264 e também no padrão de codificação para TVs Digitais no Brasil) [RICHARDSON, 2003].

O quadro de um vídeo digital no padrão H.264/AVC, é dividido em macroblocos com tamanhos de 16x16, 16x8, 8x16, 8x8 [PURI, 2004]. A escolha do tamanho da partição

depende de alguns fatores, porém nas áreas do quadro do vídeo digital onde há mais movimento o tamanho de partição escolhido será menor que aqueles escolhidos para áreas onde se tem movimento reduzido.

Entretanto, o esforço computacional envolvido para seleção das áreas a serem codificadas a partir das imagens de diferença a serem codificadas e da seleção dos vetores de deslocamento no processo de estimação de movimento é significativo.

Uma técnica recente que exibe grande potencial para a representação de imagens em diferentes níveis de resolução é a decomposição em quadtree (QT). A QT é uma estrutura de dados hierárquica que possibilita a segmentação de uma imagem em regiões bidimensionais homogêneas. A homogeneidade é definida com respeito a uma dada propriedade de interesse em blocos com diferentes níveis de resolução (dimensões). A QT origina uma árvore, cujos nós dão origem a quatro blocos filho (sub-blocos). A cada particionamento, o tamanho dos sub-blocos resultantes é a quarta parte do seu predecessor. Cada nó, por sua posição na árvore, corresponde a um sub-bloco que é único em tamanho e posição dentro da imagem. O nó raiz, que está associado à imagem completa, deve ter dimensões de potência de 2 para possibilitar a aplicação recursiva do algoritmo de decomposição.

A proposta deste projeto visa o uso de uma estrutura de seleção de regiões a serem codificadas e seleção dos vetores de deslocamento baseando-se em uma estrutura de decomposição QT para reduzir a complexidade computacional nos onerosos processos de codificação.

Os caminhos que serão seguidos por este trabalho estão voltados à descrever e documentar o funcionamento do padrão de codificação de vídeo digital H.264/AVC como também a estrutura de segmentação quadtree e trazer a conhecimento as principais métricas aplicadas aos vídeos digitais.

2. Compressão de Vídeo

Um vídeo é composto por várias imagens projetadas em sequência, cada imagem desta sequência é chamada de quadro e a quantidade de imagens que são projetadas por segundo chamada-se cadência, a medida utilizada na cadência é o fps (frames por segundo). Quanto mais frames/quadros por segundo tiver um vídeos mais realista e maior será a qualidade da imagem. Os vídeos digitais normalmente trabalham com o mesmo fps das TV, que é de 30 frames por segundo [SALOMON, 2004).

Para trabalhar os vídeos e obter um resultado final satisfatório, os codificadores de vídeos costumam dividir os quadros em macroblocos, principal unidade de codificação. Esses MBs possuem 16 amostras de largura e 16 amostras de altura (16 x 16) e subdividem-se conforme a movimentação entre os quadros em blocos de 16 x 8, 8 x 8, 8 x 4 e 4 x 4. Podemos observar na Figura 1 a divisão de um frame em vários blocos:

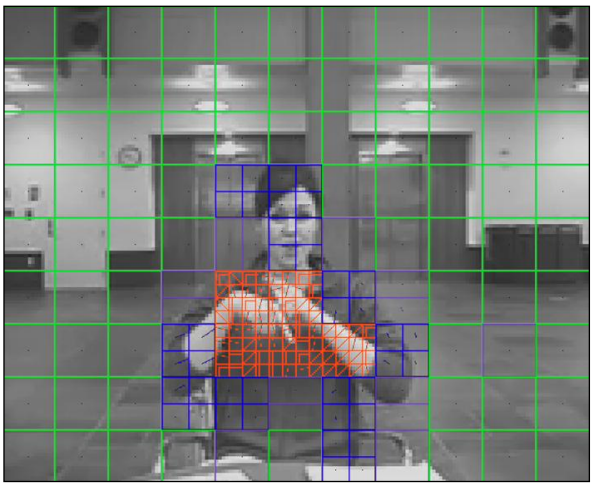


Figura 55. Divisão de um frame em macroblocos.

Uma dos métodos utilizados para diminuir o tamanho do vídeo é justamente reduzindo a quantidade de quadros por segundo que além de reduzir o tamanho do vídeo, consequentemente reduzirá a qualidade quanto ao realismo do vídeo, pois haverá quebra de quadro, isto é, os movimentos do vídeo ficarão "truncados".

Utiliza-se também para reduzir o tamanho dos vídeos, uma técnica de compressão de imagem, que através da analise dos macroblocos possibilita remover das imagens informações que já foram projetadas. Um exemplo seria o de um vídeo onde se tem uma pessoa parada apenas falando, no quadro inicial a imagem é projetada por completa, mas nos quadros seguintes os pedaços da imagem que são idênticos ao quadro anterior são removidos atualizando apenas a área em que há movimento, se só a boca esta se movimentando, somente a área da boca será desenhada nos quadros seguintes. Pelo fato de mostrar apenas o primeiro quadro por completo, esta técnica economiza uma quantidade enorme de espaço, os demais quadros só mostram o que muda em relação ao quadro anterior. Esses quadros incompletos são chamados quadros delta (delta frames) [RIBEIRO e TORRES, 2009)]

3. Modelo Temporal

O objetivo do modelo temporal é explorar a redundância entre os quadros prevendo um quadro através de outro quadro anteriormente codificado (quadro de referência). Encontra-se a diferença entre ambos os quadros, resultando em um resíduo. O resíduo é codificado e enviado ao decodificador, que então adiciona o resíduo do quadro de referência. Na maior parte dos casos utiliza-se apenas o quadro anterior como referência, mas também pode-se usar um ou mais quadros passados ou futuros para se prever os quadros seguintes. Todo este processo de exploração da redundância existente entre os quadros chama-se predição Inter-quadros.

Na predição Inter, os quadros são codificados um de cada vez e a exploração da redundância é feita pela diferença do quadro atual com uma versão predita do quadro atual. A diferença encontrada é chamada de resíduo ou erro de predição. A predição do quadro atual consiste em buscar em outros quadros anteriormente codificados e reconstruídos (decodificados localmente), informações que permitam reduzir a energia do resíduo. Os quadros usados para a geração do quadro predito são denominados quadros de referência. Do lado do decodificador, o quadro predito é recriado e somado ao resíduo gerando o quadro atual decodificado [RICHARDSON, 2003].

Segundo Wiegand (2003) a principal maneira de se fazer a predição é utilizando o

quadro imediatamente anterior como predito do quadro atual e obtendo o resíduo através da diferença entre esses dois quadros. A Figura 2 mostra os dois primeiros quadros da sequência "Foreman" e a diferença gerada pelos dois. No entanto, mesmo os dois quadros pertencendo à mesma cena do vídeo (sem nenhuma mudança bruta de cena), o resíduo gerado apresenta uma quantidade de energia elevada nas regiões de movimento, seja movimento de câmera ou de objetos. Uma boa forma de predição pode ser feita apenas compensando-se o movimento existente na cena. O quadro predito não precisa obrigatoriamente ser um dos quadros anteriores por inteiro, ele pode ser uma composição dos mesmos. Esta composição é chamada justamente de Compensação de Movimento (MC). Porém, é necessário que se determine os movimentos no quadro atual e isto é feito pela Estimação de Movimento (ME).

O módulo de ME, foi implantado apenas no codificador do padrão H.264/AVC, que junto com a MC, formam a predição inter-quadros. É na predição inter onde se encontra a maior complexidade computacional de um codificador H.264/AVC [PURI, 2004]. Este custo computacional é muito elevado devido à várias ferramentas novas e inovadoras que foram implantadas nesta etapa de codificação. Devido a estas diversas ferramentas, é neste módulo onde se encontra o maior potencial de ganhos do H.264/AVC, em termos de compressão, em relação aos padrões anteriores de compressão de vídeos [WIEGAND, 2003].

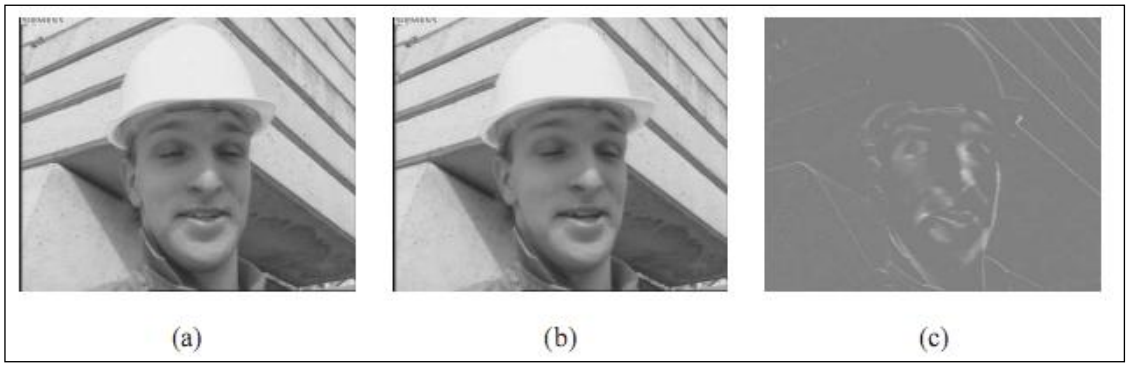


Figura 2. Sequência Foreman: (a) Quadro 1; (b) Quadro 2; (c) Diferença entre Quadros 1 e 2.

3.1 Estimação de Movimento

O objetivo da ME é encontrar, nos quadros de referência, qual o bloco se assemelha mais com o bloco do quadro atual. Esta busca é realizada em uma determinada área ao redor do bloco que esteja em análise, esta área denomina-se área de pesquisa. Assim que o melhor casamento é encontrado, a ME gera um vetor de movimento (MV – Motion Vector) que indica qual é o deslocamento da posição bloco do quadro atual para o bloco do quadro de referência. O valor do MV é enviado junto à codificação do macrobloco no bitstream. A Figura 3 demonstra o processo de ME [PORTO, 2008].

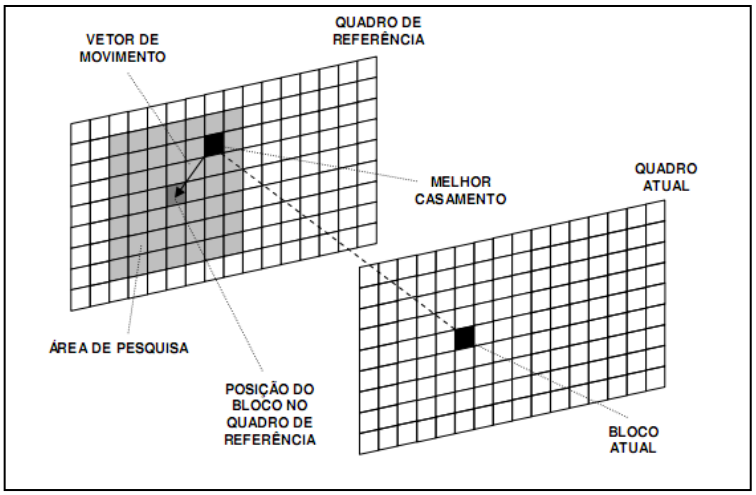


Figura 3. Determinação do vetor de movimento de um bloco na ME.

Geralmente os deslocamentos ocorrem pela movimentação de algum objeto ou pela movimentação da câmera em relação à cena observada. Desta maneira, no caso da Figura 5, de dois quadros subsequentes, determinada região de um quadro, se deslocou para outra região em outro quadro.

No entanto, há a possibilidade de que a percepção de um objeto, por exemplo, nos dois quadros seja diferente, seja por mudança na iluminação do objeto, deslocamento do próprio objeto, deslocamento da câmera ou até mesmo deslocamento de um outro objeto sobre o outro. Usando o primeiro quadro como referência, pode-se fazer uma busca do objeto presente no segundo quadro e determinar o quanto ele se deslocou. Essa diferença entre as posições do objeto nos dois quadros é o MV, e o conjunto dos MVs de um quadro é chamado de fluxo óptico [RIBEIRO e TORRES, 2009].

Durante o desenvolvimento de padrões de codificação, pode-se observar que a ME de objetos inteiros pode em alguns casos, ser bastante complexa, principalmente porque pode ser bastante difícil determinar qual parte do quadro pertence a um objeto qualquer. Outra ideia seria fazer a busca de cada pixel individualmente, o que permitiria um MV exato para cada pixel, porém o custo computacional de fazer buscas para todos os pixels é extremamente alto. Decidiu-se então por fazer a busca para blocos de pixels. Esses blocos são de tamanho 16x16 pixels e são chamados de macroblocos [PEIXOTO, 2008].

A ME é feita então bloco a bloco, buscando nos quadros de referência o bloco mais parecido possível com o bloco atual. Para encontrar o bloco de referência que tem uma melhor relação com o atual, usa-se técnicas baseadas na diferença entre os blocos. As técnicas mais comuns são a Soma das Diferenças Absolutas (SAD - Sum of Absolute Differences) e a Soma da Diferença Quadrática (SSD - Sum of Squared Differences) [MEGRICH, 2009].

Segundo Richardson (2003) a estimação de movimento é aplicada apenas ao componente de luminância do macrobloco. Pelo fato de os componentes de crominância possuírem metade da resolução horizontal e vertical da componente de luminância, basta que seja feito uma divisão por dois nos componentes horizontal e vertical de cada vetor de movimento de luminância para serem aplicadas aos blocos de crominância.

4. Segmentação no Padrão H.264

O padrão H.264 trouxe consigo um avanço muito considerável que é a possibilidade de subdividir o macrobloco em blocos menores de pixels. Essa partição chamada de

Compensação de Movimento em Estrutura de Árvore. Ela guarda certa semelhança com a estrutura quadtree (mostrada na Figura 4(a)). Nas áreas de processamento de imagens e vídeos e computação gráfica, a estrutura quadtree consiste na subdivisão de blocos de pixels em quatro quadrantes. Cada quadrante pode ser novamente dividido em quadrantes e assim por diante, daí o nome que se refere a uma árvore de quadrantes. Esta estrutura pode ser usada para uma série de finalidades, incluindo compressão e decomposição de imagens para classificação de regiões [ITU-T, 2003].

A principal diferença entre a estrutura de quadtree e a estrutura usada no H.264 é que, apesar de no H.264 também consistir uma estrutura de árvore, os blocos não precisam ser divididos em quadrantes, podendo ser quebrados em pares de blocos retangulares, o que de certa forma permite maior flexibilidade. Desta forma, um macrobloco pode ser dividido em dois blocos de 16x8 pixels, dois blocos de 8x16 pixels ou quatro blocos de 8x8 pixels. Os blocos de 8x8 pixels podem ser divididos em sub-blocos de 8x4 pixels, 4x8 pixels ou 4x4 pixels, como pode ser visto na Figura 4(b).

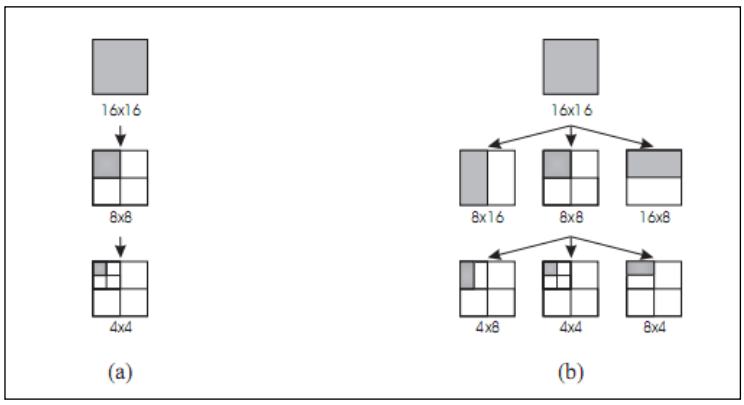


Figura 4. Estruturas em Árvore para Partição de Macrobloco: (a) Quad-Tree; (b) H.264.

Quando o macrobloco é dividido, cada região originada pode ter um MV diferente, seguindo o mesmo princípio que levou o quadro a ser dividido em macroblocos, dentro de um macrobloco, várias regiões podem guardar algumas semelhanças das partes do quadro de referência que não pertençam a um mesmo macrobloco. Apesar de serem necessários mais bits para esses MVs, diferentemente dos poucos usados para apenas um vetor no caso do macrobloco completo, o ganho que se obtém pela redução do resíduo de cada partição acaba por ser vantajoso.

A decisão se há ou não vantagem em dividir o macrobloco depende da quantidade de bits necessários para armazenar tanto os resíduos quanto os MVs. Somando-se esses bits aos demais resultantes da codificação, tem-se a taxa referente ao macrobloco. Outra avaliação para decidir dividir ou não está na distorção. O custo de um macrobloco é determinado em função dessas duas métricas.

Conforme especificado pela ITU-T (2003) para encontrar a melhor partição, o macrobloco é classificado em modos: 16x16, 16x8, 8x16, 8x8. O codificador verifica quanto será o custo para cada modo. Somente se o modo 8x8 for de menor custo, os blocos serão divididos e classificados em sub-modos: 8x4, 4x8, 4x4.

5. Segmentação Quadtree

A quadtree é uma estrutura de dados hierárquica, possui ótima eficiência na descrição de regiões bidimensionais. Não é uma estrutura nova, primeiramente foi utilizada para armazenas figuras "binarizadas" (apenas pixels 0 e 1), depois por codificadores de imagens com tons de cinza e atualmente codificadores de vídeo que a utilizam em conjunto com a discrete cosine transform (DCT) foram propostos [SALOMON, 2004].

Na Figura 5 há um exemplo da segmentação de um frame em quadtree baseado na diferença entre este frame e o frame anterior compensado. Primeiro a imagem foi dividida em blocos de 32x32 pixels de tal forma que se adequasse as dimensões da imagem, que no caso apresentado é uma imagem no padrão CIF com 352x288 pixels, uma das exigências da decomposição regular da quadtree, é que as dimensões do nó raiz sejam potências de 2, de modo a permitir a aplicação recursiva do algoritmo. Desta forma cada bloco de 32x32 pixels da imagem apresentada é um nó raiz que dará origem a uma quadtree cujas folhas foram definidas como sendo de 4x4 pixels.

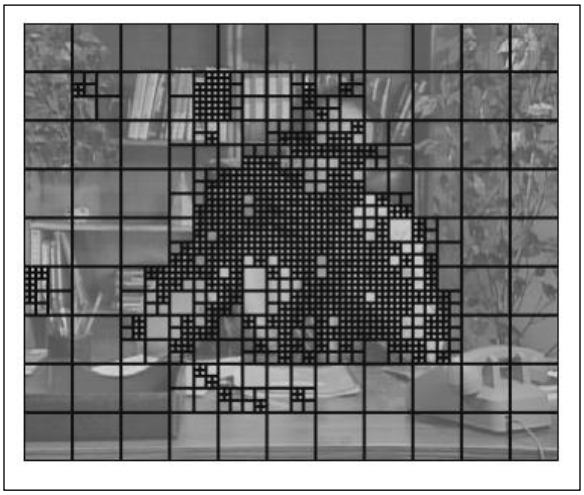


Figura 5. Composição real em quadtree.

6. H.264/AVC JM Reference Software 17.2

O JM é um código de referência do padrão H.264/AVC que engloba em si todas as especificações detalhadas na norma deste padrão. Ele é formado por um codificador chamado lencod e por um decodificador chamado ldecod. Neste projeto foi trabalhado apenas com o codificador.

Foi utilizada a versão 17.2, lançada em 27 de maio de 2010, atualmente encontra-se na versão 18.0, lançada em 03 de maio de 2011. O codificador lencod possui vários parâmetros de entrada, todos especificados em um arquivo texto no mesmo diretório do arquivo lencod.exe. Dependendo do perfil desejado, existem arquivos de texto já definidos para codificar um vídeo com as especificações recomendadas pelo padrão h.264, são eles:

a) "encoder_extended.cfg": perfil *extended*;

b) "encoder_main.cfg": perfil *main;*

c) "encoder_baseline.cfg": perfil *baseline*;

d) “encoder.cfg”: Perfil *high*;

e) “encoder_yuv422.cfg”: perfil *High 4:2:2*;

f) “encoder_tonemapping.cfg”: perfil *High 4:4:4*.

Neste projeto foi usado o arquivo pré definido “encoder_baseline.cfg” que especifica as ferramentas utilizadas pelo perfil baseline, muito recomendado para aplicações de vídeo conferência.

7. Abordagem por Meio da Estrutura Quadtree

Assim como especificado no padrão H.264/AVC, o JM tem desenvolvido em sua estrutura todas as possibilidades de segmentação de macroblocos utilizando tanto blocos retangulares como blocos regulares. Sabe-se que a segmentação abordada no padrão H.264 possibilita a subdivisão de um macrobloco em blocos e sub-blocos retangulares e a segmentação baseada na abordagem em H.264 com quadree possibilita a subdivisão de um macrobloco em blocos e sub-blocos retangulares regulares.

Alterando-se as configurações dos parâmetros de entrada do codificador pode-se modificar a estrutura de referência JM para uma estrutura baseada em quadtree para segmentar os macroblocos na predição inter. Manipulando-se o arquivo de configuração do lencod (“encoder_baseline.cfg”) onde são definidos os parâmetros de entrada e de codificação, constatou-se a possibilidade de desativar as opções de segmentação em blocos retangulares (16 x 8, 8 x 16, 8 x 4 e 4 x8) e deixar apenas as segmentações de blocos regulares (16 x 16, 8 x 8 e 4 x 4), estes por sua vez que compõem uma quatree. Neste mesmo arquivo é definida a taxa de bits da codificação. A Figura 6 mostra as linhas alteradas do arquivo de configuração do lencod.

```
###############################################################################
# PSlice Mode types
###############################################################################
PSliceSkip              = 1   # P-Slice Skip mode consideration  (0=disable, 1=enable)
PSliceSearch16x16       = 1   # P-Slice Inter block search 16x16 (0=disable, 1=enable)
PSliceSearch16x8        = 0   # P-Slice Inter block search 16x8  (0=disable, 1=enable)
PSliceSearch8x16        = 0   # P-Slice Inter block search  8x16 (0=disable, 1=enable)
PSliceSearch8x8         = 1   # P-Slice Inter block search  8x8  (0=disable, 1=enable)
PSliceSearch8x4         = 0   # P-Slice Inter block search  8x4  (0=disable, 1=enable)
PSliceSearch4x8         = 0   # P-Slice Inter block search  4x8  (0=disable, 1=enable)
PSliceSearch4x4         = 1   # P-Slice Inter block search  4x4  (0=disable, 1=enable)
```

Figura 6. Alteração no método de segmentação.

8. Resultados e Análises

Aqui serão apresentados, comparados e discutidos os resultados obtidos com os testes de codificação realizados. Os resultados serão apresentados por meio de gráficos de taxa distorção, utilizando a PSNR como medida de qualidade de reconstrução e tempo de codificação em segundos. Os resultados têm por base os valores do componente de luminância (Y). Os testes realizados são para os três componentes YUV, mas como os componentes UV dependem do componente Y e o sinal de brilho (Y) possui mais detalhes, a sua análise tornou-se mais importante.

Os testes foram realizados codificandos todos os frames de cada sequência (Carphone 382 frames, Foreman 300 frames e Salesman 449 frames) a uma taxa de 30fps com no máximo 5 frames de referência para a codificação.

A seguir serão apresentadas as médias de tempo de codificação e de PSNR gerados ao final de cada codificação. Cada imagem constitui o gráfico com o resultado das três sequências, comparando o resultado da codificação feita com a segmentação do padrão H.264 com a segmentação quadtree.

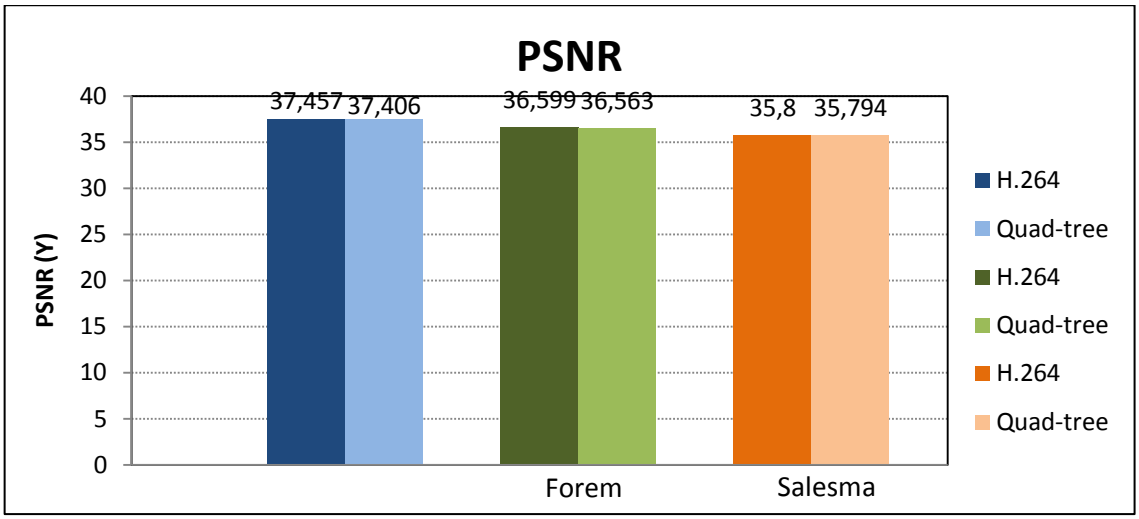


Figura 7. PSNR resultante de cada sequência.

Analisando a Figura 7, na qual são apresentados os valores de PSNR do componente de luminância gerados ao fim de cada codificação, nota-se que a redução na qualidade medida através dessa métrica objetiva é insignificante ficando em alguns casos na casa dos centésimos.

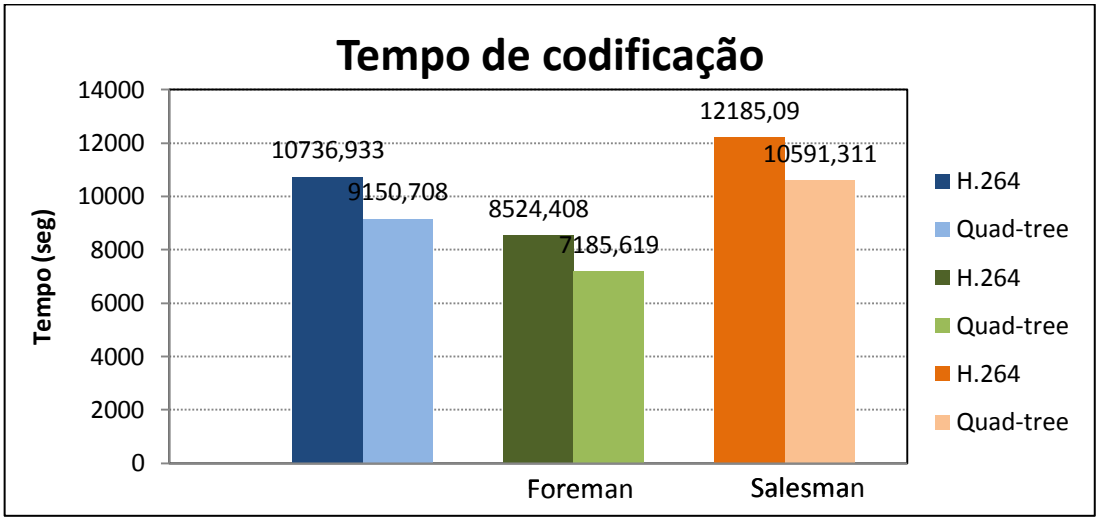


Figura 8. Tempo de codificação de cada sequência.

Dos resultados apresentados na Figura 8 é importante notar a grande redução do custo computacional através do tempo necessário para se codificar cada sequência. Com o uso da segmentação quadtree foi possível baixar consideravelmente o tempo de codificação das sequências, reduzindo 26 minutos de uma codificação que levaria 3 horas e 18 minutos (Salesman).

9. Conclusão

Neste trabalho, discutiram-se aspectos de segmentação para redução de complexidade computacional que podem ser utilizados em esquemas de codificação de vídeo digitais. Foi realizada uma abordagem envolvendo um breve histórico sobre vídeos digitais, principais métodos aplicados para compressão de dados e o padrão de codificação de vídeo H.264/AVC e seu modelo temporal de codificação. Especial atenção foi dada à estrutura de segmentação regular de macroblocos baseada em quadtree, com seus princípios e diferenças em relação à estrutura de referência usada pelo padrão H.264.

A proposta de utilizar uma estrutura baseada em quadtree para a segmentação dos macroblocos foi motivada pela possibilidade de redução da complexidade computacional. Esta estrutura possui apenas duas possibilidades de segmentação de macrobloco. Já o padrão H.264, possui seis possibilidades de segmentar um macrobloco. Na estrutura baseada em quadtree, por possuir menos opções de segmentação, são requeridos menos cálculos e comparações para a escolha da forma de segmentação dos macroblocos.

Analisando-se os resultados, constatou-se que com o uso da estrutura baseada em segmentação com quadtree obteve-se uma considerável redução do custo computacional, baseando-se na diminuição de até 15% do tempo de codificação. Contudo, constatou-se reduzida perda de qualidade da sequência ao final do processo de codificação/decodificação, promovendo uma baixa redução de qualidade onde a redução maior foi de 0,051 dB e a menor foi de 0,006 dB nos resultados apresentados pela métrica objetiva, sendo essa redução desconsiderável para a métrica subjetiva.

Referências

ITU-T. (2003) “Recommendation H.264: Advanced video coding for generic audiovisual services”.

ITU-T. Recommendation H.264 and ISO/IEC 14496-10 (MPEG-4 part 10) Advanced Video Coding, 2005.

Megrich, A. (2009) “Televisão digital”. 1. Ed. São Paulo: Érica.

Peixoto, E. (2008) “Transcodificador de vídeo Wyner-Ziv/H.263 para comunicação entre dispositivos móveis”. Dissertação (Mestrado) — Universidade de Brasília.

Porto, R. (2008) “Desenvolvimento Arquitetural para Estimação de Movimento de Blocos de Tamanhos Variáveis Segundo o Padrão H.264/AVC de Compressão de Vídeo Digital”. Dissertação (Mestrado em Ciência da Computação) – Instituto de Informática, UFRGS, Porto Alegre, RS.

Puri, A. (2004) “Video Coding Using the H.264/MPEG-4 AVC Compression Standard”. Elsevier Signal Processing: Image Communication, p.793–849.

Ribeiro, N.; Torres, J. (2009) “Tecnologias de Compressão Multimédia”. 1. Ed. Lisboa: FCA - Editora de Informática.

Richardson, I. (2003) “H.264/AVC and MPEG-4 Video Compression - Video Coding for Next-Generation Multimedia”. Chichester: John Wiley and Sons.

Salomon, D. (2004) “Data compression: the complete reference”. 3rd ed California: Springer.

Wiegand, T. (2003) “Overview of the H.264/AVC Video Coding Standard”. IEEE Transactions on circuits and systems for video technology, vol. 13, no. 7.