Universidade de Brasília
Instituto de Ciências Exatas
Departamento de Ciência da Computação

Codificação de Vídeo utilizando modelos de texturas

Juan Jesus Cruz Cartajena

Documento apresentado como requisito parcial
para a conclusão do Mestrado em Informática

Orientador
Prof. Dr. Ricardo Lopes de Queiroz

Coorientador
Prof. Dr. Bruno Luiggi Macchiavello Espinoza

Brasília
2013

Universidade de Brasília — UnB
Instituto de Ciências Exatas
Departamento de Ciência da Computação
Mestrado em Informática

Coordenador: Prof. Dr. Alba Cristina Magalhães de Melo

Banca examinadora composta por:

Prof. Dr. Ricardo Lopes de Queiroz (Orientador) — CIC/UnB
Prof. Dr. Flávio de Barros Vidal — CIC/UnB
Prof. Dr. Diogo Caetano Garcia — FGA/UNB

CIP — Catalogação Internacional na Publicação

Cruz Cartajena, Juan Jesus.

Codificação de Vídeo utilizando modelos de texturas / Juan Jesus Cruz Cartajena. Brasília : UnB, 2013.
67 p. : il. ; 29,5 cm.

Dissertação (Mestrado) — Universidade de Brasília, Brasília, 2013.

1. Compressão de Vídeo, 2. H.264/AVC, 3. textura, 4. segmentação, 5. *warping*, 6. síntese.

CDU 004

Endereço: Universidade de Brasília
Campus Universitário Darcy Ribeiro — Asa Norte
CEP 70910-900
Brasília–DF — Brasil

Universidade de Brasília
Instituto de Ciências Exatas
Departamento de Ciência da Computação

Codificação de Vídeo utilizando modelos de texturas

Juan Jesus Cruz Cartajena

Documento apresentada como requisito parcial
para conclusão do Mestrado em Informática

Prof. Dr. Ricardo Lopes de Queiroz (Orientador)
CIC/UnB

Prof. Dr. Flávio de Barros Vidal
CIC/UnB

Prof. Dr. Diogo Caetano Garcia
FGA/UNB

Prof. Dr. Alba Cristina Magalhães de Melo
Coordenador do Mestrado em Informática

Brasília, 15 de Agosto de 2013

Resumo

O esquema de codificação de vídeo proposto nesta tese é baseado em *warping* e síntese de texturas. Em vez de utilizar técnicas de codificação de imagens inteiras ou resíduos da predição depois da estimação de movimento, são utilizados modelos de movimento para reconstruir texturas que sejam classificadas como estáticas e síntese de textura para criar texturas dinâmicas. As regiões texturizadas são segmentadas pelo método de *watershed* e logo classificadas por métodos simples com base em suas características espectrais mediante a transformada DCT e simples diferenças entre quadros vizinhos. As regiões reconstruídas por *warping* e síntese são avaliadas para seu uso mediante simples métodos estatísticos. O esquema proposto trabalha em conjunto com o padrão de vídeo H.264/AVC, onde só são codificados os blocos que não sejam processados pelos modelos de textura, mostrando um potencial para diminuir o custo computacional da codificação dos blocos remanescentes. Os resultados mostram que a qualidade foi mantida com pontuações subjetivas em comparação ao padrão H.264/AVC enquanto resultados com métricas objetivas mostram pequenas perdas.

Palavras-chave: Compressão de Vídeo, H.264/AVC, textura, segmentação, *warping*, síntese.

Abstract

The video coding scheme proposed in this thesis is based on texture warping and synthesis. Instead of using techniques such as whole-image coding or prediction residues after motion estimation, motion models are used to reconstruct static textures and texture synthesis is used to create dynamic textures. Textured regions are segmented by the watershed method and then classified with simple methods based on their spectral characteristics by DCT transform and simple differences between neighboring frames. Regions reconstructed by warping and synthesis are evaluated for their use by simple statistical methods. The proposed scheme works together with the standard of video H.264/AVC, which only codes blocks that are not processed by models of texture, showing a potential to decrease computational cost of coding of blocks that remain. The results show that the quality was maintained with subjective scores compared to standard H.264/AVC, while objective results show small losses.

Keywords: Video Compression, H.264/AVC, texture, segmentation, warping, synthesis.

Sumário

Lista de Figuras

Lista de Tabelas

Capítulo 1

Introdução

1.1 Introdução

Sinais de vídeos digitais são utilizados em varias aplicações, como em videoconferências, em transmissões de *broadcast*[1], televisão digital de alta resolução (HDTV), DVD/Blu-Ray players, câmeras, entre muitos outros [1]. Para poder manipular estes sinais com eficiência é essencial a compressão de vídeo, pois permite reduzir a quantidade de dados para transmissão e armazenamento dos mesmos. A redução é feita de maneira tal que não haja perdas significativas de informação, através da exploração da redundância da informação contida em sequências de vídeo [4]. Para tal fim o padrão de compressão de vídeo H.264/AVC [5], que é um recente padrão de compressão de vídeo, se mostra mais eficiente que os padrões anteriores, tais como MPEG-2 [6] e H.263 [7]. Esta eficiência de compressão foi obtida a custa de um acréscimo computacional nos módulos do padrão H.264/AVC em relação aos padrões anteriores [8].

Na maioria de casos, o alvo da compressão de vídeo é oferecer uma boa qualidade subjetiva, em vez de simplesmente produzir as imagens mais semelhantes ás originais. Com base nesta hipótese, é possível conceber um sistema de compressão, onde uma estrutura baseado em análise/síntese é empregada em vez da convencional abordagem de minimização de taxa de bits [9]. A análise de textura é comumente usado em computação gráfica, onde o conteúdo sintético é gerado usando métodos espaciais e temporais [3]. Se o esquema fosse prático, este esquema poderia oferecer menores taxas de bits através da redução residual e codificação dos vetores de movimento [9]. Esta proposta foi reportada por autores tais como Ndjiki-Nya [10], Bosch [11], Bryne [12], e Zhang e Bull [13]. Embora estas abordagens tenham mostrado um potencial significativo, uma série de problemas são introduzidos pela análise e síntese de texturas, as quais ainda tem que ser resolvidos.

Neste trabalho, apresenta-se um esquema de codificação utilizando modelos de textura baseado no trabalho de Zhang e Bull [9]. O esquema é usado em conjunto com o padrão de codificação H.264/AVC. No esquema são aplicados métodos de *warping*[2] e síntese de textura para regiões classificadas como estáticas e dinâmicas respectivamente. Tais regiões são classificadas de acordo a suas características estatísticas e espectrais. Os principais benefícios desta proposta é que a codificação residual é geralmente não é necessário, e ape-

[1] *Broadcast* é o processo pelo qual se transmite ou difunde determinada informação, tendo como principal característica que a mesma informação está sendo enviada para muitos receptores ao mesmo tempo.

[2] *Warping* é um modelo linear translacional simples.

nas a informação lateral (parâmetros de movimento e mapas de deformação e texturas) tem que ser codificados. Outra contribuição, é a simplificação dos métodos de classificação e síntese do trabalho [9], mostrando um potencial para reduzir a complexidade na codificação em comparação do H.264.

1.2 Descrição do Problema

Apesar do padrão H.264/AVC ser bastante eficiente [8], acredita-se que ainda possam torná-lo ainda mais eficiente. Quando este padrão foi proposto, uma série de técnicas foram usadas em conjunto para melhorar as várias etapas do processo de codificação, de forma que a melhoria global fosse bem elevada. Entretanto, algumas dessas técnicas não vislumbram toda uma possibilidade de abordagens de forma que ainda podem ser exploradas. A etapa mais custosa do codificador do H.264 é a estimação de movimento (responsável pela busca de deslocamentos entre quadros), com um gasto de aproximadamente 90% do tempo total de codificação [14]. Por causa disso, qualquer alteração feita para evitar que um bloco no codificador não passe por esta etapa de estimação de movimento terá um impacto final considerável no tempo de execução da codificação.

O emergente codec *High Efficiency Video Coding* (HEVC) [15] que é proposto e atualmente é desenvolvido por *Moving Picture Experts Group* (MPEG) e *Joint Collaborative Team on Video Coding* (JCT-VC). As técnicas atualmente em avaliação para o HEVC são extensões do convencional modelo de compressão (compensação de movimento e transformação de bloco).

Um esquema de codificação em conjunto com o padrão H.264 foi apresentado [9], utilizando um esquema de *warping* e síntese ao invés da abordagem de minimização de taxa de bits convencional. Em tal esquema reduz-se a taxa de bits com os modelos de textura e mantém-se a qualidade subjetiva em comparação com o padrão H.264, mas com alto custo de complexidade computacional pelo uso da estratégia de codificação de duas passagens e mostrando-se inviável para aplicações em tempo real. Embora este esquema tenha-se mostrado com alta complexidade, ela tem um potencial significativo para reduzir o número de operações na estimação de movimento que realiza o padrão H.264.

Finalmente para compressão de vídeo baseado em sínteses, métricas objetivas de distorção como PSNR e SSIM [16] não são apropriadas [9]. Uma métrica subjetiva é necessária para avaliar a qualidade do conteúdo decodificado e para fornecer comparação com métodos concorrentes.

1.3 Objetivos

Este trabalho tem por objetivo geral a implementação de um esquema de codificação em conjunto com o padrão de codificação de vídeo H.264 que torne prático o trabalho de Zhang e Bull [9] e mantenha a qualidade subjetiva.

Como objetivos específicos temos:

- Realizar um estudo do padrão de codificação H.264/AVC.
- Realizar um estudo dos sistemas de compressão baseado em síntese de textura.

- Analisar a complexidade do esquema proposto em comparação do esquema base.
- Realizar experimentos com diferentes sequências de vídeo para verificar se a qualidade subjetiva é mantida.

1.4 Organização do Manuscrito

O trabalho é organizado a seguir: o Capitulo II apresenta alguns conceitos básicos cobrindo compressão de vídeo, o padrão H.264/AVC e métricas de qualidade. No Capitulo III os conceitos básicos de Sistemas de Compressão de Vídeo baseado em sínteses e a proposta do esquema de codificação de vídeo são descritos, explicando os modelos de textura utilizados. Os resultados são apresentados no Capitulo IV. Finalmente, as conclusões estão contidas no Capitulo V.

Capítulo 2

Fundamentação Teórica

Este capitulo revisa brevemente os conceitos relacionados com este trabalho, tais como: Compressão de vídeo, padrão H.264/AVC e métricas de avaliação da qualidade do vídeo.

2.1 Compressão de Vídeo

A compressão de vídeos é uma técnica que utiliza algoritmos para diminuir tamanho em disco de uma sequência de vídeo explorando as informações redundantes que a formam [4]. Em uma sequência de vídeo digital existe muita redundância na informação decorrente da similaridade existente entre pontos vizinhos espacialmente e também entre quadros vizinhos temporalmente. É uma compressão de sequência de imagens com um componente temporal [4], que faz parte do grupo mais geral de tecnologias de compressão de dados, que se classificam em compressão sem perdas e com perdas. No primeiro caso, após o processo de compressão/descompressão, o sinal reconstituído é idêntico ao original, enquanto no segundo caso há uma degradação desse sinal, denominada distorção [4].

Na compressão sem perdas pode-se compactar um sinal de duas a dez vezes [17], limita-se a um pequeno nicho de aplicações que não toleram qualquer distorção, como vídeos médicos ou sistemas de arquivamento. Um tipo de compressão de informação sem perdas é o codificador de entropia, que é dependente ou do modelo de probabilidades empregado ou da adaptabilidade do dicionário ou algoritmo. Dependendo da técnica de compressão utilizada, é possível converter uma série de símbolos que representam dados, *pixels* ou elementos da sequência do vídeo em uma sequência menor de bits, sendo mais adequada assim para a transmissão e o armazenamento.

Na compressão com perdas a taxa resultante depende apenas da distorção admitida, e apresenta um aumento na compressão que é bastante significativo. Com as tecnologias de compressão mais recentes pode-se comprimir para uma taxa até 100 vezes menor do que a taxa original, com uma distorção ainda aceitável [17]. A diferença da compressão sem perdas ocorre pela presença do bloco quantizador e, além disso, é a mais usada e difundida, já que certas distorções podem ser imperceptíveis ao olho humano.

2.1.1 Espaço de Cores e Sub-Amostragem de Cores

São vários os espaços de cores usados para representar imagens digitais, os mais utilizados são o RGB e YCbCr [19]. O espaço de cores RGB é o mais utilizado nos monitores

coloridos e TVs, os componentes de RGB representam três cores primárias captadas pelo sistema visual humano: vermelho (*Red*), verde (*Green*) e azul (*Blue*). O espaço YCbCr representa os três componentes: luminância (Y), que define a intensidade luminosa ou o brilho, crominância azul (Cb) e crominância vermelha (Cr) [20]. A vantagem da separação em YCbCr parte do princípio de que o sistema óptico humano é mais sensível à intensidade de luz do que às cores, e pode utilizar uma resolução menor na amostragem dos sinais de crominância quase sem prejuízo à qualidade subjetiva das imagens [1]. A exploração desta característica é denominada sub-amostragem de cores. Existem diferentes formatos na sub-amostragem de cores, os mais comuns são mostrados na Figura 2.1, os quais são:

- $4:4:4$, onde os três componentes de cor possuem a mesma resolução, ou seja, para cada quatro amostras Y existem quatro amostras Cb e quatro amostras Cr.
- $4:2:2$, onde as amostras de crominância possuem a mesma resolução vertical das amostras de luminância, mas metade da resolução horizontal, ou seja, para cada quatro amostras Y na direção horizontal, existem apenas duas amostras de Cb e Cr.
- $4:2:0$, onde as amostras de crominância possuem metade da resolução horizontal e vertical do que as amostras de luminância, ou seja, para cada quatro amostras Y, existem apenas uma amostra de Cb e Cr. Este formato teria que ser denominado $4:1:1$, mas é chamado $4:2:0$ por motivos históricos [1].

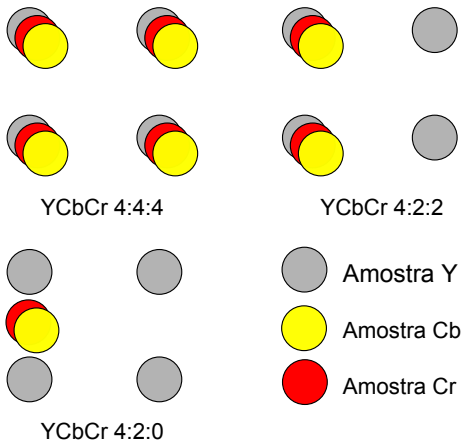


Figura 2.1: Diferentes formatos de sub-amostragem de cores YCbCr.

2.1.2 Redundância de Dados

O objetivo da compressão de vídeo é explorar e eliminar as informações redundantes nos quadros de um vídeo digital, e assim, diminuir o número de bits necessários para a sua representação [4]. Pode-se classificar em quatro tipos de redundâncias:

- A redundância espacial ou redundância intra-quadro [21], é a correlação existente entre os *pixels* espacialmente distribuídos em um quadro. Esta correlação pode ser

percebida tanto no domínio espacial quanto no domínio das frequências [1]. No domínio espacial é visualmente percebida quando são observados, como mostra o exemplo da Figura 2.2, onde os valores dos *pixels* vizinhos em um região tendem a ser semelhantes.

Figura 2.2: Valores dos *pixels* vizinhos de uma região.

- A redundância temporal ou redundância inter-quadro [21], é a correlação existente entre quadros temporalmente próximos em uma sequência de vídeo. Já que os valores dos *pixels* não mudam de valor de um quadro para outro. A Figura 2.3 mostra a diferença entre dois quadros correlatos, onde a cor cinza (ampla maioria) representa diferença nula entre os quadros. A exploração eficiente da redundância temporal conduz a elevadas taxas de compressão, o que é fundamental para o sucesso dos codificadores [1].

Figura 2.3: Diferença de dois quadros correlatos.

- A redundância entrópica está relacionada com as probabilidades de ocorrência dos símbolos codificados [1]. A entropia é uma medida da quantidade média de informação transmitida por símbolo do vídeo [19].
- A redundância psico-visual, é relativa às características do sistema visual humano, que fazem com que não consigamos captar alguns tipos de informações na imagem. Algumas informações da imagem, como o brilho, por exemplo, são mais importantes para o sistema visual humano do que outras, como a crominância. Este tipo de

redundância é explorada na sub-amostragem de cores, pois utilizando um subespaço de amostragem do tipo YCbCr 4 : 2 : 0, já existe uma eliminação das informações de cor do vídeo, fazendo com que o vídeo já esteja representado com uma compressão referente a este tipo de redundância.

2.2 Padrão H.264/AVC

O padrão de codya pificação de vídeo H.264/AVC (ou MPEG-4 parte 10) foi aprovado em outubro de 2003 pela Joint Video Team (JVT), grupo formado por especialistas do ITU-T Video Coding Experts Group (VCEG) e ISO/IEC Moving Picture Experts Group (MPEG). O padrão H.264/AVC oferece melhor compressão, reduzindo a taxa de bits de duas a três vezes mais que o MPEG-2 (Padrão largamente difundido e utilizado nos DVDs atuais, na maioria das transmissões digitais de televisão, etc.) [8].

O processo de codificação no padrão H.264/AVC (Figura 2.4) tem como dado de entrada um quadro Atual que é dividido em 'slices', e cada um destes é dividido em unidades básicas de codificação denominadas macroblocos [1]. Considerando que os dados de entrada do vídeo estão no espaço de cores YCbCr [20], um macrobloco corresponde a uma matriz de 16×16 amostras de luminância e matrizes de amostras de crominância correspondentes. Por exemplo, no formato $4 : 2 : 0$, cada macrobloco consistirá de uma matriz 16×16 de amostras de luminância e duas matrizes 8×8 de amostras de crominância (uma para Cb e outra para Cr).

A Figura 2.4 mostra o diagrama de blocos simplificado do codificador. O processo inicia pela escolha do modo de predição para o Quadro Atual, Inter ou Intra. Na predição Inter, estão contidos os módulos de Compensação de Movimento (CM) e Estimação de Movimento (EM). Logo após da predição, o menor resíduo vai para os módulos de Transformação (T), Quantização (Q), e Codificação de Entropia a fim de reduzir o numero de bits na codificação final.

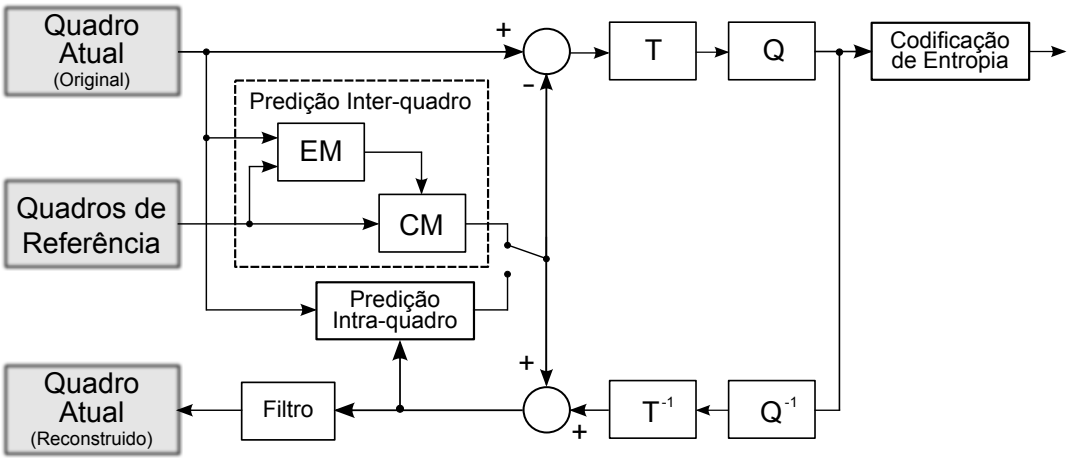


Figura 2.4: Diagrama de blocos do codificador H.264/AVC [5].

O codificador também faz o processo de decodificação a fim de obter o Quadro Atual (reconstruído), o qual é usado como Quadro de Referência nas predições Intra e Inter. O quadro reconstruído realiza a Quantização Inversa (Q^{-1}) e Transformação Inversa (T^{-1}).

O resultado é reconstruído de acordo com o modo da predição correspondente, seja Intra o Inter. Finalmente, aplica-se o filtro para minimizar o efeito de bloco. Na Figura 2.5 é mostrado este processo de decodificação.

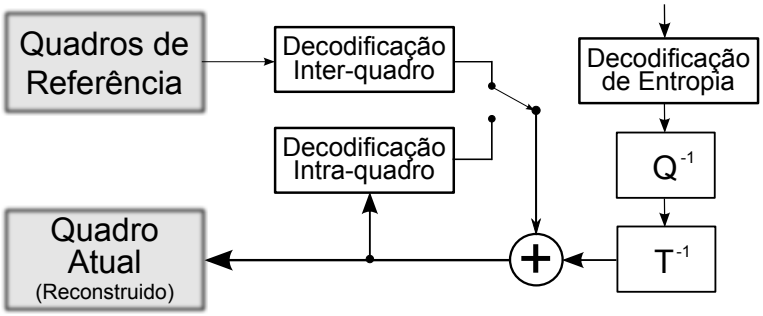


Figura 2.5: Diagrama de blocos do decodificador H.264/AVC [5].

No padrão H.264 são definidos diferentes tipos de slices [5], alguns deles são:

- I (Intra), onde cada bloco ou macrobloco do slice é predito utilizando dados anteriormente preditos contidos no mesmo slice.
- P (Preditivo), codifica do tipo I e, também alguns macroblocos ou subpartições podem ser codificados utilizado predição inter com no máximo uma referência por bloco predito.
- B (Bi-preditivo), codifica do tipo P e, também alguns macroblocos podem ser preditos a partir de duas referências a mais, podendo ser quadros anteriores ou posteriores.

2.2.1 Predição Intra Quadro

A predição Intra é uma técnica que estima cada macrobloco usando informações de macroblocos já codificados e pertencentes ao mesmo quadro. É realizada através de uma interpolação direcional com múltiplos modos, empregados em diferentes posições espaciais. O padrão H.264/AVC define nove modos diferentes de predição intra-quadro [8] para blocos de luminância 4×4 *pixels*, como mostrado na Figura 2.6 [5], onde a direção espacial é indicada pela direção da seta [1].

Contudo, a predição intra-quadro pode ocorrer com blocos de luminância com tamanho de 16×16 *pixels*. Neste caso, existem quatro modos possíveis de se realizar a predição, conforme está apresentado na Figura 2.7 [5]. Os modos '0' e '1' são simples cópias na vertical ou na horizontal das amostras reconstruídas das bordas, o modo '2' é o modo DC e é formado por uma média simples dos elementos das bordas, cujo resultado é copiado para todas as posições do bloco. O modo '3' aplica uma função linear que considera as amostras horizontais e verticais (H e V na Figura 2.7) das bordas.

2.2.2 Predição Inter Quadro

Considerando que a câmera permaneça fixa, as diferenças entre quadros vizinhos serão devidas a pequenas movimentações de objetos, e o fundo permanece constante. Para

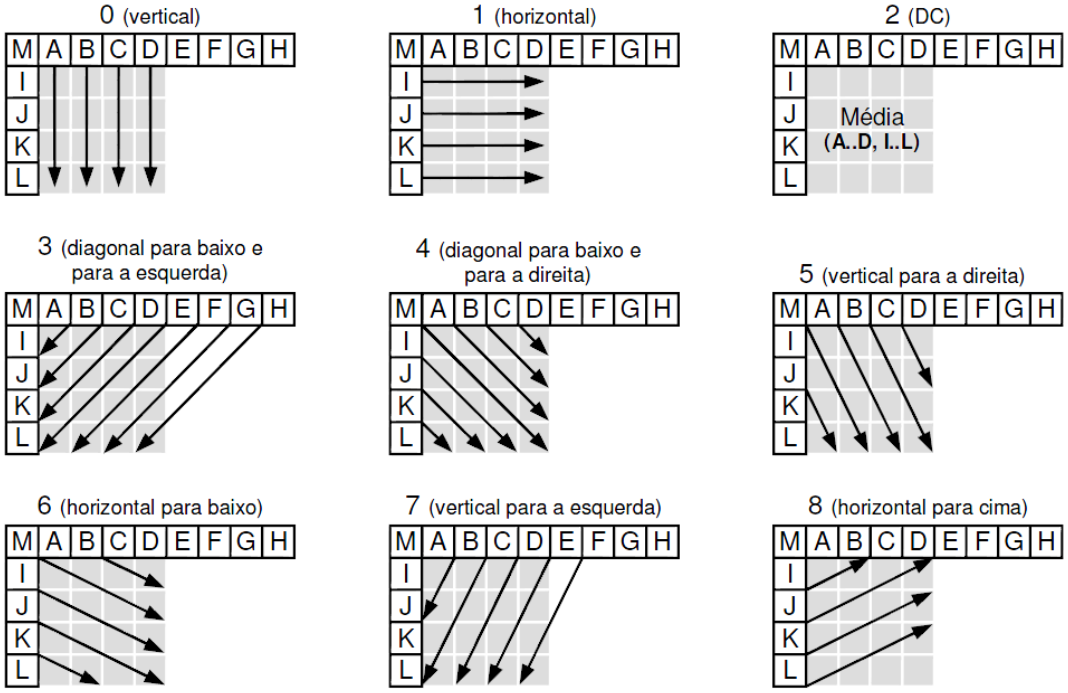


Figura 2.6: Nove modos de predição Intra-Quadro para blocos de 4×4 *pixels* [1].

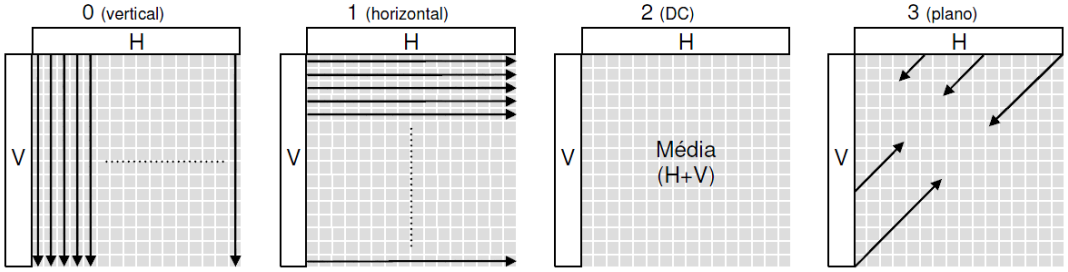


Figura 2.7: Quatro modos de predição Intra-Quadro para blocos de 16×16 *pixels* [1].

esta predição inicialmente, deve-se estimar os movimentos ocorridos entre dois quadros (quadro atual e o quadro de referência já codificado) por meio do módulo da estimação de movimento e após, mapear os movimentos através de um vetor de movimento pelo módulo da compensação de movimento. Para isso, inicialmente cada macrobloco 16×16 *pixels* pode ser dividido em blocos de 16×8, 8×16 e 8×8 *pixels* e logo preditos. Se o bloco de 8×8 *pixels* representasse a melhor escolha, pode novamente ser dividido em blocos de 4×8, 8×4, e 4×4 *pixels* [5]. Os tamanhos destes blocos são chamados partições *quad-tree* os quais são mostrados na Figura 2.8. Esta predição representa a maior complexidade computacional no codificador do H.264/AVC [8].

A estimação de movimento realiza um processo de predição nos quadros de referência para localizar qual macrobloco mais se assemelha ao macrobloco atual. Para isso são comparados os dois macroblocos com um critério para para medir a semelhança (ou diferença), costuma-se utilizar como métrica o erro médio quadrático, MSE(*Mean Squared*

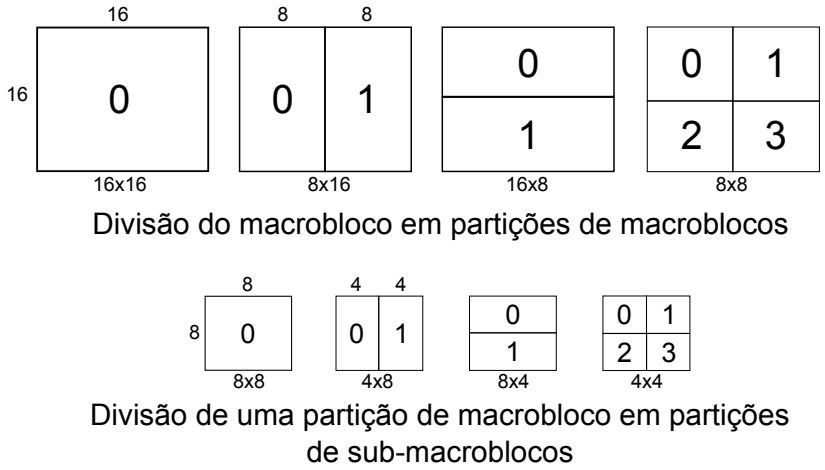


Figura 2.8: Partições de modo *quad-tree* do macrobloco [1].

error) definida na Equação (2.1), ou a soma das diferenças absolutas, SAD (*Sum of absolute differences*), definida na Equação (2.2):

$$MSE = \frac{1}{M} \sum_i \sum_j (P_{ij} - Pr_{ij}(x,y))^2, \tag{2.1}$$

$$SAD = \sum_i \sum_j |P_{ij} - Pr_{ij}(x,y)|, \tag{2.2}$$

onde P_{ij} é um *pixel* do macrobloco do quadro atual, $Pr_{ij}(x,y)$ é um *pixel* do macrobloco no quadro de referência (previamente codificado e reconstruído) candidato a predição (deslocado em x e y) e M é o número de *pixels* pertencentes ao bloco. O macrobloco escolhido é o que minimiza a função escolhida. Pode-se utilizar funções de custo mais complexas que levam em conta tanto taxa quanto distorção.

Com a estimação de movimento é gerado um vetor de movimento indicando a melhor posição do macrobloco no quadro de referência, esse vetor deve ser inserido junto com a codificação do macrobloco. A estimação de movimento do macrobloco é realizado apenas no componente de luminância e, nos componentes de crominância é considerado uma fração do vetor do movimento da luminância, por exemplo no formato $4:2:0$, os componentes horizontal e vertical de cada vetor de movimento são divididos por dois para serem aplicados aos blocos de crominância [1]. O módulo da estimação de movimento está presente apenas nos codificadores.

A compensação de movimento é o processo de compensação para o deslocamento de objetos movidos de um quadro para outro. É função da compensação, localizar os blocos de melhor casamento na memória de quadros anteriormente codificados (quadros passados e/ou futuros) a partir do vetor de movimento gerado na estimação de movimento, e montar o quadro predito. Este quadro será subtraído do quadro atual para gerar o quadro de resíduos que passará pela transformada.

A compensação de movimento é realizada tanto no codificador quanto no decodificador. No codificador a compensação pode ser simplificada, tendo em vista as possíveis simplificações que podem ser realizadas na estimação de movimento no codificador. No

decodificador a compensação deve ser completa, isto é, deve estar apta a operar sobre todas as funcionalidades do padrão [22].

2.2.3 Transformada

Este processo consiste em transformar os valores das amostras de um bloco de modo a reduzir redundâncias espaciais, já que amostras próximas possuem valores semelhantes. Neste módulo estão contidas: a Transformada Discreta do Cosseno (DCT) e a Transformada Hadamard (TH), a execução das transformadas na codificação e fazer primeiro a DCT e logo a TH.

A DCT [23] é uma transformada muito usada em padrões de compressão de vídeo devido à sua capacidade de descorrelação e à sua eficiência na compactação de energia. Por capacidade de descorrelação entende-se a habilidade de converter um conjunto de dados altamente correlacionados em outro conjunto de dados relativamente independentes (reduzindo a redundância estatística). Já a eficiência em compactação de energia refere-se à habilidade de compactar o conteúdo energético de um sinal na menor quantidade possível de coeficientes. O fato de a DCT ser uma transformada independente de dados e possuir algoritmos rápidos são argumentos que justificam sua popularidade [24].

Para um sinal de $N \times N$ amostras, tomado aqui como uma fração (bloco) de um quadro a ser comprimido, os coeficientes de sua matriz de transformação $\mathbf{C}$ são obtidos por funções de cossenos [4], conforme verificado na Equação (2.3):

$$[\mathbf{C}]_{i,j} = \begin{cases} \sqrt{\frac{1}{N}} \cos \frac{(2j+1)i\pi}{2N} & i = 0, j = 0, 1, ..., N-1, \\ \sqrt{\frac{2}{N}} \cos \frac{(2j+1)i\pi}{2N} & i = 1, 2, ..., N-1, j = 0, 1, ..., N-1. \end{cases} \tag{2.3}$$

O resultado da aplicação da DCT em um bloco de $N \times N$ amostras é um conjunto de $N \times N$ coeficientes representando o bloco no domínio transformado, coeficientes que podem ser considerados como ponderações para um conjunto de matrizes de base, ilustrado na Figura 2.9 para o caso em que $N = 8$. Dessa forma, a representação no domínio transformado pode ser encarada como a representação do sinal pela combinação de todas as $N \times N$ matrizes de base, cada qual multiplicada por seu fator de ponderação apropriado [5].

A DCT do padrão H.264/AVC é uma aproximação inteira da DCT 2-D real mostrada na equação (2.3). Esta aproximação foi realizada para reduzir a complexidade do cálculo e evitar erros de casamento entre o codificador e o decodificador. É aplicada a todas as amostras de entrada, tanto de luminância quanto de crominância.

A transformada $\mathbf{Y}$ do bloco $\mathbf{X}$ de entrada é dada pela equação (2.4), onde a matriz $\mathbf{C}_f$ é da DCT inteira em uma dimensão, a matriz $\mathbf{C}_f^T$ é transposta da matriz da DCT e a última matriz $\mathbf{E}_f$ é a dos fatores de escala. O símbolo '$\cdot$' na equação indica produto escalar.

$$\mathbf{Y} = \mathbf{C}_f \mathbf{X} \mathbf{C}_f^T \cdot \mathbf{E}_f. \tag{2.4}$$

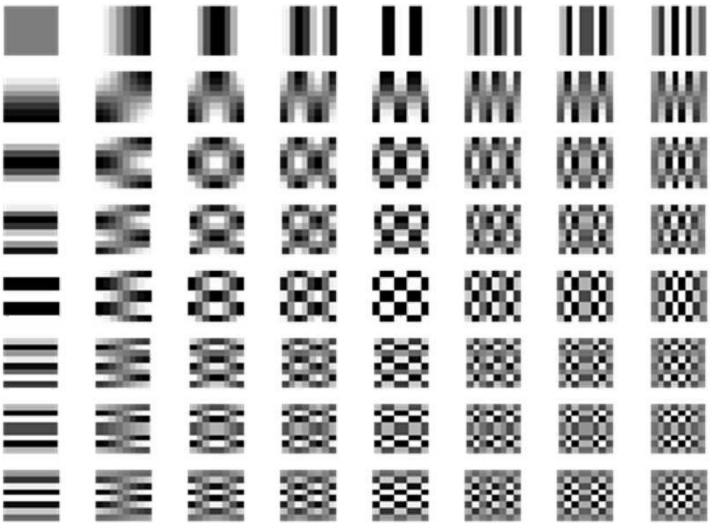


Figura 2.9: Bases da DCT para blocos de 8×8 *pixels*. Adaptado de [2]

.

Como os blocos da matriz $\mathbf{X}$ serão de 4×4, a equação ficaria como é mostrado em (2.5), com as constantes para as letras 'a' e 'b' mostradas na equação (2.6).

$$\mathbf{Y} = \left(\begin{bmatrix} 1 & 1 & 1 & 1 \\ 2 & 1 & -1 & -2 \\ 1 & -1 & -1 & 1 \\ 1 & -2 & 2 & -1 \end{bmatrix} [\mathbf{X}] \begin{bmatrix} 1 & 2 & 1 & 1 \\ 1 & 1 & -1 & -2 \\ 1 & -1 & -1 & 2 \\ 1 & -2 & 2 & -1 \end{bmatrix} \right) \cdot \begin{bmatrix} a^2 & \frac{ab}{2} & a^2 & \frac{ab}{2} \\ \frac{ab}{2} & \frac{b^2}{4} & \frac{ab}{2} & \frac{b^2}{4} \\ a^2 & \frac{ab}{2} & a^2 & \frac{ab}{2} \\ \frac{ab}{2} & \frac{b^2}{4} & \frac{ab}{2} & \frac{b^2}{4} \end{bmatrix}, \tag{2.5}$$

$$a = \frac{1}{2}, b = \sqrt{\frac{2}{5}}. \tag{2.6}$$

Na decodificação à inversa $\mathbf{Y}'$ da DCT está definida pela equação 2.7 e de maneira expandida na equação 2.8, de acordo com o padrão H.264/AVC, onde $\mathbf{X}'$ é a matriz 4×4 de entrada, $\mathbf{C}_i$ é a matriz da inversa DCT inteira em uma dimensão, $\mathbf{C}_i^T$ é a transposta de $\mathbf{C}_i$ e $\mathbf{E}_i$ é a matriz de fatores de escala. As letras 'a' e 'b' na matriz $\mathbf{E}_i$ são as mesmas constantes definidas em (2.6).

$$\mathbf{Y}' = \mathbf{C}_i^T (\mathbf{X}' \cdot \mathbf{E}_i) \mathbf{C}_i, \tag{2.7}$$

$$\mathbf{Y}' = \begin{bmatrix} 1 & 1 & 1 & \frac{1}{2} \\ 1 & \frac{1}{2} & -1 & -1 \\ 1 & -\frac{1}{2} & -1 & 1 \\ 1 & -1 & 1 & -\frac{1}{2} \end{bmatrix} \left([\mathbf{X}'] \cdot \begin{bmatrix} a^2 & ab & a^2 & ab \\ ab & b^2 & ab & b^2 \\ a^2 & ab & a^2 & ab \\ ab & b^2 & ab & b^2 \end{bmatrix} \right) \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & \frac{1}{2} & -\frac{1}{2} & -1 \\ 1 & -1 & -1 & 1 \\ \frac{1}{2} & -1 & 1 & -\frac{1}{2} \end{bmatrix}. \tag{2.8}$$

No caso de amostras com informação de crominância ou com informação de luminância cuja predição tenha sido do tipo intra-quadro 16×16, um cálculo adicional com a TH é realizado sobre os coeficientes DC resultantes da DCT. O coeficiente DC é o primeiro

elemento do bloco transformado. A TH é uma inovação do padrão H.264/AVC e foi inserida com o objetivo de atingir ainda mais compressão em áreas homogêneas do vídeo. A Transformada de Hadamard explora uma correlação residual que ainda permaneça sobre os coeficientes da DCT [20].

Os coeficientes DC dos blocos de luminância cuja predição tenha sido intra-quadro 16×16 são submetidos a uma TH 4×4 direta. Então os 16 elementos DC dos 16 blocos do macrobloco irão formar a matriz de entrada de 4×4 ($\mathbf{W}_D$) para a TH direta 4×4 como mostra a equação (2.9), onde $\mathbf{Y}_D$ e o resultado da transformação. A TH 2×2 é aplicada apenas para amostras DC de crominância. Esta transformada é utilizada tanto para Cb quanto para Cr. O cálculo da TH 2×2 definido pelo padrão H.264/AVC é representado na equação (2.10), onde $\mathbf{Y}_{QD}$ e o resultado da transformação da matriz de entrada de 2×2 ($\mathbf{W}_{QD}$).

$$\mathbf{Y}_D = \left(\begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \\ 1 & -1 & 1 & -1 \end{bmatrix} [\mathbf{W}_D] \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \\ 1 & -1 & 1 & -1 \end{bmatrix} \right) /2 \tag{2.9}$$

$$\mathbf{Y}_{QD} = \left(\begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} [\mathbf{W}_{QD}] \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \right) /2 \tag{2.10}$$

2.2.4 Quantização

O módulo de quantização é usado para reduzir a precisão dos coeficientes resultantes da transformação. O processo de quantização é equivalente a uma divisão seguida por um arredondamento mostrado em (2.11), onde P_q é o valor quantizado, P é o valor a ser quantizado, Q_p é o passo de quantização e Z em (2.12) é o valor recuperado [5]. O processo de quantização é mais complexo, pois utiliza apenas aritmética inteira (multiplicação e deslocamento de bits), entretanto o efeito é o mesmo.

Para decodificar é realizado a quantização inversa, onde é usada a equação (2.12). Os valores resultantes em Z após este processo são menores, necessitando menos bits na codificação, contudo existe uma perda na precisão. Quando esses valores forem reconstituídos, serão obtidos valores que diferem dos valores originais (já que a operação de arredondamento pode causar uma "perda" de informação). A maior parte da distorção, portanto, é resultante do processo de quantização.

$$P_q = round\left(\frac{P}{Q_p}\right) \tag{2.11}$$

$$Z = P_q.Q_p \tag{2.12}$$

2.2.5 Codificação de Entropia

A codificação de entropia está presente tanto nos codificadores quanto nos decodificadores. O padrão H.264/AVC introduz algumas ferramentas que aumentam bastante a sua eficiência de codificação. Nos níveis hierárquicos superiores (quadros, etc.), os elementos sintáticos são codificados usando códigos binários fixos ou de comprimento variável.

A partir do nível de slices ou abaixo (macroblocos, blocos, etc.), os elementos sintáticos são codificados usando a codificação aritmética adaptativa ao contexto (CABAC) [5] ou códigos de comprimento variável (VLC).

No caso do uso de VLC, a informação residual como são os coeficientes quantizados das transformadas é codificada usando a codificação de comprimento variável adaptativa ao contexto (CAVLC) [5], enquanto que as demais unidades de codificação são codificadas usando códigos Exp-Golomb. A principal inovação introduzida na codificação de entropia do padrão H.264/AVC, tanto na codificação aritmética quanto na codificação VLC é o uso de codificação adaptativa baseada em contextos. Nela, a maneira com que são codificados os diversos elementos sintáticos depende do elemento a ser codificado, da fase em que se encontra o algoritmo de codificação e dos elementos sintáticos que já foram codificados [1].

2.2.6 Filtro Redutor de Efeito do Bloco

Uma característica particular da codificação baseado em blocos é a produção acidental de estruturas de bloco visíveis. As bordas dos blocos são tipicamente reconstruídos com menos precisão do que os *pixels* internos e o efeito do bloco é geralmente considerado como um dos mais visíveis artefatos [25]. Por esta razão, o H.264/AVC define um filtro adaptativo para suavizar o efeito de bloco do quadro reconstruído, antes dele ser usado para fazer a predição de um novo macrobloco do tipo inter-quadro. O filtro de deblocagem opera sobre um macrobloco após a compensação de movimento e da codificação residual se o macrobloco é codificado no modo inter-quadro, ou sobre um macrobloco após a predição intra-quadro e a codificação residual. O resultado na saída do filtro de deblocagem é armazenado para ser usado como quadro de referência, com exceção para aquelas figuras que não serão usadas como quadro de referência.

O filtro atua nas bordas do macrobloco e do bloco. A operação do filtro de deblocagem é adaptativa em relação a um conjunto de vários fatores, o parâmetro de quantização do macrobloco atual e dos vizinhos, a magnitude do vetor de movimento e o tipo de codificação do macrobloco. Também é levado em consideração os valores dos *pixels* que serão filtrados, tanto do macrobloco atual como os valores dos macroblocos vizinhos [26].

2.3 Avaliação da Qualidade

A avaliação da qualidade das sequências de vídeo é importante para dimensionar distorções que podem reduzir a qualidade no momento de exibição. Existem duas formas de se obter a avaliação da qualidade de uma imagem: através de métodos subjetivos, isto é, por meio de notas produzidas por observadores humanos, ou através de métodos objetivos, onde algoritmos simulam o comportamento do sistema visual humano produzindo a avaliação da qualidade.

2.3.1 Métodos Subjetivos

A medida da qualidade percebida em vídeos requer o uso de métodos subjetivos. A condição para tal medida apresentar significado é haver uma relação entre as característi-

cas físicas do estímulo (um vídeo distorcido apresentado aos observadores em um teste), a magnitude, e a natureza da sensação causada pelo estímulo.

Diversos métodos experimentais para avaliação subjetiva foram criados voltados para diferentes propósitos [27], como:

- *Double Stimulus Impairment Scale* (DSIS), aonde os vídeos são apresentados em sequência. Primeiramente é apresentado o vídeo de referência e, em seguida, o vídeo processado. É solicitado ao observador que avalie o grau de distorção do vídeo processada com relação ao vídeo de referência.
- *Double Stimulus Continuous Quality Scale* (DSCQS), onde o observador assiste aos vídeos de referência e distorcido. A ordem temporal dos vídeos durante as seqüências é alterada de forma pseudo-aleatória, e não é dado o conhecimento aos observadores sobre qual vídeo é a referência. Ambos vídeos de cada seqüência devem ser avaliadas. O número de repetições depende da duração do teste.
- *Single Stimulus* (SS): os vídeos sob teste são apresentados um por vez e avaliados independentemente com uma escala de categorias. Este método especifica que após cada apresentação o observador deve avaliar a qualidade do vídeo exibido. Uma escala de cinco níveis de qualidade geralmente é usada.
- Comparação de Estímulos, aonde os dois vídeos de mesmo conteúdo, um de referência e outro sob teste, são mostrados simultaneamente, e o observador deve dar uma nota correspondente à relação entre os vídeos.

Obtidas as avaliações subjetivas para um vídeo por um grupo de observadores, é necessário que seja feita uma média das notas a fim de se ter um único valor correspondente à qualidade do vídeo.

O MOS (*Mean Opinion Score*) fornece a média das notas obtidas dos observadores para um vídeo. Para seu cálculo é necessário que um vídeo seja avaliado por um grupo de observadores através de qualquer um dos métodos descritos anteriormente. O MOS_j fornece a média das notas de i observadores para o vídeo j via:

$$MOS_j = \frac{1}{N}\sum_{i=1}^{N} m_{ij}, \tag{2.13}$$

onde m_j é a nota dada pelo observador i para o vídeo j sendo avaliado e N é o número de observadores que avaliaram o vídeo em questão.

2.3.2 Métodos Objetivos

A medida objetiva da qualidade, em oposição à medida subjetiva produzida por observadores humanos, procura determinar a qualidade de imagens de forma algorítmica [28]. Assim, quer-se fornecer uma nota quantitativa computacionalmente que descreva o grau de similaridade/fidelidade ou, de modo oposto, o nível de erro/distorção entre imagens.

Em sistemas de vídeo digital utiliza-se a medida chamada PSNR -*Peak Signal to Noise Ratio*, que poderia ser traduzida como relação entre o pico do sinal e o ruído. A PSNR é definida pela equação 2.14, onde MSE, Equação (2.1), é calculada entre a imagem comprimida e a imagem original, e o numerador é o quadrado do valor máximo que um

pixel pode assumir nessa imagem, dependente do número de bits por pixel. Por exemplo, se 8 bits são usados, o valor de pico seria 255. O resultado da PSNR é dado em decibéis.

$$PSNR_{dB} = 10log_{10}\frac{(2^n - 1)^2}{MSE}, \tag{2.14}$$

A maior diferença de resultados entre a percepção subjetiva e a PSNR é que esta ultima dá pesos iguais a todos os *pixels* da imagem. Uma grande variação em um único pixel (ou região), em geral, não degrada muito a PSNR de uma imagem, enquanto o sistema visual humano é capaz de perceber essas variações [2].

Um dos modelos mais recentes e já consolidados para as métricas objetivas de avaliação da qualidade de vídeos é o SSIM (*Strucutural Similarity*) ou Índice de Similaridade Estrutural [16]. A métrica SSIM foi proposta com base no pressuposto de que o Sistema Visual Humano é altamente adaptado para extrair informações estruturais de imagens.

No método SSIM os *pixels* possuem forte dependência entre si e ela aumenta consideravelmente de acordo com a proximidade entre os *pixels*. Essa dependência carrega informações importantes sobre a estrutura dos objetos na imagem e que quantificar a mudança estrutural de uma imagem pode fornecer uma boa aproximação para a qualidade percebida.

O SSIM é um algoritmo que utiliza a estatística da imagem para a avaliação da qualidade, ou seja, os atributos que constituem a informação estrutural dos objetos da imagem, que dependem da média da luminância e do contraste da imagem [16]. O SSIM divide a imagem em blocos de $n \times n$ *pixels* e então são calculados, para cada bloco, média, desvio padrão e covariância. A média e o desvio padrão são considerados estimativas aproximadas da luminância e do contraste da imagem, respectivamente. A covariância é a medida de quanto um sinal é diferente do outro [29]. A medida entre dois blocos x e y é dado pela equação (2.15), onde μ_x e μ_y são as médias de x e y respectivamente, σ_x e σ_y são as variâncias, σ_{xy} é a covariância, e as duas variáveis C_1 e C_2 são para estabilizar a divisão [28].

$$SSIM(x, y) = \frac{(2\mu_x\mu_y + C_1)(2\sigma_{xy} + C_2)}{(\mu_x^2 + \mu_x^2 + C_1)(\sigma_x^2\sigma_y^2 + C_2)}. \tag{2.15}$$

Capítulo 3

Compressão baseado em Modelos de Textura

3.1 Sistemas de compressão baseadas em síntese

Os sistemas de compressão de vídeos baseados em síntese consistem normalmente em três módulos: segmentação de textura, análise de textura e avaliação de qualidade [9], o funcionamento destes sistemas é ilustrado na Figura 3.1. A segmentação de textura separa imagens em regiões, nas quais são realizadas a análise e síntese pelo módulo de análise de textura utilizando modelos de síntese adequadas. As texturas resultantes da síntese são analisadas pelo módulo de avaliação de qualidade, identificando inaceitáveis artefatos para a sua codificação. As regiões onde não seja realizada a síntese são codificadas usualmente com algum codificador de vídeo. Para estes processos são apresentados alguns conceitos básicos.

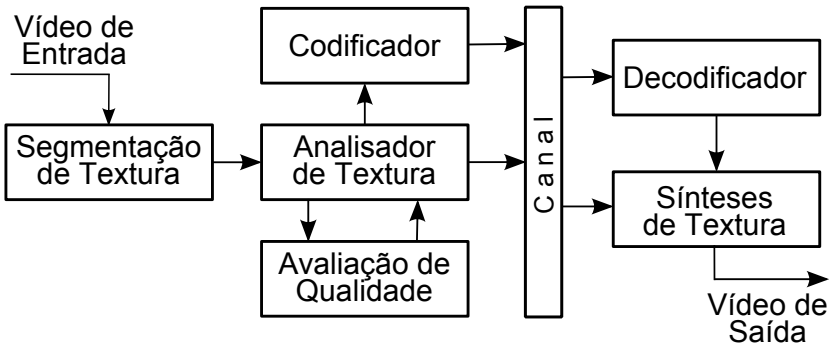


Figura 3.1: Compressão de imagem o vídeo baseado em síntese [3].

3.1.1 Textura

A textura possui um papel central na percepção humana, provendo dados para reconhecimento e interpretação, e por isso é uma importante característica/propriedade para a análise de imagens digitais. Em algumas tarefas de visão computacional, como inspeção de superfícies, classificação de cenas e determinação de formas, existe uma forte dependência

com a correta extração e compreensão das informações da textura. Algumas definições do termo de textura:

- "O termo textura geralmente se refere à repetição de elementos básicos de textura chamados texels. O texel contem vários *pixels*, cuja colocação pode ser aleatória ou periódica. Texturas Naturais são geralmente aleatórias, ao passo que texturas artificiais são freqüentemente determinísticas ou periódicas. Textura pode ser áspera, fina, suave, granulada, ondulada, irregular, regular, ou linear. " [30]
- "Modelos que caracterizam objetos são chamados texturas em Processamento de Imagens." [31]
- "Nós intuitivamente vemos este descritor como provedor de uma medida de propriedades tal como suavidade, asperidade e regularidade." [32]

Em imagens digitais, a caracterização de uma textura pode ser percebida através de variações das intensidades ou cores capturadas. Apesar de não haver nenhuma informação quanto à causa das variações, as diferenças existentes nos *pixels* da imagem fornecem meios práticos de analisar as propriedades das texturas de um objeto.

3.1.2 Segmentação de Texturas

Uma definição de segmentação de imagens texturas poderia ser o processo pelo qual uma imagem é dividida em regiões, onde cada uma delas contêm uma textura diferente das regiões vizinhas. No entanto, esta definição explica muito pouco sobre as dificuldades inerentes e limitações práticas envolvidas neste problema.

Uma imagem consiste em um array de *pixels* e o objetivo da segmentação é dar a um nome (etiqueta) a cada um deles de forma que possamos agrupá-los em regiões. Uma região é formada por um grupo de *pixels* que possui o mesmo nome, mas o que constitui uma região 'correta'? Nem sempre é possível definir claramente a fronteira que separa duas amostras de texturas diferentes ou nem sempre queremos que texturas muito parecidas sejam separadas como se fossem duas amostras.

É importante perceber que, para a segmentação propriamente dita, não é necessário identificar quais texturas foram usadas e sim onde ficam as fronteiras entre as diferentes texturas. Na Figura 3.2, por exemplo, para resultado da segmentação, não é importante saber que "a textura do canto esquerdo inferior é composto de tijolos brancos".

Existem diferentes métodos de segmentação, alguns baseados em descontinuidade, similaridade e transformada de *watershed* [32]. Os métodos de descontinuidade observam que descontinuidades numa imagem representam os limites das regiões significativas. As localizações destas bordas descontínuas têm frequentemente um gradiente local máximo. O gradiente é o operador mais comum em detecção de bordas [32] e pode ser obtido por meio de métodos como o operador de Sobel [33] ou o detector de bordas Canny [34]. A segmentação baseada em similaridade foca na relação entre os *pixels* dentro de uma região homogênea [32]. Embora estes métodos sejam simples e fáceis de usar, o seu desempenho na segmentação de imagens texturizadas é muitas vezes deficiente no caso das regiões altamente texturizadas. Alem disto, na segmentação de imagens complexas, estes métodos podem conduzir à geração de regiões disjuntas. A transformada de *watershed* é um método de fusão dos conceitos de crescimento de região e detecção de descontinuidade.

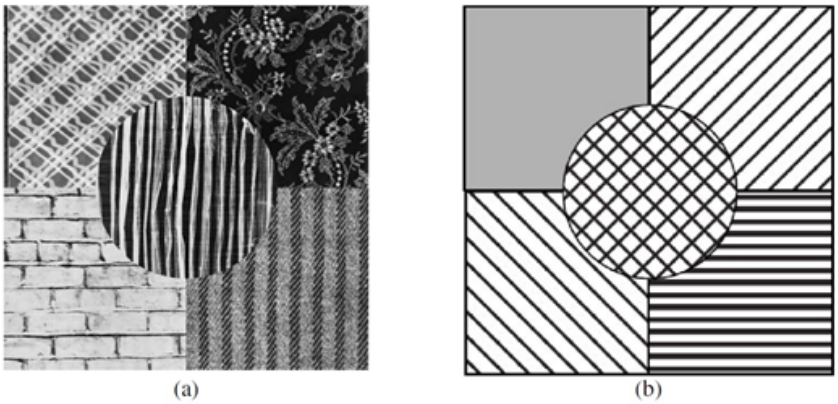


Figura 3.2: Objetivo da segmentação de texturas. a) Imagem composta por cinco texturas diferentes; em b) o resultado da segmentação ideal/perfeita da imagem [3].

3.1.3 Segmentação por Watershed

A transformada de *watershed*, é um método de segmentação baseado em regiões. Este método classifica os *pixels* segundo a sua proximidade espacial, o gradiente de seus níveis de cinza e a homogeneidade das suas texturas [32] [35]

A técnica pode ser entendida fazendo uma analogia a topografia, em que regiões com *pixels* de valores altos, ou regiões claras, representam as montanhas ou picos, e regiões com *pixels* de valores baixos, ou regiões escuras, representam os vales, estes conceitos são ilustrados na Figura 3.3. Suponha que há uma perfuração em cada vale e que ocorre a inundação dessas regiões; a água irá subindo até encontrar águas de outros vales. No momento em que elas se tocam é criada uma linha perpendicular à superfície da água passando sobre o ponto de encontro entre a água desses dois vales, formando assim linhas divisórias para que a água de uma determinada região não se toque com a outra. Desse modo, quando a água atinge seu ponto no relevo, o processo tem fim [36].

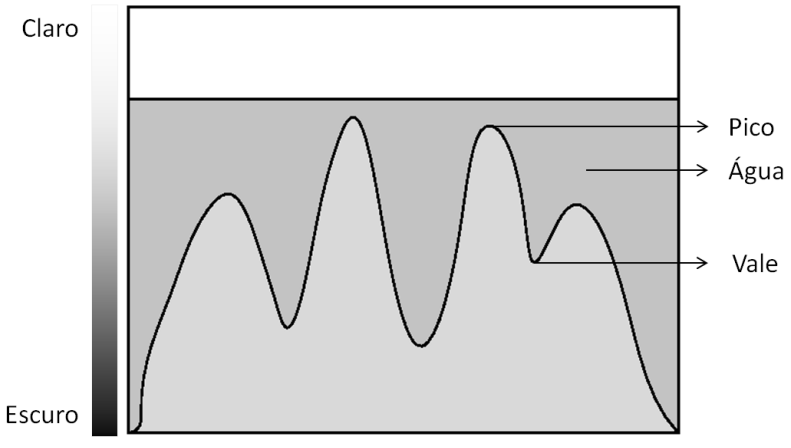


Figura 3.3: Conceitos da *watershed* fazendo analogia a topografia.

Existem diferentes métodos de *watershed* (p.e. [37], [38]) baseados no trabalho original de Lantuéjoul [39]. Por ser uma técnica dependente dos valores dos *pixels*, há uma alta sensibilidade a ruídos, gerando resultados sobre-segmentados [40], que consiste na divisão

da imagem de maneira exagerada, devido ao fato de mínimos locais não significativos originarem bacias de segmentação. A forma de evitar ou diminuir esse efeito é a escolha de marcadores para a região onde se encontra o objeto de interesse. A simplificação da imagem a partir de algum processo de filtragem ou redução de mínimos locais e, consequentemente, bacias de segmentação, também podem conduzir a uma diminuição do efeito da sobre-segmentação. Um método de simplificação foi abordada por autores como O'Callaghan [41], que apresentou um método baseado nas características DT-CWT (*Dual-Tree Complex Wavelet Transform*) [42], que combina métodos de *watershed* com cluster espectral.

3.1.4 Análise e Síntese de Textura

Para caracterizar as propriedades de textura de uma região de imagem podem ser usados alguns parâmetros, tais como suavidade, aspereza e regularidade. O análise de textura tem atraído atenção significativa nos últimos anos, enfocando modelos estatísticos, modelos espectrais, modelos estruturais e modelos estocásticos. Uma revisão detalhada de métodos de análise de textura e de síntese é apresentada em Mirmedhdi et. al. [3]. Alguns métodos adequados na codificação de vídeo são apresentados a seguir.

Ndjiki-Nya et. al. [10] assumem que algumas cenas de vídeo podem ser classificadas em texturas subjetivamente relevantes e irrelevantes. Texturas subjetivamente irrelevantes são definidas como regiões altamente texturizadas, enquanto as relevantes são as restantes. Regiões texturizadas são aquelas onde o espectador percebe o conteúdo semântico da textura ao invés do que os detalhes específicos nela. A análise de textura é feita no codificador enquanto a síntese de textura é feita no decodificador. A tarefa da análise de textura consiste em identificar regiões de texturas irrelevantes que são sintetizadas no decodificador com a correspondente informação lateral. O método de síntese considera os quadros do vídeo como volumes tridimensionais, baseados no trabalho de Kwatra [43]. Contudo, este método necessita de uma ampla memória para salvar os quadros de referência, e só trabalha com texturas temporalmente estáticas.

Um trabalho a ressaltar é o de Doretto [44], que apresenta a definição de textura dinâmica de vídeo como uma saída de um processo auto-regressivo de média móvel. Com base nesse modelo, quadros ilimitados podem ser sintetizados a partir de um modelo de treinamento usando um número finito de quadros. Isso proporciona excelentes resultados em síntese de textura dinâmica, produzindo imagens naturais e consistentes.

3.1.5 Codificação de Vídeo com Modelos de Textura

Vários codificadores de vídeo têm métodos baseados em síntese de textura, propostos para reduzir a taxa de bits para uma dada qualidade visual comparado a métodos de compressão convencionais. No codec H.264/AVC o particionamento de quadros de vídeo nesses métodos se baseia em blocos. No entanto, para a síntese de textura, métodos baseados em regiões são mais atraentes, já que regiões segmentadas possuem texturas homogêneas que não são divididas exatamente em blocos.

Nos trabalhos de Ndjiki-Nya foram propostos uma série de métodos de codificação de vídeo empregando análise e síntese de textura [10][45][46]. Dentre eles, destaca-se um método de codificação de vídeo genérico com *warping* e síntese de textura integrado com

o H.264/AVC [10]. Este método tem um desempenho competitivo em comparação do H.264/AVC em termos de economia de taxa de bits, e fornece uma boa qualidade visual subjetiva. Contudo, a técnica de síntese utilizada pode introduzir artefatos temporais e diminuir a qualidade subjetiva, principalmente para texturas complexas.

Uma alternativa de codificação de vídeo é a proposta por Bosch [11] que aumenta a eficiência de compressão usando um modelo de textura espacial, o qual consiste na extração de características, segmentação e *warping* de textura. Relata-se uma economia de taxa de bits de até 26% em comparação com H.264/AVC. Em um segundo algoritmo [47], alcançam-se economias adicionais na taxa de bits através da classificação de movimento. Neste caso, o fundo de cada quadro de tipo B é estimado via *warping* por meio de um modelo de movimento global, sem se codificar o resíduo, enquanto que o primeiro plano é codificado de forma tradicional. Contudo, o referido trabalho relata um artefato temporal que faz com que o vídeo não apareça natural.

No trabalho de Zhang e Bull [9], propõe-se um esquema de codificação usando modelos de textura baseada em regiões, classificando-as segundo seu movimento em estáticas e dinâmicas. Para as texturas estáticas aplica-se o *warping*, enquanto nas texturas dinâmicas é aplicada a síntese. As regiões não classificadas nem como estáticas nem como dinâmicas são codificadas com o codec utilizado. Além disso, propõe-se uma métrica de avaliação de qualidade para codificação de vídeo baseado em síntese. Os resultados deste esquema com as métricas PSNR e SSIM mostram perdas, enquanto com MOS e a métrica proposta, mostram ganhos em relação ao codec de vídeo utilizado. Este esquema é utilizado como inspiração para o presente trabalho.

Outros algoritmos de codificação de imagens e vídeos baseados em modelos de textura incluem métodos de compressão de imagens com fatores que repetem conteúdo [48], métodos baseados em *image inpainting* [49], métodos que aplicam extrapolação de textura dinâmica [50] e modelos de campos aleatórios 2-D Gaussian-Markov [51].

3.2 Esquema do Codec

A motivação básica que sustenta o presente trabalho é a de desenvolver um método eficiente com baixa complexidade para reconstruir texturas em imagens em movimento. As regiões texturizadas em imagens em movimento compreendem padrões regulares ou irregulares, além do movimento associado.

As regiões texturizadas em sequências de imagens e imagens estáticas diferem pelo simples fato que existem texturas estáticas (por exemplo madeira, piso e parede) e texturas dinâmicas (por exemplo água em movimento, fogo, folhas ao vento). As texturas estáticas têm movimentos simples (tipicamente devido ao movimento da câmera) e as texturas dinâmicas têm movimentos complexos. Desse fato as regiões texturizadas podem ser classificadas em estáticas e dinâmicas, de forma que as regiões estáticas são reconstruídas por meio de modelos de movimento de ordem superior (*warping*) e as regiões dinâmicas são analisadas e sintetizadas através de uma janela temporal.

O esquema do codec proposto trabalha numa estrutura de grupo de imagens (GOP) da forma IBBBPBBBP. Os processos do esquema são aplicados a cada cinco imagens como é ilustrado na Figura 3.4. Nesta figura os quadros de tipo I e P são definidos como quadros chave, ou seja são codificados pelo codificador do vídeo H.264. Os quadros B intermediários são processados de forma especial conforme descreveremos a seguir.

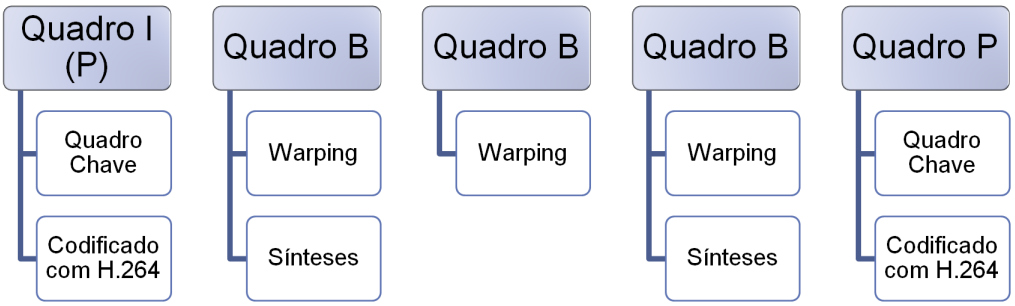


Figura 3.4: Esquema do grupo de imagens.

3.2.1 Esquema do Codificador

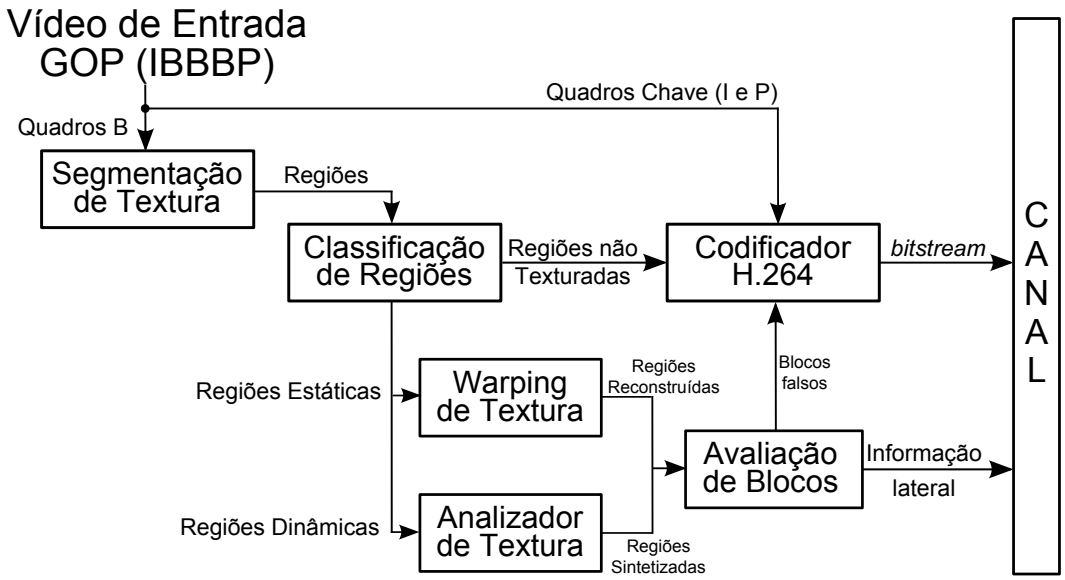


Figura 3.5: Esquema de codificação.

O processo proposto de codificação é realizado como é ilustrado no digrama do blocos da Figura 3.5. Nesta figura, a entrada do esquema é um GOP de cinco imagens, codificando os quadros chave (1° e 5° quadro do GOP) com o padrão H.264 e os quadros B intermediários com os seguintes passos:

- Cada quadro do tipo B é segmentado em regiões baseado na textura contida nela.
- Cada região é classificada em: estática, dinâmica ou não texturizada.
- As regiões não-texturizadas são codificadas com o H.264 sem processamento.
- As regiões estáticas são processadas por um modelo iterativo de perspectiva de movimento chamado *warping* de textura.
- As regiões dinâmicas são analisadas e sintetizadas usando o modelo auto-regressivo de síntese de textura. Realiza-se apenas nos 2° e 4° quadros do GOP, as regiões sintetizadas após são avaliadas.

- Cada região reconstruída ou sintetizada é avaliada, determinando os blocos falsos que serão codificados pelo H.264 e os blocos que serão parte da informação lateral.

Os blocos pertencentes às regiões reconstruídas geram informação lateral, as quais contêm mapas das regiões estáticas e dinâmicas e parâmetros do *warping* de textura. A informação lateral é codificada por um codificador de entropia para reduzir a taxa de bits e tal informação é enviada em conjunto com o bitstream dos dados codificados pelo H.264 ao canal.

3.2.2 Esquema do Decodificador

A reconstrução da sequência do vídeo é feita como é ilustrado na Figura 3.6. O bitstream recebido pelo canal é decodificado pelo decodificador do H.264, obtendo os quadros chave e as regiões não-texturizadas. Os mapas das regiões texturizadas e os parâmetros de *warping* são obtidos pela decodificação da informação lateral. Com os dados decodificados (quadros chave e mapas) as regiões estáticas são reconstruídas. Finalmente, é realizada a síntese de textura para reconstruir as regiões dinâmicas. As regiões são recombinadas para gerar o vídeo reconstruído.

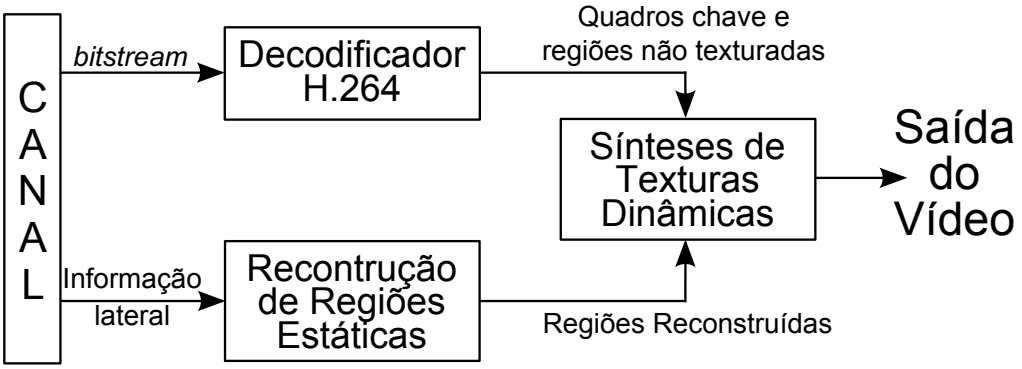


Figura 3.6: Esquema de decodificação.

3.3 Segmentação de Texturas

Cada quadro do tipo B pertencente ao GOP é segmentado em pequenas regiões homogêneas usando a técnica de *watershed* e com os Filtros de Gabor para obter as diferentes características de texturas [52].

Normalmente, a aplicação diretamente do *watershed* sobre a imagem gera sobre-segmentação, que é causada pela presença de ruido. Os efeitos de texturas ou de objetos presentes pouco relevantes que não se desejam segmentar provocam a aparição de falsos contornos associados com vales ou zonas de depressão que não correspondem com objetos ou regiões.

Para solucionar esta sobre-segmentação pode-se eliminar os contornos irrelevantes ou aplicar filtragens antes de aplicar o *watershed*. Neste trabalho, se aplica primeiro o filtragem de Gabor [53] para realçar as características da imagem após *watershed*. Final-

mente um método simples para reduzir a sobre-segmentação das regiões é realizado a partir dos seguintes passos:

1. Primeiro, é realizada a extração das características pela aplicação do filtro de Gabor à imagem de entrada como mostrado na Figura 3.8, que é o resultado da aplicação do filtro sobre a imagem da Figura 3.7. O filtro de Gabor usado é bidimensional com simetria par, o qual executa uma filtragem passa-banda através da orientação ϕ e um filtragem passa-banda ortogonal à orientação ϕ [54].

Figura 3.7: Segundo quadro pertencente ao vídeo Coastguard.

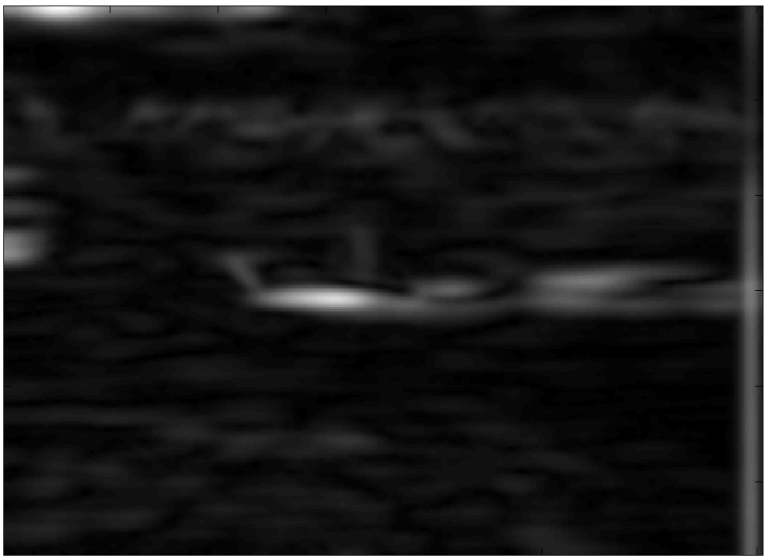


Figura 3.8: Aplicação do filtro de Gabor.

2. É aplicada a transformada de *watershed* [36] na imagem filtrada. O resultado é mostrado na Figura 3.9.

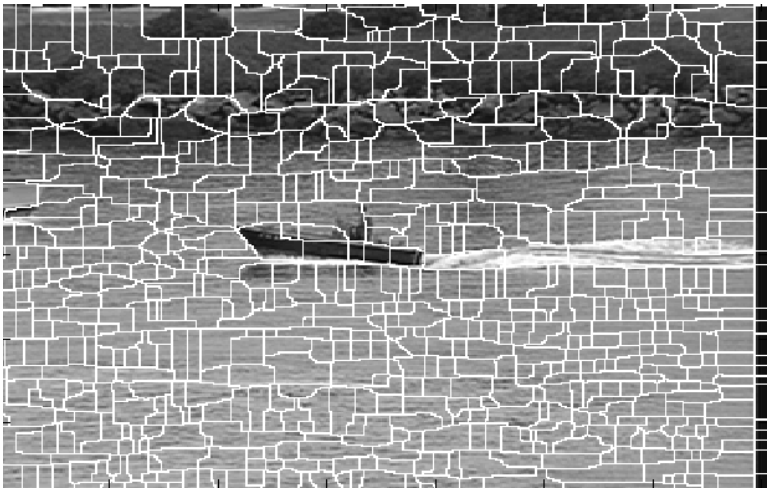


Figura 3.9: Aplicação da transformada de *watershed*. Cada região é mostrada em uma diferente escala de tons de cinza.

3. Para resolver o problema da sobre-segmentação mostrado na Figura 3.9, é implementado um algoritmo para fundir regiões, que é baseado na média dos valores dos *pixels* contidos em cada região. Os passos do algoritmo são:

   (a) Para cada região se acha a média do valor dos *pixels* contidos na mesma.

   (b) A média de uma região é comparada com a média das regiões vizinhas.

   (c) Se a diferença absoluta das médias de duas regiões se encontra abaixo de um limiar Lim_{mindif}, então as regiões são fundidas.

   (d) O crescimento do tamanho da região fundida é limitado por o limiar Lim_{cresc}, que é um porcentagem do tamanho do quadro. A limitação do tamanho é pelo fato que uma região de grande tamanho terá mais processamento.

   O resultado da fusão das regiões mostrado na Figura 3.10.

Figura 3.10: Resultado da fusão das regiões sobre-segmentadas. Cada região fundida é mostrada em uma diferente escala de tons de cinza.

4. Finalmente, para os processamentos posteriores de cada região, é necessário trabalhar em blocos; por exemplo blocos de 16×16 *pixels*. No caso em que se tem diferentes *pixels* pertencentes a regiões distintas em um bloco, este bloco é considerado como parte da região que tenha mais *pixels* dentro do referido bloco. O resultado final é mostrado na Figura 3.11, onde as regiões segmentadas seguem as fronteiras dos blocos de 16×16 *pixels*.

Figura 3.11: Resultado da segmentação em blocos de 16×16 *pixels*. Cada região é mostrada em uma diferente escala de tons de cinza.

3.4 Classificação de Regiões

As regiões segmentadas dos quadros do tipo B são classificadas em três tipos: regiões estáticas, regiões dinâmicas e não-texturizadas. As regiões texturizadas (estáticas ou dinâmicas) contêm componentes de mais alta frequência em comparação a regiões suaves [9].

Cada região pode ser classificada como texturizada ou não texturizada com base nos componentes de alta frequência da maioria dos blocos contidos numa região. Para determinar os componentes de alta frequência é usado a transformada DCT. Os passos da classificação de são:

- Dividir a imagem original em blocos de 16×16 *pixels*, denominados $M_{16\times16}$.
- Dividir o bloco $M_{16\times16}$ em quatro blocos de 8×8 *pixels*, denominados $M_{8\times8}$.
- Aplicar a transformada DCT em cada bloco $M_{8\times8}$.
- Do bloco transformado considera-se apenas a soma dos componentes AC maiores que um limiar Lim_{AC}. Lembrando que os componentes AC representam as altas frequências da transformada DCT. Sendo AC_k os coeficientes AC de M_{8x8}, a soma é dado por:

$$Sum_{8\times8} = \sum_{k} (|AC_k| > Lim_{AC}). \tag{3.1}$$

- Acha-se a média das somas achadas dos quatro blocos pertencentes ao bloco $M_{16\times16}$, denominada Med_{AC}.

  A média achada para cada bloco de 16×16 *pixels* da imagem original representa o valor dos componentes de alta frequência no bloco referido. Os passos anteriores podem ser entendidos com ajuda da Figura 3.12. Um exemplo deste resultado é mostrado na Figura 3.13, demonstrando as médias achadas do quadro do vídeo 'Coastguard' (Figura 3.7).

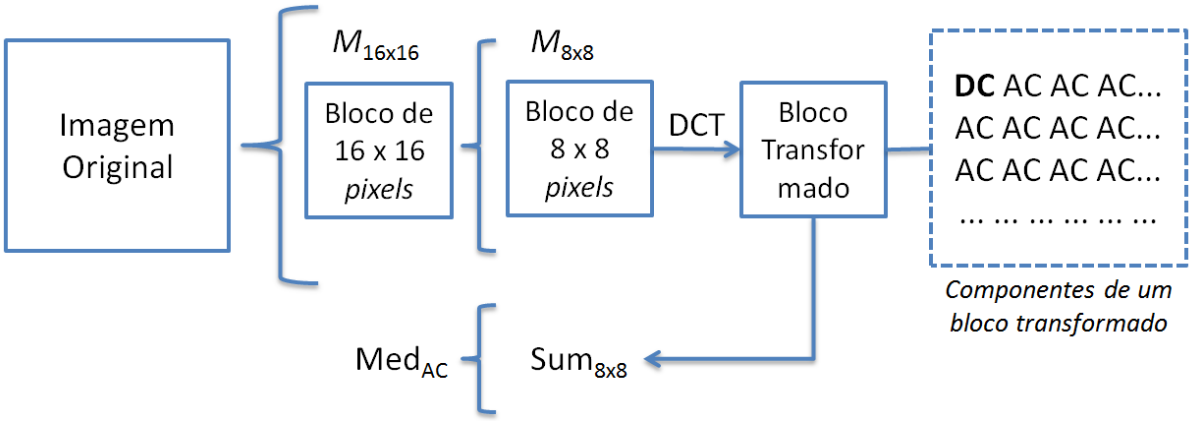


Figura 3.12: Passos para achar a média de cada bloco de 16×16 *pixels* da imagem original.

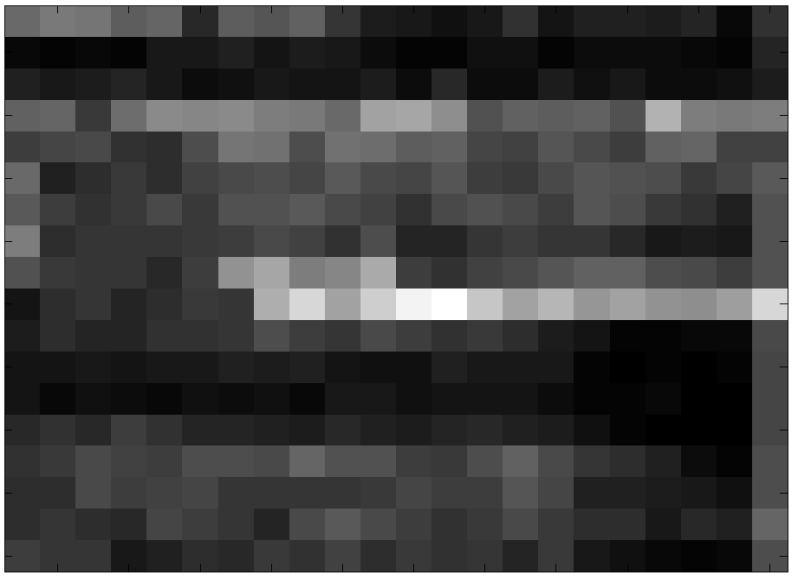


Figura 3.13: Componentes AC para classificação, transformação do segundo quadro do vídeo 'coastguard'.

- Cada Med_{AC} achada da imagem original é relacionado a um bloco de 16×16 *pixels* da imagem segmentada correspondentemente. Desse jeito, em uma região dada R da imagem segmentada, é contabilizado o número de médias Med_{AC} menores a um limiar Lim_{TEX}, denominado Num_{TEX}.

- Finalmente, de acordo com a maioria das médias pertencentes a uma região segmentada se define se a região é texturizada o não, ou seja, se o número Num_{TEX} representa a maioria do número de blocos na região R, então R é considerado como texturizada. Esta condição de textura é dada por:

$$\frac{Num_{TEX}}{Num_R} > \rho, \tag{3.2}$$

  onde Num_R é o número de blocos de 16×16 *pixels* pertencentes a R e ρ é o porcentagem que limita a condição de textura.

As regiões que não sejam classificadas como texturizadas são codificadas pelo H.264. As regiões texturizadas são classificadas em estáticas (movimento simples) e dinâmicas (movimento complexo) de acordo com análises de movimento. A classificação é realizada com base em simples diferenças entre quadros vizinhos, já que se o movimento é baixo então o resíduo também será mínimo. Para isso é feito os seguintes passos:

- Para o quadro original é realizado uma diferença absoluta com seus dois quadros vizinhos, obtendo dois quadros-diferença.
- Cada quadro-diferença é dividido em blocos de 16×16 *pixels*, e em cada bloco são somados os valores dos *pixels* contidos, seja esta soma para um bloco dado Sum_{dif}.
- Em um quadro-diferença, cada Sum_{dif} achado é relacionado a um bloco de 16×16 *pixels* da imagem segmentada correspondentemente. Desse jeito, em uma região texturizada dada R_T, é contabilizado o número de somas Sum_{dif} menores a um limiar Lim_{EST}, denominado Num_{EST}.
- É escolhido o maior dos dois Num_{EST} achados.
- Finalmente, de acordo com a maioria de blocos diferença pertencentes a uma região texturizada se define se a região é estática ou dinâmica, ou seja, se o maior número Num_{EST} representa a maioria do número de blocos na região R_T, então R_T é considerada como estática e no caso contrario é considerada de textura dinâmica. Esta condição de textura estática é dada por:

$$\frac{Num_{EST}}{Num_{R_T}} > \sigma, \tag{3.3}$$

  onde Num_{R_T} é o número de blocos de 16×16 *pixels* pertencentes a R_T e σ é o porcentagem que limita a condição de textura estática.

Um exemplo da classificação de textura é mostrado na Figura 3.14. O resultado mostra como regiões não-texturizadas (área azul) a zona do barco e parte das pedras que contêm alta variedade de textura. Como regiões estáticas estão a parte da grama e grande parte da água (área verde). Finalmente, como regiões dinâmicas estão as partes da água que mudaram de valor (*pixels*) de um quadro para outro (área vermelha).

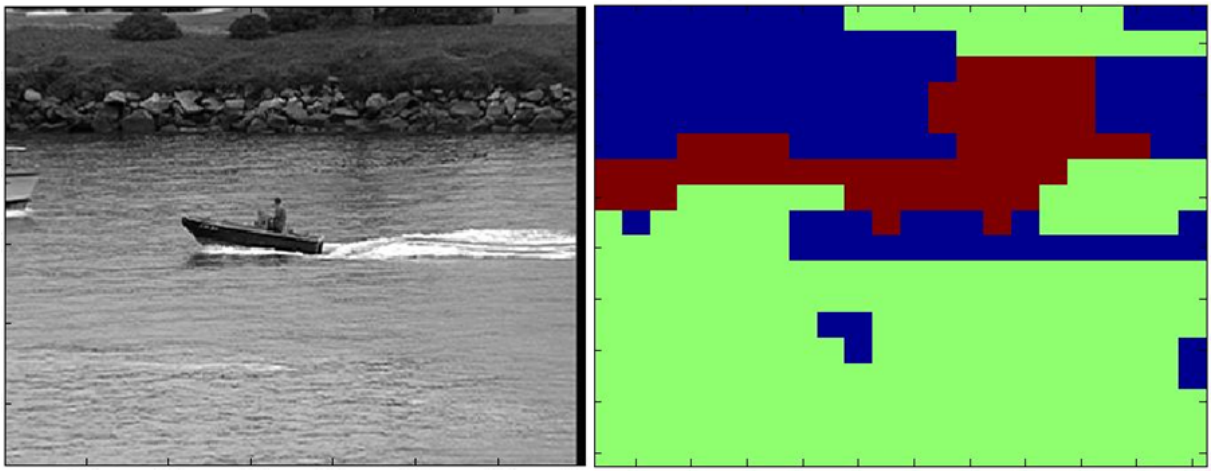


Figura 3.14: Resultado da classificação. Imagem da esquerda é o segundo quadro do vídeo 'coastguard'. Na direita, a área azul representa a região não texturizada, a região verde indica as regiões estáticas e a região vermelha indica a região dinâmica.

3.5 Warping de Regiões Estáticas

Em algoritmos modernos de compressão, a estimação de movimento (baseado em um modelo de movimento dado) é frequentemente usado para explorar a decorrelação temporal e assim reduzir a redundância entre os quadros do vídeo. A estimação de movimento entre duas imagens é considerado como uma transformação projetiva bidimensional com a suposição de que as imagens são capturados por todas as câmaras em perspectiva [9]. Neste trabalho, é empregado o *warping* como um modelo linear translacional baseado em blocos.

O *warping* de textura bidirecional é aplicado para cada região classificada como estática nos quadros do tipo B, baseado no modelo de oito parâmetros [9]. O modelo de perspectiva de movimento é dado por:

$$\begin{cases} x' = (a_1 + a_2 x + a_3 y)/(1 + a_7 x + a_8 y), \\ y' = (a_4 + a_5 x + a_6 y)/(1 + a_7 x + a_8 y), \end{cases} \tag{3.4}$$

onde (x, y) e (x', y') são pontos correspondentes entre o quadro atual e o quadro de referência (quadro chave). Para computar os parâmetros do modelo de perspectiva ($\mathbf{a} = [a_1, a_2, a_3, a_4, a_5, a_6, a_7, a_8]^T$) é usado o método de mínimos quadrados [55].

O *warping* consiste em três passos: coletar dados, computar os parâmetros de movimento e reconstruir as regiões. Como nosso modelo é bidirecional, é feito o *warping* de cada região com referência aos dois quadros chave do GOP, escolhendo só um deles para sua reconstrução. A vantagem do *warping* de textura é que uma região pode ser reconstruída a partir dos parâmetros de movimento e o quadro de referência. Na codificação são realizados os três passos. Na decodificação, só é realizado o passo da reconstrução das regiões.

3.5.1 Coletar dados

São necessários quatro pares de pontos conhecidos para computar os oito parâmetros do modelo. É feita a estimação de movimento convencional com métrica MSE de quatro blocos pertencentes a uma região. A região deve ter no mínimo quatro blocos, de acordo com o passo de classificação das regiões. É assumido que cada vetor de movimento representa o movimento do ponto central do bloco estimado.

3.5.2 Computando os parâmetros de movimento

Com os quatro pares de pontos, os parâmetros de movimento são achados com os seguintes passos: a equação (3.4) é re-escrita como:

$$\begin{cases} a_1 + a_2x + a_3y - xx'a_7 - x'ya_8 = x', \\ a_4 + a_5x + a_6y - xy'a_7 - yy'a_8 = y'. \end{cases} \tag{3.5}$$

ou

$$\mathbf{Pa} = \mathbf{q} \tag{3.6}$$

onde

$$\mathbf{P} = \begin{bmatrix} 1 & x_1 & y_1 & 0 & 0 & 0 & -x_1x'_1 & -x'_1y_1 \\ 0 & 0 & 0 & 1 & x_1 & y_1 & -x_1y'_1 & -y_1y'_1 \\ 1 & x_2 & y_2 & 0 & 0 & 0 & -x_2x'_2 & -x'_2y_2 \\ 0 & 0 & 0 & 1 & x_2 & y_2 & -x_2y'_2 & -y_2y'_2 \\ . & . & . & . & . & . & . & . \\ . & . & . & . & . & . & . & . \\ . & . & . & . & . & . & . & . \end{bmatrix}, \tag{3.7}$$

e

$$\mathbf{q} = \begin{bmatrix} x'_1 & y'_1 & x'_2 & y'_2 & \dots \end{bmatrix}^T. \tag{3.8}$$

Baseado no método de mínimos quadrados, os parâmetros de movimento podem ser computadas invertendo 3.6:

$$\mathbf{a} = \left(\mathbf{P}^T\mathbf{P}\right)^{-1}\mathbf{P}^T\mathbf{q} \tag{3.9}$$

Para comprovar que os parâmetros de movimento foram computados com precisão pode ser usado um método iterativo para remover os pontos defeituosos. Depois de obter o vetor $\mathbf{a}$, pode-se gerar o vetor $\hat{\mathbf{q}}$ a partir da equação (3.4). Logo, são comparados os pontos dos vetores $\hat{\mathbf{q}}$ e $\mathbf{q}$, achando a maior diferença entre eles. Finalmente essa diferença e limitada por um limiar Lim_a.

$$Max(\left\{(\hat{x} - x')^2 + (\hat{y} - y')^2\right\} < Lim_a, (\hat{x}, \hat{y}) \in \hat{\mathbf{q}}. \tag{3.10}$$

No caso em que a maior diferença esteja abaixo do limiar Lim_a, os pontos escolhidos são usados. Caso contrário, são escolhidos novos pontos em um número limitado de tentativas até que os parâmetros do modelo sejam satisfeitos ou não haja mais pontos para escolher. Neste ultimo caso a região não é processada. Quando se faz a nova escolha dos pontos, para cada ponto é feito a estimação de movimento salvando os vetores de movimento caso o ponto seja escolhido novamente. No pior caso, faz-se a estimação de movimento de todos os blocos de 16×16 *pixels* pertencentes à região.

3.5.3 Reconstrução de regiões estáticas

A reconstrução de uma região pode ser realizada conhecendo-se o vetor **a**, os blocos da região estática e o quadro chave de referência. Todos os blocos da região podem ser reconstruídas com a equação (3.4), substituindo os parâmetros de movimento achados (**a**) e os valores dos pontos (x, y) pelas posições dos blocos a reconstruir, achando assim as posições (x', y') dos blocos no quadro de referência. Um exemplo da reconstrução de uma região é mostrado na Figura 3.15, a qual mostra uma região original e a mesma região reconstruída por meio do *warping* de textura. A região estática original pertence ao segundo quadro do vídeo 'container', e foi reconstruída com referência ao primeiro quadro chave que é o primeiro quadro da mesma sequência.

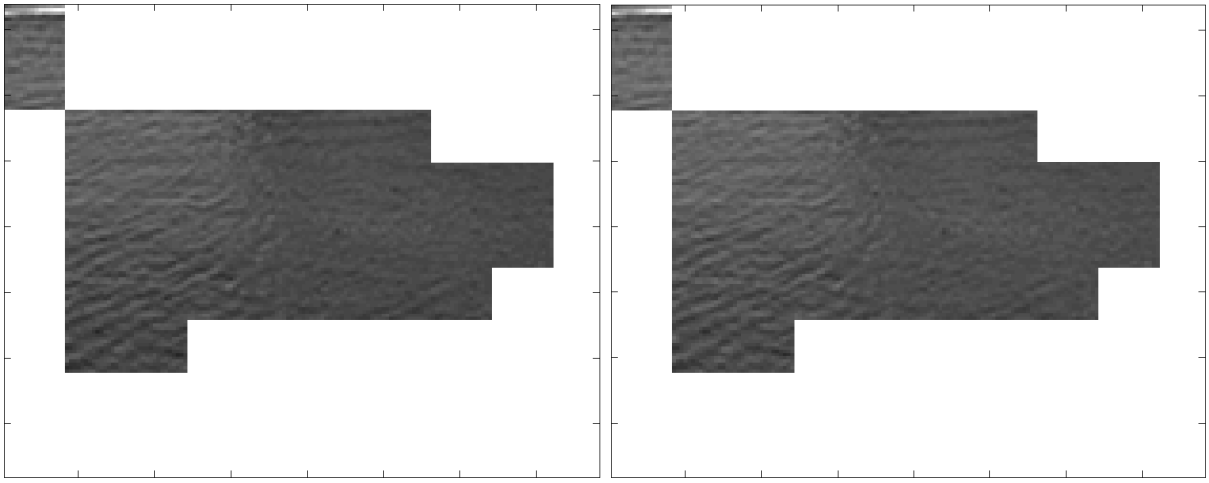


Figura 3.15: Resultado do *warping* de textura. Imagem da esquerda é uma região original classificada como estática pertencente ao segundo quadro do vídeo 'container'. Na direita, a região reconstruída por *warping* de textura.

A reconstrução mediante o *warping* de textura é realizada tanto na codificação como na decodificação para o resultado final. Na codificação é realizada para avaliar os blocos reconstruídos.

3.6 Síntese de Regiões Dinâmicas

Para realizar a síntese de textura, é usado o método de síntese de Zhang e Bull [9], baseado no trabalho original de Doretto [44], onde as texturas dinâmicas são definidas como saída de um processo auto-regressivo de movimento médio. O método é descrito pela Equação (3.11).

$$\begin{cases} \mathbf{X}(t+1) = \mathbf{A}\mathbf{X}(t) + \mathbf{B}\mathbf{V}(t) \\ \mathbf{Y}(t) = \mathbf{C}\mathbf{X}(t) + \mathbf{Y}_{media} \end{cases}, t = 1, 2, ...N. \tag{3.11}$$

onde $\mathbf{Y}$ representa a sequência de vídeo com N imagens, onde por exemplo $\mathbf{Y}(1)$ é o primeiro quadro do vídeo. $\mathbf{Y}_{media}$ é a média dos N quadros de treinamento. $\mathbf{X}$ é considerado como a entrada do modelo, $\mathbf{X}(1)$ seria a entrada inicial do modelo. $\mathbf{A}$, $\mathbf{B}$ e $\mathbf{C}$

são matrizes paramétricas e $\mathbf{V}(t)$ é assumido como uma variável aleatória independente e identicamente distribuída. Com este sistema cada imagem $\mathbf{Y}(t)$ pode ser gerado a partir

O método para reconstruir as regiões classificadas como dinâmicas pode ser divido em dois passos: treinamento do sistema e reconstrução das texturas dinâmicas. A reconstrução local também é feita na codificação para validar os blocos que teriam os melhores resultados e quais blocos não seriam processados. A validação de tais blocos é feito no passo da avaliação de qualidade.

3.6.1 Treinamento do Sistema

O método de síntese é baseado na suposição de que a variação de textura é temporalmente uniforme com um grupo pequeno de imagens [9], nosso caso, um grupo de cinco imagens (GOP). No GOP, só no segundo e quarto quadros são reconstruídas as regiões dinâmicas, e para sintetizar estes quadros (2° e 4°) é necessário achar as matrizes paramétricas de entrada do modelo (Equação 3.11).

As matrizes paramétricas têm que ser achadas a partir de quadros já codificados, os quais são os quadros chave (1° e 5°) e o 3° quadro com as regiões estáticas reconstruídas. Estes três quadros são analisados por o modelo:

$$\begin{cases} \mathbf{X}(t+2) = \mathbf{A}\mathbf{X}(t) + \mathbf{B}\mathbf{V} \\ \mathbf{Y}(t) = \mathbf{C}\mathbf{X}(t) + \mathbf{Y}'_{media} \end{cases}, t = 1, 3 \text{ e } 5, \tag{3.12}$$

onde $\mathbf{Y}$ representa aos quadros três quadros analisados (1°, 3° e 5°), $\mathbf{Y}'_{media}$ é a média dos quadros $\mathbf{Y}(1)$, $\mathbf{Y}(3)$ e $\mathbf{Y}(5)$.

Para achar a matriz de entrada $\mathbf{X}(1)$ e as matrizes paramétricas $\mathbf{A}$, $\mathbf{B}$ e $\mathbf{C}$, utiliza-se a decomposição em valores singulares (SVD - *Singular Value Decomposition*) [56].

3.6.2 Reconstrução de texturas dinâmicas

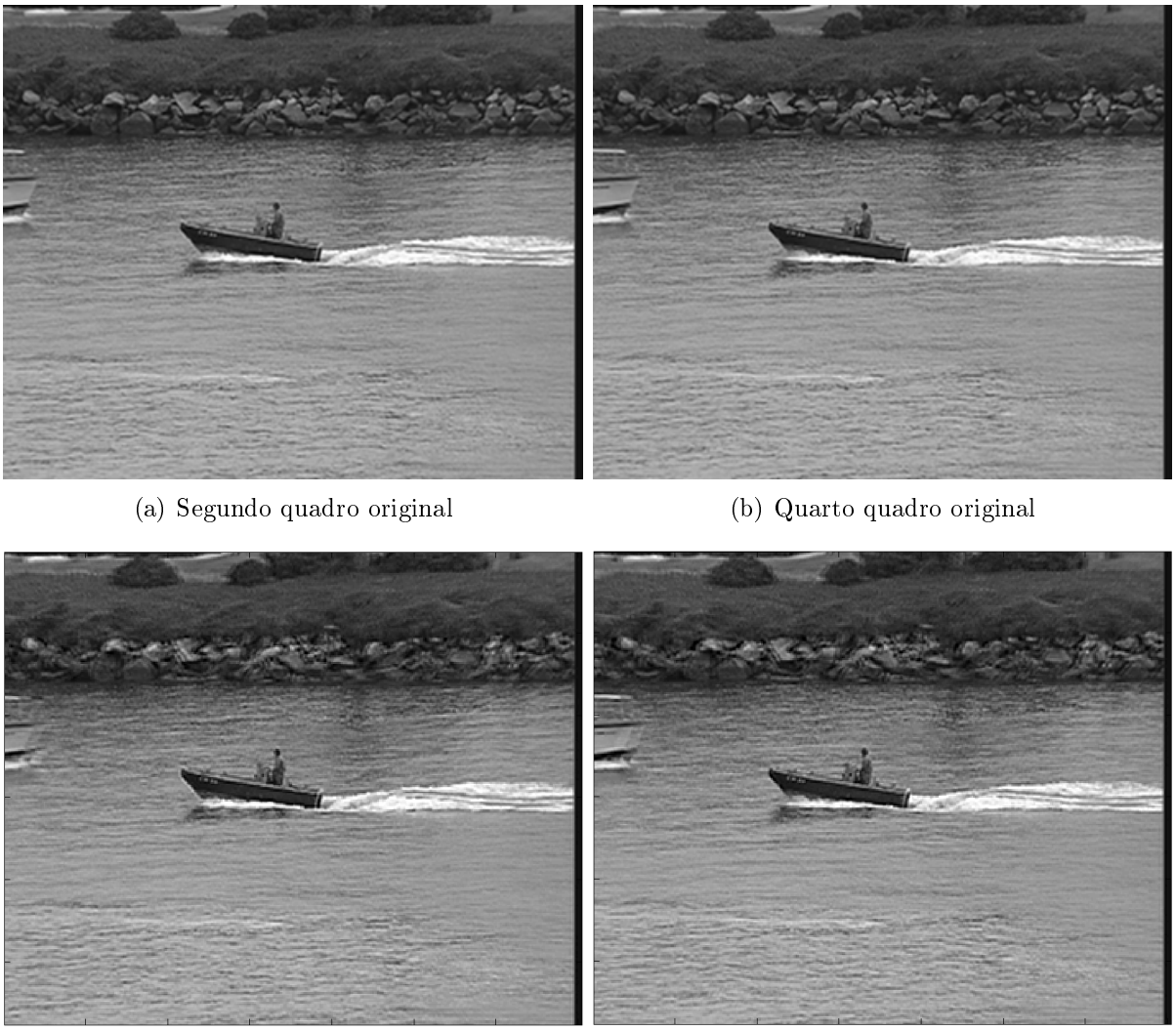
Para realizar a síntese em um grupo de cinco imagens com os dados de entrada $\mathbf{X}(1)$, $\mathbf{A}$, $\mathbf{B}$ e $\mathbf{C}$ obtidos anteriormente, é necessário modificar os valores das matrizes $\mathbf{A}$ e $\mathbf{B}$, já que estas matrizes foram achadas com um grupo de três imagens. Ou seja é necessário alterar a velocidade do sistema com a variação das matrizes paramétricas [56]. Os novos valores para as matrizes serão:

$$\mathbf{A}' = sqrtm(\mathbf{A}), \mathbf{B}' = (\mathbf{A}' + \mathbf{I})^{-1}\mathbf{B}, \tag{3.13}$$

onde a função $\mathbf{X} = sqrtm(\mathbf{A})$ define-se como a principal raiz quadrada da matriz $\mathbf{A}$, isto é $\mathbf{X}\mathbf{X} = \mathbf{A}$. Com as novas matrizes adequadas para realizar a síntese em cinco quadros, é usado o seguinte modelo:

$$\begin{cases} \mathbf{X}(t+1) = \mathbf{A}'\mathbf{X}(t) + \mathbf{B}'\mathbf{V} \\ \mathbf{Y}(t) = \mathbf{C}\mathbf{X}(t) + \mathbf{Y}'_{media} \end{cases}, t = (1, 2, \ldots, 5) \tag{3.14}$$

Assim, apenas de posse das matrizes de entrada conseguimos sintetizar cinco quadros, dos quais só consideramos as regiões dinâmicas do 2° e 4° quadros. Uma comparação entre os quadros originais e os quadros achados a partir do modelo anterior é mostrado na Figura 3.16.



(a) Segundo quadro original

(b) Quarto quadro original

(c) Segundo quadro sintetizado

(d) Quarto quadro sintetizado

Figura 3.16: Exemplo de quadros sintetizados do vídeo 'Coastguard'. As imagens 3.16(a) e 3.16(b) são os 2° e 4° quadros originais respectivamente, e os quadros 3.16(c) e 3.16(d) são os 2° e 4° quadros sintetizados respectivamente.

3.7 Avaliação de Qualidade

Para assegurar que os blocos reconstruídos ofereçam um bom resultado, são realizados diferentes métodos para identificar diferentes artefatos (*blurring*, estimação de similaridade e bordas) presentes no vídeo [9]. Nossa proposta identificamos erros com base nas estatísticas dos blocos reconstruídos nas regiões estáticas e dinâmicas.

Depois de reconstruir as regiões estáticas ou dinâmicas em um vídeo, pode-se perceber uma cintilação nestas regiões, para diminuir esse efeito é calculado a diferença entre as regiões reconstruídas e as regiões originais correspondentes de cada quadro do tipo B reconstruído. São aplicadas duas operações estatísticas para cada bloco de 16×16 *pixels*, para determinar se o bloco sera codificado pelo codec H.264/AVC ou não.

As operações utilizadas são a média e a variância. Se os resultados destas operações

caem baixo de limiares Lim_{med} e Lim_{var} respectivamente, os blocos não são codificados pelo codec H.264/AVC. Estas condições podem ser melhor entendidas com ajuda das comparações na Equação 3.15, onde M_{rec} é o bloco reconstruído e M_{ori} é o bloco original. Com estas condições foram detectados os erros de reconstrução nos blocos avaliados.

$$\begin{cases} média(|M_{ori} - M_{rec}|) & < Lim_{med}, \\ variância(|M_{ori} - M_{rec}|) & < Lim_{var}. \end{cases} \tag{3.15}$$

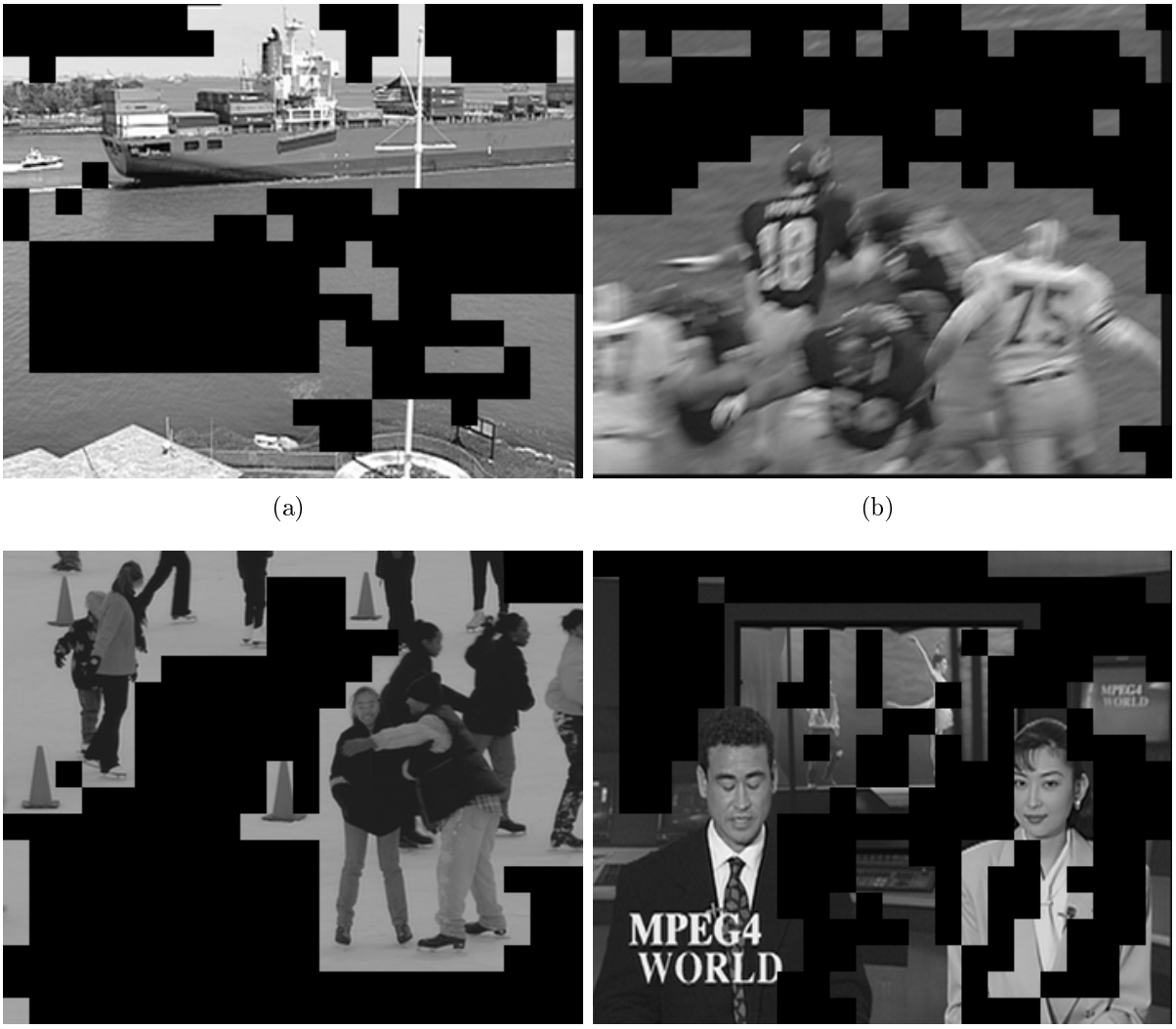
3.8 Codificação

Zhang e Bull [9] realizam a codificação dupla da sequência do vídeo. Na primeira codificação usa-se o padrão H.264 para avaliar o número de bits consumidos por cada bloco. Na segunda codificação usa-se os métodos de *warping* e síntese. Para ganhar em taxa de bits é comparado os bits consumidos pela primeira e segunda codificação, utilizando apenas os blocos onde apresentem menores taxa de bits. Já o esquema proposto realiza apenas uma passagem de codificação, primeiramente realizando o pré-processo nas regiões texturizadas e logo codificando a sequência do video pelo codec H.264.

No padrão H.264 é codificado uma sequência que contem as regiões não-texturizadas e as regiões restantes estão com o valor zero para todos os *pixels* contidos. Então, no codificador do H.264 cada bloco de 16×16 com os valores zero sera codificado na predição INTRA 16×16 no modo DC ou na predição INTER no modo SKIP se o bloco na mesma posição do quadro anterior fosse também zero. Exemplos destas sequências modificadas são mostradas na Figura 3.17, que apresenta um quadro com as regiões não-texturizadas para quatro diferentes vídeos. As regiões com os valores zero são as regiões que foram processadas com *warping* ou síntese de textura. A modificação é realizada a fim de evitar o processamento adicional destas regiões no codificador H.264.

As regiões processadas por *warping* ou síntese geram dados tais como os mapas de posições das regiões e os parâmetros do *warping*, que são codificados com um codificador de entropia, como por exemplo com o ZIP. A informação lateral (arquivo codificado com codificador de entropia) e o *bitstream* gerado pelo codificador do H.264 são enviados em conjunto ao canal usado.

Para gerar a sequência de vídeo original, são decodificados o *bitstream* e a informação lateral por o decodificador do H.264 e o decodificador de entropia respectivamente. Em todos os quadros do tipo B são reconstruídas as regiões estáticas e dinâmicas, que substituem as posições correspondentes na sequência decodificada pelo H.264, gerando o vídeo final.



(a) (b)

(c) (d)

Figura 3.17: Exemplo de quadros que mostram as regiões classificadas como não-texturizadas que serão codificados pelo H.264. As figuras (a), (b), (c) e (d) correspondem ao 2° quadro de 'Container', 78° quadro de 'Football', 14° quadro de 'Ice' e 14° quadro de 'News' respectivamente.

Capítulo 4

Resultados Experimentais

4.1 Configurações dos Experimentos

O esquema proposto foi implementado em MATLAB® e o código utilizado para implementar o padrão H.264 foi o JM 18.4 [57]. Para a avaliação do trabalho, foram usados como sequências de teste as sequências *Coastguard*, *Container*, *Football*, *Ice*, *News* e *Waterfall* de tamanho CIF (resolução 352×288 *pixels*) e as sequências *In to tree* e *Ducktakeoff* de tamanho HD (resolução 1280×720 *pixels*). As sequências de tamanho CIF e HD têm frequência de quadros de 30 Hz e 50 Hz respectivamente, com uma duração de cinco segundos. A Figura 4.1 mostra o primeiro quadro de todas as sequências.

Todos os testes foram executados com perfil de codificação *Main* do H.264/AVC; CAVLC como codificador de entropia e *rate-distortion optimazation*. A estrutura de predição no GOP foi IBBBPBBBPB..., onde o primeiro quadro é de tipo I e após de cada três quadros do tipo B é codificado um quadro do tipo P. Foram usados dois quadros de referência para as predições dos quadros do tipo B e o tamanho da janela de busca é de 16×16 *pixels*. O modo *full search* para quarto de *pixel* foi escolhido. Os parâmetros de quantização (QP) usados foram similares aos utilizados no trabalho de Zhang e Bull [9], para as sequências de vídeo 'coastguard' e 'waterfall' variaram de 14 a 32, para as outras sequências CIF variaram de 14 a 34 e para as sequências HD variaram de 14 a 28. Os limiares (definidos empiricamente) usados nos experimentos são mostrados na Tabela 4.1.

Os testes subjetivos foram realizados segundo a metodologia descrita em [58], com menos de 100 pessoas para todos os testes. O mesmo monitor (resolução de 1366×768), mesma iluminação e mesmo contraste foram utilizados em todos os testes. Os QPs para os testes subjetivos são 14, 18 e 22. Foram apresentados os vídeos usando o método *Double Stimulus Continuous Quality Scale*, onde são apresentados ao observador dois vídeos ao mesmo tempo: um vídeo codificado pelo H.264 e o outro codificado e reconstruído com o modelo apresentado. O tempo de apresentação foi de 20 segundos para cada sequência de vídeo, mais um tempo adicional para a atribuição das notas. O ângulo máximo de visão da tela para o observador da foi menor de 30°. Uma escala de cinco níveis de qualidade foi usada. A Tabela 4.2 apresenta os conceitos referentes a cada nível de qualidade que foi atribuído ao vídeo sendo testado. Para o processamento das notas foi usado o MOS.

(a) Coastguard (b) Container (c) Football

(d) Ice (e) News (f) Waterfall

(g) In to tree (h) Ducktakeoff

Figura 4.1: Primeiro quadro das sequências de teste. (a), (b), (c), (d), (e) e (f) de tamanho CIF; (g) e (h) de tamanho HD.

Tabela 4.1: Limiares usados nos experimentos.

Limiar	Valor	Uso do limiar
ϕ	45°	Orientação do filtro de Gabor
Lim_{mindif}	12	Diferença das regiões para fundir
Lim_{cresc}	10%	Crescimento da região em relação ao quadro
Lim_{AC}	10	Componentes AC
Lim_{TEX}	750	Médias de componentes AC
ρ	75%	Condição de textura
Lim_{EST}	1000	Diferença de blocos estáticos
σ	60%	Condição de textura estática o dinâmica
Lim_a	0.01	Validação do parâmetros do vetor a
Lim_{med}	8	Condições de uso do bloco
Lim_{bor}	120	

Tabela 4.2: Conceito para avaliação.

Conceito	Qualidade
5	Excelente
4	Boa
3	Razoável
2	Pobre
1	Ruim

4.2 Complexidade da Codificação

Um bloco de tamanho $N \times N$ *pixels*, assumindo uma janela de busca de tamanho $M \times M$ *pixels*, realiza $N^2 \times M^2$ operações durante a busca completa. Esta operação deve ser repetida para cada partição e subpartição do macrobloco na codificação do H.264. Por outro lado, para cada quadro B, todas essas operações devem ser repetidos em pelo menos dois quadros de referência e devem ser testados todos os modos INTRA possíveis. Esta complexidade pode ser reduzida utilizando técnicas de estimação rápida de movimento [59] [60] e realizando implementações que re-utilizam cálculos de partições menores para realizar a estimação de partições maiores [61].

Em nossa proposta, cada quadro do tipo B e segmentado com *watershed* que é um método iterativo. Após cada região é classificada em texturizada e não-texturizada mediante a transformada DCT de blocos de 8×8 *pixels*. A classificação entre textura estática ou dinâmica é feita com a simples avaliação de diferenças e médias dos blocos, as quais introduzem pouca complexidade computacional. Dos blocos pertencentes a uma região classificada como textura estática, apenas realiza-se a estimação de movimento em quatro blocos (escolhidos aleatoriamente) de tamanho 16×16 e sem subpartições. Com os quatro vetores de movimento achados na estimação resolve-se um sistema de equações mediante o método dos mínimos quadrados, o qual tem uma complexidade significativamente inferior a qualquer método de estimação de movimento com subpartições. A estimação de movimento dos blocos selecionados é realizado no pior caso para todos os blocos pertencentes à

região em questão. No caso da regiões classificadas como dinâmicas, a síntese é realizada sem estimação de movimento, já que apenas é necessário saber a localização das regiões e temos a informação dos quadros vizinhos para a síntese de tais regiões. No passo da avaliação dos blocos reconstruídos, são utilizados operações de baixa complexidade como o cálculo de média e variância do bloco, sem passar por qualquer tipo de filtragem de maior complexidade, em contraste com o padrão H.264, onde é utilizado um filtro redutor do efeito de bloco. Finalmente, as regiões processadas aumentam a complexidade no decodificador com as reconstruções feitas nelas.

Para comparar as complexidades do esquema proposto e o padrão H.264, foi implementado na mesma ferramenta (MATLAB®) o esquema proposto e a estimação de movimento do H.264, contabilizando nas mesmas condições os tempos de processamento que são mostrados na Tabela 4.3. Onde mostra o porcentagem de blocos reconstruídos e o tempo em segundos de processamento (codificação e decodificação) de um GOP (IBBBP). Enquanto ao padrão H.264, mostra o tempo da estimação de movimento de três quadros do tipo B inteiros, com referencia a dois quadros, para blocos de 16×16, 16×8, 8×16 e 8×8 *pixels*. Além disso mostra o tempo de estimação proporcional ao porcentagem de blocos reconstruídos pelo esquema proposto. Finalmente, $\triangle$Tempo é a diferença entre o tempo do H.264 (Blocos reconstruidos) e o tempo do GOP.

Tabela 4.3: Comparação de tempos de processamento .

Sequências	Esquema Proposto		Estimação no H.264		$\triangle$Tempo (H.264 - GOP) (s)
	Blocos Reconstruídos %	Tempo GOP (s)	Quadros B Inteiro (s)	Blocos Reconstruídos (s)	
coastguard(CIF)	15.66%	8.92	127.11	19.91	10.99
container(CIF)	33.59%	44.04	127.54	42.84	-1.20
football(CIF)	15.64%	51.68	127.77	19.98	-31.70
ice(CIF)	35.94%	42.43	127.46	45.81	3.38
news(CIF)	42.92%	56.41	126.96	54.49	-1.92
waterfall(CIF)	26.44%	83.51	127.71	33.77	-49.74
in to tree(HD)	16.11%	239.89	1167.48	188.08	-51.81
ducktakeoff(HD)	18.34%	164.44	1162.81	213.26	48.82
Média	25.58%	86.42	386.86	77.27	-9.15

Considerando apenas os tempos da estimação de movimento, os tempos referentes ao esquema proposto apresentam um potencial para reduzir o tempo de codificação com relação ao padrão H.264, já que a diferença média mostra que a estimação é superior em apenas 9.15 segundos. No entanto, não foi considerado outros tempos no H.264, como a estimação realizada no caso que fosse selecionado como melhor opção o bloco de tamanho 8×8, onde é realizado a estimação em suas subpartições (8×4, 4×8 e 4×4 *pixels*) com precisão de quarto de *pixel*, e também, não é considerado o tempo de outras operações realizadas no codificador e decodificador (como os modos INTRA, as transformadas, a quantização e o filtro).

4.3 Análise dos Resultados

Uma comparação entre os quadros decodificados pelo H.264 e o esquema proposto é mostrada nas Figuras 4.2, 4.3 e 4.4, onde também são mostradas as regiões não codificadas pelo H.264. As informações estruturais são fatores importantes para a percepção visual [9]. Em nosso esquema, considera-se que as informações estruturais têm componentes de alta frequência e são classificadas como não-texturizadas. Assim, objetos estruturais tais como barcos em 'coastguard', o navio em 'container', jogadores em 'football', as pessoas em 'ice' e 'news', a casa em 'in to tree' e os patos em 'ducktakeoff' são codificados pelo padrão H.264. Na Tabela 4.4 é mostrado os porcentagens dos blocos reconstruídos utilizando *warping* e síntese de textura do esquema proposto. Os porcentagens foram contabilizados apenas com blocos pertencentes aos quadros do tipo B.

Tabela 4.4: Porcentagem dos blocos processados dos quadros tipo B.

Sequência	Porcentagem
Coastguard	32.73%
Container	30.45%
Football	12.65%
Ice	53.77%
News	40.79%
Waterfall	11.55%
In to tree	22.11%
Ducktakeoff	15.76%
Média	24.48%

As sequências 'coastguard' e 'container' mostram altos porcentagens de blocos reconstruídos, blocos em maioria como parte da água e céu como mostra as Figs. 4.2(b) e 4.2(e). No entanto, as maiores porcentagens de blocos reconstruídos foram nas sequências 'Ice' e 'News', onde os blocos são parte do gelo ou parede, mostrado respectivamente nas Figs. 4.3(b) e 4.3(e). A partir destas altas porcentagens pode-se dizer que as regiões que não contem muita informação estrutural são reconstruídas pelo esquema proposto. Desse jeito, a baixa porcentagem de blocos reconstruídos em 'Football' reflete o conteúdo da informação estrutural contida, já que o vídeo é focado em jogadores.

As curvas de taxa-distorção são mostradas nas Figuras 4.5 a 4.12, onde são comparados o esquema proposto e o H.264 com base nas métricas objetivas PSNR e SSIM para todas as sequências de teste. Na Tabela 4.5 são apresentados $\triangle PSNR$, $\triangle PSNR\text{-}Rate$, $\triangle SSIM$ e $\triangle SSIM\text{-}Rate$, calculados usando o método Bjontegaard [62]. Quando o valor do $\triangle PSNR$ ou $\triangle SSIM$ calculado é negativo significa que o método proposto não supera o padrão H.264, enquanto o valor do $\triangle PSNR\text{-}Rate$ ou $\triangle SSIM\text{-}Rate$ calculado é positivo significa que o método proposto não supera em taxa de bits ao padrão H.264.

Nas curvas PSNR pode-se observar que um desempenho inferior ao do H.264 normal, como é apresentado na Tabela 4.5, a média do $\triangle PSNR$ é de -1.19 e o $\triangle PSNR\text{-}Rate$ é 53.01%. Tal resultado se dá por que a media do error quadrático introduzida pelo *warping* e pela síntese de textura é significativa, sobretudo nas sequências 'coastguard' e 'news'.

Tabela 4.5: Avaliações Bjontegaard para PSNR e SSIM.

Sequências	$\triangle PSNR$ (dB)	$\triangle PSNR$-*Rate*	$\triangle SSIM$ (dB)	$\triangle SSIM$-*Rate*
coastguard(CIF)	-1.98	86.30%	-0.0084	52.91%
container(CIF)	-1.22	33.57%	-0.0124	28.64%
football(CIF)	-1.77	97.04%	-0.0112	67.79%
ice(CIF)	-0.62	41.50%	-0.0022	66.09%
news(CIF)	-1.47	66.16%	-0.0096	52.76%
waterfall(CIF)	-0.46	12.66%	-0.0069	13.43%
in to tree(HD)	-0.86	18.40%	-0.0023	81.38%
ducktakeoff (HD)	-1.13	68.47%	-0.0019	74.65%
Média	-1.19	53.01%	-0.0069	54.71%

Curvas SSIM também mostram um baixo desempenho para o esquema proposto, como mostra as médias do $\triangle SSIM$ (-0.0069) e $\triangle SSIM$-*Rate* (54.71%) na Tabela 4.5. Mas as diferenças são mínimas enquanto a $\triangle SSIM$, como nas sequências 'ice', 'in to tree' e 'ducktakeoff'. Os resultados negativos nas curvas PSNR e SSIM eram de se esperar, já que estas métricas não se adequam a sistemas de compressão de video baseados em síntese [9].

Os resultados da avaliação subjetiva são apresentados nas Figuras 4.13, 4.14 e 4.15, onde cada figura mostra dois curvas de taxa-distorção correspondentes ao esquema proposto e ao padrão H.264/AVC. As curvas apresentam ganhos de qualidade para o esquema proposto em todas as sequências de teste, embora tais ganhos só sejam alcançados em alguns pontos enquanto em outros poucos há perdas. Na Tabela 4.6 são apresentados $\triangle MOS$ e $\triangle Rate$, calculados usando o método Bjontegaard [62], onde as médias do $\triangle$MOS (0.1395) e $\triangle$Rate (-5.42%) representam uma melhora no método proposto, sendo mínimas estas médias pode-se afirmar que a qualidade subjetiva é mantida.

Tabela 4.6: Avaliação Bjontegaard para MOS.

Sequências	$\triangle$MOS (dB)	$\triangle$Rate
coastguard	0.3662	1.95%
container	0.2333	20.97%
football	0.0527	-59.84%
ice	0.0502	86.46%
news	0.0728	-25.99%
waterfall	0.0620	-56.07%
Média	0.1395	-5.42%

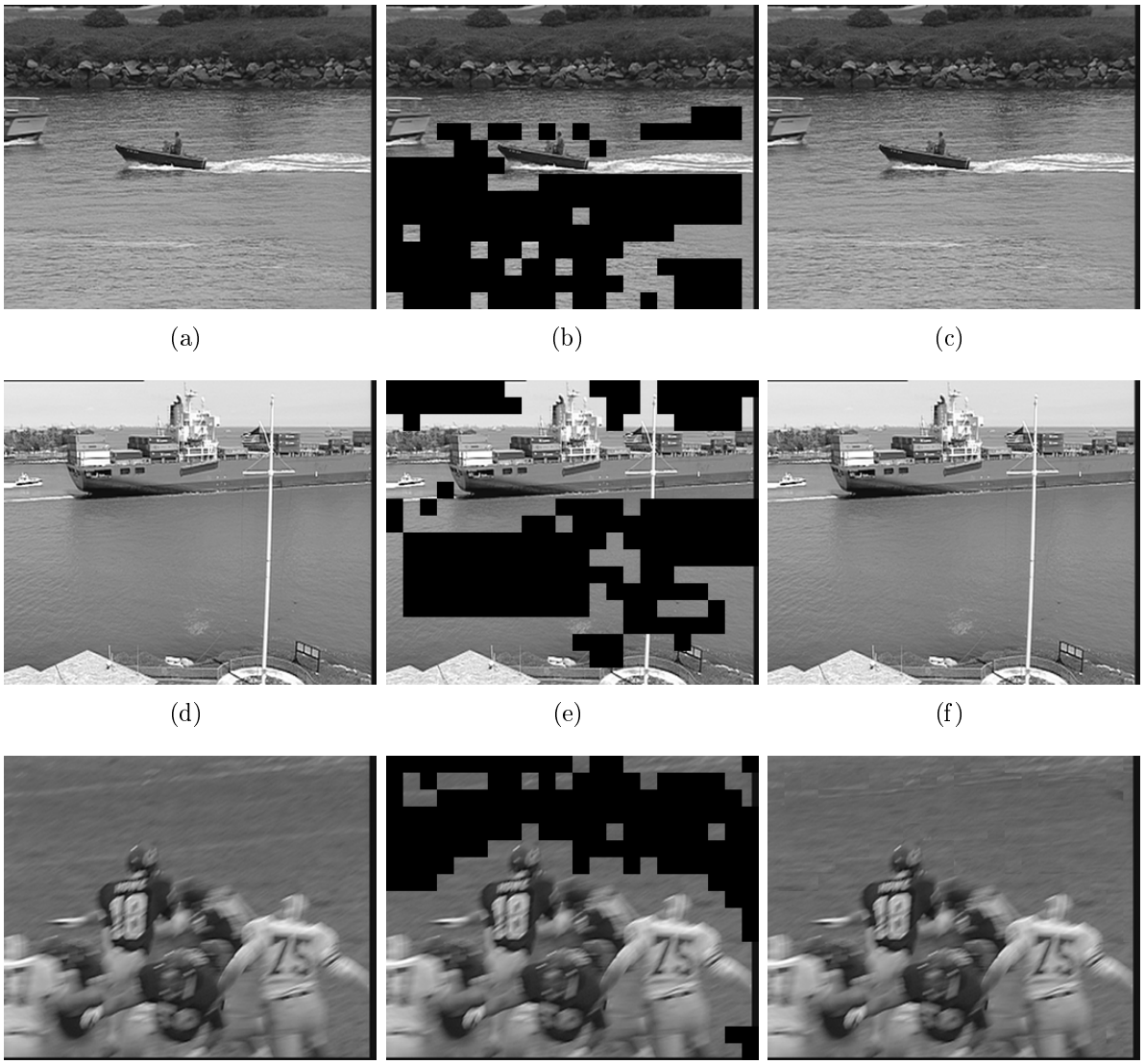


Figura 4.2: Exemplos de quadros reconstruídos por *warping* e síntese. (a), (d) e (g) correspondem aos quadros reconstruídos pelo H.264 para as sequências de video *coastguard*, *container* e *football* respetivamente. (b), (e) e (h) mostram as regiões de preto que são processados pelo esquema proposto destas sequências. (c), (f) e (i) mostram os quadros reconstruídos pelo esquema proposto.

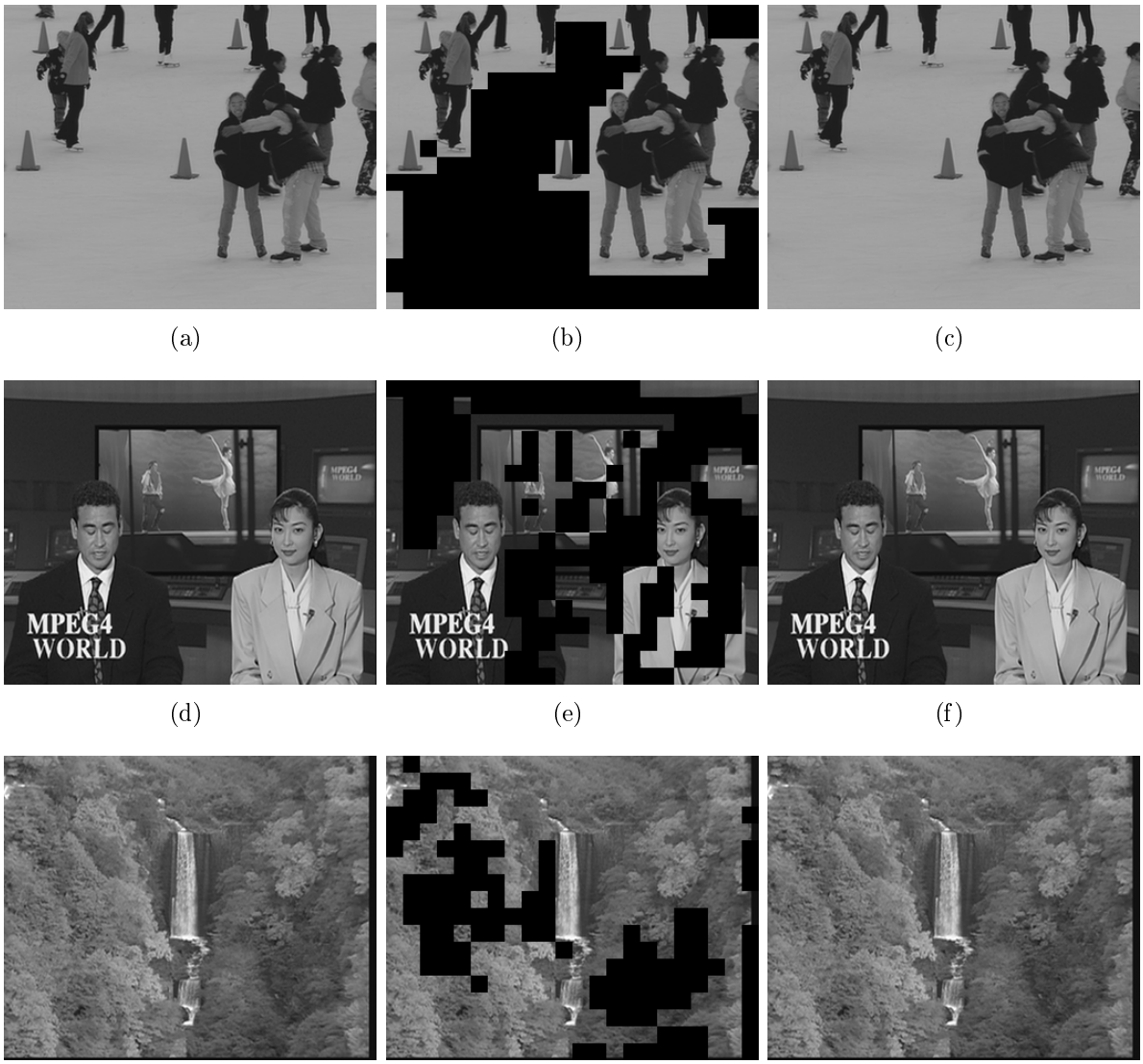
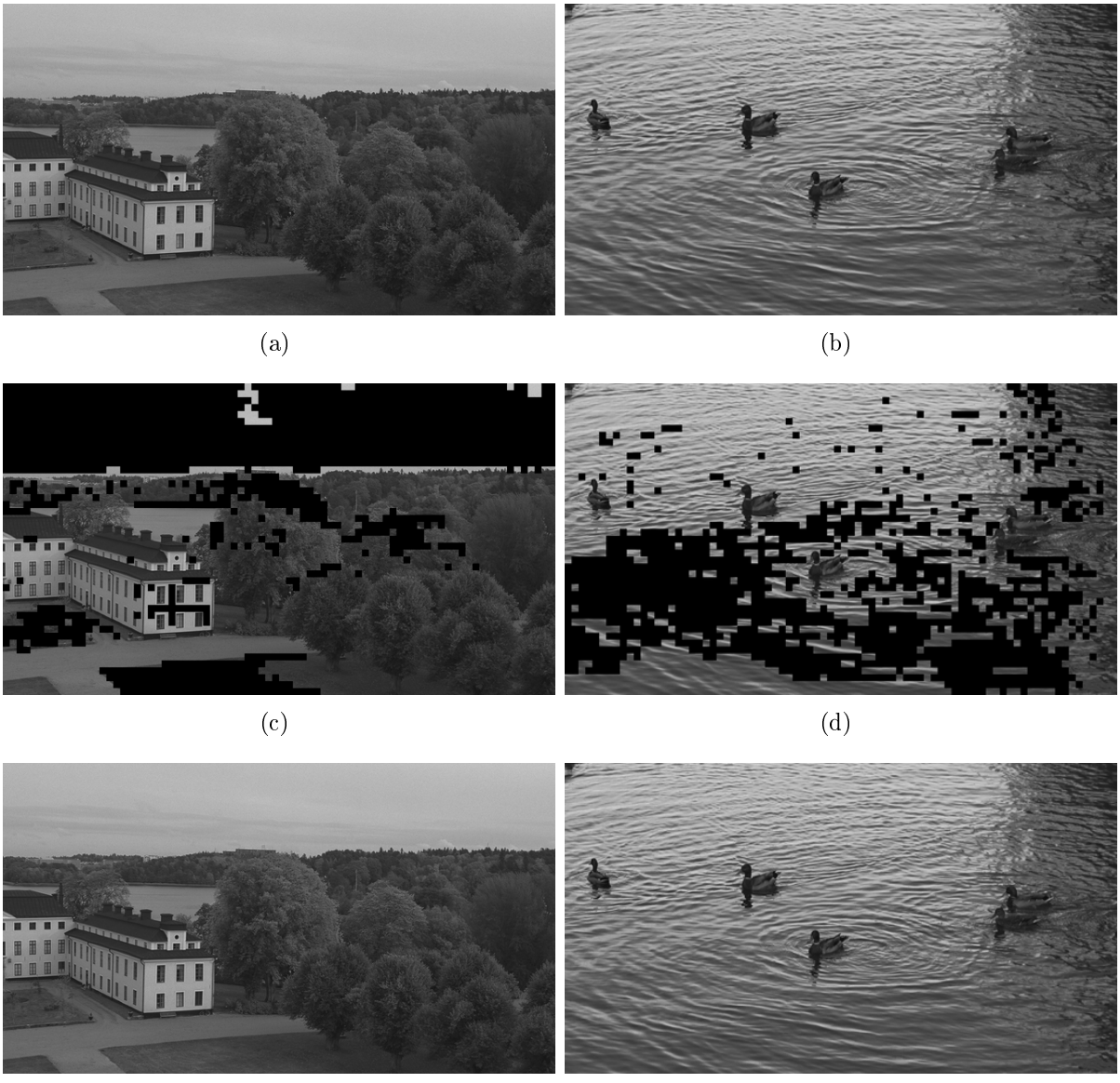


Figura 4.3: Exemplos de quadros reconstruídos por *warping* e síntese. (a), (d) e (g) correspondem aos quadros reconstruídos pelo H.264 para as sequências de video *ice*, *news* e *waterfall* respetivamente. (b), (e) e (h) mostram as regiões de preto que são processados pelo esquema proposto destas sequências. (c), (f) e (i) mostram os quadros reconstruídos pelo esquema proposto.

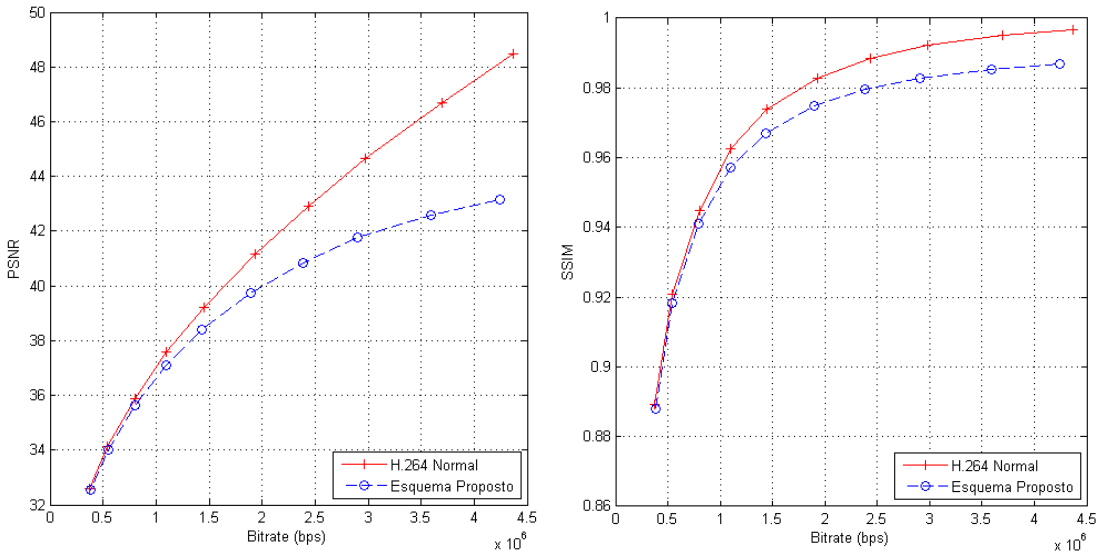


(a) (b)

(c) (d)

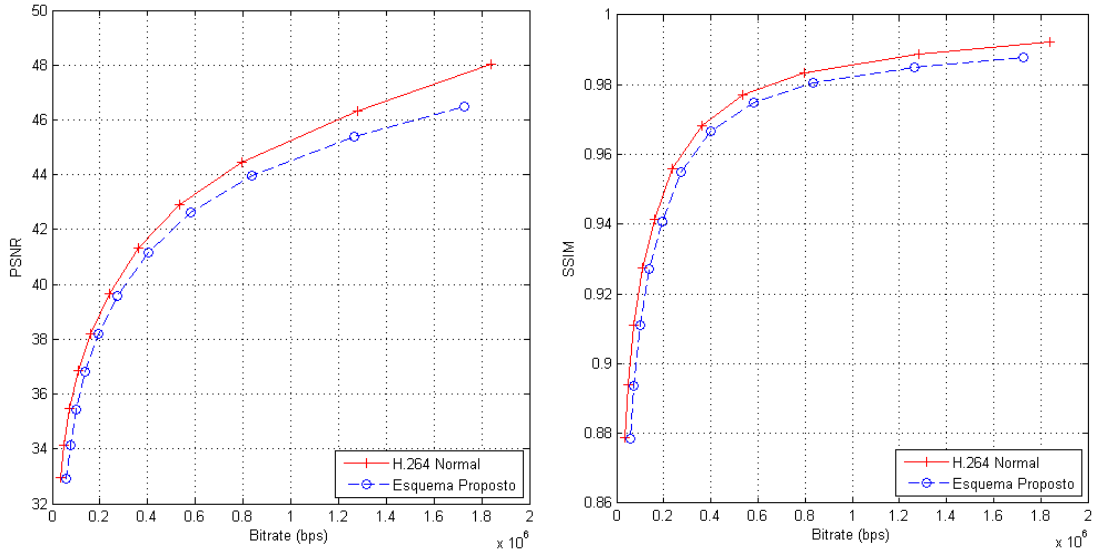
(e) (f)

Figura 4.4: Exemplos de quadros reconstruídos por *warping* e síntese. (a), e (b) correspondem aos quadros reconstruídos pelo H.264 para as sequências de video *in to tree* e *ducktakeoff* respetivamente. (c) e (d) mostram as regiões de preto que são processados pelo esquema proposto destas sequências. (e) e (f) mostram os quadros reconstruídos pelo esquema proposto.



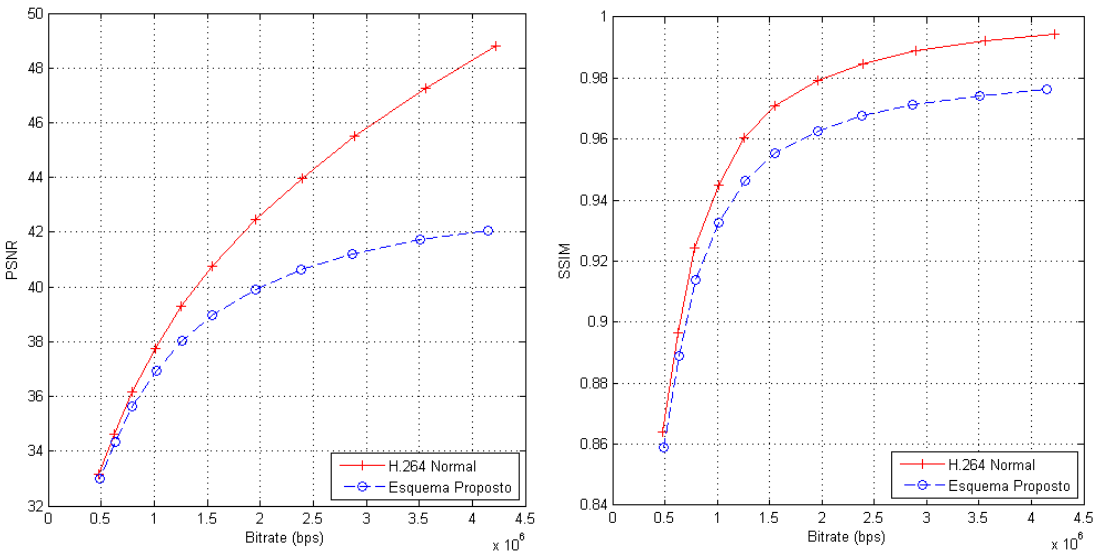
(a) PSNR (b) SSIM

Figura 4.5: Curvas PSNR e SSIM do vídeo COASTGUARD.



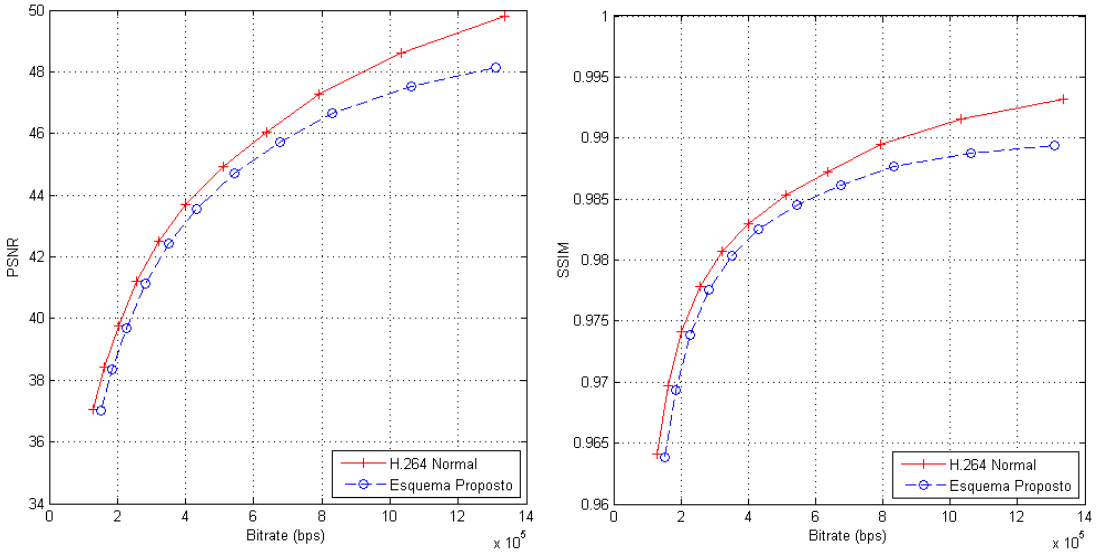
(a) PSNR (b) SSIM

Figura 4.6: Curvas PSNR e SSIM do vídeo CONTAINER.



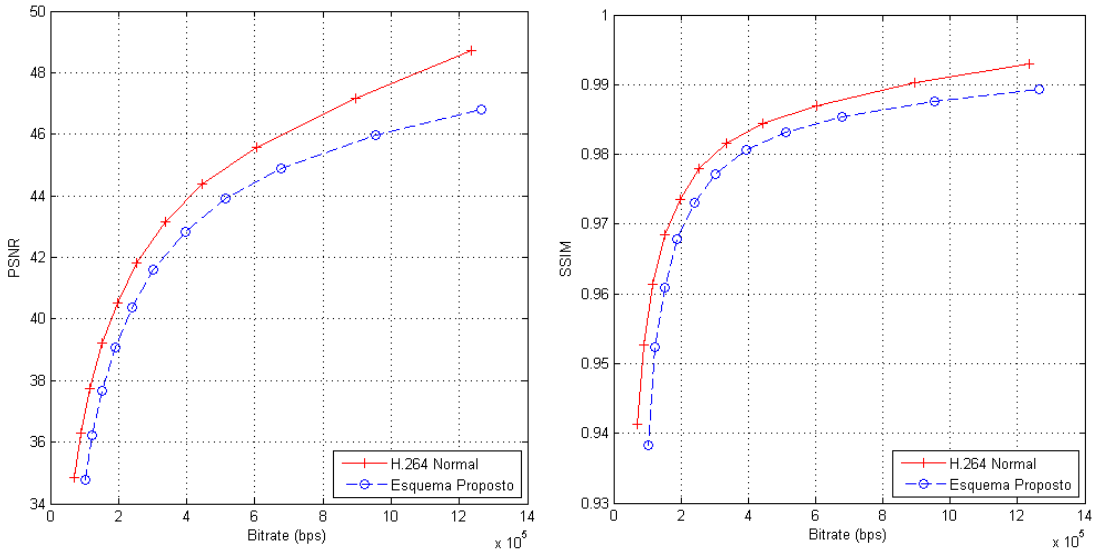
(a) PSNR (b) SSIM

Figura 4.7: Curvas PSNR e SSIM do vídeo FOOTBALL.



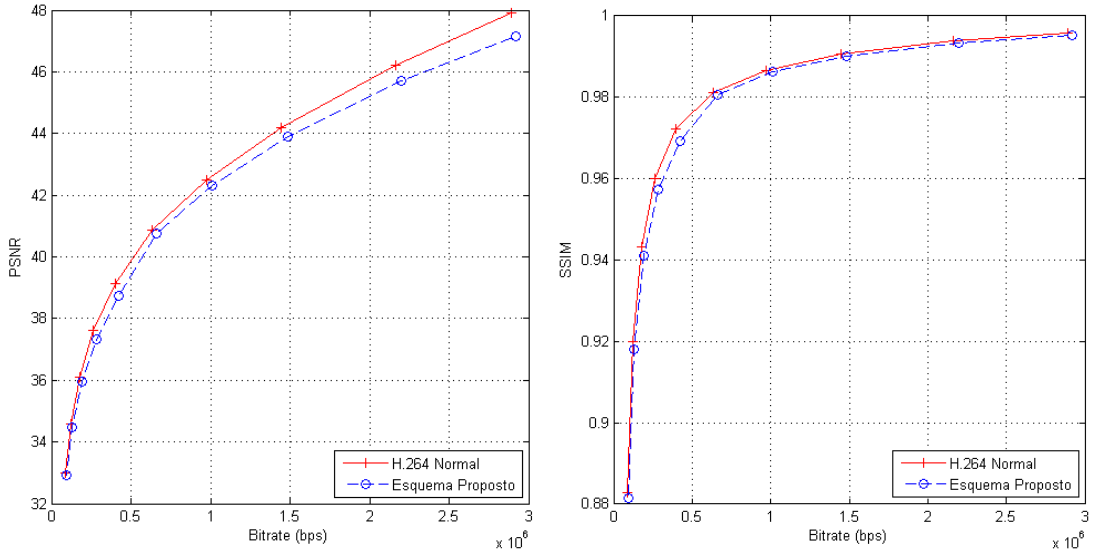
(a) PSNR (b) SSIM

Figura 4.8: Curvas PSNR e SSIM do vídeo ICE.



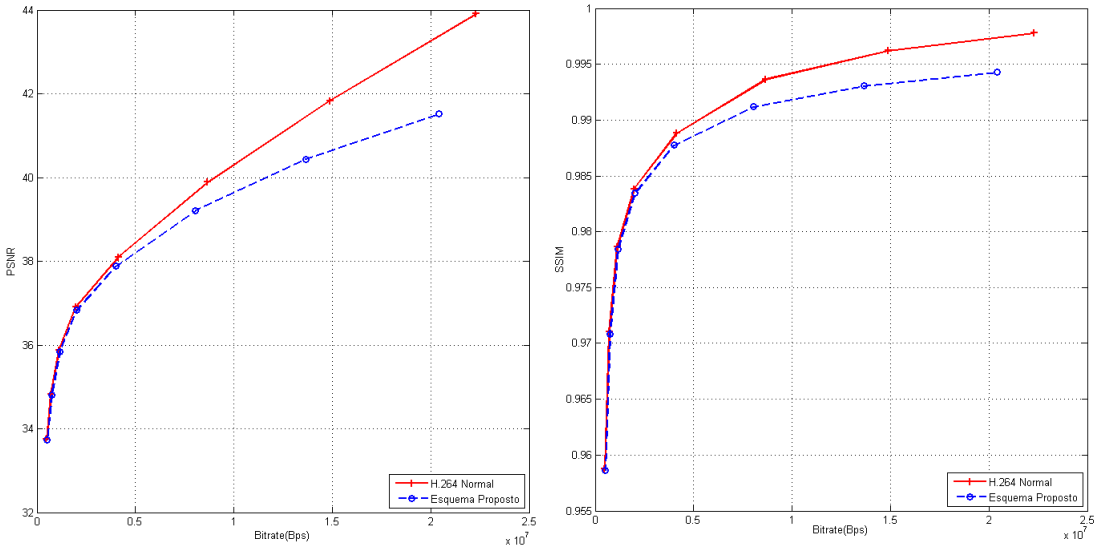
(a) PSNR (b) SSIM

Figura 4.9: Curvas PSNR e SSIM do vídeo NEWS.



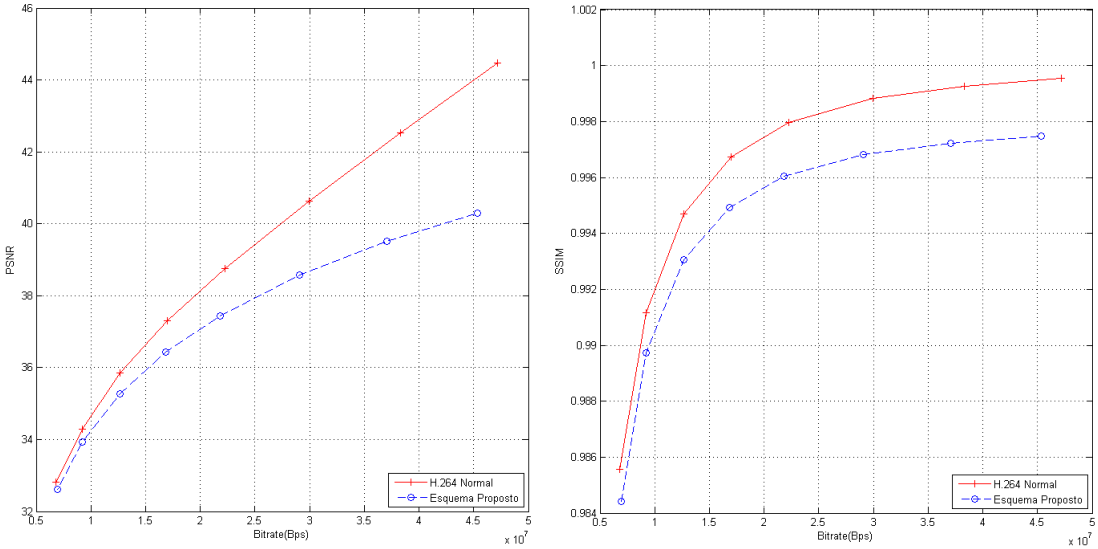
(a) PSNR (b) SSIM

Figura 4.10: Curvas PSNR e SSIM do vídeo WATERFALL.



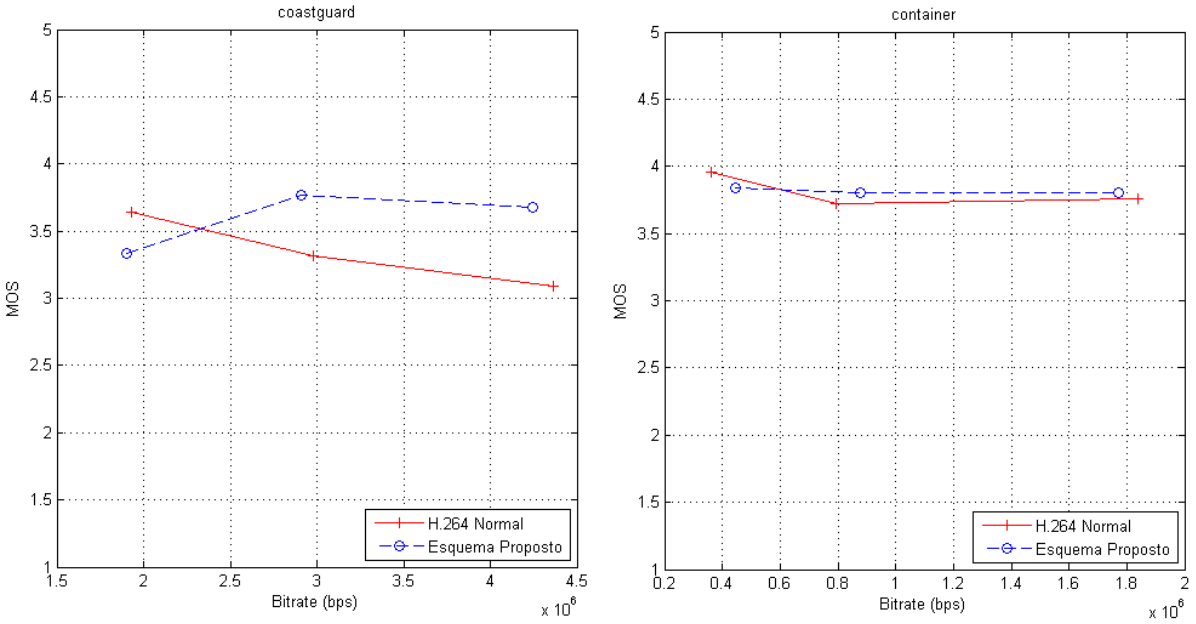
(a) PSNR (b) SSIM

Figura 4.11: Curvas PSNR e SSIM do vídeo IN TO TREE.



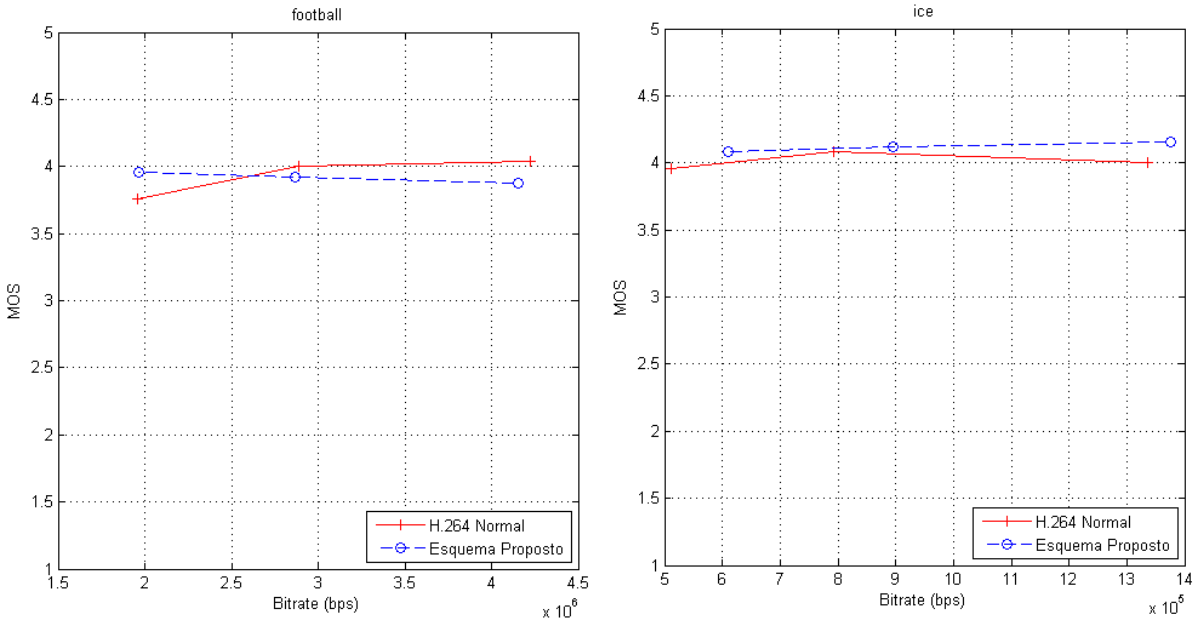
(a) PSNR (b) SSIM

Figura 4.12: Curvas PSNR e SSIM do vídeo DUCKTAKEOFF.



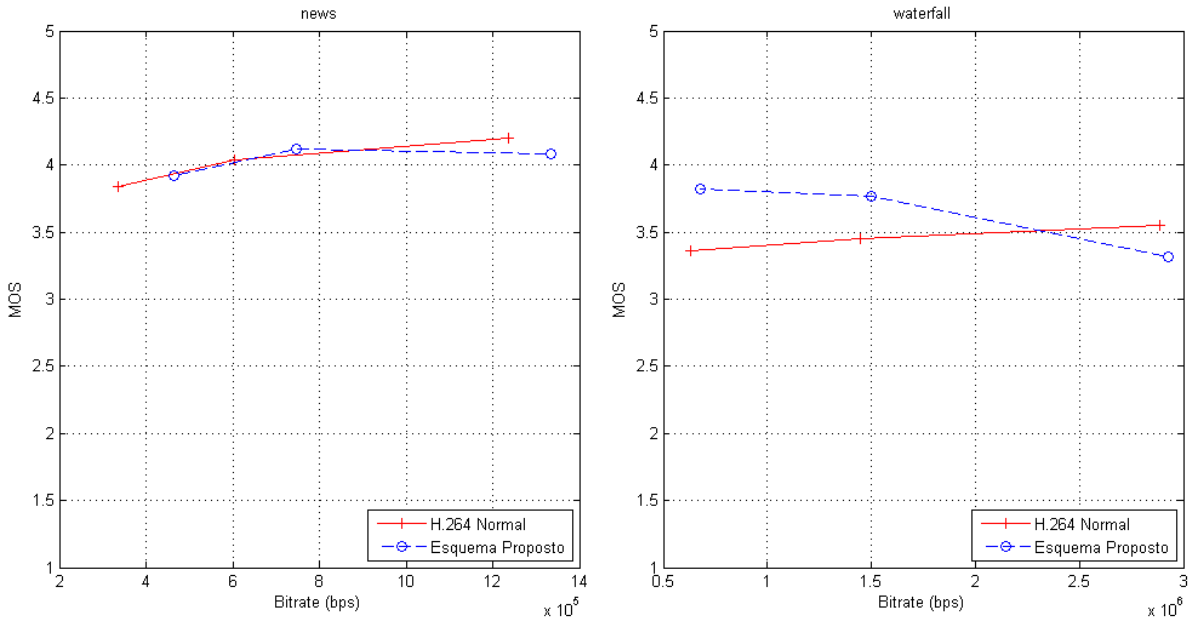
(a) Coastguard (b) Container

Figura 4.13: Curvas de taxa-distorção utilizando pontuações subjetivas (MOS) para as sequências COASTGUARD e CONTAINER.



(a) Football (b) Ice

Figura 4.14: Curvas de taxa-distorção utilizando pontuações subjetivas (MOS) para as sequências FOOTBALL e ICE.



(a) News

(b) Waterfall

Figura 4.15: Curvas de taxa-distorção utilizando pontuações subjetivas (MOS) para as sequências NEWS e WATERFALL.

Capítulo 5

Conclusões

Neste trabalho foi proposto um esquema de codificação usando modelos de textura e métodos simples de classificação. O esquema foi focado para tornar pratico e reduzir a complexidade o esquema proposto por Zhang e Bull [9] e mesmo assim manter a qualidade subjetiva do vídeo codificado. Mostrando também um potencial para reduzir a complexidade em comparação do padrão H.264. As contribuições são realizadas em cada passo do esquema apresentado, as quais são:

- No passo da segmentação ao invés de utilizar *watershed* e transformada complexa de wavelet como em [9], foi utilizado *watershed* com o filtragem de Gabor para extrair as características de textura. Para evitar a sobre-segmentação, foram fundidas as regiões com maior semelhança. As regiões resultantes da segmentação são limitadas pela quantidade de seus blocos pertencentes, já que o *warping* de regiões com maior número de blocos requer mais iterações, o que significa mais complexidade.
- A classificação das regiões entre texturizadas e não-texturizadas foi realizada usando a transformada DCT. A classificação de regiões texturizadas entre estática e dinâmica é realizada usando simples diferenças entre quadros vizinhos. A transformada e as diferenças aumentam um pouco a complexidade na codificação em comparação ao trabalho de Zhang e Bull [9], onde se utiliza para sua classificação a estimação de movimento de todos os blocos de 16×16 *pixels* de cada região e a transformada complexa wavelet.
- O processo do *warping* de textura foi realizado com os mesmos passos descritos por Zhang e Bull [9], onde usa-se a estimação de movimento para blocos de 8×8 *pixels*. Para diminuir o número de vezes que é realizado a estimação, foram utilizados blocos de 16×16 *pixels*.
- Na síntese de textura de Zhang e Bull [9], para determinar as matrizes paramétricas, realiza-se o processo de *warping* para o primeiro, terceiro e quinto quadro do GOP, com referencia ao segundo e quarto quadros. Estes passos não são necessários em nosso esquema, já que achamos as matrizes paramétricas utilizando quadros reconstruídos (os quadros chave do GOP e o terceiro quadro reconstruído por *warping*).
- Na avaliação de qualidade dos blocos reconstruídos foram usados simples comparações estatísticas com baixa complexidade, identificando os blocos que serão recon-

struídos pelo esquema proposto e os blocos que serão deixados para sua codificação pelo H.264.

Os experimentos mostraram que:

- Como esperado [9], as curvas com as métricas objetivas e a avaliação com a métrica Bjontegaard para PNSR e SSIM mostram resultados inferiores ao padrão H.264/AVC.
- A qualidade subjetiva foi mantida em comparação ao padrão H.264/AVC, os resultados com a métrica Bjontegaard mostram melhores pontuações na escala MOS em média.

O esquema proposto foi realizado fora do código-fonte do software de referência do padrão H.264/AVC, para comparar os tempos de codificação foi implementado a estimação de movimento do H.264 na mesma ferramenta do esquema proposto. Os resultados mostram um potencial do esquema proposto para reduzir o tempo de codificação em comparação do padrão H.264 mantendo a qualidade subjetiva.

Como trabalhos futuros podemos propor:

- A segmentação realizada nos quadros de tipo B poderia ser realizada somente no segundo e quarto quadros do GOP, e utilizada para definir as regiões do terceiro quadro, para reduzir ainda mais a complexidade.
- Para melhorar a classificação e aumentar o número de regiões classificadas como estáticas, pode-se realizar a estimação de movimento de um bloco central pertencente à região.
- Definir um modo de codificação para os blocos processados no código-fonte do padrão H.264/AVC, para evitar a avaliação destes blocos nos diferentes modos disponíveis.
- Aproveitar a segmentação e classificação das regiões para atribuir um adequado parâmetro de quantização às regiões que sejam codificadas pelo H.264/AVC.
- Embutir o código-fonte no padrão H.264/AVC para medir sua complexidade computacional.
- Utilizar outro codec de vídeo como o emergente *High Efficiency Video Coding* (HEVC/H.265).

Referências

[1] L. V. Agostini, *Desenvolvimento de Arquiteturas de Alto Desempenho Dedicadas à Compressão de Vídeo Segundo o Padrão H.264/AVC*. Tese de Doutorado, Universidade Federal do Rio Grande do Sul UFRGS, Porto Alegre, Brasil, Agosto 2007. viii, 1, 5, 6, 7, 8, 9, 10, 14

[2] E. M. Hung, *Compensação de movimento utilizando blocos Multi-escala e forma variável em um codec de vídeo híbrido*. Tese de Mestrado, Universidade de Brasília UNB, Brasília, Brasil, Julho 2007. viii, 12, 16

[3] M. Mirmedhdi, X. Xie, and J. Suri, *Handbook of Texture Analysis*. Imperial College Press, 2008. viii, 1, 17, 19, 20

[4] K. Sayood, *Introduction to Data Compression*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2nd ed., 2000. 1, 4, 5, 11

[5] I. Richardson, *H.264 and MPEG-4 Video Compression: Video Coding for Next-generation Multimedia*. John Wiley and Sons, Chichester, UK, 2003. viii, 1, 7, 8, 9, 11, 13, 14

[6] B. Haskell, A. Puri, and A. Netravali, *Digital Video: An Introduction to MPEG-2*. Digital multimedia standards series, Springer, Heidelberg, 1997. 1

[7] ITU-T, "Video coding for low bit rate communication," *ITU-T Recommendation H.263*, Novembro 2000. 1

[8] A. Puri, "Video coding using the H.264/MPEG-4 AVC compression standard," in *Signal Processing: Image Communication*, vol. 19, pp. 793–849, Octubro 2004. 1, 2, 7, 8, 9

[9] F. Zhang and D. Bull, "A parametric framework for video compression using region-based texture models," *IEEE Journal of Selected Topics in Signal Processing*, vol. 5, pp. 1378–1392, Novembro 2011. 1, 2, 17, 21, 26, 29, 31, 32, 33, 34, 36, 40, 41, 51, 52

[10] P. Ndjiki-Nya, T. Hinz, and T. Wiegand, "Generic and robust video coding with texture analysis and synthesis," in *Proc. IEEE International Conference on Multimedia and Expo*, (Beijing), pp. 1447–1450, Julho 2007. 1, 20, 21

[11] M. Bosch, F. Zhu, and E. Delp, "Spatial texture models for video compression," in *Proc. IEEE International Conference on Image Processing*, vol. 1, (San Antonio, TX), pp. 93–96, Octubro 2007. 1, 21

[12] J. Byrne, S. Ierodiaconou, D. Bull, D. Redmill, and P. Hill, "Unsupervised image compression-by-synthesis within a JPEG framework," in *Proc. IEEE International Conference on Image Processing*, (San Diego, CA), pp. 2892–2895, Octubro 2008. 1

[13] F. Zhang and D. Bull, "Enhanced video compression with region-based texture models," in *Picture Coding Symposium*, (Nagoya), pp. 54–57, Dezembro 2010. 1

[14] Y.-W. Huang, B.-Y. Hsieh, S.-Y. Chien, S.-Y. Ma, and L.-G. Chen, "Analysis and complexity reduction of multiple reference frames motion estimation in H.264/AVC," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 16, pp. 507–522, Maio 2006. 2

[15] G. J. Sullivan and J.-R. Ohm, "Recent developments in standardization of high efficiency video coding (HEVC)," *SPIE Proceedings*, vol. 7798, Agosto 2010. 2

[16] Z. Wang, A. Bovik, H. Sheikh, and E. Simoncelli, "Image quality assessment: from error visibility to structural similarity," *IEEE Transactions on Image Processing*, vol. 13, pp. 600–612, Abril 2004. 2, 16

[17] O. M. Filho and H. V. Neto, *Processamento Digital de Imagens.* Série Acadêmica, BRASPORT, Rio de Janeiro, 1999. 4

[18] C. E. Shannon, "A mathematical theory of communication," in *Bell System Technical Journal*, vol. 27, Octubro 1948.

[19] Y. Shi and H. Sun, *Image and Video Compression for Multimedia Engineering: Fundamentals, Algorithms, and Standards.* CRC Press Inc., Boca Raton, FL, USA, 1999. 4, 6

[20] I. Richardson, *Video Codec Design: Developing Image and Video Compression Systems.* John Wiley and Sons, Chichester, UK, 2002. 5, 7, 13

[21] M. Ghanbari, *Standard Codecs: Image Compression to Advanced Video Coding.* Institution of Engineering and Technology, London, 2003. 5, 6

[22] A. M. C. Da Silva, *Desenvolvimento de um Compensador de Movimento para o Padrão H.264 de Compressão de Vídeo.* Tese de Graduação, Universidade Federal de Pelotas UFPel, Pelotas/RS, Brasil, Agosto 2006. 11

[23] W. Pratt, *Digital Image Processing.* John Wiley and Sons, Chichester, UK, 2007. 11

[24] H. R. Wu and K. R. Rao, *Digital Video Image Quality and Perceptual Coding (Signal Processing and Communications).* CRC Press Inc., Boca Raton, FL, USA, 2005. 11

[25] T. Wiegand, G. J. Sullivan, G. Bjøntegaard, and A. Luthra, "Overview of the h.264/avc video coding standard," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 13, pp. 560–576, Julho 2003. 14

[26] P. List, A. Joch, J. Lainema, G. Bjontegaard, and M. Karczewicz, "Adaptive deblocking filter," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 13, pp. 614–619, Julho 2003. 14

[27] ITU-T, "Subjective video quality assessment methods for multimedia applications," *ITU-T Recommendation P.910*, Novembro 1999. 15

[28] R. C. C. De Souza, *Avaliação de Imagens Através de Similaridade Estrutural e do Conceito de Mínima Diferença de Cor Perceptível.* Tese de Mestrado, Universidade do Estado do Rio de Janeiro UERJ, Rio de Janeiro, Brasil, Octubro 2009. 15, 16

[29] M. Vranjes, S. Rimac-Drlje, and D. Zagar, "Objective video quality metrics," in *Proc. 49th International Symposium ELMAR*, (Zadar, Croatia), pp. 45–49, Setembro 2007. 16

[30] A. Jain, *Fundamentals of digital image processing.* Prentice Hall, New Jersey, EUA., 1989. 18

[31] B. Jähne, *Digital Image Processing.* Springer, Heidelberg, 1995. 18

[32] R. Gonzalez and R. Woods, *Processamento de Imagens Digitais.* Edgard Blucher, São Paulo, 2000. 18, 19

[33] I. E. Sobel, *Camera Models and Machine Perception.* Ph. D. Dissertation, Stanford University, Stanford, CA, EUA, Agosto 1970. 18

[34] J. Canny, "A computational approach to edge detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 8, pp. 679–698, Janeiro 1986. 18

[35] E. Dougherty and J. Astola, *An Introduction to Nonlinear Image Processing.* SPIE Optical Engineering Press, Bellingham, Washington, EUA, 1994. 19

[36] F. Meyer, "Topographic distance and watershed lines," *Signal Process*, vol. 38, pp. 113–125, Julho 1994. 19, 24

[37] L. Vincent and P. Soille, "Watersheds in digital spaces: an efficient algorithm based on immersion simulations," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 13, pp. 583–598, Agosto 1991. 19

[38] L. Shafarenko, M. Petrou, and J. Kittler, "Automatic watershed segmentation of randomly textured color images," *IEEE Transactions on Image Processing*, vol. 6, pp. 1530–1544, Agosto 1997. 19

[39] C. Lantuejoul, *La Squelettisatoin et son application aux mesures topologiques des mosaiques polycristalines.* School of Minews, Paris, France. 19

[40] R. Burduk, M. Kurzynski, M. Wozniak, and A. Zolnierek, *Computer Recognition Systems 4*, vol. 4. Springer, Heidelberg, 2011. 19

[41] R. J. O'Callaghan and D. R. Bull, "Combined morphological-spectral unsupervised image segmentation," *IEEE Transactions on Image Processing*, vol. 14, pp. 49–62, Janeiro 2005. 20

[42] N. Kingsbury, "Complex wavelets for shift invariant analysis and filtering of signals," *Journal of Applied and Computational Harmonic Analysis*, vol. 10, pp. 234–253, Maio 2001. 20

[43] V. Kwatra, A. Schödl, I. Essa, G. Turk, and A. Bobick, "Graphcut textures: image and video synthesis using graph cuts," in *Proc. ACM Transactions on Graphics*, vol. 22, (New York, USA), pp. 277–286, Julho 2003. 20

[44] G. Doretto, A. Chiuso, Y. N. Wu, and S. Soatto, "Dynamic textures," *International Journal of Computer Vision*, vol. 51, pp. 91–109, Fevereiro 2003. 20, 31

[45] P. Ndjiki-Nya, T. Hinz, C. Stuber, and T. Wiegand, "A content-based video coding approach for rigid and non-rigid textures," in *Proc. IEEE International Conference on Image Processing*, (Atlanta, GA), pp. 3169–3172, Octubro 2006. 20

[46] P. Ndjiki-Nya, B. Makai, G. Blattermann, A. Smolic, H. Schwarz, and T. Wiegand, "Improved h.264/avc coding using texture analysis and synthesis," in *IEEE International Conference on Image Processing ICIP*, vol. 3, pp. 49–52, Setembro 2003. 20

[47] M. Bosch, F. Zhu, and E. Delp, "Video coding using motion classification," in *Proc. IEEE International Conference on Image Processing ICIP*, (San Diego, CA), pp. 1588–1591, Octubro 2008. 21

[48] H. Wang, Y. Wexler, E. Ofek, and H. Hoppe, "Factoring repeated content within and among images," *Journal ACM Transactions on Graphics*, vol. 27, pp. 1–10, Agosto 2008. 21

[49] D. Liu, X. Sun, F. Wu, S. Li, and Y.-Q. Zhang, "Image compression with edge-based inpainting," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 17, pp. 1273–1287, Octubro 2007. 21

[50] A. Stojanovic, M. Wien, and J.-R. Ohm, "Extended dynamic texture prediction for h.264/avc inter coding," in *Proc. IEEE International Conference on Image Processing*, (San Diego, CA), pp. 1608–1611, Octubro 2008. 21

[51] R. Chellappa, S. Chatterjee, and R. Bagdazian, "Texture synthesis and compression using gaussian-markov random field models," in *IEEE Transactions on Man and Cybernetics Systems*, vol. SMC-15, pp. 298–303, 1985. 21

[52] J. G. Daugman, "Two-dimensional spectral analysis of cortical receptive field profiles," *Vision Research*, vol. 20, no. 10, pp. 847–856, 1980. 23

[53] M. Tuceryan and A. K. Jain, "Handbook of pattern recognition & computer vision," ch. Texture analysis, pp. 235–276, World Scientific Pub Co Inc, River Edge, NJ, USA, 1993. 23

[54] R. E. Abreu M. and S. A. Bullones S., "Implementación de un filtro de gabor modificado para el mejoramiento de la imagen de huellas dactilares en un sistema verificador biométrico," in *V CIBELEC 2012*, (Merida, Venezuela), Maio 2012. 24

[55] C. T. Kiang, *Novel block-based motion estimation and segmentation for video coding*. Ph.D. Dissertation, University of Bristol, U.K., Agosto 2007. 29

[56] G. Doretto and S. Soatto, *Editable Dynamic Textures*. Technical Report TR020001, UCLA Computer Science Department, Los Angeles, CA, 2002. 32

[57] ISO/IEC JTC1/SC29/WG11 e ITU-T SG16 Q.6, "Jm 18.4." Março, 2013. 36

[58] ITU-R, "Methodology for the subjective assessment of the quality of television pictures," *ITU-R Recommendation BT.500*, Janeiro 2012. 36

[59] Y. Ismail, J. McNeely, M. Shaaban, H. Mahmoud, and M. Bayoumi, "Fast motion estimation system using dynamic models for H.264/AVC video coding," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 22, pp. 28–42, Janeiro 2012. 38

[60] M.-Y. Chiu, *Fast motion estimation techniques and algorithms for H.264/AVC video compression*. Faculty of Engineering, The Hong Kong Polytechnic University, Hong Kong, 2011. 38

[61] H. Ates and Y. Altunbasak, "SAD reuse in hierarchical motion estimation for the H.264 encoder," in *Proc. IEEE International Conference on Acoustics, Speech, and Signal Processing*, vol. 2, pp. 905–908, março 2005. 38

[62] G. Bjontegaard, "Calculation of average psnr differences between rd-curves," in *presented at the 13th VCEG-M33 Meeting*, (Austin, TX), Abril 2001. 40, 41