Gabriel Tedgue Beltrão

Rápida Predição da Direção do Bloco para Aplicação com Transformadas Direcionais

Campinas
2012

Universidade Estadual de Campinas

Faculdade de Engenharia Elétrica e de Computação

Gabriel Tedgue Beltrão

Rápida Predição da Direção do Bloco para Aplicação com Transformadas Direcionais

Dissertação de mestrado apresentada à Faculdade de Engenharia Elétrica e de Computação da Universidade Estadual de Campinas, como parte dos requisitos exigidos para a obtenção do título de Mestre em Engenharia Elétrica.
Área de concentração: Telecomunicações e Telemática.

Orientador: Prof. Dr. Yuzo Iano
Co-orientador: Prof. Dr. Rangel Arthur

Este exemplar corresponde à versão final da dissertação defendida pelo aluno Gabriel Tedgue Beltrão, e orientada pelo Prof. Dr. Yuzo Iano

Campinas
2012

FICHA CATALOGRÁFICA ELABORADA PELA
BIBLIOTECA DA ÁREA DE ENGENHARIA E ARQUITETURA - BAE -
UNICAMP

B419r

Beltrão, Gabriel Tedgue
Rápida predição da direção do bloco para aplicação com transformadas direcionais / Gabriel Tedgue Beltrão. --Campinas, SP: [s.n.], 2012.

Orientador: Yuzo Iano.
Coorientador: Rangel Arthur.
Dissertação de Mestrado - Universidade Estadual de Campinas, Faculdade de Engenharia Elétrica e de Computação.

1. Compressão de imagens. 2. Processamento de sinais. 3. MPEG (Padrão de codificação de vídeo). 4. Televisão digital. 5. Teoria da codificação. I. Iano, Yuzo, 1950-. II. Arthur, Rangel, 1977-. III. Universidade Estadual de Campinas. Faculdade de Engenharia Elétrica e de Computação. IV. Título.

Título em Inglês: Fast block direction prediction for directional transforms
Palavras-chave em Inglês: Image compression, Signal processing, MPEG (Video coding standard), Digital television, Coding theory
Área de concentração: Telecomunicações e Telemática
Titulação: Mestre em Engenharia Elétrica
Banca examinadora: Yuzo Iano, Luiz César Martini, David Fernandes
Data da defesa: 05-12-2012
Programa de Pós Graduação: Engenharia Elétrica

COMISSÃO JULGADORA - TESE DE MESTRADO

Candidato: Gabriel Tedgue Beltrão

Data da Defesa: 5 de dezembro de 2012

Título da Tese: "Rápida Predição da Direção do Bloco para Aplicação com Transformadas Direcionais"

Prof. Dr. Yuzo Iano (Presidente): _____

Prof. Dr. David Fernandes: _____

Prof. Dr. Luiz César Martini: _____

A meu avô, por todos os exemplos que podem ser dados.

Agradecimentos

Agradeço,

à minha família pelo suporte em todos os momentos.

aos Professores Yuzo Iano e Rangel Arthur, pela oportunidade, pela anos de orientação, por todo o conhecimento técnico, e principalmente pela inestimável troca de experiências.

aos colegas do DECOM por todas as discussões em nossas reuniões semanais, que resultaram em inúmeras contribuições para este trabalho.

ao "grupo jovem" pela crescente amizade e produção científica.

aos demais colegas de trabalho pela enriquecedora convivência diária.

a CNPq, Faepex/Unicamp, RNP/CTIC, PRP/Unicamp e FAPESP, pelo apoio financeiro durante todo este trabalho.

em especial aos meus amigos, aos quais incontáveis vezes eu disse "hoje não vai dar... semana que vem eu vou", e que nunca desistiram!

Agradecimentos

Agradeço,

em especial ao programa CAPES RH-TVD, da Coordenação de Aperfeiçoamento de Pessoal de Nível Superior, tanto pelo apoio financeiro quanto pelo incentivo acadêmico para que este trabalho pudesse ser realizado.

De vez em quando você tem que fazer uma pausa e visitar a si mesmo.

Audrey Giorgi

Resumo

As transformadas derivadas da DCT são amplamente utilizadas para compressão de vídeo. Recentemente, muitos autores têm destacado que os resíduos de predição normalmente apresentam estruturas direcionais que não podem ser eficientemente representadas pela DCT convencional. Nesse contexto, muitas transformadas direcionais têm sido propostas como forma de suplantar a deficiência da DCT em lidar com tais estruturas. Apesar do desempenho superior das transformadas direcionais sobre a DCT convencional, para a sua aplicação na compressão de vídeo é necessário avaliar o aumento no tempo de codificação e a complexidade para sua implementação. Este trabalho propõe um rápido algoritmo para se estimar as direções existentes em um bloco antes da aplicação das transformadas direcionais. O codificador identifica as direções predominantes em cada bloco e aplica apenas a transformada referente àquela direção. O algoritmo pode ser usado em conjunto com qualquer proposta de transformadas direcionais que utilize a técnica de otimização por taxa-distorção (RDO) para a seleção da direção a ser explorada, reduzindo a complexidade de implementação a níveis similares a quando apenas a DCT convencional é utilizada.

Palavras-chave: HEVC, transformadas direcionais, compressão de vídeo

Abstract

DCT-based transforms are widely adopted for video compression. Recently, many authors have highlighted that prediction residuals usually have directional structures that cannot be efficiently represented by conventional DCT. In this context, many directional transforms have been proposed as a way to overcome DCT's deficiency in dealing with such structures. Although directional transforms have superior performance over the conventional DCT, for application in video compression it is necessary to evaluate increase in coding time and complexity for its implementation. This work proposes a fast algorithm for estimating blocks directions before applying directional transforms. The encoder identifies predominant directions in each block, and only applies the transform referent to that direction. The algorithm can be used in conjunction with any proposed algorithm for directional transforms that uses the rate-distortion optimization (RDO) process for selection of the direction to be explored, reducing implementation complexity to similar levels when only conventional DCT is used.

Key-words: HEVC, directional transforms, video compression

Lista de Figuras

Lista de Tabelas

Lista de Acrônimos e Notação

CODEC	Encoder/Decoder
RGB	Red/Green/Blue
DCT	Discrete Cosine Transform
DCT-2D	Discrete Cosine Transform - Two dimensional
RDO	Rate-Distortion Optimized
HEVC	High Efficiency Video Coding
H.264/AVC	H.264/Advanced Video Coding
QCIF	Quarter Common Intermediate Format
CIF	Common Intermediate Format
SD	Standard-Definition
HD	High-Definition
HDTV	High-Definition Television
MPEG	Moving Picture Experts Group
GOP	Group of Pictures
IDR	Instataneous Decoder Refresh
DWT	Discrete Wavelet Transform
KLT	Karhunen-Loève Transform
DFT	Discrete Fourier Transform
CABAC	Context-Based Adaptive Binary Arithmetic Encoding
QoE	Quality of Experience
ITU	International Telecommunications Union
PSNR	Peak Signal to Noise Ratio
SSIM	Structural Similarity index
VQM	Video Quality Metrics
BD-PSNR	Bjontegaard-Delta PSNR
ITU-T	ITU - Telecommunication Standardization Sector
VCEG	Video Coding Experts Group
ISO	International Organization for Standardization
IEC	International Electrotechnical Commission
JCT-VC	Joint Collaborative Team on Video Coding
CU	Coding Unit
PU	Prediction Unit

Lista de Acrônimos e Notação (2)

TU	Transform Unit
LCU	Largest Coding Unit
AMVP	Advanced Motion Vector Prediction
RQT	Residual Quadtree
DST	Discrete Sine Transform
ROT	Rotational Transform
CAVLC	Context Adaptive Variable Length Coding
RA	Random Access
LD	Low Delay
HE	High Efficiency

Sumário

Capítulo 1

Introdução Geral

A demanda por serviços de multimídia (televisão, celulares, vídeo 3D, Blu-ray, etc.) tem crescido rapidamente, assim como a expectativa pela qualidade desses serviços. De uma forma geral, para se obter a máxima qualidade possível, evitando os ruídos e distorções inerentes aos sistemas analógicos, os sinais originais (fala, aúdio, imagens ou vídeo) são digitalizados para fins de armazenamento e transmissão, e reconstruídos no receptor final antes da exibição.

O primeiro problema que surge é que essa informação digitalizada, principalmente o vídeo, normalmente é muito volumosa e, mesmo com evolução tecnológica dos meios de transmissão e armazenamento, lidar com toda a demanda gerada pelos diversos tipos de serviço ainda não é uma questão trivial.

Dessa forma, sistemas que possam reduzir/comprimir toda essa informação, tornando a entrega desses serviços fisicamente possível e atrativa de um ponto de financeiro, têm um papel crucial nesse novo contexto tecnológico. Comprimir significa representar os dados com uma menor quantidade de bits. Normalmente envolve um par de sistemas: o compressor (codificador), que converte a fonte de dados para um formato reduzido, antes da transmissão ou do armazenamento, e o descompressor (decodificador), que reconverte o vídeo ao seu formato original, antes da exibição. Daí vem o nome, normalmente utilizado para representar o sistema como um todo: CODEC (CODificador/DECodificador). A problemática da compressão de vídeo pode ser vista de duas formas: como representar o vídeo com a maior fidelidade possível, dentro de uma determinada taxa de dados, ou como representar o vídeo na menor taxa possível, dentro de limites aceitáveis de qualidade. Em qualquer um dos casos, existe um comprimisso entre taxa de dados e qualidade.

A Figura 1.1 mostra algumas aplicações possíveis de um codec de vídeo. Por exemplo, um sinal de vídeo em alta definição, digitalizado com 1920×1080 pixels de resolução, 8 bits por pixel, no padrão RGB e com taxa de 30 frames/s, gera nada menos que 1,3905 Gbps de dados. Para que seja possivel a sua transmissão por um canal comum de TV (6MHz/18Mbps), é necessária uma compressão de aproximadamente 80 vezes.

O papel do estágio de transformada em um codec de vídeo é representar a imagem (normalmente os resíduos de predição) em outro domínio onde as redundâncias espaciais ainda presentes possam ser exploradas com maior eficiência, sem degradação perceptível a visão humana. A maioria dos padrões existentes utiliza a DCT (*Discrete Cosine Transform*), ou suas

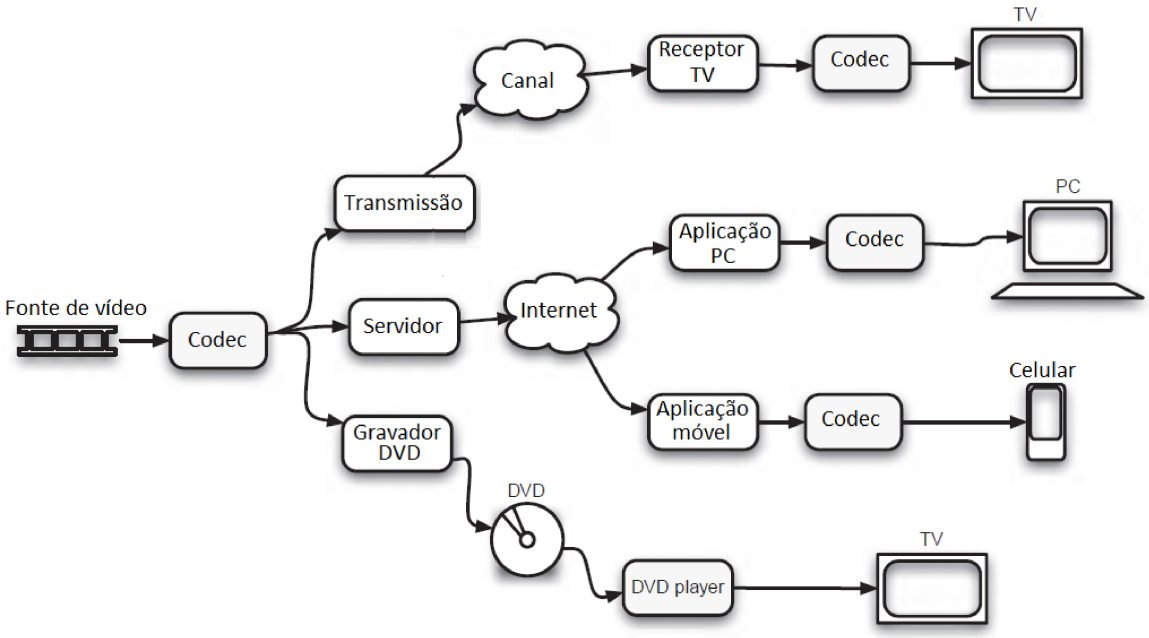


Figura 1.1: Aplicações de um codec de vídeo (adaptado de [1]).

variantes, como principal ferramenta dessa etapa. Recentemente, muitos autores têm sugerido a utilização de transformadas direcionais, como forma de suprir a deficiência da DCT em lidar com imagens que contenham estruturas direcionais, as quais, são muito comuns nos resíduos de predição temporal e espacial.

Apesar das transformadas direcionais apresentarem desempenho muito superior à DCT-2D convencional [2], para a sua aplicação na compressão de vídeo é necessário avaliar o aumento de complexidade para sua implementação. A maioria das transformadas direcionais propostas recentemente, baseia-se na técnica de otimização por taxa-distorção [3] (RDOs) para a escolha da direção que será utilizada. Essa é um técnica de força bruta, na qual, para cada opção de transformada disponível, o bloco passa pelos processos de transformada, quantização, codificação de entropia, quantização inversa e transformada inversa. Em seguida, o custo taxa-distorção é calculado e a opção de menor custo é escolhida. Esse processo aumenta consideravelmente a complexidade do codificador e pode se tornar impeditivo para aplicações em tempo real.

No intuito de reduzir essa complexidade para a aplicação das transformadas direcionais, buscou-se neste trabalho uma forma de se eliminar o cálculo das RDOs, a partir de uma rápida estimação da direção predominante em cada bloco a ser codificado. Dessa forma, antes de aplicar a transformada direcional, o codificador identifica a direção predominante e aplica apenas a transformada referente àquela direção, ou, caso não haja uma direção predominante, o codificador pode optar por aplicar a DCT-2D convencional. O método pode ser aplicado em conjunto com qualquer proposta de transformadas direcionais que utilize a técnica RDO como forma de escolha da direção a ser explorada, reduzindo consideravelmente a complexidade de implementação para níveis similares aos da DCT-2D convencional.

Os capítulos seguintes estão organizadas da seguinte forma: O Capítulo 2 faz uma revisão dos conceitos gerais de codificação de vídeo, buscando alinhar o entendimento básico das ferramentas

que serão utilizadas nas seções seguintes. O Capítulo 3 aborda o HEVC (*High Efficiency Video Coding*), que está sendo desenvolvido como futuro sucessor do H.264/AVC, e que representa o estado da arte nas técnicas de compressão de vídeo. O Capítulo 4 apresenta a contribuição deste trabalho e sua integração a uma das propostas de transformadas direcionais da literatura, seguido das conclusões gerais no Capítulo 5.

Capítulo 2

Conceitos de Codificação de Vídeo

2.1 Vídeo Digital, Amostragem e Formatos

A representação de uma cena real na forma de um vídeo digital requer a amostragem dessa cena tanto espacial quanto temporalmente. A amostragem espacial nada mais é do que a representação de uma imagem estática, por um conjunto limitado de pequenos elementos de imagem, chamados pixels (*picture elements*), dispostos em linhas e colunas, no formato de uma matriz (frame). O número total de pixels (resolução espacial) e a quantidade de bits utilizados para representar cada pixel têm impacto direto na qualidade e riqueza de detalhes dessa imagem.

A sucessão (amostragem temporal) de vários desses frames, a uma determinada taxa, dá a impressão de movimento contínuo e suave. Quanto maior essa taxa, mais realista se torna o movimento na cena. Em televisão, por exemplo, normalmente utilizam-se taxas de 25, 30 ou 50 frames por segundo, que podem ser progressivos (o vídeo é amostrado como uma sucessão de frames completos) ou entrelaçados (os frames são divididos em linhas pares e ímpares, e cada conjunto de linhas - campo - é exibido a cada intervalo de amostragem temporal). O entrelaçamento melhora a aparência de movimento nas cenas, sem aumentar a taxa de dados necessária para sua representação. A Figura 2.1 ilustra as amostragens espacial e temporal em um vídeo.

As cores, por sua vez, também precisam de uma representação no formato digital. As imagens monocromáticas (preto e branco) são representadas por apenas um valor que indica a luminância (brilho) de cada pixel. Normalmente utilizam-se três valores para representar cada pixel de uma imagem colorida. Os tipos de representação das cores são chamados de Espaço de Cores e o mais conhecido deles é o formato RGB (Vermelho-Verde-Azul).

No formato RGB, cada pixel é representado por três valores que indicam a proporção dessas três cores nesse pixel. Ao variar-se essas proporções pode-se criar uma infinidade de cores. Como nesse formato as três cores são igualmente importantes, elas devem ser armazenadas com a mesma resolução (mesmo número de bits). Porém, o sistema visual humano é menos sensível à cor (crominância) do que ao brilho (luminância), e então, pode-se pensar numa forma mais eficiente de representação das cores, que utilize uma maior resolução para a luminância e uma menor para a crominância. Dessa forma, pode-se diminuir a quantidade de dados, sem degradação perceptível à visão humana.

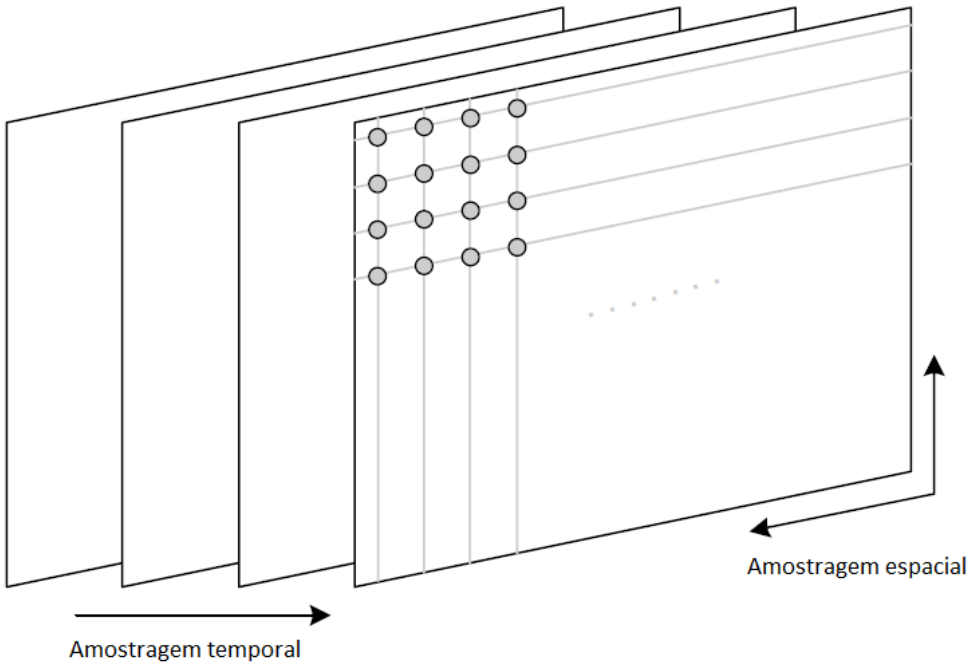


Figura 2.1: Amostragem espacial e temporal de vídeo (adaptado de [1]).

O formato YCbCr se utiliza dessa propriedade. Nele, a componente de luminância pode ser calculada como uma média ponderada das componentes R, G e B:

$$Y = k_r R + k_g G + k_b B \tag{2.1}$$

A recomendação ITU-R BT.601 [4] define $k_r = 0.299$, $k_b = 0.114$ e $k_g = 0.587$. A informação de cor, por sua vez, é representada pelas componentes de crominância, que são calculadas pela diferença entre as componentes RGB e a componente de luminância:

$$C_r = R - Y, C_b = B - Y, C_g = G - Y \tag{2.2}$$

O espaço de cores pode, a princípio, ser representado por essas quatro componentes: Y, Cr, Cb e Cg. Como o somatório ponderado de Cr, Cb e Cg é constante, apenas duas das três componentes precisam ser armazenadas. A terceira pode, sempre que necessário, ser calculada a partir das outras duas, reduzindo assim em um quarto a quantidade de dados necessários para representar a imagem. Comumente Cr e Cb são mais utilizadas e Cg é sempre recalculada, justificando o nome do formato.

Uma outra importante vantagem reside do fato de que as componentes de brilho e cor agora estão separadas. Dessa forma pode-se amostrá-las com taxas diferentes, de acordo com a sensibilidade do olho humano. Isto reduz, mais uma vez, a quantidade de dados resultante, com pouco ou nenhum impacto perceptível na qualidade visual.

Três formatos de amostragem das componentes Y, Cb e Cr são amplamente utilizados: 4:4:4, 4:2:2 e 4:2:0. O formato 4:4:4 significa que todas as componentes tem a mesma amostragem, ou seja, nenhuma redução foi feita e existe uma amostra de cada componente para cada pixel da imagem, o que preserva a total fidelidade de cores. No formato 4:2:2, também conhecido como YUY2, as componentes de crominância estão subamostradas por um fator de dois, na direção horizontal, ou seja, para cada quatro amostras de luminância na direção horizontal, existem duas de Cb e duas de Cr. No formato 4:2:0, também conhecido como YV12 (12 bits por pixel) e o mais popular dos três, Cr e Cb tem metade da amostragem, tanto na direção horizontal, quanto na vertical. Como cada componente de crominância tem um quarto do número de amostras de luminância, o formato 4:2:0 utiliza exatamente metade das amostras do 4:4:4, ou do RGB.

A Figura 2.2 mostra os três padrões e a relação entre as amostragens. Ao analisar a região tracejada, percebe-se que, usando a amostragem 4:4:4, ela é representada por 12 amostras, 4 de cada componente, o que resulta num total de 96 bits (8 bits por amostra) e uma média de 24 bits por pixel. Para a amostragem 4:2:0, tem-se 6 amostras na região (4 de Y, 1 de Cb e 1 de Cr), e um total de 48 bits, ou 12 bits por pixel (YUV12).

Existem diversos tamanhos de frame, com resoluções que se estendem desde 128×96 até $4K \times 4K$ pixels. A escolha da resolução depende em geral da aplicação desejada e da quantidade de recursos disponíveis para armazenamento e transmissão. As aplicações vão desde dispositivos móveis até televisão de alta definição (HDTV). Alguns dos formatos mais usados e sua resolução podem ser vistos na Tabela 2.1.

Nesse momento, já se pode ter uma estimativa da quantidade de dados envolvidos no armazenamento ou transmissão de vídeo digital. Por exemplo, no formato 576p, amostrado com 8 bits por amostra, tem-se $(720 \times 576 \times 3 \times 8) = 9.953.280$ bits por frame, que se exibidos a uma taxa de 30 frames/s implicam em 298.598.400 bits a cada um segundo de vídeo. Para vídeo em alta definição, 1080p, por exemplo, tem-se $(1920 \times 1080 \times 3 \times 8) = 49.766.400$ bits por frame um total de 1.492.992.000 bits/s.

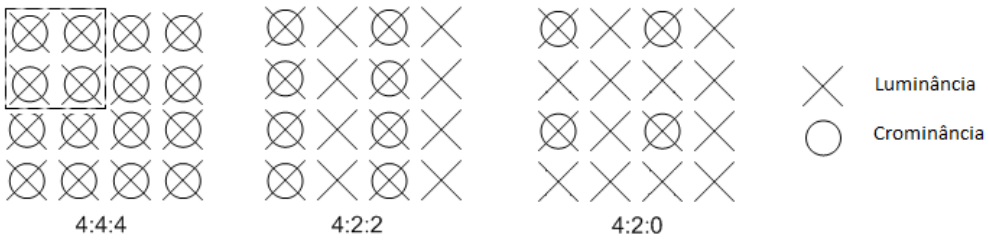


Figura 2.2: Formatos de amostragem das componentes de luminância e crominância.

Fica evidente a necessidade de ferramentas que possam reduzir essa quantidade de dados a valores factíveis de armazenamento e transmissão, compatíveis com as tecnologias existentes hoje no mercado. Por exemplo, para aplicações de TV aberta, a banda disponível de transmissão é de aproximadamente 18Mbps. Um ponto de acesso à internet residencial tem em média de 1 a 10Mbps. Os algoritmos de compressão de vídeo aparecem nesse contexto como grandes aliados

para proporcionar a viabilidade dessas aplicações.

Tabela 2.1: Formatos de vídeo e resoluções

Formatos	**Resolução**
Sub-QCIF	128×96
CIF	352×288
4CIF	704×576
576p (SD - progressivo)	720×576
720p (HD - progressivo)	1280×720
1080p (Full HD - progressivo)	1920×1080
1080i (Full HD - entrelaçado)	1920×1080

2.2 Codecs de Vídeo

Como foi visto anteriormente, a representação do vídeo no formato digital requer uma enorme quantidade de dados, como, por exemplo, um vídeo em alta definição, 1080p, que necessita de aproximadamente 1,4 Gb/s. Dessa forma, as ferramentas de codificação (compressão) de vídeo são peças fundamentais para possibilitar o seu armazenamento e transmissão.

Comprimir, nesse contexto, significa representar os dados com uma menor quantidade de bits. Normalmente envolve um par de sistemas: o compressor (encoder), que converte a fonte de dados para um formato reduzido, antes da transmissão ou do armazenamento, e o descompressor (decoder), que reconverte o vídeo ao seu formato original, antes da exibição, conforme a Figura 2.3. A problemática da compressão de vídeo pode ser vista de duas formas: como representar o vídeo com a maior fidelidade possível, dentro de uma determinada taxa de dados, ou como representar o vídeo na menor taxa possível, dentro de limites aceitáveis de qualidade. Em qualquer um dos casos, existe um comprimisso entre taxa de dados e qualidade.

Existem dois tipos básicos de compressão de imagens: com perdas e sem perdas. Ambas baseiam-se na remoção das redundâncias presentes nas imagens. Os codecs que trabalham sem perdas exploram redundâncias estatísticas (entropia), que podem ser removidas sem perda de qualidade, de forma que os dados resultantes são uma cópia perfeita dos dados originais. Porém, esse processo sozinho é limitado e possibilita uma taxa de compressão em torno de 3 a 4 vezes. A maioria dos codecs existentes, buscando atingir uma maior capacidade compressão, operam com algum tipo de degradação nas imagens. essas perdas exploram redundâncias subjetivas, removendo elementos de imagem que não afetam significativamente a qualidade visual percebida pelo usuário. Dois tipos de redundância subjetiva são normalmente exploradas: temporal, dada pela elevada correlação entre frames vizinhos, ou seja, temporalmente adjacentes; e espacial, dada pela elevada correlação entre os pixels vizinhos dentro de um mesmo frame (espacialmente adjacentes).

Dessa forma, um típico codec de vídeo é composto por 3 blocos principais [1]: um modelo temporal, um modelo espacial e um codificador de entropia, conforme a Figura 2.4. A entrada do codec é um vídeo no seu formato original "bruto". O modelo temporal explora a correlação entre os frames vizinhos, subtraindo blocos de pixels já processados dos blocos em processamento. A

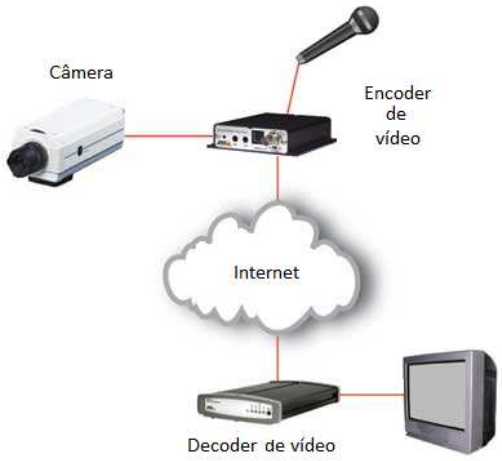


Figura 2.3: Encoder/Decoder.

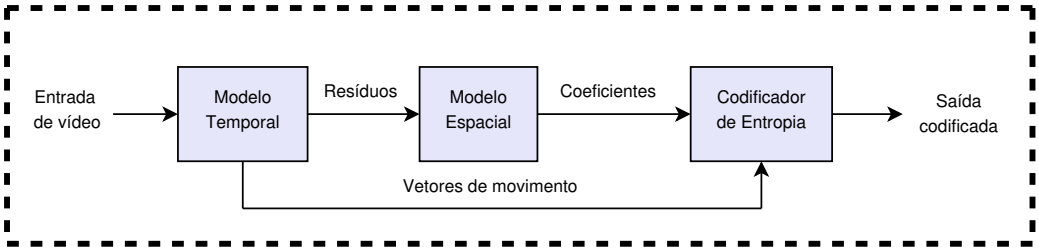
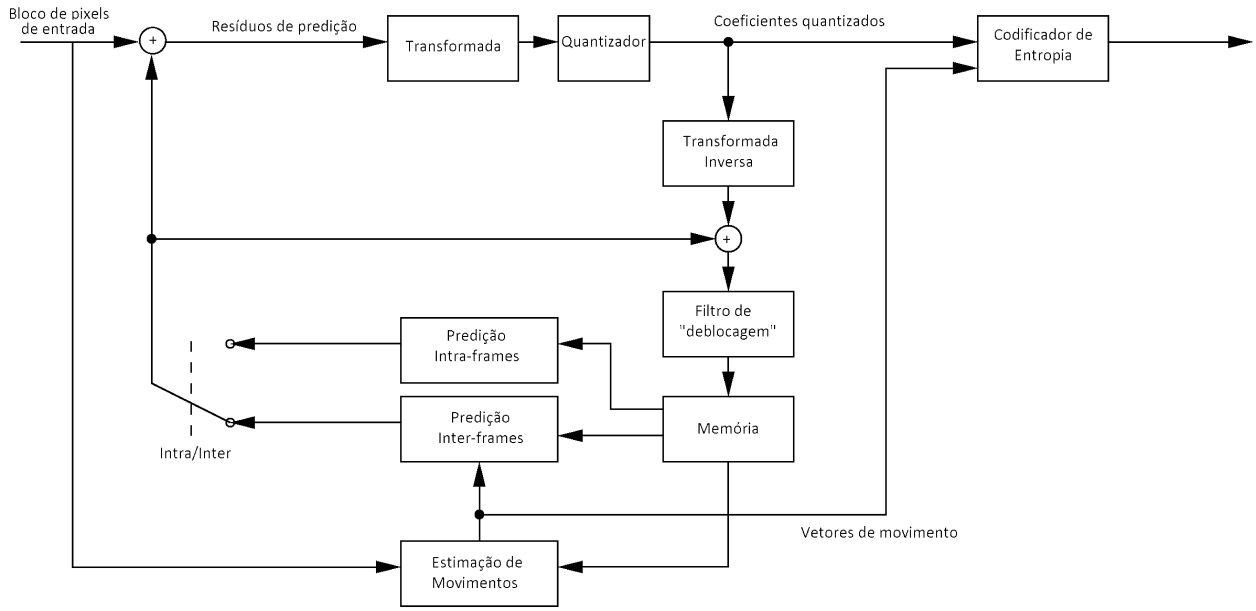


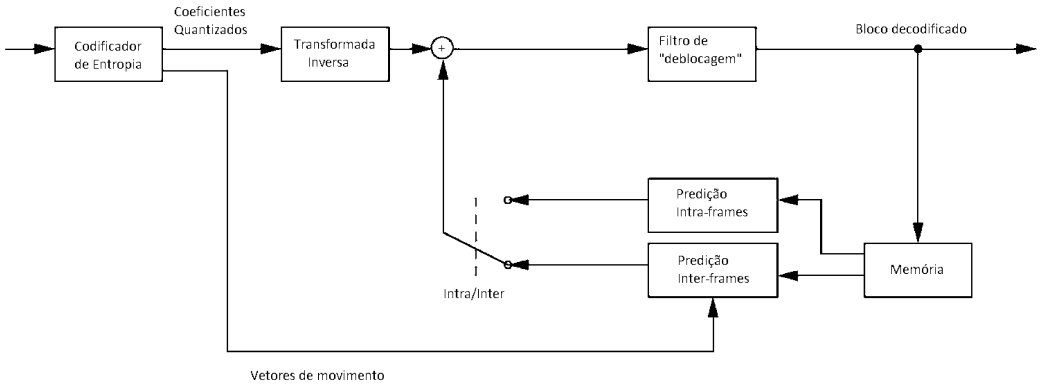
Figura 2.4: Blocos básicos utilizados na codificação de vídeo.

saída do modelo temporal é composta pelos "resíduos" dessa "predição" e pelos "vetores de movimento" que indicam os blocos que foram utilizados como "referência". Os resíduos vão para o modelo espacial, responsável por extrair redundâncias locais, causadas pela elevada correlação entre amostras vizinhas, dentro de um mesmo frame. Normalmente isto acontece através do uso de "transformadas", que levam os dados para o domínio da frequência, onde muitos dos coeficientes podem ser eliminados no processo de "quantização", com pouca ou nenhuma degradação perceptível ao olho humano. Os coeficientes restantes, junto com os vetores de movimento, são a entrada do codificador de entropia, responsável por extrair as redundâncias estatísticas presentes nos dados. A saída do sistema é um *stream* de dados codificados, compostos pelos resíduos de predição, vetores de movimento e algum cabeçalho. O decodificador tem a função de, a partir desses dados codificados, recuperar o vídeo original, ou uma aproximação dele. Normalmente isso é feito a partir de operações inversas às do codificador.

A maioria dos codecs existentes atualmente segue uma estrutura denominada híbrida, que explora ambas as redundâncias espacial e temporal. O diagrama básico de um codificador pode ser visto na Figura 2.5 (a). esse esquema, ou algo bem similar, pode ser visto nos codecs H.261, H.263, MPEG-1, MPEG-2, MPEG-4 Visual e H.264, por exemplo. O decodificador conceitualmente trabalha de forma inversa, a partir do codificador de entropia, conforme pode ser visto na Figura 2.5 (b). Os detalhes de cada bloco serão vistos nas seções a seguir. A maioria dos algoritmos que serão descritos a seguir são os utilizados no padrão H.264/AVC, que



(a)



(b)

Figura 2.5: Diagrama geral de um codificador (a) e decodificador (b) de vídeo (adaptado de [5]).

é o estado da arte e referência internacional na área de compressão de vídeo.

2.2.1 Estrutura de Codificação

Ambos os processos, codificação e decodificação, ocorrem frame-a-frame, onde cada frame é sub-dividido em blocos menores de codificação, normalmente chamados de macro-blocos ou sub-macroblocos, os quais são processados um-a-um. Um conjunto de frames inter-relacionados é chamado de Grupo de Imagens (GOP). Para uma imagem colorida, cada macro-bloco ou sub-macrobloco é composto pelas componentes de luminância e as respectivas componentes de crominância. Essa estrutura pode ser vista na Figura 2.6.

O tipo de predição (temporal ou espacial) utilizada em cada frame determina a estrutura de cada GOP. Esses podem ser compostos por frames dos tipos I, P ou B. Os frames I são independentes e codificados explorando apenas a redundância espacial presente neles mesmos. Já os frames P exploram também a redundância temporal existente entre ele e algum outro frame adjacente, antecedente a ele, chamado de frame de referência. Os frames do tipo B também exploram redundâncias temporais, porém podem fazer uso de mais de um frame de referência, em um processo denominado de bi-predição. Esses frames de referência, não necessariamente precisam ser anteriores (em ordem de exibição) ao frame em processamento. Uma sequência codificada de vídeo sempre se inicia com um frame chamado de IDR (*Instataneous Decoder Refresh*), que é um frame do tipo I com a função de sinalizar ao decodificador o início (e consequentemente o fim) de uma sequência.

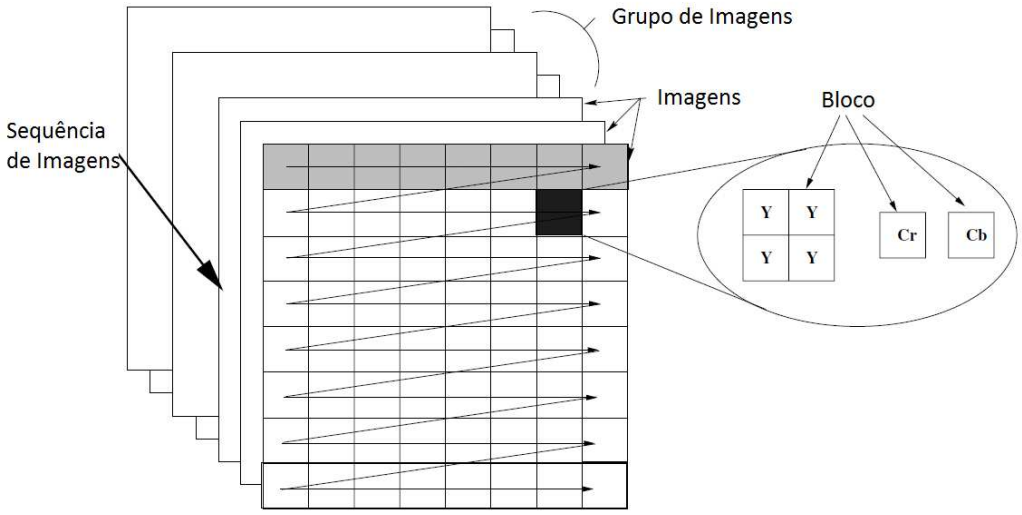


Figura 2.6: Estrutura de codificação (adaptado de [6]).

A Figura 2.7 mostra alguns possiveis formatos utilizados no H.264/AVC. Um GOP pode ser composto apenas por frames I e P, de acordo com a Figura 2.7 (a). Nesse formato o primeiro frame (I) é codificado com predição espacial, enquanto os frames subsequentes (P) são codificados com predição temporal, utilizando o frame anterior como referência. Alternativamente, não

apenas o frame imediatamente anterior, mas "N" frames anteriores podem ser usados como referência para os frames P (Figura 2.7 (b)). O tipo de GOP mais comum é formado por sucessivas duplas de frames B, intercalados entre frames I e P conforme a Figura 2.7 (c). Os frames de tipo P podem utilizar como referência frames I ou P antecedentes, enquanto os frames de tipo B podem utilizar frames I ou P anteriores ou posteriores. GOPs apena com frames I também são possíveis, os quais não fazem uso de nenhuma predição temporal.

2.2.2 Predição Intra-frames (espacial)

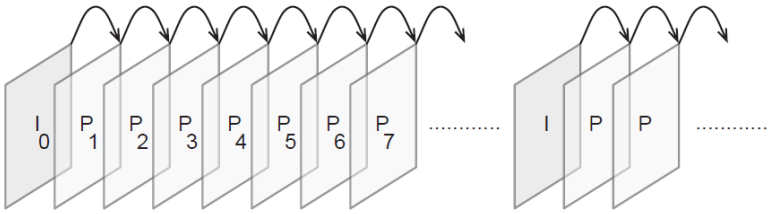
O processo de predição intra-frames (espacial) envolve prever uma amostra ou grupo de amostras a partir de outras anteriormente codificadas/transmitidas dentro desse mesmo frame. A Figura 2.8 mostra um determinado pixel X que está prestes a ser codificado. Os pixels vizinhos (já codificados) são usados como referência para formar uma predição de X, que será subtraída de X, de forma que só o resíduo dessa subtração (menor energia) será passado para as etapas posteriores do processamento. O decodificador, por sua vez, faz a mesma predição e soma ao resíduo decodificado para restaurar o valor do pixel X. A eficiência desse processo depende da qualidade da predição realizada. Se a predição for boa, a energia residual será baixa e possivelmente, o processo de transformada e quantização gerará coeficientes nulos que poderão ser descartados. A escolha das amostras que serão usadas como referência pode ser feita de diversas maneiras, e essa escolha deve ser sinalizada para o decodificador.

No H.264/AVC, cada bloco de 4×4 pixels é predito por amostras espacialmente vizinhas como mostrado na Figura 2.9 (a). As 16 amostras (a-p) de um bloco podem ser preditas a partir das amostras adjacentes já codificadas (A-M). Para cada bloco, o codificador pode escolher entre nove modos (direções) possíveis de predição. Além do modo DC, oito modos direcionais são especificados, de acordo com a Figura 2.9 (b). A Figura 2.10 mostra os nove possíveis modos de predição intra-frames para blocos 4×4. esses modos são adequados para prever estruturas, texturas ou bordas direcionais das imagens, nos vários ângulos possíveis.

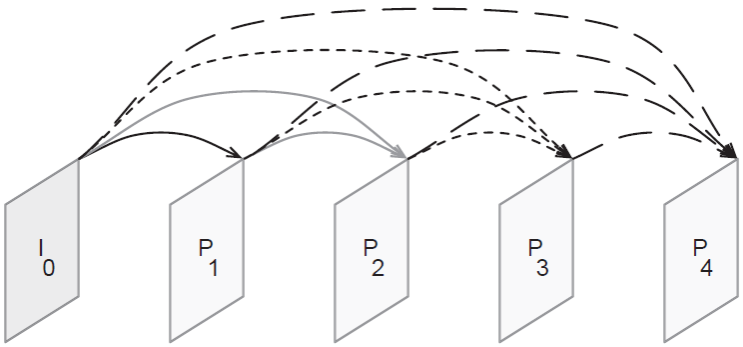
2.2.3 Predição Inter-frames (temporal)

A predição inter-frames (temporal), consiste na redução das redundâncias existentes entre frames temporalmente vizinhos. Para isto, gera-se uma predição do frame atual, a partir de um frame anterior, chamado de frame de referência, e essa predição é subtraída do frame em processamento. Apenas a diferença entre os frames segue para as próximas etapas do codificador. esse processo pode ser visto na Figura 2.11. Pode-se observar que ainda existe uma quantidade significativa de energia no frame residual (indicada pelas áreas claras). esse resíduo é decorrente da movimentação dos objetos na cena.

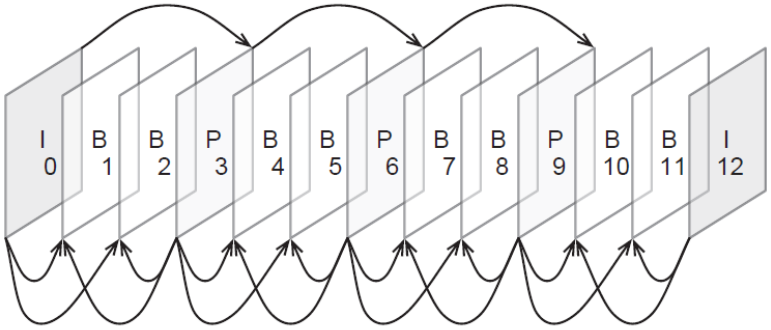
A qualidade dessa predição pode ser melhorada através compensação do movimento entre o frame atual e o de referência. A movimentação dos objetos em uma cena pode ser modelada por um deslocamento linear de pixels, ou blocos de pixels, entre os frames. Dessa forma, pode-se procurar nos diversos frames já processados a área que mais se aproxima do bloco em processamento. Isto melhora consideravelmente a predição, e consequentemente reduz a energia dos frames residuais. Essa técnica é conhecida como Estimação e Compensação de Movimentos



(a) GOP IPPP... com possibilidade de referência apenas ao frame anterior



(b) GOP IPPP... com possibilidade de referência a "N" frames anteriores



(c) GOP IBBP... com possibilidade de referência a frames anteriores e/ou posteriores

Figura 2.7: Tipos de GOP (frames I,P e B) (adaptado de [1]).

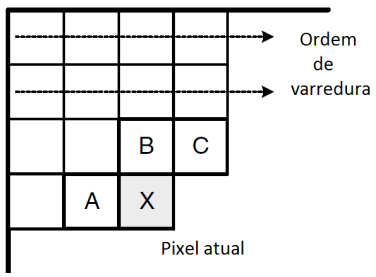
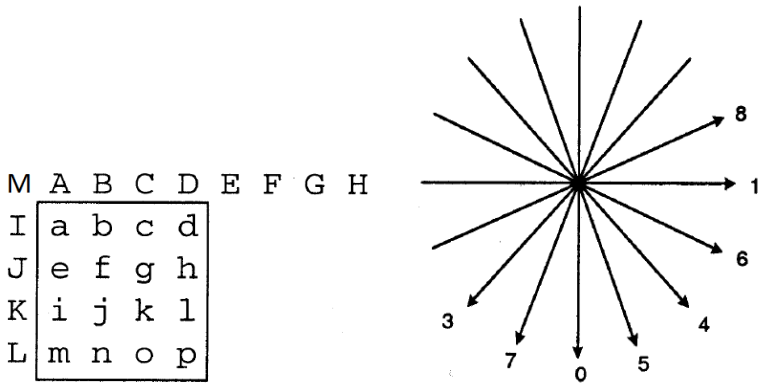


Figura 2.8: Predicao espacial (adaptado de [7]).



(a) Predição intra-frames das amostras a-p, a partir das amostras vizinhas A-M

(b) Oito possíveis direções de predição

Figura 2.9: (a) Predição das amostras a-p, a partir de A-M. (b) Direções de predição (adaptado de [8]).

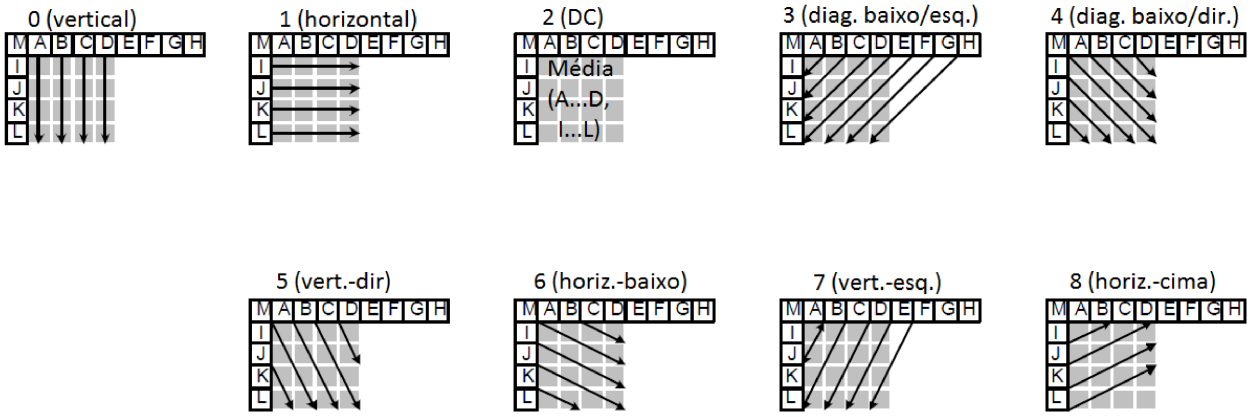


Figura 2.10: Modos (direções) de predição intra-frames especificados no H.264/AVC (adaptado de [1]).

e será detalhada a seguir.

Estimação e Compensação de Movimentos

A Estimação e Compensação de Movimentos funciona da seguinte forma: primeiro ocorre uma busca, no frame de referência, por um bloco que melhor represente o bloco em processamento. Essa busca é feita numa área em torno da posição do bloco que se quer prever. O critério de escolha do bloco que será usado como referência normalmente se dá pela comparação do resíduo (resto) gerado pela subtração entre o bloco candidato e o bloco em processamento. O resíduo de menor energia determina o bloco que será usado como referência. esse processo de determinação do melhor bloco é conhecido como estimação de movimento. A Figura 2.12 ilustra esse processo. Em seguida o bloco de referência escolhido é subtraído do bloco em processamento para formar o bloco de resíduo (compesação de movimento). O bloco residual, junto com os vetores que indicam a posição do bloco de referência usado (vetor de movimento) são passados para as etapas posteriores do processamento. O decodificador utiliza os vetores de movimento recebidos para recriar a região de predição, decodifica o bloco residual e soma ao preditor para reconstruir a versão original do bloco.

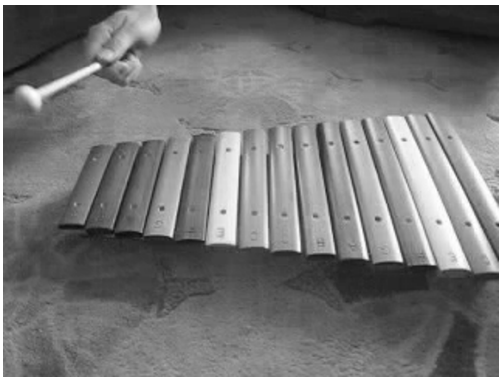
Essa técnica tem algumas desvantagens como, por exemplo, o fato de que objetos reais numa cena nem sempre tem bordas nítidas que se encaixam dentro dos blocos utilizados. Da mesma forma, alguns tipos de movimentos são difíceis de compensar, como, por exemplo, objetos que se deformam ou rotacionam.

Em alguns casos, uma melhor predição é encontrada ao se procurar nas amostras interpoladas da região de referência. O encoder acha a melhor predição entre os pixels inteiros da imagem de referência, e, em seguida, busca nos pixels interpolados dessa região outra predição que minimize ainda mais o resíduo. Essa técnica é conhecida como Compensação de Movimento em Sub-pixels e seu uso tende a melhorar a precisão do método.

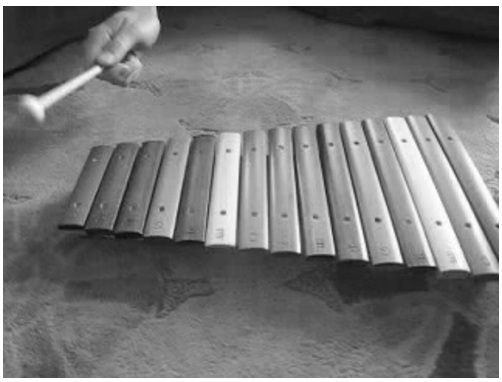
2.2.4 Transformadas

O papel do estágio de transformada em um codec é representar a imagem (normalmente os resíduos de predição) em outro domínio onde possa ser explorada a redundância (correlação) espacial ainda presente. esse tipo de representação, geralmente no domínio da frequência, concentra a maior parte da informação presente no sinal em algumas amostras (baixa frequência), permitindo a eliminação de outras (altas frequências) no processo de quantização, sem que haja degradação perceptível da imagem (o sistema visual humano é menos sensível às variações de alta frequência). O princípio básico desse processo pode ser visto na Figura 2.13. Um grupo de pixels é transformado e passa a ser representado pelos coeficientes da transformada. A maior parte da energia está concentrada em alguns coeficientes de forma que, após o processo de quantização, muitos deles podem ser zerados, e somente os restantes transmitidos. No receptor ocorre o processo da transformada inversa, e os pixels são recuperados a partir dos coeficientes recebidos.

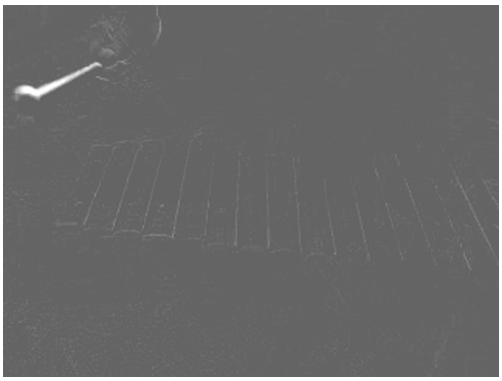
Existem muitos tipos de transformadas aplicáveis à compressão de vídeo. Algumas transformadas são aplicadas na imagem completa. Essas, muitas vezes têm desempenho melhor do que



(a) Frame atual



(b) Frame de referência



(c) Resíduo

Figura 2.11: Predição inter-frames. Frame atual (a), de referência (b) e resíduo (c).

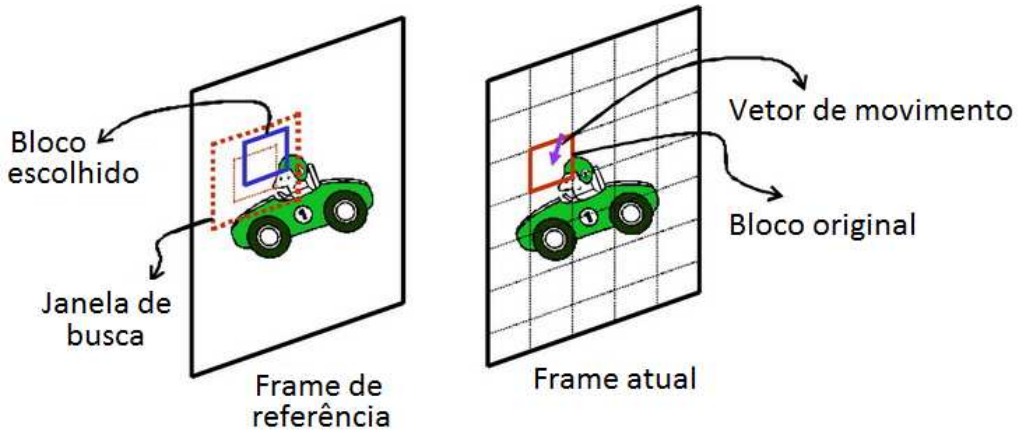


Figura 2.12: Estimação e compensação de movimentos.

transformadas aplicadas em blocos de pixels, porém com altos requerimentos de memória. A mais comum delas é a DWT (*Discrete Wavelet Transform*), utilizada em codecs de vídeo como o DIRAC [9].

As transformadas de maior destaque são as transformadas baseadas em blocos, que normalmente operam em blocos quadrados de pixels de imagem ou resíduos de predição. O seu uso se justifica principalmente pelos baixos requerimentos de memória e pela adaptação ao processo de Estimação e Compensação de Movimentos, que também é realizado em blocos. As transformadas de bloco mais utlizadas são a DCT [10] e suas derivadas, a KLT (*Karhunen-Loève Transform*) [11], Transformada Haar [12] e Transformada Walsh-Hadamard [13]. A escolha da transformada adequada é um compromisso entre a capacidade de cada uma de concentrar energia e seu custo computacional. A KLT é considerada ótima em termos de concentração de energia, porém com implementação complexa. A DCT é quase-ótima e é a que apresenta o melhor comprimisso entre eficiência e complexidade.

As transformadas em bloco tendem a gerar artefatos nas bordas dos blocos devido ao casamento imperfeito dessas fronteiras com os objetos reais em uma cena. esse efeito é conhecido como efeito de bloco ou simplesmente "blocagem" (*blockiness*). Blocos menores se adaptam melhor às bordas dos objetos e reduzem esse efeito, ao custo de uma maior complexidade computacional. Em muitos codecs, utiliza-se um filtro de "deblocagem" que suaviza as bordas dos blocos, com o objetivo de reduzir ou até mitigar a percepção desse efeito.

A próxima seção trata da DCT, que é a transformada mais utilizada nos codecs atuais.

DCT

A definição mais comum da DCT de uma sequência $f(x)$ 1-D de tamanho N é [10]:

$$C(u) = \alpha(u) \sum_{x=0}^{N-1} f(x) \cos\left[\frac{\pi(2x+1)u}{2N}\right] \tag{2.3}$$

para $u = 0, 1, 2, ..., N-1$. Da mesma forma, a transformada inversa é definida como:

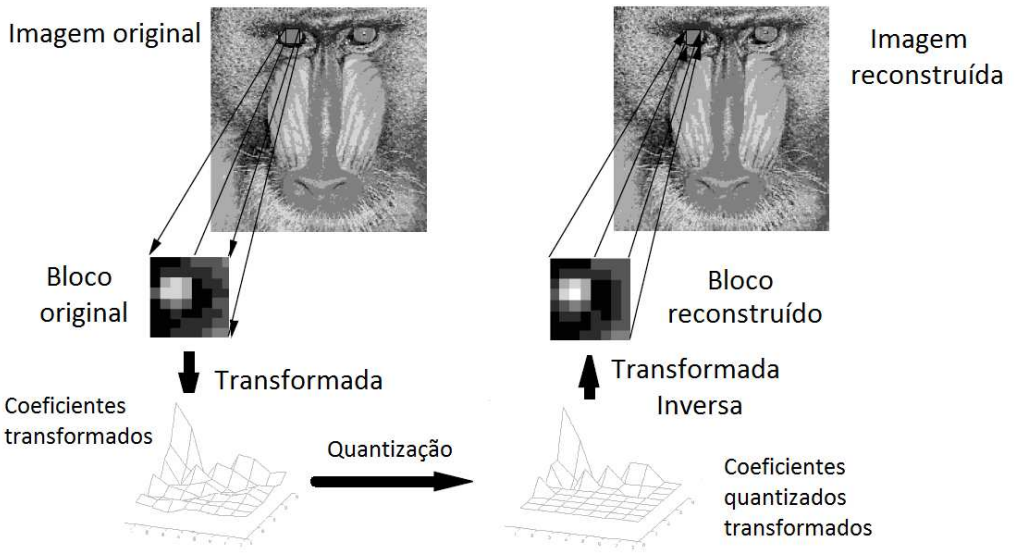


Figura 2.13: Princípio básico do processo de transformada para compressão de vídeo (adaptado de [14]).

$$f(x) = \sum_{u=0}^{N-1} \alpha(u)C(u)\cos[\frac{\pi(2x+1)u}{2N}] \tag{2.4}$$

Para ambas equações, $\alpha(u)$ vale:

$$\alpha(u) = \begin{cases} \sqrt{\frac{1}{N}}, & \text{se } u = 0 \\ \sqrt{\frac{2}{N}}, & \text{se } u \neq 0 \end{cases} \tag{2.5}$$

A DCT representa uma sequência finita de dados em termos de uma soma de cossenos de diferentes frequências. A Figura 2.14 mostra um exemplo dessas cossenóides para N=8. Essas formas de onda são chamadas de funções-base ou *kernels* da transformada. Para $u = 0$, a função-base tem valor constante e dá origem ao coeficiente DC da transformada, enquanto, para $u = 1, 2, ...7$, as funções-base são cossenóides de frequências crescentes e resultam nos chamados coeficientes AC da transformada.

A DCT é uma transformada similar a DFT (*Discrete Fourier Transform*), porém com duas vantagens principais, decorrentes do fato de suas funções-base serem estritamente cossenoidais: 1) é uma transformada real, em contraste com a DFT, que tem exponenciais complexas como funções-base; 2) suas propriedades de simetria par fazem com que a extensão periódica do sinal no tempo não apresente descontinuidades nas bordas (Figura 2.15), o que não acontece com a DFT, na qual a extensão periódica do sinal a ser transformado tem simetria ímpar. Sabe-se que essas descontinuidades no tempo resultam em artefatos no domínio da frequência, que diminuem a taxa de convergência da série de Fourier, implicando na necessidade de mais coeficientes para uma representação precisa do sinal. Disto decorre o grande poder de compactação de energia e

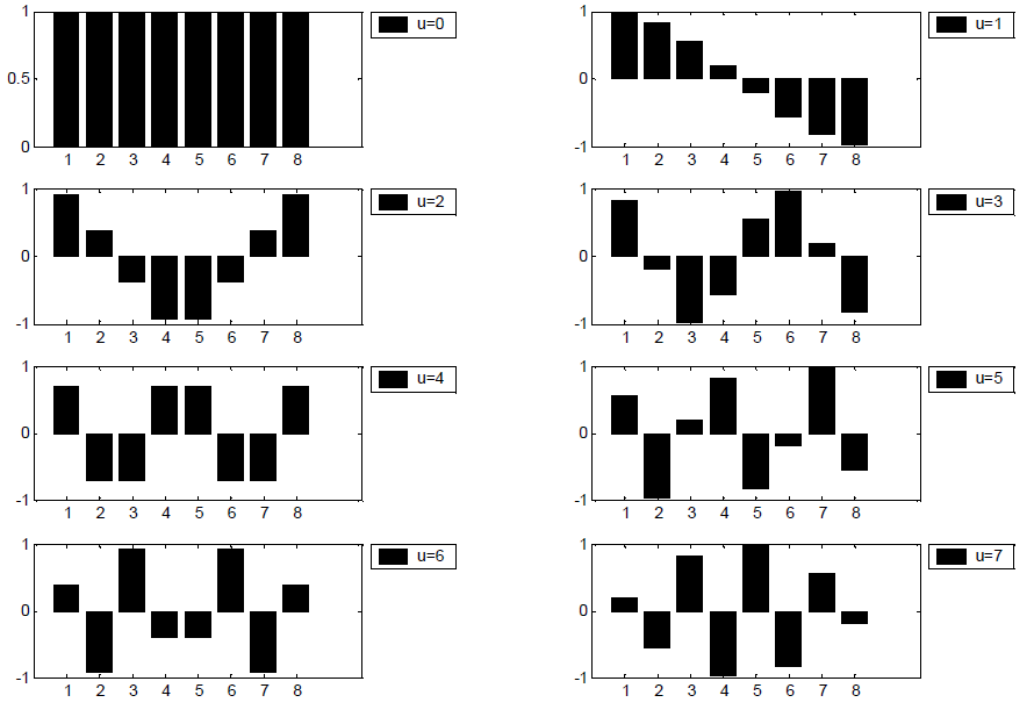


Figura 2.14: Funções-base da DCT 1-D (N=8).

descorrelação da DCT, aproximando-a do limite ótimo da transformada Karhunen-Loève, porém com implementação bem menos complexa.

A DCT-2D é uma extensão direta de 2.3 e é dada por:

$$C(u,v) = \alpha(u)\alpha(v)\sum_{x=0}^{N-1}\sum_{y=0}^{N-1} f(x,y)\cos[\frac{\pi(2x+1)u}{2N}]\cos[\frac{\pi(2y+1)u}{2N}] \tag{2.6}$$

para $u, v = 0, 1, 2, ..., N-1$ e $\alpha(u)$ e $\alpha(v)$ são definidos em 2.5. Da mesma forma, a DCT-2D inversa é definida como:

$$f(x,y) = \sum_{u=0}^{N-1}\sum_{v=0}^{N-1} \alpha(u)\alpha(v)C(u,v)\cos[\frac{\pi(2x+1)u}{2N}]\cos[\frac{\pi(2y+1)u}{2N}] \tag{2.7}$$

As funções-base para N=8 podem ser vistas na Figura 2.16. Mais uma vez pode-se notar que as funções-base são cossenóides que vão aumentando de frequência tanto na vertical quanto na horizontal. Dessa forma, na saída da DCT-2D, os coeficientes de baixa frequência (incluindo o valor DC) ficam localizados no canto superior esquerdo da matriz, enquanto os coeficientes de maiores frequências vão se distribuindo em ordem crescente ao longo da matriz, na direção do canto inferior direito.

As propriedades citadas anteriormente fazem da DCT a transformada mais utilizada para aplicações de compressão de vídeo. É importante notar que a transformada em si não introduz

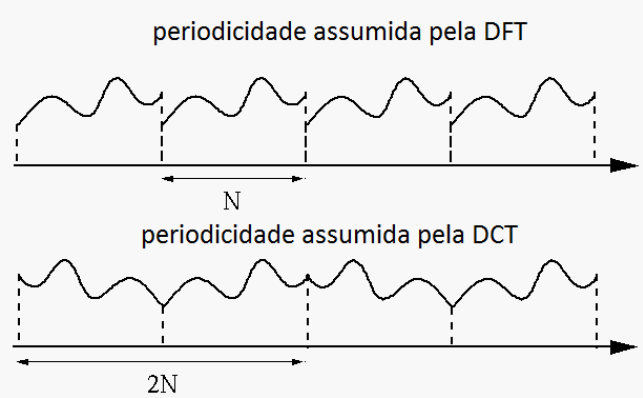


Figura 2.15: Extensões periódicas da DFT e da DCT.

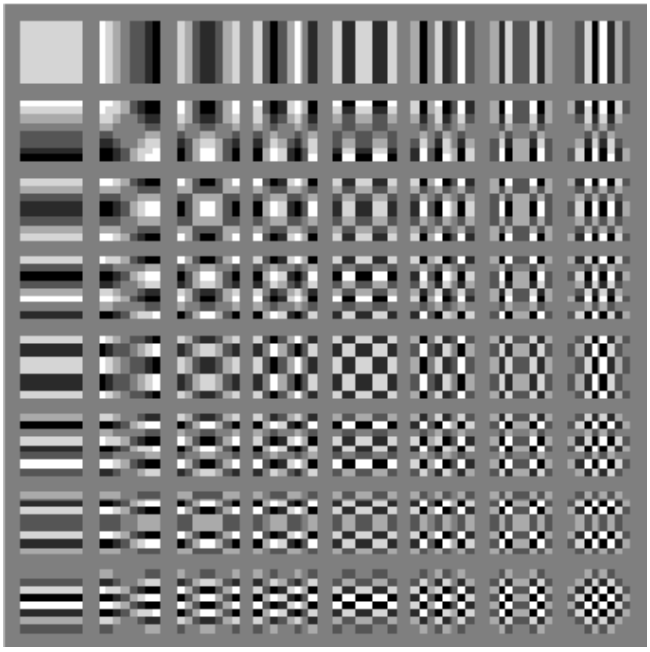
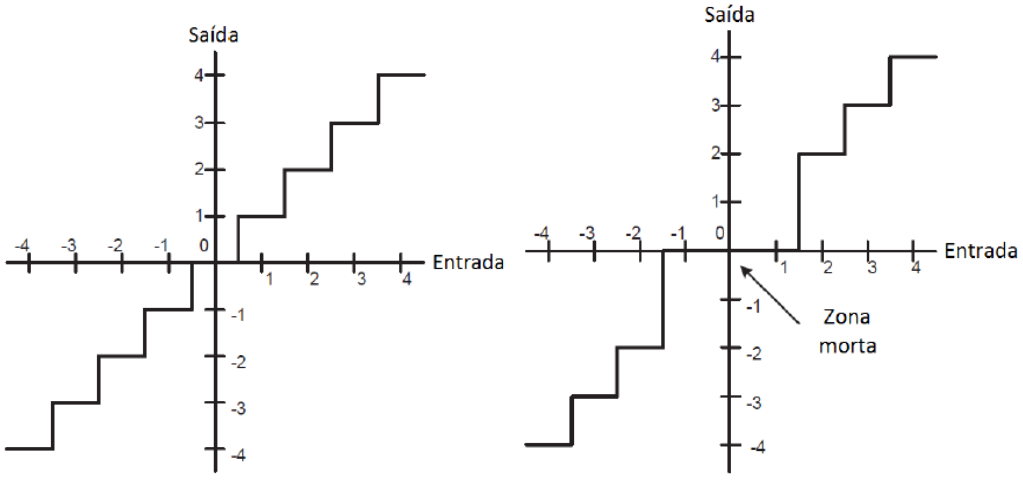


Figura 2.16: Funções-base da DCT 2-D (N=8).

nenhum ganho de compressão ao codec. Sua principal função é transformar os dados para o domínio da frequência, de forma que o quantizador possa eliminar os coeficientes menos significantes (alta frequência), com pouca ou nenhuma degradação perceptível à visão humana.

2.2.5 Quantização

Um quantizador mapeia um sinal de entrada com uma determinada faixa de valores contínuos, para outra faixa de valores, discreta e finita, de forma que o sinal na sua saída possa ser representado com um número finito de bits. O processo pode ser entendido como um mapeamento do conjunto de números reais para conjunto dos inteiros ($\mathbb{R}$ para $\mathbb{Z}$). Como no decodificador não é possível determinar o valor original da amostra, a quantização é um processo irreversível que implica em perdas na imagem. A Figura 2.17 mostra dois exemplos de quantizadores: linear, com um mapeamento uniforme entre os valores de entrada e saída; e não-linear com "zona morta", na qual pequenos valores são mapeados para zero. Nos dois casos, a faixa real de valores da abcissa é mapeada no conjunto de valores inteiros das ordenadas.



(a) Quantizador linear (b) Quantizador não-linear com zona morta

Figura 2.17: Exemplos de quantizadores (adaptado de [1]).

Normalmente o processo de quantização é feito com o auxílio das matrizes de quantização, e pode ser modelado por:

$$C_{i,j} = round(\frac{D_{i,j}}{QP_{i,j}}) \tag{2.8}$$

onde $C_{i,j}$ é o valor quantizado, $D_{i,j}$ a amostra original (normalmente um coeficiente da DCT), e $QP_{i,j}$ o parâmetro de quantização, também chamado de "passo" do quantizador. O parâmetro de quantização é variado de acordo com a taxa de dados que se deseja obter na saída do codificador. Quanto maior, mais coeficientes podem ser zerados, implicando em uma menor taxa de dados na saída, porém numa maior degradação da imagem. Sua escolha também é feita de forma

a explorar a menor sensibilidade da visão humana às altas frequências presentes na imagem: usam-se valores menores de $QP_{i,j}$ para os coeficientes de baixa frequência, e valores maiores para os de alta. Dessa forma pode-se atingir um maior ganho de compressão com uma degradação menos perceptível a visão humana. A Figura 2.18 mostra um exemplo da aplicação de uma matriz de quantização aos dados resultantes da DCT.

$$D = \begin{bmatrix} 162.3 & 40.6 & 20.0 & 72.3 & 30.3 & 12.5 & -19.7 & -11.5 \\ 30.5 & 108.4 & 10.5 & 32.3 & 27.7 & -15.5 & 18.4 & -2.0 \\ -94.1 & -60.1 & 12.3 & -43.4 & -31.3 & 6.1 & -3.3 & 7.1 \\ -38.6 & -83.4 & -5.4 & -22.2 & -13.5 & 15.5 & -1.3 & 3.5 \\ -31.3 & 17.9 & -5.5 & -12.4 & 14.3 & -6.0 & 11.5 & -6.0 \\ -0.9 & -11.8 & 12.8 & 0.2 & 28.1 & 12.6 & 8.4 & 2.9 \\ 4.6 & -2.4 & 12.2 & 6.6 & -18.7 & -12.8 & 7.7 & 12.0 \\ -10.0 & 11.2 & 7.8 & -16.3 & 21.5 & 0.0 & 5.9 & 10.7 \end{bmatrix}$$

(a) Coeficientes resultantes da DCT

$$Q_{50} = \begin{bmatrix} 16 & 11 & 10 & 16 & 24 & 40 & 51 & 61 \\ 12 & 12 & 14 & 19 & 26 & 58 & 60 & 55 \\ 14 & 13 & 16 & 24 & 40 & 57 & 69 & 56 \\ 14 & 17 & 22 & 29 & 51 & 87 & 80 & 62 \\ 18 & 22 & 37 & 56 & 68 & 109 & 103 & 77 \\ 24 & 35 & 55 & 64 & 81 & 104 & 113 & 92 \\ 49 & 64 & 78 & 87 & 103 & 121 & 120 & 101 \\ 72 & 92 & 95 & 98 & 112 & 100 & 103 & 99 \end{bmatrix}$$

(b) Matriz de quantização

$$C = \begin{bmatrix} 10 & 4 & 2 & 5 & 1 & 0 & 0 & 0 \\ 3 & 9 & 1 & 2 & 1 & 0 & 0 & 0 \\ -7 & -5 & 1 & -2 & -1 & 0 & 0 & 0 \\ -3 & -5 & 0 & -1 & 0 & 0 & 0 & 0 \\ -2 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

(c) Coeficientes quantizados

Figura 2.18: Exemplos da aplicação do quantizador sobre os coeficientes da DCT.

2.2.6 Codificação de Entropia

Após a quantização, os blocos (matrizes) devem ser convertidos em um vetor de coeficientes que seguirá para o codificador de entropia. A ordenação dos coeficientes é utilizada principalmente em conjunto com a transformada, de forma que os coeficientes significativos (maior energia) sejam "lidos" primeiro e agrupados conjuntamente. Após a aplicação da DCT, esses coeficientes ficam concentrados na região superior esquerda do bloco transformado (Figura 2.19). Esse tipo de ordenação é conhecido como varredura zig-zag ou *zig-zag scan*. Como muitos desses coeficientes quantizados tem valor nulo, o vetor resultante tem alguns coeficientes significativos nas primeiras amostras, seguidos de sequências de zeros, que são chamadas de "corridas de zeros". Esse formato típico pode ser representado de forma simplificada a partir do par (run,level), correspondente ao número de zeros (run) que precedem um determinado coeficiente significativo (level). Por exemplo, o par (13,8) indica a presença de 13 zeros seguidos por um coeficiente de valor 8. Um dos formatos de zig-zag scan utilizado em blocos 8x8 no H.264/AVC pode ser visto na Figura 2.20.

Em seguida ocorre a codificação de entropia propriamente dita. A função do codificador de entropia é converter uma série de símbolos que representam a sequência de vídeo em um *stream* de bits que possa ser transmitido ou armazenado. Os símbolos de entrada podem ser não apenas os coeficientes citados anteriormente, mas também vetores de movimento, marcadores de sincronização, cabeçalhos ou informações adicionais da codificação.

O codificador procura um determinado código que maximize a eficiência de codificação, reduzindo o tamanho médio da palavra do alfabeto de saída. Essa redução é atingida a partir da remoção de redundâncias ainda presentes nos dados. O menor tamanho de código possível é a própria entropia do alfabeto de entrada (daí o nome da codificação). É indispensável que o código resultante seja unicamente decodificável, de forma que o decodificar consiga, a partir dos dados de entrada, restaurar exatamente os dados originais.

As técnicas mais comuns de codificação de entropia podem ser classificadas em dois grupos: técnicas estatísticas e técnicas baseadas em dicionários. Na primeira existe a necessidade de que as probabilidades dos símbolos da fonte estejam disponíveis antes da codificação. Se toda a sequência de dados já é conhecida antes da codificação, essas probabilidades (distribuição) podem ser pré-calculadas. Caso contrário, cálculos estatísticos vão sendo realizados na medida em que se vai conhecendo os dados da fonte. Nas técnicas baseadas em dicionários, sempre existe a possibilidade de que os símbolos do dicionário não sejam uma boa representação de toda a sequência de dados que está por vir, o que implicará em um modelo estatístico equivocado e, consequentemente, numa codificação ineficiente. Por isso a maioria dos codificadores opera "on-line", inferindo o dicionário disponível a partir de partes anteriores da mensagem. Dessa forma, o dicionário é implicitamente transmitido, dado que o decodificador pode, em tempo real, fazer os mesmos ajustes ao seu dicionário.

Os valores dos pixels em um frame de vídeo digital são pré-quantizados em palavras de tamanho fixo, tipicamente com 8 ou 16 bits de precisão por componente de cor. Entretanto, nem todos os valores de cor aparecem com a mesma probabilidade numa determinada cena. Pode-se então reduzir o número médio de bits por pixel ao associar-se palavras maiores aos valores que aparecem com menor frequência e palavras menores aos valores mais frequentes.

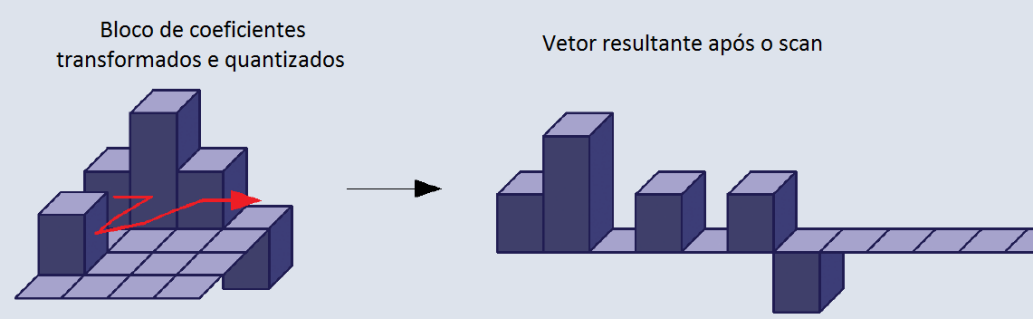


Figura 2.19: Organização dos coeficientes após a DCT e vetor resultante (adaptado de [5]).

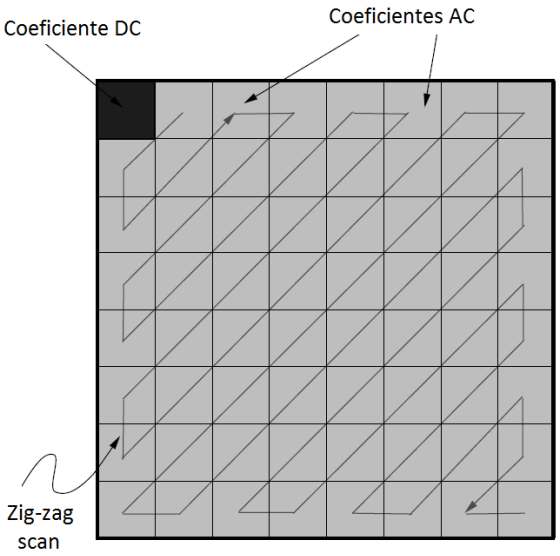


Figura 2.20: Ordenação em zig-zag dos coeficientes.

Esse método, chamado de Codificação de Tamanho Variável, é amplamente utilizado nos codecs atuais, porém, sofre da desvantagem de utilizar um número inteiro de bits para cada símbolo. A Codificação Aritmética aparece como uma alternativa que permite uma maior compressão, aproximando-se do limite teórico de entropia. O H.264/AVC faz uso dessas duas técnicas, que serão detalhadas a seguir.

Codificação Huffman

A Codificação Huffman gera códigos de comprimento variável, nos quais os símbolos mais frequentes têm tamanhos menores do que os símbolos mais raros. De acordo com o esquema original proposto em [15], é necessário que se conheça, a priori, a probabilidade de ocorrência dos símbolos da fonte.

O processo de codificação ocorre da seguinte forma: os símbolos são ordenados de acordo com sua probabilidade de ocorrência (P_i), e, a cada passo, os dois símbolos de menor probabilidade são combinados formando um novo símbolo imaginário, com probabilidade igual a soma das

duas anteriores. Esse processo continua até que se tenha apenas um símbolo (de probabilidade 1) que represente todo o alfabeto da fonte. A representação gráfica desse processo gera a Árvore de Huffman. Um exemplo pode ser visto na Figura 2.21. A partir dela, pode-se determinar a representação de cada símbolo, partindo do símbolo final em direção aos símbolos iniciais, atribuindo, em cada nó, um "1" aos ramos superiores, e um "0" aos inferiores. O código de cada símbolo é dado pelo conjunto de zeros e uns atribuídos sequencialmente aos ramos percorridos da ponta até esse símbolo. De acordo com o exemplo, a sequência "Q2Q3Q1Q5Q1Q1Q2Q6Q7Q1", teria sua versão codificada em formato binário como "001-010-000-100-000-000-001-101-110-000", com 30 bits e tamanho médio de 3 bits por símbolo; e sua versão após a codificação Huffman como "011-001-1-0100-1-1-011-0001-00001-1", com 26 bits e tamanho médio de 2,6 bits por símbolo.

Para decodificar os dados recebidos, o decodificador deve possuir uma cópia local da árvore utilizada na codificação. Ela pode ser transmitida, ou gerada no próprio decodificador, a partir do conjunto de probabilidades do alfabeto (que deve então ser transmitido antes dos dados codificados). Isto gera uma sobrecarga nos dados que precisam ser transmitidos, prejudicando o desempenho geral da codificação Huffman. Outra desvantagem é que o conjunto de probabilidades do alfabeto fonte não pode ser calculado até que se tenha conhecimento de toda a sequência a ser codificada (todo o vídeo). Isto geraria um atraso inaceitável em termos práticos. O H.264 utiliza-se de tabelas pré-calculadas, baseadas na distribuição de probabilidades de um material genérico de vídeo. Dessa forma não é necessária a transmissão de nenhuma informação adicional.

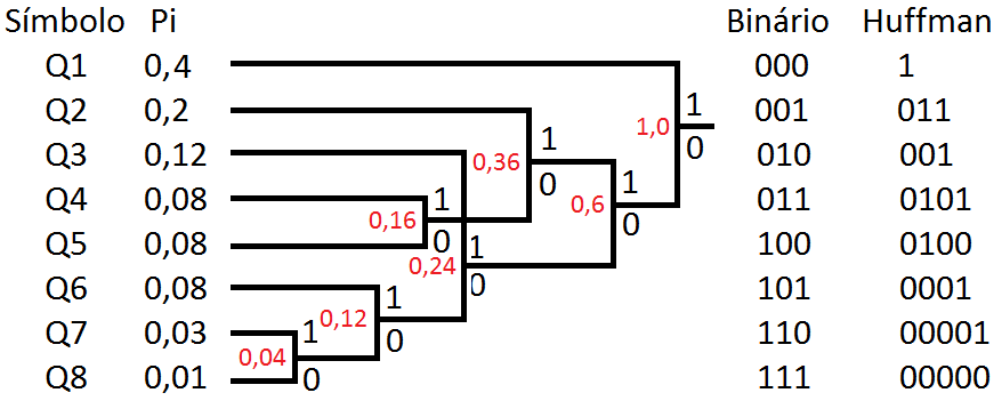


Figura 2.21: Exemplo de Codificação Huffman.

Codificação Aritmética

Os códigos de comprimento variável têm a desvantagem de atribuir um número inteiro de bits a cada símbolo. Sabe-se, pela teoria da informação, que o número ótimo de bits para um símbolo depende do conteúdo de informação que ele possui, ou seja, da quantidade de informação que ele representa dentro do alfabeto. Esse valor é dado por:

$$h(z_i) = \log \frac{1}{p(z_i)} \tag{2.9}$$

onde $h(z_i)$ é relativo à quantidade de informação do símbolo no alfabeto Z, e $p(z_i)$ é a probabilidade de ocorrência desse símbolo. Ainda segundo a teoria da informação, a entropia do alfabeto, que é o limite teórico de compressão, é dada por:

$$H(Z) = \sum_i p(z_i)h(z_i) \tag{2.10}$$

Para atingir uma compressão ótima, cada símbolo codificado deve ter exatamente $\log \frac{1}{p(z_i)}$ bits, e esse número normalmente é fracionário. A eficiência de codificação dos códigos de comprimento variável cai consideravelmente caso o alfabeto tenha um símbolo com probabilidade maior que 50%, dado que não será possível representar o símbolo de maior aparição com menos de 1 bit.

A Codificação Aritmética contorna esse problema obtendo uma maior eficiência de codificação em relação aos códigos de comprimento variável. O codificador converte a sequência dos símbolos de entrada num único número fracional que representa toda a sequência.

O processo de codificação ocorre da seguinte forma: dado o alfabeto de entrada, com as respectivas probabilidades de ocorrência de cada símbolo, atribui-se para cada símbolo um *sub-range* dentro do *range* total inicial [0,1). Por exemplo, dado os símbolos "-2;-1;0;1;2", com as respectivas probabilidades de ocorrência "0,1;0,2;0,4;0,2;0,1", atribuiria-se ao símbolo "-2" o *sub-range* "0 a 0,1", ao símbolo "-1" o *sub-range* "0,1 a 0,3", e assim sucessivamente até o símbolo "2" com o *sub-range* "0,9 a 1". Feita essa distribuição, cada vez que um novo símbolo é codificado, o *range* total é reduzido ao *sub-range* desse símbolo, e novamente sub-dividido em *sub-ranges*, até que se tenha um único *range* (valor) que represente toda a sequência a ser codificada.

O H.264/AVC utiliza um esquema de codificação denominado CABAC (*Context-Based Adaptive Binary Arithmetic Encoding*), originado de [16], o qual atualiza os modelos de probabilidade dos símbolos de acordo com a sequência que está sendo codificada, a partir de estatísticas locais e temporais da imagem.

2.3 Qualidade de Vídeo

Para se avaliar os sistemas de comunicação por vídeo é necessário determinar a qualidade do vídeo que possa ter sofrido algum tipo de degradação ao passar por algum processo, como, por exemplo, transmissão ou compressão. O termo Qualidade de Experiência (QoE) vem sendo comumente usado para descrever a qualidade de vídeo e outros serviços multimídia e engloba não apenas a qualidade das imagens em si, mas vários aspectos que vão desde as características individuais de cada expectador, suas expectativas e experiências anteriores, até o ambiente de exibição (monitor, claridade, distância, qualidade do áudio), dentre outros [17–19].

Percebe-se que esse tipo de avaliação, com tantas variáveis, e a maioria delas de caráter subjetivo, é uma tarefa muito complexa. Muitas das métricas utilizadas hoje levam em consideração apenas um ou alguns desses fatores, e normalmente resumem-se em analisar as distorções

introduzidas no vídeo por algum tipo de processamento. Dessa forma, as métricas para avaliação de vídeo se dividem em dois grupos: subjetivas e objetivas.

As métricas subjetivas baseiam-se no fato de que a percepção visual de uma cena é o resultado de uma interação complexa entre os componentes do sistema visual humano, os olhos e o cérebro e pode ser afetada por uma série de fatores como os já citados anteriormente. Diversos procedimentos de testes para a avaliaçao subjetiva são definidos pela ITU [20]. A maioria deles consiste submeter um grupo de espectadores à exibição de diversos vídeos, dentre os quais estão às versões originais e modificadas dos vídeos. Os espectadores atribuem notas e a média de todas as notas para um determinado vídeo, chamada de MOS (*Mean Opnion Score*), é calculada. É evidente que esses testes têm grande variabilidade, a depender do grupo de espectadores e das sequências de videos apresentados. Dessa forma, o MOS deve ser interpretado mais como uma distribuição estatística do que como um número exato. Para reduzir esse problema, os testes precisam ser repetidos diversas vezes, com diferentes espectadores e sequências de vídeo. É evidente que esse processo é caro e demorado.

A complexidade e custos das medidas subjetivas tornam atrativa a idéia de se medir a qualidade dos vídeos automaticamente utilizando um algoritmo. As métricas objetivas são basicamente algoritmos que, aplicados numa determinada sequência de vídeo, buscam estimar a qualidade desse vídeo e prever o MOS. Elas se dividem em três grupos, Referência Total (*Full-Reference*), Referência Reduzida (*Reduced-Reference*) e Sem Referência (*No-Reference*), a depender da possibilidade, ou não, de se comparar o vídeo codificado com o vídeo original. Existem diversas tipos de medidas objetivas e todas buscam balancear complexidade computacional (velocidade) e correlação com as medidas subjetivas (*benchmark*). Dentre as medidas com referência total, as que mais se destacam são o PSNR (*Peak Signal to Noise Ratio*), o SSIM (*Structural Similarity index*) [21], e o VQM (*Video Quality Metrics*) [22]. O PSNR e sua variante, o BD-PSNR (Bjontegaard-Delta PSNR), são as mais utilizadas e serão detalhadas a seguir.

2.3.1 PSNR

O PSNR (*Peak Signal to Noise Ratio*) é a medida objetiva mais utilizada dada sua simplicidade e velocidade de cálculo. Ele é medido em escala logarítimica e depende do erro médio quadrático (*Mean Square Error* - MSE) entre o frame original e o frame em análise:

$$MSE = \frac{\sum_{i=1}^{M}\sum_{j=1}^{N}\left(f(i,j) - F(i,j)\right)^2}{MxN} \tag{2.11}$$

$$PSNR_{dB} = 10\log_{10}\frac{(2^n-1)^2}{MSE} \tag{2.12}$$

onde, $f(i,j)$ é o frame original, $F(i,j)$ o frame processado, M×N é o tamanho do frame (linhas por colunas) e n o número de bits por pixel.

O resultado é um valor único, em dB, que varia normalmente entre 30 e 40 para sequências de média a alta qualidade. O PSNR sofre de limitações e a principal delas é que ele não é

bem correlacionado com as medidas subjetivas [1]. Um exemplo dessa descorrelação pode ser visto na Figura 2.22, na qual todas as imagens têm o mesmo PSNR e qualidade visuais bem diferentes.

Figura 2.22: Imagens com diferentes degradações e mesmo PSNR [23].

2.3.2 BD-PSNR

O PSNR fornece valores que permitem comparar frame-a-frame os vídeos codificados. Para se analisar a qualidadade de toda a sequência de vídeo, normalmente calcula-se uma média desses valores para diferentes taxas de codificação. O resultado é um gráfico de Taxa×PSNR, no qual cada curva representa um vídeo com determinado processamento, e/ou o vídeo original, conforme pode ser visto na Figura 2.23. Ao se comparar duas curvas, observando um mesmo valor de abscissa, a curva superior indica o codec ou algoritmo que obteve melhor desempenho, pois alcançou melhor PSNR para uma mesma taxa. Normalmente, os valores da curva são extrapolados a partir de alguns poucos valores realmente observados.

Para se obter uma informação comparativa mais precisa, foi desenvolvido em [24], um método de se calcular a diferença média entre duas dessas curvas. Dessa forma, pode-se estimar a taxa economizada de dados entre um algoritmo e outro, para um mesmo valor de PSNR, ou seja, o ganho de codificação (compressão), para uma mesma qualidade objetiva. Esse método é chamado de BD-PSNR (Bjontegaard-Delta PSNR), e é hoje a forma mais utilizada de se comparar o desempenho entre dois codecs, ou entre diferentes ferramentas de um mesmo codec. Os resultados normalmente são exibidos em forma de tabelas. A Figura 2.24 mostra um exemplo de uma dessas tabelas. Ela mostra a taxa de bits economizada ao se utilizar duas novas propostas (CS1 e CS2) em relação a uma terceira proposta de referência. Por exemplo, ao se utilizar a

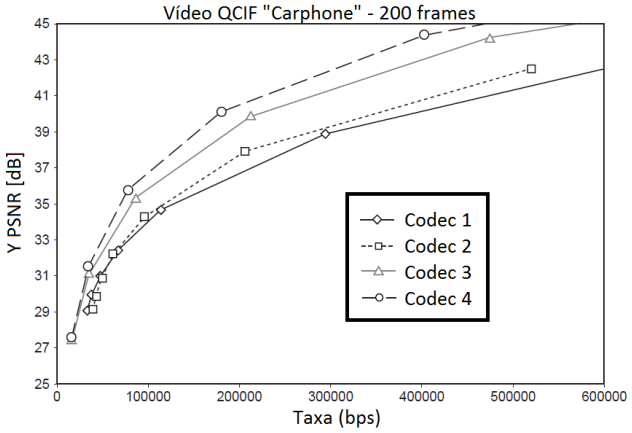


Figura 2.23: Gráfico de comparação Taxa×PSNR.

proposta “CS1” para codificar a sequência “Kimono”, houve uma redução de 38,30% na taxa de dados, para uma mesma qualidade objetiva, em comparação com a proposta de referência.

Grupo	Sequência	Taxa BD CS 1 [%]	Taxa BD CS 2 [%]
A (2560x1600)	Traffic	-27.92	n/a
	People	-17.92	n/a
	Média	**-22.92**	n/a
B1 (1920x1080)	Kimono	-38.30	-38.11
	ParkScene	-24.28	-20.85
	Média	**-31.29**	**-29.84**
B2 (1920x1080)	Cactus	-29.22	-23.43
	BasketballDrive	-35.97	-34.42
	BQTerrace	-41.92	-31.45
	Média	**-35.70**	**-29.77**
C (832x480)	BasketballDrill	-31.88	-14.74
	BQMall	-29.71	-31.45
	PartyScene	-28.09	-16.55
	RaceHorses	-29.67	-22.39
	Média	**-29.84**	**-21.28**
D (416x240)	BasketballPass	-21.97	-14.38
	BQSquare	-43.96	-13.68
	BlowingBubbles	-23.60	-7.44
	RaceHorses	-19.64	-14.23
	Média	**-27.29**	**-12.43**
E (1280x720)	Vidyo 1	n/a	-28.49
	Vidyo 3	n/a	-22.58
	Vidyo 4	n/a	-28.65
	Média	n/a	**-26.57**
	Média total	**-29.60**	**-22.68**

Figura 2.24: Exemplo de tabela para análise do BD-PSNR (adaptado de [25]).

Capítulo 3

HEVC

3.1 Introdução

A padronização dos formatos de codificação de vídeo é importante para assegurar a interoperabilidade entre produtos de diferentes fabricantes. Até recentemente, o último passo nessa direção foi dado em 2003 com o surgimento do padrão H.264/AVC [26], desenvolvido pelo ITU-T VCEG (*Video Coding Experts Group*) em conjunto com o ISO/IEC MPEG (*Moving Picture Experts Group*). Esse rapidamente se tornou referência internacional para inúmeras aplicações de vídeo digital.

Entretanto, a crescente popularidade da TV de alta definição, dos serviços de broadcast para dispositivos móveis, do vídeo 3D, Blu-ray®, além de inúmeras outras aplicações multimídia, fizeram surgir novas demandas de compressão. Essas novas necessidades, aliadas com evolução nas técnicas de codificação, parecem indicar que chegou o momento de se buscar um novo padrão tecnológico.

Frente a isso, um acordo foi feito em janeiro de 2010, criou o JCT-VC (*Joint Collaborative Team on Video Coding*) [27], com a responsabilidade de receber e avaliar propostas (*Call for Proposals - CfP*) [28], numa nova iniciativa de padronização conhecida como HEVC (*High Efficiency Video Coding*). Almeja-se um ganho de compressão em torno de 100% (metade da taxa) comparado com o H.264/AVC, mantendo a mesma qualidade de imagem. A previsão de inicial de lançamento é para 2012.

Como resposta a essa chamada, 27 propostas foram submetidas e avaliadas [29], e constataram-se melhorias significantes em relação ao H.264/AVC. Criou-se então um modelo de teste para investigação (TMuC) [30] que combina elementos-chave de algumas das propostas mais bem ranqueadas [31–36].

O TMuC, por sua vez, foi a base para a primeira versão (HM1) [37] e sua implementação em software [38], a qual vem sendo vem sendo aprimorada ao longo de cada reunião do JCT-VC. Toda documentação dos encontros está disponível em [27]. Um resumo da chamada e seus resultados pode ser visto em [39, 40].

É importante lembrar que o HEVC ainda é um projeto e até o seu lançamento certamente sofrerá inúmeras modificações. Novas ferramentas serão testadas e incluídas, e outras retiradas. Algumas delas são citadas no TMuC, porém ainda estão sob investigação e, portanto, não estão

implementadas. Buscou-se neste trabalho, descrever as ferramentas já incluídas no HM6 [41] (fevereiro de 2012). Algumas ferramentas do TMuC ou de outras versões anteriores do HM são citadas, dada sua relevância.

3.2 Estrutura Geral de Codificação

A estrutura proposta para o HEVC segue o tradicional formato de codificação híbrido (todos os algoritmos propostos seguiram esse mesmo formato), combinando predição e compensação de movimentos inter-frames com predição intra-frames, transformação espacial dos resíduos, filtros de "deblocagem" *in-loop*, e codificação entrópica adaptativa.

Duas configurações de codificação foram sugeridas: Main e Alta Eficiência (*High Efficiency* - HE10). Elas combinam determinados grupos de algoritmos que buscam obter um desempenho específico (se assemelham às definições de perfis do H.264/AVC). As ferramentas para a configuração HE10 têm como objetivo obter alto desempenho, em termos de compressão, sem se preocupar tanto com a complexidade. As ferramentas da configuração Main visam obter um desempenho razoável, mantendo a baixa complexidade do codec. Dentro dessas duas configurações, três tipos de estruturas temporais são possíveis: Intra, Baixo Atraso e Acesso Aleatório.

O modo de codificação Intra permite apenas frames I, ou seja, não há nenhum tipo de predição temporal. Além disso, o mesmo valor para o parâmetro de quantização deve ser usado para todos os frames da sequência. O modo de Baixo Atraso, destinado para aplicações de comunicação em tempo real, não permite reordenamento dos frames entre o processamento no decodificador e sua saida. Dessa forma a predição só pode ser realizada utilizando-se de frames anteriores ao atual, ou seja, os frames de referência da predição temporal devem vir antes do frame em processamento, em termos da ordem de exibição. Apenas o primeiro frame é do tipo I. Os outros podem ser do tipo P ou B. Os valores do parâmetro de quantização podem ser mudados frame a frame, apenas adicionando um offset ao valor do frame inicial. No modo de Acesso Aleatório, frames I devem ser inseridos ciclicamente (em torno de 1s). Os outros frames podem ser P ou B. O valor do parâmetro de quantização pode ser calculado com um offset em relação ao frame I mais próximo. É destinado para aplicações de *broadcast*.

3.3 Particionamento da Imagem

Um dos elementos mais importantes para o ganho de desempenho do HEVC foi a introdução de estruturas de bloco maiores, com um mecanismo flexível e hierárquico de particionamento, em um esquema denominado de *Treeblock Partitioning*, originado das propostas [35, 36, 42].

De forma a facilitar a sintaxe de representação dos blocos, três novos conceitos foram criados: Unidade de Codificação (CU - *Coding Unit*), Unidade de Predição (PU - *Prediction Unit*) e Unidade de Transformada (TU - *Transform Unit*). A estrutura de codificação é caracterizada pelos diversos tamanhos de CU, PU e TU, definidos de uma maneira recursiva e simplificada.

Os frames são divididos em slices, que são compostos por uma sequência de blocos chamados Unidades de Codificação (CU). A estrutura de codificação e o tamanho de cada bloco são definidos de uma maneira recursiva, uma vez que o tamanho máximo da CU (LCU) e a profundidade

da hierarquia são escolhidos. Esses parâmetros são ajustáveis e especificados no cabeçalho do slice.

3.3.1 Unidade de Codificação (CU)

A CU é a unidade básica de processamento e é formada por blocos quadrados que não se sobrepõem. Tem um papel similar ao do macrobloco ou sub-macrobloco no H.264/AVC, porém seu tamanho pode variar de 8x8 amostras de luminância até o tamanho da LCU, e contém uma ou mais PUs e TUs. A possibilidade de se escolher arbitrariamente o tamanho da CU permite ao codec ser otimizado para vários tipos de conteúdos, aplicações e dispositivos. Blocos maiores podem ser usados para codificar grandes áreas homogêneas, o que permite um melhor aproveitamento da redundância espacial. Da mesma forma, blocos menores podem ser usados para regiões com muitos detalhes ou para dispositivos de baixa resolução.

A partir da indicação de um flag, a LCU pode ser sub-dividida em 4 CUs, e assim sucessivamente até a CU de maior profundidade. Por exemplo, se o tamanho da LCU for igual a 128, e a profundidade da hierarquia igual a 5, a LCU será subdividida até blocos de 8x8, ou seja, serão 5 tamanhos possíveis para a CU (128×128, 64×64, 32×32, 16×16 e 8×8), conforme pode ser visto na Figura 3.1. Se o tamanho da LCU for igual a 16 , e a profundidade da hierarquia igual a 2, a estrutura de codificação será similar aos blocos e macro-blocos do H.264/AVC.

Uma vez que o processo de divisão é feito, métodos de predição (*skip*, inter ou intra) são especificados para cada CU da última camada (que não pode se dividir mais). Essas CUs são sub-divididas em Unidades de Predição, de acordo com o método de predição escolhido. A sub-divisão das CUs em PUs pode ser vista na Figura 3.2.

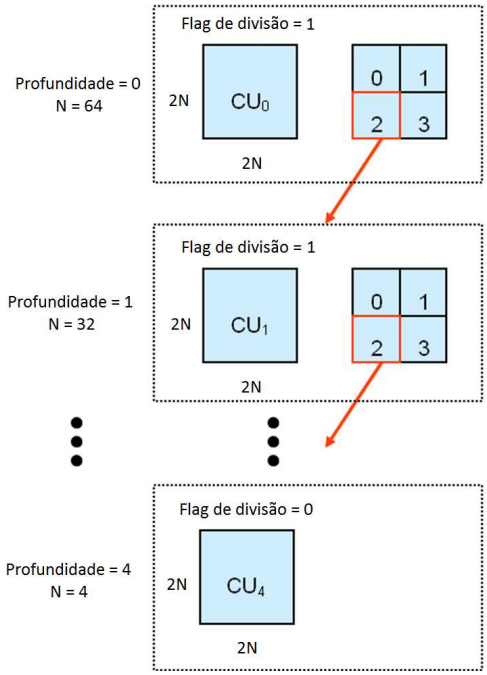


Figura 3.1: Divisão recursiva da CU [36].

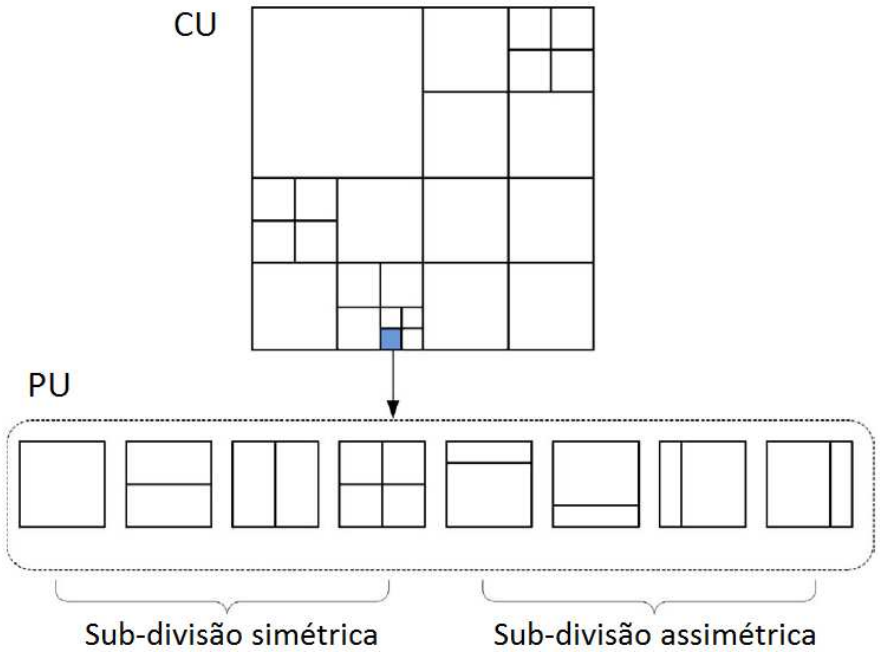


Figura 3.2: Sub-divisão da CU em PUs [42].

3.3.2 Unidade de Predição (PU)

A PU é a unidade básica do processo de predição e carrega todas as informações relacionadas a ele. O formato de particionamento das PUs é definido de acordo com o tipo de predição, e pode ser: 2Nx2N, NxN, 2NxN e Nx2N. Na proposta original , quatro particionamentos assimétricos são também possíveis para o modo inter. A utilização desses formatos não-quadrados e assimétricos busca uma melhor adaptação dos blocos à regiões irregulares de bordas de objetos reais numa imagem. Um exemplo dessa aplicação pode ser visto na Figura 3.3.

Como no H.264/AVC, o tipo de predição pode ser intra, inter ou *skip*. Uma vez que o particionamento e o tipo de predição foram definidos, os vetores de movimento, direção de intra-predição e índices de referência são especificados para cada PU.

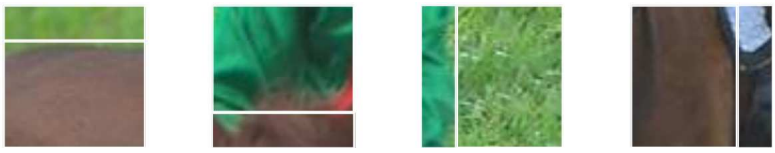


Figura 3.3: Aplicações do particionamento assimétrico das PUs [36].

3.3.3 Unidade de Transformada (TU)

A Unidade de Transformada (TU) é a unidade básica do processo de transformada e quantização. O seu tamanho se ajusta ao modo de particionamento da PU: se a PU for quadrada a TU também será e terá tamanhos de 4×4 até 32×32 amostras de luminância. Quando

PU não for quadrada a TU também não será e poderá assumir um dos tamanhos entre 32×8, 8×32, 16×4 ou 4×16 amostras de luminância. Os formatos não-quadrados de 32×8 e 8×32 também podem ser aplicados na codificação das componentes de crominância. Cada CU poderá conter uma ou mais TUs. Os processos de transformada e quantização serão descritos com mais detalhamento nos itens subsequentes.

As definições propostas acima (CU, PU e TU) permitem uma representação sintática independente do tamanho do bloco. Ao contrário do H.264/AVC, no qual os elementos sintáticos da codificação dos blocos (modo de predição, flag da transformada, flag de bloco codificado, etc.) eram codificados de forma diferente para cada tamanho de bloco, esse novo método permite que esses elementos sejam especificados de uma forma genérica, e que isso seja utilizado nos próximos passos da codificação. Essa propriedade simplifica consideravelmente o esforço de especificação dos elementos assim como o processo de parsing, especialmente quando forem utilizados diversos tamanhos diferentes de blocos (elevada profundidade da hierarquia).

A Figura 3.4 e a Tabela 3.1 fazem um resumo da estrutura de divisão das CUs, PUs e TUs.

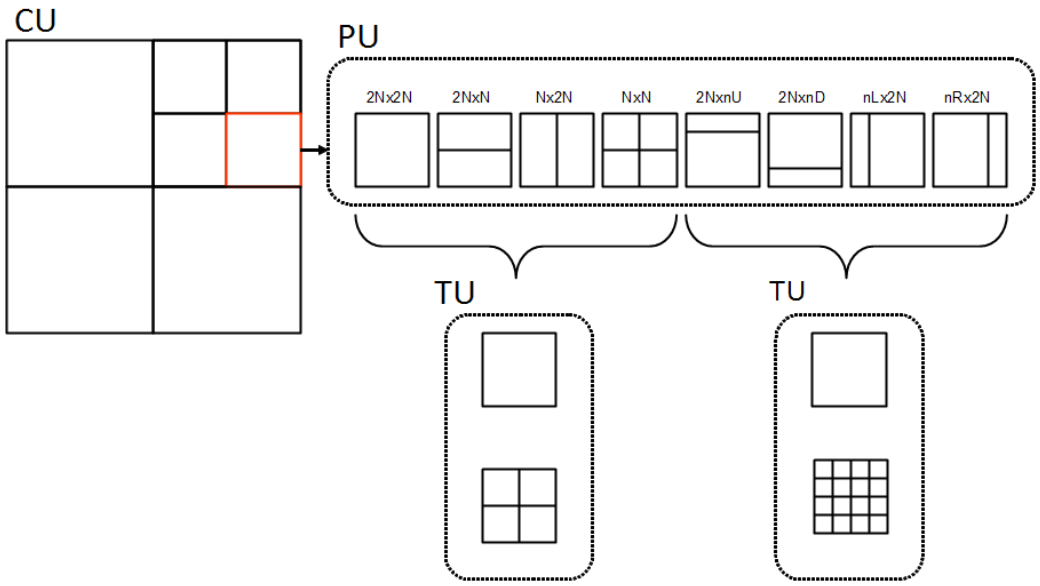


Figura 3.4: Estrutura de divisão das CUs, PUs e TUs (adaptado de [35]).

Tabela 3.1: Resumo da estrutura de particionamento

flag de divisão = 1 (divisão NxN)	flag de divisão = 0 (Cu_d, profundidade máx. = d)			
	Tipo de Predição	Divisão da PU	TU para Main	TU para HE10
Processamento recursivo da CU para Cu_{d+1} de tamanho NxN	INTRA (PUs 2Nx2N e NxN) INTER (todas PUs) SKIP (PUs 2Nx2N)	2Nx2N 2NxN Nx2N NxN	2Nx2N NxN	2Nx2N NxN N/2xN/2

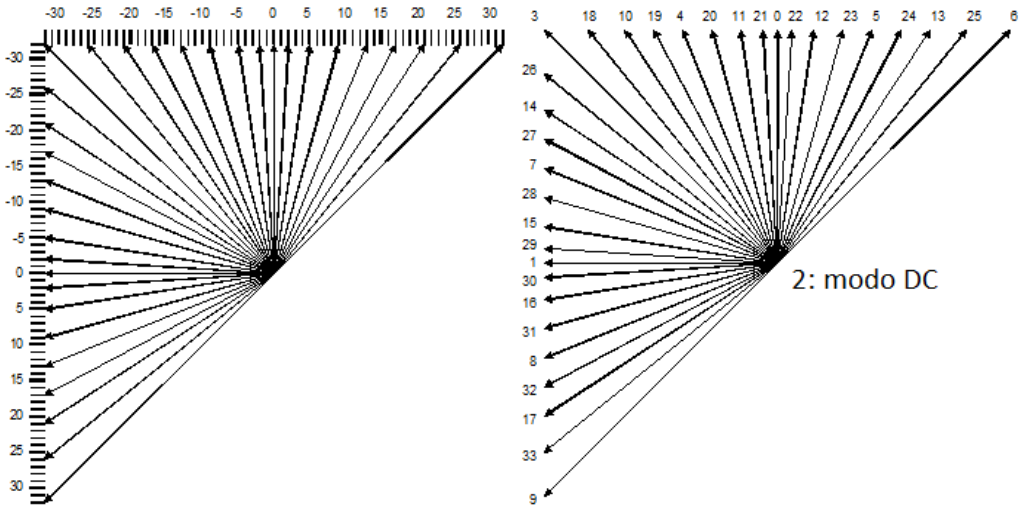
3.4 Predição Intra-frames

O modelo de predição intra-frames do HEVC combina dois métodos propostos em [33] e [35], e os simplifica num modelo unificado de predição [43]. São definidos até 35 modos de predição, sendo 33 direcionais, além dos modos Planar e DC. O número de modos disponíveis depende do tamanho da PU correspondente, conforme a Tabela 3.2.

Os 33 modos direcionais estão associados a índices e ângulos, conforme pode ser visto na Figura 3.5. Para os tamanhos de PU que não possuem todos os modos de predição disponíveis, os primeiros N modos podem ser usados, de acordo com os índices correspondentes. No caso de predição vertical, o ângulo é indicado pelo deslocamento da última linha do bloco da PU, em relação à linha de referência (bloco reconstruído) acima da PU. No caso de predição horizontal, o ângulo é indicado pelo deslocamento da coluna mais a direita da PU, em relação à coluna de referência logo a esquerda da PU. Esse deslocamento é especificado com precisão inteira de pixel. O modo DC utiliza o valor médio das amostras da linha superior e da coluna a esquerda do bloco em processamento. O modo Planar é utilizado para áreas homogêneas, provendo máxima continuidade e suavização nas bordas dos blocos.

Tabela 3.2: Número de modos (direções) disponíveis para cada PU [41].

Tamanho da PU	Número de modos disponíveis
4	17
8	35
16	35
32	35
64	35



(a) Possiveis ângulos (direções) de predição intra-frames (b) Mapeamento entre direções e índices

Figura 3.5: Associação entre ângulos (a) e índices (b) dos modos de predição intra-frames [41].

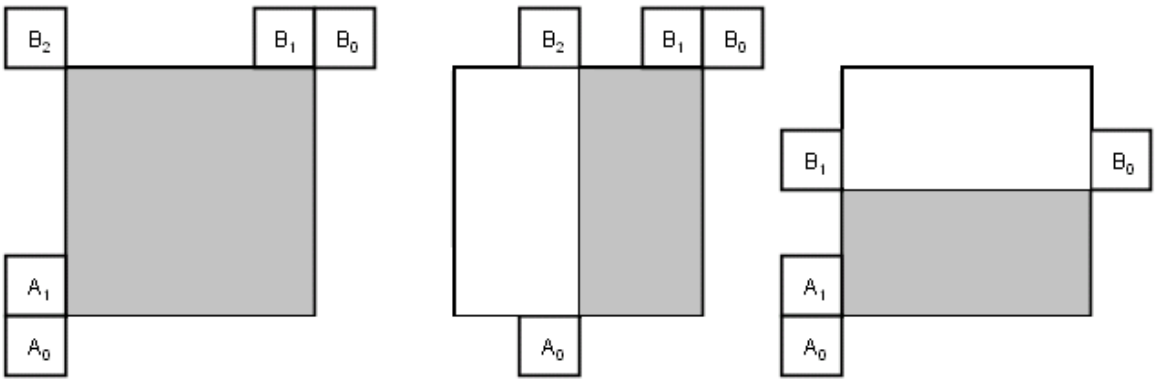
3.5 Predição Inter-frames

Cada PU no processo de inter-predição deve ser associada a um conjunto de parâmetros de movimento: vetores de movimento, índice do frame de referência, índice da direção de predição, etc. Esses parâmetros podem ser implícita ou explícitamente sinalizados. Dois modos são possíveis para esse processo: *skip* e inter.

O modo *skip* pode ser usado tanto no nível de PU quanto de CU. Quando uma CU é codificada no modo *skip*, nenhuma subdivisão em PUs é permitida e os parâmetros de movimento são atribuídos à própria CU. Eles são calculados a partir de uma nova técnica chamada *Motion Merge*, originada de [25, 32], que consiste em achar uma PU vizinha, já codificada, na qual seus parâmetros de movimento podem ser inferidos para a CU em processamento. O codificador pode escolher entre vários candidatos, formados pelas PUs espacialmente ou temporalmente vizinhas, e transmitir o índice correspondente indicando a PU escolhida.

Caso o modo inter seja escolhido, a CU é subdividada em PUs, e os parâmetros de movimento podem ser calculados através do processo normal de estimação de movimento, ou inferidos de outras PUs, também pela técnica de *Motion Merge*, agora aplicada no nível da PU. A Figura 3.6 mostra as posições dos candidatos ao processo de *Motion Merge* espacial para os diversos tamanhos de PU. Nela, os parâmetros de movimento da PU em processamento (sombreada) podem ser inferidos das PUs vizinhas já processadas (A_0, A_1, B_0, B_1 e B_2).

Uma outra técnica chamada AMVP (*Advanced Motion Vector Prediction*), originada de [35, 42], pode ser usada para a predição dos vetores de movimento das PUs em modo inter. Essa técnica, adaptada à nova estrutura de codificação, permite, a seleção do melhor preditor, a partir de um conjunto de três vetores espacialmente adjacentes, a mediana entre eles e um outro vetor temporalmente adjacente.



(a) PU NxN (b) PU Nx2N (c) PU 2NxN

Figura 3.6: Posições dos candidatos ao Motion Merge espacial (adaptado de [41]).

3.5.1 Filtros de Interpolação

Para permitir a utilização de vetores de movimento com precisão de sub-pixel, os blocos de referência precisam ser interpolados até a precisão desejada. No H.264/AVC, para a precisão de

$\frac{1}{2}$-pixel é usado um filtro de interpolação (6-tap Wiener). Para as amostras de $\frac{1}{4}$-pixel é usado uma combinação bi-linear das amostras originais com as geradas no processo anterior. No HEVC esse processo é simplificado. Apenas um procedimento de filtragem é suficiente para prover as amostras em qualquer precisão de sub-pixel (até $\frac{1}{4}$-pixel no momento). Os coeficientes dos filtros são otimizados para minimizar o número de operações aritméticas (adições e deslocamentos).

Para as amostras de luminância é utilizado um filtro de interpolação 8-tap, separável, baseado na DCT, conforme pode ser visto na Tabela 3.3. De forma similar, um filtro 4-tap é utilizado para prover as amostras de 1/8-pixel de crominância, conforme a Tabela 3.4. Para o processo de predição bi-direcional, a saída do filtro de interpolação tem 14 bits por amostra, independente da entrada.

Tabela 3.3: Coeficientes do filtro de interpolação 8-tap para as amostras de luminância [41].

Posição	**Coeficientes**
1/4	-1,4,-10,57,19,-7,3,-1
2/4	-1,4,-11,40,40,-11,4,-1
3/4	-1,3,-7,19,57,-10,4,-1

Tabela 3.4: Coeficientes do filtro de interpolação 8-tap para as amostras de crominância [41].

Posição	**Coeficientes**
1/8	-3,60,8,-1
2/8	-4,54,16,-2
3/8	-5,46,27,-4
4/8	-4,36,36,-4
5/8	-4,27,46,-5
6/8	-2,16,54,-4
7/8	-1,8,60,-3

3.6 Transformada e Quantização

Para o processo de transformada dos resíduos de predição, a CU pode ser dividida em TUs menores. Esse particionamento é chamado de RQT (*Residual Quadtree*) e ocorre de forma de forma análoga ao particionamento das CUs em PUs. São suportados blocos de 4×4 até 32×32 amostras de luminância. A introdução de blocos maiores traz um significante aumento de desempenho em termos de compactação de energia e redução dos erros de quantização para áreas homogêneas, principalmente nos vídeos de alta resolução. A transformada principal é baseada na DCT e o aumento de complexidade é compensado pelo uso do algoritmo rápido de Chen para a DCT [44]. Transformadas baseadas na DST (*Discrete Sine Transform*) [45] também são especificadas para uso em conjunto com a DCT.

Algumas transformadas secundárias ainda estão sob investigação: *Mode Dependent Directional Transform* [46], que pode ser utilizada para codificar blocos 4x4 e 8x8 e é associada ao modo de predição intra-frames; e ROT (*Rotational Transform*) [47], a ser aplicada após a DCT, em blocos de 16x16 ou maiores. A principal idéia por trás da Transformada Rotacional é mudar

a orientação do sistema de bases da transformada, ao invés da rotação direta dos dados de entrada.

Os mesmos métodos de escalonamento, quantização (quantizador escalar com zona morta) e scan do H.264/AVC são utilizados, com novas matrizes de escalamento para os tamanhos de 16×16 e 32×32.

3.7 Codificação de Entropia

Algoritimos similiares aos do H.264/AVC são especificados no HM6, sempre de forma adaptativa ao contexto, ou seja, a codificação do coeficiente atual de um bloco é relacionada ao estado dos coeficientes previamente codificados: uma versão aprimorada do CABAC (*Context Adaptive Binary Arithmetic Coding*), para a codificação dos coeficientes transformados no modo de alta eficiência (HE10); e CAVLC (*Context Adaptive Variable Length Coding*).

3.8 Filtros de Deblocagem

Filtros de "deblocagem" podem ser aplicados nas CUs 8×8 (luminância e crominância), primeiramente nas bordas verticais (filtro horizontal) e depois nas bordas horizontais (filtro vertical). A decisão de aplicar ou não a filtragem, e o tipo de filtro a ser utilizado, depende de um cálculo que determina a "força" das bordas de um bloco em processamento. O tipo do bloco em processamento (CU, PU ou TU) também influencia diretamente no processo de filtragem: por exemplo, bordas de CUs sempre entram no processo de filtragem pois elas também são necessariamente bordas de PUs e TUs. Diferentemente do H.264/AVC, por motivos de complexidade, optou-se por não filtrar blocos 4x4.

3.9 Simulações de Desempenho

Para análise do desempenho objetivo do HEVC, foram geradas curvas de comportamento (Taxa×PSNR), para algumas das sequências sugeridas em [28], que estão listadas na Tabela 3.5. As mesmas estão disponíveis em [48], no formato YUV 4:2:0, com 8 bits/pixel. Optou-se por codificar 2 segundos de cada sequência, conforme sugerido em [49]. Foi utilizada a versão 1.0 do HM, disponível em [50].

Tabela 3.5: Sequências de vídeo utilizadas

Sequência	**Resolução**	**Taxa (fps)**
RaceHorses	416×240	30
BlowingBubbles	416×240	50
BasketballDrill	832×480	50
ParkScene	1080p	24

Conforme já citado anteriormente, o comitê do JCT-VC especifica seis configurações de referência para o HEVC. São duas condições de restrição − que representam dois cenários de aplicação: acesso aleatório (RA), para aplicações de broadcast, por exemplo, com pontos de acesso aleatório durante a sequência (1s), e baixo atraso (LD), para aplicações em tempo real, sem reordenamento de imagens − para três modos de codificação: Main ou LC (*Low Complexity*) e HE10 ou HE (*High Efficiency*), e um modo INTRA (apenas frames I), perfazendo um total de seis configurações (Intra HE, Intra LC, RA HE, RA LC, LD HE e LD LC). As curvas de desempenho do H.264/AVC seguem as configurações, propostas em [28], para as sequências âncoras de referência (*High Profile*). Seus valores foram transcritos de [51].

3.9.1 Resultados e Discussão

As Figuras 3.7, 3.8, 3.9 e 3.10 mostram algumas das curvas de desempenho para ambos os codecs. Como esperado, observa-se que o HEVC supera o H.264/AVC em todas as configurações, inclusive para as de baixa complexidade (LC), atingindo, em alguns casos, mais de 2 dB de ganho.

Essa melhora tem o custo de uma maior complexidade, e isso significa, maiores tempos de codificação. Dado o atual estágio de desenvolvimento do HEVC, considerações sobre o tempo de codificação ainda não são relevantes pois seus algoritmos serão aprimorados, o que implicará em menores tempos de codificação.

É evidente que as configurações de alta eficiência superam as de baixa complexidade, porém isso se refletiu em tempos de codificação até três vezes maiores. Isso também foi percebido ao se comparar as configurações RA com LD, onde a LD mostrou-se mais complexa.

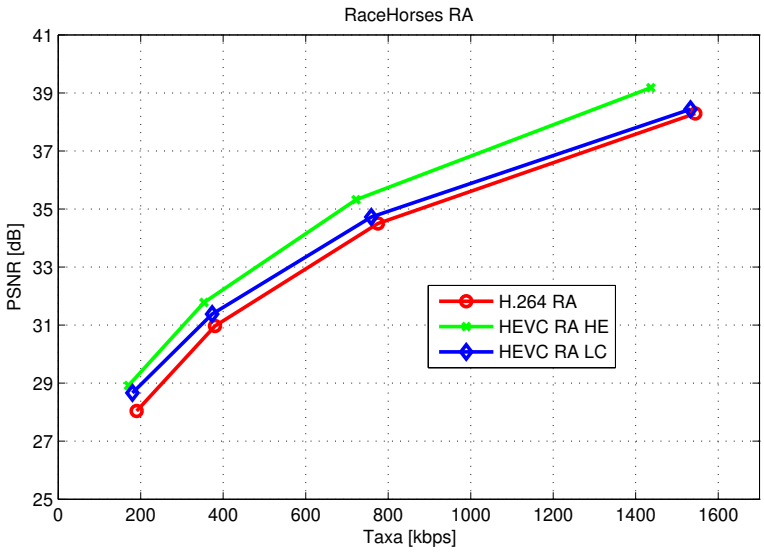


Figura 3.7: Taxa×PSNR – RaceHorses – Acesso aleatório (RA).

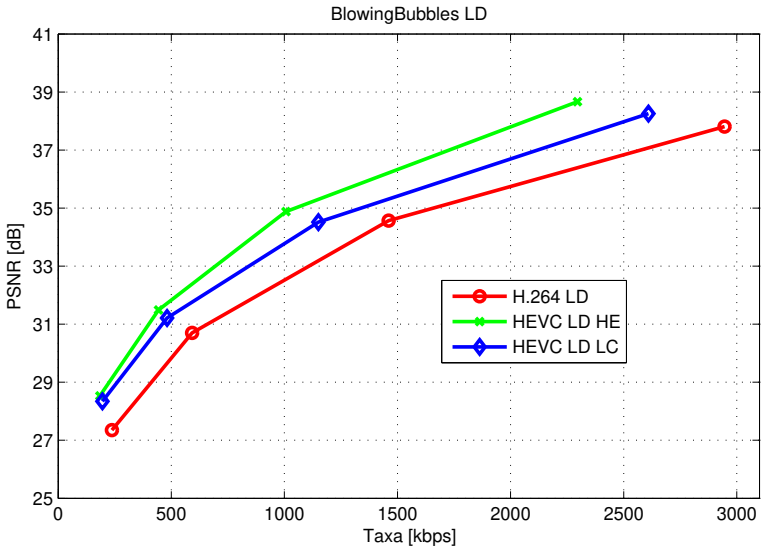


Figura 3.8: Taxa×PSNR – BlowingBubbles – Baixo atraso (LD).

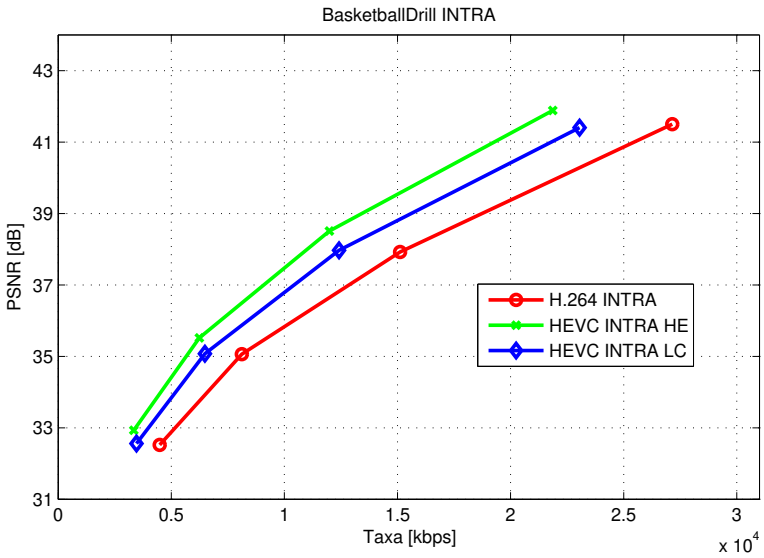


Figura 3.9: Taxa×PSNR – BasketballDrill – Intra (INTRA).

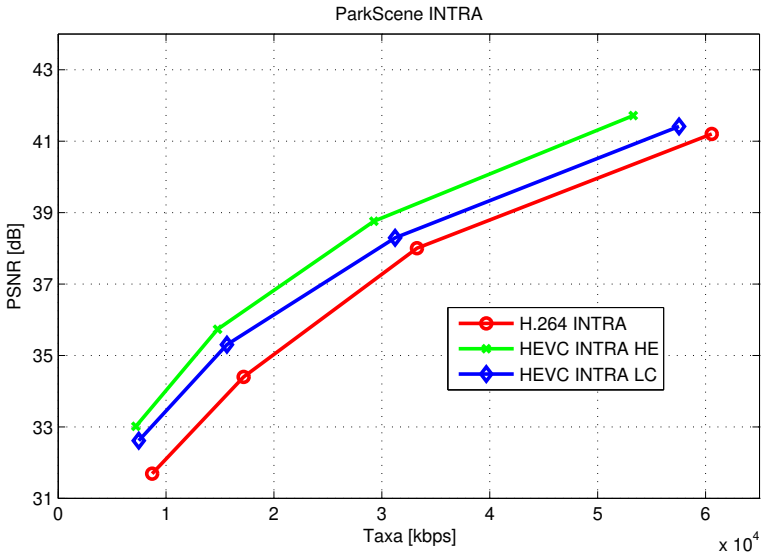


Figura 3.10: Taxa×PSNR – ParkScene – Intra (INTRA).

Capítulo 4

Rápida Predição da Direção do Bloco para Transformadas Direcionais

4.1 Introdução e Motivação

As transformadas espaciais baseadas na DCT são amplamente adotadas para a compressão de vídeo. Recentemente, vários autores têm destacado o fato de que os resíduos de predição, sobre os quais as transformadas são aplicadas, normalmente apresentam estruturas direcionais que não podem ser representadas eficientemente pela DCT convencional.

A maioria dos codecs atuais utiliza a DCT-2D (ou uma derivada dela) como principal transformada espacial. Conforme detalhado no Capítulo 2, a DCT-2D tem funções-base com estruturas verticais e horizontais, e, visando uma redução da complexidade computacional, normalmente é implementada de forma separável, a partir de duas transformadas 1-D ao longo das direções vertical e horizontal. Esse tipo de implementação a torna ineficiente para a representação de estruturas com outras direcionalidades além da vertical ou horizontal, dado que muitos *kernels* unidirecionais precisam ser usados para aproximar o sinal, produzindo assim muitos coeficientes de alta frequência. Além de diminuir a eficiência de codificação, a eliminação desses coeficientes de alta frequência, após o processo de quantização, implica na geração dos chamados "artefatos de Gibbs", causando danos severos à qualidade do vídeo comprimido, principalmente para baixas taxas [2, 52].

Em um codec de vídeo, as transformadas normalmente são aplicadas sobre os resíduos do processo de predição temporal ou espacial. Diversos estudos [53–57] mostram que esse tipo de dado tem características bem diferentes de uma imagem comum (dos pixels originais de um frame de vídeo), resultantes da predominância de bordas nesses blocos residuais [58]. Isso acontece porque os métodos de predição não funcionam bem nas regiões da imagem original que contém limites entre objetos em movimento, bordas ou texturas com muitos detalhes. Mais do que isso, Kamisli e Lim provaram em [54, 59] que os resíduos da predição temporal (interframes) têm como característica predominante estruturas locais direcionais e unidimensionais. Dessa forma, a aplicação de transformadas bidimensionais como a DCT 2-D, com funções base estritamente 2-D, não é adequada para esse tipo de dado.

Nesse cenário, as transformadas direcionais têm ganhado destaque na codificação de vídeo,

apresentando desempenho superior aos algoritmos convencionais e melhorando consideravelmente o "estado-da-arte" dos sistemas de codificação de vídeo [2].

Basicamente, as transformadas direcionais se dividem em três grupos: 1) as que baseiam-se na reorganização dos blocos de imagem de acordo com uma determinada direção, seguido da aplicação de transforamadas convencionais [60, 61]; 2) as que utilizam-se de métodos de *lifting* para alterar as transformadas convencionais, tornando-as direcionais [62–65]; 3) e o terceiro grupo que utiliza a direcionalidade dos dados a serem transformados para a construção da transformada a ser aplicada [66, 67].

Entretanto, a maioria delas leva em consideração apenas os resíduos da predição espacial (intra-frames). Inclusive, na recente documentação publicada pelo JCT-VC [27], referente ao desenvolvimento do HEVC, as transformadas direcionais propostas só são aplicadas nos resíduos intra-frames. De uma forma geral, poucos trabalhos têm abordado as transformadas sobre os resíduos temporais (inter-frames), os quais têm fundamental importância no processo de compressão de vídeo, dado que, apesar de normalmente apresentarem menos energia do que os resíduos intra-frames, o número de frames com predição temporal normalmente é muito maior do que o de frames com predição espacial apenas.

Um *overview* sobre transformadas direcionais pode ser visto em [2].

Apesar das transformadas direcionais apresentarem desempenho muito superior à DCT-2D convencional, para a sua aplicação na compressão de vídeo é necessário avaliar o aumento de complexidade para sua implementação. A maioria das transformadas direcionais propostas recentemente, baseia-se na técnica de otimização por taxa-distorção (RDO) [3] para a escolha da direção que será utilizada. Essa é um técnica de força bruta, na qual, para cada opção de transformada disponível, o bloco passa pelos processos de transformada, quantização, codificação de entropia, quantização inversa e transformada inversa (Figura 4.1). Em seguida o custo taxa-distorção é calculado e a opção de menor custo é escolhida. Esse processo aumenta consideravelmente a complexidade do codificador e pode se tornar impeditivo para aplicações em tempo real. Dado os novos tamanhos possíveis da Unidade de Transformada (4×4 a 32×32) do padrão HEVC, essas considerações são ainda mais relevantes visando a aplicação das transformadas direcionais nesse novo codec.

No intuito de reduzir essa complexidade para a aplicação das transformadas direcionais, buscou-se neste trabalho uma forma de se eliminar o cálculo das RDOs, a partir de uma rápida estimação da direção predominante em cada bloco a ser codificado. Dessa forma, antes de aplicar a transformada direcional, o codificador identifica a direção predominante e aplica apenas a transformada referente àquela direção, ou, caso não haja uma direção predominante, o codificador pode optar por aplicar a DCT-2D convencional. Dessa forma, o método pode ser aplicado em conjunto com qualquer proposta de transformadas direcionais que utilize a técnica RDO como forma de escolha da direção a ser explorada, reduzindo consideravelmente a complexidade de implementação para níveis similares aos da DCT-2D convencional.

A tese original [59], e o artigo publicado em [54] foram a base de partida para este trabalho e serão detalhados na seção seguinte.

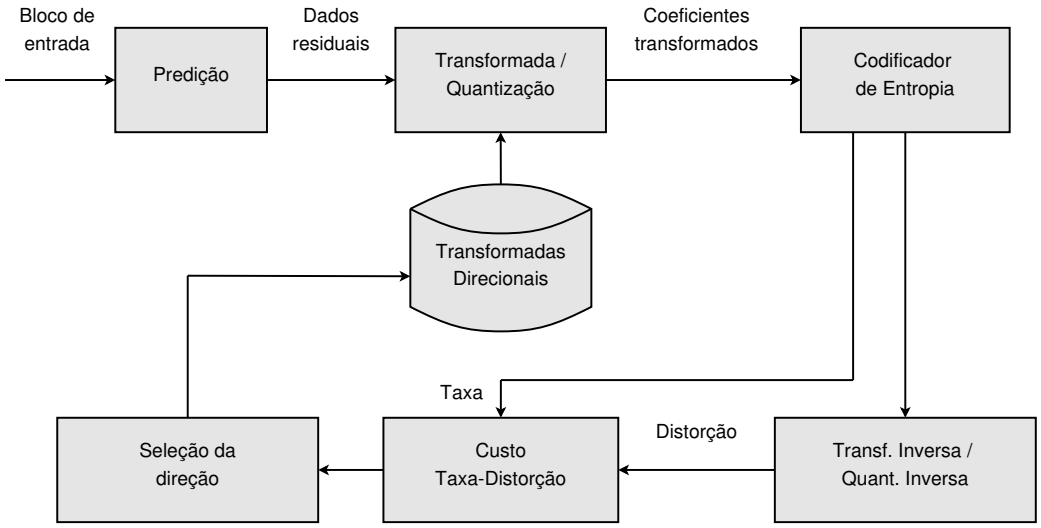


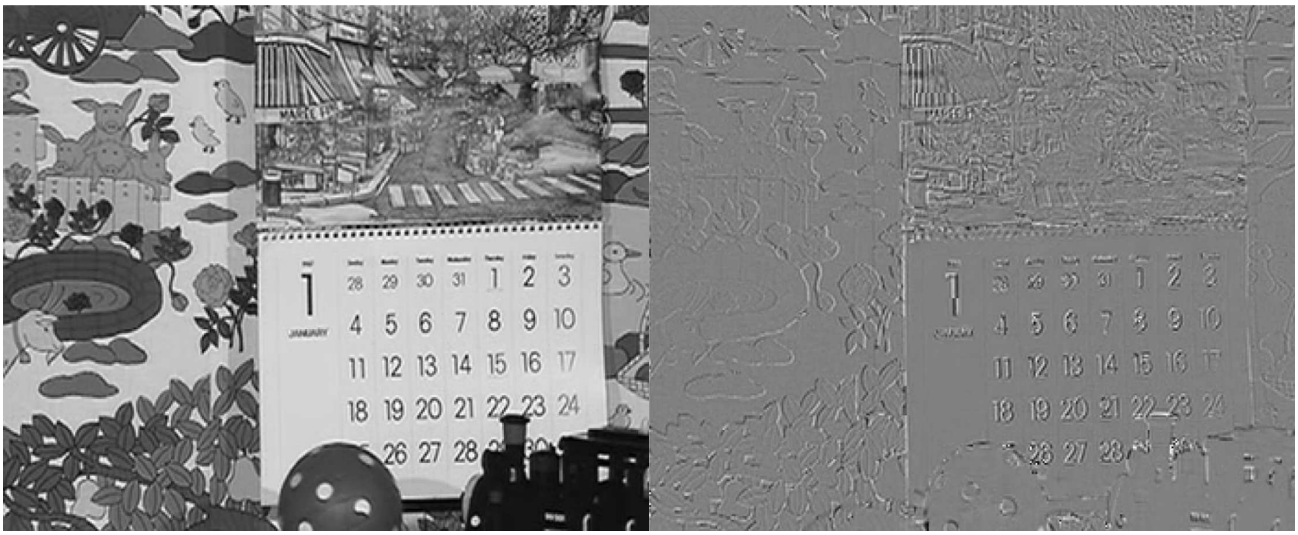
Figura 4.1: Seleção da transformada direcional baseada na técnica RDO.

4.2 Transformadas 1-D para os Resíduos de Compensação de Movimento

A análise realizada por Kamisli mostrou que os resíduos gerados pelo processo de estimação e compensação de movimento têm características espaciais diferentes das imagens de intensidade. Os resíduos não têm regiões lisas ou regulares como as imagens originais pois os métodos de predição funcionam muito bem nessas regiões. A maior parte da energia fica então concentrada em regiões de difícil predição (limites de objetos em movimentos ou bordas), ou seja, os pixels de maior intensidade se concentram ao longo dessas bordas, formando estruturas unidimensionais com alguma direção predominante. Sendo assim, para esse tipo de região, as transformadas convencionais, derivadas da DCT-2D, podem não ser a melhor escolha para esse tipo de resíduo. Um exemplo de uma imagem residual pode ser visto na Figura 4.2.

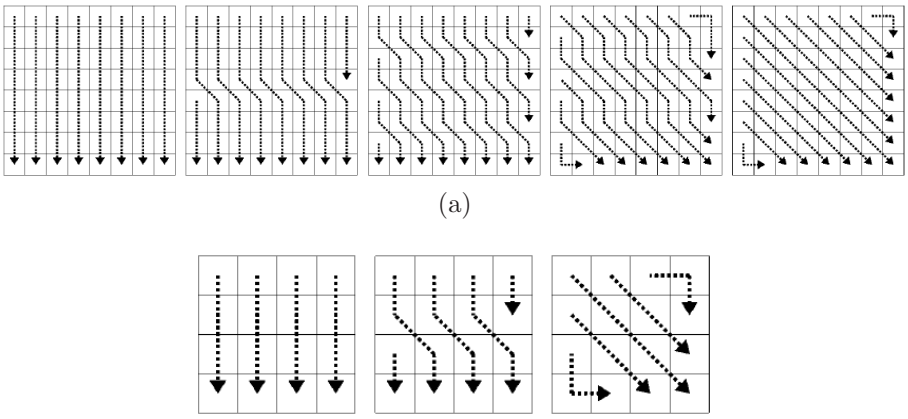
O autor então propõe o uso das transformadas 1-D direcionais mostradas na Figura 4.3. Foram escolhidas 16 direções para blocos 8x8 (a), e 8 para os blocos 4x4 (b), que juntas cobrem 180° (apenas algumas direções são mostradas; as outras são simétricas). Os pixels de um bloco são reorganizados em vetores, de acordo com uma determinada direção (sentido das setas na figura), e, em seguida, é aplicada uma transformada DCT-1D em cada um desses vetores. Após as transformadas, os coeficientes são reorganizados de acordo com o posicionamento original, formando novamente um bloco, que é então escaneado de acordo com a direção da transformada que foi aplicada. A Figura 4.4 mostra os padrões direcionais de scan para blocos de 8x8 (a) e 4x4 (b). Esses padrões foram criados de forma que os coeficientes com maior probabilidade de serem quantizados para zero fiquem posicionados no final do vetor resultante.

As Figuras 4.5 e 4.6 mostram blocos residuais sintéticos sobre os quais foram aplicadas a DCT-2D convencional e a transformada 1-D proposta, alinhada com a direção predominante da estrutura presente no bloco. Percebe-se claramente que a DCT-2D precisa de mais coeficientes não nulos para a representação do bloco. Nesses casos extremos, a transformada proposta precisa de apenas um coeficiente para representar todo o bloco. No caso da Figura 5, mesmo com o



(a) Imagem original (b) Resíduo

Figura 4.2: Imagem original e resíduo da compensação de movimentos [54].



(a)

(b)

Figura 4.3: Transformadas 1-D direcionais em blocos 8x8 (a) e 4x4 (b) [54].

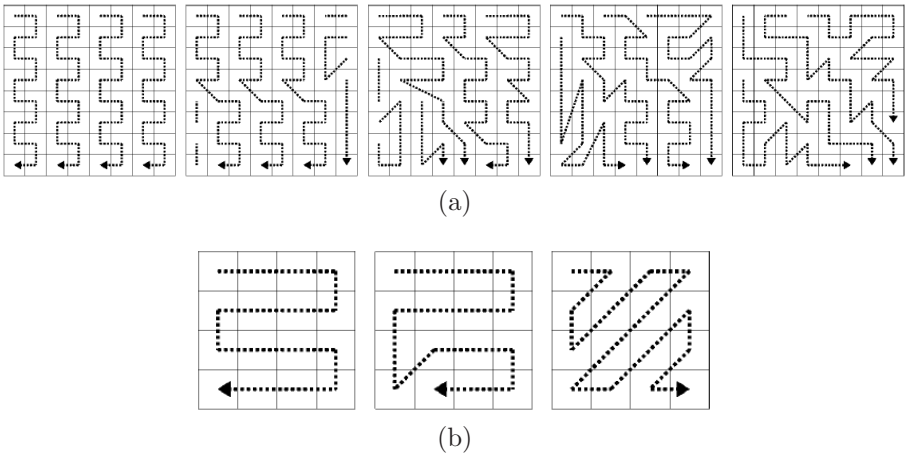


Figura 4.4: Padrões de scan direcional em blocos 8x8 (a) e 4x4 (b) [54].

(a) Bloco residual (b) DCT-2D (c) Transformada 1-D proposta

Figura 4.5: Comparação da DCT-2D com a transformada 1-D proposta, aplicadas em um bloco sintético com estrutura 1-D diagonal [54].

alinhamento da estrutura sendo consistente com a direção das bases da DCT-2D, a transformada 1-D tem um desempenho melhor. Note que como a DCT-2D é separável, e implementada a partir de duas DCTs 1-D, uma horizontal e outra vertical, a transformada proposta, para as direções horizontal e vertical, corresponde a apenas uma das duas etapas da transformada convencional.

O esquema proposto foi integrado ao H.264/AVC (JM 10.2) para análise do desempenho de codificação, da qualidade subjetiva do vídeo codificado e da complexidade computacional. Para cada macrobloco, todas as transformadas (4x4 e 8x8, 1-D, e 2-D convencional) eram testadas e a que oferecia o menor custo de taxa de dados por qualidade de imagem era escolhida. Os resultados, apresentados através de gráficos "Taxa×PSNR" e da métrica de Bjontegaard-delta [24], mostraram ganhos de codificação para todas as sequências testadas, em todas as taxas, e reduções na taxa de dados resultante, de até 11,4% na média, para uma mesma qualidade de imagem. Os ganhos foram mais expressivos para altas taxas de codificação. Isso acontece porque, nessas taxas, uma maior porcentagem da taxa total dos dados resultantes é atribuída aos coeficientes da transformada, o que tende a "realçar" o ganho de compressão pelo uso das transformadas 1-D. Outra explicação é que o *overhead* gerado para sinalizar ao decodificador as transformadas escolhidas é menor para altas taxas. Os ganhos foram dependentes também do

(a) Bloco residual (b) DCT-2D (c) Transformada 1-D proposta

Figura 4.6: Comparação da DCT-2D com a transformada 1-D proposta, aplicadas em um bloco sintético com estrutura 1-D vertical [54].

tamanho do bloco utilizado, sendo mais expressivos para os blocos maiores (8x8) pois quanto maior o bloco, mais evidente se torna o ganho de uma transformada sobre a outra. Houve também um ganho de qualidade subjetiva nas imagens. O autor também fez um comparação com as transformadas 2-D direcionais propostas em [68], as quais tiveram um desempenho inferior ao das transformadas 1-D propostas.

4.2.1 Análise de Complexidade das Transformadas 1-D

Para tentar estimar o aumento no tempo de total de codificação ao utilizar a transformada proposta, o autor fez uma análise teórica da complexidade do método. Dado que a escolha da transformada a ser utilizada é feita através da técnica de RDO, na qual todas as opções de codificação possíveis são testadas e a que apresentou o menor custo é escolhida, o aumento no tempo de codificação é atribuído a esse processo de testes e pode ser estimado através da multiplicação entre o tempo necessário para se testar uma das opções pelo número de opções disponíveis. Para cada opção de transformada disponível, o bloco passa pelos processos de transformada (T), quantização (Q), codificação de entropia (E), quantização inversa (Q) e transformada inversa (T). O tempo total desse processo (TQEQT) é uma fração do tempo total de codificação e aumenta linearmente com o número de opções possíveis. Outra consideração é que a transformada convencional é 2-D enquanto a proposta é 1-D e, portanto, menos complexa. Dessa forma, o fator de aumento no tempo total de codificação pode ser modelado por:

$$F_a = \alpha N_{tr} TQEQT + (100 - TQEQT) \tag{4.1}$$

onde α é um fator de escala constante e menor que 1, relativo à complexidade de uma transformada, N_{tr} é o número total de transformadas possíveis e TQEQT é a porcentagem do tempo total de codificação, utilizado no cálculo das TQEQTs.

Nos experimentos realizados apenas com blocos de 8x8, 30% do tempo de codificação foi utilizado no cálculo das TQEQTs com a transformada convencional. Logo o fator de aumento do tempo total de codificação, para 16 transformadas adicionais e considerando α igual a 1, foi de 5,8. Caso as transformadas 1-D fossem implementadas com aritmética inteira, a porcentagem

de tempo estimada para o cálculo das TQEQTs é de 6% para blocos 8x8, e 8% para blocos 4x4. Sendo assim, o fator de aumento do tempo total de codificação seria de 1,9 e 1,6 respectivamente (para blocos 4x4, são apenas 8 transformadas adicionais). O tempo de decodificação não aumenta já que o decodificador executa apenas a transformada que foi sinalizada pelo codificador. Não foram apresentados os valores reais do tempo de codificação com o método proposto.

4.3 Estimando a Direção Predominante no Bloco

O método utilizado foi originalmente proposto em [69], e consiste em achar a direção predominante do bloco a partir das direções existentes nas amostras dentro desse bloco. Para isso, calcula-se, para cada amostra $p(x, y)$, a magnitude $M(x, y)$ e orientação $\theta(x, y)$ do gradiente, a partir das diferenças de pixels $Gx_{x,y}$ e $Gy_{x,y}$:

$$Gx_{x,y} = p(x+1, y) - p(x-1, y) \tag{4.2}$$

$$Gy_{x,y} = p(x, y+1) - p(x, y-1) \tag{4.3}$$

$$M(x, y) = \sqrt{Gx_{x,y}^2 + Gy_{x,y}^2} \tag{4.4}$$

$$\theta(x, y) = \arctan\left(\frac{Gy_{x,y}}{Gx_{x,y}}\right) \tag{4.5}$$

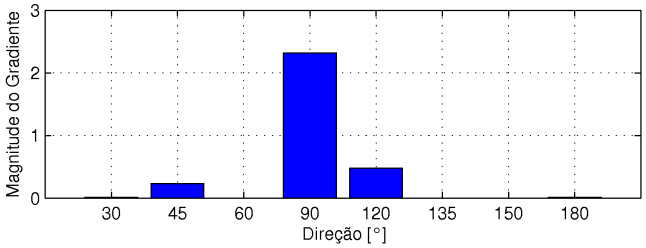
Para otimizar a implementação, a raiz dos quadrados na Equação 4.4 pode ser substituída por $|Gy_{x,y}| + |Gy_{x,y}|$, e a função arctan na Equação 4.5 pode ser substituída por $Gy_{x,y}/Gx_{x,y}$.

Monta-se então um histograma das direções, no qual cada amostra adicionada é ponderada pelo seu gradiente de magnitude. Os picos no histograma de direções correspondem às direções dominantes dos gradientes locais. Um simples limiar pode ser utilizado para escolher apenas uma direção. A discretização do histograma deve ser escolhida de acordo com o conjunto de tranformadas direcionais disponível. Cada direção possível de transformada deve ser associada com um bin do histograma.

A Figuras 4.7, 4.8, 4.9 e 4.10 mostram blocos residuais (a) e o histograma calculado (b) para cada bloco. O histograma de direções tem 8 bins cobrindo 180°.

Caso o algoritmo tome uma decisão errada sobre a direção predominante no bloco, a transformada direcional utilizada não será consistente com a estrutura presente no resíduo. Isso implicará na geração de coeficientes desnecessários, prejudicando o desempenho de codificação e degradando a qualidade da imagem recuperada após a decodificação. Dessa forma, os limiares devem ser cuidadosamente escolhidos de forma que a escolha da direção predominante seja feita com alto grau de certeza. Caso não haja forte predominância de uma direção em relação às

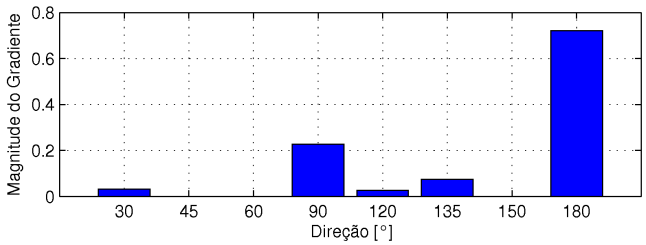
(a) Bloco residual



(b) Histograma de direções

Figura 4.7: Bloco residual com estrutura a 90° (a) e histograma de direções associado (b).

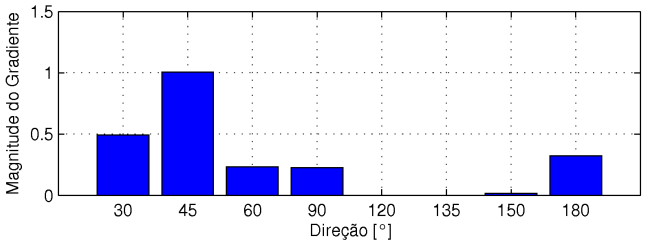
(a) Bloco residual



(b) Histograma de direções

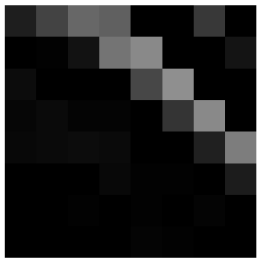
Figura 4.8: Bloco residual com estrutura a 180° (a) e histograma de direções associado (b).

(a) Bloco residual

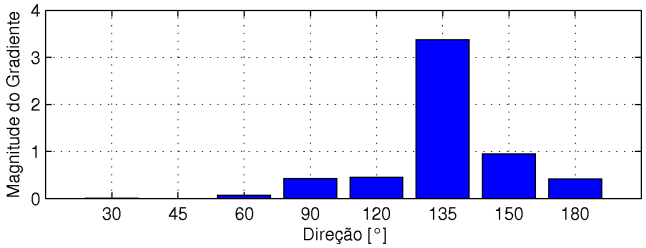


(b) Histograma de direções

Figura 4.9: Bloco residual com estrutura a 45° (a) e histograma de direções associado (b).



(a) Bloco residual



(b) Histograma de direções

Figura 4.10: Bloco residual com estrutura a 135° (a) e histograma de direções associado (b).

outras, o algoritmo opta então por não indicar nenhuma direção e a DCT-2D convencional deve ser utilizada.

Um algoritmo similar, baseado no cálculo do histograma de direções, foi utilizado em [70] para acelerar a escolha dos modos de predição espacial (intra-frames) no H.264/AVC, e recentemente no HEVC [71].

4.4 Integração com Transformadas Direcionais

Após a determinação de uma direção predominante, a transformada 1-D direcional, relativa àquela direção, pode ser aplicada. Os coeficientes são quantizados, scaneados de acordo com a direção determinada, e o bloco segue então para o codificador de entropia. Caso não seja determinada uma direção predominante, o codificador aplica a DCT-2D convencional. A Figura 4.11 mostra o diagrama em blocos da solução conjunta.

O algoritmo foi integrado com as transformadas direcionais propostas em [54, 59]. A integração foi feita de forma que, sempre que o algoritmo de predição indicasse alguma direção predominante, o desempenho daquela transformada direcional fosse melhor que todas as outras, incluindo a DCT-2D convencional. Para fins de análise, 8 das 16 transformadas direcionais propostas foram utilizadas: 30°, 45°, 60°, 90°, 120°, 135°, 150° and 180°. Direções detectadas no terceiro ou quarto quadrantes (direções negativas) foram mapeadas para direção positiva correspendente (primeiro e segundo quadrantes). Os valores reais calculados devem ser aproximados para os valores inteiros dos bins disponíveis de acordo com algum critério de aproximação. Resultados experimentais mostraram que, para essas 8 direções, 5° é um bom critério. Por exemplo, direções entre 25° e 35° são contabilizadas no bin de 30°. Valores fora do range de cada bin não são utilizados pelo algoritmo. Como já foi dito anteriormente, caso não haja forte predominância de uma direção em relação as outras, o algoritmo decide não indicar nenhuma direção e a DCT-2D convencional deve ser usada.

Como sugerido em [70, 71], o algoritmo pode ser usado para selecionar um número menor de transformadas direcionais possíveis como candidatas no processo de seleção pela técnica de RDO.

4.5 Resultados Experimentais e Discussão

4.5.1 Algoritmo de Predição da Direção do Bloco

Para validar os conceitos apresentados, o algoritmo de predição da direção predominante nos blocos foi implementado em MATLAB®, e testado com frames das sequências sugeridas pelo JCT-VC para os testes com o HEVC [28]. Foram gerados frames residuais e esses foram processados bloco a bloco. Para esses exemplos foram utilizadas apenas quatro direções: 45°, 90°, 135° e 180°. As Figuras 4.12 e 4.13 mostram os resultados obtidos com duas das sequências testadas.

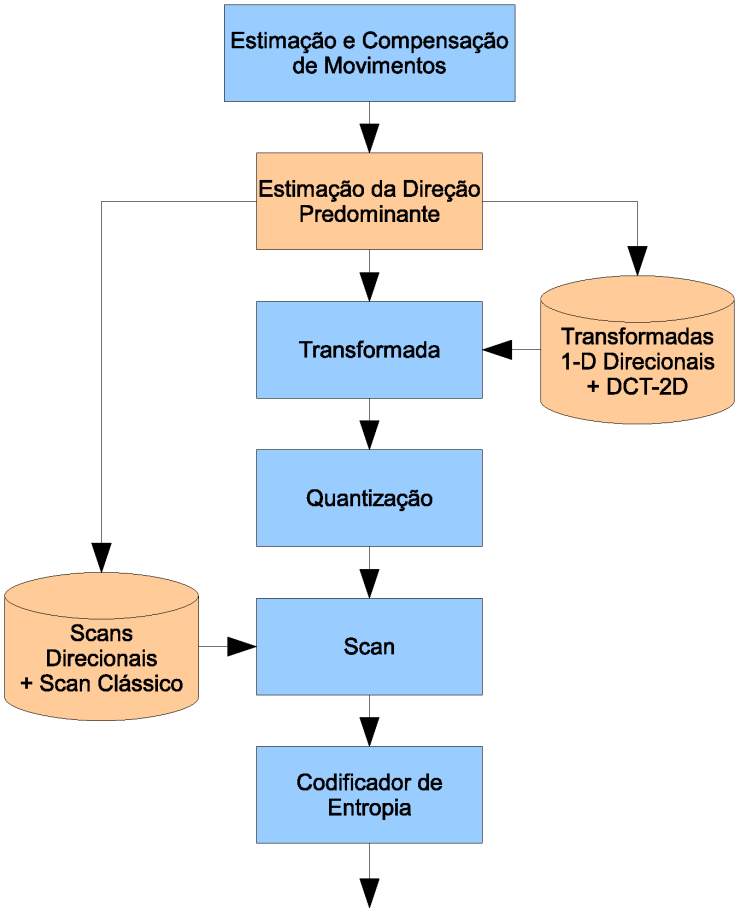


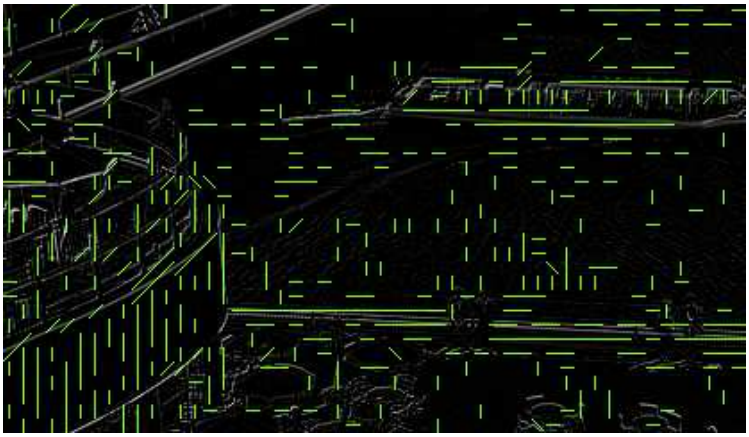
Figura 4.11: Diagrama em blocos da solução conjunta.

4.5.2 Solução conjunta

As Figuras 4.14 e 4.17 mostram alguns dos resultados obtidos com a solução integrada. Primeiramente o algoritmo de predição detecta uma direção predominante dentro do bloco e, em seguida, a transformada 1-D relativa àquela direção é aplicada, seguida do scan também direcional. Os resultados foram analisados através de gráficos que mostram a porcentagem de energia contida nos coeficientes, na ordem de leitura do scan relativo àquela direção. Quanto maior for a quantidade de energia contida nos primeiros coeficientes (em ordem de leitura), melhor foi o desempenho da transformada.

Pode-se observar que, quando a transformada 1-D é utilizada, quase 90% da energia residual fica concentrada nos dez primeiros coeficientes lidos, enquanto que ao utilizar a DCT 2-D a energia fica espalhada por muitos coeficientes, o que vai piorar consideravelmente o desempenho da codificação. Por exemplo, na Figura 4.14 (e), o segundo coeficiente da transformada 1-D direcional já concentra mais de 80% de toda energia do bloco, enquanto que, com a DCT-2D convencional, aproximadamente 28 coeficientes são necessários para que se tenha a mesma quantidade de energia.

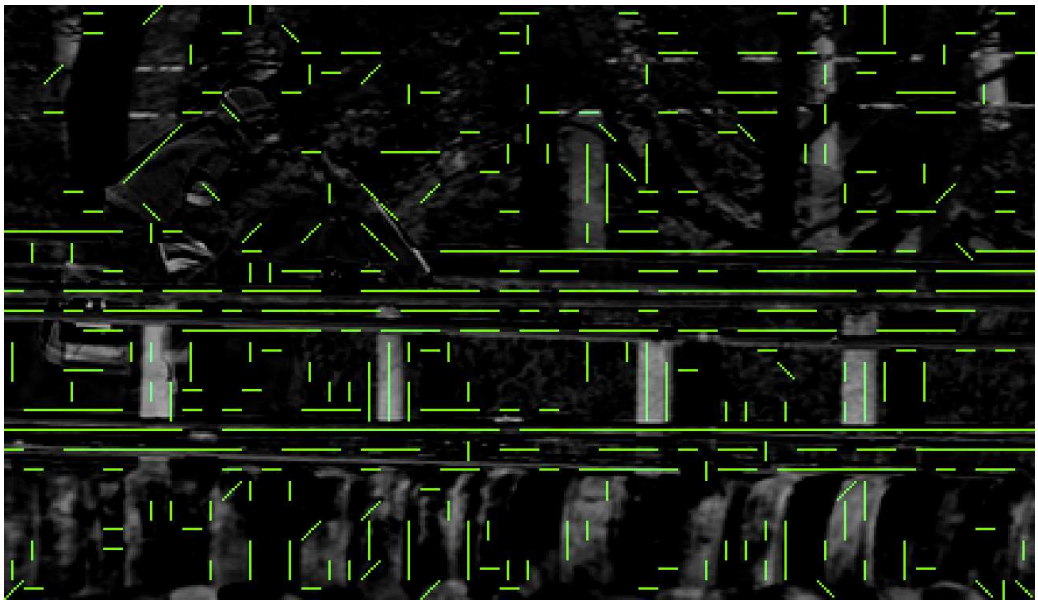
(a) Frame original



(b) Direções detectadas

Figura 4.12: Frame original (a) e overplot (b) das direções detectadas (linhas verdes).

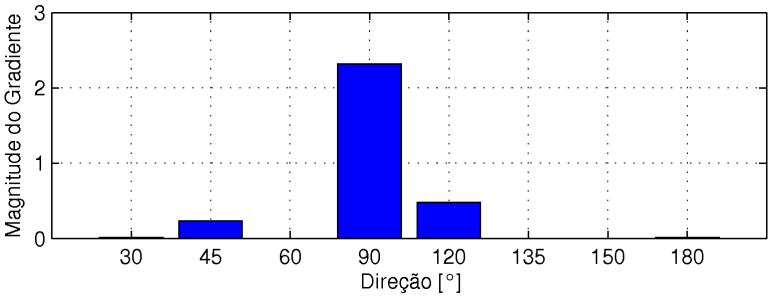
(a) Frame original



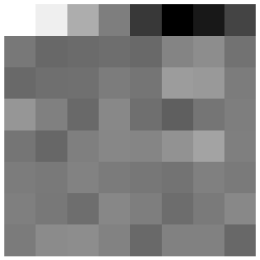
(b) Direções detectadas

Figura 4.13: Frame original (a) e overplot (b) das direções detectadas (linhas verdes).

(a) Bloco residual (8x8)

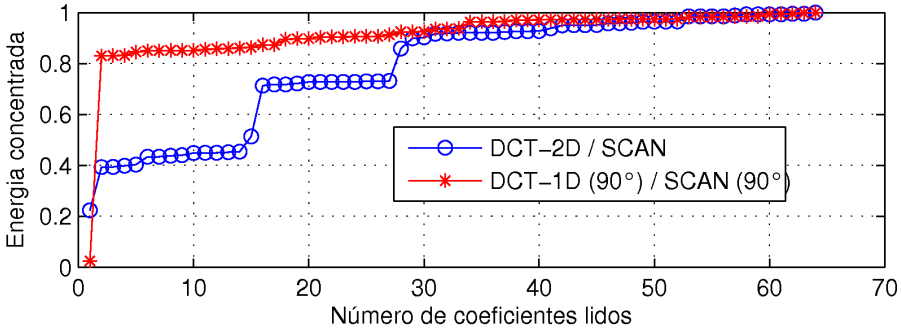


(b) Histograma associado



(c) Coeficientes da DCT-2D

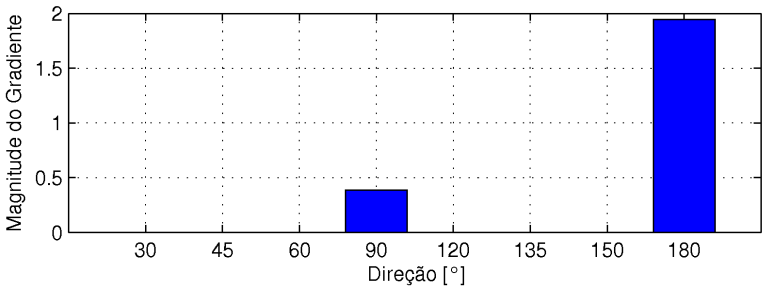
(d) Coeficientes da DCT-1D (90°)



(e) Energia concentrada dos coeficientes transformados após o scan direcional

Figura 4.14: Resultados da solução integrada (90°).

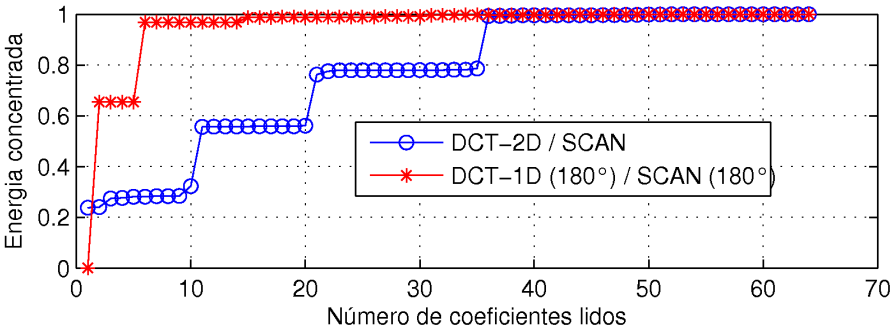
(a) Bloco residual (8x8)



(b) Histograma associado

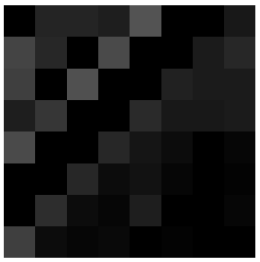
(c) Coeficientes da DCT-2D

(d) Coeficientes da DCT-1D (180°)

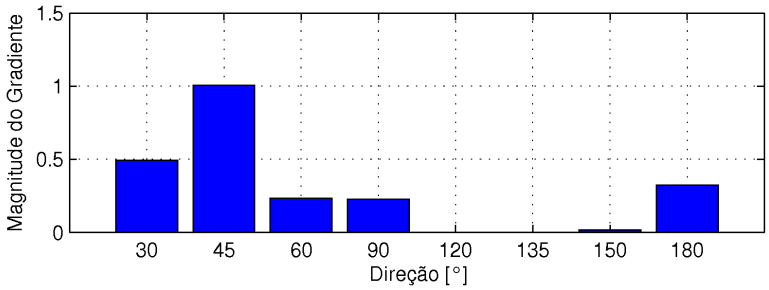


(e) Energia concentrada dos coeficientes transformados após o scan direcional

Figura 4.15: Resultados da solução integrada (180°).

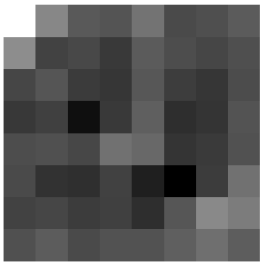


(a) Bloco residual (8x8)

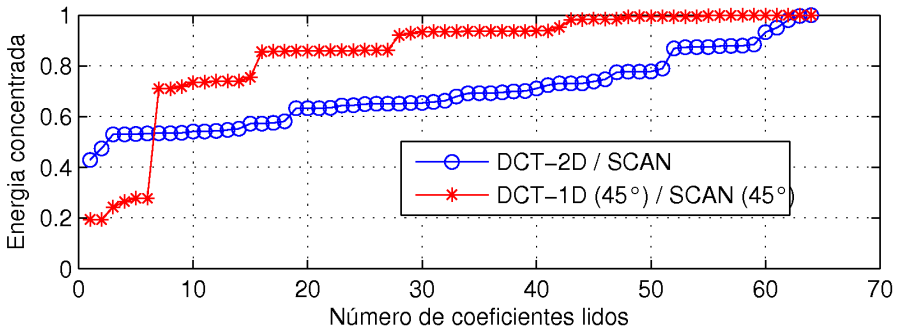


(b) Histograma associado

(c) Coeficientes da DCT-2D

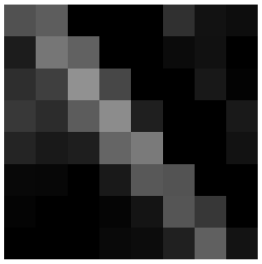


(d) Coeficientes da DCT-1D (45°)

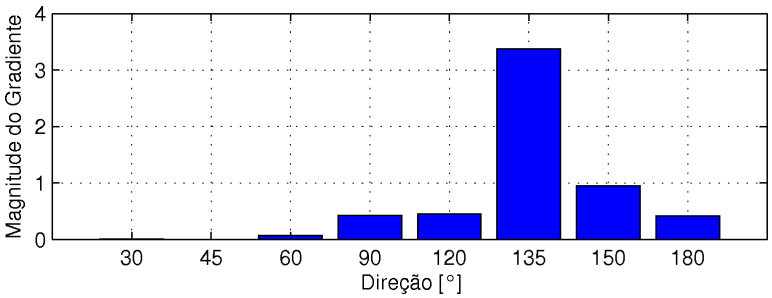


(e) Energia concentrada dos coeficientes transformados após o scan direcional

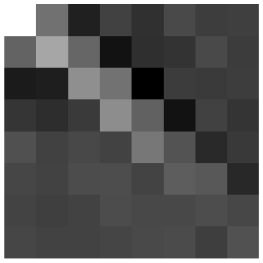
Figura 4.16: Resultados da solução integrada (45°).



(a) Bloco residual (8x8)

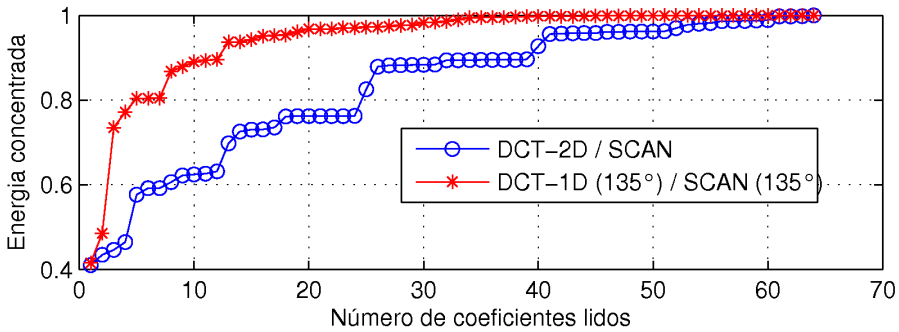


(b) Histograma associado



(c) Coeficientes da DCT-2D

(d) Coeficientes da DCT-1D (90°)



(e) Energia concentrada dos coeficientes transformados após o scan direcional

Figura 4.17: Resultados da solução integrada (135°).

4.5.3 Análise de Complexidade

Conforme citado anteriormente, nos experimentos de Kamisli com o H.264/AVC por exemplo, quando somente a transformada convencional era utilizada, os cálculos de RDO consumiam cerca de 6% (blocos 8x8) e 8% (blocos 4x4) do tempo total de codificação. Sendo assim, o fator de aumento estimado no tempo total de codificação, ignorando a complexidade da transformada em si, seria de 1.9 (16 transformadas adicionais) e 1.6 (8 transformadas adicionais).

Ao se utilizar o algoritmo de predição da direção do bloco, apesar das diversas possibilidades de transformada direcional, como o algoritmo de estimação previamente escolhe a transformada que será aplicada (uma das direcionais ou a DCT-2D), para o encoder é como se existisse apenas uma possibilidade de transformada, o que elimina a necessidade dos testes de seleção com a técnica RDO, e reduz a complexidade para níveis similares ao caso convencional, em que apenas a DCT-2D é aplicada. Dessa forma, em relação as transformadas propostas por Kamisli, em que eram testadas 16 direções para os blocos 8x8, e 8 direções para os blocos 4x4, o número de cálculos das RDOs cairia de 17 para 1, e de 9 para 1, respectivamente. E a redução estimada no tempo total de codificação seria de 47,4% para blocos 8x8 e 37,5% para blocos 4x4.

Capítulo 5

Conclusões e Perspectivas

O papel do estágio de transformada em um codec de vídeo é representar a imagem (normalmente os resíduos de predição) em outro domínio onde as redundâncias espaciais ainda presentes possam ser exploradas com maior eficiência, sem degradação perceptível a visão humana. A maioria dos padrões existentes utiliza a DCT, ou uma variação dela, como principal ferramenta dessa etapa.

Estudos recentes mostraram que os resíduos de predição, sobre os quais as transformadas são aplicadas, apresentam características direcionais que não podem ser representadas eficientemente pela DCT-2D convencional. Mais do que isso, o estudo realizado por Kamisli em [54, 59], mostrou que os resíduos da predição temporal (inter-frames) têm como característica predominante estruturas locais direcionais e unidimensionais. Kamisli conclui que, para esse tipo de dado, a aplicação de transformadas bidimensionais como a DCT 2-D, com funções base estritamente 2-D, não é a solução mais adequada.

Muitas transformadas direcionais têm sido propostas, no intuito de melhor explorar essa direcionalidades, reduzindo o número de coeficientes necessários para a representação da imagem, e, assim, aumentando o poder de compressão dos codecs.

Muitas dessas propostas utilizam a técnica de RDO para a seleção da melhor transformada a ser aplicada. Esse método é extremamante custoso pois todas as possíveis direções de transformada são testadas e a de melhor desempenho é escolhida. Esse processo de seleção aumenta consideravelmente a complexidade computacional do codificador, implicando também em maiores tempos de codificação e podendo até inviabilizar a solução para aplicações em tempo real.

No intuito de melhorar esse processo de escolha, reduzindo a complexidade para a aplicação das transformadas direcionais, a principal contribuição deste trabalho foi a proposta de um algoritmo que, de forma simples e rápida, possa estimar a direção predominante em um bloco de resíduo, indicando previamente ao codificador qual a direção da transformada que deve ser aplicada. O método pode ser aplicado em conjunto com qualquer proposta de transformadas direcionais que utilize a técnica RDO como forma de escolha da direção a ser explorada. Evita-se assim a necessidade de que todas as possibilidades precisem ser testadas, reduzindo a complexidade a níveis similares ao caso em que apenas uma transformada é aplicada.

O algoritmo baseia-se no cálculo de um histograma com as direções existentes em um bloco residual, e, a partir de limiares previamente escolhidos, toma uma decisão sobre a existência,

ou não, de uma direção predominante no bloco, indicando qual seria essa direção.

O mesmo foi implementado em MATLAB®e testado com algumas das sequências sugeridas pelo JCT-VC. Como forma de validar visualmente o funcionamento do algoritmo, foram geradas imagens com a sobreposição das direções detectadas nos diversos blocos de um frame, comprovando o bom funcionamento da proposta.

Em seguida o algoritmo foi integrado à proposta de transformadas direcionais de Kamisli [54, 59]. As transformadas também foram implementadas em MATLAB®e a seleção da direção a ser explorada foi feita com o algoritmo proposto. Os resultados mostraram que a aplicação da transformada/scan com a direção indicada pelo algoritmo, consegue concentrar a energia em muito menos coeficientes do que quando utilizado a DCT-2D convencional. Isso indica, mais uma vez, que o algortimo identificou corretamente a direção predominante no bloco residual.

Ao se utilizar o algoritmo de predição da direção do bloco, apesar das diversas possibilidades de transformada direcional, como o algoritmo de estimação previamente escolhe a transformada que será aplicada (uma das direcionais ou a DCT-2D), para o encoder é como se existisse apenas uma possibilidade de transformada, o que elimina a necessidade dos testes de seleção com a técnica RDO, e reduz a complexidade para níveis similares ao caso convencional, em que apenas a DCT-2D é aplicada. Dessa forma, em relação as transformadas propostas por Kamisli, em que eram testadas 16 direções para os blocos 8x8, e 8 direções para os blocos 4x4, o número de cálculos das RDOs cairia de 17 para 1, e de 9 para 1, respectivamente, e a redução estimada no tempo total de codificação é de 47,4% para blocos 8x8 e 37,5% para blocos 4x4.

Como contribuição secundária, no Capítulo 3, foram apresentadas as características do novo codec HEVC, que está sendo desenvolvido como futuro sucessor do H.264/AVC. Algumas de suas ferramentas foram descritas, assim como suas evoluções técnicas em relação ao seu antecessor. Também foram apresentadas curvas de desempenho para avaliação objetiva e comparação. O HEVC apresentou um desempenho superior ao H.264/AVC, para todas as configurações, ao custo de uma maior complexidade e maiores tempos de codificação. Acredita-se que, neste momento, considerações sobre o tempo de codificação ainda não são relevantes pois o HEVC está em desenvolvimento e seus algoritmos ainda serão aprimorados, implicando, futuramente, em menores tempos de codificação.

Sendo assim, as principais contribuições deste trabalho podem ser resumidas como:

- Apresentação do novo codec HEVC. Descrição de suas características, principais ferramentas e evoluções técnicas em relação ao H.264/AVC. Também foram apresentadas curvas de desempenho para avaliação objetiva e comparação.

- Proposta de um algoritmo de estimação da direção predominante em um bloco residual, permitindo a rápida determinação da melhor transformada direcional a ser aplicada. O método pode ser aplicado em conjunto com qualquer proposta de transformadas direcionais que utilize a técnica RDO como forma de escolha da direção a ser explorada. Evita-se assim a necessidade de que todas as possibilidades precisem ser testadas, reduzindo a complexidade a níveis similares ao caso em que apenas uma transformada é aplicada.

Sugestões para Trabalhos Futuros

O próximo passo para este trabalho é a completa integração da proposta conjunta (algoritmo de estimação da direção do bloco juntamente com uma proposta de transformada direcional), aos codecs H.264/AVC e HEVC, permitindo analisar de forma integral o desempenho das transformadas direcionais, com e sem o algoritmo de estimação da direção do bloco.

Publicações

- G. Beltrão, R. Arthur, Y. Iano, "Recent Video Codecs: Comparison and Future Research", International Display Research Conference and LatinDisplay 2010, São Paulo, Brasil, Novembro 2010.
- G. Beltrão, R. Arthur, Y. Iano, "Introdução ao HEVC − High Efficiency Video Coding", 2° Simpósio de Processamento de Sinais da UNICAMP, São Paulo, Brasil, Outubro, 2011.
- G. Beltrão, J. León, R. Arthur, Y. Iano, "Fast Block Direction Prediction for Directional Transforms", Cyber Journals: Multidisciplinary Journals in Science and Technologies, Journal of Selected Areas in Telecommunications (JSAT), September Edition, 2012.

Publicações (Solo)

- B. Pompeo, L. Pralon, M. Pralon, H. Cioqueta, G. Beltrão, R. Mendes, "Radiation Pattern Generation Using a Modified Least Squares Method", Radar Methods and Systems Workshop (RMSW 2012), Kiev, Ukraine, September, 2012.
- B. Pompeo, L. Pralon, G. Beltrão, H. Cioqueta, B. Cosenza, J. Moreira, "On a Comparison Between Radar System Performance when Random and Linear Frequency Modulation Considering Sidelobe Suppression Techniques are Employed", International Conference on Noise Radar Technology (NRT-2012), Yalta, Ukraine, September, 2012.
- P.L. Vyplavin, G.Beltrão, J. Moreira, K.A.Lukin, "Noise Radar operating in High PRF Mode", International Conference on Noise Radar Technology (NRT-2012), Yalta, Ukraine, September, 2012.
- L. Pralon, B. Pompeo, G. Beltrão, H. Cioqueta, B. Cosenza, João Moreira, "On a Blind Zone Elimination Method Based on Partial Compression Filter Design Using Random Waveforms For Monostatic Pulsed Radars", International Conference on Radar Systems (RADAR-2012), Glasgow, UK, October, 2012.
- L. Pralon, B. Pompeo, G. Beltrão, H. Cioqueta, B. Cosenza, J. P. Fortes, "Random Phase/Frequency Modulated Waveforms for Noise Radar Systems Considering Phase Shift", European Radar Conference (EURAD-2012), European Microwave Week, Amsterdam, November, 2012.

Bibliografia

[1] I. Richardson, *The H.264 advanced video compression standard*. John Wiley & Sons, 2011.

[2] J. Xu, B. Zeng, and F. Wu, "An Overview of Directional Transforms in Image Coding," in *Circuits and Systems (ISCAS), Proceedings of 2010 IEEE International Symposium on*, pp. 3036–3039, IEEE, 2010.

[3] G. Sullivan and T. Wiegand, "Rate-distortion Optimization for Video Compression," *Signal Processing Magazine, IEEE*, vol. 15, no. 6, pp. 74–90, 1998.

[4] I. Recommendation, "601-6: Studio encoding parameters of digital television for standard 4:3 and wide-screen 16:9 aspect ratios," *International Telecommunication Union*, 1995.

[5] J. Ostermann, J. Bormans, P. List, D. Marpe, M. Narroschke, F. Pereira, T. Stockhammer, and T. Wedi, "Video coding with h. 264/avc: tools, performance, and complexity," *Circuits and Systems magazine, IEEE*, vol. 4, no. 1, pp. 7–28, 2004.

[6] J. Chen, U. Koc, and K. Liu, *Design of digital video coding systems: a complete compressed domain approach*, vol. 12. CRC, 2002.

[7] I. Richardson, *H. 264 and MPEG-4 video compression*, vol. 20. Wiley Online Library, 2003.

[8] T. Wiegand, G. Sullivan, G. Bjontegaard, and A. Luthra, "Overview of the h. 264/avc video coding standard," *Circuits and Systems for Video Technology, IEEE Transactions on*, vol. 13, no. 7, pp. 560–576, 2003.

[9] S. Issa and O. Khalifa, "Performance analysis of dirac video codec with h. 264/avc," in *Computer and Communication Engineering (ICCCE), 2010 International Conference on*, pp. 1–6, IEEE, 2010.

[10] N. Ahmed, T. Natarajan, and K. Rao, "Discrete cosine transform," *Computers, IEEE Transactions on*, vol. 100, no. 1, pp. 90–93, 1974.

[11] M. Najim, "Karhunen loeve transform," *Modeling, Estimation and Optimal Filtering in Signal Processing*, pp. 335–340, 2010.

[12] P. Roeser and M. Jernigan, "Fast haar transform algorithms," *Computers, IEEE Transactions on*, vol. 100, no. 2, pp. 175–177, 1982.

[13] B. Fino and V. Algazi, "Unified matrix treatment of the fast walsh-hadamard transform," *Computers, IEEE Transactions on*, vol. 100, no. 11, pp. 1142–1146, 1976.

[14] B. Girod, "Transform coding," *EE398A Image and video Compression*.

[15] D. Huffman, "A method for the construction of minimum-redundancy codes," *Proceedings of the IRE*, vol. 40, no. 9, pp. 1098–1101, 1952.

[16] D. Marpe, G. Blättermann, and T. Wiegand, "Adaptive codes for h. 26l," *ITU-T SG16/6 document VCEG-L13, Eibsee, Germany*, 2001.

[17] A. Ahumada and C. Null, "Image quality: A multidimensional problem," *digital images and human vision*, pp. 141–148, 1993.

[18] S. Klein, "Image quality and image compression: a psychophysicist's viewpoint," in *Digital images and human vision*, pp. 73–88, MIT Press, 1993.

[19] S. Jumisko-Pyykkö, J. Häkkinen, and G. Nyman, "Experienced quality factors-qualitative evaluation approach to audiovisual quality," *Multimedia on Mobile Devices 2007*, vol. 6507, pp. 6507–21, 2007.

[20] I. Recommendation, "500-11, methodology for the subjective assessment of the quality of television pictures," *International Telecommunication Union, Geneva, Switzerland*, 2002.

[21] Z. Wang, L. Lu, and A. Bovik, "Video quality assessment based on structural distortion measurement," *Signal processing: Image communication*, vol. 19, no. 2, pp. 121–132, 2004.

[22] M. Pinson and S. Wolf, "A new standardized method for objectively measuring video quality," *Broadcasting, IEEE Transactions on*, vol. 50, no. 3, pp. 312–322, 2004.

[23] Z. Wang, A. Bovik, H. Sheikh, and E. Simoncelli, "Image quality assessment: From error visibility to structural similarity," *Image Processing, IEEE Transactions on*, vol. 13, no. 4, pp. 600–612, 2004.

[24] G. Bjontegard, "Calculation of average psnr differences between rd-curves," *ITU-T VCEG-M33*, 2001.

[25] D. Marpe, H. Schwarz, S. Bosse, B. Bross, P. Helle, T. Hinz, H. Kirchhoffer, H. Lakshman, T. Nguyen, S. Oudin, *et al.*, "Video compression using nested quadtree structures, leaf merging, and improved techniques for motion representation and entropy coding," *Circuits and Systems for Video Technology, IEEE Transactions on*, vol. 20, no. 12, pp. 1676–1687, 2010.

[26] *Advanced Video Coding*. ITU-T Rec. H.264 ISO/IEC 14496-10:2009, Mar 2010.

[27] "ITU-T JCT-VC website. [online]. Available: `http://www.itu.int/en/ITU-T/studygroups/com16/video/Pages/jctvc.aspx`."

[28] *Joint Call for Proposals on Video Compression Technology.* ITU-T SG16 Q6 document VCEG-AM91 and ISO/IEC JTC1/SC29/WG11 document N11113, ITU-T SG16 Q6 and ISO/IEC JTC1/SC29/WG11, Jan. 2010.

[29] V. Baroncini, J. Ohm, and G. Sullivan, "Report of subjective test results of responses to the joint call for proposals (cfp) on video coding technology for high efficiency video coding (hevc)," in *Proc. 1st JCTVC Meeting*, 2010.

[30] JCTVC-A205, "Test Model Under Consideration," Tech. Rep. JCTVC-A205, ITU-T SG16 WP3 and ISO/IEC JTC1/SC29/WG11, Joint Collaborative Team on Video Coding (JCT-VC), Dresden, Germany, Apr. 2010.

[31] I. Amonou et al., "Description of Video Coding Technology Proposal by France Telecom, NTT, NTT DOCOMO, Panasonic and Technicolor," Tech. Rep. JCTVC-A114, ITU-T SG16 WP3 and ISO/IEC JTC1/SC29/WG11, Joint Collaborative Team on Video Coding (JCT-VC), Dresden, Germany, Apr. 2010.

[32] M. Winken et al., "Video Coding Technology Proposal by Fraunhofer HHI," Tech. Rep. JCTVC-A116, ITU-T SG16 WP3 and ISO/IEC JTC1/SC29/WG11, Joint Collaborative Team on Video Coding (JCT-VC), Dresden, Germany, Apr. 2010.

[33] K. Ugur et al., "Video Coding Technology Proposal by Tandberg,Nokia, and Ericsson," Tech. Rep. JCTVC-A119, ITU-T SG16 WP3 and ISO/IEC JTC1/SC29/WG11, Joint Collaborative Team on Video Coding (JCT-VC), Dresden, Germany, Apr. 2010.

[34] M. Karczewicz et al., "Video Coding Technology Proposal by Qualcomm," Tech. Rep. JCTVC-A121, ITU-T SG16 WP3 and ISO/IEC JTC1/SC29/WG11, Joint Collaborative Team on Video Coding (JCT-VC), Dresden, Germany, Apr. 2010.

[35] W. J. Han et al., "Video Coding Technology Proposal by Samsung (and BBC)," Tech. Rep. JCTVC-A124, ITU-T SG16 WP3 and ISO/IEC JTC1/SC29/WG11, Joint Collaborative Team on Video Coding (JCT-VC), Dresden, Germany, Apr. 2010.

[36] T. Davies et al., "Video Coding Technology Proposal by Samsung (and BBC)," Tech. Rep. JCTVC-A125, ITU-T SG16 WP3 and ISO/IEC JTC1/SC29/WG11, Joint Collaborative Team on Video Coding (JCT-VC), Dresden, Germany, Apr. 2010.

[37] JCTVC-C402, "Encoder-side description of HEVC Test Model (HM)," Tech. Rep. JCTVC-C402, ITU-T SG16 WP3 and ISO/IEC JTC1/SC29/WG11, Joint Collaborative Team on Video Coding (JCT-VC), Guangzou, CN, Oct. 2010.

[38] JCTVC-D404, "HEVC Reference Software Manual," Tech. Rep. JCTVC-D404, ITU-T SG16 WP3 and ISO/IEC JTC1/SC29/WG11, Joint Collaborative Team on Video Coding (JCT-VC), Daegu, KR, Jan. 2011.

[39] G. Sullivan and J. Ohm, "Recent Developments in Standardization of High Efficiency Video Coding (HEVC)," in *Proc. SPIE*, vol. 7798, pp. 7798–30, 2010.

[40] T. Wiegand, J. Ohm, G. Sullivan, W. Han, R. Joshi, T. Tan, and K. Ugur, "Special Section on the Joint Call for Proposals on High Efficiency Video Coding (HEVC) Standardization," *Circuits and Systems for Video Technology, IEEE Transactions on*, vol. 20, no. 12, pp. 1661–1666, 2010.

[41] JCTVC-H1002, "HM6: High Efficiency Video Coding (HEVC) Test Model 6 Encoder Description," Tech. Rep. JCTVC-H1002, ITU-T SG16 WP3 and ISO/IEC JTC1/SC29/WG11, Joint Collaborative Team on Video Coding (JCT-VC), San JosÃ©, CA, USA, Feb. 2012.

[42] W. J. Han et al., "Improved Video Compression Efficiency Through Flexible Unit Representation and Corresponding Extension of Coding Tools," *Circuits and Systems for Video Technology, IEEE Transactions on*, vol. 20, no. 12, pp. 1709–1720, 2010.

[43] JCTVC-B100, "Unification of the directional intra prediction methods in tmuc," Tech. Rep. JCTVC-B100, ITU-T SG16 WP3 and ISO/IEC JTC1/SC29/WG11, Joint Collaborative Team on Video Coding (JCT-VC), Geneva, CH, Jul. 2010.

[44] W. Chen, C. Smith, and S. Fralick, "A Fast Computational Algorithm for the Discrete Cosine Transform," *Communications, IEEE Transactions on*, vol. 25, no. 9, pp. 1004–1009, 1977.

[45] J. Han, A. Saxena, and K. Rose, "Towards Jointly Optimal Spatial Prediction and Adaptive Transform in Video/Image Coding," in *Acoustics Speech and Signal Processing (ICASSP), 2010 IEEE International Conference on*, pp. 726–729, IEEE, 2010.

[46] JCT-VC, "CE7: On Secondary Transforms for Intra Prediction Residual," Tech. Rep. JCTVC-G108_v4, ITU-T SG16 WP3 and ISO/IEC JTC1/SC29/WG11, Joint Collaborative Team on Video Coding (JCT-VC), Geneva, CH, Nov. 2011.

[47] JCT-VC, "Non-CE 7: Experimental Results for the ROT," Tech. Rep. JCTVC-H0456, ITU-T SG16 WP3 and ISO/IEC JTC1/SC29/WG11, Joint Collaborative Team on Video Coding (JCT-VC), San Jose, USA, Feb. 2012.

[48] "HEVC ftp test sequences. [online]. available: `"ftp://hevc@ftp.tnt.uni-hannover.de/orig-cfp/`."

[49] K. S. Kenji Kondo and Teruhiko, "Coments on common test conditions," Tech. Rep. JCTVC-C168, ISO/IEC JTC1/SC29/WG11, Joint Collaborative Team on Video Coding (JCT-VC), Oct. 2010.

[50] "HEVC Software website. [online]. Available: `https://hevc.hhi.fraunhofer.de/svn/svn_HEVCSoftware/tags/`."

[51] S. Park, J. Park, and B. Jeon, "Report on the evaluation of hm versus jm," *ISO/IEC JTC1/SC29/WG11, JCTVC-D181*, 2011.

[52] J. Fu and B. Zeng, "A comparative study of compensation techniques in directional dct's," in *Circuits and Systems, 2007. ISCAS 2007. IEEE International Symposium on*, pp. 521–524, IEEE, 2007.

[53] J. Dong, K. Ngan, C. Fong, and W. Cham, "2-d order-16 integer transforms for hd video coding," *Circuits and Systems for Video Technology, IEEE Transactions on*, vol. 19, no. 10, pp. 1462–1474, 2009.

[54] F. Kamisli and J. S. Lim, "1-D Transforms for the Motion Compensation Residual," *Image Processing, IEEE Transactions on*, no. 99, pp. 1036–1046, 2011.

[55] K. Hui and W. Siu, "Extended analysis of motion-compensated frame difference for block-based motion prediction error," *Image Processing, IEEE Transactions on*, vol. 16, no. 5, pp. 1232–1245, 2007.

[56] C. Chen and K. Pang, "The optimal transform of motion-compensated frame difference images in a hybrid coder," *Circuits and Systems II: Analog and Digital Signal Processing, IEEE Transactions on*, vol. 40, no. 6, pp. 393–397, 1993.

[57] W. Niehsen and M. Brunig, "Covariance analysis of motion-compensated frame differences," *Circuits and Systems for Video Technology, IEEE Transactions on*, vol. 9, no. 4, pp. 536–539, 1999.

[58] J. Dong and K. Ngan, "Two-layer directional transform for high performance video coding," *Circuits and Systems for Video Technology, IEEE Transactions on*, vol. 22, no. 4, pp. 619–625, 2012.

[59] J. Lim, F. Kamışlı, *et al.*, *"Transforms for prediction residuals in video coding"*. PhD thesis, Massachusetts Institute of Technology, 2010.

[60] B. Zeng and J. Fu, "Directional discrete cosine transforms: a new framework for image coding," *Circuits and Systems for Video Technology, IEEE Transactions on*, vol. 18, no. 3, pp. 305–313, 2008.

[61] C. Chang and B. Girod, "Directionadaptive partitioned block transform for image coding," in *Image Processing, 2008. ICIP 2008. 15th IEEE International Conference on*, pp. 145–148, IEEE, 2008.

[62] W. Ding, F. Wu, X. Wu, S. Li, and H. Li, "Adaptive directional lifting-based wavelet transform for image coding," *Image Processing, IEEE Transactions on*, vol. 16, no. 2, pp. 416–427, 2007.

[63] H. Xu, J. Xu, and F. Wu, "Lifting-based directional dct-like transform for image coding," *Circuits and Systems for Video Technology, IEEE Transactions on*, vol. 17, no. 10, pp. 1325–1335, 2007.

[64] C. Chang and B. Girod, "Direction-adaptive discrete wavelet transform for image compression," *Image Processing, IEEE Transactions on*, vol. 16, no. 5, pp. 1289–1302, 2007.

[65] Y. Liu and K. Ngan, "Weighted adaptive lifting-based wavelet transform for image coding," *Image Processing, IEEE Transactions on*, vol. 17, no. 4, pp. 500–511, 2008.

[66] Y. Ye and M. Karczewicz, "Improved h. 264 intra coding based on bi-directional intra prediction, directional transform, and adaptive coefficient scanning," in *Image Processing, 2008. ICIP 2008. 15th IEEE International Conference on*, pp. 2116–2119, IEEE, 2008.

[67] X. Peng, F. Wu, and J. Xu, "Directional filtering transform," in *Multimedia and Expo, 2009. ICME 2009. IEEE International Conference on*, pp. 1–4, IEEE, 2009.

[68] B. Zeng and J. Fu, "Directional discrete cosine transforms for image coding," in *Multimedia and Expo, 2006 IEEE International Conference on*, pp. 721–724, IEEE, 2006.

[69] D. Lowe, "Distinctive Image Features from Scale-invariant Keypoints," *International journal of computer vision*, vol. 60, no. 2, pp. 91–110, 2004.

[70] F. Pan, X. Lin, S. Rahardja, K. Lim, Z. Li, D. Wu, and S. Wu, "Fast Mode Decision Algorithm for Intraprediction in H.264/AVC Video Coding," *Circuits and Systems for Video Technology, IEEE Transactions on*, vol. 15, no. 7, pp. 813–822, 2005.

[71] W. Jiang, H. Ma, and Y. Chen, "Gradient Based Fast Mode Decision Algorithm for Intra Prediction in HEVC,"