Nuno Miguel Fernandes Carvalho

Controlo de Complexidade do Codificador de Vídeo 3D-HEVC

Setembro 2015

· U C ·

Universidade de Coimbra

Departamento de Engenharia Eletrotécnica e de Computadores
Faculdade de Ciências e Tecnologia
Universidade de Coimbra

Mestrado Integrado em Engenharia Eletrotécnica e de Computadores

Controlo de Complexidade do Codificador de Vídeo 3D-HEVC

Nuno Miguel Fernandes Carvalho

Sobre a supervisão
Prof. Doutor. Luís Alberto da Silva Cruz
(Orientador)

Júri:
Henrique José Almeida da Silva
Luís Alberto da Silva Cruz
Vítor Manuel Mendes da Silva
Fernando José Pimentel Lopes

Coimbra
Setembro 2015

Resumo

O aumento do número de aplicações de vídeo 3D impulsionou o desenvolvimento de uma extensão ao codificador de vídeo H.265/HEVC (*High Efficiency Video Coding)*, designada 3D-HEVC para suportar a codificação de conteúdos de vídeo 3D. Esta extensão permite codificar de forma eficiente informação de vídeo 3D suportando sinais de vídeo multi-vista e informação de profundidade. A extensão 3D-HEVC aumentou a eficiência da compressão, quando comparada com a codificação independente das vistas e componentes (textura e profundidade) mas a sua complexidade computacional, medida em tempo de codificação é muito grande. Este é um problema grave, especialmente para aplicações em tempo real.

Nesta dissertação são apresentados três algoritmos, para lidar com o aumento da complexidade computacional do 3D-HEVC. O primeiro algoritmo é uma adaptação de um método de controlo de complexidade, inicialmente desenvolvido para o H.265/HEVC. Este método aplica um conjunto de restrições ao processo de codificação para que a complexidade computacional deste não ultrapasse um valor alvo. O segundo algoritmo é uma modificação do método anterior que introduz ganhos adicionais em termos de exatidão do controlo. Nesta versão o controlador, responsável por definir as restrições, foi alterado com o objetivo de melhorar a exatidão com que o algoritmo ajusta o tempo de codificação ao tempo alvo. O último algoritmo é um método de redução de complexidade computacional que usa análise estatística e a informação redundante existente dentro da mesma trama e entre tramas diferentes, para reduzir a complexidade e consequentemente diminuir o tempo de codificação associada a este processo, sem prejuízo significativo para a eficiência do processo de codificação.

Palavras-chave: 3D-HEVC, controlo de complexidade computacional, redução de complexidade computacional, modos de predição e unidades de codificação.

Abstract

The growing number of 3D video applications has fostered the development of a 3D video extension of the HEVC coding standard, called 3D-HEVC. This extension is capable of encoding 3D video information in an efficient way supporting the encoding of multiview video and texture plus depth based formats. Although 3D-HEVC has increased the compression efficiency of 3D video when compared to independent encoding, its computational complexity is very high. This is a serious problem especially for real-time applications.

This dissertation describes three algorithms that address the 3D-HEVC high computational complexity problem. The first algorithm is an adaptation of a method for computational complexity scaling, initially developed for H.265/HEVC. This method applies a set of constraints to the encoding process with the goal of keeping the actual computational complexity below a target value. The second algorithm is a modified version of the previous method. In this version the controller responsible for defining the restrictions has been modified, in order to improve the accuracy of the encoding time control algorithm. The last method proposed is based on an algorithm to reduce the computational complexity which is based on a statistical analysis of the intra and inter frame information coding mode and structures redundancies.

Keywords: 3D-HEVC, Computational complexity scaling, Computational complexity reduction, prediction modes and coding units.

Agradecimentos

Agradeço ao meu orientador, Prof. Doutor Luís Alberto da Silva Cruz, a orientação científica proporcionada ao longo do trabalho, a disponibilidade demonstrada para me ajudar a ultrapassar as fases mais complicadas e a abertura que sempre demonstrou ao discutir e analisar as minhas ideias e sugestões.

Agradeço de igual forma ao Doutor Guilherme Corrêa, o acesso ao seu trabalho sobre algoritmos de controlo e redução de complexidade computacional do H.265/HEVC e por todo o apoio e cooperação proporcionado durante este trabalho.

Agradeço a todos os meus amigos, colegas de curso e de laboratório pela amizade e companheirismo demonstrado ao longo do trabalho. E finalmente à minha família pelo seu apoio incondicional e às outras pessoas que me ajudaram ao longo do trabalho.

Índice

Lista de Tabelas

Lista de Figuras

Lista de Abreviaturas

3D-HEVC – *3D-High Efficiency Video coding*

BD-PSNR – *Bjøntegaard Delta PSNR*

BD-rate – *Bjøntegaard Delta RATE*

CB – *Coding Block*

CCUPU – *Constrained Coding Units and Prediction Units*

CD – *Compact Disc*

CD-ROM – *Compact Disc-Read Only Memory*

CIF – *Common Intermediate Format*

CTB – *Coding Tree Block*

CTU – *Coding Tree Units*

CU – *Coding Units*

DCP – *Disparity-Compensated Prediction*

DF – *Deblocking Filter*

DIBR – *Depth Image Based Rendering*

DPB – *Decoded Picture Buffer*

DPCM – *Differential Pulse Code Modulation*

DVD – *Digital Versatile Disc*

EM – Estimação de Movimento

FCA – *Fast Coding Algorithm*

FDP – Função Densidade de Probabilidade

FES – *Fast Encoding Scheme*

FMDARTA – *Fast Mode Decision Algorithm for Real-Time Applications.*

GOP – *Group Of Pictures*

GPB – *Generalized P or B*

HD – *High Definition*

H.265/HEVC – *H.265/High Efficiency Video Coding*

IDR – *Instantaneous Decoding Refresh*

IPP – *Intra-Picture Prediction*

ISO/IEC – *International Organization for Standardization / International Electrotechnical Commission*

ITU-T - *International Telecommunication Union Telecommunication Standardization Sector*

MPEG – *Motion Picture Experts Group*

MSE – *Mean Squared Error*

MSM – *Merge/Skip Mode*

QCIF – *Quarter Common Intermediate Format*

QP – *Quantization Parameter*

PSNR – *Peak-Signal-to-Noise-Ratio*

PU – *Prediction Units*

RDO – *Rate-Distortion Optimization*

SAO - *Sample Adaptive Offset*

T.C. – Tempo de codificação

TU – *Transform Units*

UHD – *Ultra High Definition*

VCEG – *Video Coding Experts Group*

Capítulo 1 – Introdução

A evolução tecnológica da eletrónica nas últimas décadas possibilitou o aumento dos recursos computacionais disponíveis. Este aumento permitiu o desenvolvimento de novas técnicas de codificação de vídeo, capazes de atingir níveis de compressão cada vez mais elevados com perdas de qualidade impercetíveis ao olho humano.

A compressão de vídeo baseia-se na eliminação de informação visualmente irrelevante e na redução da quantidade de informação redundante características de sinais de vídeo, dessa forma diminuindo o seu volume possibilitando assim a transmissão e armazenamento de forma mais eficiente. Esta tecnologia é a base da generalidade das aplicações de vídeo contemporâneas, tais como; os dispositivos de armazenamento *Digital Versatile Disc* (DVD), serviços de videoconferência e sistemas de distribuição de televisão. Para estas tecnologias o uso de vídeo não comprimido é incompatível com as restrições existentes a nível de largura de banda, espaço de armazenamento ou congestionamento da rede, em particular em serviços com transmissão via internet.

A tecnologia atual possibilita um elevado nível de compressão, contudo, o surgimento de novas resoluções e aplicações de vídeo, tais como vídeo de ultra alta definição (*Ultra-High-Definition* -UHD), e as aplicações de *streaming* de vídeo impõem a necessidade de aumentar os níveis de compressão. Porém, a confirmação dos dispositivos móveis como os dipositivos de eleição por parte dos consumidores traz consigo restrições à capacidade computacional disponível. O que origina que as novas técnicas de codificação de vídeo, capazes de aumentar os níveis de compressão, sofram restrições mais severas no uso dos recursos computacionais.

O processo de compressão e descompressão de um sinal de vídeo digital é efetuado pelo codificador e descodificador respetivamente, sendo que estes podem ser realizados por conjunto de circuitos eletrónicos ou implementados em *software*. Estes processos apresentam alguns inconvenientes tais como; perda de qualidade visual do vídeo, o uso de elevados recursos computacionais e imposição de um atraso devido ao processamento.

Um dos primeiros codificadores de vídeo, segundo a norma H.120 foi lançado na década de 80 para ser utilizado em sinais de vídeo PAL (*Phase Alternating Line*) e NTSC (*National Television System Committee*), este codificador utilizava *Differential Pulse Code Modulation* (DPCM) e a quantização escalar para reduzir o *bitrate* gerado. O H.120 operava pixel-a-pixel o

que lhe permitia obter boa resolução espacial contudo, apresentava grandes limitações ao nível da resolução temporal. No final da década de 80 foi projetado o H.261 para codificar vídeo em formato *Common Intermediate Format* (CIF), de resolução 352×288 pixéis, e vídeo em formato *Quarter Common Intermediate Format* (QCIF) de resolução 176×144 pixéis. Os bons resultados em termos de *bitrate* gerado (384*kbits/s* em imagens CIF) tornaram-no uma referência para o desenvolvimento de novos codificadores. No início da década de 90, o grupo *Motion Picture Experts Group* (MPEG) usou o H.261 como ponto de partida para o desenvolvimento do MPEG-1. Este *codec* foi desenvolvido com o objetivo de permitir um armazenamento eficiente de vídeos em dispositivos como o *Compact Disc-Read Only Memory* (CD-ROM). Na mesma década surgiu o H.262/MPEG-2 especialmente desenvolvido para codificar vídeos de resolução 720×576/480 pixéis e vídeo de alta resolução, 1920×1080 pixéis. O H.262/MPEG-2 adicionou uma importante funcionalidade à codificação – a escalabilidade. A escalabilidade permite extrair de um único *bitstream* um conjunto de vídeos com diferentes resoluções espaciais, temporais e de qualidade. O crescimento do uso de dispositivos móveis com câmaras fotográficas incorporadas, como os telemóveis, origina o desenvolvimento de um novo *codec*- o MPEG-4, especializado em vídeos com baixa resolução e *bitrate*. Porém, este *codec* nunca atingiu plenamente os seus objetivos. Estes foram atingidos pelo H.263 cujos projetistas aprendendo com as falhas do MPEG, desenvolveram um codec capaz de codificar de forma eficiente vídeos de baixa resolução e *bitrate*. Em 2013 foi terminado o primeiro codificador de vídeo segundo a norma H.264/AVC. Este *codec* foi projetado com o objetivo de superar a redução do *bitrate* obtida pelos seus antecessores, mantendo uma qualidade de vídeo aproximada. O aumento da eficiência na compressão de vídeo, alcançada por este codificador, tornou-o o *standard* de referência e motivou que fosse adotado em diversas aplicações de codificação de vídeo, tais como; o *Ipod* e a *Playstation Portable*.

O mais recente método normalizado para compressão de vídeo é o H.265/HEVC [1]. Este codificador foi projetado para reduzir em 50% o *bitrate* gerado relativamente ao H.264/AVC [2], seu antecessor. No entanto, este enorme aumento da eficiência da codificação foi acompanhado por um aumento da complexidade computacional do codificador entre 9% e 502% [3]. Este crescimento de complexidade pode representar um problema para aplicações em tempo real, que têm restrições temporais rígidas.

O aumento do número de aplicações de vídeo 3D levou ao desenvolvimento de uma extensão do H.265/HEVC – designada por 3D-HEVC [4], para codificação eficiente de vídeos 3D. Esta extensão é capaz de codificar eficientemente dados de profundidade que em alguns formatos de vídeo 3D acompanham a textura, além de, suportar codificação de múltiplas vistas.

Neste tipo de vídeo, no caso mais geral cada vista é constituída por tramas de textura e profundidade, como ilustrado na Figura 1.1. Para manter a compatibilidade com H.265/HEVC uma das vistas, a vista independente, é codificada usando H.265/HEVC e as restantes vistas, vistas dependentes, são codificadas usando ferramentas de predição específicas à extensão 3D. A complexidade computacional associada à extensão 3D do H.265/HEVC cresce linearmente com o número de vistas.

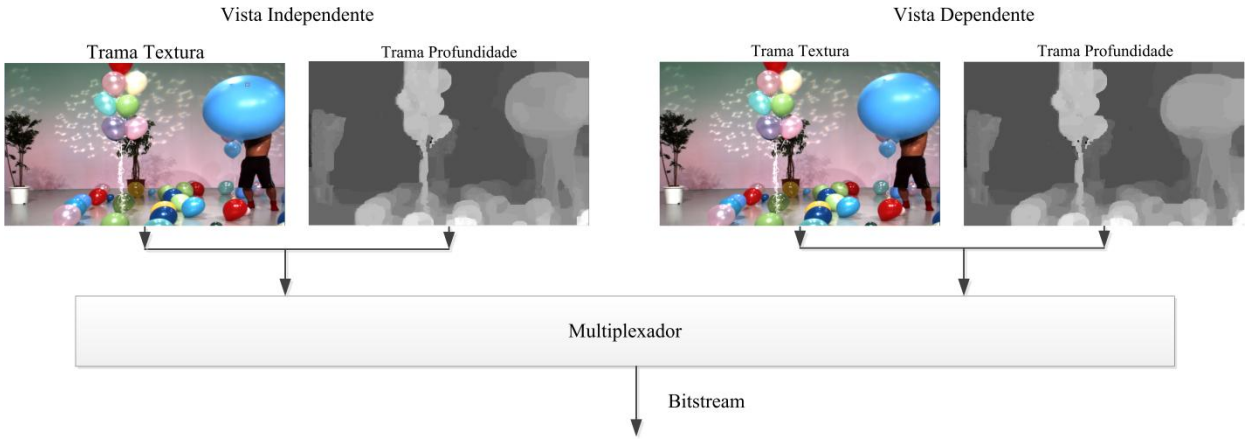


Fig. 1.1 - Estrutura de codificação do 3D-HEVC.

O principal foco desta dissertação é o estudo, proposta e avaliação de soluções para o controlo e a redução da complexidade computacional da extensão 3D-HEVC mantendo uma elevada eficiência de compressão.

1.1 Resumo da Dissertação

Com o objetivo de ajustar a complexidade computacional do 3D-HEVC mantendo-a abaixo de um valor alvo, foi desenvolvida uma versão do algoritmo *Constrained Coding Units and Prediction Units* (CCUPU) [5] criado originalmente para H.265/HEVC [1], que é aqui adaptado à extensão 3D-HEVC.

O algoritmo CCUPU implementado no codificador H.265/HEVC permitia ajustar a complexidade computacional do H.265/HEVC a uma complexidade alvo. A primeira versão deste algoritmo adaptado ao 3D-HEVC tem como objetivo manter a complexidade de codificação das tramas de textura da vista dependente abaixo de um valor alvo.

Posteriormente, para melhorar o controlo de complexidade computacional foi desenvolvida uma segunda versão do algoritmo, onde foi adicionado um novo controlador que permite controlar de forma mais precisa a complexidade computacional.

O último algoritmo apresentado tem como objetivo reduzir a complexidade computacional do 3D-HEVC mantendo uma elevada eficiência de codificação. No desenvolvimento deste algoritmo foram realizadas diversas análises estatísticas e estudos da correlação entre os modos de predição *inter* e a profundidade máxima da árvore de codificação nas diferentes vistas, diferentes tramas dentro da mesma vista e de diferentes regiões dentro da mesma trama, com a finalidade de acelerar os processos de decisão do 3D-HEVC, diminuindo assim a complexidade computacional associada.

De seguida, é apresentada a lista de algoritmos desenvolvidos:

Algoritmos de controlo de complexidade computacional do 3D-HEVC:

- CCUPU adaptado ao 3D-HEVC. Este método permite que a complexidade computacional associada às tramas de textura da vista dependente não ultrapasse um valor de complexidade alvo (*target*) pré definido.
- CCUPU com um sistema de controlo preditivo. Este método é implementado na extensão 3D-HEVC e impõe que a complexidade computacional das tramas de textura e de profundidade da vista dependente, não ultrapasse um *target* pré definido.

Algoritmo de redução de complexidade do 3D-HEVC:

- *Fast Encoding Scheme* (FES) é um algoritmo que reduz a complexidade computacional da codificação 3D-HEVC. Esta redução é obtida através da diminuição do número de modos de predição *inter* testados durante o processo de codificação e da limitação da profundidade máxima da árvore de codificação.

1.2 Organização da Dissertação

A dissertação é organizada da seguinte forma:

No segundo capítulo é apresentada uma breve descrição do funcionamento do codificador de vídeo H.265/HEVC e da sua extensão 3D, o 3D-HEVC. São introduzidos métodos de avaliação dos efeitos da distorção resultantes do processo de codificação e descodificação nas sequências de vídeo. É apresentado o modelo de Bjøntegaard que é usado actualmente para comparar o desempenho de dois algoritmos de compressão de vídeo. Por último, é realizada uma breve discussão sobre a complexidade computacional associada à codificação de H.265/HEVC e 3D-HEVC.

No terceiro capítulo é apresentada uma revisão do estado da arte relativa à redução e controlo da complexidade computacional do codificador de vídeo H.265/HEVC e 3D-HEVC. Na secção do estado da arte associado ao H.265/HEVC é realizada uma descrição geral dos diversos algoritmos existentes. É apresentada uma revisão mais detalhada sobre os algoritmos de maior relevância para o tema, projetados para a extensão 3D-HEVC. A segunda parte deste capítulo introduz a descrição do algoritmo CCUPU.

O quarto e quinto capítulo apresentam o conjunto de algoritmos de controlo e redução de complexidade computacional resultantes do trabalho realizado nesta dissertação. Após a apresentação de cada algoritmo é realizada uma breve discussão sobre os resultados obtidos.

As principais conclusões deste trabalho são apresentadas no sexto capítulo, assim como, possíveis temas de investigação centrados no problema da redução de complexidade da extensão 3D-HEVC e possíveis extensões ao trabalho apresentado.

Capítulo 2 – Codificação de vídeo

Este capítulo introduz um conjunto de conceitos básicos sobre a codificação de vídeo. Estes conceitos permitem adquirir competências necessárias para contextualizar o problema e as soluções apresentadas. Inicialmente são descritas as características e funcionalidades de maior relevância do codificador H.265/HEVC e da sua extensão 3D. Para avaliar os efeitos negativos do processo de codificação na qualidade do vídeo são introduzidas medidas de avaliação de qualidade e o modelo de Bjøntegaard, que permite comparar a performance de dois algoritmos de codificação de vídeo. A avaliação é feita tendo por base os valores de *Peak-Signal-to-Noise-Ratio* (PSNR) e *bitrate*. Por último, é apresentada uma breve reflexão sobre a redução de complexidade computacional do H.265/HEVC e do 3D-HEVC, que é o foco principal da dissertação.

2.1 H.265/HEVC

O H.265/HEVC é uma norma de codificação de vídeo desenvolvido pelas equipas dos grupos *Video Coding Experts Group* (VCEG) do *International Telecommunication Union Telecommunication Standardization Sector (ITU-T)* e *Moving Picture Experts Group* (MPEG) do *International Organization for Standardization*/*International Electrotechnical Commission* (ISO/IEC), com o objetivo de aumentar a eficiência de codificação da norma antecessora, H.264/AVC. Esta necessidade deve-se, entre outros fatores, ao uso cada vez mais frequente de vídeos de ultra alta resolução, onde o fator de compressão conseguido através do H.264 não é muito elevado e à necessidade de melhoria de características, tais como, a robustez e a capacidade de processamento paralelo, ou seja, permitir que diferentes regiões pertencentes à mesma trama sejam codificadas/descodificadas em paralelo.

O H.265/HEVC usa uma abordagem híbrida, ou seja, sobre o sinal de vídeo são aplicados os seguintes processo; i) predição *intra* e predição *inter*, ii) sobre o resido da predição é aplicada uma transformada para descorrelacionar os *pixies*, no domínio espacial. iii) Os coeficientes resultantes desta transformada são posteriormente quantificados e codificados. A predição *intra* explora a redundância espacial existente dentro de uma trama, nesta predição um conjunto de pixéis é predito com base nos pixéis pertencentes à mesma trama, anteriormente codificados. Quando a região de predição é proveniente de uma trama destinta, designa-se por predição *inter*.

Na Figura 2.1 encontra-se descrito sob a forma de um diagrama de blocos a estrutura de um codificador de vídeo H.265/HEVC .

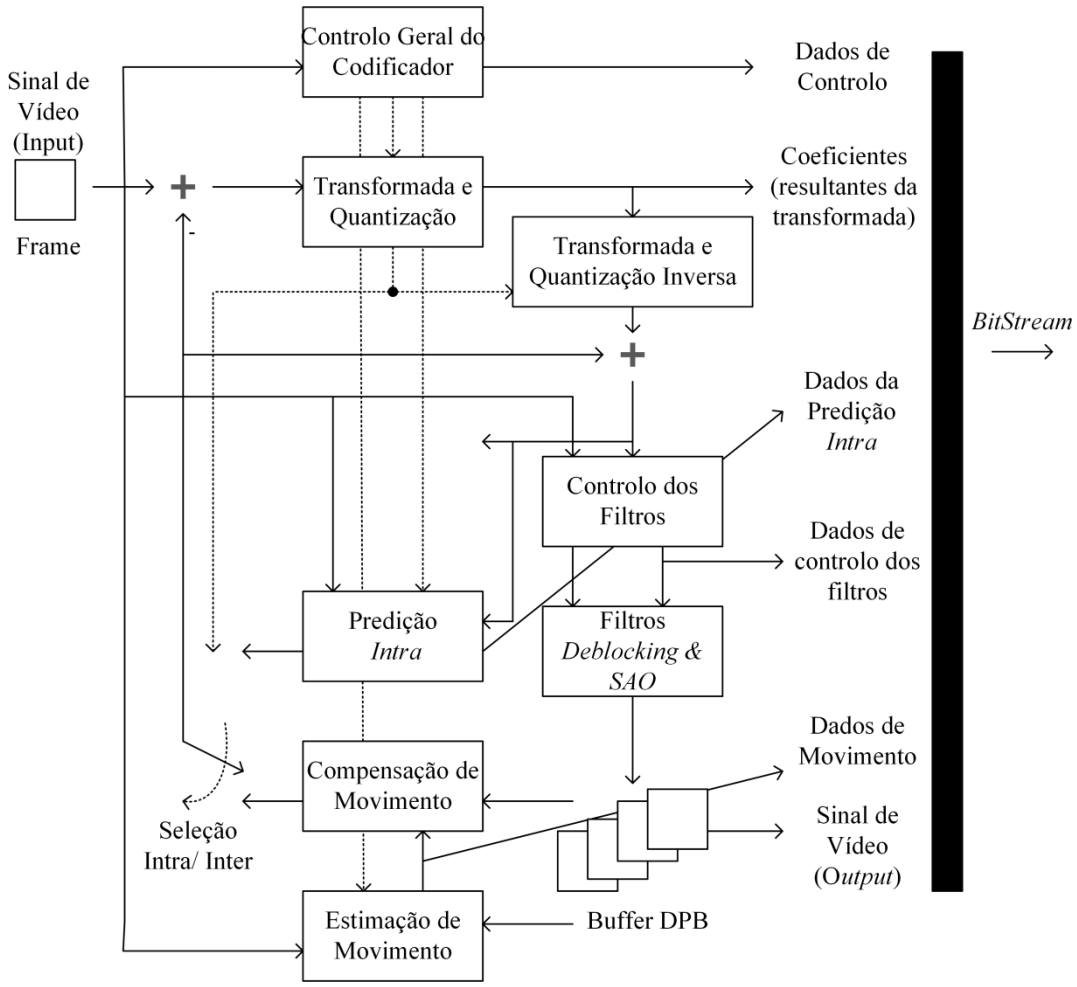


Fig. 2.1 - Codificador de vídeo H.265/HEVC adaptado de [1].

No H.265/HEVC cada sequência de vídeo é divida em grupos de tramas, designadas por *Group Of Pictures* (GOP). Um GOP é limitado por duas tramas I, que são codificadas só com predição *intra*, estas constituem pontos de acesso onde o vídeo é descodificado sem que existam dependências entre tramas. As tramas dentro de cada GOP são do tipo B, ou seja, usam ambas as predições *intra* e *inter*, como ilustrado na Figura 2.2.

Fig. 2.2 - Estrutura de um GOP adaptado [6].

As tramas no H.265/HEVC encontram-se no formato YUV [7], Y representa a luminância (informação do preto e branco) e o U e o V representam as crominâncias (componentes da cor). Este formato permite que o processo de codificação beneficie da maior

sensibilidade do sistema visual humano à informação da luminância do que à informação das crominâncias.

Sobre as componentes das crominâncias são aplicados vários formatos de sub-amostragem da cor, entre os quais; 4:4:4, 4:2:2 e 4:2:0. No primeiro formato as três componentes (YUV) têm a mesma resolução, ou seja, existe uma amostra das três componentes para cada *pixel*. No formato 4:2:2 as crominâncias apresentam a mesma resolução vertical, que a luminância, mas metade da resolução horizontal, no último modelo as crominancias têm metade da resolução vertical e horizontal, Figura 2.3.

As três componentes do formato YUV podem ser obtidas através do modelo de cores RGB, sistema de cores utilizado em ecrãs de computadores, telemóveis, câmaras fotográficas e televisões, onde R representa a cor vermelha, G é a cor verde e B a cor azul, Equações 2.1, 2.2 e 2.3.

$$Y=0.299R+0.587G+0.114B \tag{2.1}$$

$$Cb=0.564(B-Y)+128 \tag{2.2}$$

$$Cr=0.713(R-Y)+128 \tag{2.3}$$

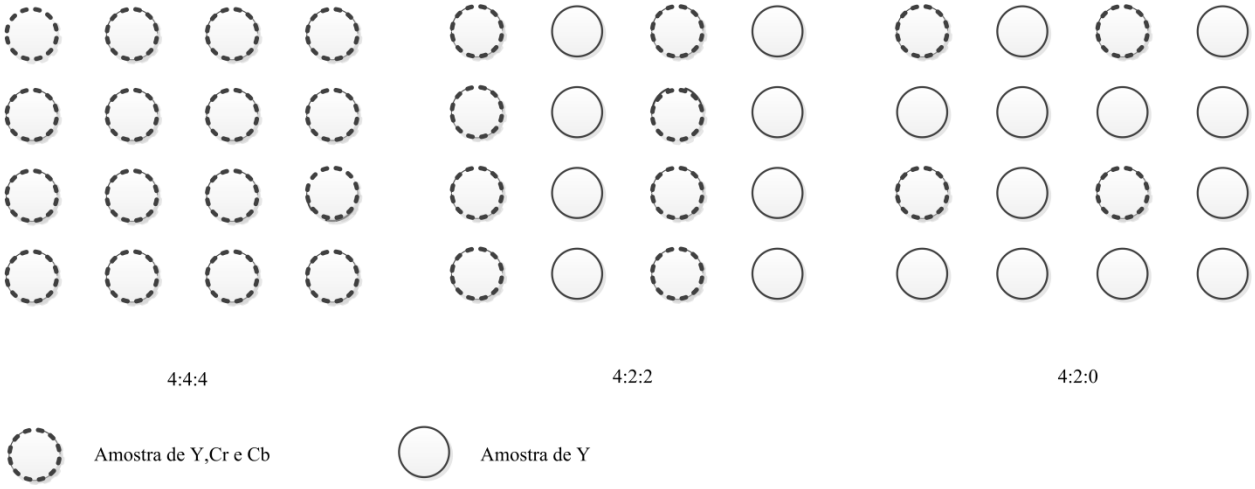


Fig. 2.3 - Formatos de sub-amostragem da cor.

Cada trama pode ser dividida em unidades chamas *slices* que são codificadas independentemente umas das outras. Esta característica é útil para lidar com perda de dados e implementar estratégias de paralelismo. As *slices* podem ser diferenciadas em três tipos; I, P e B, dependendo da predição usada. *Slices* do tipo I são codificadas só com predição *intra* as *slices* do tipo P e B são codificadas com ambas as predições. No tipo P a predição *inter* só usa regiões de

predição provenientes de tramas anteriores, segundo a ordem de reprodução do vídeo. No tipo B é permitido usar tramas anteriores e posteriores, segundo a ordem de reprodução do vídeo.

Cada *slice* é dividida em blocos de tamanho 64×64 pixéis, chamados *Coding Tree Units* (CTU), como ilustrado na Figura 2.4. Considerando uma amostragem da cor com o formato 4:2:0 a CTU é composta por uma luminância de tamanho N×N chamada *Coding Tree Block* (CTB) e as correspondentes CTBs de crominâncias de tamanho N/2×N/2, onde N pode ser 8, 16,32 e 64.

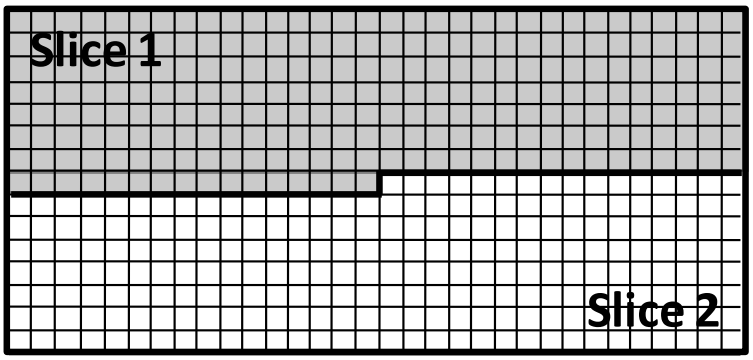


Fig. 2.4 - Trama dividida em duas *slices* e cada *slice* dividida em diversas CTUs [6].

As CTUs podem ser divididas em unidades menores chamadas *Coding Unit* (CU), usando uma estrutura em árvore. Cada CU contém uma unidade denominada *Coding Block* (CB) de luminância e as duas unidades de crominâncias. Cada CTU pode conter uma ou mais CUs, dependendo do número de vezes que for dividida. As CUs são repartidas em unidade chamadas *Prediction Units* (PU), sendo que o seu conteúdo partilha o mesmo modo de predição, e em unidades chamadas *Transform Units* (TU) que representam o conjunto de pixéis de resíduos resultantes da predição a serem codificados.

A Figura 2.5 apresenta a estrutura em árvore consequente da divisão de uma CTU em diversas CUs. A raiz desta estrutura em árvore está associada à CTU. A profundidade da árvore de codificação igual a zero indica que a CTU não é dividida e só contem uma CU de tamanho 64×64 pixéis. Uma profundidade de árvore de codificação igual a um significa que a CTU é repartida em quatro CUs de tamanho 32×32 pixéis. A divisão pode prosseguir até ao tamanho mínimo de 8×8 pixéis por CU, que representa uma profundidade da árvore de codificação igual a três.

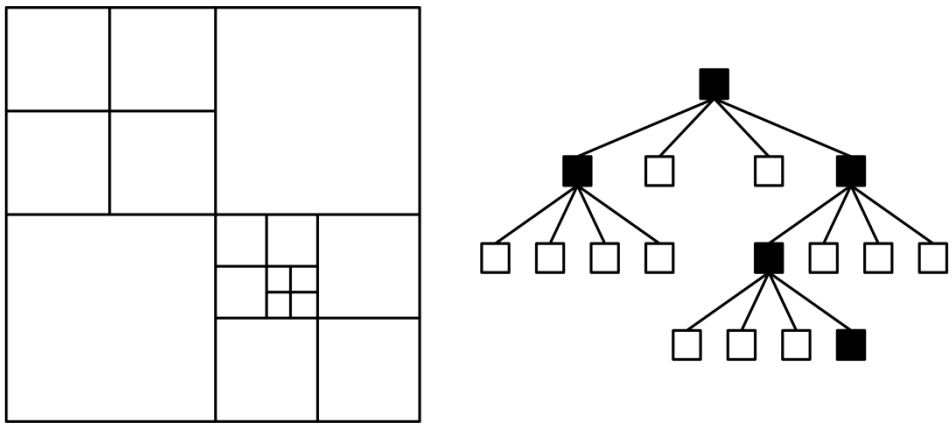


Fig. 2.5 - Subdivisão de uma CTU em CUs.

Cada CU tem associada uma estrutura de divisão em árvore para as PUs, como foi anteriormente referido. Quando a predição escolhida é *inter*, um dos modos mostrados na Figura 2.6 é selecionado para dividir a CU atual em PUs. Caso o modo de predição seja *intra* os modos de predição escolhidos dependem do tamanho da CU. CUs de tamanho 8×8 testam os modos 2N×2N e N×N, nos restantes tamanhos só o modo 2N×2N é permitido.

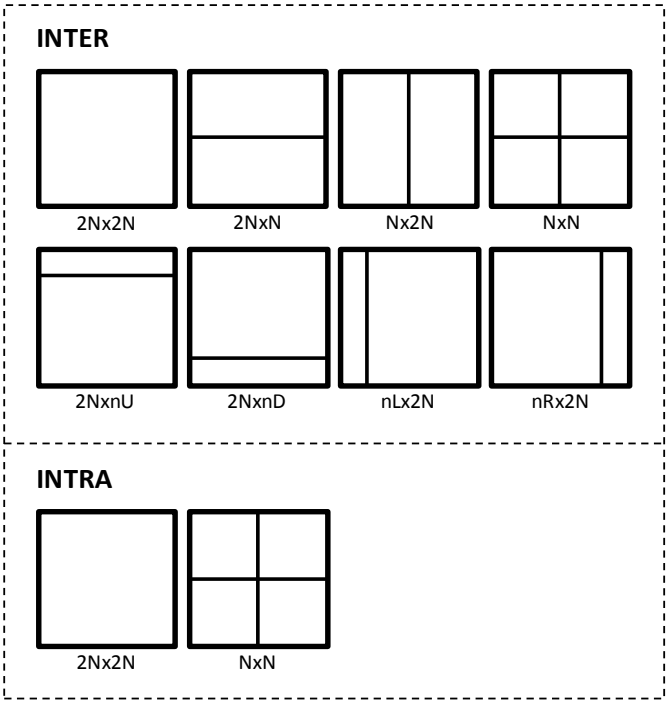


Fig. 2.6 - Modos de divisão de uma CU em PUs, para a predição *inter* e *intra*.

A seleção da melhor configuração, entre todas as combinações possíveis para os diversos tamanhos e partições das CUs, PUs e TUs é realizada através de um processo de pesquisa exaustiva, designado por *Rate-Distortion Optimization* (RDO), que procura determinar a combinação de partições que minimiza uma combinação da distorção introduzida no vídeo, e do *bitrate* gerado, isto é uma função taxa-distorção (*rate-distortion*). Este problema de optimização é resolvido através do uso de métodos lagrangeanos [8], contudo, este processo apresenta um

elevado custo ao nível da complexidade computacional, especialmente quando são testadas todas as configurações possíveis.

Após a divisão da *slice* em CTUs, das CTUs em CUs e posteriormente em PUs é calculado, para cada PU, o sinal residual obtido através do processo de predição *inter* ou *intra*.

Na predição *inter*, os componentes do codificador chamados *Estimação de Movimento* (EM) e *Compensação de Movimento*, Figura 2.1, são usados para ajudar a localizar as regiões de melhor predição provenientes de tramas anteriormente codificadas. O movimento estimado pelo EM é quantificado através de vetores de movimento que representam a localização dos melhores blocos de predição. Posteriormente estes vetores são aplicados no processo de compensação de movimento.

A predição *intra* só envolve pixéis da mesma trama, usando pixéis descodificados posicionados nas margens das PUs vizinhas à PU a codificar, como referência para a predição espacial, como ilustrado na Figura 2.7. A Figura 2.8 apresenta os modos de predição *intra* suportados pelo H.265/HEVC; trinta e três modos direcionais, um modo DC e o modo planar. Quando um modo direcional é escolhido, o valor de cada pixel de referência é utilizado como predição para os *pixies* localizados, dentro da PU a codificar, segundo a orientação escolhida. Na Figura 2.7 é ilustrado o modo direcional 2. No modo DC é calculada a média dos pixéis de referência, e esse valor é utilizado como predição para todos pixéis dentro da PU. Alternativamente, no modo planar é utilizada uma função linear para interpolar os pixéis de referência e gerar a predição dos pixéis.

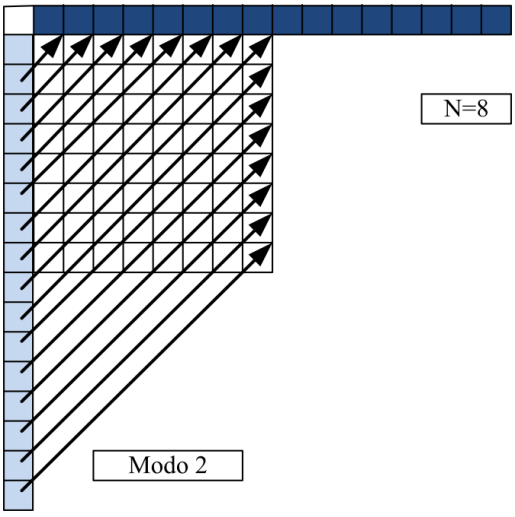


Fig. 2.7 – Pixéis de referência usados na predição *intra* de uma PU N×N de tamanho [1].

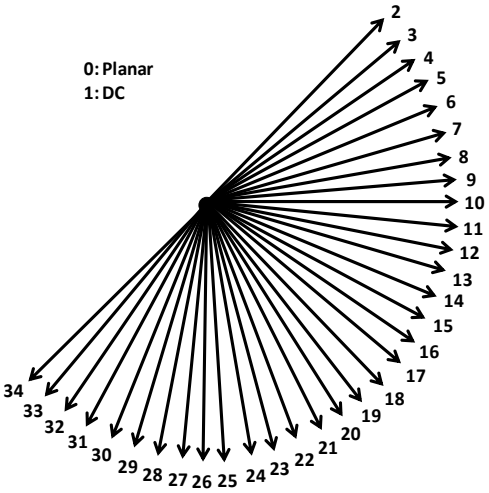


Fig. 2.8 - Modos de predição *intra* testados em cada PU [1].

Ao sinal residual resultante do processo de predição é aplicada uma transformada, similar à transformada discreta do cosseno (DCT) com o objetivo de remover a redundância entre os resíduos vizinhos. Os coeficientes resultantes da aplicação da transformada são quantificados, codificados entropicamente [9] e transmitidos juntamente com a informação de predição.

O descodificador H.265/HEVC segue um procedimento que é essencialmente o inverso do efetuado durante a codificação. O descodificador recebe o *bitstream* ao qual aplica os processos de descodificação entrópica, desquantização e a transformada inversa com o objetivo de recuperar o resíduo. O resíduo é adicionado à predição gerada localmente pelo descodificador, que é idêntica à predição gerada durante a codificação, de forma a reconstruir as amostras. Estas amostras são filtradas pelos filtros *Deblocking Filter* (DF) [10] e pelo *Sample Adaptive Offset* (SAO) [11] para suavizar artefactos que possam ter sidos introduzidos durante o processo de codificação. No final deste processo obtêm-se as imagens reconstruídas, que são armazenadas no *Decoded Picture Buffer* (DPB) para posteriormente serem utilizadas na predição de tramas subsequentes. Estas imagens são utilizadas pelos processos associados à predição *inter* com o objetivo de obter as melhores regiões de predição para cada PU a codificar, o que permite que o resíduo de predição seja menor e consequentemente o *bitrate* gerado na sua codificação seja também menor.

2.2 3D-HEVC

3D-HEVC designa a extensão da norma H.265/HEVC que permite a codificação de vídeos com conteúdo tridimensional (3D). A extensão mantém as principais características presentes no H.265/HEVC, tais como; a divisão das tramas em CTUs, as estruturas em árvores e o uso de predição *intra* e *inter*. A estas, foram adicionadas novas ferramentas para lidar com as especificidades do conteúdo 3D e beneficiar da redundância existente entre as diversas vistas e

entre as tramas de textura e de profundidade. A Figura 2.9 representa a estrutura de codificação do 3D-HEVC. Como foi referido no primeiro capítulo, os vídeos 3D são compostos por múltiplas vistas, cada uma composta por tramas de textura e tramas de profundidade. A primeira vista a ser codificada é sempre a vista independente, seguindo-se as vistas dependentes. Estas não têm uma ordem pré-definida de codificação. Dentro de cada vista, as tramas de textura também são codificadas antes das tramas de profundidade.

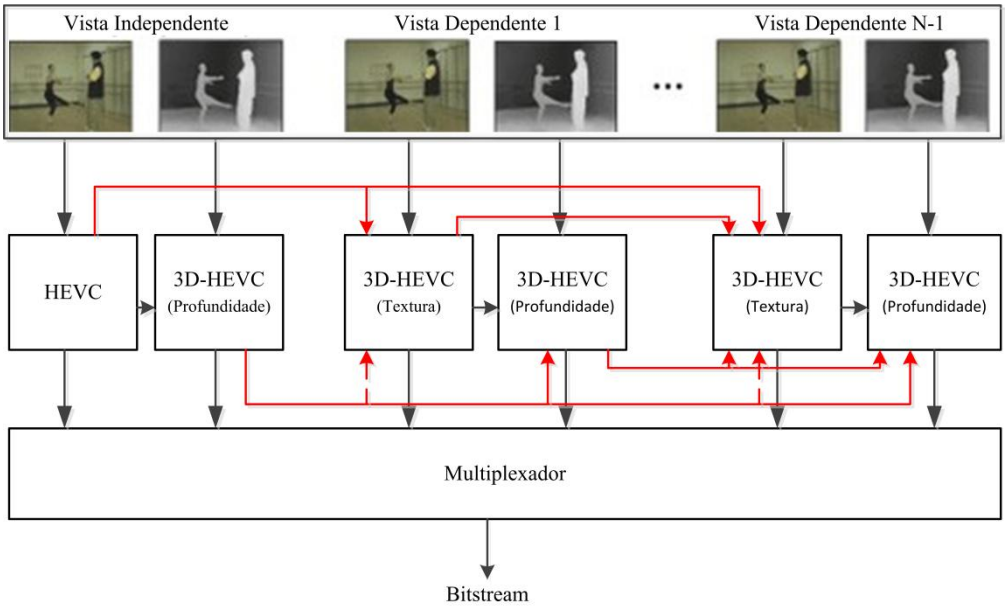


Fig. 2.9 - Estrutura de codificação do 3D-HEVC adaptada [12].

A quantidade de dados produzida pela codificação de vídeos 3D aumenta linearmente com o número de vistas dependentes a codificar. Para atenuar este problema foram desenvolvidos métodos como o *Depth Image Based Rendering* (DIBR) [13] que geram vistas intermédias renderizadas a partir das vistas codificadas. A Figura 2.10 ilustra um exemplo onde três vistas são codificadas e transmitidas. Após descodificação destas três vistas, usando o método DIBR são geradas e posteriormente exibidas mais sete vistas intermédias. Gerar artificialmente as vistas intermédias, invés de as codificar e transmitir, permite obter uma elevada poupança no *bitrate* gerado e reduzir a complexidade computacional associada ao processo de codificação.

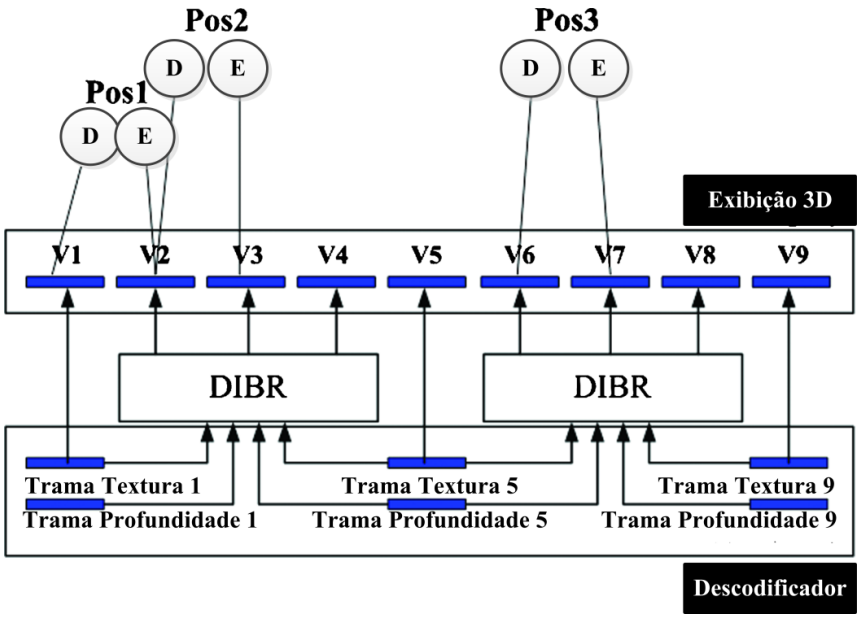


Fig. 2.10 - Vistas intermédias geradas através do DIBR; Pos: ponto de referência, D: Olho direito, E: Olho esquerdo, V:vista [13].

Um conjunto de novas funcionalidades foram adicionadas ao 3D-HEVC para suporte da codificação das vistas dependentes. Duas vistas vizinhas partilham conteúdo visual, visto que representam a mesma cena, capturada a partir de pontos diferentes, assim, é possível encontrar uma região localizada numa vista vizinha, do mesmo instante temporal, que partilhe o conteúdo visual com a região a codificar, através de um vector denominado de vetor de disparidade (Figura 2.11). A ferramenta *Disparity-Compensated Prediction* (DCP) [4] localiza na vista de referência a região com conteúdo visual similar à região a codificar e gera o respetivo vetor de disparidade, como ilustrado na Figura 2.11 e Figura 2.13.

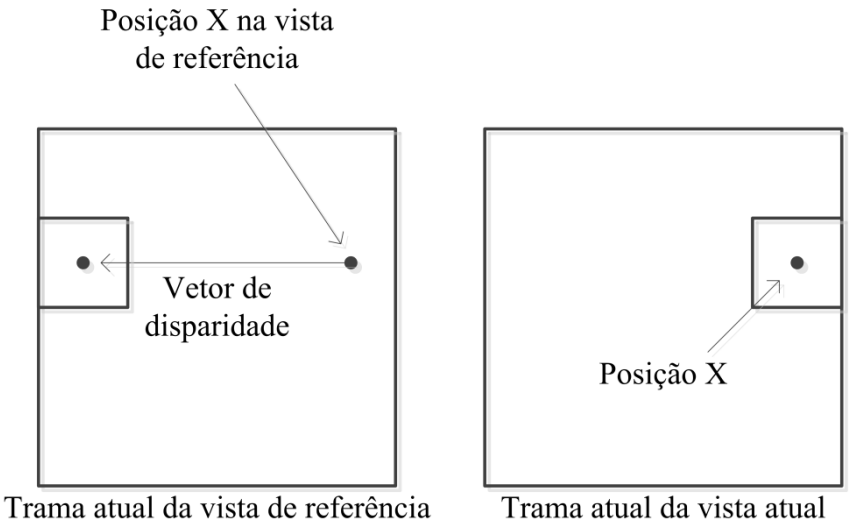


Fig. 2.11 - Vetor de disparidade.

A ferramenta denominada predição dos parâmetros do movimento entre vistas (*Inter-View Motion Parameter Prediction*-IVMPP) [4] estima os parâmetros de movimento da região a

codificar a partir de outra vista, do mesmo instante temporal, que já tenha sido codificada e transmitida. Esta predição é suportada pela premissa de que os parâmetros de movimento entre duas vistas vizinhas são similares. A Figura 2.12 ilustra o processo anteriormente referido. Com o objetivo de estimar os parâmetros de movimento para a região a codificar, vetores de disparidade são utilizados com a finalidade de encontrar a região equivalente na vista de referência. Se essa região tiver sido codificada usando *Motion Compensation Prediction* (MCP) os seus parâmetros de movimento podem ser utilizados na região a codificar. O método MCP é usado na predição *inter* entre tramas pertencentes à mesma vista, este usa vetores de movimento calculados durante o processo de estimação de movimento, com o objetivo de encontrar a região a codificar em tramas pertencentes à mesma vista mas de instantes temporais diferentes, Figura 2.13.

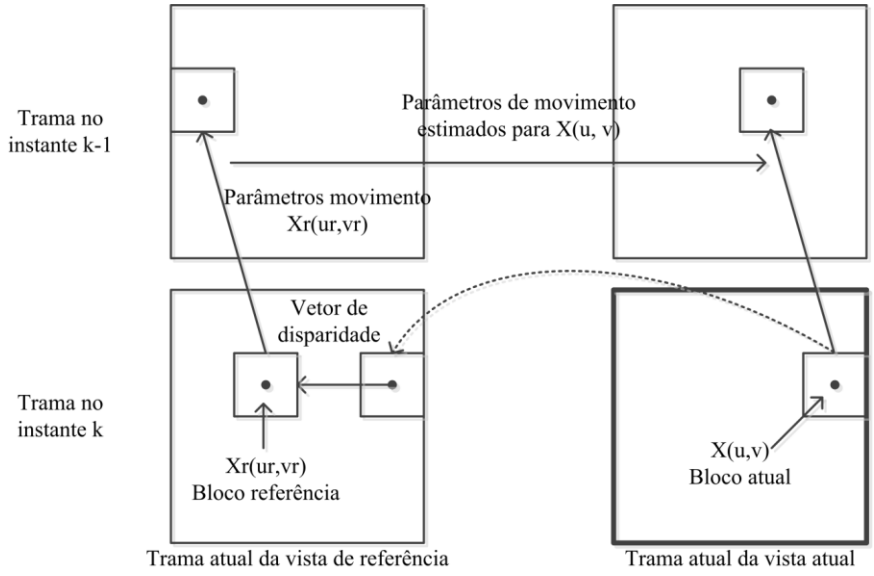


Fig. 2.12 - Ilustração do funcionamento da ferramenta IVMPP.

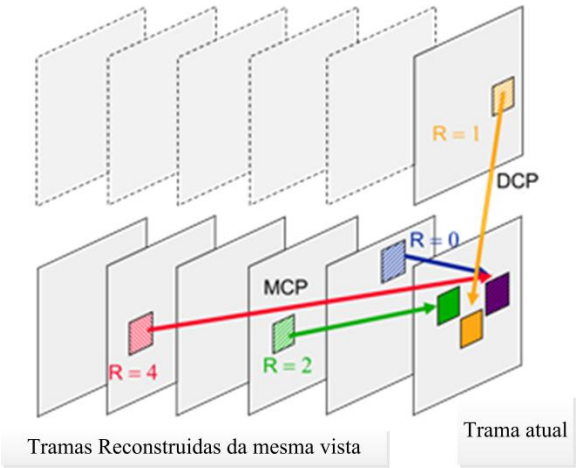


Fig. 2.13 - Ilustração do funcionamento das ferramentas MCP e DCP [12].

O método denominado Ajuste do QP de Textura Baseado em Dados de Profundidade (*Adjustment of QP of Texture on Depth Data)* [12] utiliza o conteúdo das tramas de profundidade para ajustar o parâmetro de quantização (*Quantization Parameter* - QP) usado nas tramas de textura. O QP é um parâmetro que define o intervalo de amostragem durante o processo de quantização, ou seja, a atribuição de valores discretos a partir de valores contínuos.

Durante a codificação de cada CU o valor do QP é ajustado relativamente à distância a que cada objeto se encontra da câmara, com o objetivo de melhorar a qualidade dos objetos que estão em primeiro plano (diminuindo o valor do QP) e de aumentar o fator de compressão dos objetos em segundo plano (incrementado o valor do QP). A Equação 2.4 [12] permite calcular o valor do QP ajustado. Nesta equação QP' é o QP ajustado para a CU com a correspondente disparidade do pixel em (x,y),

$$QP' = QP - 2.6 + 8 \cdot \left(\frac{255 - \max_{x,y \in CU} d_{x,y}}{256}\right)^2 \tag{2.4}$$

Os mapas de profundidade apresentam características particulares, tais como, arestas abruptas e longas regiões com valores aproximadamente constantes. Quatro novos modos de predição *intra* denominados modos de modelação de profundidade *intra* (*Intra Depth Modeling Modes*) foram adicionados ao 3D-HEVC, para permitir uma codificação eficaz do conteúdo 3D. Estes modos são usados nas tramas de profundidade com o intuito de preservar a informação sobre as arestas durante o processo de predição. Os quatro novos modos de predição constroem um modelo aproximado da região de profundidade a codificar dividindo-a em duas regiões não quadradas, cada uma representada por um valor constante. Esta divisão é realizada através de dois métodos distintos, denominados por *Wedgelets* e contornos (*Contours*), ilustrados nas Figuras 2.14 e 2.15 respetivamente. Analogamente aos restantes modos de predição *intra* anteriormente mencionados, um resíduo representando a diferença entre o modelo aproximado e a região a codificar é posteriormente transformado, quantificado, codificado e transmitido.

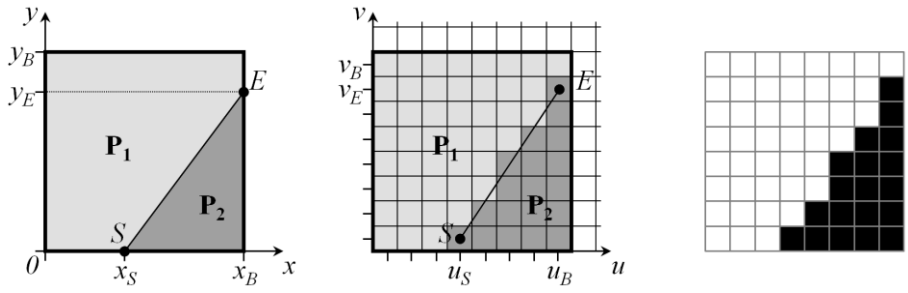


Fig. 2.14 - Wedgelets: sinal continuo (esquerda) e discreto (centro) e o respetivo padrão de divisão [12].

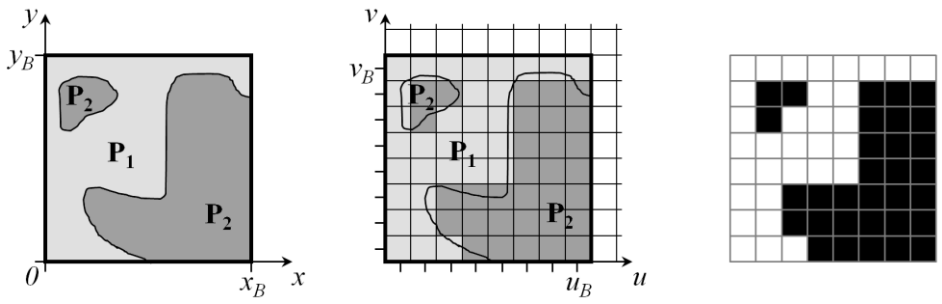


Fig. 2.15 - Contornos: sinal continuo (esquerda) e discreto (centro) e o respetivo padrão de divisão [12].

Os quatro novos modos de predição *intra* referidos; *Explicit Wedgelet signalling, Intra-predicted Wedgelet partitioning, Restricted signalling and inter-component prediction of Wedgelet partitions* e *Inter-component-predicted Contour partitioning* diferem no modo de divisão escolhido (contornos ou *Wedgelets*) e no modo como são transmitidos. Mais detalhes sobre cada um deles encontra-se em [12].

2.3 Medidas de Distorção

O processo de compressão utilizado na codificação de vídeo introduz perda de qualidade nas imagens reconstruídas. A quantificação objetiva dessas perdas não é fácil, visto que a perceção da distorção por parte de observadores humanos é difícil de medir e compreender. Atualmente, ainda não existem métodos de avaliação subjetiva suficientemente fiáveis, que permitam uma avaliação da distorção perceptualmente correta.

Peak sinal-noise ratio

Não obstante este problema, é comum usar a relação sinal-ruído de pico (PSNR) para avaliar a qualidade das tramas de um vídeo, após codificação seguida de descodificação. O PSNR mede o rácio entre o valor máximo do sinal e o ruído que o afeta. Este rácio é expresso em decibéis (dB) e calculado de acordo com a Equação 2.5, onde MAX representa o valor máximo que a grandeza característica de um *pixel* (usualmente luminância) pode tomar. Para uma representação com oito bits por *pixel* o valor máximo seria 255. Também na mesma equação *Mean Squared Error* (MSE) é o erro médio quadrático, ou seja, a média (abrangendo a imagem) dos quadrados das diferenças entre a imagem original e a imagem reconstruída. Este valor MSE é calculado através da Equação 2.6, onde *m* e *n* são as dimensões da imagem e R e O representam a imagem reconstruída e original. Usualmente, no contexto de atividades de codificação de vídeo o PSNR é calculado apenas para a componente de luminância (Y).

$$PSNR = 20\log_{10}\left(\frac{\mathrm{MAX}}{\sqrt{\mathrm{MSE}}}\right) \quad (2.5)$$

$$MSE = \frac{1}{\mathrm{m.n}}\sum_{\mathrm{i}=0}^{\mathrm{m}-1}\sum_{\mathrm{j}=0}^{\mathrm{n}-1}(\mathrm{R_{i,j}} - \mathrm{O_{i,j}})^2 \quad (2.6)$$

Delta de Bjøntegaard

Gisle Bjøntegaard propôs em [14] um modelo que compara a eficiência de compressão de dois algoritmos de codificação diferentes. Para cada algoritmo, em teste, a mesma sequência de vídeo é codificada com vários passos de quantização, originando um par (PSNR, *bitrate*) para cada passo de quantização e para cada algoritmo em comparação. Estes valores são usados pelo modelo para calcular o Bjøntegaard Delta PSNR (BD-PSNR) e o Bjøntegaard Delta RATE (BD-rate). O BD-PSNR é o aumento médio do PSNR de um algoritmo em relação a outro, considerando o mesmo *bitrate* e o BD-rate é o aumento médio do *bitrate* (%) considerando o mesmo PSNR. Algoritmos que apresentem BD-rate positivo e BD-PSNR negativo, quando comparados com o algoritmo de referência, possuem um pior desempenho.

No cálculo do BD-PSNR Gisle Bjøntegaard propõe que o ajuste da curva de *Rate-Distortion* (R-D), obtida através dos valores de PSNR e *Bitrate* (Figura 2.16) seja aproximada por uma função polinomial logarítmica de terceira ordem, como mostra a Equação 2.7, onde R é um valor de Bitrate, a,b,c e d são parâmetros de ajuste e D(r) é a distorção (PSNR).

$$D(r) = a.\log^3 R + b.\log^2 R + c.\log R + d \quad (2.7)$$

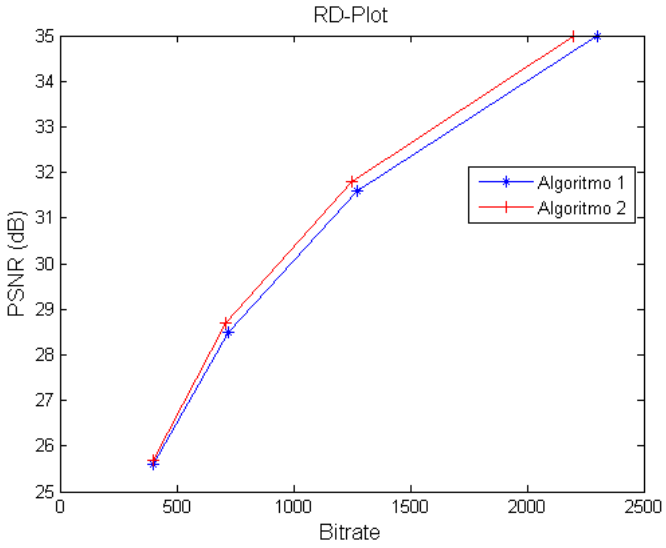


Fig. 2.16 - Curva de *Rate-Distortion* (R-D).

A média da diferença do PSNR entre as duas curvas R-D provenientes de cada um dos algoritmos pode ser aproximada pela diferença entre os integrais das curvas ajustadas a dividir

pelos intervalos de integração, Equação 2.8 onde ΔD é o BD-PSNR entre as duas curvas D2 e D1 e rh e rl são os limites de integração.

$$\Delta D = \frac{\int_{rl}^{rh}(D2-D1).dr}{Rh-Rl} \tag{2.8}$$

Para representar o *Bitrate* em função da distorção Gisle Bjøntegaard propõe que a função polinomial de terceira ordem representada na Equação 2.9 seja utilizada para ajustar a curva R-D.

$$r(D) = a.D^3 + b.D^2 + c.D + d \tag{2.9}$$

A média da diferença do *Bitrate* entre as duas curvas R-D é aproximada através da Equação 2.10, onde Δr é o BD-Rate entre as duas curvas r1(D) e r2(D) e Dh e Dl são os limites de integração.

$$\Delta R = 10^{\frac{1}{DH-DL}\int_{Dl}^{Dh}[r2(D)-r1(D)]dD}\text{-}1 \tag{2.10}$$

Estes métodos de quantificação do desempenho dos codificadores serão usados ao longo deste trabalho para comparar os vários algoritmos propostos.

2.4 Complexidade Computacional

Não existe uma definição única de complexidade computacional. Esta pode ser avaliada tendo em conta diferentes paramentos, tais como, o consumo energético, o número de instruções do processador ou o tempo de codificação. Neste trabalho, complexidade computacional é definida como o tempo gasto a codificar uma sequência de vídeo pelo codificador H.265/HEVC ou pelo codificador 3D-HEVC, consoante o contexto.

O H.265/HEVC permite reduzir até 50% de *bitrate*, comparativamente com o seu antecessor H.264/AVC. Esta redução é feita à custa de um grande aumento da complexidade computacional, entre 9% a 502 % [3]. Em boa parte este aumento é resultante do uso de estruturas de particionamento das tramas organizadas segundo árvores quaternárias, cuja estrutura ótima é necessário determinar, CTU a CTU, durante a codificação, com recurso a métodos de pesquisa exaustiva. Isto é, parte considerável desse tempo de codificação é gasto a testar todas a combinações possíveis de CUs, PUs e TUs em cada nível da árvore de codificação, de forma a obter a melhor eficiência de codificação para cada CTU.

O facto de a extensão 3D-HEVC servir normalmente para codificar vídeo com múltiplas vistas, introduz um fator adicional de aumento da complexidade computacional. As soluções

desenvolvidas neste trabalho para diminuir a complexidade computacional associada ao 3D-H.265/HEVC, visam acelerar o processo de escolha da divisão ideal das CUs e PUs, através do uso de métodos rápidos de decisão. Estes métodos usam informação de regiões de vistas vizinhas e/ou de regiões vizinhas dentro da própria trama, para tomar a sua decisão. As soluções propostas enquadram-se em dois grupos. No primeiro, procura-se controlar a complexidade computacional de forma a que esta não ultrapasse um valor máximo, normalmente expresso sob a forma de um tempo máximo de codificação por trama. No segundo grupo, procura-se apenas reduzir a complexidade computacional minimizando as perdas de eficiência da codificação efetuada com inclusão das heurísticas propostas.

Capítulo 3 – Estado da arte

Neste capítulo é apresentado o estado da arte relativo aos algoritmos de controlo e redução de complexidade dos codificadores H.265/HEVC e 3D-HEVC e uma descrição do algoritmo CCUPU [5]. O conjunto de algoritmos apresentados para reduzir a complexidade computacional destes codificadores permite obter uma visão global da heterogeneidade das soluções atualmente existentes. Algumas destas soluções obtiveram muito bons resultados em termos de redução de complexidade e eficiência de codificação, sendo que foram adicionadas ao *software* de referência que acompanha a norma H.265/HEVC e ao que acompanha a extensão 3D-HEVC, vulgarmente designados por HM [15] e HTM [16].

3.1 Estado da arte

H.265/HEVC

Os grupos de trabalho VCEG do ITU-T e MPEG da ISO/IEC estabeleceram um acordo de cooperação para desenvolver a atual geração de codificadores de vídeo. Desta cooperação surgiu o H.265/HEVC, cuja norma foi finalizada no início de 2013. Este *codec* atingiu grandes reduções de *bitrate*, comparativamente com os seus antecessores. Contudo, esta redução originou um grande aumento da complexidade computacional.

Vários investigadores deram o seu contributo no desenvolvimento de algoritmos capazes de atenuar o problema da complexidade computacional, alguns dos quais alcançaram grandes reduções de complexidade com perdas reduzidas ao nível da eficiência de codificação. Corrêa *et al.* [5] apresenta um algoritmo de controlo de complexidade computacional bastante eficaz. Este método ajusta dinamicamente a profundidade máxima da árvore de codificação das CUs e os modos de predição *inter* testados, para que o tempo de codificação não exceda um determinado valor ajustável. Em [17] é proposto um método em que para cada CU é avaliada a necessidade de esta continuar a ser dividida em CUs de menor dimensão, evitando a pesquisa exaustiva de todas as divisões possíveis. O método baseia-se no teorema de Bayes para tomar esta decisão. Yinbo Liu *et al.* [18] propõe um algoritmo que evita o teste de alguns modos de predição *intra*. Neste caso esta decisão é tomada tendo por base uma estimativa da complexidade das tramas de textura, sendo o método aplicável à codificação *intra*. Em JCTVC-F045 [19] é proposto um algoritmo que permite terminar antecipadamente a codificação da CU atual evitando que todos os modos de predição sejam testados. A decisão é tomada após a análise dos valores de

luminância e crominância para cada PU testada. Em [20] Zhaoqing Pan *et al.* propõe um método que escolhe antecipadamente o modo MERGE, evitando o teste de outros modos de predição. O modo MERGE [21] é um modo especial de predição *inter* usado em PUs de tamanho 2N×2N, este modo permite que os parâmetros de movimento sejam herdados a partir de PUs temporalmente e espacialmente vizinhas.

3D-HEVC

Recentemente, o grupo de trabalho composto pela VCEG e pela MPEG desenvolveram a extensão 3D-HEVC, no essencial uma versão 3D e multi-vista do H.265/HEVC. O aumento de complexidade representado pelos novos mecanismos adicionados ao H.265/HEVC para definição da extensão foi muito grande, o que motivou o desenvolvimento de várias soluções para redução da complexidade, publicadas em revistas científicas e atas de conferências recentes.

Em [22] os autores exploraram as relações de interdependência existentes entre a textura e a profundidade, para evitar o teste de todas as divisões possíveis durante a codificação das CUs das tramas de profundidade. As tramas de textura e profundidade da mesma vista encontram-se estreitamente ligadas, visto que representam a mesma cena, no instante temporal e a partir do mesmo ponto de referência. O estudo desta relação permitiu aos autores formularem três pressupostos.

O primeiro pressuposto, reivindica que **"uma CU de textura é pelo menos tão dividida como a CU de profundidade co-localizada"**. O uso deste pressuposto evita que para uma CU de profundidade, sejam testadas as CUs de dimensões inferiores à dimensão usada na CU de textura co-localizada. O segundo pressuposto que indica que **"uma CU de textura é pelo menos um nível menos dividida que a CU de profundidade co-localizada"**. Este pressuposto permite que o modelo de redução de complexidade computacional minore as perdas de eficiência na codificação, visto que permite a cada CU de profundidade ser dividida em CUs de dimensão inferior, relativamente ao pressuposto anterior. Uma abordagem mais agressiva em termos de redução computacional usa o terceiro pressuposto que menciona que **"a CU de profundidade é pelo menos um nível menos dividida que a CU de textura co-localizada** ".

Durante a codificação de uma CU de textura são testados oito partições da PU, com diferentes tamanhos para o caso de predição *inter* e duas partições da PU para o caso de predição *intra*. Em Zhang *et al.* [23], com o objetivo de evitar o teste de todos os modos de predição, propõe-se um método que antecipa a decisão do modo MERGE como o modo de predição ótimo para uma CU de textura da vista dependente. A antecipação desta decisão sem necessidade de testar os restantes modos de predição, permite uma grande poupança do tempo de codificação. O

algoritmo procura identificar duas condições; (i) **"todas as cinco CUs vizinhas da CU co-localizada da vista independente, após a respetiva compensação de disparidade, são codificadas com o modo MERGE"** e (ii) **"para a CU actual o modo de predição SKIP apresenta melhor performance que o modo de predição 2N×2N"**. Se estas condições forem verdadeiras o modo MERGE é identificado como ótimo e os restantes modos de predição não são testados. Também é proposto um segundo método que evita que a CU atual continue a ser dividida mediante a identificação de duas condições; (i) **"a divisão da CU atual é igual ou maior que a divisão máxima das cinco CUs vizinhas da CU co-localizada da vista independente"** e (ii) **"o Modo SKIP é selecionado como o melhor modo de predição para a CU atual"**. Se ambas as condições forem verdadeiras a codificação da CU atual termina sem que esta sofra mais divisões. O modo SKIP [21] é, tal como o modo MERGE, um modo especial de predição *inter* usado em PUs de tamanho 2N×2N. Este modo permite que o bloco seja codificado sem enviar o erro de predição e os parâmetros movimento. O modo SKIP é usado para codificar tramas ou regiões estáticas, onde o erro de predição é tendencialmente baixo.

O método proposto em [24], adequado ao uso com 3D-HEVC, é aplicado às vistas dependentes e evita o teste de todas as possíveis divisões de uma CU. Após a codificação da vista independente, a profundidade máxima da árvore de codificação de cada CU da trama atual dessa mesma vista é armazenada. Estas profundidades são usadas como estimativas iniciais para as CUs co-localizadas das vistas dependentes. Durante a codificação de qualquer vista dependente essas estimativas sofrem um refinamento e é gerada uma lista com a profundidade para cada CU. Cada CTU é dividida de acordo as profundidades presentes na referida lista.

O algoritmo apresentado em [25] foi desenvolvido especialmente para aplicações em tempo real do 3D-HEVC. Este algoritmo apresenta dois métodos que usam dados da codificação de CUs vizinhas para tomar decisões que aceleram o processo de codificação da CU atual. O primeiro método presente neste algoritmo deteta antecipadamente o modo de predição SKIP como sendo o modo ótimo, sem necessidade de testar a totalidade dos modos de predição. Inicialmente, antes de cada CU ser codificada é classificada como sendo do tipo I ou tipo II. Nas CUs do tipo I o modo SKIP é selecionado antecipadamente como sendo o ótimo e os restantes modos de predição (*intra* e *inter*) não são testados.

As CUs são classificadas como sendo do tipo I quando um dos seguintes conjuntos de CUs (Figura 3.1) é codificado com o modo SKIP; i) $\{P_s\ P_{dt}\}$, ii) $\{P_{ul}\ P_s\}$, iii) $\{P_s\ P_t\}$, iv) $\{P_{nv}\ P_{dt}\}$, v) $\{P_{nv}\ P_{ul}\}$, vi) $\{P_{ul}\ P_{dt}\}$, vii) $\{P_s\ P_t\}$ e viii) $\{P_{dt}\ P_t\}$. Onde P_s são as CUs vizinhas situadas; à esquerda, no topo e no topo-esquerdo, P_{dt} é a CU co-localizada da trama de textura (da mesma vista), P_{ul} é a CU pertencente ao nível de profundidade superior, P_{nv} é a CU co-localizada numa

vista vizinha após a respetiva compensação de disparidade e P_t são as CUs co-localizadas pertencentes a duas tramas já codificadas com ordem de reprodução superior e inferior à atual. Quando nenhuma das anteriores condições é satisfeita a CU é classificada como sendo do tipo II.

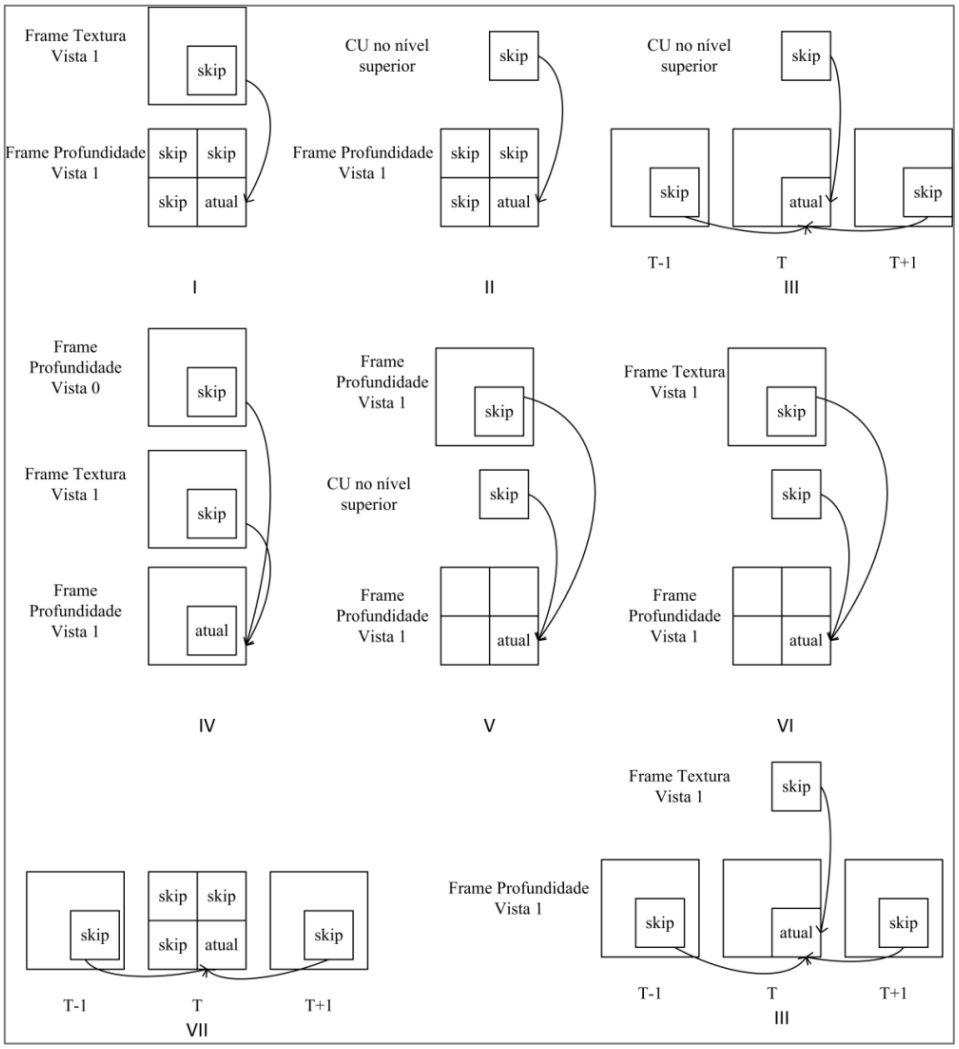


Fig. 3.1 - Condições para deteção do modo SKIP como ótimo [25].

O segundo método reduz o número de modos de predição testados durante a codificação de cada CU do tipo II. O algoritmo usa os modos de predição utilizados na codificação de CUs vizinhas para estimar as características de cada CU do tipo II através dos seguintes pressupostos; modos de predição *intra*/*inter* de tamanho 2N×2N são tendencialmente utilizados em regiões homogéneas, modos de predição *inter* de tamanho inferior são escolhidos para CUs com movimento e o modo de predição *intra* N×N é utilizado em regiões onde aparecem novos objetos ou de movimento complexo. Posteriormente as CUs do tipo II são agrupadas num dos seguintes modos: CUs simples, CUs complexas e CUs normais. Para uma CU simples são testados os modos *inter-2N×2N* e *intra-2N×2N*, para CUs complexas são testados os modos *inter-N×N*, *inter-2N×nU*, *inter-2N×nD*, *inter-nL×2N* e *inter-nR×2N*. No outro caso todos os modos são testados.

3.2 Constrained Coding Units and Prediction Units (CCUPU)

Neste subcapítulo é descrito o algoritmo *Constrained Coding Units and Prediction Units* (CCUPU). Inicialmente, este método foi desenvolvido para permitir o controlo de complexidade computacional do codificador H.265/HEVC [5], ou seja, dado um rácio de complexidade alvo o algoritmo aplica restrições ao nível da codificação das CTUs de cada trama, de modo a reduzir o tempo de codificação e garantir que a complexidade associada a esta codificação seja aproximadamente igual à do rácio.

O algoritmo recebe como parâmetros iniciais o tempo de codificação alvo por trama, designado por Tt, e o número de tramas por GOP (Nf). Os seus valores permitem calcular o tempo alvo por GOP (Tgop) através da Equação 3.1.

$$Tgop = Tt \times Nf \tag{3.1}$$

As tramas de cada GOP partilham as mesmas restrições, impostas com o objetivo de garantir que o tempo de codificação é próximo do tempo alvo. Após a codificação de cada GOP é calculado um rácio α que representa a razão entre o tempo alvo e o tempo efetivamente gasto na sua codificação. Esse rácio é calculado de acordo com a Equação 3.2, onde CTgop é o tempo de codificação do GOP precedente. Como se verá um pouco mais à frente, este rácio tem um papel importante no algoritmo de ajuste da complexidade.

$$\alpha = \frac{Tgop}{CTgop} \tag{3.2}$$

O algoritmo opera impondo restrições ao nível das CTUs, de forma que para cada trama é definido um número de CTUs cuja codificação é efetuada respeitando várias restrições aos modos de predição *inter* testados e ao número de divisões permitidas para cada CU. O algoritmo apresenta dois tipos destintos de restrições. No primeiro tipo, as CTUs são codificadas unicamente com PUs de tamanho 2N×2N, ou seja, os únicos modos de predição *inter* testados para cada CU são o SKIP, o MERGE e o 2N×2N. No segundo tipo de restrições, a profundidade máxima da árvore de codificação é dada pelo maior valor da profundidade máxima usada nas CTUs vizinhas situadas à esquerda, no topo, no topo-esquerdo, no topo-direito e na co-localizada da trama anteriormente codificada, como mostra a Figura 3.2. As restrições que limitam a profundidade máxima da árvore de codificação só são aplicadas quando se verifica a seguinte condição: **"todas as CTUs das tramas pertencentes ao GOP atual estão limitadas ao teste de PUs de tamanho 2N×2N"**.

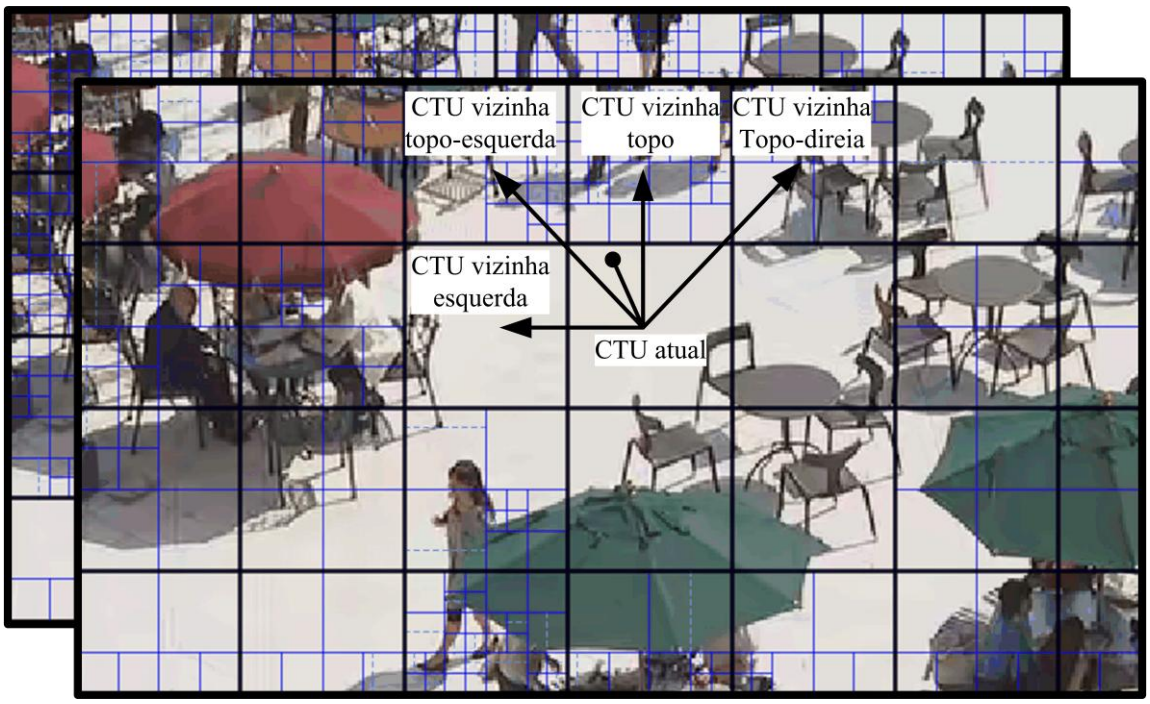


Fig. 3.2 - Esquema que limita a profundidade máxima da árvore de codificação da CTU atual [26].

O rácio α é usado como parâmetro de controlo para determinar o número de CTUs que sofrem restrições por trama (N_c^k), durante a codificação do próximo GOP. Esse valor é obtido através da multiplicação do rácio α pelo número de CTUs sujeitas a restrições no GOP anterior (3.3).

$$N_c^k \leftarrow \alpha . N_c^k \tag{3.3}$$

No parâmetro N_c^k k representa o tipo de restrição aplicada à codificação da CTU. Para k igual a um só são testadas PUs de tamanho 2N×2N e para k igual a dois é limitado o número de divisões permitidas para cada CU.

Através do processo anteriormente descrito é calculado para cada trama o número de CTUs que sofrem restrições. Quando esse número é inferior ao número total de CTUs por trama é gerada uma lista a indicar quais as CTUs que sofrem restrições.

Nessa lista as CTUs de cada trama são ordenadas por ordem crescente, tendo por base o custo RD da CTU co-localizada da última trama codificada. Isto é, as CTUs que têm menor custo RD são as primeiras a sofrer restrições, visto apresentam menor probabilidade de diminuir a eficiência de codificação quando sujeitas às restrições. Esta decisão é suportada pelo pressuposto de que existe uma forte relação entre o custo RD da CTU co-localizada na trama anterior e da CTU da trama atual. De forma contrária, as CTUs no fim da lista são as últimas a serem escolhidas, porque elas podem provocar um aumento significativo do *bitrate* e/ou da

distorção, que tem como consequência uma maior perda de eficiência de codificação. A Figura 3.3 representada o pseudo-código do algoritmo descrito anteriormente.

No próximo capítulo o presente algoritmo é adaptado ao 3D-HEVC, com o objetivo de impedir que a complexidade associada à codificação das tramas de textura da vista dependente, não ultrapasse um rácio de complexidade alvo.

```
01-Inicio
02-Iniciar um novo GOP
03-Para todas as tramas t do GOP i
04-     Para todas as CTUs j das tramas t
05-         Marcar CTU j como não restringida
06-         Codificar CTU j
07-         Se é a última trama da sequência de vídeo retorna à linha 1
08-Calcular Tgop
09-Calcular α
10-Se N_c^1 < Número de CTUs
11-     Calcular N_c^1
12-Se não
13-     Calcular N_c^2
14-Iniciar novo GOP
15-Para todas as tramas t do GOP i
16-     Ordenar por ordem crescente de custo RD
17-     Para todas as CTU j da trama t
18-         Se j < N_c^k
19-             Marcar CTU j como restringida
20-         Se não
21-             Marcar CTU j como não restringida
22-         Codificar CTU j
23-         Se é a última trama da sequência de vídeo retorna à linha 1
24-Retorna á linha 8
```

Fig. 3.3 - Pseudo-código do método CCUPU adaptado [5].

Capítulo 4 – Métodos de controlo de complexidade para o 3D-HEVC

Este capítulo apresenta a descrição dos algoritmos, das experiências, e das pesquisas que foram realizadas durante a dissertação em termos de controlo da complexidade computacional. Em suma, foram desenvolvidos dois algoritmos que têm como objetivo ajustar a complexidade computacional da extensão 3D-HEVC a um *target* alvo.

O primeiro algoritmo desenvolvido foi uma adaptação do método CCUPU, descrito no subcapítulo 3.2, à extensão 3D-HEVC. O CCUPU foi aplicado a esta extensão com o objetivo de manter a complexidade associada à codificação das tramas de textura, da vista dependente, abaixo de um rácio alvo. Após a implementação e análise dos resultados foi desenvolvida uma versão melhorada do mesmo algoritmo, com o propósito de aumentar o seu desempenho do ponto de vista da exatidão do controlo e o estender às tramas de profundidade da vista dependente. Para aumentar a exatidão com que tempo de codificação se mantém próximo do tempo alvo foi desenvolvido um sistema de controlo que atua antes de cada trama da vista dependente ser codificada. Este sistema baseia-se numa estimativa do tempo de codificação da próxima trama da vista dependente (textura ou profundidade) a ser codificada, para calcular as restrições necessárias para que a sua codificação demore um tempo próximo do tempo alvo. A estimativa do tempo de codificação de cada trama da vista dependente foi calculada beneficiando da forte correlação existente entre os tempos de codificação das tramas da vista dependente e independente.

4.1 CCUPU adaptado ao 3D-HEVC

A adaptação do algoritmo descrito anteriormente, ao 3D-HEVC tem como objetivo que a complexidade associada à codificação das tramas de textura da vista dependente fique abaixo de um rácio alvo pré-definido. Comparando com o algoritmo original as maiores mudanças residem nas restrições que limitam a profundidade máxima da árvore de codificação. Estas mudanças foram introduzidas para melhorar o desempenho do algoritmo, permitindo que este beneficiasse da forte correlação existente entre a vista dependente e independente.

Na extensão 3D-HEVC as tramas de textura e de profundidade, da vista independente, são as primeiras a ser codificadas. No algoritmo proposto, estas tramas são codificadas sem restrições, portanto segundo o procedimento seguido no modelo de referência.

O algoritmo usa o tempo alvo por GOP (Tgop) e o rácio α, calculado através da Equação 3.2, para determinar as restrições necessárias que garantem que o tempo de codificação do próximo GOP, que contém as tramas de textura da vista dependente, não ultrapasse o tempo alvo. O primeiro GOP é codificado sem restrições e o seu tempo de codificação é utilizado para calcular o rácio α e determinar as restrições a impor à codificação do próximo GOP.

Tal como na versão original, o algoritmo aplica dois tipos de restrições. No primeiro tipo, em cada CTU apenas são testados os modos de predição *inter*; SKIP, MERGE e 2N×2N. No segundo tipo, a profundidade máxima da árvore de codificação de cada CTU é limitada ao valor máximo entre: (i) a profundidade máxima da árvore de codificação das CTUs situadas à esquerda, no topo, no topo-esquerdo e no topo-direito da mesma trama. (ii) A profundidade máxima da árvore de codificação da CTU co-localizada na trama anteriormente codificada e (iii) a profundidade máxima da árvore de codificação da CTU co-localizada na trama de textura da vista independente. A informação de profundidade da árvore da CTU co-localizada na trama de textura da vista independente foi adicionada, em relação à versão original, para que o algoritmo possa beneficiar da redundância existente entre as vistas. Este processo é exemplificado na Figura 4.1.

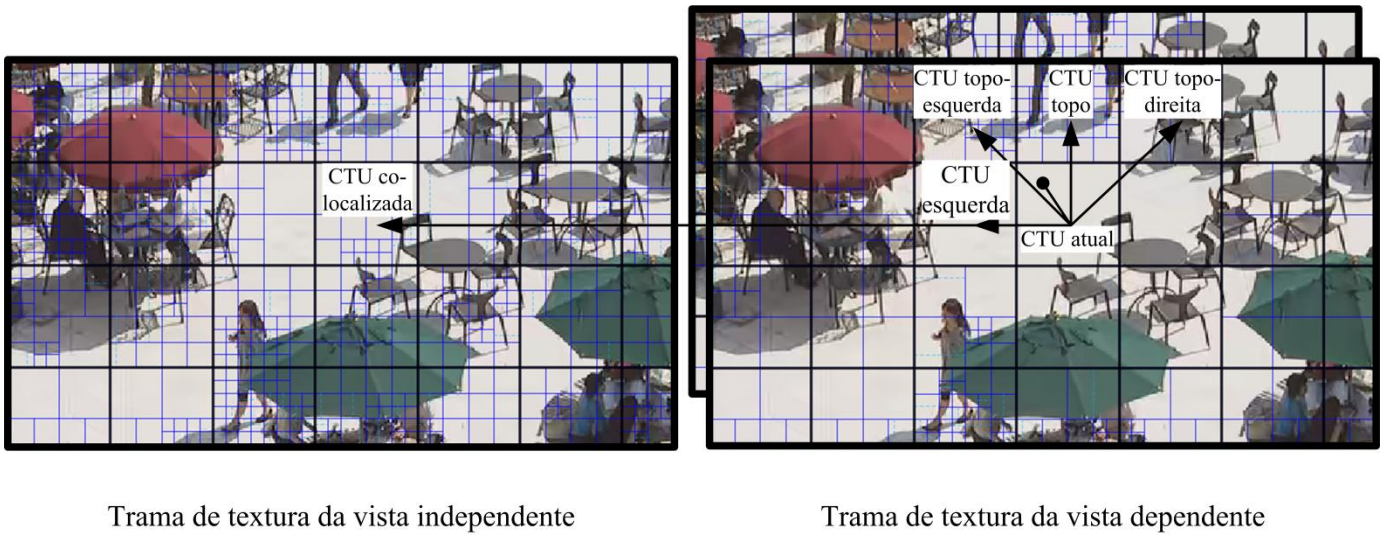


Fig. 4.1 - Esquema que limita a profundidade máxima da árvore de codificação da CTU actual.

Para limitar as perdas na eficiência de codificação, e analogamente ao algoritmo original, as CTUs encontram-se ordenadas por ordem crescente, tendo por base o custo RD das CTUs co-localizadas, da última trama codificada. Garantindo assim, que as CTUs que apresentam menor

probabilidade de diminuir a eficiência de codificação sejam as primeiras a sofrer restrições. Na Figura 4.2 e na Figura 4.3 é exibido o pseudo-código e o diagrama de fluxo do algoritmo proposto.

```
01-Inicio
02-Para todas as tramas de todas as vistas do instante t
03-     Codificar a trama de textura da vista independente
04-     Codificar a trama de profundidade da vista independente
05-     Se a trama de textura da vista dependente pertencer ao primeiro GOP da sequencia de vídeo
06-          Codificar trama de textura da vista dependente
07-          Se é a ultima trama do GOP
08-               Calcular α
09-               Calcular Tgop
10-     Se não
11-          Ordenar CTUs por ordem crescente de custo RD
12-          Para todas as CTU j da trama de textura da vista dependente
13-               Se j < N_c^k
14-                    Marcar CTU j como restringida
15-               Se não
16-                    Marcar CTU j como não restringida
17-               Codificar CTU j
18-          Se é a ultima trama do GOP
19-               Calcular alfa
20-               Calcular Tgop
21-               Se N_c^1 < Número de CTUs
22-                    Calcular N_c^1
23-               Se não
24-                    Calcular N_c^2
25-     Codificar a trama de profundidade da vista dependente
26-     Se é a última trama retorna a linha 1
```

Fig. 4.2 - Pseudo-código do algoritmo CCUPU adaptados ao 3D-HEVC.

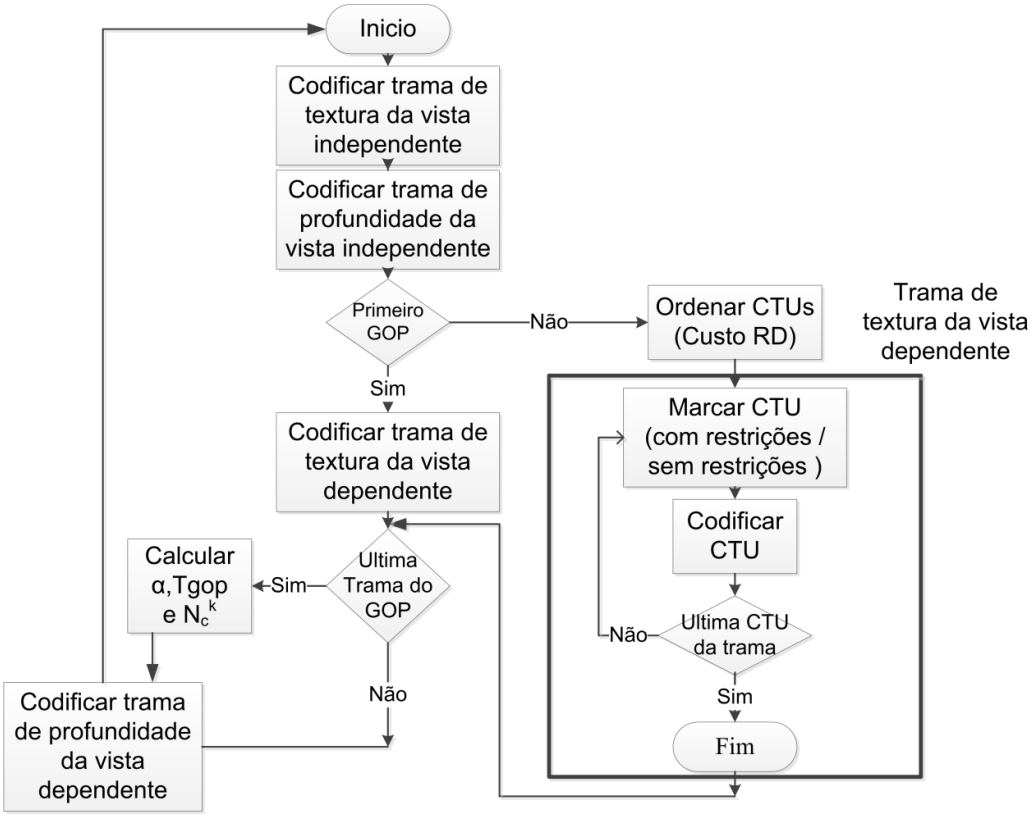


Fig. 4.3 - Diagrama de fluxo do algoritmo CCUPU adaptados ao 3D-HEVC.

Análise de desempenho do algoritmo CCUPU adaptado ao 3D-HEVC

O desempenho do algoritmo é medido através da sua capacidade de manter a complexidade computacional, associada à codificação das tramas de textura da vista dependente, abaixo do rácio de complexidade alvo e do efeito sobre a eficiência de codificação.

Para a realização dos testes foram utilizados dois conjuntos de sequências de vídeo. O primeiro conjunto é constituído por vídeos com resolução 1024×768 pixéis (*Balloons*, *Kendo* e *Newspaper*) e o segundo conjunto por sequências de vídeo com resolução 1920×1020 pixéis (*Dancer*, *Shark*, *PoznanHall2* e *GTFly*), Tabela 4.1. Os testes foram repetidos para quatro QPs diferentes (25, 30, 35 e 40), utilizando a versão HTM-11.0 do *software* de referência com a configuração *Random Access Encoder* (anexo D). Cada sequência de vídeo codificada contém duas vistas, uma independente e outra dependente, cada uma com tramas de textura e de profundidade. Os rácios alvo são 75%, 80%, 85%, 90% e 95% do tempo obtido por uma codificação sem restrições. Estes tempos de codificação foram medidos, através da função *clock()* da libraria *ctime*, com elevada precisão, uma vez que, as funções do sistema que o medem contabilizam somente o tempo em que o programa está a ser efetivamente processado. Os testes foram realizados num *cluster* baseado em *Intel Xeon E5520 (2,27 GHz)* com o sistema operativo *Windows Server 2008 HPC*.

Tabela 4.1 - Características das sequências de vídeo.

Sequências:	Resolução	Tramas	Frame Rate (HZ)	Formato de vídeo	Vistas
Balloons	1024×768	300	30	4:2:0	3-1
Kendo	1024×768	300	30	4:2:0	3-1
Newspaper	1024×768	300	30	4:2:0	4-2
GTFly	1920×1088	200	25	4:2:0	1-5
PoznanHall2	1920×1088	200	25	4:2:0	5-6
PoznanStreet	1920×1088	250	25	4:2:0	3-4
Dancer	1920×1088	250	25	4:2:0	5-1
Shark	1920×1088	300	25	4:2:0	5-1

A Tabela 4.2 mostra os resultados em termos de BD-rate e BD-PSNR das tramas de textura da vista dependente, para cada rácio alvo. Os valores BD-rate e BD-PSNR das tramas da vista independente não são tidos em consideração, visto que, esta é codificada de forma independente das restantes vistas, não sendo assim afetada pelo algoritmo proposto. Rácios alvo são calculados tendo como referência a codificação com o HTM 11.0 sem restrições. Na Figura 4.4 é ilustrada a diminuição do BD-PSNR em função dos rácios alvo. A exatidão com que o

algoritmo converge a complexidade da codificação para o rácio de complexidade alvo é apresentada na Figura 4.5.

O algoritmo consegue ajustar a complexidade da codificação das tramas de textura da vista dependente a rácios alvo superiores a 75%. Para rácio inferiores a este valor, a discrepância existente não é aceitável, assim, a discussão dos resultados terá como foco os rácios superiores a 75%. O algoritmo impõe elevadas perdas, a nível da eficiência de codificação. Estas perdas manifestam-se através de um grande aumento de BD-rate e de uma considerável descida do BD-PSNR, como mostra a Tabela 4.2.

Tabela 4.2 - Resultados do algoritmo proposto em termos de BD-rate e BD-PSNR.

Vídeos de resolução 1024×768 pixéis					
Rácio Alvo:	**75%**	**80%**	**85%**	**90%**	**95%**
BD-PSNR (Y)	-0.041 dB	-0.037 dB	-0.031 dB	-0.021 dB	-0.013 dB
BD-rate	1.157 %	1.046 %	0.866 %	0.565 %	0.341 %
Percentagem do Tempo Codificação	80.1 %	82.0 %	83.5 %	86.5 %	89.8 %
Vídeos de resolução 1920×1020 pixéis					
Rácio Alvo:	**75%**	**80%**	**85%**	**90%**	**95%**
BD-PSNR (Y)	-0.035 dB	-0.029 dB	-0.025 dB	-0.023 dB	-0.010 dB
BD-rate	1.548 %	1.298 %	1.114 %	1.047 %	0.443 %
Percentagem do Tempo Codificação	78.6 %	81.6 %	83.6 %	88.6 %	91.7 %

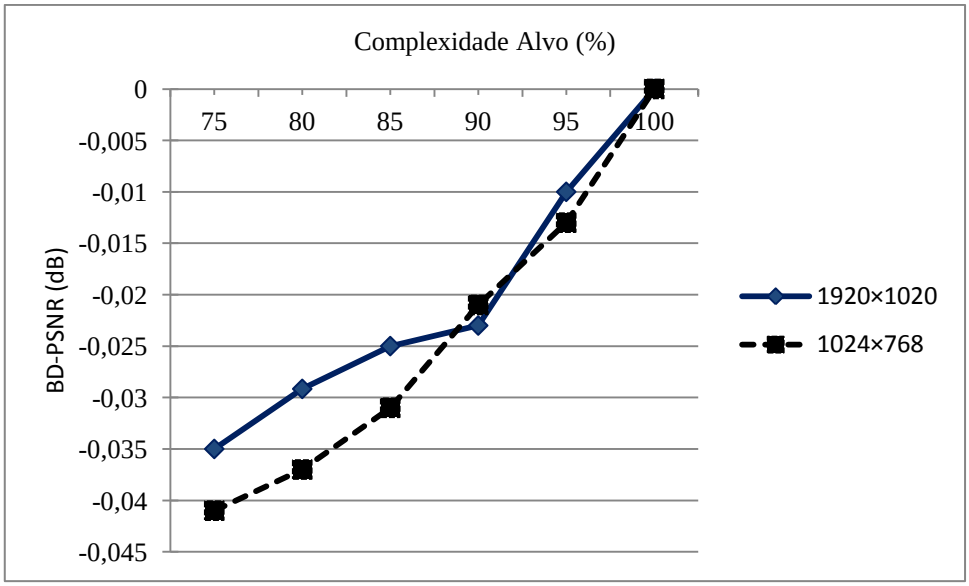


Fig. 4.4 - Resultados do algoritmo proposto em termos de BD-PSNR.

Os vídeos de resolução 1920×1020 pixéis apresentam tendencialmente melhores resultados em termos de redução de BD-PSNR e de exatidão do controlo, como mostram as Figuras 4.3 e 4.4. Contudo, o melhor desempenho em termos de exatidão de controlo, é acompanhado de um ligeiro aumento no valor do BD-rate, linha nove da Tabela 4.2 (***BD-rate***), comparativamente ao caso das sequências de menor resolução, linha quatro Tabela 4.2 (***BD-rate***). Este facto acontece porque os vídeos de resolução superior apresentam tipicamente mais detalhe a nível da textura, que os vídeos de menor resolução. O que provoca que durante a sua codificação as CUs sejam tendencialmente mais divididas e o algoritmo proposto, com o objetivo de atingir o rácio de complexidade alvo, aplique restrições mais severas o que origina uma maior perda da eficiência de codificação traduzida no aumento do BD-rate.

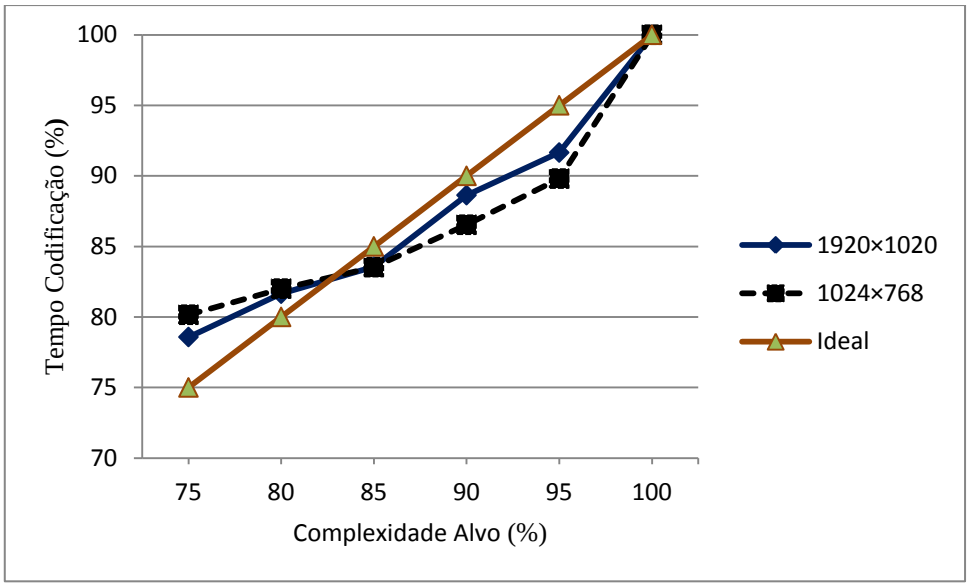


Fig. 4.5 - Resultados do algoritmo proposto em termos de exatidão de controlo.

Na Tabela 4.3 é possível comparar, em termos de BD-rate e BD-PSNR, o método CCUPU originalmente implementado no H.265/HEVC e o CCUPU adaptado ao 3D-HEVC onde ***Percentagem do Tempo de Codificação*** representa o valor percentual de tempo de codificação, relativamente a uma codificação sem restrições. Nesta tabela só estão representados dois rácios alvo, contudo é possível verificar que o algoritmo original, implementado no H.265/HEVC, consegue ajustar a complexidade computacional da codificação aos rácios alvo impondo perdas negligenciáveis a nível da eficiência de codificação.

Os resultados obtidos pelo algoritmo adaptado ao 3D-HEVC ficam muito distantes dos resultados obtidos pelo algoritmo original. Ambos os algoritmos apresentam valores similares a nível da exactidão do controlo da complexidade, contudo, existe uma discrepância muito grande nos valores de BD-PSNR e BD-rate. Este facto é originado pela adição ao *software* de referência de diversos métodos de decisão rápida usados na codificação da vista dependente, que limitam os

ganhos da abordagem aqui proposta. Estes métodos baseiam-se na exploração das redundâncias existentes entre vistas e entre as tramas de textura e de profundidade, para acelerar o processo de codificação das CUs.

Tabela 4.3 - Comparação de resultados do CCUPU implementado no H.265/HEVC e no 3D-HEVC.

Rácio Alvo:	CCUPU 3D-HEVC			CCUPU H.265/HEVC		
	Percentagem do Tempo de Codificação	**BD-rate**	**BD-PSNR**	**Percentagem do Tempo de Codificação**	**BD-rate**	**BD-PSNR**
90%	89%	0.81%	0,02dB	93%	0,03%	0,00dB
80%	82%	1,17%	0,03dB	83%	0,23%	-0,01dB

Em suma, o algoritmo permite que a complexidade associada à codificação das tramas de textura da vista dependente, não ultrapasse um rácio alvo maior ou igual a 75% do tempo do tempo obtido durante a codificação sem restrições. Esta redução da complexidade computacional é obtida impondo uma perda de eficiência média na codificação, traduzida numa diminuição máxima de -0,041dB do BD-PSNR e um aumento de 1,570 % do BD-rate. Em termos de exatidão do algoritmo de controlo o desvio, em módulo, entre a complexidade da codificação das tramas de textura da vista dependente e o rácio de complexidade alvo (75%, 80%, 85%, 90% e 95%), foi em média de 3,2406%.

4.2 CCUPU com sistema de controlo preditivo

Neste subcapítulo o algoritmo anterior foi estendido às tramas de profundidade da vista dependente. Contudo, antes de avançar com a implementação foi realizada uma pesquisa com o objetivo de desenvolver um novo sistema de controlo, que permita aumentar a exatidão com que o algoritmo, descrito no subcapítulo 4.1, ajusta a complexidade computacional a um rácio alvo.

Contrariamente ao controlo GOP a GOP, realizado pelo algoritmo descrito no subcapítulo anterior, o novo sistema de controlo atua antes de cada trama da vista dependente ser codificada. O controlador estima o tempo de codificação da trama e calcula as restrições necessárias para que este não ultrapasse o tempo alvo.

Para desenvolver o novo controlador foi estudada a relação entre o tempo de codificação das tramas da vista dependente e o tempo de codificação das tramas da vista independente. Os seus tempos de codificação apresentam uma alta correlação. Esta correlação é especialmente evidente quando a trama da vista independente apresenta um aumento do seu tempo de codificação, em relação à trama da mesma vista codificada anteriormente, uma vez que, esta subida do tempo de codificação é igualmente verificada na trama da vista dependente. A correlação existente entre as tramas das duas vistas pode ser usada para antecipar a necessidade de impor restrições a uma trama da vista dependente, antes de esta ser codificada.

Por outro lado, variações negativas dos tempos de codificação de duas tramas, da vista independente, codificadas sequencialmente não implicam que a mesma tendência seja seguida pelas tramas da vista dependente. Este facto acontece porque a vista independente tem tramas do tipo I. As tramas I só usam predição *intra* o que reduz a complexidade computacional associada ao seu processo de codificação, visto que, as operações realizadas na predição *inter* tal como; a estimação de movimento e compensação de movimento são algumas das operações que introduzem mais complexidade computacional ao processo de codificação.

As Figuras 4.6 e 4.7 mostram o tempo de codificação das tramas de textura da vista independente e da vista dependente, das sequências de vídeo *Balloons* e *Shark*. Na Tabela 4.4 encontra-se descriminado os QP usados, o número de tramas, as vistas usadas e os coeficientes de correlação entre o tempo de codificação da vista dependente e da vista independente, obtidos através da Equação 4.1. Estes coeficientes permitem medir o nível de correlação entre os dois tempos de codificação e assim avaliar se um pode ser estimado a partir do outro.

$$\rho_{Ti,Td} = \frac{Cov(Ti,Td)}{\sqrt{Var(Ti).Var(Td)}} \tag{4.1}$$

Na Equação 4.1 $\rho_{Ti,Td}$ representa o coeficiente de correlação entre o tempo de codificação das tramas de textura da vista independente e dependente. Ti_1, Ti_2, ..., Ti_n são os valores dos tempos de codificação das tramas de textura da vista independente, Td_1, Td_2, ..., Td_n são os valores dos tempos de codificação das tramas de textura da vista dependente, Cov é a covariância e Var é a variância. Na Tabela 4.4 são ainda descriminados os coeficientes de correlação entre as diferenças do tempo de codificação das tramas nos instantes k e k-1 da vista independente e as diferenças do tempo de codificação das tramas nos instantes k e k-1 da vista dependente.

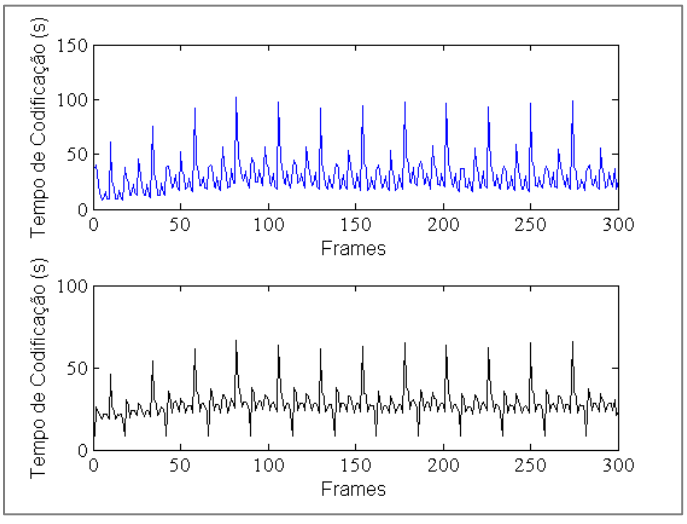


Fig. 4.6 - Tempo de codificação das tramas de textura da vista dependente (sub-figura superior) e da vista independente (sub-figura inferior) da sequência Balloons (QP 25).

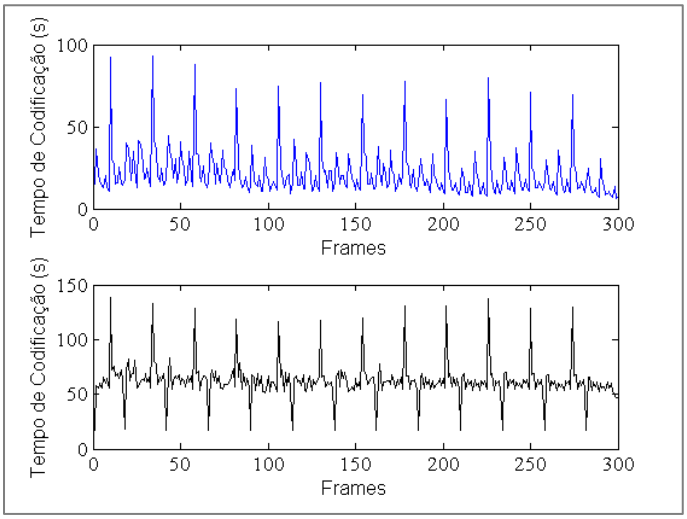


Fig. 4.7 - Tempo de codificação das tramas de textura da vista dependente (sub-figura superior) e da vista independente (sub-figura inferior) da sequência Shark (QP 40).

Tabela 4.4 - Tabela com detalhes sobre as sequências teste usadas e os valores dos coeficientes de correlação.

Sequências:	**QP**	**Tramas**	**Vista Independente**	**Vista Dependente**	**Coeficientes de Correlação dos Tempos de Codificação**	**Coeficientes de Correlação das Variações dos Tempos Codificação**
Balloons	25	300	Vista 3	Vista 1	0.79	0.76
PoznanHall2	30	200	Vista 6	Vista 7	0.50	0.55
GT-Fly	35	250	Vista 5	Vista 9	0.84	0.84
Shark	40	300	Vista 5	Vista 1	0.73	0.76

Os valores dos coeficientes de correlação presentes na Tabela 4.4 indicam que a correlação entre o tempo de codificação da vista dependente e da vista independente é moderadamente elevada, por isso é de esperar que seja possível estimar um dos tempos a partir do outro, com alguma precisão.

Para o cálculo do valor da estimativa é assumido que a variação do tempo de codificação entre as tramas, da vista independente, pertencentes aos instantes k e k-1 é refletida no tempo de codificação da trama da vista dependente, do instante *k.* Esta suposição é suportada pelos valores moderadamente elevados dos coeficientes de correlação entre a variação do tempo de codificação das tramas nos instantes k e k-1 na vista independente e na vista dependente. Contudo, e como é verificável através do conteúdo visual das Figuras 4.6 e 4.7, esta suposição é menos válida quando existe uma variação negativa do tempo de codificação. Para tornar a estimativa mais fiável, a lidar com as diminuições abruptas do tempo de codificação da vista independente, foi adicionada uma condição que limita as variações negativas do tempo de codificação de uma trama da vista dependente. Esta condição impede que a estimativa do tempo

de codificação de uma trama da vista dependente (textura ou profundidade) para o instante k possa ser inferior ao resultado da diferença entre a média do tempo de codificação das tramas precedentes e o seu desvio padrão. Deste modo, a estimativa para cada instante é obtida através da Equação 4.2 e posteriormente é aplicada a condição 4.3, quando esta condição se verifica a estimativa é recalculada através da Equação 4.4.

$$\hat{Y}_{Dep}(k) = \hat{Y}_{Dep}(k-1) + \left(Y_{Ind}(k) - Y_{Ind}(k-1)\right) - E(k-1) \tag{4.2}$$

Na equação 4.2, $\hat{Y}_{Dep}(k-1)$ representa a estimativa do tempo de codificação de uma trama da vista dependente no instante k-1. $Y_{Ind}(k) - Y_{Ind}(k-1)$ é a variação do tempo de codificação, entre as tramas da vista independente dos instantes k-1 e k e E(k-1) é o erro da estimativa no instante k-1.

$$\text{if}\,(\hat{Y}_{Dep}(k) \leq \theta_{Dep} - \sqrt{\hat{V}ar_{Dep}}) \tag{4.3}$$

$$\hat{Y}_{Dep}(k) = \theta_{Dep} - \sqrt{\hat{V}ar_{Dep}} \tag{4.4}$$

Na condição 4.3 e na Equação 4.4 θ_{Dep} é a média do tempo de codificação da trama (textura ou profundidade) da vista dependente e $\sqrt{\hat{V}ar_{Dep}}$ é a estimativa do respetivo desvio padrão.

Com o objetivo de averiguar a fiabilidade das estimativas geradas pelo método descrito anteriormente foram estimados os tempos de codificação de três sequências de vídeo; Kendo, Balloons e PoznanHall2. A Tabela 4.5 apresenta os detalhes sobre as sequências de teste usadas, os seus tempos de codificação, a média e a variância do erro normalizado e os coeficientes de correlação. O erro normalizado é obtido através da Equação 4.5, em que TEst representa o tempo de codificação estimado e TReal o tempo de codificação observado.

$$Erro_n = \frac{(TEst\text{-}TReal)}{TReal} \tag{4.5}$$

Nas Figuras 4.8 a 4.13 são representados os tempos de codificação das tramas de textura e de profundidade, os tempos de codificação estimados e o erro normalizado.

Os valores dos coeficientes de correlação presentes na Tabela 4.5 demonstram que existe uma correlação moderadamente elevada entre o tempo de codificação das tramas da vista dependente (textura e profundidade) e os tempos estimados, consequentemente, estes são considerados estimativas válidas dos tempos de codificação e o seu valor é utilizado para

antecipar o cálculo das restrições a impor às tramas da vista dependente, antes de estas serem codificadas.

Tabela 4.5 - Tabela com os detalhes sobre as sequências de teste usadas e os tempos de codificação.

Vídeo:	**QP**	**Tipo**	**Tramas**	**Vista Dependente**	**Média do Erro**	**Variância do Erro**	**Coeficientes de Correlação**
Kendo	25	Textura	300	Vista 1	0.0927	0.0929	0.64
		Profundidade		Vista 1	0.0797	0.2277	0.48
Balloons	30	Textura	300	Vista 1	0.0998	0.1433	0.64
		Profundidade		Vista 1	0.0846	0.4444	0.81
PoznanHall2	40	Textura	200	Vista 6	0.1707	0.2808	0.64
		Profundidade		Vista 6	0.1838	0.1838	0.82

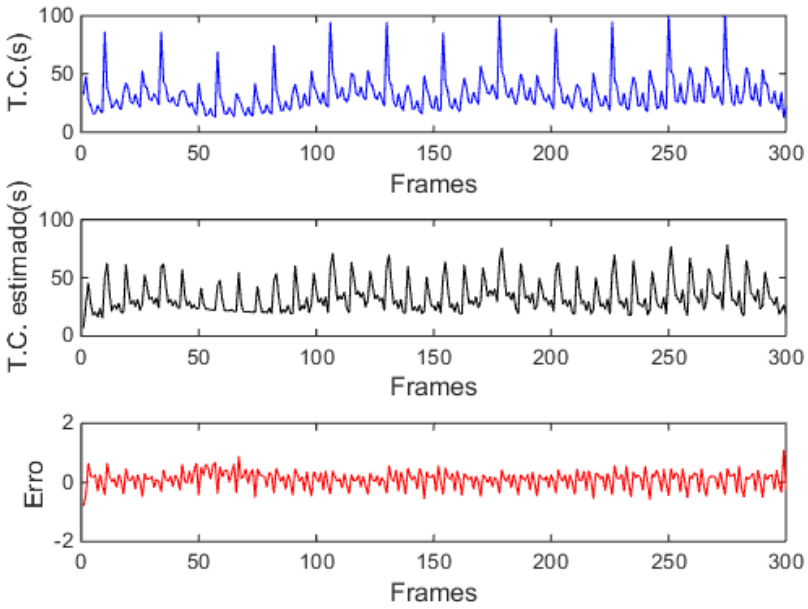


Fig. 4.8 - Tempo de codificação das tramas de textura da vista dependente, tempo estimado e o erro normalizado para a sequência de teste Kendo (QP 25).

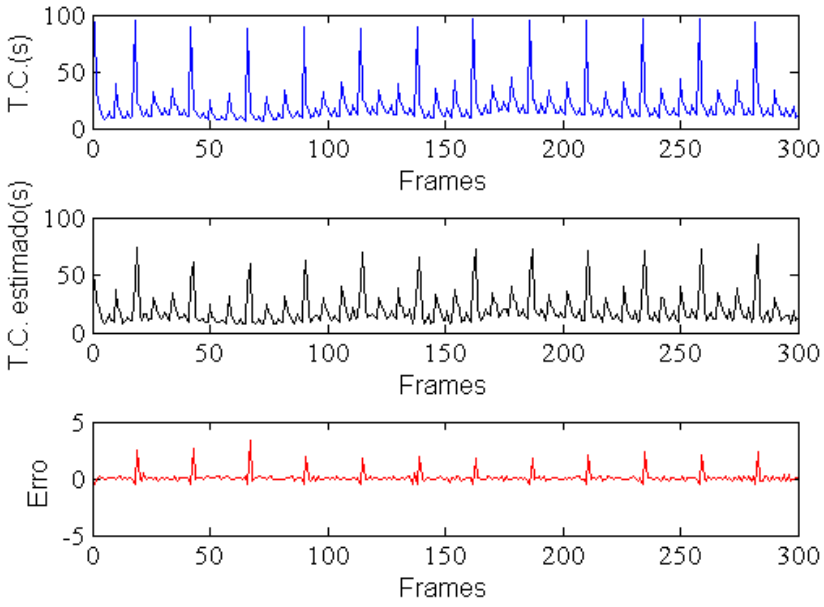


Fig. 4.9 - Tempo de codificação das tramas de profundidade da vista dependente, tempo estimado e o erro normalizado para a sequência de teste Kendo (QP 25).

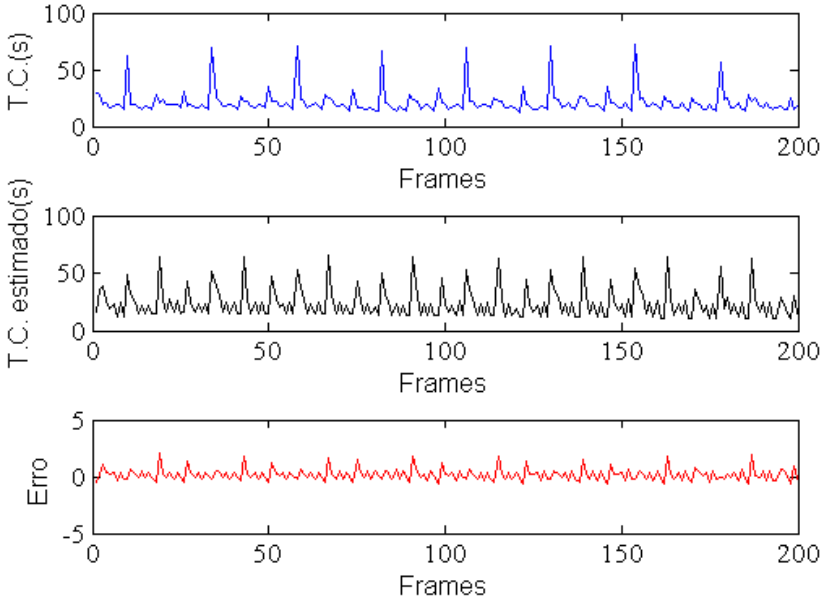


Fig. 4.10 - Tempo de codificação das tramas de textura da vista dependente, tempo estimado e o erro normalizado para a sequência de teste PoznanHall2 (QP 40).

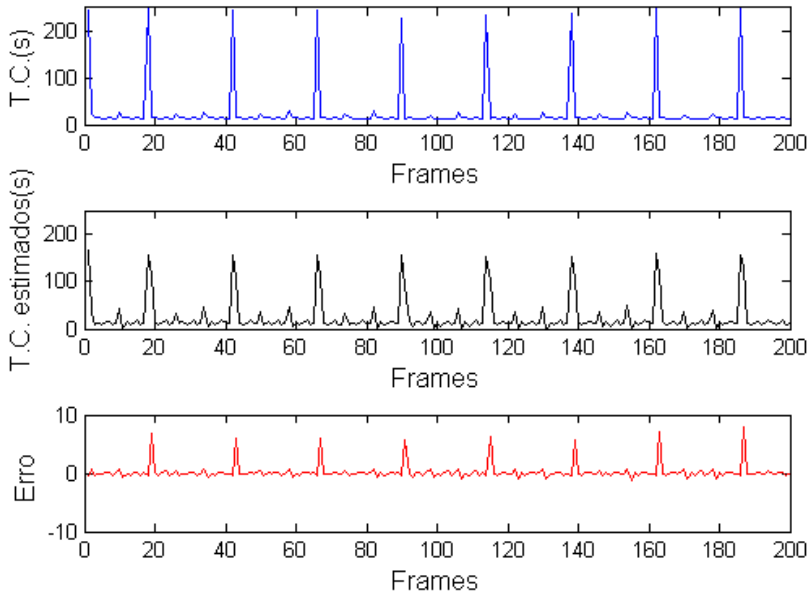


Fig. 4.11 - Tempo de codificação das tramas de profundidade da vista dependente, tempo estimado e o erro normalizado para a sequência de teste PoznanHall2 (QP 40).

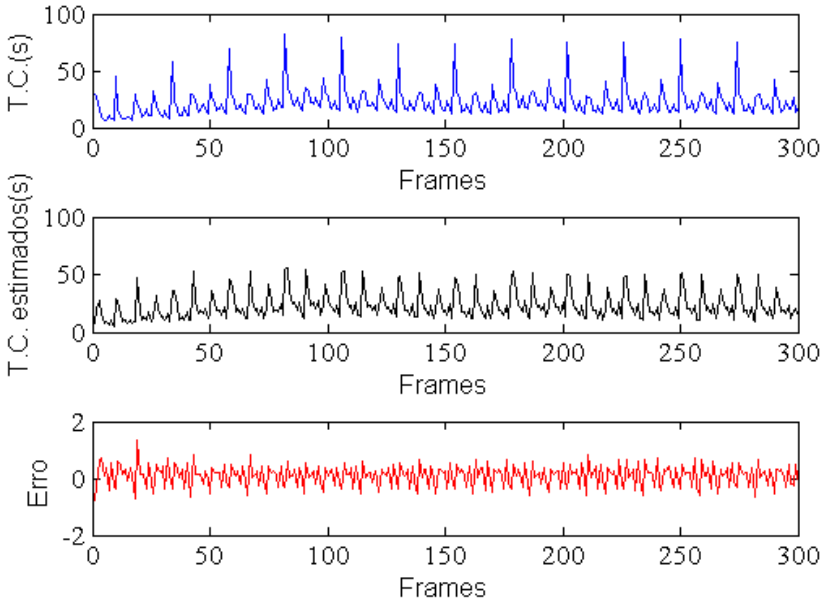


Fig. 4.12 - Tempo de codificação das tramas de textura da vista dependente, tempo estimado e o erro normalizado para a sequência de teste Balloons (QP 30).

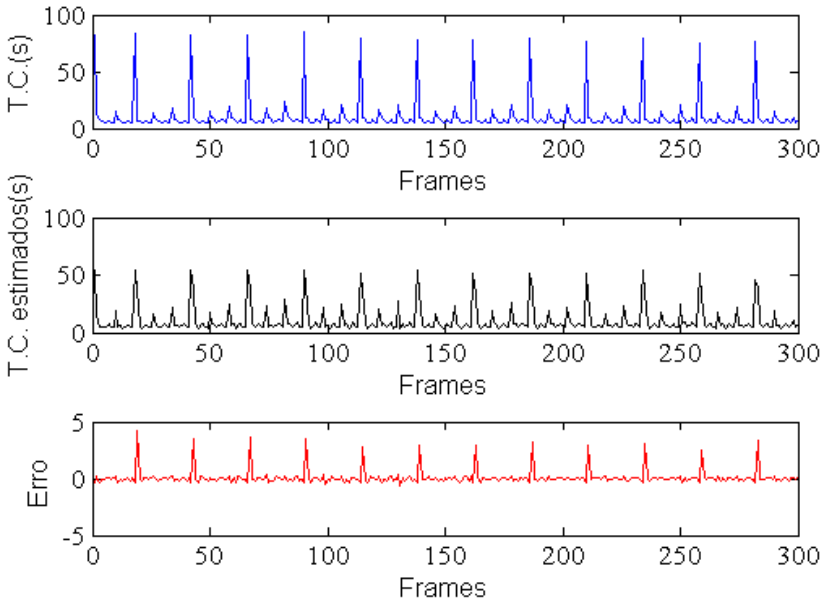


Fig. 4.13 - Tempo de codificação das tramas de profundidade da vista dependente, tempo estimado e o erro normalizado para a sequência de teste Balloons (QP 30).

Os parágrafos seguintes apresentam a descrição do algoritmo, descrito no subcapítulo 4.1, após a adição do novo controlador e da sua extensão às tramas de profundidade da vista dependente. Na Figura 4.14 e Figura 4.15 encontra-se descrito o pseudo-código e representado o diagrama de fluxo do algoritmo proposto.

O algoritmo codifica as tramas de textura e profundidade, da vista independente, sem restrições. Após essa codificação são calculadas as estimativas dos tempos de codificação das tramas de textura e profundidade da vista dependente, através das Equações 4.2 e 4.4 e da condição 4.3.

Os valores estimados são utilizados para estimar a média do tempo de codificação da vista dependente (tramas de textura e de profundidade) que permite o cálculo do rácio α, obtido através da equação 4.5. Nesta equação $\hat{\theta}$Profundidade e $\hat{\theta}$Textura representam a estimativa da média do tempo de codificação das tramas de profundidade e de textura da vista dependente. θ_{Target} representa a média do tempo de codificação alvo, das tramas da vista dependente.

$$\alpha = \frac{\hat{\theta}_{Profundidade} + \hat{\theta}_{Textura}}{\theta_{Target}} \tag{4.5}$$

O rácio α permite calcular, de forma análoga ao algoritmo do subcapítulo 4.1, o número de CTUs por trama, sobre o qual vão ser aplicadas restrições. Devido ao melhor desempenho por parte das CTUs de profundidade, em termos de eficiência de codificação, estas são as primeiras a

sofrer restrições. Quando as restrições aplicadas, nas tramas de profundidade, são insuficientes para convergir a complexidade computacional a um rácio alvo, estas são estendidas às CTUs das tramas de textura.

Em seguida as tramas da vista dependente são codificadas. A primeira trama de textura e profundidade da sequência vídeo são codificadas sem qualquer restrição. As restantes são codificadas de forma similar ao do algoritmo descrito no subcapítulo 4.1, isto é, para cada trama da vista dependente é gerada uma lista com a ordem de seleção das CTUs, sujeitas a restrições. Esta lista é usada para selecionar as CTUs com restrições e este processo é aplicado para todas as CTUs das tramas de textura e de profundidade.

```
01-Inicio
02-Para todas as tramas do instante t
03-     Codificar a trama de textura da vista independente
04-     Codificar a trama de profundidade da vista independente
05-     Estimar tempo de codificação das tramas da vista dependente
06-     Calcular α
07-     Calcular restrições da trama de Profundidade da vista dependente
08-     Se N_c^1 < Número de CTUs
09-             Calcular N_c^1
10-     Se não
11-             Calcular N_c^2
12-     Calcular restrições da trama de textura da vista dependente
13-     Se N_c^1 < Número de CTUs
14-             Calcular N_c^1
15-     Se não
16-             Calcular N_c^2
17      Se é a primeira trama da sequencia de vídeo
18-             Codificar trama de textura e profundidade da vista dependente sem restrições
19-     Se não
20-             Ordenar por ordem crescente de custo RD
21-             Para todas as CTU j da trama de textura da vista dependente
22-                     Se j< N_c^k
23-                             Marcar CTU j como restringida
24-                     Se não
25-                             Marcar CTU j como não restringida
26-                     Codificar CTU j
27-             Para todas as CTU j da trama de profundidade da vista dependente
28-                     Se j < N_c^k
29-                             Marcar CTU j como restringida
30-                     Se não
31-                             Marcar CTU j como não restringida
32-                     Codificar CTU j
```

Fig. 4.14 - Pseudo-código do algoritmo CCUPU com sistema de controlo preditivo.

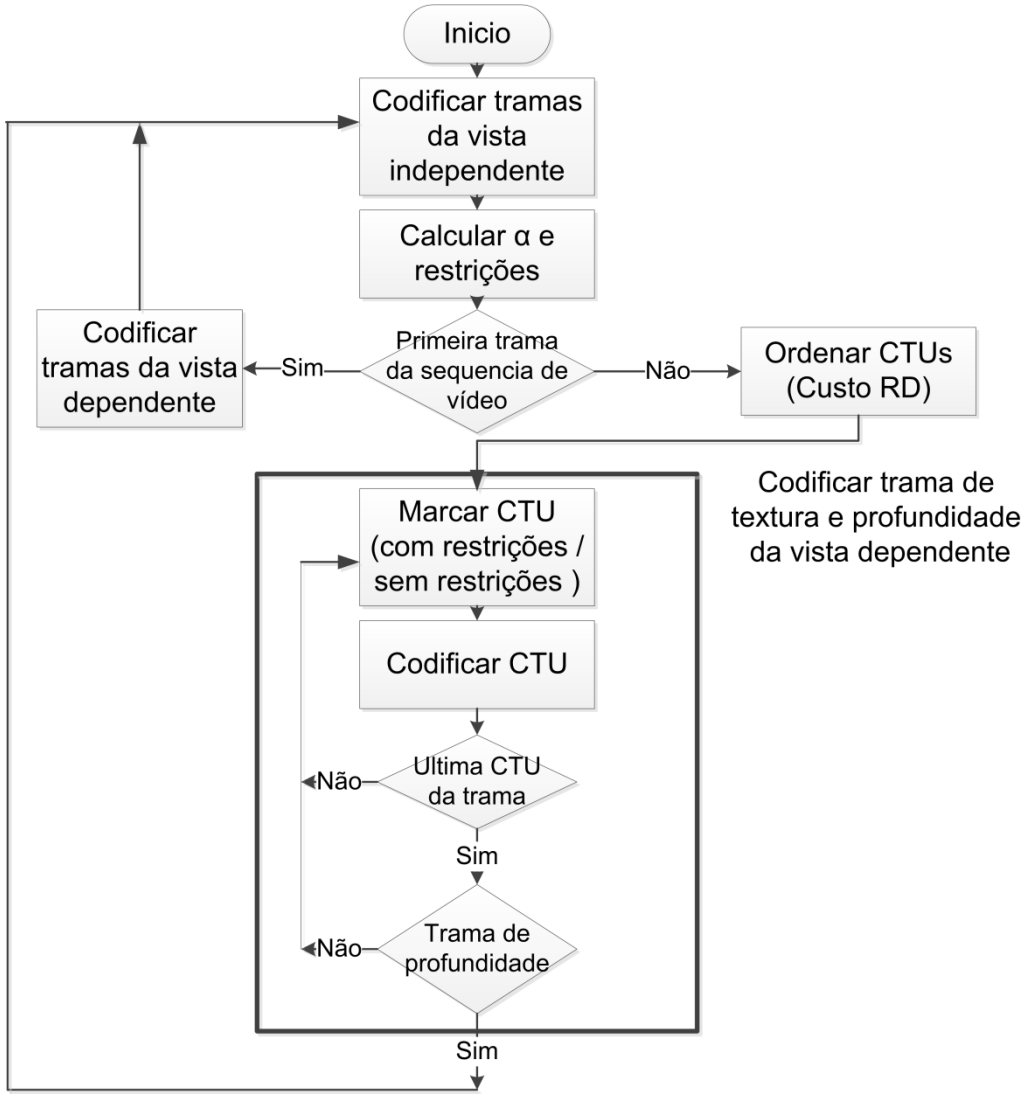


Fig. 4.15 - Diagrama de fluxo do algoritmo CCUPU com sistema de controlo preditivo.

Análise de desempenho do algoritmo CCUPU com sistema de controlo preditivo

As condições de teste usadas para avaliar o desempenho do presente algoritmo são similares às do subcapítulo 4.1.

As Tabelas 4.6 e 4.7 mostram os resultados em termos de BD-rate e BD-PSNR, das tramas de textura e de profundidade, da vista dependente. A exatidão do algoritmo de controlo é apresentada na Figura 4.16. As Figuras 4.17 e 4.18 ilustram os valores de BD-PSNR, em função dos rácios de complexidade alvo, para as tramas de textura e profundidade respetivamente.

O algoritmo consegue ajustar a complexidade de codificação da vista dependente aos rácios de complexidade alvo superiores a 75%. Para rácios inferiores a este valor, a discrepância existente não é aceitável, assim, analogamente ao anterior algoritmo, a discussão dos resultados terá como foco os rácios alvo superiores a 75%.

Tabela 4.6 - Resultado do algoritmo em termos de BD-PSNR e BD-rate para as tramas de textura.

Vídeos Resolução 1024×768 pixéis					
Rácio Alvo:	**75%**	**80%**	**85%**	**90%**	**95%**
BD-PSNR:	-0.056 dB	-0.041 dB	-0.024 dB	-0.009 dB	-0.002 dB
BD-rate:	1.591 %	1.154 %	0.678 %	0.262 %	-0.123 %
Percentagem do Tempo Codificação:	75.8 %	80.5 %	85.0 %	89.7 %	94.4 %
Vídeos Resolução 1920×1020 pixéis					
Rácio Alvo:	**75%**	**80%**	**85%**	**90%**	**95%**
BD-PSNR:	-0.044 dB	-0.040 dB	-0.038 dB	-0.027 dB	-0.021 dB
BD-rate:	1.930 %	1.783 %	1.668 %	1.229 %	0.915 %
Percentagem do Tempo Codificação:	79.3%	81.9%	85.3%	89.7%	93.9%

Tabela 4.7 - Resultado do algoritmo em termos de BD-PSNR e BD-rate para as tramas de profundidade.

Vídeos Resolução 1024×768 pixéis					
Rácio Alvo:	**75%**	**80%**	**85%**	**90%**	**95%**
BD-PSNR	-0.014 dB	-0.014 dB	-0.009 dB	-0.007 dB	0.002 dB
BD-rate	0.393 %	0.427 %	0.238 %	0.144 %	-0.124 %
Percentagem do Tempo Codificação:	75.8 %	80.5 %	85.0%	89.7 %	94.4 %
Vídeos Resolução 1920×1020 pixéis					
Rácio Alvo:	**75%**	**80%**	**85%**	**90%**	**95%**
BD-PSNR	-0.029 dB	-0.030 dB	-0.012 dB	-0.017 dB	-0.014 dB
BD-rate	0.670 %	0.625 %	0.371 %	0.382 %	0.369 %
Percentagem do Tempo Codificação:	79.3%	81.9%	85.3%	89.7%	93.9%

A Tabela 4.8 apresenta os valores do desvio médio entre a complexidade, resultante da codificação das sequências de vídeo utilizando o presente algoritmo, e o rácio de complexidade alvo para os algoritmos dos subcapítulos 4.1 e 4.2. A generalidade dos desvios apresentados pelo presente algoritmo são significativamente inferiores aos obtidos pelo método descrito no subcapítulo 4.1.

A Figura 4.16 mostra que a complexidade associada à codificação dos vídeos de resolução 1024×768 pixéis é muito aproximada à do rácio alvo, assim o controlador apresenta um desempenho quase ideal. Nos vídeos de maior resolução também houve uma melhoria, relativamente ao algoritmo anterior, contudo, estas sequências ainda apresentam alguma discrepância para rácios alvo inferiores a 80%. O aumento da exatidão do controlador diminui a ocorrência de ajustes superiores aos valores alvos, que têm como consequência perdas desnecessárias de eficiência na codificação.

Tabela 4.8 - Valores do desvio médio entre a percentagem do tempo de codificação e a percentagem alvo.

Vídeos Resolução 1024×768 pixéis					
Rácio Alvo:	**75%**	**80%**	**85%**	**90%**	**95%**
Desvio Médio (Subcapítulo 4.1)	-5,2%	-2,0%	1,5%	3,4%	5,2%
Desvio Médio (Subcapítulo 4.2)	-1,8%	-0,5%	0,0%	0,3%	0,6%
Vídeos Resolução 1920×1020 pixéis					
Rácio Alvo:	**75%**	**80%**	**85%**	**90%**	**95%**
Desvio Médio (Subcapítulo 4.1)	-3,6%	-1,6%	1,4%	1,4%	3,3%
Desvio Médio (Subcapítulo 4.2)	-4,3%	-1,9%	-0,3%	0,3%	1,1%

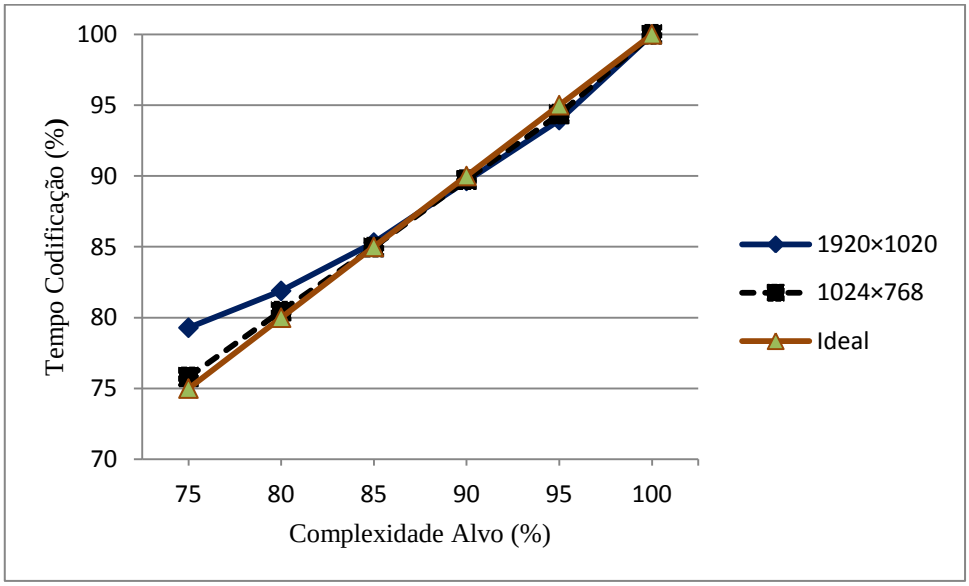


Fig. 4.16 - Resultados em termos da exatidão com que o algoritmo ajusta a complexidade computacional a um rácio alvo.

Como seria expectável os valores do BD-rate e BD-PSNR, para as tramas de textura, continuam muito elevados, visto que, o presente algoritmo mantém o mesmo modelo de restrições, que o método descrito no subcapítulo 4.1. Comparativamente ao último método, os valores de BD-rate e BD-PSNR pioraram ligeiramente. Esta ligeira degradação a nível da eficiência de codificação deve-se ao facto do peso que o processo de codificação das tramas de profundidade tem na complexidade computacional da vista dependente variar consoante as características do vídeo. Quando a complexidade computacional associada à codificação das tramas de profundidade é inferior à das tramas de textura, as tramas de textura sofrem restrições mais severas para que o tempo de codificação da vista dependente atinga o rácio de complexidade alvo.

As Figuras 4.17 e 4.18, mostram que o algoritmo impõe perdas mais severas, ao nível da eficiência de codificação, nas sequências de vídeo com maior resolução e que as tramas de profundidade apresentam perdas diminutas, na eficiência codificação, comparativamente com as

tramas de textura. O pior resultado em termos de aumento BD-rate e diminuição de BD-PSNR, para este tipo de tramas, é de 0.67% e -0.03dB respetivamente. Estes valores são muito inferiores aos 1.93% e -0.056dB verificado nas tramas textura. Este facto deve-se as características intrínsecas dos mapas de profundidade, que são tipicamente caracterizados por grandes zonas de valores constantes, que permitem que as restrições impostas não inflijam perdas elevadas na eficiência da codificação.

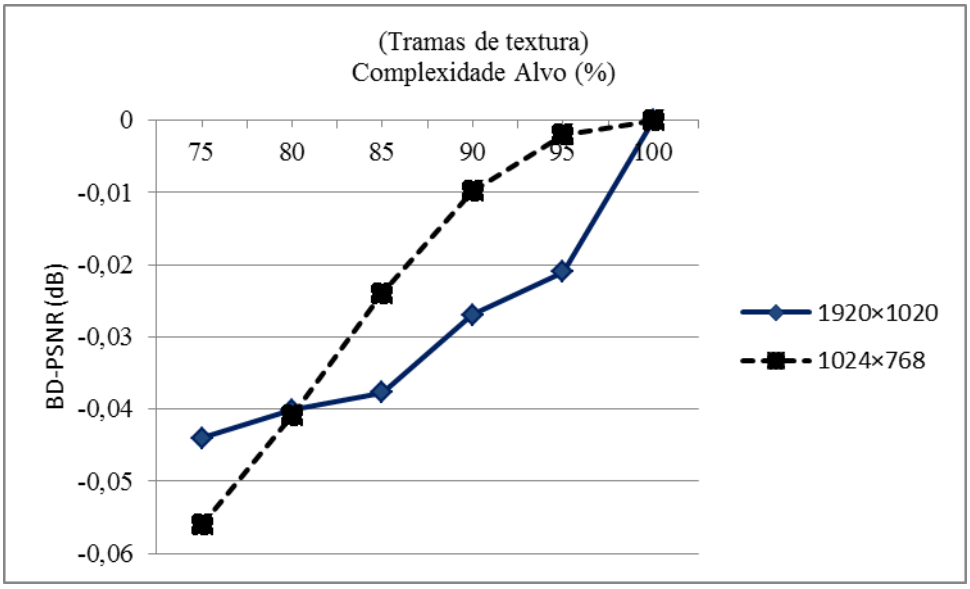


Fig. 4.17 - Resultados do BD-PSNR, das tramas de textura, em função do valor do tempo alvo.

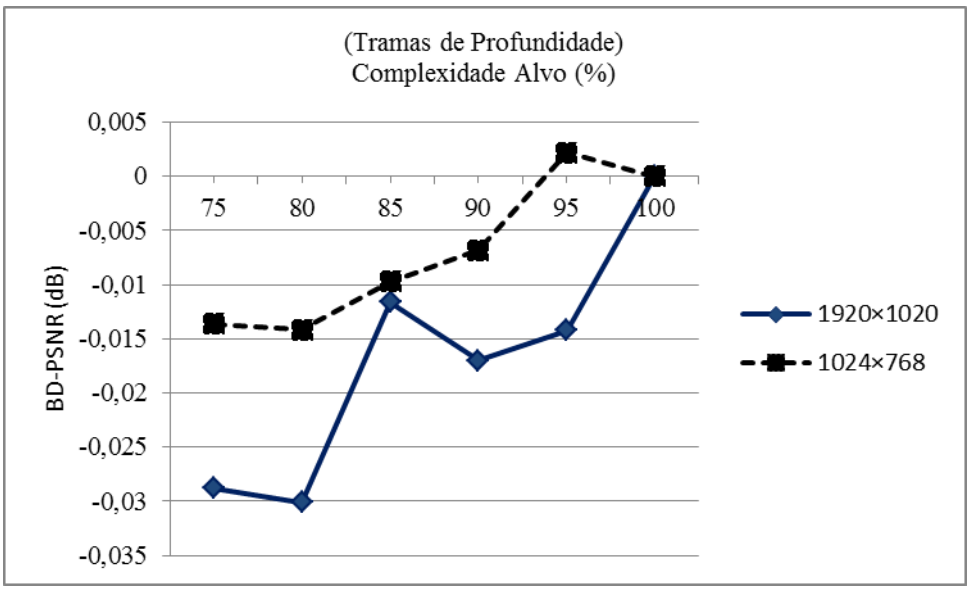


Fig. 4.18 - Resultados do BD-PSNR, medido diretamente das tramas de profundidade, em função do valor do tempo alvo.

Em suma, o algoritmo permite que a complexidade da codificação das tramas da vista dependente, não ultrapasse um rácio de complexidade alvo maior ou igual a 75% do tempo do tempo obtido durante a codificação sem restrições. Esta redução da complexidade computacional é obtida impondo uma perda de eficiência da codificação de magnitude moderada ou baixa,

traduzida numa diminuição máxima de 0,06 dB do BD-PSNR e um aumento de 1,93 % no BD-rate, para as tramas de textura. Nas tramas de profundidade existe uma diminuição máxima de 0,03 dB do BD-PSNR e um aumento de 0,67 % do BD-rate. Em termos de exatidão do algoritmo de controlo o desvio, em módulo, entre a complexidade da codificação das tramas de textura da vista dependente e o rácio de complexidade alvo (75%, 80%, 85%, 90% e 95%), foi em média de 1,4435%.

A Tabela 4.9 apresenta um quadro síntese sobre os algoritmos descritos nos últimos dois subcapítulos. O primeiro algoritmo é capaz de reduzir 25% do tempo de codificação das tramas de textura da vista dependente e o segundo 25% de toda a vista dependente. Este último apresenta melhores resultados em termos de capacidade de ajuste do tempo de codificação a um tempo alvo, com um desvio médio entre a complexidade de codificação e o valor do rácio desejado de 1,44% contra o 3,24% obtido pelo CCUPU. Em termos de avaliação da perda de eficiência de compressão ambos os algoritmos apresentam valores similares, sendo que, o algoritmo do presente subcapítulo impõe perdas de eficiência da codificação ligeiramente superior à do algoritmo do subcapítulo 4.1.

Tabela 4.9 - Tabela de síntese dos algoritmos de controlo de complexidade computacional do subcapítulo 4.1 e 4.2.

Algoritmo	**Diminuição Máxima do BD-PSNR (textura)**	**Aumento Máximo de BD-rate (textura)**	**Valor médio dos desvios (textura)**
Subcapítulo 4.1	-0,041dB	1,548%	3,24%
Subcapítulo 4.2	-0.056dB	1,930%	1,44%

Capítulo 5 – Métodos de redução de complexidade para 3D-HEVC

Neste capítulo é apresentado o algoritmo *Fast Encoding Scheme*. Este algoritmo tem como objetivo reduzir a complexidade computacional da codificação 3D-HEVC minimizando as perdas de eficiência de codificação. A redução de complexidade é obtida através da aplicação de três restrições. A primeira restrição baseia-se na aplicação de um conjunto de testes, com o objetivo de determinar em que condição é possível evitar o teste de outros modos de predição *inter* além do 2N×2N e do MSM (*Merge/Skip Mode*). Na segunda restrição, o número de modos de predição *inter* testados durante a codificação de cada CU é reduzido a um pequeno subconjunto de modos candidatos. Os modos candidatos pertencentes a este subconjunto são os modos que foram utilizados durante a codificação das CUs das CTUs vizinhas. A última restrição limita a profundidade máxima da árvore de codificação, tendo em conta a redundância existente entre vistas, a redundância entre tramas pertencentes à mesma vista e a redundância *intra*-trama.

Primeira Restrição – Decisão com base em rácios de distorção e bits

A primeira restrição foi inicialmente desenvolvida para as CUs de textura da vista dependente e posteriormente foi estendida às CUs de textura da vista independente. Esta restrição antecipa a escolha do modo 2N×2N ou do modo MSM como o modo de predição que codifica a CU, evitando o teste de outros modos de predição *inter*. A próxima subsecção descreve a aplicação desta restrição às CUs de textura da vista dependente.

Primeira restrição aplicada à vista dependente

Na codificação da CU de textura são testados vários modos de predição *inter*, de todos, os que têm maior probabilidade de serem escolhidos são os 2N×2N e MSM. A Figura 5.1 representa a frequência de escolha destes dois modos, considerando a profundidade da árvore de codificação da CU codificada. A sua análise permite concluir que independentemente do nível de profundidade observado, os modos de predição *inter* 2N×2N e MSM são os mais selecionados durante o processo de codificação da CU. A sua escolha antecipada traria uma redução da complexidade computacional.

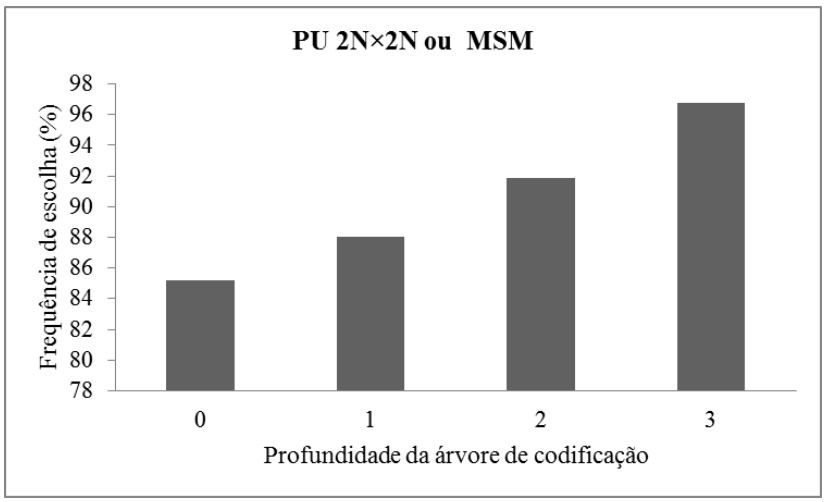


Fig. 5.1 - Frequência de escolha do modo de predição *inter* 2N×2N ou MSM ser o modo escolhido, dependente da profundidade da árvore de codificação.

O procedimento proposto para antecipar a escolha dos modos 2N×2N e MSM, assenta no pressuposto que os valores da distorção e do número de bits por *pixel* resultantes dos processos de codificação de uma região em vistas diferentes, apresentam uma forte correlação. Os parágrafos seguintes descrevem este processo de decisão.

Na codificação de cada CU os primeiros modos de predição testados são os 2N×2N e MSM. Posteriormente é calculada a distorção e o número de bits por *pixel* resultantes da codificação da CU, sem que nenhum outro modo de predição seja testado.

Os valores apurados são relacionados com os da CTU co-localizada da vista independente, após a respetiva compensação de disparidade, através do cálculo de um par de rácios, denominados *RácioDis* e *RácioBit*. Este processo é ilustrado na Figura 5.2.

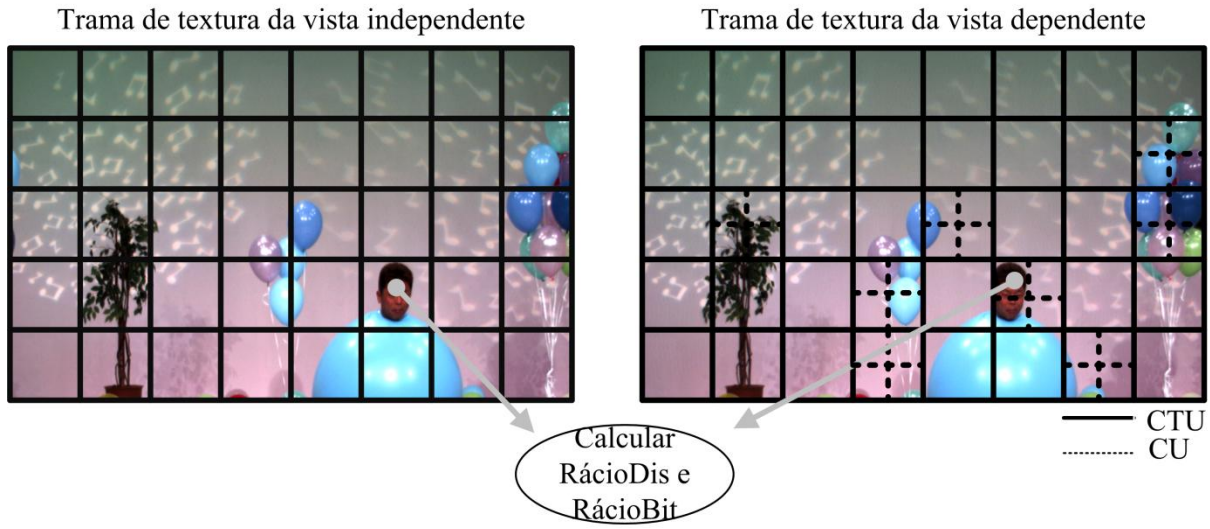


Fig. 5.2 - Calculo dos rácios *RácioDis* e *RácioBit*.

O valor do rácio *RácioDis* é obtido através da Equação 5.1, onde $DisPerPixel_{i,j,k}^{Dep}$ é a distorção por *pixel* da CU *i* pertencente à CTU *j* da vista dependente no instante *k*. $DisPerPixel_{j,k}^{Ind}$ é a distorção por *pixel* da CTU co-localizada *j* da vista independente, após a respetiva compensação, no instante *k*.

$$RacioDis = \frac{DisPerPixel_{i,j,k}^{\mathrm{Dep}}}{DisPerPixel_{j,k}^{\mathrm{Ind}}} \tag{5.1}$$

O rácio *RácioBit* é obtido através da Equação 5.2, em que $BitPerPixel_{i,j,k}^{Dep}$ é o número de bits por *pixel* da CU *i* pertencente à CTU *j* da vista dependente no instante *k*. $BitPerPixel_{j,k}^{Ind}$ é o número de bits por *pixel* da CTU co-localizada *j* da vista independente, após a respetiva compensação de disparidade, no instante *k*.

$$RacioBit = \frac{BitPerPixel_{i,j,k}^{\mathrm{Dep}}}{BitPerPixel_{j,k}^{\mathrm{Ind}}} \tag{5.2}$$

As CUs de textura da vista dependente são agrupadas em duas classes distintas, Be2N × 2N e Not2N × 2N. Na classe Be2N × 2N encontram-se as CUs em que o modo de predição *inter* escolhido é o 2N×2N ou o MSM. Na classe Not2N × 2N estão as CUs codificadas com outros modos de predição.

Os valores dos rácios calculados durante o processo de codificação de cada CU irão determinar a sua classificação, Be2N × 2N ou Not2N × 2N. A ideia base é que durante o processo de codificação de cada CU, após o teste dos modos de predição 2N×2N e MSM, sejam calculados os valores destes rácios e em seguida seja averiguada com que frequência esses valores ocorrem para cada uma das classes. Os valores dos rácios *RácioDis* e *RácioBit* cuja frequência de ocorrência seja muito baixa, por exemplo, para a classe Be2N×2N e seja relativamente elevada para a classe Not2N×2N poderão indicar conveniência em testar outros modos de predição além do 2N×2N e o MSM.

Seguindo estas ideias, como primeiro passo, determinou-se experimentalmente a frequência de ocorrência dos valores de cada rácio para cada classe de CUs. Para tal, foram construídos para cada nível de profundidade da árvore de codificação dois histogramas por rácio. Os primeiros dois histogramas, um para o rácio *RácioDis* e outro para o *RácioBit*, mostram o número de ocorrências normalizado de cada um dos rácios, nas CUs da classe Be2N×2N. O segundo par de histogramas mostra o número de ocorrências normalizado de cada um dos rácios, nas CUs da classe Not2N×2N. A Figura 5.3 ilustra o histograma resultante da sequência de vídeo

PoznanStreet (QP 22,32 e 42) para uma profundidade de árvore igual a zero e tendo em conta o valor do rácio *RácioDis.* Os histogramas que descrevem a frequência de ocorrência do *RácioBit* e do *RácioDis* para outras profundidades da árvore de codificação encontram-se no anexo C.

A etapa seguinte consistiu em realizar o teste não-paramétrico de *Kolmogorov-Smirnov* (nível de significância de 5%). O teste *Kolmogorov-Smirnov* é utilizado para determinar se uma distribuição de probabilidade difere de uma distribuição hipótese. Este teste observa a diferença absoluta entre a função distribuição assumida para os dados, e a função de distribuição empírica dos dados. O teste de *Kolmogorov-Smirnov* foi usado para avaliar as seguintes hipóteses; i) H0: os valores dos rácios *RácioDis*/*RácioBit* seguem uma distribuição normal e ii) H1: os valores dos rácios *RácioDis*/*RácioBit* não seguem uma distribuição normal. O resultado do teste é um valor escalar, denominado valor-p (*p-value*), compreendido entre [0 1], quando o valor-p é menor que 0.05 o teste rejeita a hipótese H0.

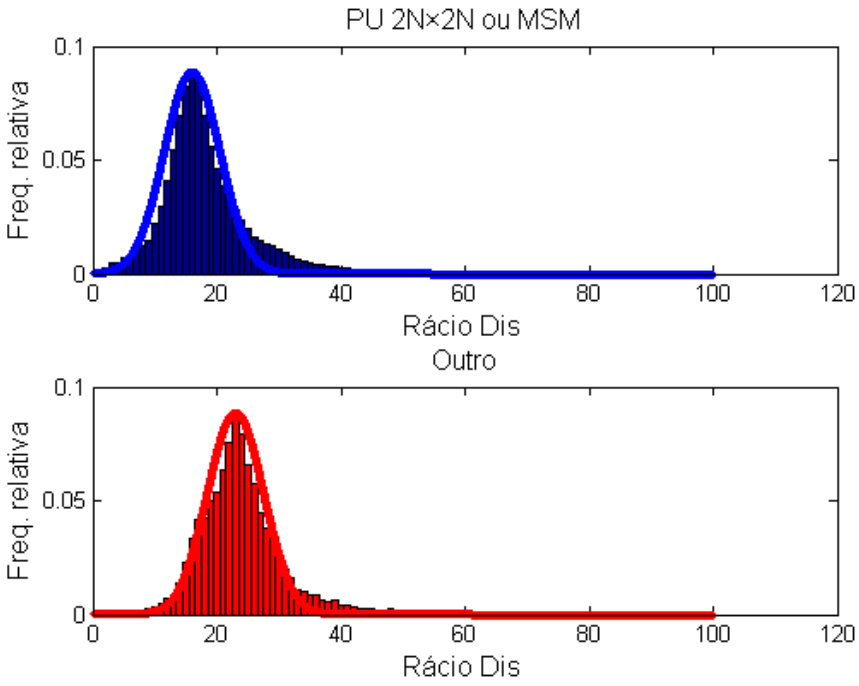


Fig. 5.3 - Histograma para os valores do Rácio*Dis* na classe de CUs Be2N×2N (topo) e Not2N×2N (baixo), para profundidade da árvore de codificação igual a zero (*bins* 0.07).

A Tabela 5.1 apresenta os valores-p resultantes do teste de *Kolmogorov-Smirnov*, estes não apresentam evidências para rejeitar a hipótese de normalidade do rácio *RácioDis*, contudo, o teste rejeitou a hipótese do rácio seguir uma distribuição normal.

Tabela 5.1 - Valores-p resultantes do teste de *Kolmogorov-Smirnov*.

Profundidade:	**RácioDis**		**RácioBit**	
	Valor-p (CUs Be2N×2N)	**Valor-p (CUs Not2N×2N)**	**Valor-p (CUs Be2N×2N)**	**Valor-p (CUs Not2N×2N)**
0	0,320	0,288	0,01	0,01
1	0,320	0,275	0,01	0,01
2	0,163	0,537	0,01	0,01
3	0,238	0,279	0,01	0,03

O teste de *Kolmogorov-Smirnov* indica a não normalidade da distribuição de probabilidade do rácio *RácioBit*, contudo, para este rácio não foi encontrada outra distribuição que caraterizasse de forma adequada o seu comportamento. Assim, tendo em conta a necessidade de simplificar a sua análise e através do conteúdo visual dos histogramas foi utilizado um conjunto de funções gaussianas para ajustar o comportamento dos diversos histogramas, como ilustra a Figura 5.3. As funções gaussianas utilizadas para ajustar os histogramas dos rácios *RácioBit* e *RácioDis*, para as diferentes profundidades da árvore de codificação, estão representados no anexo C.

Na Tabela 5.2, encontram-se os valores das médias e variâncias das várias funções gaussianas usadas para representar o comportamento de todos os histogramas, estas funções encontram-se ilustradas nas Figuras 5.4 a 5.7. É importante salientar que as gaussianas que representam as distribuições de probabilidade das CUs pertencentes à classe Be2N × 2N tendem a ter médias e desvios padrões diferentes do que CUs da classe Not2N × 2N.

Tabela 5.2 - Funções Gaussianas que ajustam o comportamento dos histogramas de cada rácio.

Profundidade:	**RácioDis**		**RácioBit**	
	$N(m,v)^{classe}$	**$N(m,v)^{classe}$**	**$N(m,v)^{classe}$**	**$N(m,v)^{classe}$**
0	$N(16,4.5)^{Be2N\times2N}$	$N(23,4.5)^{Not2N\times2N}$	$N(2,0.95)^{Be2N\times2N}$	$N(4,4.8)^{Not2N\times2N}$
1	$N(16,4.8)^{Be2N\times2N}$	$N(23,4.8)^{Not2N\times2N}$	$N(1,0.95)^{Be2N\times2N}$	$N(4,4.05)^{Not2N\times2N}$
2	$N(16,6.5)^{Be2N\times2N}$	$N(24,6)^{Not2N\times2N}$	$N(1,1.4)^{Be2N\times2N}$	$N(6,5.25)^{Not2N\times2N}$
3	$N(19,8.8)^{Be2N\times2N}$	$N(26,9)^{Not2N\times2N}$	$N(3,3)^{Be2N\times2N}$	$N(4,5.5)^{Not2N\times2N}$

As funções gaussianas representam a função densidade de probabilidade (FDP) associada a cada um dos histogramas. A decisão de não testar outros modos de predição *inter* além do 2N×2N ou MSN é suportada pelo seguinte pressuposto **"se para os valores dos rácios *RácioDis* e *RácioBit* os valores dados pela função densidade de probabilidade que representam CUs da classe Be2N×2N forem superiores aos da classe Not2N×2N (para o mesmo valores dos rácios) então não são testados outros modos de predição *inter* além do 2N×2N e MSN, caso contrário são testados todos os modos"**.

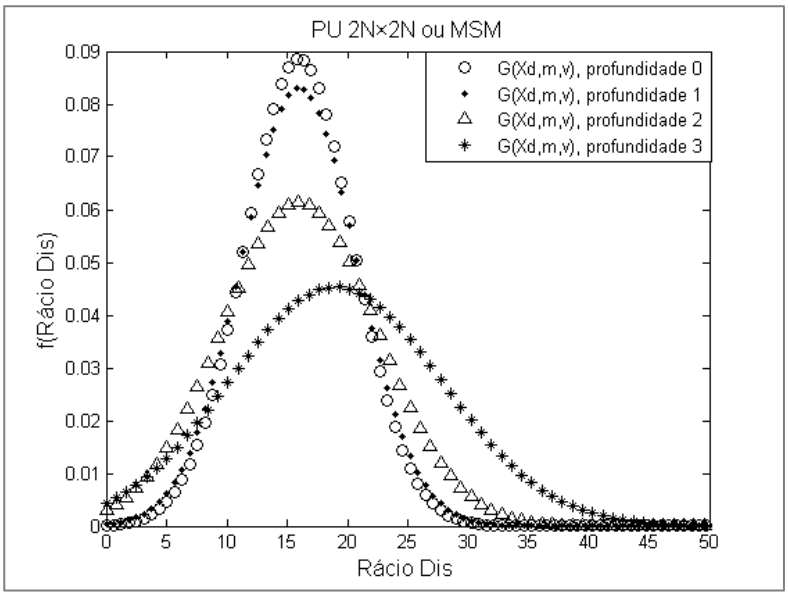


Fig. 5.4 - Funções Gaussianas que aproximam os histogramas do rácio *RácioDis*, referentes às CUs da classe Be2N×2N.

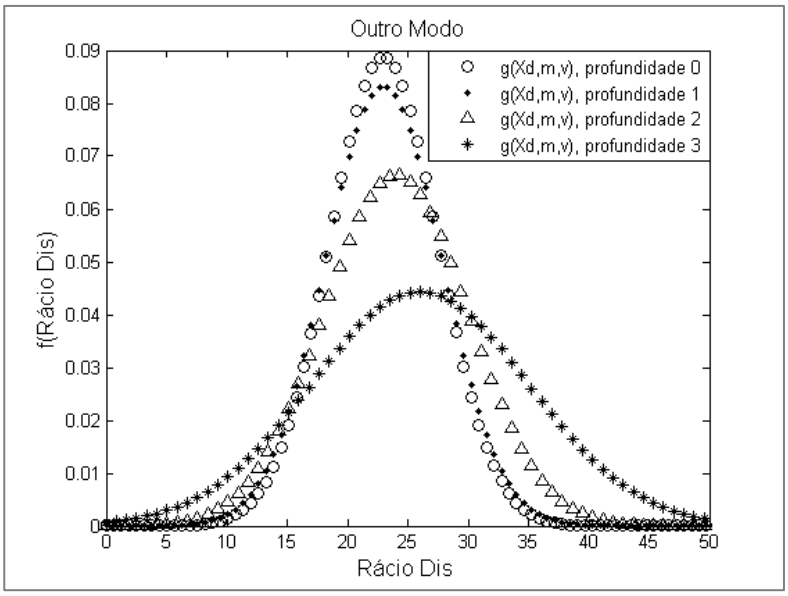


Fig. 5.5 - Funções Gaussianas que aproximam os histogramas do rácio *RácioDis*, referentes às CUs da classe Not2N×2N.

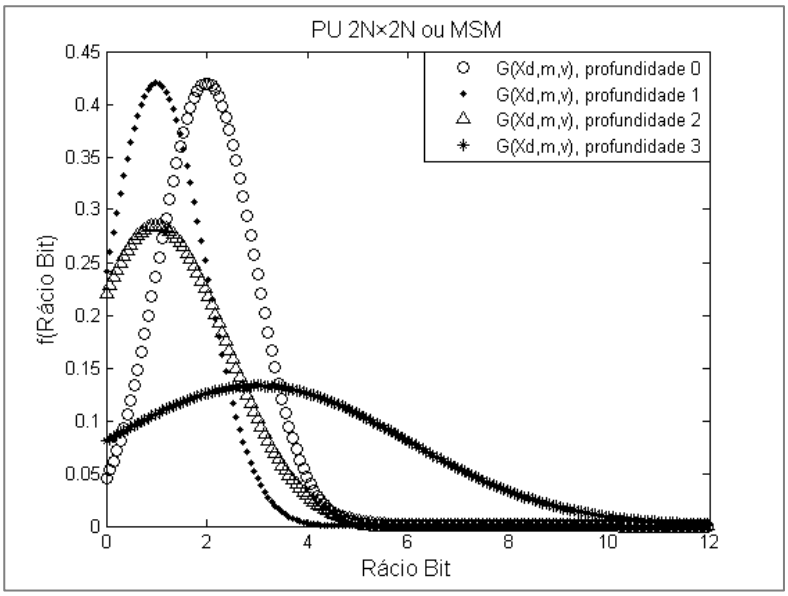


Fig. 5.6 - Funções Gaussianas que aproximam os histogramas do rácio *RácioBit*, referentes às CUs da classe Be2N×2N.

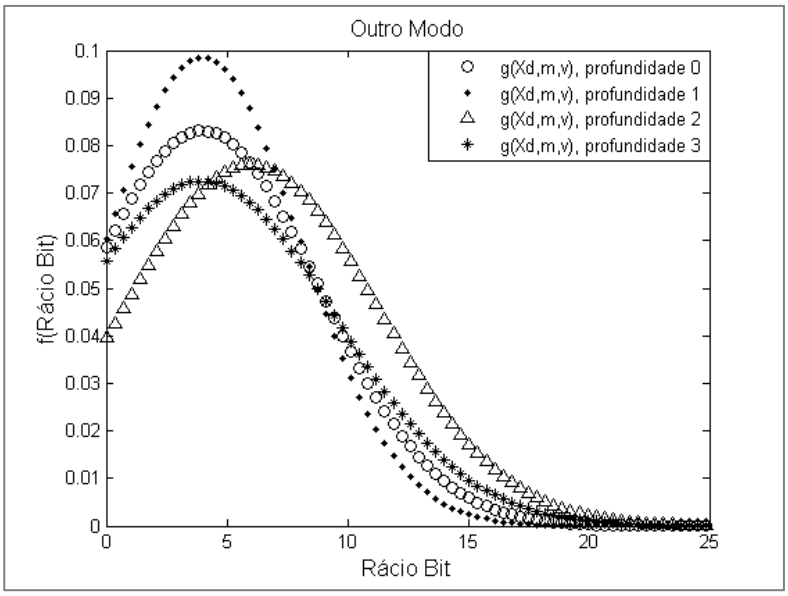


Fig. 5.7 - Funções Gaussianas que aproximam os histogramas do rácio *RácioBit*, referentes às CUs da classe Not2N×2N.

Os coeficientes *B* e *NT* são calculados com o objetivo de relacionar os valores da função densidade de probabilidade obtidos para cada um dos rácios.

O coeficiente *B* é calculado para as CUs da classe Be2N×2N a partir da Equação 5.3, em que $N(m_d,v_d)^{Be2N\times2N}_{RacioDis}$ é o valor da função densidade de probabilidade para a profundidade *d* quando o *RácioDis* é igual a x_d. $N(m_d,v_d)^{Be2N\times2N}_{RacioBit}$ é o valor da função densidade de probabilidade para a profundidade d quando o *RácioBit* é igual a x_b.

$$B(X_d,X_b)= N(m_d,v_d)^{Be2N\times2N}_{RacioDis}\times N(m_d,v_d)^{Be2N\times2N}_{RacioBit} \quad (5.3)$$

O coeficiente NT é calculado para as CUs pertencentes à classe Not2N×2N através da Equação 5.4, onde $N(m_d, v_d)^{Not2N\times2N}_{RacioDis}$ é o valor da função densidade de probabilidade para a profundidade d quando o *RácioDis* é igual a x_d. $N(m_d, v_d)^{Not2N\times2N}_{RacioBit}$ é o valor da função densidade de probabilidade para a profundidade d quando o *RácioBit* é igual a $x_{b.}$

$$NT(X_d, X_b) = N(m_d, v_d)^{Not2N\times2N}_{RacioDis} \times N(m_d, v_d)^{Not2N\times2N}_{RacioBit} \quad (5.4)$$

No último passo é verificado se o valor do coeficiente *B* é superior ao do coeficiente *NT*, caso se verifique, assume-se que para os rácios *RácioDis* e *RácioBit* calculados a CU atual deve ser considerada como pertencente à classe Be2N × 2N, ou seja, a codificação da CU termina sem testar outros modos de predição *inter*. Caso esta condição não se verifique, todos os restantes modos de predição são testados.

O desempenho deste método é suportado pelas distribuições de *RácioDis* e *RácioBit* para as duas classes de CUs. Estas distribuições identificam os valores dos rácios (*RácioBit*/*RácioDis*) que ocorrem mais frequentemente em cada classe permitindo enquadrar a CU atual.

Em suma, durante a codificação das CUs de textura da vista dependente são testados os modos de predição 2N×2N e MSM, em seguida são calculados os rácios *RácioDis* e *RácioBit*. Através das funções ilustradas nas Figuras 5.4 a 5.7 são obtidos os valores dos coeficientes *NT* e *B.* A CU termina sem que outros modos de predição *inter* sejam testados quando o valor do coeficiente *B* for superior ao do coeficiente *NT*.

A Figura 5.16 apresenta o diagrama de fluxo descrevendo a aplicação desta restrição nas tramas de textura da vista dependente. Na próxima subsecção é descrita a aplicação desta restrição às CUs de textura da vista independente.

Primeira restrição aplicada à vista independente

Esta abordagem foi replicada para a codificação das CUs de textura da vista independente, com o objetivo de reduzir a complexidade computacional associada à sua codificação. Uma vez que a vista independente é a primeira vista codificada, não existe outra região que contenha a mesma informação visual que permita calcular os rácios *RácioDis* e *RácioBit*. Para superar esta restrição é procurada na trama da vista independente, que foi codificada no instante anterior ao da trama atual, uma região que partilhe a mesma informação visual que a região a codificar. Conforme processo abaixo descrito.

Neste processo, durante a codificação da CU de tamanho 64×64 pixéis da vista independente é testado o modo de predição *inter* $2N \times 2N$. Verifica-se se os vetores de movimento associados a esse modo são nulos. Quando estes vetores são nulos, é assumido que o conteúdo visual dessa região não sofre movimento e que existe uma probabilidade razoável de a CTU atual apresentar um conteúdo visual semelhante ao da CTU co-localizada, da trama codificada no instante temporal anterior. Deste modo é possível aplicar à CU atual um procedimento semelhante ao aplicado no cálculo dos rácios *RácioDis* e *RácioBit*, em que a informação da distorção e do número de bits por *pixel* é proveniente da região co-localizada da trama codificada no instante temporal anterior.

As Figuras 5.8 e 5.9 ilustram duas tramas da sequência de vídeo *BQ Square* codificadas sequencialmente. A sua análise permite identificar as semelhanças entre os conteúdos visuais das duas tramas, nas CUs que apresentam vetores de movimentos nulos (pontos a vermelho e a azul).

Resumindo, durante a codificação de cada CTU da vista independente é testado, para a CU de tamanho 64×64 pixeis, o modo de predição *inter* 2N×2N. Após o teste é averiguado se os vetores de movimento associados a este modo são nulos. Caso se verifique, o conteúdo visual da região a codificar é considerado estático e para cada CU de menor dimensão são calculados os rácios *RácioDis* e *RácioBit.* Os coeficientes *B* e *NT* são obtidos através dos valores das FDPs ilustradas nas Figuras 5.4 a 5.7. Cada CU termina sem que outros modos de predição *inter* sejam testados além do 2N×2N e do MSM, quando o valor do coeficiente *B* é superior ao do coeficiente *NT*, caso contrário, todos os modos são testados.

Fig. 5.8 - Trama da sequência de vídeo BQ Square no instante k-1 [27].

Fig. 5.9 - Trama da sequência de vídeo BQ Square no instante k [27].

A Figura 5.15 apresenta o diagrama de fluxo descrevendo a aplicação desta restrição nas tramas de textura da vista independente.

Segunda Restrição – Decisão com base nos modos de predição utilizados nas CTUs vizinhas

A segunda restrição é aplicada às tramas de profundidade e de textura da vista dependente. Esta restrição limita a um pequeno conjunto de modos candidatos, o número de modos de predição *inter* a testar durante o processo de codificação de cada CU.

Inicialmente para determinar os modos candidatos foi realizado um estudo que relaciona os modos de predição *inter* usados na CTU codificada e os modos de predição *inter* usados nas CTUs vizinhas. As CTUs vizinhas consideradas foram: (i) CTU do topo, (ii) CTU do topo-esquerdo e (iii) CTU da esquerda. Para as CTUs das tramas de profundidade foram ainda considerados os modos de predição *inter* usados na CTU co-localizada da trama de textura, da mesma vista.

A Figura 5.10 e a Figura 5.11 mostram qual a frequência de co-ocorrência dos modos de predição *inter* usados na CTU codificada e nas CTUs vizinhas, para as tramas de textura e profundidade das sequências de vídeo; *Balloons*, *Kendo*, *PoznanHall2* e *GTFly* (QPs:25, 30, 35 e 40).

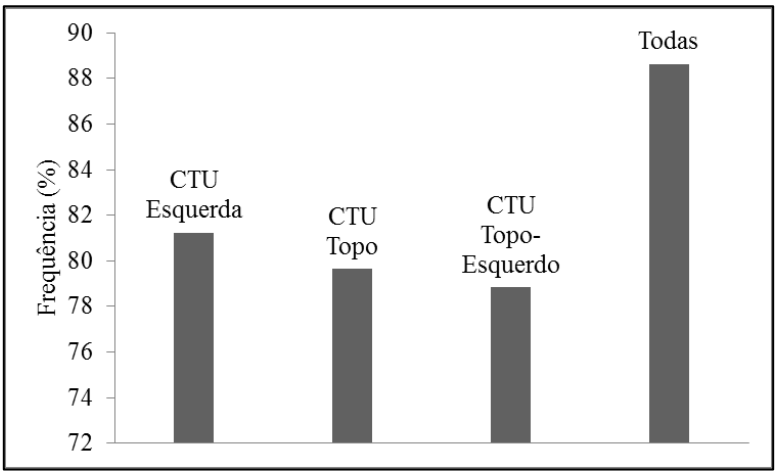


Fig. 5.10 - Percentagem de modos de predição *inter* usados na CTU codificada e nas CTUs vizinhas (textura).

Os resultados do estudo mostram que para as tramas de textura a percentagem de modos de predição *inter* usados na CTU codificada e simultaneamente numa das três CTUs vizinhas é moderadamente elevada. Contudo, em 12% dos casos os modos de predição não podem ser obtidos a partir das CTUs vizinhas. Porém se a este grupo de modos adicionarmos os modos usados na CTU co-localizada da vista independente esta predição melhora e a frequência dos casos em que o modo escolhido não é um dos usados nas CTUs vizinhas é menor.

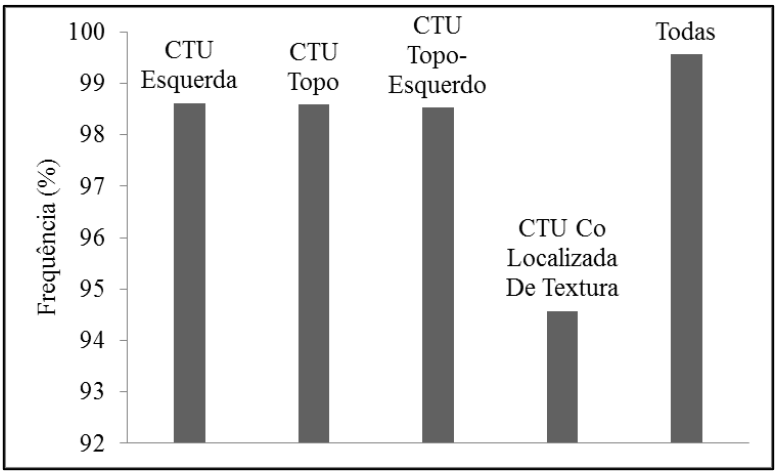


Fig. 5.11 - Percentagem de modos de predição *inter* usados na CTU de profundidade codificada e simultaneamente nas CTUs vizinhas da mesma trama e na CTU co-localizada da trama de textura.

Os resultados presentes na Figura 5.11 permitem verificar que para as tramas de profundidade, a percentagem de modos de predição *inter* que são utilizados simultaneamente nas CTUs vizinhas e na CTU a codificar é superior à percentagem obtida nas tramas de textura. Só em 6% dos casos os modos de predição *inter* usados nas CTUs vizinhas, não são os modos usados para codificar a CTU atual. Ao adicionar os modos *inter* usados na CTU co-localizada da trama de textura da mesma vista, este valor reduz-se para menos de 1%.

Os parágrafos seguintes descrevem aplicação da segunda restrição, desenvolvida a partir do estudo anterior.

As tramas da vista independente são codificadas sem restrições, portanto segundo o procedimento seguido no modelo de referência. Durante a codificação das tramas de textura e profundidade da vista dependente o número de modos de predição *inter* a testar é reduzido a um conjunto dos modos candidatos.

Os modos candidatos para codificar as CUs das tramas de textura da vista dependente são os modos usados em; i) CTU vizinha à esquerda, ii) CTU vizinha do topo-esquerdo, iii) CTU vizinha do topo e iv) CTU co-localizada da trama de textura da vista independente, após a respetiva compensação de disparidade. A Figura 5.12 ilustra este procedimento.

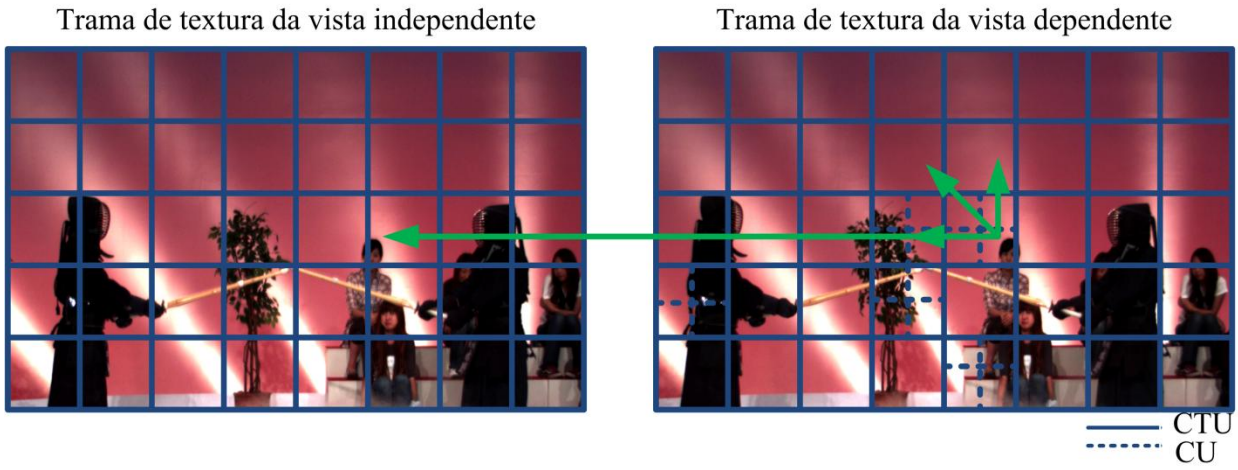


Fig. 5.12 – Conjunto de modos de predição *inter* candidatos para codificar CUs de textura da vista dependente.

Nas tramas de profundidade da vista dependente os modos de predição *inter* candidatos são os modos usados em; i) CTU vizinha à esquerda, ii) CTU vizinha do topo-esquerdo, iii) CTU vizinha do topo, iv) CTU co-localizada das tramas de textura da mesma vista e v) CTU co-localizada das tramas de profundida da vista independente, após a compensação de disparidade, Figura 5.13.

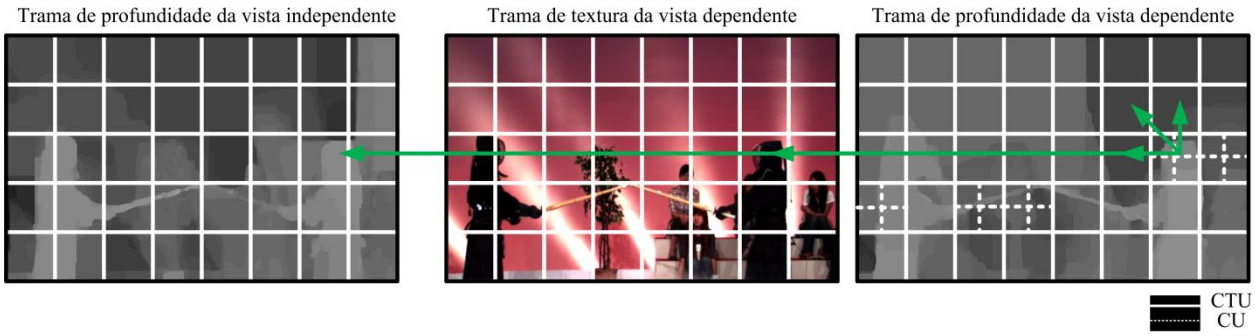


Fig. 5.13 – Conjunto de modos de predição *inter* candidatos para codificar CUs de profundidade da vista dependente.

Concluindo, durante o processo de codificação de cada CU (textura ou profundidade), da vista dependente, é testado um conjunto de modos candidatos com o objetivo de evitar o teste de todos os modos de predição *inter* e assim reduzir a complexidade computacional associada à sua

codificação. A Figura 5.16 apresenta o diagrama de fluxo descrevendo a aplicação desta restrição nas tramas de textura e profundidade da vista dependente.

Terceira Restrição – Decisão com base na redundância existente entre vistas, a redundância entre tramas pertencentes à mesma vista e a redundância *intra*-trama.

A última restrição consiste em reduzir a profundidade máxima da árvore de codificação das tramas de profundidade da vista dependente e independente. Esta restrição é somente aplicada às tramas de profundidade, visto que, estas são tipicamente menos particionadas que as tramas de textura. Com efeito, verifica-se experimentalmente que nas tramas de profundidade a profundidade máxima da árvore de codificação das CTUs vizinhas é tendencialmente maior ou igual à profundidade máxima usada na CTU codificada. Os resultados presentes na Tabela 5.3 indicam que em aproximadamente 95% das vezes a profundidade da CTU codificada é inferior à profundidade usada pelas CTUs vizinhas. Porém esta percentagem aumenta se considerarmos as profundidades máximas da árvore de codificação: i) da CTU co-localizada da trama de textura da mesma vista ii) e da CTU co-localizada da vista independente, após a respetiva compensação de disparidade.

Tabela 5.3 - Máxima profundidade da árvore de codificação utilizada nas CTUs de profundidade vizinha (GTFly, QPs 25,30,35,40).

CTU vizinha:	**Máxima profundidade:**	
	Maior ou Igual	**Menor**
Esquerda	95,6%	3,4%
Topo-Esquerdo	94,9%	5,1%
Topo	95,7%	3,3%
Topo-Direito	95,0%	5,0%

As próximas subsecções descrevem a aplicação desta restrição às CUs de profundidade da vista independente e da vista dependente.

Terceira restrição aplicada às tramas de profundidade da vista independente

Para cada CTU da trama de profundidade da vista independente é definido um valor para a máxima profundidade da árvore de codificação. Este valor é dado pelo máximo entre; i) a profundidade máxima da árvore de codificação das CTUs situadas à esquerda, no topo, no topo-esquerdo e no topo-direito da mesma trama. ii) A profundidade máxima da árvore de codificação da CTU co-localizada da trama de textura da vista independente.

Durante a codificação das tramas de profundidade da vista independente, cada CTU só pode ser dividida até à profundidade máxima igual ao limite calculado. A Figura 5.15 apresenta

o diagrama de fluxo descrevendo a aplicação desta restrição nas tramas de profundidade da vista independente.

Terceira restrição aplicada às tramas de profundidade da vista dependente

Para cada CTU da trama de profundidade é gerada uma estimativa para a máxima profundidade da árvore de codificação. A estimativa é dada pelo valor máximo entre; i) a profundidade máxima da árvore de codificação das CTUs situadas à esquerda, no topo, no topo-esquerdo e no topo-direito da mesma trama. ii) A profundidade máxima da árvore de codificação da CTU co-localizada da trama de profundidade da vista independente.

Durante o particionamento da CTU, para cada CU, a estimativa gerada anteriormente é comparada com o valor máximo entre; i) a profundidade máxima da árvore de codificação da CTU co-localizada da trama de profundidade da vista independente, após compensação de disparidade. ii) A profundidade máxima da árvore de codificação da CTU co-localizada da trama de textura da mesma vista, Figura 5.14. A Figura 5.16 apresenta o diagrama de fluxo descrevendo a aplicação desta restrição nas tramas de profundidade da vista dependente.

Fig. 5.14 – Esquema de redução da profundidade máxima da árvore de codificação.

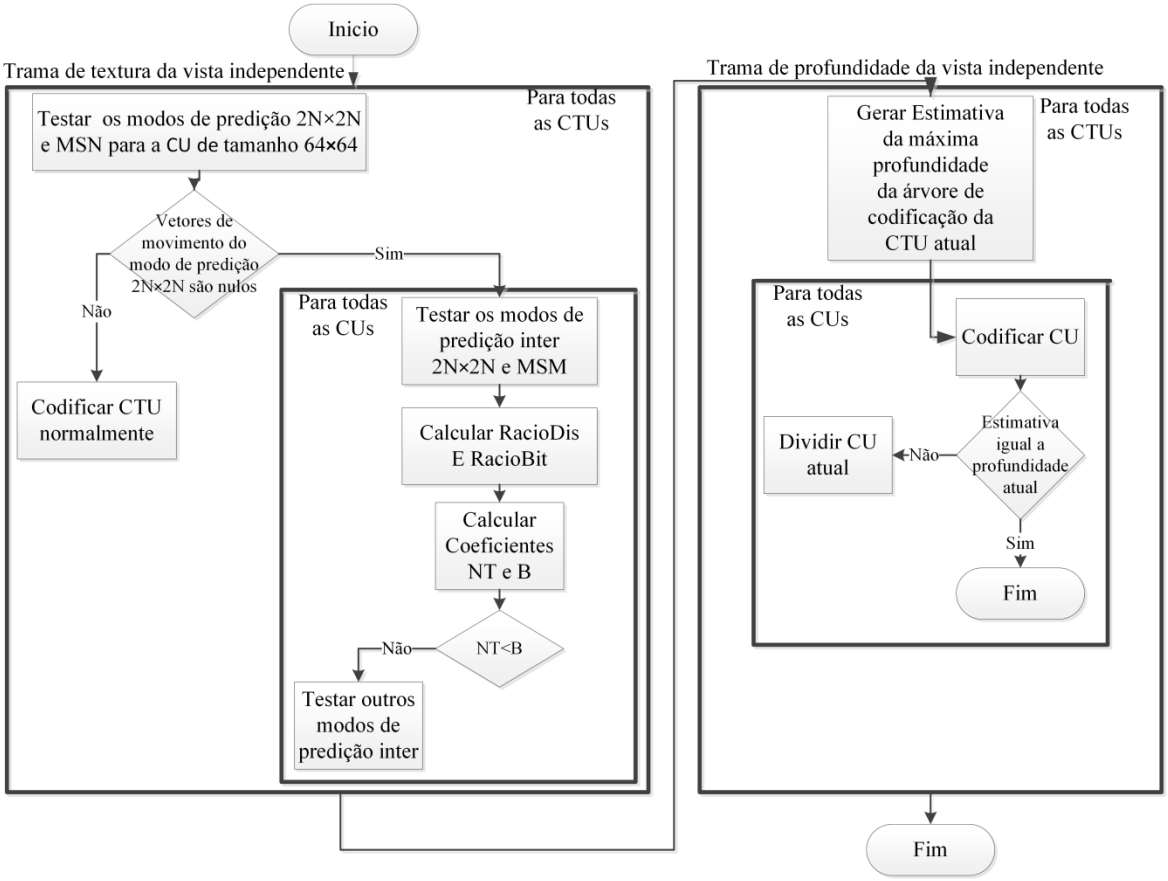


Fig. 5.15 - Diagrama de fluxo que descreve a aplicação da primeira e da terceira restrição nas tramas da vista independente.

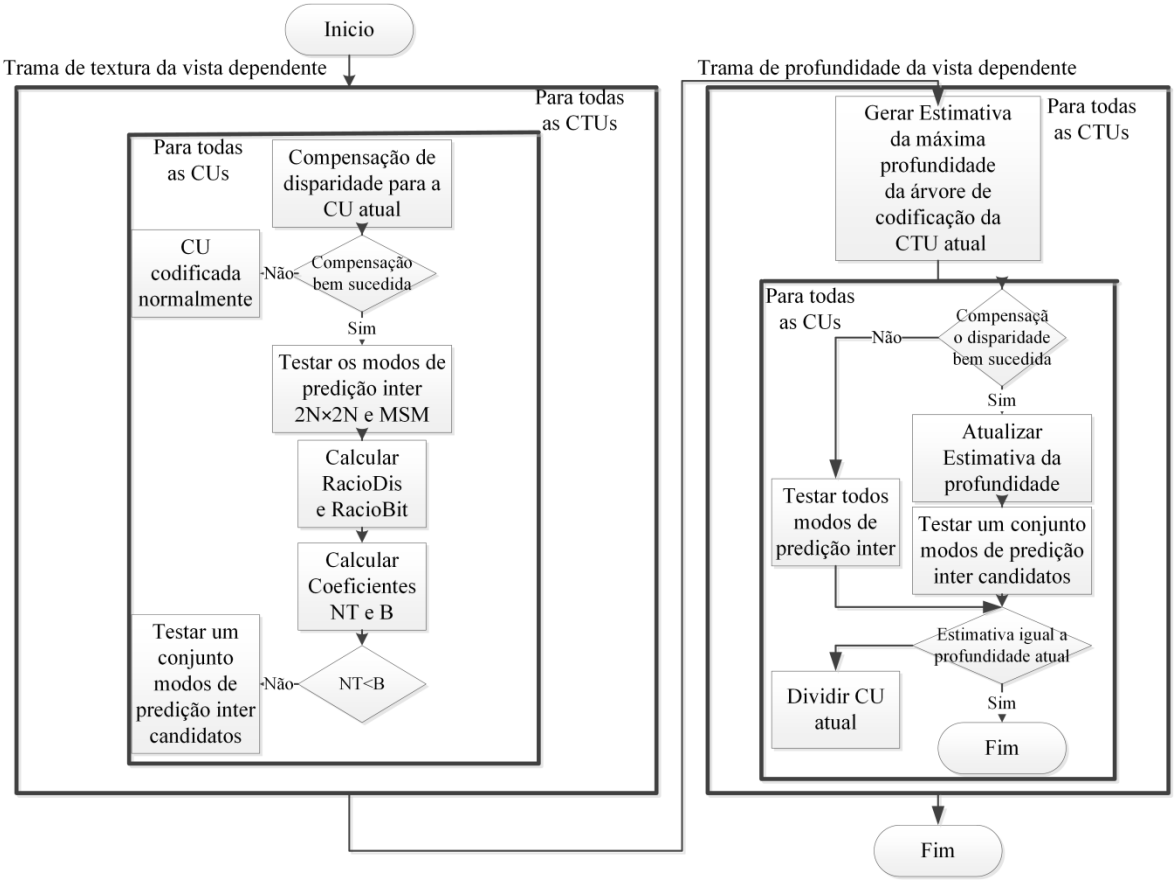


Fig. 5.16 - Diagrama de fluxo que descreve a aplicação das três restrições às tramas da vista dependente.

Na Figura 5.17 está representado o pseudo-código que descreve a aplicação conjunta das três restrições descritas anteriormente.

```
01-Inicio
02-Para todas as tramas do instante k
03-     Para todas as CU l da CTU j pertencentes a trama da textura da vista independente
04-             Testar os modos de predição inter 2N×2N e MSM
05-             Se os vectores de movimento do modo 2N×2N para a CU 64×64 forem nulos
06-                     Calcular: Rácios Bit/Dis e coeficientes NT/B.
07-                     Se coef. NT  é maior que o coef. B
08-                             Testar todos os modos de predição inter
09-             Se não
10-                     Testa todos os modos de predição inter

11-     Para todas as CTU j da trama de profundida da vista independente
12-             Gerar estimativa
13-             Para todas as CU l da CTU j
14-                     Codificar a CU l
15-                     Se profundidade actual maior que estimativa
16-                             Fim da codificação da CU l

18-     Para todas as CU l da CTU j da trama de textura da vista dependente
19-             Testar os modos de predição inter 2N×2N e MSM
20-             Se existir uma CTU co-localizada na vista independente, após a compensação de
                disparidade.
21-                     Calcular: Rácios Bit/Dis e coeficientes NT/B.
22-                     Se coef. NT  é maior que o coef. B
23-                             Testar conjunto de modos de predição inter candidatos
24-             Se não
25-                     Testa todos os modos de predição i

26-     Para todas as CTU j da trama de profundida da vista dependente
27-             Gerar estimativa
28-             Para todas as CU l da CTU j
29-                     Se existir uma CTU co-localizada na trama de profundidade da vista
                        independente, após a compensação de disparidade.
30-                             Actualizar estimativa para a profundidade máxima da CU l
31-                             Testa conjunto de modos de predição inter candidatos
32-                     Se não
33-                             Testa todos os modos de predição inter
34-                     Se profundidade actual maior que a estimativa
35-                             Fim da codificação da CU l
```

Fig. 5.17 – Pseudo-código descrevendo a aplicação conjunta das três restrições.

Análise de desempenho do algoritmo FES

O desempenho do algoritmo é medido através da sua capacidade de reduzir o tempo global de codificação, de um conjunto de sequências de teste, introduzindo um impacto reduzido na eficiência de codificação. O algoritmo foi implementado em duas versões distintas do *software* de referência - HTM-11.0 e HTM-6.2, para permitir que o seu desempenho fosse comparado com dois algoritmos presentes no estado da arte. Por último, este foi ainda comparado com o algoritmo descrito no subcapítulo 4.2.

Testes realizados na versão 6.2 do HTM

As condições de teste a que o presente algoritmo, implementado na versão HTM-6.2 do *software* de referência, esteve sujeito respeitam as normas presentes em [27]. Em resumo, foram usados nos testes dois conjuntos de sequências de vídeo. O primeiro conjunto é constituído por vídeos com resolução 1024×768 pixéis (*Balloons*, *Kendo* e *Newspaper*) e o segundo conjunto por sequências de vídeo com resolução 1920×1020 pixéis (*Dancer*, *Shark*, *PoznanHall2* e *GTFly*). Os testes foram repetidos para quatro QPs diferentes (25, 30, 35 e 40) com a configuração *Random Access Encoder*. Cada sequência de vídeo é constituída por três vistas, contendo tramas de textura e profundidade. Após descodificação destas três vistas, foram geradas artificialmente duas vistas intermédias.

A Tabela 5.4 mostra os resultados em termos de perdas na eficiência de codificação, medido em BD-rate, e de redução da complexidade computacional obtida. As colunas dois (***BD-rate Vista Ind.***), três (***BD-rate Vista Dep.1***) e quatro (***BD-rate Vista Dep.2***) apresentam o valor de BD-rate para as tramas de textura da vista independente e para as duas vistas dependentes. A quinta coluna (***BD-rate Text.***) mostra o BD-rate tendo em conta todas as vistas da sequência de vídeo. Neste cálculo somente foi considerado o PSNR e o *bitrate* das tramas de textura. A mesma avaliação é apresentada na sexta (***BD-rate Total***) e sétima coluna (***BD-rate Rend.***), onde é considerado o PSNR médio de todas as tramas, textura e profundidade, e o PSNR das vistas intermédias respetivamente. As vistas intermédias são geradas artificialmente, tipicamente no descodificador, a partir da geometria da câmara e dos mapas de profundidade das vistas codificadas. A qualidade dos mapas de profundidade é avaliada pela qualidade das vistas intermédias geradas a partir desses mapas. Nas últimas três colunas são apresentados as percentagens dos tempos de codificação; global, das vistas dependentes e da vista independente relativamente a uma codificação segundo o procedimento seguido no modelo de referência.

O algoritmo apresenta um tempo de codificação equivalente a 62,5% do tempo de referência, o que equivale a uma redução aproximada de 37% do tempo total de codificação. Esta

redução é alcançada com perdas negligenciáveis a nível da eficiência de codificação, como é possível verificar na última linha da Tabela 5.4. O maior aumento de BD-rate, cerca de 0,3%, verifica-se para a vista dependente número dois. A vista dependente número um apresenta uma ligeira melhoria com uma redução de -0,3% de BD-rate.

A sétima coluna (***BD-rate Rend.***) da Tabela 5.4 mostra os resultados das vistas geradas artificialmente, estas apresentam uma melhoria na eficiência de codificação, que se manifesta numa redução média de -0,1% no valor do BD-rate. A vista independente, segunda coluna da mesma tabela, não apresenta perdas de eficiência na codificação, contudo, exibe uma percentagem de tempo de codificação de 83,4% o que equivale a uma pequena redução de complexidade de 16,6%, enquanto que a percentagem de tempo de codificação das vistas dependentes é de 55,0% equivalente a uma redução de 45% do tempo de codificação. A codificação sem perdas de eficiência na codificação das tramas de textura da vista independente corrobora as premissas usadas na primeira restrição para evitar o teste de outros modos de predição *inter* além do modo 2N×2N e MSM.

Tabela 5.4 - Resultados em termos de redução da complexidade computacional e de perdas na eficiência de codificação.

Sequências	**BD-rate Vista Ind.**	**BD-rate Vista Dep.1**	**BD-rate Vista Dep.2**	**BD-rate Text.**	**BD-rate total**	**BD-rate Rend.**	**Percentagem do Tempo Codificação**	**Percentagem do Tempo Codificação Vistas Dep.**	**Percentagem do Tempo Codificação Vista Ind.**
Ballons	0,0%	-0,3%	0,4%	0,0%	0,1%	0,2%	66,6%	58.9%	88.0%
Kendo	0,0%	0,0%	-0,1%	0,0%	0,0%	-0,1%	63,7%	56.6%	84.7%
Newspaper	0,0%	0,1%	0,4%	0,2%	0,1%	0,2%	67,5%	62.7%	81.6%
GT-Fly	0,0%	-1,5%	0,9%	0,0%	0,3%	0,2%	58,6%	47.9%	86.0%
Poznan_Hall2	0,1%	-0,2%	0,8%	0,3%	-1,2%	-0,3%	58,1%	51.8%	79.3%
Poznan_Street	0,0%	-0,2%	-0,3%	0,0%	0,0%	-0,1%	58,0%	52.8%	75.2%
Dancer	0,0%	0,3%	0,2%	0,3%	0,3%	-0,6%	64,6%	54.3%	88.9%
Média	**0,0%**	**-0,3%**	**0,3%**	**0,1%**	**0,0%**	**-0,1%**	**62,5%**	**55.0%**	**83.4%**

Na Tabela 5.5 é feita uma comparação com o algoritmo *Fast coding algorithm* (FCA) proposto em [24]. Na coluna dois e quatro (***Percentagem do tempo de Codificação Vista Dep. (Textura)***) são comparados os percentuais dos tempos de codificação das tramas de textura da vista dependente. A segunda e quarta coluna (***BD-Rate Vista Dep.***) apresentam os valores máximos de aumento de BD-rate das vistas dependentes.

Tabela 5.5 - Comparação de resultados obtidos pelos algoritmos FES e FCA.

Sequências	FES		FCA	
	Percentagem do Tempo Codificação Vista Dep. (Textura)	**BD-Rate Vista Dep.**	**Percentagem do Tempo Codificação Vista Dep. (Textura)**	**BD-Rate Vista Dep.**
Ballons	52,3%	0,4%	50,1%	4,7%
Kendo	50,8%	-0,1%	50,2%	3,7%
Newspaper	55,7%	0,4%	52,2%	3,2%
GT-Fly	43,3%	0,9%	47,8%	-0,2%
Poznan_Hall2	45,4%	0,8%	48,3%	1,6%
Poznan_Street	45,8%	-0,3%	47,2%	2,1%
Dancer	50,8%	0,2%	50,2%	3,3%
Média	**49,2%**	**0,3**%	**49,5%**	**2,6**%

O *FCA* obtém uma redução de complexidade na ordem dos 51% com um tempo de codificação de 49,5%. Contudo, esta redução implica um aumento médio do BD-rate de 2,6%, impondo assim uma elevada perda de eficiência na codificação. Comparativamente o algoritmo proposto alcança uma redução da complexidade computacional superior, na ordem dos 50,8% o que equivale a uma percentagem de tempo de codificação de 49,2%, esta redução foi obtida impondo uma perda de eficiência média de 0,3%, medido em BD-rate, como é possível verificar na última linha da Tabela 5.5.

Em suma, o algoritmo apresenta bons resultados em termos de perdas de eficiência na codificação. Estas perdas podem ser consideradas negligenciáveis devido ao seu valor reduzido. Em termos de redução de complexidade o algoritmo obtém um desempenho satisfatório, conforme foi mostrado na comparação anterior com um algoritmo estado da arte.

Testes realizados na versão 11.0 do HTM

O algoritmo proposto foi implementado na versão HTM-11.0 do *software* de referência. As condições de teste encontram-se descritas em [28]. Estas condições são muito semelhantes às anteriormente aplicadas para a implementação na versão HTM-6.2. A grande diferença reside na adição de uma sequência de teste, denominada *Shark* de resolução 1920×1020 pixéis.

A Tabela 5.6 apresenta os resultados a nível de perdas de eficiência na codificação, medido em BD-rate e da redução de complexidade obtida, em que ***Percentagem do tempo de Codificação***, ***Percentagem do tempo de Codificação Vistas Dep.*** e ***Percentagem do tempo de Codificação Vista Ind.*** representam os percentuais dos tempos de codificação; total, das vistas dependentes e da vista independente, respetivamente. Na última coluna (***Percentagem do tempo de Codificação Vista Ind. Textura***) é apresentada a percentagem de tempo de codificação das tramas de textura da vista independente.

Tabela 5.6 - Resultados em termos de redução de complexidade e de perdas de eficiência na codificação.

Sequências	BD-rate Ind. View	BD-rate Dep. View 1	BD-rate Dep. View 2	BD-rate Text.	BD-rate Total	BD-rate Rend.	Percentagem do Tempo de Codificação	Percentagem do Tempo de Codificação Vistas Dep.	Percentagem do Tempo de Codificação Vista Ind.	Percentagem do Tempo de Codificação vista Ind. Textura
Balloons	0,0%	0,3%	0,2%	0,2%	0,2%	0,1%	87,3%	85,9%	91,5%	89,4%
Kendo	0,0%	0,3%	0,5%	0,3%	0,3%	0,1%	85,1%	83,7%	89,2%	86,6%
Newspaper	0,0%	0,3%	0,3%	0,2%	0,2%	0,1%	88,2%	90,6%	86,3%	79,9%
GT-Fly	0,1%	0,1%	0,1%	0,3%	0,3%	0,0%	83,6%	76,4%	92,4%	90,6%
Poznan_Hall2	0,0%	0,8%	1,0%	0,6%	0,5%	0,0%	80,2%	79.9%	81,1%	78,7%
PoznanStreet	0,0%	0,2%	0,1%	0,2%	0,1%	-0,1%	80,6%	84.9%	77,5%	69,2%
Dancer	0,0%	0,4%	0,7%	0,5%	0,5%	0,2%	85,5%	81,2%	91,1%	89,0%
Shark	0,0%	0,3%	0,3%	0,2%	0,2%	-0,1%	84,0%	75.0%	91,6%	89,5%
Média	**0,0%**	**0,4%**	**0,4%**	**0,3%**	**0,3%**	**0,0%**	**84,3%**	**82,2%**	**87,6%**	**84,1%**

O algoritmo é capaz de reduzir em aproximadamente 16% o tempo total de codificação, obtendo-se uma percentagem de tempo de codificação médio de 84,3% como mostra a oitava coluna (***Percentagem do Tempo de Codificação***) da Tabela 5.6. Em comparação com a implementação no HTM-6.2 a redução de complexidade computacional obtida é menor. Esta diferença deve-se ao facto de a versão 11.0 do HTM incluir algoritmos rápidos para a decisão do modo particionamento que não fazem parte da versão 6.2, entre os quais se encontra o algoritmo [23], descrito no terceiro capítulo e que permite uma redução de 47,1% do tempo total de codificação.

A nível das perdas de eficiência no processo de codificação, a implementação do presente algoritmo na versão HTM-11.0 apresenta valores de BD-rate ligeiramente superiores do que quando implementado na versão HTM-6.2, contudo, os valores obtidos são bons. A ausência de perdas de eficiência de codificação das tramas de textura da vista independente e a redução em 15,87% (84,13%) do tempo de codificação dessas tramas corrobora, analogamente ao que se verificava na implementação no HTM-6.2, as premissas usadas na primeira restrição para evitar o teste de outros modos de predição *inter* além do modo 2N×2N e MSM. A análise da relação entre a percentagem do tempo de codificação das tramas de textura da vista independente e a percentagem do tempo de codificação global da mesma vista, de 87,59%, permite constatar que grande parte da redução de complexidade operada nesta vista provém das tramas de textura, visto que, estas apresentam uma redução do tempo de codificação superior à redução final obtida pela vista independente.

Comparando os resultados obtidos do algoritmo de controlo de complexidade introduzido no subcapítulo 4.2 com os resultados do FES, estes são bastante superiores. O algoritmo do subcapítulo 4.2 para uma redução aproximadamente de 19% no tempo de codificação da vista dependente, teve um aumento médio de BD-rate de 1.51%, para as tramas de textura. O FES

obteve uma redução de 17.8% do tempo total de codificação da vista dependente com um aumento médio de 0.4% de BD-rate.

Em seguida, o algoritmo proposto foi comparado com o método descrito em [25], chamado *Fast Mode Decision Algorithm for Real-Time Applications* (FMDARA), implementado no software de referência HTM 9.0. Esta versão do software de referência já contém os métodos de decisão rápida anteriormente referidos. Contudo, os dois softwares de referência não são totalmente idênticos, entre outras alterações ao HTM 11.0 foram adicionados alguns métodos que permitem uma maior eficiência de codificação tais como; *Sub-PU level inter-view motion prediction* [29], *Depth-based Block Partitioning* [30] e *Disparity derived depth coding* [31].

A Tabela 5.7 apresenta os resultados da redução de complexidade e das perdas de eficiência na codificação obtidos pelo algoritmo FMDARA. A Tabela 5.8 compara a redução de complexidade obtida pelos dois algoritmos.

O algoritmo proposto apresenta resultados ligeiramente melhores a nível de perda de eficiência do processo de codificação. O FMDARA introduz um aumento de BD-rate de 0,7%, em média, para as tramas de textura da vista independente, como mostra a segunda coluna (***BD-rate Vista Ind.***) da Tabela 5.7, este valor contrasta com os 0,0% em média obtida pelo FES. Os resultados para as vista intermédias são semelhantes ao da vista independente, com o FMDARA a impor um aumento de 0,7%, em termos de BD-rate, contra os 0,0% obtidos pelo algoritmo proposto. Contudo, nas vistas dependentes o FES apresenta um resultado ligeiramente pior que o FMDARA.

O aumento de BD-rate de 0.6% e 0.5% obtido pelo FMDARA considerando o PSNR das tramas de textura e o valor do PSNR de todas tramas do vídeo incluindo tramas de profundidade, presente na quinta (***BD-rate Text.***) e sexta (***BD-rate Total***) coluna respetivamente, são ligeiramente piores que os obtidos pelo algoritmo proposto que impõem um aumento de 0,3% do BD-rate.

Tabela 5.7 - Resultados em termos de redução de complexidade e de perdas de eficiência na codificação do algoritmo FMDARA.

Sequências	BD-rate Vista Ind	BD-rate Vista Dep. 1	BD-rate Vista Dep. 2	BD-rate Text.	BD-rate Total	BD-rate Rend.	Percentagem do Tempo de Codificação
Ballons	0,3%	-0,4%	-0,2%	0,1%	0,0%	0,9%	62%
Kendo	0,3%	0,3%	0,3%	0,3%	0,2%	0,6%	66%
Newspaper	0,6%	0,4%	0,2%	0,5%	0,3%	0,7%	60%
GT-Fly	0,7%	-1,7%	-1,3%	0,9%	0,8%	0,6%	54%
Poznan_Hall2	1,1%	1,1%	-0,2%	0,8%	0,6%	1,1%	55%
Poznan_Street	0,8%	0,5%	0,9%	0,9%	0,9%	0,7%	59%
Dancer	0,8%	0,4%	-0,2%	0,8%	0,6%	0,8%	59%
Média	**0,7**%	**0,1**%	**-0,1**%	**0,6**%	**0,5**%	**0,7**%	**59%**

Contudo, apesar do baixo nível de perdas na eficiência de codificação existe uma grande discrepância entre a redução de complexidade alcançada pelos dois algoritmos. O algoritmo proposto reduz em aproximadamente 16% o tempo de codificação, este valor é bastante baixo comparado com os 41% obtidos pelo FMDARTA que equivale a uma percentagem do tempo de codificação de 59%, como mostra a Tabela 5.8.

Tabela 5.8 – Resultados em termos das percentagens dos tempos de codificação obtidos pelos algoritmos FES e FMDARTA.

Sequências	Percentagem do Tempo de Codificação	
	FES	**FMDARTA**
Ballons	88%	62%
Kendo	84%	66%
Newspaper	80%	60%
GT-Fly	81%	54%
Poznan_Hall2	86%	55%
Poznan_Street	88%	59%
Dancer	84%	59%
Média	**84%**	**59%**

Em suma, o FES apresenta algumas fragilidades na redução da complexidade computacional, onde obteve valores moderadamente baixos. Contudo, obtém bons resultados em termos de perdas de eficiência na codificação. O valor reduzido destas perdas possibilita o desenvolvimento de novas restrições que permitam aumentar a redução da complexidade computacional, especialmente nas tramas de textura da vista independente e nas tramas de profundidade da vista dependente, onde as perdas de eficiência na codificação são virtualmente nulas.

Capítulo 6 – Conclusão

O principal objetivo deste trabalho foi desenvolver algoritmos de controlo e de redução da complexidade computacional associada ao 3D-HEVC. Este *codec* suporta a codificação de múltiplas vistas, cada uma contendo dados de textura e profundidade. A complexidade associada ao 3D-HEVC cresce linearmente com o número de vistas, este facto agiganta a necessidade de desenvolver métodos que permitam atenuar este problema.

No segundo capítulo foram apresentadas as noções básicas sobre a codificação de vídeo. Nele, foram introduzidas breves descrições sobre a estrutura do codificador de vídeo H.265/HEVC e sobre a sua extensão 3D. No fim do capítulo foram apresentadas algumas métricas usadas para avaliar a qualidade do processo de codificação e descodificação.

O primeiro algoritmo de controlo da complexidade computacional teve como objetivo ajustar o tempo de codificação das tramas de textura, da vista dependente, a um tempo alvo. Verificou-se que algoritmo permitia esse ajuste da complexidade computacional, mas dentro de uma banda muito limitada, não superior a uma redução de 25% do tempo de codificação de referência. A perda de eficiência a nível de codificação foi moderada. Para uma redução da complexidade de 25% existe um aumento médio de 1,570 % do BD-rate. Em termos de exatidão o algoritmo obteve um desvio médio de 3,2% entre a complexidade da codificação das tramas de textura da vista dependente e o rácio de complexidade alvo.

O segundo algoritmo apresentado estendeu o controlo de complexidade às tramas de profundidade da vista dependente, ou seja, o algoritmo tem como objetivo manter o tempo global de codificação da vista dependente, composta por tramas de textura e profundidade abaixo de um rácio alvo. Para melhorar o seu desempenho em termos de exatidão foi implementado um novo sistema de controlo, que antes de codificar uma trama da vista dependente estima o seu tempo de codificação, de forma a calcular as restrições necessárias para que este não ultrapasse um tempo alvo. Este método permite obter um desvio médio de 1,44% entre a complexidade da codificação e o rácio de complexidade alvo.

O algoritmo final, FES, surge com o objetivo de encontrar restrições que permitam reduzir a complexidade computacional, sem apresentar um grande impacto na eficiência de codificação, como os anteriores algoritmos. A redução de complexidade é obtida através da aplicação de três restrições. A primeira restrição antecipa a escolha dos modos de predição *inter*

2N×2N e MSM, durante o processo de codificação das CUs de textura. Na segunda restrição, o número de modos de predição *inter* testados durante a codificação de cada CU da vista dependente é reduzido a um pequeno subconjunto de modos candidatos. A última restrição limita a profundidade máxima da árvore de codificação das tramas de profundidade. O FES implementado na versão HTM-11.0 do software de referência permitiu reduzir aproximadamente 16% do tempo global de codificação, sendo que nas vistas dependentes esse valor atinge os 17.8%. Esta redução é obtida com um pequeno aumento 0,4% do BD-rate nas vistas dependentes. A perda de eficiência de codificação é muito pequena, comparado com os algoritmos anteriores, para um nível similar de redução. Quando implementado na versão HTM-6.2 do *software* de referência, este apresenta resultados superiores obtendo uma redução aproximadamente 37% do tempo global de codificação, sendo que nas vistas dependentes esse valor atinge os 55,0%. As perdas na eficiência de codificação resultantes desta redução equivalem a um aumento de BD-rate de 0,3%.

Os objetivos que foram propostos no início desta dissertação ainda podem ser melhorados, nomeadamente na melhoria da capacidade de redução dos algoritmos desenvolvidos. Existe um grande campo de investigação no aperfeiçoamento das restrições desenvolvidas para o último algoritmo, de forma a permitir uma maior redução de complexidade, mantendo, no entanto, baixas perdas a nível da eficiência de codificação. Uma das vias de investigação seria desenvolver restrições mais severas, que restringissem o limite da máxima profundidade da árvore de codificação das tramas de profundidade. Outra via, é estender às tramas de profundidade das vistas dependentes e independentes e aos modos de predição *intra* a restrição que permite, que dentro de certas condições, só sejam testados os modos de predição *inter* $2N \times 2N$ e MSM. Por ultimo, as várias restrições, acima mencionadas, podem ser adaptadas ao algoritmo descrito no subcapítulo 4.2 de modo a desenvolver um método de controlo de complexidade que permita ajustar a complexidade computacional mantendo baixas perdas a nível da eficiência.

Referências

[1] Sullivan, O. Gary J., H. Jens-Rainer, W. Woo-Jin and Thomas, "Overview of the High Efficiency Video Coding (HEVC) Standard," *IEEE Transactions on Circuits and Systems for Video Technology,* vol. 22, no. 12, pp. 1649 - 1668, 2012.

[2] Wiegand, S. Thomas, B. Gary J., L. Gisle and Ajay, "Overview of the H.264/AVC Video Coding Standard," *IEEE Transacations on Circuits And Systems For Video Technology,* vol. 13, no. 7, pp. 560-576, 2013.

[3] G. Corrêa, P. Assunção, L. Agostini and L. A. d. S. Cruz, "Performance and Computational Complexity Assessment of High-Efficiency Video Encoders," *IEEE Transactions on Circuits and Systems for Video Technology,* vol. 22, no. 12, pp. 1899 - 1909, 2012.

[4] Müller, S. Karsten, M. Heiko, B. Detlev, B. Christian, B. Sebastian, H. Heribert, L. Tobias, M. Haricharan, R. Philipp, T. Franz Hunn, W. Gerhard, W. Martin and Thomas, "3D High-Efficiency Video Coding for Multi-View Video and Depth Data," *IEEE Transaction On Image Processing,* vol. 22, no. 9, pp. 3366-3378, 2013.

[5] G. Correa, P. Assuncao, L. A. d. S. Cruz and L. Agostini, "Constrained Encoding Structures for Computational Scalability in HEVC," in *Picture Coding Symposium (PCS)*, San Jose, CA, 2013.

[6] G. Corrêa, *(PhD Thesis) Computacional Complexity Reduction and Scaling for High Efficiency Video Encoders,* Coimbra: University Of Coimbra, 2015.

[7] I. E. G. Richardson, Video Codec Design - Developing Image and Video Compression Systems, Chichester:John Wiley and Sons, 2002.

[8] G. J. Sullivan and T. Wiegand, "Rate-distortion optimization for video compression," *IEEE, Signal Processing Magazine,* vol. 15, no. 6, pp. 74 - 90, 1998.

[9] D. Marpe, H. Schwarz and T. Wiegand, "Context-Based Adaptive Binary Arithmetic Coding in the H.264/AVC Video Compression Standard," *IEEE Transactions On Circuits and Systems For Video Technology,* vol. 13, no. 7, pp. 620-636, 2003.

[10] A. Norkin, G. Bjøntegaard, A. Fuldseth, M. Narroschke, I. Kenneth, M. Zhou and G. V. d. Auwera, "HEVC Deblocking Filter," *IEEE Transactions on Circuits and Systems for Video Technology,* vol. 22, no. 12, pp. 1746-1754, 2012.

[11] C.-M. Fu and e. al, "Sample Adaptive Offset in the HEVC Standard," *IEEE Transactions on Circuits and Systems For Video Technology,* vol. 22, no. 12, pp. 1755-1763, 2012.

[12] G. Tech, K. Wegner, Y. Chen and S. Yea, "3D-HEVC Test Model 3," Document: [JCT3V-C1005], Geneva, 2013.

[13] A. Smolic, K. Müller, K. Dix, P. Merkle, P. Kauff and T. Wiegand, “Intermediate View Interpolation Based On Multiview Video Plus Depth For Advanced 3D Video Systems,” in *15th IEEE International Conference on Image Processing* , San Diego, CA, 2008.

[14] G. Bjøntegaard, “Calculation of average PSNR differences between RD-curves,” Document: [JCT3V-C1005], Geneva, April 2001.

[15] B. Bross, “JCT-VC AHG report: HEVC test model editing and errata reporting (AHG2),” Document: [JCTVC-S0002], Strasbourg, FR, Oct. 2014.

[16] D. Rusanovskyy, “JCT-3V AHG Report: 3D-AVC Software Integration (AHG3),” Document: [JCT3V-J0003], Strasbourg, FR, Oct. 2014.

[17] X. Shen, L. Yu and J. Chen, “Fast Coding Unit Size Selection for HEVC based on Bayesian Decision Rule,” in *Picture Coding Symposium (PCS), 2012*, Krakow, 2012,.

[18] Liu, L. Yinbo, W. Xingang and P. Peicheng, “A Texture Complexity Based Fast Prediction Unit Size Selection Algorithm for HEVC Intra Coding,” in *IEEE 17th International Conference on Computational Science and Engineering (CSE)*, Chengdu, 2014.

[19] H. Gweon, L. Ryeong, L. Yung-Lyul and Jeongyeon, “Early Termination of CU Encoding to Reduce HEVC Complexity,” Document: [JCTVC-F045], Torino, July, 2011.

[20] P. Zhaoqing, S. Kwong, M.-T. Sun and J. Lei, “Early MERGE Mode Decision Based on Motion Estimation and Hierarchical Depth Correlation for HEVC,” *IEEE Transactions On Broadcasting,* vol. 60, no. 2, pp. 405-412, 2014.

[21] P. Helle, S. Oudin, B. Bross, D. Marpe, M. O. Bici, K. Ugur, J. Jung, G. Clare and T. Wiegand, “Block Merging for Quadtree-Based Partitioning in HEVC,” *IEEE Transactions on Circuits and Systems for Video Technology,* vol. 22, no. 12, pp. 1720 - 1731, 2012.

[22] E. G. Mora, J. Jung, M. Cagnazzo e B. Pesquet-Popescu, “Initialization, Limitation, and Predictive Coding of the Depth and Texture Quadtree in 3D-HEVC,” *IEEE Transactions On Circuits And Systems For Video Technology ,* vol. 24, n.º 9, pp. 1554-1565, 2014.

[23] N. Zhang, D. Zhao, Y.-W. Chen, J.-L. Lin e W. Gao, “Fast encoder decision for texture coding in 3D-HEVC,” *Signal Processing: Image Communication,* vol. 29, n.º 9, p. 951–961, 2014.

[24] G. Chi, X. Jin and Q. Dai, “A Fast Coding Algorithm Based on Inter-View Correlations for 3D-HEVC,” *Visual Communications and Image Processing Conference, IEEE,* pp. 374 - 377, 7-10 December 2014.

[25] L. Shen, P. An, Z. Zhang, Q. Hu and Z. Chen, “A 3D-HEVC Fast Mode Decision Algorithm for Real-Time Applications,” *ACM Trans. Multimedia Comput. Commun,* vol. 11, no. 3, p. 23, 2015.

[26] *Software Gitl HEVC analyser.*

[27] D. Rusanovskyy, K. Muller and A. Vetro, “Common Test Conditions of 3DV Core Experiments,” Document: [JCT3V-D100], Geneva, 2013.

[28] D. Rusanovskyy, K. Muller and A. Vetro, “Common Test Conditions of 3DV Core Experiments,” Document: [JCT3V-F1100] , Geneva, 2013.

[29] J. An, K. Zhang and J.-L. Lin, “3D-CE3: Sub-PU level inter-view motion prediction,” Document: [JCT3V-F0110], Geneva, 2013.

[30] F. Jäger, “CE3: Results on Depth-based Block Partitioning (DBBP),” Document: [JCT3V-G0106], San Jose, USA, 2013.

[31] K. Zhang, J. An, J.-L. Lin, Y.-L. Chang and S. Lei, “3D-CE3.h related:A disparity derived depth coding method in 3D-HEVC,” Document: [JCT3V-E0174], Vienna, 2013.

Anexo A – Resultados (algoritmo do subcapítulo 4.1)

Este anexo apresenta os resultados do algoritmo introduzido no subcapítulo 4.1. Os valores do BD-PSNR e do BD-rate, para os vídeos de resolução 1024×768 pixéis (Balloons, Kendo e Newspaper), encontram-se discriminados nas Tabelas; A1, A2, A3, A4 e A5. A Tabela A.6 apresenta os desvios entre a percentagem do tempo de codificação, resultante da codificação das sequências de vídeo utilizando o presente algoritmo, e os rácios alvo. Os desvios em módulos são apresentados na Tabela A.7. Por último, da Tabela A.8 a Tabela A.13 são descriminados os resultados, acima referenciados, para as sequências de vídeo de resolução 1920×1020 pixéis (PoznanHall2,GT-Fly, Dancer e Shark).

Sequências de vídeos de resolução 1024×768 pixéis

Tabela A.1 - Resultados em termos de BD-rate e BD-PSNR do algoritmo do subcapítulo 4.1 para as sequências de vídeo de resolução 1024×768 pixéis (target 75%).

		Balloons	*Kendo*	*Newspaper*	Média
Target 75%	BD-PSNR (dB)	-0,0347	-0,0395	-0,0499	**-0,0414**
	BD-rate (%)	0,9648	1,0853	1,4212	**1,1571**
	T.C. (%)	**78.7**	**79.4**	**82.6**	**80.2**

Tabela A.2 - Resultados em termos de BD-rate e BD-PSNR do algoritmo do subcapítulo 4.1 para as sequências de vídeo de resolução 1024×768 pixéis (target 80%).

		Balloons	*Kendo*	*Newspaper*	Média
Target 80%	BD-PSNR (dB)	-0,0340	-0,0359	-0,0416	**-0,0372**
	BD-rate (%)	0,9477	1,0244	1,1667	**1,0463**
	T.C. (%)	**80.7**	**81.6**	**83.8**	**82.0**

Tabela A.3 - Resultados em termos de BD-rate e BD-PSNR do algoritmo do subcapítulo 4.1 para as sequências de vídeo de resolução 1024×768 pixéis (target 85%).

		Balloons	*Kendo*	*Newspaper*	Média
Target 85%	BD-PSNR (dB)	-0,0331	-0,0307	-0,0291	**-0,0310**
	BD-rate (%)	0,9059	0,8595	0,8328	**0,8661**
	T.C. (%)	**81.6**	**84.4**	**84.6**	**83.5**

Tabela A.4 - Resultados em termos de BD-rate e BD-PSNR do algoritmo do subcapítulo 4.1 para as sequências de vídeo de resolução 1024×768 pixéis (target 90%).

		Balloons	*Kendo*	*Newspaper*	Média
Target 90%	BD-PSNR (dB)	-0,0216	-0,0200	-0,0209	**-0,0208**
	BD-rate (%)	0,5774	0,5437	0,5732	**0,5648**
	T.C. (%)	**85.7**	**87.6**	**86.3**	**86.5**

Tabela A.5 - Resultados em termos de BD-rate e BD-PSNR do algoritmo do subcapítulo 4.1 para as sequências de vídeo de resolução 1024×768 pixéis (target 95%).

		Balloons	*Kendo*	*Newspaper*	Média
Target 95%	BD-PSNR (dB)	-0,0109	-0,0149	-0,0130	**-0,0129**
	BD-rate (%)	0,2780	0,3802	0,3634	**0,3406**
	T.C. (%)	**91.7**	**89.8**	**87.9**	**89.8**

Tabela A.6 - Desvios entre a percentagem do tempo de codificação e o rácio alvo, para os vídeos de resolução 1920×1020 pixéis.

Vídeos/Rácios (%)	*75*	*80*	*85*	*90*	95
Balloons	-3,7%	0,7%	3,4%	4,3%	3,3%
Kendo	-4,4%	1,6%	0,6%	2,4%	5,2%
Newspaper	-7,6%	3,8%	0,4%	3,5%	7,1%
Média	**-5,2%**	**2,0%**	**1,5%**	**3,4%**	**5,2%**

Tabela A.7 - Desvios, em módulo, entre a percentagem do tempo de codificação e o rácio alvo, para os vídeos de resolução 1024×768 pixéis.

Vídeos/Rácios(%)	*75*	*80*	*85*	*90*	95
Balloons	3,7%	0,7%	3,4%	4,3%	3,3%
Kendo	4,4%	1,6%	0,6%	2,4%	5,2%
Newspaper	7,6%	3,8%	0,4%	3,5%	7,1%
Média	**5,2%**	**2,0%**	**1,5%**	**3,4%**	**5,2%**

.

Sequências de vídeos de resolução 1920×1020 pixéis

Tabela A.8 - Resultados em termos de BD-rate e BD-PSNR do algoritmo do subcapítulo 4.1 para as sequências de vídeo de resolução 1920×1020 pixéis (target 75%).

		PoznanHall2	*GtFly*	*Dancer*	*Shark*	Média
Target 75%	BD-PSNR (dB)	-0,0772	-0,0240	-0,0204	-0,0175	**-0,0348**
	BD-BR (%)	3,9103	1,0153	0,7466	0,5180	**1,5475**
	T.C. (%)	**83.3**	**79.9**	**80.6**	**70.5**	**78.6**

Tabela A.9 - Resultados em termos de BD-rate e BD-PSNR do algoritmo do subcapítulo 4.1 para as sequências de vídeo de resolução 1920×1020 pixéis (target 80%).

		PoznanHall2	*GtFly*	*Dancer*	*Shark*	Média
Target 80%	BD-PSNR (dB)	-0,0598	-0,0239	-0,019	-0,0139	**-0,0292**
	BD-rate (%)	3,0437	1,0222	0,7051	0,4217	**1,2982**
	T.C. (%)	**82.8**	**81.8**	**83.1**	**78.8**	**81.6**

Tabela A.10 - Resultados em termos de BD-rate e BD-PSNR do algoritmo do subcapítulo 4.1 para as sequências de vídeo de resolução 1920×1020 pixéis (target 85%).

		PoznanHall2	*GtFly*	*Dancer*	*Shark*	Média
Target 85%	BD-PSNR (dB)	-0,0495	-0,0238	-0,0160	-0,0112	**-0,0251**
	BD-rate (%)	2,5165	1,0148	0,5946	0,3293	**1,1138**
	T.C. (%)	**83.9**	**80.6**	**84.0**	**85.7**	**83.6**

Tabela A.11 - Resultados em termos de BD-rate e BD-PSNR do algoritmo do subcapítulo 4.1 para as sequências de vídeo de resolução 1920×1020 pixéis (target 90%).

		PoznanHall2	*GtFly*	*Dancer*	*Shark*	Média
Target 90%	BD-PSNR (dB)	-0,0453	-0,0236	-0,0122	-0,0118	**-0,0232**
	BD-rate (%)	2,3824	1,0100	0,4505	0,3447	**1,0469**
	T.C. (%)	**90.4**	**85.0**	**88.0**	**91.2**	**88.6**

Tabela A.12 - Resultados em termos de BD-rate e BD-PSNR do algoritmo do subcapítulo 4.1 para as sequências de vídeo de resolução 1920×1020 pixéis (target 95%).

		PoznanHall2	*GtFly*	*Dancer*	*Shark*	Média
Target 95%	BD-PSNR (dB)	-0,0118	-0,0189	-0,0073	-0,0017	**-0,0099**
	BD-rate (%)	0,6734	0,8008	0,2529	0,0446	**0,4429**
	T.C. (%)	**94.6**	**85.5**	**91.7**	**94.8**	**91.7**

Tabela A.13 - Desvios entre a percentagem do tempo de codificação e o rácio alvo, para os vídeos de resolução 1920×1020 pixéis.

Vídeos/Rácios (%)	*75*	*80*	*85*	*90*	95
GT-Fly	-5,0%	-1,8%	4,4%	5,0%	9,5%
Dance	-5,6%	-3,1%	1,0%	2,0%	3,3%
PoznanHall2	-8,3%	-2,8%	1,1%	-0,4%	0,4%
Shark	4,5%	1,2%	-0,7%	-1,2%	0,2%
Média	**-3,6%**	**-1,6%**	**1,4%**	**1,4%**	**3,3%**

Tabela A.14 - Desvios, em módulo, entre a percentagem do tempo de codificação e o rácio alvo, para os vídeos de resolução 1920×1020 pixéis.

Vídeos/Rácios (%)	*75*	*80*	*85*	*90*	95
GT-Fly	5,0%	1,8%	4,4%	5,0%	9,5%
Dance	5,6%	3,1%	1,0%	2,0%	3,3%
PoznanHall2	8,3%	2,8%	1,1%	0,4%	0,4%
Shark	4,5%	1,2%	0,7%	1,2%	0,2%
Média	**5,8%**	**2,3%**	**1,8%**	**2,2%**	**3,3%**

Anexo B – Resultados (algoritmo do subcapítulo 4.2)

O anexo B contém os resultados do algoritmo descrito no subcapítulo 4.2. Os valores do BD-PSNR e BD-rate para os vídeos de resolução 1024×768 (Balloons, Kendo e Newspaper) estão discriminados nas Tabelas B.1, B.2, B.3, B.4 e B.5. Os desvios entre os percentuais dos tempos de codificação e os rácios alvo são apresentados na Tabela B.6. Os desvios em módulos são apresentados na Tabela B.7. As tabelas posteriores apresentam os resultados das sequências de vídeo de resolução 1920×1020 pixéis (PoznanHall2,GT-Fly, Dancer e Shark).

Sequências de vídeos de resolução 1024×768 pixéis

Tabela B.1 Resultados em termos de BD-rate e BD-PSNR do algoritmo do subcapítulo 4.2 para as sequências de vídeo de resolução 1024×768 pixéis (target 75%).

Target:75%		*Balloons*	*Kendo*	*Newspaper*	Média
Frames Textura.	BD-PSNR (dB)	-0,0443	-0,0500	-0,0740	**-0,0561**
	BD-BR (%)	1,2326	1,4110	2,1282	**1,5906**
Frames Profundidade	BD-PSNR (dB)	-0,0276	-0,0192	0,0059	**-0,0136**
	BD-BR (%)	0,9191	0,4827	-0,2245	**0,3925**
	T.C (%)	**75.5**	**76.6**	**78.4**	**76.8**

Tabela B.2 Resultados em termos de BD-rate e BD-PSNR do algoritmo do subcapítulo 4.2 para as sequências de vídeo de resolução 1024×768 pixéis (target 80%).

Target: 80%		*Balloons*	*Kendo*	*Newspaper*	Média
Frames Textura.	BD-PSNR (dB)	-0,0308	-0,0302	-0,0613	**-0,0408**
	BD-rate (%)	0,8578	0,8607	1,7426	**1,1537**
Frames Profundidade	BD-PSNR (dB)	-0,0359	-0,0161	0,0094	**-0,0142**
	BD-rate (%)	1,1877	0,4576	-0,3657	**0,4266**
	T.C (%)	**80.4**	**80.6**	**80.3**	**80.5**

Tabela B.3 - Resultados em termos de BD-rate e BD-PSNR do algoritmo do subcapítulo 4.2 para as sequências de vídeo de resolução 1024×768 pixéis (target 85%).

Target: 85%		*Balloons*	*Kendo*	*Newspaper*	Média
Frames Textura.	BD-PSNR (dB)	-0,0171	-0,0241	-0,0307	**-0,0240**
	BD-rate (%)	0,4764	0,6849	0,8717	**0,6777**
Frames Profundidade	BD-PSNR (dB)	-0,0198	-0,0177	0,0084	**-0,0097**
	BD-rate (%)	0,6005	0,4758	-0,3611	**0,2384**
	T.C (%)	**85.2**	**85.3**	**84.4**	**85.0**

Tabela B.4 - Resultados em termos de BD-rate e BD-PSNR do algoritmo do subcapítulo 4.2 para as sequências de vídeo de resolução 1024×768 pixéis (target 90%).

Target: 90%		*Balloons*	*Kendo*	*Newspaper*	Média
Frames Textura.	BD-PSNR (dB)	-0,0079	-0,0094	-0,0121	**-0,0098**
	BD-rate (%)	0,2286	0,2361	0,3255	**0,2634**
Frames Profundidade	BD-PSNR (dB)	-0,0106	-0,0234	0,0135	**-0,0068**
	BD-rate (%)	0,3119	0,6609	-0,5418	**0,1437**
	T.C (%)	**90.1**	**90.1**	**89.0**	**89.7**

Tabela B.5 - Resultados em termos de BD-rate e BD-PSNR do algoritmo do subcapítulo 4.2 para as sequências de vídeo de resolução 1024×768 pixéis (target 95%).

Target: 95%		*Balloons*	*Kendo*	*Newspaper*	Média
Frames Textura.	BD-PSNR (dB)	-0,0069	-0,0047	-0,0046	**-0,0054**
	BD-rate (%)	0,1880	0,1299	0,1406	**0,1528**
Frames Profundidade	BD-PSNR (dB)	0,0022	-0,0060	0,0105	**0,0022**
	BD-rate (%)	-0,1076	0,1695	-0,4323	**-0,1235**
	T.C (%)	**94.5**	**94.9**	**93.7**	**94.4**

Tabela B.6 - Desvios entre a percentagem do tempo de codificação e o rácio alvo, para os vídeos de resolução 1024×768 pixéis.

Vídeos/Rácios (%)	*75*	*80*	*85*	*90*	95
Balloons	-0,5%	-0,4%	-0,2%	-0,1%	0,5%
Kendo	-1,6%	-0,6%	-0,3%	-0,1%	0,1%
Newspaper	-3,4%	-0,3%	0,6%	1,0%	1,3%
Média	**-1,8%**	**-0,5%**	**0,0%**	**0,3%**	**0,6%**

:

Tabela B.7 - Desvios, em módulo, entre a percentagem do tempo de codificação e o rácio alvo, para os vídeos de resolução 1024×768 pixéis.

Vídeos/Rácios (%)	*75*	*80*	*85*	*90*	95
Balloons	0,5%	0,4%	0,2%	0,1%	0,5%
Kendo	1,6%	0,6%	0,3%	0,1%	0,1%
Newspaper	3,4%	0,3%	0,6%	1,0%	1,3%
Média	**1,8%**	**0,5%**	**0,4%**	**0,4%**	**0,6%**

Sequências de vídeos de resolução 1920×1020 pixéis

Tabela B.8 - Resultados em termos de BD-rate e BD-PSNR do algoritmo do subcapítulo 4.2 para as sequências de vídeo de resolução 1920×1020 pixéis (target 75%).

Target:75%		*Dancer*	*PoznanHall2*	*GtFly*	*Shark*	Média
Frames Textura.	BD-PSNR (dB)	-0,0411	-0,0872	-0,0273	-0,0214	**-0,0442**
	BD-rate (%)	-0,0411	4,4108	1,1474	0,6439	**1,9301**
Frames Profundidade	BD-PSNR (dB)	-0,0634	-0,0187	-0,0126	-0,0634	**-0,0288**
	BD-rate (%)	2,1473	-0,0726	0,2382	2,1473	**0,6700**
	T.C (%)	**81.2**	**82.2**	**79.1**	**74.6**	**79.3**

Tabela B.9 - Resultados em termos de BD-rate e BD-PSNR do algoritmo do subcapítulo 4.2 para as sequências de vídeo de resolução 1920×1020 pixéis (target 80%).

Target:80%		*Dancer*	*PoznanHall2*	*GtFly*	*Shark*	Média
Frames Textura.	BD-PSNR (dB)	-0,0314	-0,0843	-0,0290	-0,0160	**-0,0402**
	BD-rate (%)	1,1697	4,2652	1,2234	0,4748	**1,7832**
Frames Profundidade	BD-PSNR (dB)	-0,0143	-0,0395	-0,0132	-0,0536	**-0,0301**
	BD-rate (%)	0,2389	0,1602	0,2572	1,8450	**0,6253**
	T.C (%)	**82.6**	**83.9**	**81.6**	**79.5**	**81.9**

Tabela B.10 - Resultados em termos de BD-rate e BD-PSNR do algoritmo do subcapítulo 4.2 para as sequências de vídeo de resolução 1920×1020 pixéis (target 85%).

Target:85%		*Dancer*	*PoznanHall2*	*GtFly*	*Shark*	Média
Frames Textura.	BD-PSNR (dB)	-0,0290	-0,0773	-0,0265	-0,0181	**-0,0377**
	BD-rate (%)	1,0768	3,9413	1,1172	0,5353	**1,6676**
Frames Profundidade	BD-PSNR (dB)	0,0052	0,0141	-0,0119	-0,0540	**-0,0116**
	BD-rate (%)	-0,0773	-0,4549	0,1254	1,8894	**0,3707**
	T.C (%)	**86.3**	**84.9**	**85.8**	**84.2**	**85.3**

Tabela B.11 - Resultados em termos de BD-rate e BD-PSNR do algoritmo do subcapítulo 4.2 para as sequências de vídeo de resolução 1920×1020 pixéis (target 90%).

Target:90%		*Dancer*	*PoznanHall2*	*GtFly*	*Shark*	Média
Frames Textura.	BD-PSNR (dB)	-0,0200	-0,0573	-0,0228	-0,0098	**-0,0275**
	BD-rate (%)	0,7365	2,9171	0,9724	0,2920	**1,2295**
Frames Profundidade	BD-PSNR (dB)	-0,0079	-0,0271	-0,0071	-0,0260	**-0,0170**
	BD-rate (%)	0,1310	0,5526	0,1211	0,7239	**0,3821**
	T.C (%)	**90.9**	**89.2**	**90.5**	**88,0**	**89.7**

Tabela B.12 - Resultados em termos de BD-rate e BD-PSNR do algoritmo do subcapítulo 4.2 para as sequências de vídeo de resolução 1920×1020 pixéis (target 95%).

Target:95%		*Dancer*	*PoznanHall2*	*GtFly*	*Shark*	Média
Frames Textura.	BD-PSNR (dB)	-0,0146	-0,0517	-0,0121	-0,0089	**-0,0218**
	BD-rate (%)	0,5189	2,3631	0,5100	0,2668	**0,9147**
Frames Profundidade.	BD-PSNR (dB)	-0,0081	-0,0411	-0,0002	-0,0075	**-0,0142**
	BD-rate (%)	0,1275	1,3309	-0,1324	0,1498	**0,3690**
	T.C (%)	**95.4**	**93.8**	**95.3**	**91.3**	**93.9**

Tabela B.13 - Desvios entre a percentagem do tempo de codificação e o rácio alvo, para os vídeos de resolução 1920×1040 pixéis.

Vídeos/Rácios (%)	*75*	*80*	*85*	*90*	95
GT-Fly	-4,1%	-1,6%	-0,8%	-0,5%	-0,3%
Dance	-6,2%	-2,6%	-1,3%	-0,9%	-0,4%
PoznanHall2	-7,2%	-3,9%	0,1%	0,8%	1,2%
Shark	0,4%	0,5%	0,8%	2,0%	3,8%
Média	**-4,3%**	**-1,9%**	**-0,3%**	**0,3%**	**1,1%**

Tabela B.14 - Desvios, em módulo, entre a percentagem do tempo de codificação e o rácio alvo, para os vídeos de resolução 1920×1040 pixéis.

Vídeos/Rácios (%)	*75*	*80*	*85*	*90*	95
GT-Fly	4,1%	1,6%	0,8%	0,5%	0,3%
Dance	6,2%	2,6%	1,3%	0,9%	0,4%
PoznanHall2	7,2%	3,9%	0,1%	0,8%	1,2%
Shark	0,4%	0,5%	0,8%	2,0%	3,8%
Média	**4,5%**	**2,1%**	**0,8%**	**1,1%**	**1,4%**

Anexo C – Histogramas

Neste anexo são apresentados dois histogramas por rácio (*RácioDis* e *RácioBit)* para cada nível de profundidade da árvore de codificação. Os histogramas ilustram o número de ocorrências normalizado dos valores dos rácios *RácioDis* e *RácioBit*, nas CUs codificadas com o modo de predição 2N×2N/MSM e nas CUs codificadas com outros modos de predição. São ainda ilustradas as funções gaussianas utilizadas para caracterizar o comportamento de cada um dos histogramas.

Profundidade da árvore de codificação igual a zero

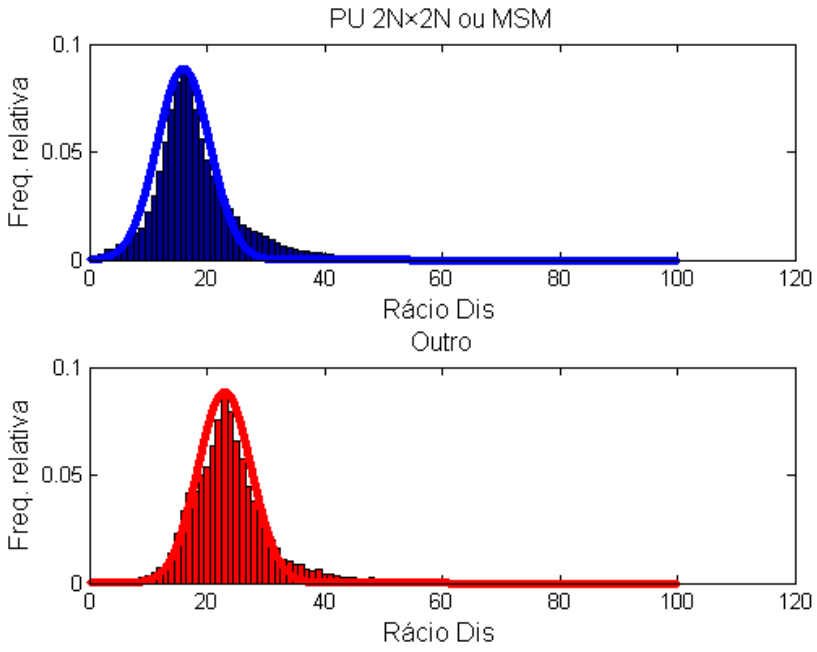


Fig. C.1 - Histograma para os valores do Rácio*Dis* na classe de CUs Be2N×2N (topo) e Not2N×2N (baixo), para profundidade da árvore de codificação igual a zero (*bins* 0.07).

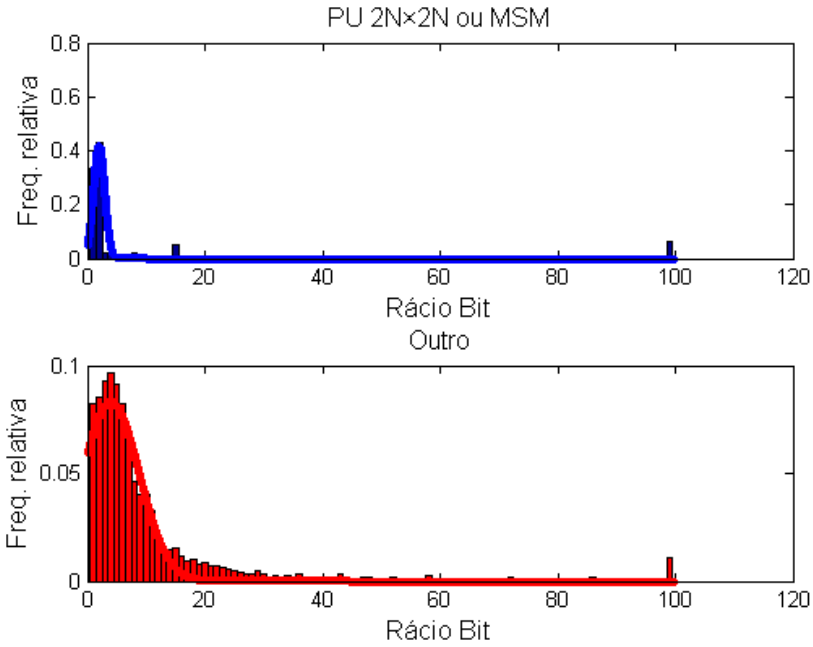


Fig. C.2 - Histograma para os valores do Rácio*Bit* na classe de CUs Be2N×2N (topo) e Not2N×2N (baixo), para profundidade da árvore de codificação igual a zero (*bins* 0.07).

Profundidade da árvore de codificação igual a um

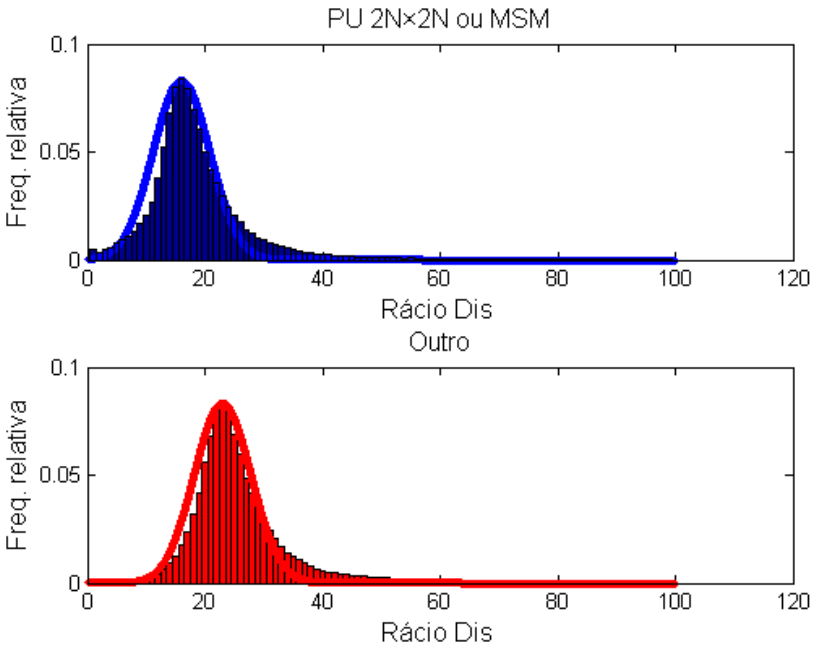


Fig. C.3 - Histograma para os valores do Rácio*Dis* na classe de CUs Be2N×2N (topo) e Not2N×2N (baixo), para profundidade da árvore de codificação igual a um (*bins* 0.07).

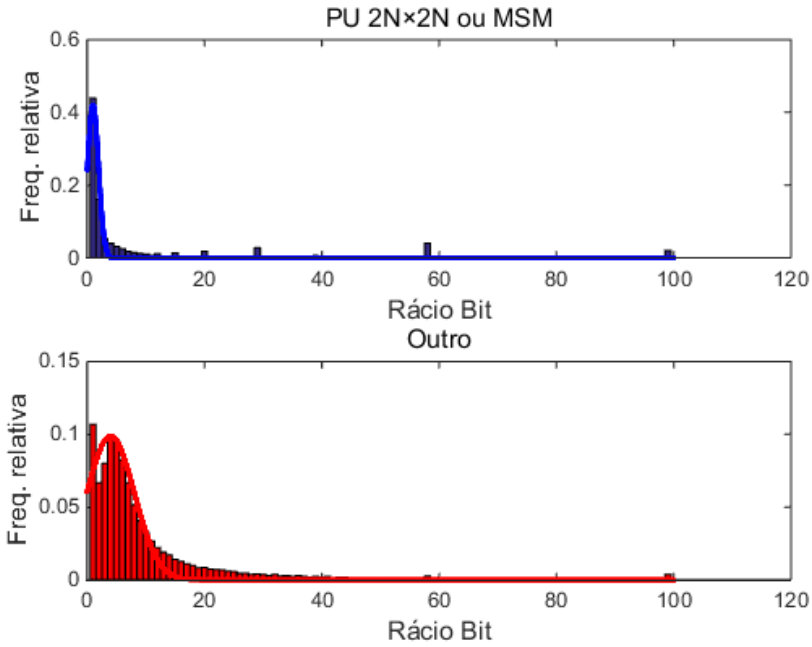


Fig. C.4 - Histograma para os valores do Rácio*Bit* na classe de CUs Be2N×2N (topo) e Not2N×2N (baixo), para profundidade da árvore de codificação igual a um (*bins* 0.07).

Profundidade da árvore de codificação igual a dois

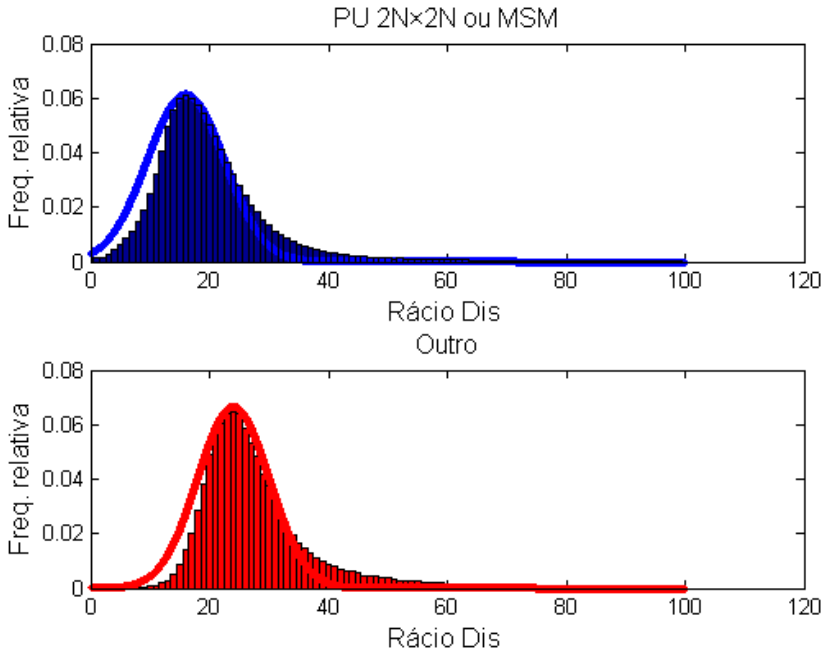


Fig. C.5 - Histograma para os valores do Rácio*Dis* na classe de CUs Be2N×2N (topo) e Not2N×2N (baixo), para profundidade da árvore de codificação igual a dois (*bins* 0.07).

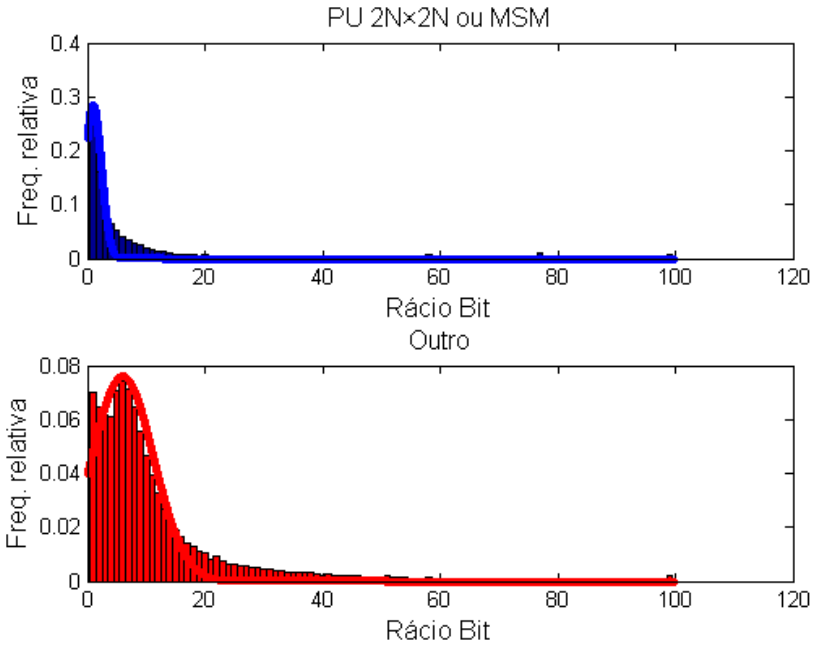
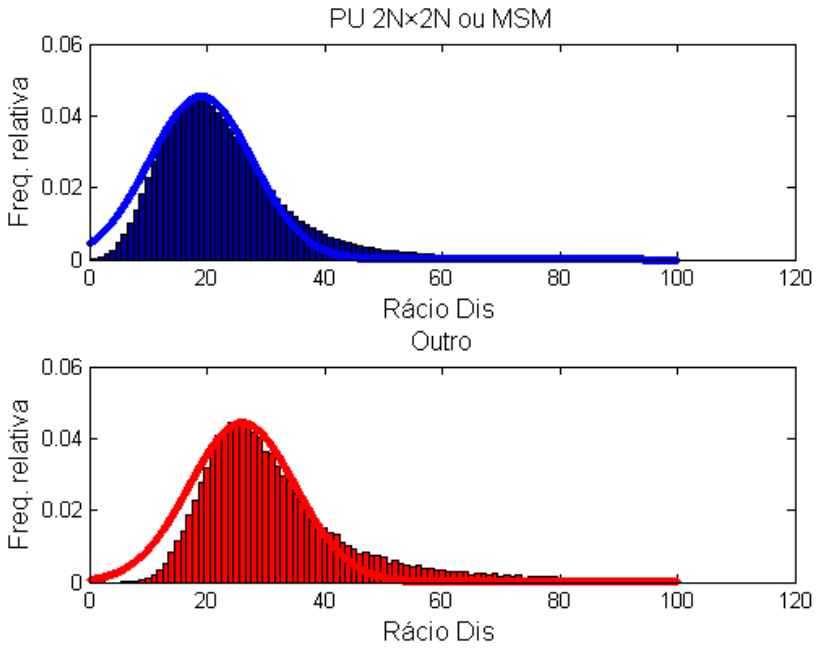


Fig. C.6 - Histograma para os valores do Rácio*Bit* na classe de CUs Be2N×2N (topo) e Not2N×2N (baixo), para profundidade da árvore de codificação igual a dois (*bins* 0.07).

Profundidade da árvore de codificação igual a três



:

Fig. C.7 - Histograma para os valores do Rácio*Dis* na classe de CUs Be2N×2N (topo) e Not2N×2N (baixo), para profundidade da árvore de codificação igual a três (*bins* 0.07).

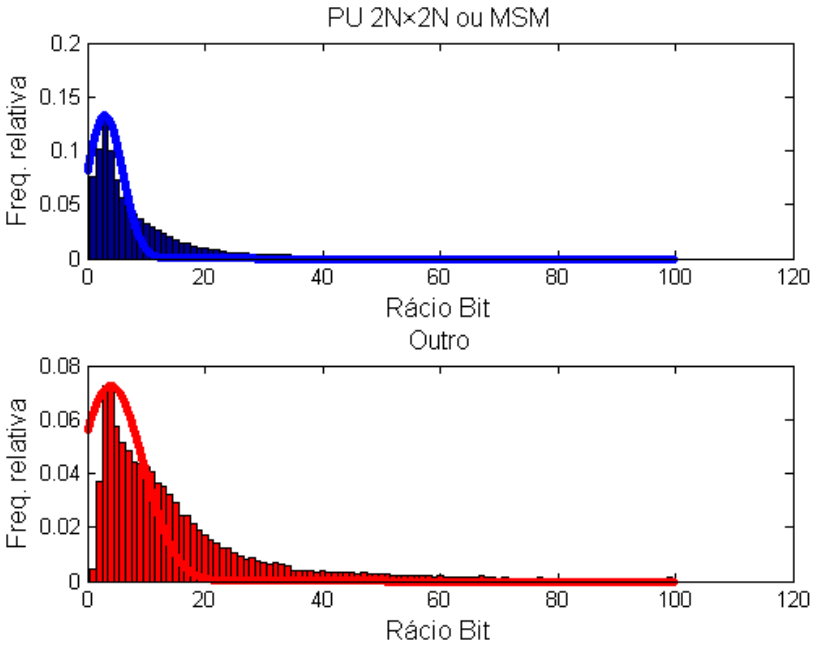


Fig. C.8 - Histograma para os valores do Rácio*Bit* na classe de CUs Be2N×2N (topo) e Not2N×2N (baixo), para profundidade da árvore de codificação igual a três (*bins* 0.07).

Anexo D – Configuração Random Access

A configuração *random access* (RA) é caracterizada pela utilização de uma estrutura hierárquica de tramas do tipo B, como ilustrado na Figura D.1. A numeração associada a cada trama representa a ordem de codificação e a ordem de exibição das tramas é da esquerda para a direita. A primeira trama é do tipo *Instantaneous Decoding Refresh* (IDR), estas tramas são codificadas só com modo *intra* e são pontos de acesso onde o vídeo é descodificado sem que exista dependência entre tramas. Na primeira camada encontra-se a trama IDR e uma trama *Generalized P or B* (GPB), esta ultima é caraterizada por só usar como tramas de referência, durante o processo de predição, as tramas com ordem de exibição inferior. Nas camadas superiores encontram-se as tramas do tipo B, as tramas pertencentes a última camada não são utilizadas como referência entre si.

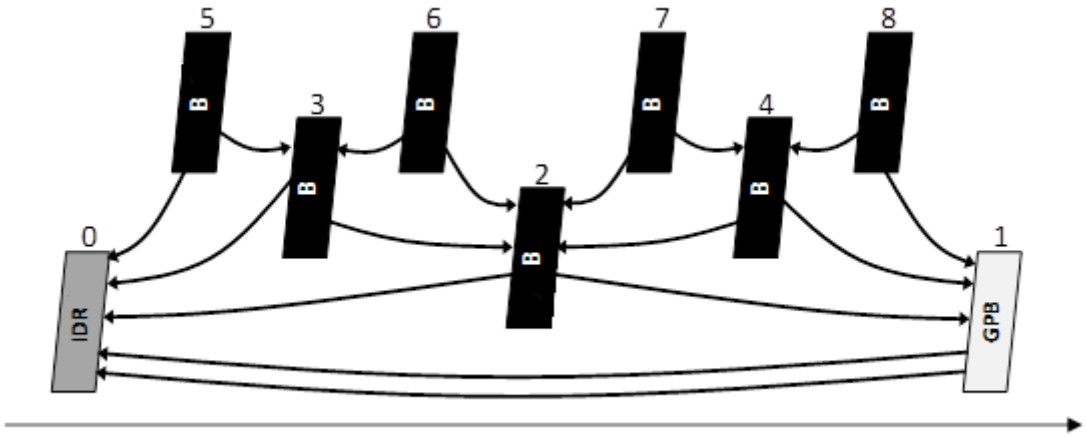


Fig. D.1 - Estrutura hierárquica de tramas do tipo B.