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

Processamento de mapas de profundidade para codificação e síntese de vídeo

Gizele Fernanda Abdon Júlio

Dissertação apresentada como requisito parcial para
conclusão do Mestrado em Informática

Orientador
Prof. Dr. Camilo Chang Dórea

Brasília
2017

Ficha Catalográfica de Teses e Dissertações

Está página existe apenas para indicar onde a ficha catalográfica gerada para dissertações de mestrado e teses de doutorado defendidas na UnB. A Biblioteca Central é responsável pela ficha, mais informações nos sítios:

http://www.bce.unb.br
http://www.bce.unb.br/elaboracao-de-fichas-catalograficas-de-teses-e-dissertacoes

Esta página não deve ser inclusa na versão final do texto.

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

Processamento de mapas de profundidade para codificação e síntese de vídeo

Gizele Fernanda Abdon Júlio

Dissertação apresentada como requisito parcial para
conclusão do Mestrado em Informática

Prof. Dr. Camilo Chang Dórea (Orientador)
CIC/UnB

Prof.a Dr.a Mylène Christine Queiroz de Farias
CIC/UnB

Dr. Eduardo Peixoto Fernandes da Silva
ENE/UNB

Prof. Dr. Bruno Luiggi Macchiavello Espinoza
Coordenador do Programa de Pós-graduação em Informática

Brasília, 17 de julho de 2017

Agradecimentos

A Deus, por sua benção e por iluminar minha trajetória, sempre se mostrando presente, ajudando nos momentos difíceis em minha vida e colocando as pessoas certas no meu caminho. À minha família, por sempre acreditarem em mim. À minha mãe e o ao meu pai que sempre me apoiaram nos desafios da vida. Quero agradecer especialmente à minha tia Gisselle que me ajudou de todas as formas nestes dois anos de mestrado. Também quero agradecer ao meu Tio Henrique, que da sua maneira sempre procurou ajudar-me como podia. Por isso, agradeço-os por serem a minha referência.

Aos meus amigos Danilo, Elton, Ennio, Dário, Daniel, Jeremias, Gustavo, Pergentino que durante estes dois anos mostraram sua amizade. Também foram ótimos amigos não somente nos estudos, mas também por me escutarem e compartilharem de suas companhias e experiências em tantos momentos! Especialmente aos meus amigos Vinícius e Iasmini que ajudaram em um momento crucial em que tudo parecia nebuloso e muito distante, deram-me forças para persistir e continuar estudando e trabalhando. A todos os meus professores durante o mestrado, que com muita sapiência conseguiram repassar seus conhecimentos e se mostraram também amigos, aconselhando e mostrando os caminhos futuros a seguir.

Agradeço em especial ao meu orientador Profº. Dr. Camilo Dórea por aceitar-me orientar e me propor este trabalho e por suas palavras compreensivas durante todos estes anos. Também quero agradecer ao Profº Dr. Bruno Macchiavello que também deu muito suporte e contribuições a este trabalho, sendo também um grande apoio durante todo o mestrado com sua disponibilidade e seu conhecimento. Agradeço também à Profª Drª Mylene Farias que além de ter sido uma ótima professora, também me apoiou e compartilhou suas experiências me dando forças, principalmente no início do mestrado. Gostaria também de dizer que sou grata ao Profº Dr. Eduardo Peixoto por ter aceitado participar desta banca de avaliação e por ter contribuído para a evolução deste trabalho com seus comentários e ajustes.

Resumo

Sistemas de múltiplas vistas são amplamente empregados na criação de vídeos 3D e de aplicações de *ponto de vista livre*. As múltiplas vistas, contendo vídeos de textura (cor) e profundidade, devem ser eficientemente comprimidas para serem transmitidas ao cliente e podem servir para síntese de vistas no receptor. Nesse contexto, a proposta deste trabalho é desenvolver um pré-processamento baseado no modelo de Distorção de Profundidade Admissível (ADD) que atue sobre os mapas de profundidade antes da codificação destes. Esse trabalho explora o modelo ADD e, adicionalmente, propõe a escolha e substituição dos valores de profundidade para aumentar a compressão dos mesmos de acordo com a distribuição dos blocos (*coding units*) empregados por codificadores padrões. Este pré-processamento tem como intuito a diminuição da carga de transmissão sem gerar perdas de qualidade na síntese da vista. Os histogramas dos mapas de profundidade após o pré-processamento são modificados, pois a alteração dos valores de profundidade dependerá da localização dos blocos. Os resultados mostram que é possível alcançar ganhos de compressão de até 13.9% usando o método da Mínima Variância no Bloco-ADD (ADD-MVB) sem a introdução de perdas por distorção e preservando a qualidade das imagens sintetizadas.

Palavras-chave: Mapas de Profundidade, Síntese de Vistas 3D, Agrupamento, Distorção Permitida de Profundidade(ADD)

Abstract

Multiview systems are widely used to create 3D video as well as in Free-Viewpoint Video applications. The multiple views, consisting of texture images and depth maps, must be efficiently compressed and trasmitted to clients where they may be used towards the synthesis of virtual views. In this context, the Allowable Depth Distorion (ADD) has been used in a pre-processing step prior to depth coding. This work explores ADD and, additionally, the choice of depth value to increase compression for transmission in accordance to the distribution of blocks (e.g., coding units) commonly employed by standardized coders without generating synthesis quality losses. Their histograms will be modified depending on the location and where the pixel belongs in the image. Experimental results show that our proposal can achieve compression gains of up to 13.9% applying the minimum variance method within a block, without introducing losses in terms of distorion and preserving synthesized image quality.

Keywords: depth maps, view synthesis 3D, clustering, Allowable Depth Distortion(ADD)

Sumário

Lista de Figuras

Lista de Tabelas

Lista de Abreviaturas e Siglas

3DV Vídeos 3D.

ADD Distorção de Profundidade Admissível.

ADD-M Distorção da profundidade Admissível-Mediana.

ADD-MVB Mínima Variância no Bloco-ADD.

CB Bloco de Codificação.

CTB Bloco de Codificação em Árvore.

CTU Unidade de Codificação em Árvore.

CU Unidade de Codificação.

DIBR Renderização Baseada em Imagem de Profundidade.

DMV Vetores de Movimentos de Disparidade.

DPCM Modulação de Código de Pulso Diferencial.

FVV Video de Ponto de Vista Livre.

HEVC *High Efficiency Video Coding*.

ISO/IEC Organização para Padronização Internacional/Comissão Eletrotécnica Internacional.

JVT Equipe de video conjunta.

MPEG Grupo de Especialistas em Imagens com Movimento.

MSE Erro Quadrático Médio.

MV Vetores de Movimentos.

MV-HEVC Multiview HEVC.

MVC Codificação de Video Multivista.

MVD Múltiplas Vistas mais Profundidade.

MVV Vídeo MultiVista.

PB Bloco de Predição.

PSNR Relação Sinal-Ruído de Pico.

PU Unidade de Predição.

RD Taxa-Distorção.

SAD Soma Absoluta das Diferenças.

SBrT XXXIV Simpósio Brasileiro de Telecomunicações e Processamento de Sinais.

TB Bloco de Transformação.

TMV Vetores de Movimentos Temporais.

TU Unidade de Transformação.

UC Unidade de Codificação.

Capítulo 1

Introdução

Diversas melhorias nas tecnologias de transmissão de dados e comunicação possibilitaram a disseminação de vários tipos de conteúdo multimídia de forma mais rápida ao usuário [Lee and Ho, 2009]. Atualmente, uma das mídias mais utilizadas são os vídeos. A propagação de vídeos tornou-se extensa, muitos usuários preferem assistir a vídeos a ler textos. Os serviços de *streaming* de vídeo ganharam um grande espaço no mercado nos últimos anos. Além disso, a TV Digital também teve seu desenvolvimento tanto no Brasil quanto no mundo e, por meio desta tecnologia, é possível ao usuário ser mais exigente em relação à qualidade da mídia recebida.

Simultaneamente à disseminação dos vídeos e da Tv Digital, uma das tecnologias que tem atraído bastante atenção nos últimos anos é a aplicação de vídeos 3D em diversos segmentos do mercado como em jogos, filmes, entre outros. Com uso da Tv Digital criou-se a possibilidade de se ter conteúdo em 3D além da possibilidade de interação do cliente com este conteúdo, utilizando o canal de retorno que já é previsto nos recursos da mesma. Por intermédio de sistemas de múltiplas vistas, em que a cena pode ser capturada de vários pontos diferentes, pode-se oferecer ao usuário a capacidade de interagir com aplicações de Video de Ponto de Vista Livre (FVV) (do inglês, *Free Viewpoint Video*). Sistemas FVV permitem a seleção de diferentes perspectivas de visualização da cena dentro de uma variedade limitada de possíveis escolhas [Merkle et al., 2007].

Para visualizar um conteúdo 3D é possível adotar *displays* estereoscópicos, em que é necessário o uso de óculos para perceber a profundidade ou auto-estereoscópicos, em que o *display* emite várias imagens, a depender da posição do telespectador, e não é necessário o uso de óculos [Tech et al., 2016a]

Esses sistemas têm sido objeto de estudo de associações internacionais como o Grupo de Especialistas em Imagens com Movimento (MPEG) (do inglês, *Moving Picture Experts Group*) da Organização para Padronização Internacional/Comissão Eletrotécnica Internacional (ISO/IEC) (do inglês, *International Organization for Standardization/International*

Electrotechnical Commission). Tecnologias como a 3DTV e o sistema de FVV [Kauff et al., 2007] tornaram-se uma realidade. Uma característica que ambas as aplicações compartilham é o uso do sistema de Vídeo MultiVista (MVV)(do inglês, *MultiView Video*) [Merkle et al., 2007]. A geração de conteúdo 3D utiliza, essencialmente, os sistemas de múltiplas vistas e técnicas de Renderização Baseada em Imagem de Profundidade (DIBR) (do inglês, *Depth Image Based Rendering*).

Para a transmissão de um conteúdo proveniente de um sistema de múltiplas vistas, o volume de dados torna-se muito maior quando comparado aos conteúdos gerados por um sistema convencional de captura. Nos sistemas de múltiplas vistas há a transferência das sequências de imagens geradas a partir de várias câmeras para uma mesma cena. Por conta disso, grupos como MPEG e Equipe de Video ConjuntaEquipe de video conjunta (JVT) (do inglês, *Joint Video Team*) começaram a padronização de um codificador específico para este ambiente, a Codificação de Video Multivista (MVC) (do inglês, *Multiview Video Coding*), favorecendo-se das redundâncias espacias, temporais e entre as sequências de câmeras existentes nesse tipo de sistema [Lee and Ho, 2009].

Um dos benefícios dos sistemas de Múltiplas Vistas mais Profundidade (MVD) (do inglês, *Multiview-Video-Plus-Depth*) é a possibilidade de se criar vistas intermediárias utilizando a técnica de DIBR e as informações das sequências de textura do vídeo e a respectiva sequência de profundidade. Esta técnica permite uma redução na quantidade de câmeras a serem enviadas para o usuário, pois é possível sintetizar vistas virtuais (posições não enviadas) a partir de câmeras reais (enviadas) utilizadas como referência. Ao utilizar esta estratégia, pode-se produzir um vídeo 3D a partir do conteúdo derivado de uma sequência dada por uma única câmera, pois gera-se uma segunda vista com o uso de algoritmos específicos, proporcionando ao usuário a sensação de profundidade. Essa redução do envio de câmeras também e possível para sistemas FVV, pois cria-se uma câmera virtual para uma posição escolhida em que não foi enviada uma câmera real [Merkle et al., 2007].

Apesar de algumas abordagens, como a técnica DIBR, já diminuírem significantemente o volume de dados a ser transmitido, ainda assim é gerada uma sobrecarga nas redes de comunicação, o que ainda dificulta o uso regular de aplicações 3D e de FVV. Uma abordagem estudada recentemente que contribui para a diminuição da sobrecarga é a utilização de métodos de codificação mais sofisticados. A codificação de vídeo é essencial para qualquer aplicação que possua restrições em relação à capacidade de armazenamento ou de largura de banda para a transmissão [Richardson, 2011].

Portanto, em razão do grande volume de dados gerados, guardados e enviados em um sistema de múltiplas vistas, técnicas eficientes de compressão de dados devem ser exploradas também [Merkle et al., 2007]. Para isso, muitos esforços em pesquisas que

visam a alcançar maior compressão sobre esses dados têm sido feitos, buscando aplicar diferentes técnicas como o uso da correlação entre os dois vídeos ou em alguns casos, aproveitando a redundância existente nos mapas de profundidade, conforme apresentado nos trabalhos de [Zhao et al., 2011], [Zhang et al., 2014], [Zhang et al., 2015] e [Zhang et al., 2016].

Quando há a criação de vistas virtuais para a concepção de profundidade em aplicações 3D, os vídeos de textura da câmera de referência e seu respectivo vídeo contendo o mapa de profundidade são transmitidos simultaneamente com o arquivo de configuração da câmera de referência. Neste caso, é possível realizar a síntese das vistas virtuais para o olho esquerdo ou direito, transladando-se a imagem de referência para uma segunda posição, criando a ilusão da profundidade da cena e, consequentemente, dando maior realismo à cena [Fehn, 2004]. As técnicas utilizadas neste processo são descritas no Capítulo 2.

Em razão das várias aplicações citadas e a crescente demanda pelo uso de tecnologias de vídeo tridimensional, o formato de vídeo 3D com múltiplas vistas mais profundidade tornou-se mais popular, exigindo assim que uma maior compressão e uma menor complexidade de codificação fossem alcançados para promover ainda mais o seu uso .

Baseado nos fatores motivadores, isto é, dado este novo paradigma de televisão e a possibilidade de interação do usuário com o sistema, é fundamental realizar estudos relativos à capacidade de compressão dos dados para a síntese de múltiplas vistas, tornando o sistema possível de utilização nos meios de transmissão existentes.

Nesse aspecto, este trabalho tem como intuito mitigar ainda mais o volume de dados a ser transmitido ao investigar a capacidade de compressão dos mapas de profundidade que são necessários para a síntese de vista. Este objetivo pode ser alcançado por meio de um pré-processamento antes da compressão dos dados.

1.1 Apresentação do Problema

Em uma simples abordagem, o envio de vídeos pode ser feito de maneira separada para a criação de um vídeo 3D no usuário final. No entanto, as pesquisas do projeto ATTEST desenvolvido pela [Sociedade Européia de Tecnologia da Informação, 2004] propuseram utilizar uma ideia mais flexível, em que utilizaria a transmissão de somente um vídeo de textura com sua respectiva informação de profundidade e com isso seria possível tanto a criação de uma segunda vista, quanto a projeção do vídeo a partir de outro ângulo de visualização. As fases do processo de criação dos vídeos 3D até a distribuição ao usuário final podem ser resumidas na Figura 1.1 [Fehn, 2004].

No processo de criação dos vídeos 3D, há duas abordagens para a obtenção da medida de profundidade dos objetos da cena: baseada em *Hardware* ou em *Software*. A primeira

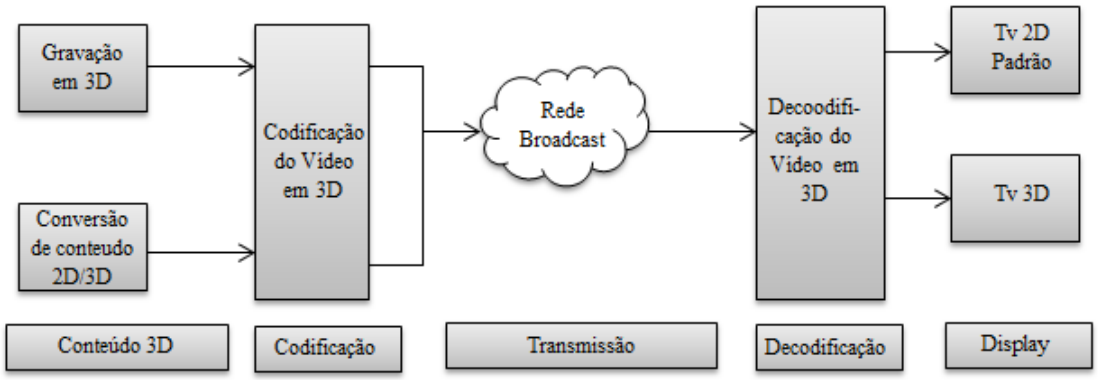


Figura 1.1: Ilustração do processo desde a criação do vídeo 3D até a entrega e reconstrução da informação do lado do cliente. Figura adaptada de Fehn [Fehn, 2004].

abordagem utiliza câmeras com emissores de raios infravermelhos na captura do vídeo para a aquisição simultânea da informação de profundidade. Este tipo de aquisição possui maior acurácia nos resultados, porém a captura da cena precisa ser feita em ambientes internos, restringindo os ambientes de uso da câmera [Lee and Ho, 2009]. Outra abordagem possível é a transformação de vídeos originalmente gravados em 2D para vídeos 3D, utilizando algoritmos para a aproximação de profundidade da cena [Fehn, 2004]. Entretanto, para se aplicar esta abordagem tem-se a necessidade de retificação das sequências de vídeos e há instabilidade ao redor das bordas descontínuas do resultado gerado.

Depois da compressão e transmissão do conteúdo, um sistema eficiente para a decodificação e síntese de vistas precisa estar disponível para renderizar o vídeo e mostrá-lo em um *display* apropriado 3D [Lee and Ho, 2009].

O grande problema para estes tipos de aplicações é a capacidade de transmissão das tecnologias de telecomunicações, pois a exigência de qualidade e resolução para vídeos tem crescido continuamente. As tecnologias têm avançado cada vez mais para a utilização de resoluções maiores de vídeos, a exemplo dos televisores 4K que possuem resoluções de 3840x2160 *pixels*, fazendo o volume de dados aumentar significativamente.

No processo de entrega de vídeos 3D, estes precisam ser codificados e transmitidos em uma infraestrutura convencional de TV 2D. Para que se alcancem menores volumes de dados e ao mesmo tempo se mantenha a qualidade do vídeo visualizado, muitas técnicas têm sido refinadas, com o intuito de proporcionar maiores taxas de compressão. A compressão deve ser realizada sobre o vídeo de textura e de profundidade e enviado sob a infraestrutura de transmissão 2D existente. Com isso, é possível utilizar a família de compressores MPEG. No caso deste trabalho, utiliza-se o codificador de dados *High Efficiency Video Coding* (HEVC) [ITU-T, 2014] [Fehn, 2004].

Os trabalhos abordados no Capítulo 3 realizam filtragens ou pré-processamentos sobre

as imagens ou vídeos de profundidade antes da aplicação da compressão, buscando alcançar maiores taxas de compressão. Ao mesmo tempo é necessário investigar o impacto da qualidade das imagens sintetizadas em relação às suas taxas de compressão e avaliar o custo-benefício de se diminuir a taxa de bits dos mapas de profundidade.

Após a codificação e transmissão da informação, é necessário reconstruir os dados do lado do cliente para que este visualize o conteúdo 3D ou escolha o ângulo de visualização da cena que preferir ao utilizar aplicações do tipo FVV.

1.2 Objetivo

O objetivo deste trabalho é realizar um pré-processamento nos mapas de profundidade com o intuito de alcançar maiores níveis de compressão sobre os mapas de profundidade que são as informações necessárias para a projeção de vistas virtuais e, consequentemente, a geração de síntese a partir de múltiplas vistas. Este pré-processamento visa a não alteração da imagem sintetizada tanto por parte do algoritmo de pré-processamento quanto por parte da compressão de dados. Esta compressão é realizada em um modo considerado sem perdas (*lossless*)

1.2.1 Objetivos Específicos

- Implementar a abordagem de pré-processamento proposta por [Zhang et al., 2016], realizando o agrupamento dos valores de profundidades para um mesmo intervalo ADD com o objetivo de se diminuir a entropia do mapa de profundidade.
- Propor uma abordagem que consiga alcançar um nível de compressão maior quando comparado ao da técnica do item anterior, utilizando a estrutura do codificador como base para a proposta. Determinar o melhor tamanho de bloco para realizar este pré-processamento. Esta proposta foi recentemente publicada em conferência de âmbito nacional XXXIV Simpósio Brasileiro de Telecomunicações e Processamento de Sinais (SBrT).

1.3 Organização do Trabalho

O capítulo 2 faz um levantamento do referencial teórico, mostrando conceitos e desenvolvendo os contextos dentro dos quais algumas premissas são utilizadas no capítulo seguinte.

O capítulo 3 apresenta alguns artigos relacionados às abordagens implementadas, mostrando as técnicas que foram utilizadas, além de outros trabalhos relacionados que serviram para o entendimento da problemática e dos desafios encontrados nesta área.

Já o capítulo 4 define a metodologia adotada e discorre sobre as técnicas escolhidas para o desenvolvimento do trabalho. Em seguida, o capítulo 5 mostra os resultados obtidos com as abordagens feitas sobre o problema. Já no capítulo 6 são feitas as conclusões, mostrando as vantagens e desvantagens das abordagens adotadas.

Capítulo 2

Referencial Teórico

Neste capítulo são descritos os fundamentos teóricos nos quais este trabalho baseou-se, mostrando conceitos e explorando as técnicas necessárias para o ambiente abordado, tanto do desenvolvimento do trabalho quanto das tecnologias utilizadas para a criação de síntese de vistas. Fundamentos relativos à codificação e compressão de vídeos, finalidade para a qual o pré-processamento nos mapas de profundidade é aplicado, também fazem parte deste capítulo.

2.1 Modelo de Câmera

Nesta seção são apresentados conceitos relativos ao modelo da câmera e suas propriedades, mostrando a relação das informações do espaço real tridimensional com sua representação em uma imagem 2D por meio de equações de transformação de suas coordenadas.

Quando um sistema de múltiplas vistas é configurado para a captura das imagens de textura e de profundidade de uma cena a partir de diferentes perspectivas, inicialmente, é necessário transformar as informações do sistema de coordenadas do espaço real tridimensional para o sistema de coordenadas da imagem. Para isso, utilizam-se as informações intrínsecas e extrínsecas de cada uma das câmeras posicionadas para a captura.

Portanto, é importante compreender algumas particularidades dos sistemas de coordenadas utilizados e como é realizada a transformação entre os sistemas de coordenadas da câmera, da imagem e do espaço real 3D. Características específicas para a projeção e síntese de uma vista virtual envolve associar os parâmetros corretos a depender das posições das câmeras de referência.

Na estrutura de um sistema de múltiplas vistas existe um único sistema de coordenadas do espaço real tridimensional, o qual é independente da posição da câmera. Já para cada câmera existe um sistema de coordenadas próprio de câmera e de imagem dependente da sua posição no arranjo [Tian et al., 2009]. Estas informações identificam a posição em que

a câmera está e são representadas por matrizes específicas que indicam o mapeamento correto dos pontos localizados do espaço real 3D para o plano da imagem da respectiva câmera [Hartley and Zisserman, 2004].

É utilizado como base o modelo de câmera *pinhole* ou "buraco de alfinete". Neste modelo básico existem três sistemas de coordenadas que estão relacionadas por meio dos parâmetros intrínsecos e extrínsecos da câmera: o sistema de coordenadas do espaço real tridimensional, o sistema de coordenadas da câmera e o sistema de coordenadas da imagem (2D).

Dado um ponto de entrada de luz $P = [X_w \ Y_w \ Z_w]^T$ no espaço real tridimensional conforme mostrado na Figura 2.1, este ponto é representado no sistema de coordenadas da câmera como $P' = [X_c, Y_c, Z_c]^T$ e projetado no plano da imagem como $p = [u \ v]^T$, sendo u e v as projeções nos eixos X e Y [Tian et al., 2009]. Os planos da câmera e da imagem são paralelos e o plano da imagem e posicionado à frente do plano da câmera por convenções matemáticas.

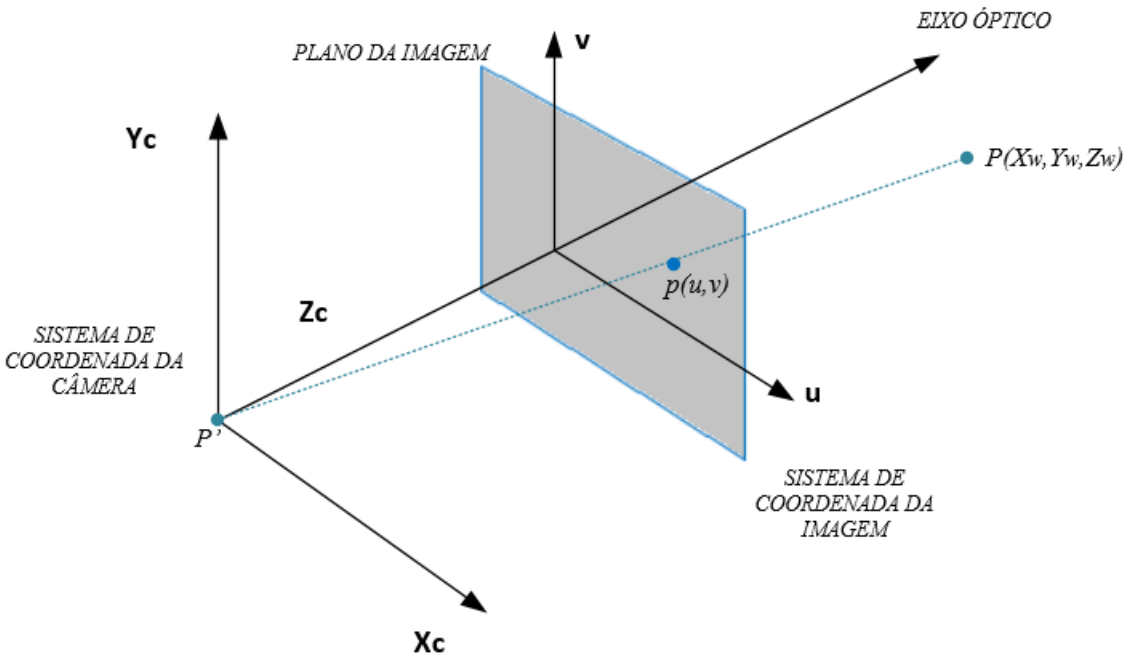


Figura 2.1: Os três planos associados à aquisição de uma cena real para o modelo de câmera *pinhole*. Figura adaptada a partir de [Hartley and Zisserman, 2004] e [Tian et al., 2009].

Para descrever o relacionamento entre os sistemas de coordenadas e obter a localização dos objetos no plano da imagem, dois conjuntos de parâmetros são definidos: a matriz intrínseca A e a matriz extrínseca E. A matriz intrínseca relaciona a transformação do

sistema de coordenadas de câmera para o sistema de coordenadas da imagem, isto para cada câmera em um sistema de múltiplas vistas. Esta matriz é descrita como:

$$A = \begin{bmatrix} f_x & 0 & o_x \\ 0 & f_y & o_y \\ 0 & 0 & 1 \end{bmatrix} \tag{2.1}$$

na qual os valores de f_x e f_y são os comprimentos focais nos eixos das abcissas e das ordenadas, respectivamente. O comprimento focal é a distância do plano da imagem até a abertura da câmera e o par (o_x,o_y) é o ponto de *offset*, isto é, a diferença entre o centro do sistema de coordenadas da câmera e o sistema de coordenadas da imagem. Entretanto, no contexto de compressão de vídeos, a origem do plano da imagem normalmente se localiza na região acima e à esquerda e, geralmente, assume um valor não zero [Tian et al., 2009].

A matriz extrínseca representa a transformação entre o sistema de coordenadas do espaço real 3D para o sistema de coordenadas da câmera. Esta matriz é descrita como $E = [\Psi|\tau]$, sendo a concatenação de duas matrizes: Ψ que é a matriz de rotação e tem tamanho 3×3 e τ que é a matriz de translação com tamanho 3×1.

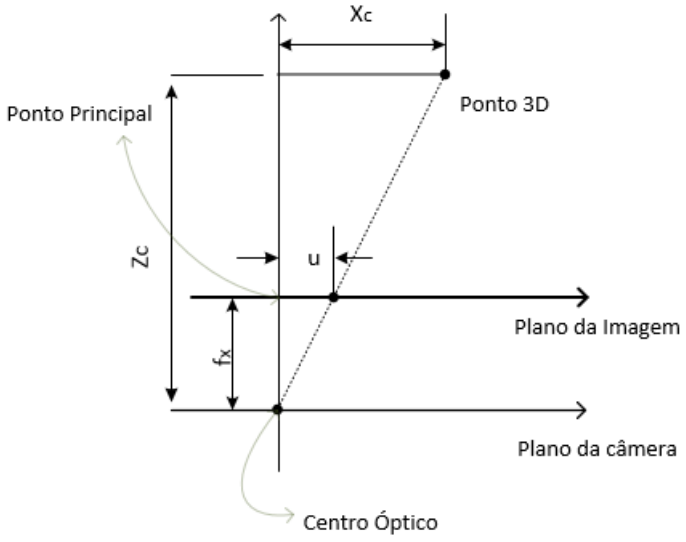


Figura 2.2: Relação geométrica entre os sistemas de coordenadas. Figura baseada em [Tian et al., 2009].

As posições dos pontos de coordenadas e da distância focal podem ser observadas na Figura 2.2. A relação entre o ponto da imagem de coordenada (u , v) e o ponto no sistema de coordenada da câmera (X_c, Y_c) pode ser dada por meio do uso da distância focal f_x e a transformação entre esses dois sistemas de coordenadas é dada pela equação [Tian et al., 2009] [Forsyth and Ponce, 2002]

$$\frac{u}{X_c} = \frac{f_x}{Z_c}. \tag{2.2}$$

Para o caso em que o valor de offset é diferente de zero, as equações se tornam

$$\begin{aligned} u &= \frac{f_x.X_c}{Z_c} + o_x \\ v &= \frac{f_x.Y_c}{Z_c} + o_y. \end{aligned} \tag{2.3}$$

É possível relacionar o sistema de coordenadas da imagem com o sistema de coordenadas da câmera usando a matriz intrínseca A conforme

$$Z_c \cdot \begin{pmatrix} u \\ v \\ 1 \end{pmatrix} = A \cdot \begin{pmatrix} X_c \\ Y_c \\ Z_c \end{pmatrix} = \begin{pmatrix} f_x & 0 & o_x \\ 0 & f_y & o_y \\ 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} X_c \\ Y_c \\ Z_c \end{pmatrix} \tag{2.4}$$

Para projetar um ponto do espaço real tridimensional para o sistema de coordenadas da câmera é utilizada a relação dada pela equação

$$\begin{pmatrix} X_c \\ Y_c \\ Z_c \end{pmatrix} = \Psi \cdot \begin{pmatrix} X_w \\ Y_w \\ Z_w \end{pmatrix} \cdot \tau \tag{2.5}$$

2.2 Sistema de Cores

Esta seção aborda os sistemas de cores mais usuais e, em particular, detalha o sistema de cor YCrCb que é amplamente empregado na representação dos sistemas de múltiplas vistas.

A luz branca é composta por um espectro de seis regiões que varia desde a cor violeta até a vermelha, sendo a passagem entre as cores dessas regiões suave. A percepção de cor dos objetos no mundo real dependerá da composição do material destes.

Quando o objeto reflete todas as cores ele é visto como branco e o contrário ocorre quando não reflete nenhuma, sendo visto como preto. Se a luz é acromática então não há reflexão de cor e seu único atributo é a intensidade. Um exemplo disto é o visto em um televisor preto e branco. O termo escala de cinzas, que será utilizado posteriormente, é uma medida da intensidade de luminância que varia de preto até o branco [Gonzalez and Woods, 2010].

Os modelos de cores servem para especificar as cores segundo um determinado padrão e podem ser definidos a partir de um sistema de coordenadas, sendo uma cor específica representada por um ponto dentro deste sistema. Este trabalho utiliza o modelo de cor

YCrCb que é obtido a partir do modelo de cores RGB e que são abordados nas subseções 2.2.1 e 2.2.2. Mais informações sobre estes sistemas podem ser encontrados em [Gonzalez and Woods, 2010].

2.2.1 Modelo RGB

O modelo RGB é o mais utilizado para os monitores e para várias câmeras. Este modelo é baseado no sistema de coordenadas cartesianas [Gonzalez and Woods, 2010]. Um objeto representado neste modelo possui a quantidade relativa a cada uma das componentes (vermelho, verde e azul) para criar a cor do objeto.

Capturar uma imagem em RGB significa filtrar cada componente de cor por um vetor de sensores separadamente. Em *displays* cada componente de cor é iluminado de forma separada e a distância causa a impressão de sobreposição destas, gerando a cor desejada [Richardson, 2011].

2.2.2 YCrCb

Outro modelo de representação de cores é o Y:Cr:Cb. Como o sistema visual do ser humano é mais sensível à luminância do que a cores, representar este componente com mais resolução é o mais apropriado.

Portanto, o espaço de cores Y:Cr:Cb é uma maneira de representar as imagens coloridas, em que irá separar a luminância representada por Y das cores, Cr e Cb. Y pode ser calculada a partir dos componentes RGB como a seguinte equação:

$$Y = K_r R + K_g G + K_b B, \tag{2.6}$$

sendo K o valor dos pesos das cores para formar a componente em tons de cinzas [Richardson, 2011]. As outras componentes são as que representam as cores e podem ser calculadas também a partir dos componentes RGB, conforme as seguintes equações:

$$\begin{aligned} C_r &= R \ - Y \\ C_b &= B \ - Y \\ C_g &= G \ - Y. \end{aligned} \tag{2.7}$$

Apesar de possuir três componentes de cores calculadas, apenas duas, (C_r e C_b) são enviadas ao usuário, sendo estas componentes os distanciamentos do cinza para as cores vermelha e azul respectivamente.

A vantagem desta representação sobre o modelo RGB é que as componentes de cores podem ser dizimadas, sendo sub-amostradas em relação ao Y, uma vez que a sensibilidade da visão humana é menor para as cores e portanto a imagem não sofre forte degradação de

qualidade. Esta é uma maneira de se aplicar compressão sobre as imagens quando representadas por esse modelo, pois as componentes de cores podem possuir menos informação do que a componente de luminância [Richardson, 2011].

Por vezes o usuário poderá não notar diferença alguma entre as imagens representadas pelo modelo RGB e YCrCb. As imagens podem ser transformadas de um modelo para o outro por motivos de compressão de informação e recuperados de volta para a visualização do usuário. As transformações aplicadas para a mudança entre os modelos utilizam as seguintes equações:

$$\begin{aligned} Y &= 0.299\,R - 0.587\,G + 0.114\,B \\ C_b &= 0.564\,(B - Y) \\ C_r &= 0.713\,(R - Y) \end{aligned} \tag{2.8}$$

e

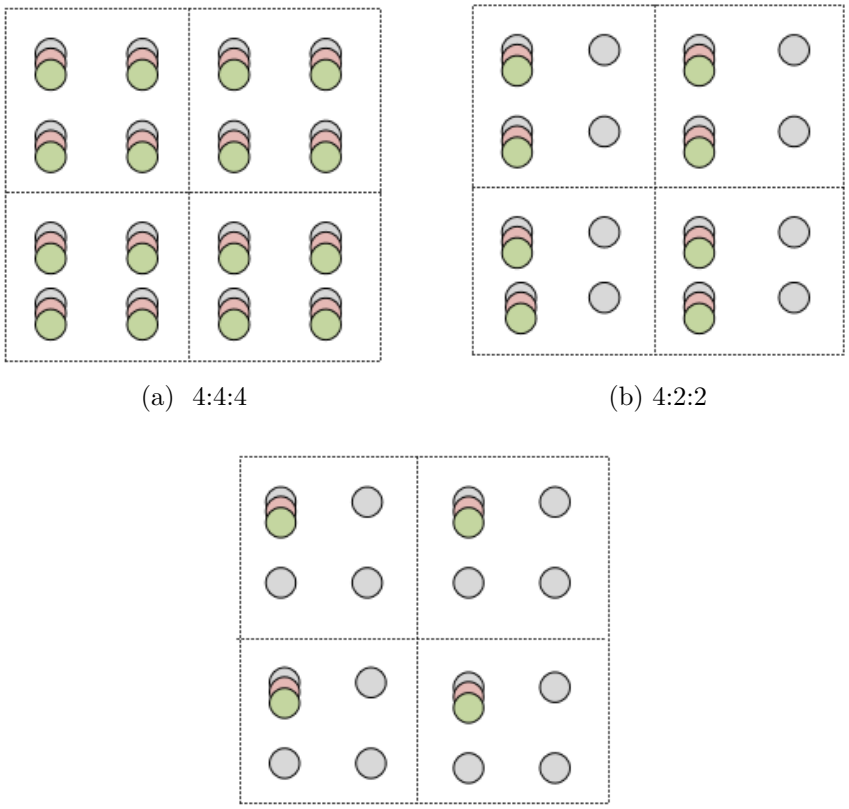
$$\begin{aligned} R &= Y + 1.402\,C_r \\ G &= Y - 0.344\,C_b - 0.714\,C_r \\ B &= Y + 1.772\,C_b. \end{aligned} \tag{2.9}$$

O modelo YCrCb pode ser subamostrado para que se tenha uma imagem com menos dados porém com a quantidade de informação suficiente para o usuário. Este modelo envia apenas as componentes de luminância e as de crominância (Cr e Cb), uma vez que a componente Cg pode ser recuperada a partir das outras duas. Neste cenário as componentes Cr e Cb podem ser dizimadas de acordo com a demanda por qualidade.

Quando a imagem possui integralmente suas três componentes, sem nenhuma dizimação, o padrão é chamado de YCrCb 4:4:4. Neste padrão, para cada quatro amostras de luminância existem quatro amostras para cada componente Cr e Cb, conforme ilustrado na Figura 2.3a.

Quando a subamostragem resulta em quatro amostras de luminância (Y) e apenas duas amostras para cada componente de cor, o padrão é chamado 4:2:2 ou YUY2. Neste padrão, as componentes de cor contêm metade da resolução horizontal (em relação a Y) conforme a Figura 2.3b [Richardson, 2011]. O YUY2 é usado em aplicações que exigem uma alta qualidade visual de cores .

Já o padrão 4:2:0, mostrado na Figura 2.3c, é chamado de modelo de "12 bits por pixel", pois para sua representação são necessárias 4 amostras da componente Y e uma amostra de cada componente Cr e Cb, totalizando $6 \times \frac{8}{4}$ bits = 12 bits por pixel. Portanto, neste padrão, as componentes de cor possuem a metade da resolução de Y, tanto na direção horizontal quanto vertical.



(a) 4:4:4 (b) 4:2:2

(c) 4:2:0

Figura 2.3: Subamostragem no modelo de cores YCrCb.

2.3 Sistema Multi-Vista

Esta subseção tem como objetivo apresentar alguns conceitos relativos aos sistemas de múltiplas vistas e mapas de profundidade, além de explorar técnicas para este ambiente como a ideia de estereoscopia que é usada para criar a sensação de profundidade para uma determinada cena.

Para prover as entradas para um sistema 3D ou um sistema FVV, um sistema de múltiplas vistas é configurado por meio de várias câmeras que capturam a cena de diferentes perspectivas.

Um exemplo de um cenário para o sistema de múltiplas vistas é ilustrado na Figura 2.4. Dispondo-se câmeras sobre um arco, de forma que estas estejam equidistantes uma das outras, a cena pode ser capturada a partir de diferentes posições. Caso uma vista desejada pelo usuário não exista, esta pode ser sintetizada a partir das câmeras de referência existentes.

Alguns pontos são importantes neste contexto, principalmente no momento de captura de cena e do pós-processamento dos conteúdos [Lee and Ho, 2009]:

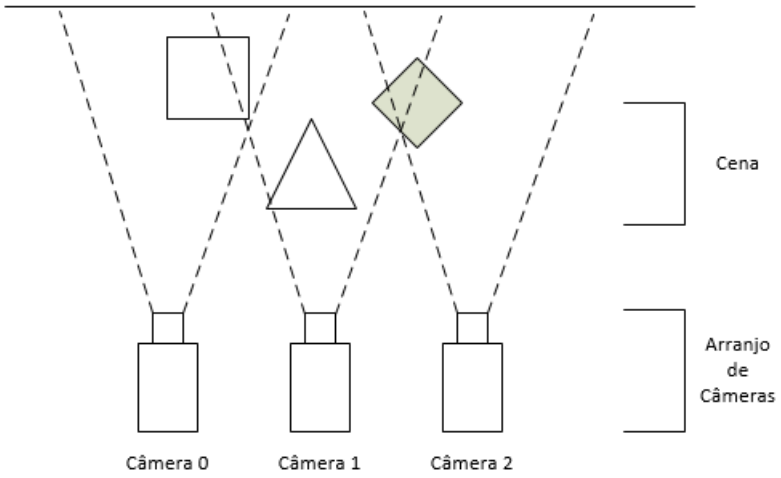


Figura 2.4: Exemplo de disposição de câmeras em um sistema com Múltiplas Vistas 1D paralelo.

- O arranjo das câmeras deve ser 1D paralelo (outros casos existem, contudo, para este trabalho, foi utilizado este tipo de arranjo).
- Quando aplicado o processo de retificação às imagens capturadas, estas devem estar paralelas, variando em apenas um dos eixos. Quando não estão alinhadas o suficiente, torna-se mais difícil extrair a informação da profundidade como descrito na seção 2.3.1.
- A sincronização temporal das câmeras é imprescindível.
- Todas as sequências devem ter suas cores corrigidas usando um método apropriado para tal. Mesmo utilizando um mesmo modelo de câmera, isto não garante que haja consistência entre as vistas.
- As informações de calibração das câmeras devem ser precisas para se obter as matrizes intrínsecas e extrínsecas das câmeras que são necessárias para as projeções posteriores dos píxeis no momento da sintetização das imagens virtuais.

Neste contexto de sintetização, alguns objetos podem estar visíveis a partir de algumas câmeras e parcialmente encobertos por outras. Dessa forma, é possível que surjam buracos de oclusão a depender das câmeras de referência empregadas no momento da síntese de vistas. Para mais informações sobre algumas técnicas utilizadas para o preenchimento desses tipos de artefatos podem ser observados os artigos Mao *et al* [Mao et al., 2013] e Machiavello *et al* [Macchiavello et al., 2014].

Para renderizar uma imagem virtual baseada em mapas de profundidade é empregada a representação de Múltiplas Vistas mais Profundidade (MVD). Esta representação apresenta em sua composição uma sequência de textura (cor) e outra sequência de profundidade.

O sistema MVD utiliza os conceitos de estereoscopia para conseguir criar a sensação de profundidade conforme abordado na seção seguinte.

2.3.1 Estereoscopia

A estereoscopia consiste em se obter duas imagens em posições distintas dando a sensação de profundidade dos objetos em uma cena. Esta técnica é utilizada naturalmente por seres humanos e animais e é aproximada por sistemas computacionais que tentam gerar esta capacidade por meio do uso de duas câmeras que estão separados por uma distância (*baseline*), simulando a separação dos olhos nos seres humanos. Por meio de duas imagens é possível encontrar correspondências e com isso calcular a profundidade dos pontos em um sistema 3D [Bradski and Kaehler, 2008].

Para a obtenção da informação de profundidade são necessárias algumas etapas, uma delas é retificar as imagens, fazendo com que passem por um processo de ajuste de ângulos e de distância entre as câmeras. Se o sistema tem variações no eixo vertical e horizontal, este é alinhado para variar em apenas um deles.

Outra fase importante é a de encontrar correspondências (*matchings*) entre pontos das imagens capturadas a partir de diferentes ângulos de uma mesma cena. A partir da diferença de posição de dois píxeis, que possuem o mesmo conteúdo, de par de coordenadas $p_1(u_1, v_1)$ e $p_2(u_2, v_2)$ localizado em duas imagens, pode-se calcular um mapa de disparidade. Este tipo de mapa informa o quanto o píxel de coordenada p_1 se deslocou entre sua posição no plano da imagem da câmera 1 em relação ao plano da imagem na câmera 2 resultando no ponto p_2 de mesmo conteúdo.

A partir do mapa de disparidade é possível criar um mapa de profundidade por meio da técnica de triangulação. Este mapa, diferentemente do mapa de disparidade, informa a profundidade de cada píxel na cena. Estes dois tipos de mapa possuem informações que são mutuamente deriváveis, isto é, desde que se tenha o mapa de profundidade é possível projetar (deslocar) o pixel para uma perspectiva diferente, criando uma imagem virtual em uma posição de câmera desejada e vice-versa. Na Figura 2.5 pode ser observado um exemplo de como este processo é realizado [Bradski and Kaehler, 2008]. Nesta figura existem dois planos coplanares com eixos ópticos paralelos e onde os valores focais das câmeras 1 e 2 são idênticos, logo $f_1 = f_2$ e, portanto, os dois valores passam a ser denotados apenas como f. Assumindo que ambas as imagens estão perfeitamente alinhadas e calibradas, os pontos principais c_1 e c_2 (o ponto principal é onde o raio principal inter-

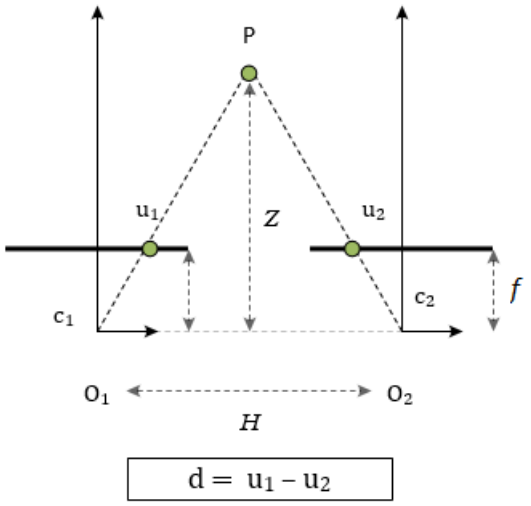


Figura 2.5: Exemplo de extração da informação de profundidade por meio da triangulação baseada em [Bradski and Kaehler, 2008].

secta o plano da imagem) representam os mesmos píxeis de coordenadas nas respectivas imagens. Dado um ponto P do mundo físico, p_1 e p_2 são as projeções do ponto P nas imagens da esquerda e direita respectivamente, os quais possuem coordenadas u_1 e u_2 no eixo das abscissas. Pode-se provar que a profundidade real Z do ponto P é inversamente proporcional à disparidade d entre as duas vistas, sendo $d = u_1 - u_2$. Assumindo-se que o ponto de interseção dos raios principais ocorre no infinito, a profundidade Z é definida por meio da semelhança de triângulos como dado pela equação seguinte [Bradski and Kaehler, 2008] :

$$\frac{H - (u_1 - u_2)}{Z - f} = \frac{H}{Z} \Rightarrow Z = \frac{fH}{u_1 - u_2}. \tag{2.10}$$

sendo H a distância entre os centros dos planos de imagem O_1 e O_2. Para o caso em que os pontos se encontram em algum ponto de distância finita, então a equação torna-se:

$$Z = \frac{fH}{d - (c_1 - c_2)}. \tag{2.11}$$

A relação entre os valores de profundidade e de disparidade são inversamente proporcional como pode ser observado na Figura 2.6. Dessa forma, quanto menor a disparidade, ou seja, mais próxima de zero, maiores são as mudanças no cálculo da profundidade. Quando o valor de disparidade é maior menores são as diferenças no cálculo da profundidade. Portanto, existe uma grande resolução de profundidade para objetos que estão mais próximos da câmera. [Bradski and Kaehler, 2008].

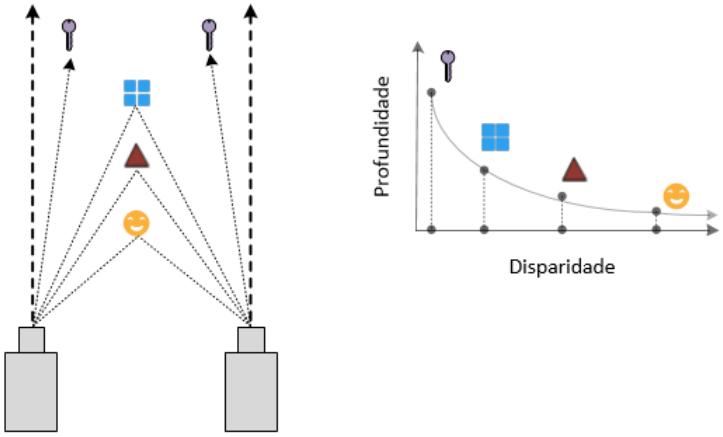


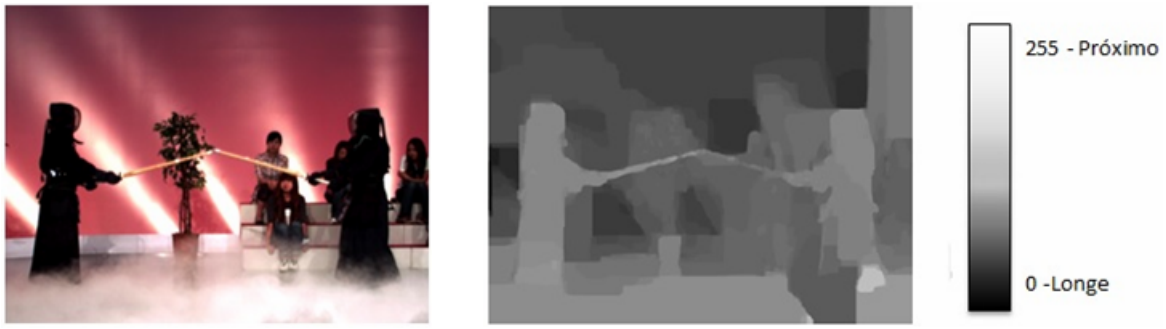
Figura 2.6: Relação inversa entre disparidade e profundidade baseada em [Bradski and Kaehler, 2008].

2.3.2 Mapas de Profundidade

A profundidade de uma cena pode ser obtida por meio de câmeras do tipo ZCam, as quais possuem um pulso de luz infra-vermelho que é utilizado na aferição da profundidade da cena no sistema de TV *broadcast* convencional. As informações de profundidade são armazenadas em arquivos chamados mapas de profundidade. Pode-se também transformar os vídeos que foram gravados originalmente em 2D para o formato 3D, inferindo a informação de profundidade da cena por meio de algoritmos específicos para este fim. O mapa de profundidade criado em ambas as situações pode ser representado como uma matriz em que os valores de seus píxeis variam entre tons de cinza, representados em valores normalizados no intervalo de 0 a 255. O valor real de profundidade pode ser recuperado a partir desta escala de valores como mostrada na seguinte equação:

$$Z = \frac{1}{\frac{\lambda}{255}(\frac{1}{Z_{near}} - \frac{1}{Z_{far}}) + \frac{1}{Z_{far}}} \tag{2.12}$$

na qual λ representa o valor do pixel quantizado, que varia entre 0 e 255. Portanto, o mapa de profundidade é representado em formato de 8 bits, conforme ilustrado na Figura 2.7b. Além disso, Z_{near} e Z_{far} são os limites do intervalo de profundidade existente no mapa, $255 \geq Z_{near} \geq Z_{far} \geq 0$. O termo Z_{near} é o valor do pixel mais próximo em relação à câmera enquanto que Z_{far} é o valor que representa a profundidade mais afastada em relação à câmera [Tian et al., 2009].



(a) Imagem de textura (b) Mapa de Profundidade

Figura 2.7: Exemplo de Mapa de profundidade baseado em [Tian et al., 2009]. Sequência Kendo [Tanimoto et al., 2008].

2.3.3 Síntese de vistas

Nesta subseção são abordados alguns conceitos básicos para a realização da síntese de vistas virtuais, mostrando algumas premissas necessárias para as técnicas de projeções usadas.

As câmeras de referências são os pontos a partir dos quais são gerados as vistas intermediárias. Para o processo de sintetização ser concluído são requeridas as informações das matrizes intrínseca e extrínseca, tanto das câmeras de referência quanto da câmera a ser sintetizada [Tian et al., 2009].

O primeiro passo para realizar a síntese de vistas é projetar os píxeis das imagens de referências para a vista virtual utilizando o mapa de profundidade. Esta projeção poderá basear-se em uma ou mais câmeras de referências que estejam disponíveis. O resultado que se deseja alcançar com a síntese é mostrado na Figura 2.8. Nesta figura pode-se perceber a distância horizontal de captura entre as câmeras.

Caso haja apenas uma câmera de referência para a síntese, há uma maior possibilidade de que problemas relativos a buracos de oclusão ocorram, pois existe menos informação disponível. Este tipo de artefato aparece quando partes da imagem que antes eram encobertas na câmera de referência passem a ser visíveis na câmera virtual, visto que a perspectiva da câmera mudou. Uma solução existente é o uso de duas ou mais imagens (câmeras) para a síntese ou a utilização de algoritmos de preenchimento de buracos de oclusão como o mostrado no artigo de Zitnick [Zitnick et al., 2004] .

Um outro problema inerente ao sistema de múltiplas vistas é a sobrecarga que este sistema gera na rede, em razão do envio de um largo número de sequências de quadros. Uma solução é adotar o envio de apenas algumas sequências com seus correspondentes mapas de profundidade e informações laterais (arquivos de configuração das câmeras) para a criação da vista virtual de posições que não foram enviadas [Tian et al., 2009]. Desta

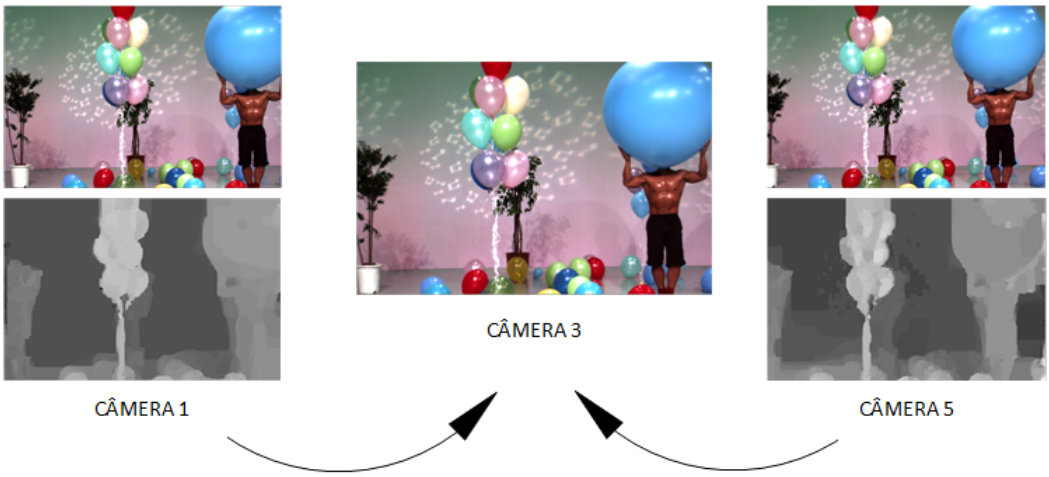


Figura 2.8: Ilustração do resultado final esperado após a síntese de vistas utilizando a sequência Balloons [Tanimoto et al., 2008]. As câmeras 1 e 5 são utilizadas como referência para a criação da câmera virtual 3.

forma, minimiza-se o consumo da rede de telecomunicação, enviando os parâmetros da câmera (matrizes intrínsecas e extrínsecas) e as sequências de textura com seus respectivos mapas de profundidades das câmeras selecionadas para referência.

2.3.4 Renderização Baseada em Imagem de Profundidade (DIBR)

O processo de renderização baseada em imagem de profundidade (DIBR) é o algoritmo utilizado para as projeções e, consequentemente, para geração das vistas virtuais. Neste processo é fundamental a utilização da sequência de textura e sua respectiva informação de profundidade. A projeção ocorre em dois passos: realizar a projeção dos píxeis de 2D para 3D e em seguida, criar a projeção 2D na posição da câmera virtual escolhida. Em computação gráfica, este método é chamado de distorção da imagem 3D [Fehn, 2004].

Além disso, procedimentos adicionais são aplicados com o intuito de melhorar a qualidade da vista virtual renderizada como o pré-processamento do mapa de profundidade e o preenchimento de buracos que são realizados posteriormente à criação da vista virtual.

Para a projeção dos píxeis das câmeras de referência para posição na câmera virtual é preciso mapeá-los para suas devidas posições, sendo imprescindível o uso do mapa de profundidade. Dado um par de coordenadas (u_1, v_1) localizado na imagem de referência, é possível calcular o par de coordenadas (u_2, v_2) na imagem virtual. Primeiramente, a partir das equações (2.4) e (2.5) pode-se obter as coordenadas 3D reais daquele ponto para uma câmera qualquer o que é obtido utilizando a seguinte equação [Tian et al., 2009]:

$$\begin{pmatrix} X_w \\ Y_w \\ Z_w \end{pmatrix} = \Psi_1^{-1} \cdot \left(z_1 \cdot A_1^{-1} \begin{pmatrix} u_1 \\ v_1 \\ 1 \end{pmatrix} - \tau_1 \right), \tag{2.13}$$

na qual z_1 é o valor de profundidade calculado para o par de coordenada de referência a partir da equação (2.12). A partir deste ponto, a próxima etapa é o cálculo da projeção do pixel de referência para o ponto correspondente na vista virtual, com a seguinte equação:

$$z_2 \cdot \begin{pmatrix} u_2 \\ v_2 \\ 1 \end{pmatrix} = A_2 \cdot \left(\Psi_2 \begin{pmatrix} X_w \\ Y_w \\ Z_w \end{pmatrix} + \tau_2 \right), \tag{2.14}$$

sendo (u_2, v_2) as coordenadas projetadas na posição da câmera virtual e $z_{2,cam}$ a respectiva profundidade que é calculada para a câmera virtual. Nesse processo, são necessárias as matrizes intrínseca $A_{3x3,2}$ e extrínseca $\Psi_{3x3,2}$ e $\tau_{3x1,2}$ da câmera virtual.

Um caso especial ocorre quando as câmeras estão retificadas. Neste cenário, as câmeras estão dispostas paralelamente em uma única dimensão e a disparidade ocorre em relação a um único eixo. O banco de dados utilizado neste trabalho apresenta variação somente no eixo u . Nesse caso específico, há variação apenas ao longo do eixo u e nenhuma variação no eixo v. Desta forma, pode-se simplificar a equação de projeção dos píxeis apresentada anteriormente. A partir das equações (2.13) e (2.14) e assumindo que as câmeras possuem a mesma distância focal, mesma matriz de rotação e valores de *offset* diferentes de zero, pode-se ter a equação de projeção e criação da vista virtual conforme a seguinte equação [Tian et al., 2009]:

$$u_2 = u_1 + \frac{f.(\tau_{x,2} - \tau_{x,1})}{z_1} + (o_{x,2} - o_{x,1}) \tag{2.15}$$

na qual f é a distância focal e $\tau_{x,2}$ e $\tau_{x,1}$ são as componentes no eixo x do vetor de translação das câmeras virtual e referência, respectivamente. As variáveis $o_{x,2}$ e $o_{x,1}$ são os pontos de *offset* das vistas virtual e referência.

Dado que denotamos $L = \tau_{x,2} - \tau_{x,1}$ (a distância entre as vistas que é chamada *baseline*) e que $d_{off} = o_{x,1} - o_{x,2}$ é a diferença de *offset*, a posição de renderização do pixel na vista virtual é definida por uma disparidade d, representada de acordo com a seguinte equação:

$$u_2 = u_1 + d_u = u_1 + (\frac{f.L}{z_1} + d_{off}). \tag{2.16}$$

Portanto, uma vez calculado o valor de profundidade z_1 da câmera de referência, pode-se projetar todos os píxeis da imagem referência para a imagem virtual usando a equação

descrita acima. Dessa maneira, realiza-se a síntese de uma vista virtual a partir de uma ou mais câmeras.

2.4 Codificação de Imagem e Vídeo

Um vídeo é composto por uma sequência de imagens. Nessas imagens há a representação de objetos do mundo real por meio de texturas, profundidades e iluminações diferentes. Este sinal de vídeo possui três componentes para sua representação: as componentes espaciais (X,Y) e a componente temporal que descreve seus quadros no tempo. Frequentemente, o modelo de cor escolhido para descrever o vídeo é o sistema Y:Cr:Cb.

Em meados dos anos de 1960, os compressores de vídeo começaram a surgir. Muitos dos compressores iniciais comprimiam os quadros que compunham os vídeos separadamente, sem levar em consideração a correlação entre eles. Entretanto, ao longo dos anos, as técnicas foram sendo aperfeiçoadas, atingindo maiores níveis de compressão [Sullivan and Wiegand, 2005].

A compressão de dados envolve um par de sistemas: um compressor e um descompressor. Este par é geralmente descrito como um CODEC e codifica uma imagem de entrada ou vídeo para uma forma comprimida e no destino, a informação é descomprimida para produzir uma cópia ou aproximação da imagem/vídeo de entrada. A compressão tenta representar aquela informação inserida na entrada com um mínimo de bits e com a maior fidelidade possível [Richardson, 2011].

Um codificador é composto por três tipos de recursos: modelo de predição temporal, modelo d predição espacial e codificador de entropia. Os vídeos e imagens podem ser codificados gerando a perda ou não de dados.

Neste contexto, a codificação de vídeo tem como objetivo diminuir o montante de dados a ser transferido, ou seja, diminuir a quantidade de dados a ser enviado preservando ao máximo qualidade da informação. Para o caso em que a compressão ocorre sem perdas de dados, toda a informação que é comprimida e codificada é recuperada exatamente no seu destino. Diferentemente, ao se inserir perdas, o conteúdo recuperado a partir da descompressão é aproximado daquele que foi enviado pela fonte.

A codificação e compressão de vídeos difere da compressão de imagens pela característica temporal que possui. Por meio deste aspecto, é possível aproveitar as redundâncias temporais que ocorrem entre quadros (*Inter-Frame*) e as redundâncias espaciais que existem dentro do próprio quadro (*Intra-Frame*). Nestas predições são utilizados blocos já renderizados para a predição e codificação apenas da diferença entre os valores dos blocos preditos e o de referência. Portanto, o codificador escolhe entre os diversos tipos de predições possíveis no momento da codificação de um vídeo.

Todos os resultados de predições Intra e Inter quadros, os vetores de movimento e os coeficientes das transformadas são comprimidos pelo codificador de entropia, que remove as informações estatísticas redundantes criando uma sequência binária (*bitstream*) que será transmitida ou armazenada [Richardson, 2011].

Uma visão geral de alguns processos que ocorrem na compressão de dados pode ser vista na Figura 2.9. A fase de quantização somente é utilizada para compressões com perdas. O fluxo criado pelo codificador de entropia é transmitido ao destino.

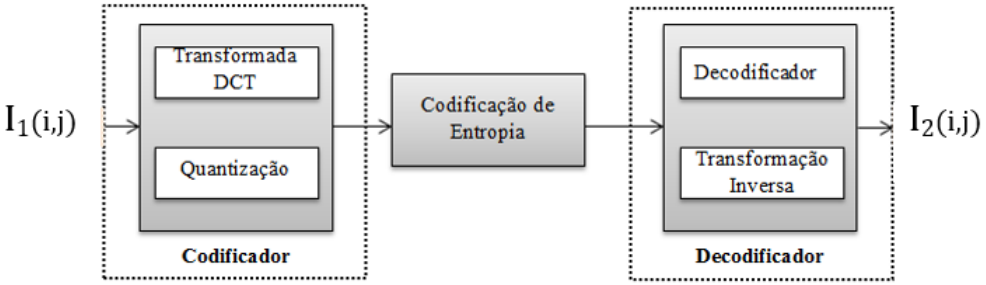


Figura 2.9: Processo de codificação e decodificação com perdas.

A maioria dos algoritmos de compressão de dados usam a vantagem que há na alta correlação entre as imagens de uma sequência para melhorar a compressão com a remoção da redundância. Além disso, uma abordagem holística é frequentemente utilizada com o objetivo de melhorar a compressão do vídeo. Um exemplo de um procedimento usado neste tipo de abordagem é o pré-processamento das imagens (quadros) que formam o vídeo com o intuito de reduzir a sua entropia e, portanto, aumentar a taxa de compressão. Na seção seguinte, introduziremos o conceito de entropia.

2.4.1 Entropia

A entropia está associada à quantidade média de informação que se tem sobre um experimento ou, caso esse experimento seja uma fonte que envia símbolos, está associada à medida do número médio de elementos binários necessários para codificar esses símbolos. Para o caso de uma fonte de informação que envia símbolos, o conjunto de símbolos é chamado alfabeto e cada símbolo deste alfabeto de letra [Sayood, 2012].

O conceito de entropia é muito amplo, entretanto pode-se entendê-lo como a medida de incerteza de uma variável aleatória. Se cada elemento dessa sequência é independente e igualmente distribuído (i.i.d.), a equação para entropia é:

$$H(\Omega) = -\sum_{i} p_i \log(p_i), \tag{2.17}$$

na qual p_i indica a probabilidade de um evento com uma variável aleatória ocorrer e Ω indica a sequência provinda da fonte.

No contexto desse trabalho a entropia está relacionada diretamente com a quantidade de bits necessários para se codificar uma fonte e refere-se à entropia de primeiro grau. Portanto, a entropia de uma imagem indica a quantidade de bits necessários para se codificar os seus símbolos, sendo que quanto menor a quantidade de símbolos nessa imagem menor o número de bits necessários para codificá-la.

2.4.2 Compressão de dados

As subseções seguintes abordam as técnicas convencionais usadas para a compressão de vídeos, descrevendo brevemente alguns conceitos utilizados. Tais técnicas são tipicamente utilizadas em codificadores de vídeo a fim de alcançar maiores níveis de compressão.

Um codificador que utiliza partições em blocos realiza uma divisão do quadro em blocos de tamanhos variados. No codificador padrão HEVC, estes blocos são mais flexíveis e possuem quatro diferentes conceitos: Unidade de Codificação em Árvore (CTU) (do inglês, *Coding Tree Unit*), Unidade de Codificação (CU) (do inglês, *Coding Unit*), Unidade de Predição (PU) (do inglês, *Prediction Unit*) e Unidade de Transformação (TU) (do inglês, *Transform Unit*). Uma mesma unidade de codificação (CU) compartilha o mesmo modo de predição espacial ou temporal [Kim et al., 2012].

Desta maneira, os modelos de predição tentam diminuir a redundância ao prever as informações atuais de um quadro baseado em alguma informação já codificada. Se a informação de predição do quadro atual é obtida a partir de quadros vizinhos é chamada de predição temporal. Caso a informação de predição venha do mesmo quadro, a partir de blocos vizinhos já codificados, ela é denominada predição espacial.

2.4.3 Estrutura de Partição em Blocos

Esta subseção apresenta a estrutura de partição em blocos para o padrão HEVC, uma vez que este é o padrão mais atual de codificação, sendo a evolução do codificador H.264/AVC. Neste padrão existem quatro principais estruturas que possuem papéis diferentes e bastante definidos: CTU, CU, PU e TU. Existem também os conceitos de Bloco de Codificação em Árvore (CTB) (do inglês, *Coding Tree Block*), Bloco de Codificação (CB) (do inglês, *Coding Block*), Bloco de Predição (PB) (do inglês, *Prediction Block*) e Bloco de Transformação (TB) (do inglês, *Transform Block*) que estão associados com as unidades de codificação já citadas [Kim et al., 2012].

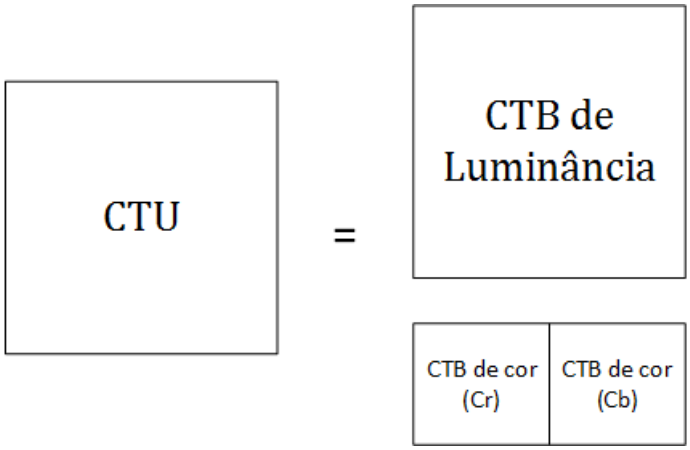


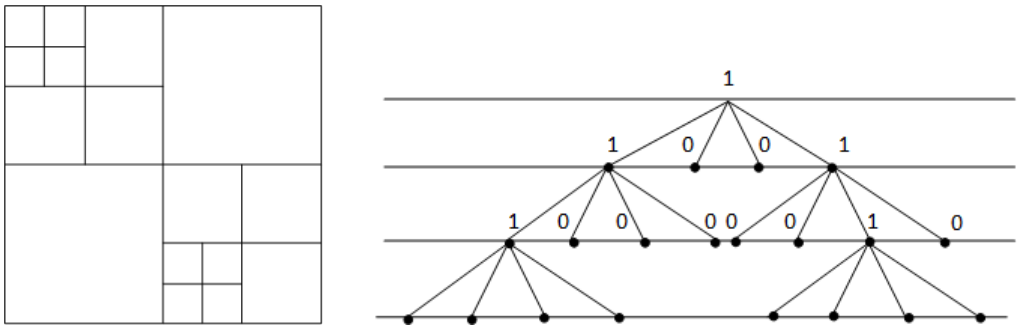
Figura 2.10: Composição de uma CTU.

Uma CTU é análoga ao conceito de macrobloco no H.264/AVC e é composta por três CTB's: uma CTB de luminância e duas CTB's de cores, conforme ilustrado pela Figura 2.10.

Cada CTB possui o mesmo tamanho da CTU correspondente e pode ser: 64×64, 32×32, 16×16 e 8×8. Cada CTB pode ainda ser dividido em parte menores chamadas CB's. Após a divisão no tamanho de CB ideal, o conjunto dos 3 CB's (Y,Cr,Cb) é chamado Unidade de Codificação (CU). Durante o texto, as CU's serão chamadas, de forma mais genérica, de blocos. Assim, quando mencionada a palavra bloco, estaremos nos referindo a menor unidade de partição para a realização da predição e da transformada.

Portanto, um quadro ao ser codificado possui múltiplas CTU's e estas CTU's podem ser divididas até o menor tamanho de CU para adaptação às informações de características locais. Uma *quadtree* é a estrutura utilizada para dividir uma CTU em múltiplas CU's, conforme pode ser observado na Figura 2.11b [Kim et al., 2012]. A Figura 2.11b representa a estrutura em árvore da partição de um quadro e das CTU's deste quadro da Figura 2.11a em unidades menores. Para cada CTU é decidido a necessidade de se aumentar a profundidade da partição. A decisão de aumentar a profundidade é marcado como 1 na Figura 2.11b, enquanto que a decisão de não particionamento é marcado como 0. Quando é alcançada a menor partição esta unidade é denominada CU. No HEVC, cada CU especifica o tipo de esquema de predição usado para as predições Intra e Inter quadros [Kim et al., 2012].

Quando o processo de particionamento é finalizado, são aplicados os particionamentos das CU's em PU's e TU's. Uma PU é uma unidade de predição e a região que é englobada por uma mesma PU compartilha o mesmo esquema de predição. Para uma mesma CU podem ser especificadas uma ou mais PU's. O HEVC define dois tipos de tamanho para o particionamento de PU para a predição Intra e oito tipos de tamanho para as partições de predição Inter.



(a) Particionamento de um quadro em CTU's.

(b) Estrutura de árvore das CTU's. Cada nó-folha representa uma CU.

Figura 2.11: Exemplo de particionamento de um quadro em CTU's e CU's. Figura adaptada de [Kim et al., 2012].

Já as Unidades de Transformações (TU) definem o tamanho da região onde ocorrerá uma transformada, logo uma TU conterá os coeficientes das transformadas. Uma CU pode conter uma ou mais TU's. As transformadas devem possuir o mesmo tamanho da TU. Assim, as escolhas dos tamanhos para a predições e para as transformadas são independentes.

Esta forma de particionar o quadro para codificação possui a vantagem de adaptar o tamanhos das unidades de codificação a depender das resoluções e do conteúdo da imagem. No HEVC, o tamanho máximo de partição é maior que em seu antecessor, o H.264/AVC. Com isso, consegue ter maior flexibilidade nas escolhas dos modos de predição ao dispor de mais maneiras de particionar as Unidades de Codificação (CU's).

2.4.3.1 Modelo de Predição Temporal

De forma geral, no modelo de predição temporal (Inter-Frame), o codificador tenta diminuir redundâncias ao buscar similaridades nas vizinhanças de um quadro. Estas similaridades podem ser buscadas por compensação de movimento baseada em blocos. Esta técnica busca um bloco já codificado que seja o mais similar possível ao que está para ser codificado. A saída desse processo é um bloco contendo apenas as diferenças entre os blocos estimado e de referência.

Detalhando um pouco mais esta predição, o modelo de predição temporal cria uma predição da imagem atual a partir dos blocos já codificados de quadros vizinhos. Subtrai o quadro a ser predito do quadro de referência e codifica apenas os valores de diferença entre os píxels, chamados resíduos. A maioria da energia residual ocorre devido ao movimento dos objetos da cena que pouco muda de um quadro para o quadro vizinho [Richardson, 2011].

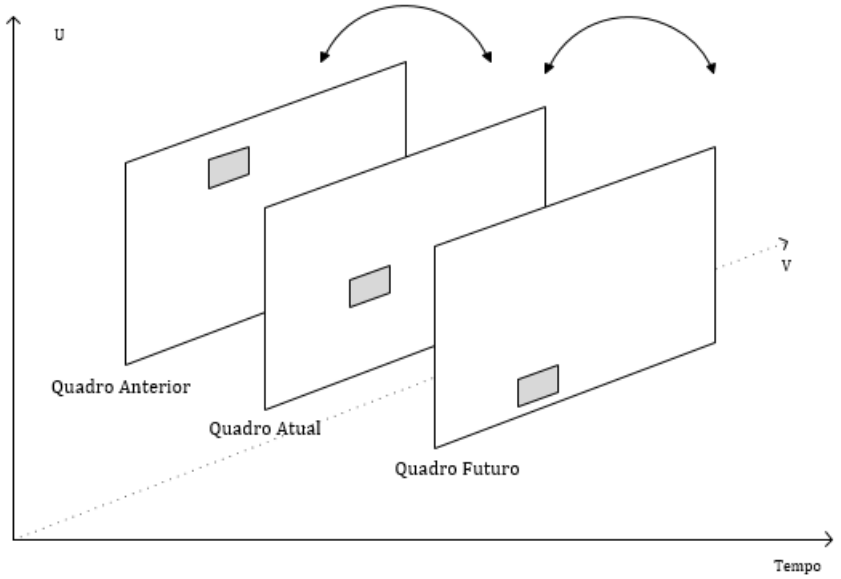


Figura 2.12: Estimação do Movimento de um bloco.

Na estimação do movimento, os vetores de movimento trabalham a nível de bloco e buscam no quadro de referência, que pode ser um quadro anterior ou o quadro seguinte ao atual, o bloco mais similar ao bloco atual conforme ilustrado na Figura 2.12. Quando é usado um quadro posterior ao atual, este necessita ser codificado primeiro. Desse modo, os quadros podem estar fora de ordem no arquivo codificado [Richardson, 2011].

Para diminuir o tempo de busca de um bloco de referência nos quadros vizinhos, é delimitada uma área de busca. Para se escolher o bloco de referência são utilizadas técnicas como a Soma Absoluta das Diferenças (SAD) para se encontrar a diferença mínima e definir o bloco mais semelhante ao bloco a ser predito.

Dado que se encontrou o bloco mais semelhante, a informação escrita no arquivo codificado é o seu vetor de deslocamento. Isto dará a trajetória daquele conjunto de píxeis descritos no bloco. O conjunto de pares de coordenadas que indica essas trajetórias é denominado fluxo óptico da imagem.

2.4.3.2 Modelo de Predição Espacial

Esta predição utiliza amostras de blocos previamente codificados no próprio quadro. Várias possibilidades de amostras de predição podem ser obtidas ao se explorar a vizinhança do bloco a ser codificado. Os píxeis mais próximos do bloco atual são os que, provavelmente, possuem mais alta correlação com ele. Na Figura 2.13 é ilustrado um bloco a ser codificado, logo este bloco procura na região já codificada o bloco mais semelhante

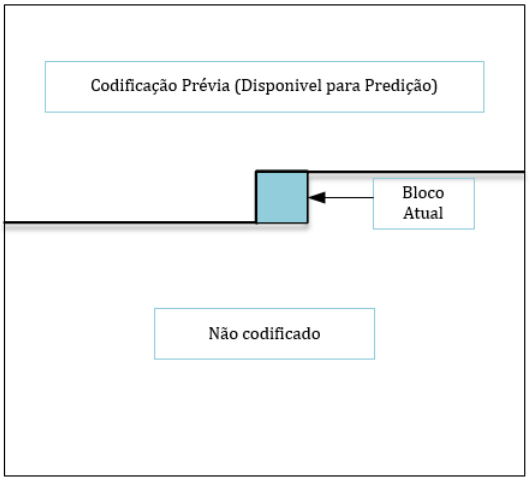


Figura 2.13: Predição Intra de um Bloco na Imagem. Figura Baseada em [Richardson, 2011]

a ele. A região acima e à esquerda já tem seus dados codificados, logo estes blocos são candidatos a serem utilizados como referência para a predição do bloco atual.

2.4.4 Transformada DCT

Após a fase anterior, em que há predição de alguns quadros em relação aos que servem de referência, é aplicada a Transformada Discreta de Cossenos (DCT, do inglês *Discrete Cosine Transform*). Esta transformada pode ser aplicada sobre quadros integrais (referência) da sequência ou mesmo sobre quadros que são preditos.

A transformada DCT possui este nome devido à matriz de transformação ser composta de vários cossenos [Sayood, 2012]. Esta técnica é amplamente empregada no contexto de compressão de dados e nos codificadores padrões, alcançando os melhores níveis de compressão quando comparada a outras técnicas de mesmo contexto [Gonzalez and Woods, 2010].

Dada a característica de ortogonalidade da transformada, este é um processo reversível. A DCT gera coeficientes para a representação da informações transformadas e possui a vantagem de representar, em seus coeficientes, somente a parte real do sinal, diferentemente de outras transformadas que possuem ambas as partes, imaginária e real. Outra vantagem pela qual ela é aplicada em muitas técnicas de compressão de áudio e vídeo é o fato de concentrar muita energia em poucos coeficientes, podendo os coeficientes com menos energia serem descartados sem introduzir uma distorção significante no sinal original [Diniz et al., 2010].

Esta transformada é usada em conjunto com a técnica Modulação de Código de Pulso Diferencial (DPCM) (do inglês, *Differential Pulse-Code Modulation*). Desta forma, a

codificação é aplicada na diferença entre amostras. A vantagem de se utilizar DPCM está no fato de se reduzir a variância das intensidades dos píxeis das sequências, já que diferenças entre amostras semelhantes tendem a ser pequenas e se mantêm em um intervalo menor de valores. Mais informações a respeito destas técnicas podem ser encontradas em Gonzalez [Gonzalez and Woods, 2010] e Richardson [Richardson, 2011].

2.4.5 Principais Padrões de Codificação

2.4.5.1 H.264/AVC

O padrão de Codificação de Vídeo Avançada (H.264/AVC *Advanced Video Coding*) é um sucessor de uma família em evolução de codificadores de vídeo comerciais e demonstrou uma aumento significativo na capacidade de compressão de vídeos. Este padrão foi desenvolvido pelo *Joint Video Team* do ITU-T *Video Coding Experts Group* (VCEG) e também pelo *Moving Picture Experts Group* (MPEG) do ISO/IEC [Wiegand et al., 2003].

Esta subseção tratará de alguns aspectos adicionais e que diferenciam este codificador do resto de sua família. Algumas características continuaram sendo propagadas de versão em versão [Sullivan and Wiegand, 2005]:

- A precisão de pixel para compensação de movimento. Adicionou-se a precisão de um quarto de amostra para a componente de luminância e de um oitavo de amostra para as componentes de cor na definição dos vetores de movimento. Nas versões anteriores, utilizava-se somente a precisão inteira ou de metade de amostra(versões MPEG-1, MPEG-2 e H.263).
- A característica de predição bilateral, ou seja, a predição do quadro atual, advém tanto de quadros no passado quanto de quadros no futuro. Isto tem estado presente em todas a s versões desde o MPEG-1 e ajuda quando a cena possui objetos ocultos ou quando há muito movimento desses objetos entre quadros.

Além destas características o padrão foi adaptado para ter a flexibilidade para utilização em várias aplicações. Portanto, foi dividido em duas camadas: a camada de conteúdo de vídeo (VCL, *Video Coding Layer*) e a camada de abstração de rede (NAL, *Network Abstraction Layer*).

A camada VCL possui resumidamente: predição espacial e temporal dos blocos e subdivisão dos tamanhos dos blocos; classificação dos quadros em *slices* dos tipos I (intra), *slices* do tipo P(intra e um quadro de referência para a compensação de movimento) e *slices* do tipo B (intra com base em duas referências para a compensação de movimento); predições Inter em *slices* do tipo P e B; transformada, escalamento e quantização; codificação de entropia; codificação adaptativa dos quadros [Sullivan and Wiegand, 2005].

2.4.5.2 Codificação H.265/HEVC

O projeto de Codificação de Vídeo de Alta Eficiência (HEVC, do inglês *High Efficiency Video Coding*) é o mais novo das organizações: Grupo Especialista de Codificação de Vídeo (VCEG) e do MPEG ISO/IEC. Trabalharam colaborativamente como a parceria *Joint Collaborative Team on Video Coding* (JCT-VC) [Uhrina et al., 2014].

Muito do que já havia no padrão anterior persistiu, havendo algumas mudanças e alguns acréscimos, como citado por [Uhrina et al., 2014]:

- Mais flexibilidade no tamanho dos particionamentos e nos tamanhos dos blocos e modos de predição e transformação.
- Consegue ocupar menos espaço de armazenamento e transmissão para uma mesma qualidade e tamanho de vídeo em relação aos seus antecessores. Em uma mesma largura de banda conseguem-se resoluções maiores.

Este é um padrão desenvolvido para se adequar a crescente demanda por resoluções e qualidade dos vídeos, com isso, engloba uma maior eficiência da compressão, facilidade no sistema de integração transporte e elasticidade na perda de dados [Uhrina et al., 2014].

Uma revolução desses padrões foram as versões desenvolvidas para atender a demanda dos sistema de múltiplas vistas, conforme abordado na seção seguinte.

2.4.6 Codificação de Vídeo Multivista mais Profundidade

Trabalhos de expansão dos codificadores para atender o desenvolvimento dos sistemas Vídeos 3D (3DV) têm sido feitos desde 2001 quando o grupo MPEG começou a padronização dos vídeos e áudios para 3D. O grupo MPEG juntamente com o JVT vem desenvolvendo desde 2004 um padrão para os sistemas multivistas denominado Codificação de Video Multivista (MVC), sendo distribuído em 2009 [Lee and Ho, 2009].

Vários codificadores comerciais atuais, como o H.264/AVC e sua posterior versão HEVC/H.265, adaptaram os seus produtos para incluir a codificação de múltiplas vistas e também para múltiplas vistas com profundidade, criando extensões como o 3D-AVC para o H.264/AVC e o Multiview HEVC (MV-HEVC) e 3D-HEVC para o codificador HEVC. O 3D-HEVC foi o último a ser lançado em fevereiro de 2015 [Tech et al., 2016b]. Nesses tipos de codificadores são aproveitadas as redundâncias, tanto espaciais quanto temporais, como também as redundâncias entre as sequências de quadros de câmeras distintas.

Os mapas de profundidade são bastante úteis nestas versões, pois têm a característica de serem monocromáticos (contendo apenas a luminância) e são utilizados para predições juntamente com a sequência de textura dos vídeos [Merkle et al., 2007].

A extensão MV-HEVC, que faz parte da segunda versão do HEVC, permite que codificações mais eficientes sejam feitas em relação ao simulcast [1] ao explorar as redundâncias entre as diversas câmeras de um sistema de múltiplas vistas. Esta extensão reutiliza os decodificadores *single-layer* sem precisar mudar os módulos de processamento a nível de bloco. A codificação neste nível se mantém inalterada e apenas a sintaxe de alto nível é adaptada. Já o 3D-HEVC codifica as múltiplas vistas com os seus respectivos mapas de profundidade. Este método consegue aumentar sua eficiência de compressão quando comparado ao MV-HEVC, pois explora, explicitamente, as dependências estatísticas entre o vídeo de textura e os mapas de profundidade, introduzindo ferramentas de compressão como [Tech et al., 2016a]:

- Tratamento das características únicas dos mapas de profundidades.
- Exploração, tanto das redundâncias entre as câmeras, por exemplo o MV-HEVC, como aproveitamento das relações das texturas com seus respectivos mapas de profundidade.

Neste tipo de sistema a abordagem que se faz é multicamada, diferentemente do HEVC que realiza a compressão para sequências de imagens únicas. Cada camada pode ser um quadro de textura, profundidade ou informações laterais para uma determinada câmera e todas as camadas que pertencem a uma câmera são denotadas como uma mesma vista. Neste caso, as camadas são multiplexadas em apenas um *bitstream* e podem depender uma das outras. Essas dependências podem ser predições entre camadas, explorando similaridades. Todas as informações de uma câmera determinam uma vista [Tech et al., 2016a].

A Figura 2.14 mostra a estrutura de codificação para um sistema de múltiplas de vistas. No sistema MV-HEVC, as predições temporais são denotadas (T) e usam quadros da mesma componente[2] e mesma vista, porém de diferentes unidades de acesso (AU's, do inglês Access Unit), que são todas as imagens associadas a um mesmo tempo de captura e, portanto, estão contidas em uma mesma unidade de acesso; predições realizadas em uma mesma componente e na mesma AU, porém em vistas diferentes são denotadas por (S).

Nesta figura é mostrado quatro camadas, sendo duas vistas e duas componentes(textura e profundidade). O primeiro quadro da primeira camada é chamado camada base e as camadas seguintes são denominadas camadas de aprimoramento. A ordem deve ser a mesma dentro das AUs para todas as vistas e para todos os componentes, sendo que um quadro de profundidade sempre deve seguir o seu respectivo quadro de textura.

[1] Simulcast é a transmissão simultânea em diferentes meios ou locais.

[2] É o conjunto que carrega as camadas de mesmo tipo de informações como quadros somente de textura ou de profundidade no contexto de compressão de vídeos 3D.

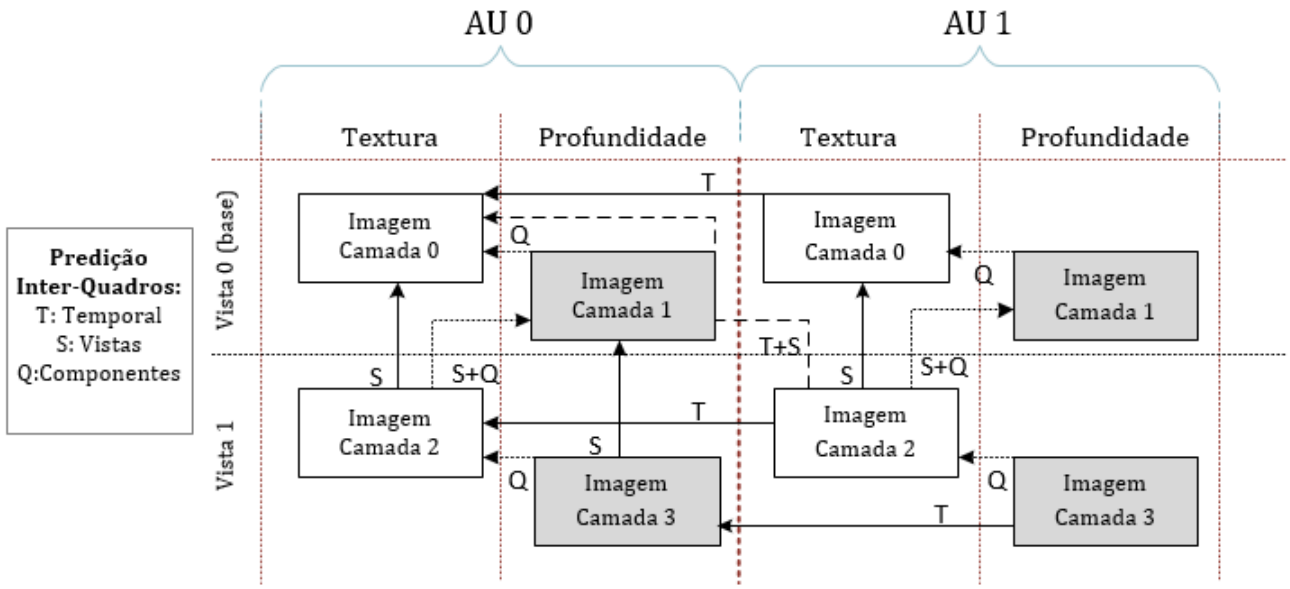


Figura 2.14: Predições Inter-Quadros para o caso MV-HEVC, 3D-HEVC(somente textura), 3D-HEVC. Baseada em [Tech et al., 2016a]

Ainda no contexto do sistema MV-HEVC, os Vetores de Movimentos (MV) podem ser Vetores de Movimentos Temporais (TMV) relacionados com uma mesma vista ou entre vistas diferentes só que pertencentes à uma mesma AU, chamados de Vetores de Movimentos de Disparidade (DMV).

Para o 3D-HEVC a predição Inter-Camada extende a MV-HEVC para aumentar o nível de compressão ao permitir diferentes tipos de predições. É possível realizar predições temporais combinadas com a predição entre vistas denominada (T+S), referenciando a imagem de uma mesma componente mas em diferentes vistas e AUs; predições entre componentes (Q), sendo imagens de uma mesma AU e mesma vista porém em componentes diferentes; e por último, pode combinar a predição entre componentes e vistas (Q+S) sendo feita com imagens de uma mesma AU porém em diferentes componentes e vistas [Tech et al., 2016a].

Uma limitação na predição entre vistas tanto para o MV-HEVC quanto para o 3D-HEVC, é que as ferramentas utilizadas são mais eficientes quando os sinais de imagens estão retificados em um arranjo coplanar 1D, pois as predições são alcançadas por meio de compensações de disparidade baseada em bloco, mais informações sobre a codificação de sistemas múltiplas vista pode ser encontrado em [Tech et al., 2016a].

2.4.7 Métricas para Avaliação de Qualidade da Imagem

Um algoritmo de compressão pode ser avaliado por meio de diversos fatores, isto é, podem ser realizadas medidas a respeito de sua complexidade, a quantidade de compressão que este alcança e também o quanto a informação reconstruída é semelhante à original [Sayood, 2012]. Para se avaliar a qualidade da multimídia reconstruída e obter uma mensuração do quanto ela se modificou por conta da compressão são utilizadas as métricas de avaliação da qualidade da imagem.

Muitas vezes a informação comprimida e enviada por um meio de comunicação pode sofrer degradações. Estas podem ser causadas pelos ruídos presentes no meio. Da mesma forma, são aplicadas as técnicas de mensuração da qualidade para a avaliação do conteúdo recebido. Existem tantos fatores a se avaliar que somente este tópico é um extenso assunto de pesquisa.

Estas técnicas de avaliação da qualidade fazem uma comparação entre a informação original e a informação recebida ou reconstruída a partir da compressão e podem ser de dois tipos: subjetivas ou objetivas.

Uma avaliação subjetiva depende do conteúdo e dos objetivos do usuário em relação à multimídia. Um exemplo dessa diferença em relação ao objetivo do usuário ocorre quando este assiste a um filme comum em que o mesmo terá uma atitude passiva em relação ao conteúdo, diversamente de sua atitude ao observar um vídeo de vigilância no qual seja necessário detectar uma pessoa, sendo a atitude do usuário ativa e a qualidade da multimídia incomodará mais conforme seja o seu insucesso em reconhecer o agente alvo do vídeo [Richardson, 2011]. Tais aplicações são bastante distintas e portanto poderão causar mais desconforto e dificuldade em se atingir seus objetivos.

Nesse tipo de avaliação, o usuário fornece a sua opinião sobre a qualidade do que ele assiste. As informações obtidas são completamente pessoais e, portanto, é necessário realizar esses testes com um grupo amostral suficiente para se tirar alguma conclusão do sistema. Neste processo é fundamental levar em consideração também alguns fatores que podem influenciar a opinião das pessoas como o estado do usuário e de sua mente no momento da avaliação que pode afetar a sua decisão, condições como o ambiente de observação e o quanto o usuário interage com a cena observada. Todos este fatores podem modificar a opinião fornecida naquele momento [Richardson, 2011].

Dada a complexidade e o alto custo de se realizar experimentos de avaliação de qualidade subjetiva, outra forma de estimativa da qualidade é a métrica objetiva em que se elaboram equações matemáticas que aproximam-se do resultado de uma avaliação subjetiva. Entretanto, o sistema humano é muito complexo e se torna muito difícil reproduzir exatamente uma avaliação humana [Richardson, 2011]. Várias medidas matemáticas podem ser feitas para se estimar a percepção de qualidade. A escolha de uma métrica

depende do que se deseja observar, ou seja, quais artefatos devem ser melhor detectados para indicar uma distorção no conteúdo avaliado.

Uma abordagem natural ao se tentar identificar distorções nas imagens é a estimação das diferenças entre o sinal de teste e o original. Por mais de 50 anos uma das métricas dominantes na avaliação objetiva da qualidade de um sinal foi o Erro Quadrático Médio (MSE) (do inglês, *Mean-Squared Error*). Portanto, o objetivo desta métrica é comparar dois sinais, produzindo uma medida quantitativa da similaridade ou distorção existente entre eles [Wang and Bovik, 2009].

Supondo duas imagens, uma original denotada como I_1 e I_2 que é a imagem de saída do sistema (teste), a diferença entre estas imagens de tamanho $M \times N$ é obtida utilizando a seguinte equação:

$$MSE(I_1, I_2) = \frac{1}{M \times N} \sum_{i=1}^{M} \sum_{j=1}^{N} (I_{1\,(i,j)} - I_{2\,(i,j)})^2. \tag{2.18}$$

Apesar de esta técnica receber críticas quando se trata da avaliação de sinais de voz e de imagem, ela é simples, de complexidade baixa e possui um significado bem claro: define a energia do erro da imagem [Wang and Bovik, 2009].

Outra opção e que é bastante utilizada na literatura para se medir a qualidade de uma imagem é a Relação Sinal-Ruído de Pico (PSNR) (do inglês, *Peak-Signal-to-Noise Ratio*). Esta é uma medida da quantidade de erro em relação ao valor de pico do sinal, ou seja, o valor mais alto na imagem. A medida do PSNR é dada para uma imagem em tons de cinza com 8 bits/pixel conforme a seguinte equação:

$$PSNR(dB) = 10 \log_{10} \frac{255}{MSE}. \tag{2.19}$$

o PSNR também usa uma escala logarítmica e depende do MSE da imagem original e a imagem de teste do sistema. O PSNR aproxima-se do infinito assim como o MSE aproxima-se de zero para indicar uma alta qualidade de imagem. Por outro lado, se o PSNR apresenta valores pequenos, isto indica que há uma alta diferença na imagem [Hore and Ziou, 2010].

Apesar da medida do PSNR apresentar péssima discriminação no conteúdo estrutural de imagens com diferentes tipos de degradações, alguns estudos mostram que que o MSE e, consequentemente, o PSNR possuem melhor performance em medir a qualidade de imagens ruidosas [Hore and Ziou, 2010] .

O PSNR consegue medir a distorção presente em uma imagem, dizendo o quanto a imagem está diferente em relação à original. Uma desvantagem do uso do PSNR é a sua ineficiência em relação a medição de erros perceptíveis ao ser humano. Portanto, se

o objetivo for saber o quanto duas imagens diferem (em relação a suas energias), esta métrica consegue ser simples e objetiva para se ter um resultado.

Capítulo 3

Revisão da Literatura

Este capítulo tem como objetivo mostrar trabalhos correlatos que serviram como base para o desenvolvimento deste. Portanto, os artigos seguintes embasam as premissas assumidas e a abordagem utilizada neste trabalho de dissertação, expondo as técnicas que tem como objetivo aumentar a qualidade da vista sintetizada ou que buscam a melhor relação de taxa-distorção, promovendo maiores níveis de compressões sobre os mapas de profundidade.

3.1 Trabalhos Relevantes

O artigo de Lee e Ho [Lee and Ho, 2009] tem como finalidade promover uma síntese de vista com o mínimo de imperfeições e com a capacidade de suportar navegação livre e vídeos 3D. Neste artigo são descritos também os processos para síntese de vista. Considerando a posição escolhida pelo usuário para visualização da cena, buracos de oclusão podem surgir, sendo proposta uma solução para este problema. Uma solução para a compensação do erro da síntese também é discutido neste trabalho, em que se propõe uma remoção do ruído das bordas, o qual ocorre em razão da falta de acurácia dos métodos de estimação de profundidade. Este método de remoção de ruído aumenta a qualidade objetiva da imagem sintetizada. A contribuição deste trabalho para a dissertação é o de uma abordagem geral sobre o sistema de múltiplas vistas, sendo discutidos temas desde a captura até a codificação dos vídeos em 3D, além disso, trata das equações de projeção e de síntese de imagens.

O artigo de Tian *et al* [Tian et al., 2009] foi importante para melhor explorar as técnicas de projeção em sistemas de múltiplas vistas. Neste artigo é apresentada a técnica DIBR e suas respectivas equações para a realização da síntese de vistas, utilizando os mapas de profundidade para este fim. Além das técnicas para as projeções, o artigo aborda também os algoritmos de preenchimento de buracos. As equações expostas neste artigo

serviram como base para a implementação dos algoritmos para a síntese de vistas. As vistas sintetizadas são utilizadas para a comparação entre os resultados posteriores aos pré-processamentos.

O trabalho de Vetro [Vetro et al., 2011] faz uma introdução sobre a técnica de codificação de vídeo utilizada no padrão H.264/MPEG-4, mostrando suas características principais. O artigo aborda o padrão para a codificação específica para os sistemas de múltiplas vistas, no qual prevê predição entre vistas (câmeras diferentes) para obter maiores níveis de compressão, além das predições utilizadas nos padrões comerciais para fluxo único de vídeo, como as predições temporal e espacial. O artigo também examina a extensão do padrão H.264/MPEG-4 para o formato de vídeo em múltiplas vistas (MVC), que possibilitou uma predição mais específica, levando em consideração as redundâncias entre vistas e gerando ganhos significativos (quando comparados à codificação *simulcast* independente).

3.2 Distorção Profundidade Admissível

O extenso volume de dados necessários para o envio do vídeo de textura e dos respectivos mapas de profundidade é um problema inerente aos sistemas de múltiplas vistas e das aplicações que envolvem vídeos 3D e FVV. Por conseguinte, uma solução proposta é a diminuição da quantidade de câmeras, em que são enviados apenas alguns dos fluxos de vídeos aos usuários. Estes fluxos são utilizados como entrada para a técnica DIBR e a imagem virtual é criada no lado do cliente [Fehn, 2004]. Em um sistema FVV, o usuário escolhe o ângulo de visualização da cena (dentro de uma quantidade limitada) e, caso este ângulo não exista, uma imagem intermediária é renderizada entre duas câmeras fisicamente existentes.

Uma outra solução que tem sido cada vez mais explorada por vários pesquisadores como Zhao *et al* [Zhao et al., 2011], Kang e Ho [Kang and Ho, 2012], Zhang *et al* [Zhang et al., 2013], Zhang *et al* [Zhang et al., 2014], Zhang *et al* [Zhang et al., 2015] e Zhang *et al* [Zhang et al., 2016] consiste em otimizar o nível de compressão alcançado durante o processo de codificação das informações dos mapas profundidade em sistemas múltiplas vistas. Estas otimizações podem levar a erros de projeção na criação da imagem virtual e, portanto, é preciso manter uma qualidade aceitável da vista sintetizada.

Neste sentido, esta subseção analisa o conceito de Distorção de Profundidade Admissível (ADD) (do inglês, *Allowable Depth Distortion*) que é um conceito utilizado para a otimização do nível de compressão dos mapas de profundidade. Esta é uma abordagem que explora o método de projeção de imagens DIBR, manipulando os intervalos de projeções dos píxeis, ou seja, seus valores disparidade, para atingir níveis de compressão superiores

às abordagens convencionais em que não há qualquer tipo de pré-processamento sobre os mapas de profundidade. O emprego do conceito ADD proporciona a alteração dos valores do mapa profundidade, respeitando os intervalos de projeção [1] pré-determinados.

Este intervalos são definidos conforme os valores de disparidade calculados para cada valor de píxel em um mapa de profundidade. Portanto, agrupam-se em um mesmo intervalo todas os valores de profundidade que geram um mesmo cálculo de disparidade. Com isso, dependendo da forma de manipulação dos valores do intervalo, pode-se reduzir a variância e atingir uma codificação que seja mais eficiente.

Este conceito foi mencionado pela primeira vez no trabalho desenvolvido por Zhao *et al* [Zhao et al., 2011], no qual eles descrevem um modelo para definir os intervalos ADD. Eles salientam que a síntese de imagens é muito sensível às distorções, principalmente nas bordas dos objetos, uma vez que nestas regiões ocorre uma mudança abrupta nos valores de profundidades. Logo, erros podem ser introduzidos na compressão de dados. Os autores aplicam o modelo descrito (ADD) e comprimem os mapas de profundidade no modo sem perdas (*lossless*) do codificador, ou seja, o processo não introduz distorções, explorando apenas os intervalos de intensidade de forma a obter uma maior eficiência de compressão. Este trabalho propõe uma suavização (gaussiana) que faz com que os valores de profundidade alterados pelo filtro gaussiano não ultrapassem o intervalo ADD. Desta forma, os valores de profundidade dos mapas de referência, que são utilizados para a projeção, são modificados sem que distorções sejam introduzidas na imagem virtual. Ademais, os autores medem o tamanho da janela de suavização gaussiana, adequando-a com o propósito de respeitar os limites dos intervalos ADD. A filtragem proposta não causa nenhuma nova oclusão ou erro de sintetização, conseguindo aumentar a eficiência da codificação *Intra-Frame* do mapa de profundidade.

Em Zhang *et al* [Zhang et al., 2014], um modelo baseado na propriedade ADD e na utilização da taxa-distorção para ajudar na decisão dos modos de codificação é proposto. Desta forma, é possível escolher entre os tamanhos de blocos para codificação: 4x4, 8x8, 16x16, 8x16, e assim por diante. Igualmente, é proposto um algoritmo para diminuir a taxa de bits dos mapas de profundidade ao mesmo tempo que visa a manter a qualidade da vista sintetizada.

Conforme os autores, as otimizações feitas na codificação dos mapas de profundidade levam em consideração que estes são compostos apenas pela parte da luminância de uma imagem em cores e que os mapas de profundidade são planos e lisos, possuindo, provavelmente, ruído e inconsistências temporais devido à estimação de profundidade.

Ainda de acordo com o artigo, dado que o mapa de profundidade é apenas utilizado como informação geométrica e para a criação da vista virtual, não sendo visto diretamente

[1] Estes intervalos são inerentes a uma determinada sequência de imagem.

pelo usuário, distorções causadas pela sua codificação podem acarretar em artefatos na imagem sintetizada, podendo corromper as bordas dos objetos ou, ainda, criar falsos contornos. Isto posto, é desejável que não haja a degradação da imagem sintetizada devido à diminuição da taxa de bits no momento da codificação dos mapas de profundidade. Os resultados deste artigo mostram por meio da codificação intra-frame dos mapas de profundidade que há um pertinente ganho ao se utilizar o algoritmo proposto.

Uma outra proposta é mostrada em [Zhang et al., 2015], em que também é utilizada a abordagem ADD para realizar um processamento semelhante a uma filtragem sobre o mapa de profundidade para a geração de arquivos mais comprimidos. Os autores mostram uma análise da distorção da profundidade admissível baseada na otimização da função do custo de Taxa-Distorção (RD) (do inglês, *Rate-Distortion*). Os autores desenvolvem um algoritmo para a escolha do quadro de referência para o vídeo. Com isso, conseguem melhorar a eficiência da codificação devido à decisão otimizada de taxa-distorção (RD) que influencia na escolha do quadro referência e também na escolha dos intervalos ADD.

Ainda neste artigo, os autores apresentam um método eficiente para codificação dos valores de profundidade relacionados aos mapas de profundidade, baseado nos intervalos ADD. Portanto, dada uma relação dos valores de profundidade que geram mesmos valores de disparidade, o mapa de profundidade será modificado para conter valores únicos que representem cada intervalo (grupo) ADD.

De acordo com [Zhang et al., 2015], os valores de profundidade são apenas utilizados para o cálculo da projeção dos píxeis na imagem virtual. Em virtude do não casamento entre os valores de profundidade e disparidade, isto é, os valores de profundidade podem variar entre 0 e 255 enquanto que os valores de disparidade variam em média de 0 a 30. Isto levará a um mapeamento "muitos-para-um", ou seja, existem vários valores de profundidade que resultam em um mesmo cálculo de disparidade, o que é chamado no artigo de [Zhang et al., 2014] de intervalo ADD. O algoritmo proposto neste artigo integra a decisão rápida do modo, a seleção do quadro e a otimização da relação taxa-distorção (RD) a fim de conseguir diminuir a complexidade de codificação da profundidade, mantendo a mesma qualidade da imagem sintetizada pela síntese de vistas.

O artigo de Zhang *et al* [Zhang et al., 2014] descreve o conceito ADD matematicamente ao derivar as equações que definem esta propriedade. Este trabalho começa a descrever a técnica ADD ao mostrar uma forma de se renderizar os pixeis das câmeras de referência para a imagem virtual. Portanto, um ponto renderizado na imagem virtual a partir de uma imagem da referência conforme o artigo pode ser dado por

$$p_2 = z_1 A_2 \psi_2 \psi_1^{-1} A_1^{-1} p_1 - A_2 \psi_2 \psi_1^{-1} \tau_1 + A_2 \tau_2, \tag{3.1}$$

em que são utilizadas as informações intrínsecas (A_1 e A_2), extrínsecas de rotação (ψ_1

e ψ_2) e extrínsecas de translação (τ_1 e τ_2) da câmera, conforme mostrado no capítulo 2; $p_2 = [u_2, v_2, z_2]^T$ e $p_1 = [u_1, v_1, 1]^T$ são, respectivamente, os píxeis nas imagens virtual e de referência, sendo z_1 o valor de profundidade para a imagem de referência.

A disparidade entre os pontos p_1 e p_2 em cada um dos eixos de coordenadas pode ser dado por

$$d_u = \Phi(u_1 - \frac{u_2}{z_2}) \tag{3.2}$$

e

$$d_v = \Phi(v_1 - \frac{v_2}{z_2}) \tag{3.3}$$

em que o arrendondamento Φ é dado pela equação

$$\Phi(x) = \frac{\lfloor x \cdot 2^\eta + 2^{\eta-1} \rfloor}{2^\eta}. \tag{3.4}$$

Esta função depende do nível de acurácia desejada na síntese de vistas, em que η é a precisão de pixel para a renderização, podendo assumir valores iguais a 0, 1 ou 2. Esta variável indica se a precisão é inteira, metade de pixel ou quarto de pixel.

O cálculo de disparidade das Equações 3.2 e 3.3 são aplicados para o caso geral. Porém, existe o contexto específico para quando as câmeras estão retificadas, ou seja, a variação entre as câmeras existe somente em um dos eixos de coordenadas. Neste caso, como citado no Capítulo 2, as câmeras são paralelas e seus parâmetros intrínsecos são iguais. Logo, $A_1 = A_2$, $\psi_1 = \psi_2$ e $\tau_1 - \tau_2 = [L, 0, 0]^T$, sendo L a distância de *baseline* conforme explicado na seção 2.3.1. Neste cenário retificado existe, portanto, a variação e a disparidade somente em d_u, sendo d_v igual a 0 [Zhang et al., 2014]. O artigo considera que o valor de *offset* é igual a 0. Logo, aplica-se o arredondamento sobre a disparidade d_u da Equação (2.16) usando a seguinte equação:

$$d_u = \Phi(\frac{fL}{Z}), \tag{3.5}$$

sendo f o valor da distância focal no eixo da abcissa compartilhada pela câmera virtual e a câmera de referência e Z é a profundidade real para o pixel da câmera de referência, dada pela equação (2.12).

A partir das equações (2.12) e (3.5), substituímos o valor de profundidade Z e a equação de disparidade é definida como:

$$d_u = \Phi(Lf(K_1\lambda + K_2)), \tag{3.6}$$

sendo λ o valor de profundidade do pixel quantizado e representado no mapa de profundidade e $K_1 = \frac{1}{255}(\frac{1}{Z_{near}} - \frac{1}{Z_{far}})$ e $K_2 = \frac{1}{Z_{far}}$.

De acordo com o artigo [Zhang et al., 2014] durante o processo de quantização que ocorre na compressão do fluxo de profundidade, distorções nos valores codificados podem ocorrer, introduzindo um valor $\Delta\lambda$ ao valor original de profundidade λ, produzindo um valor igual a $\lambda+\Delta\lambda$ na descompressão do valor transmitido. É possível calcular a diferença entre os valores de disparidade produzidos pelo valor de profundidade original e o valor de distorcido conforme observado na equação:

$$\Delta d_u = g(\lambda, \Delta\lambda) = \Phi(Lf(K_1\lambda + K_2)) - \Phi(Lf_(K_1(\lambda + \Delta\lambda) + K_2)). \tag{3.7}$$

Em razão do processo de arrendondamento aplicado sobre esta equação, apresentado pela equação 3.4, é possível que não haja diferença no cálculo da disparidade distorcida, ainda que distorções dos valores de profundidade ($\Delta\lambda$) sejam introduzidos. Dessa forma, este processo é caracterizado por um mapeamento "muitos-para-um" e Δd_u permanece o mesmo e igual a 0.

Assim, vários valores de distorção $\Delta\lambda_i$ geram uma mesma disparidade d_u. Conforme observado na Figura 3.1, em vez de um único valor λ_i ser mapeado para uma determinada disparidade, múltiplos pontos $\lambda_i + \Delta\lambda_i$, com $\Delta\lambda_i \in [-\Delta\lambda_i^-, \Delta\lambda_i^+]$, serão mapeados para a mesma disparidade d_i. Aqui, $\Delta\lambda_i$ é o valor distorcido $\Delta\lambda$ para a profundidade λ_i, sendo $-\Delta\lambda_i^-$ e $\Delta\lambda_i^+$ os limites inferior e superior para a distorção $\Delta\lambda_i$.

À vista disso, se os valores de λ_i e o valor com distorção $\lambda_i+\Delta\lambda_i$ estiverem contidos em um mesmo intervalo $[\lambda_i - \Delta\lambda_i^-, \lambda_i + \Delta\lambda_i^+]$ é possível definir que a diferença de disparidade é igual a zero e não haverá nenhum erro na sintetização na vista virtual. Este intervalo é chamado de Distorção da Profundidade Permitida (ADD) em síntese de vistas [Zhang

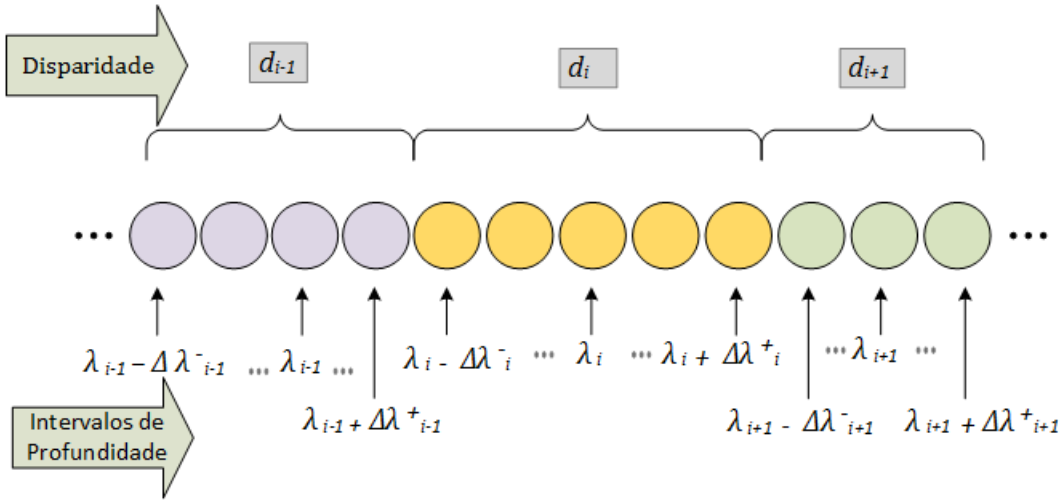


Figura 3.1: Ilustração dos intervalos ADD [Zhang et al., 2015]

et al., 2014]. Logo, pode-se explorar o intervalo para a escolha de valores que permitam maior compressão dos dados, sem que isto ocasione erros de sintetização.

Capítulo 4

Metodologia Proposta

O objetivo deste capítulo é explicar as definições e conceitos para os métodos empregados neste trabalho de dissertação, mostrando as premissas assumidas para a implementação dos algoritmos de pré-processamentos aplicados sobre os mapas de profundidade.

4.1 Visão Geral

Para este trabalho implementou-se os algoritmos de pré-processamento baseados no conceito ADD que foi abordado no Capítulo 3.

Baseado nesta ideia, duas abordagens foram feitas sobre o problema de otimização de codificação dos mapas de profundidade. A primeira abordagem desenvolvida foi intitulada Distorção da profundidade Admissível-Mediana (ADD-M) e utiliza o artigo de Zhang *et al* [Zhang et al., 2016] como base. O segundo algoritmo proposto neste trabalho foi denominado Mínima Variância no Bloco-ADD (ADD-MVB) e é uma implementação que aproveita a estrutura do codificador para realizar um pré-processamento sobre o mapa de profundidade a fim de que a eficiência de compressão seja maior em relação à primeira abordagem.

A Figura 4.1 mostra um gráfico produzido a partir do cálculo da disparidade pela Equação (2.15) para o mapa de profundidade do primeiro quadro da sequência Kendo [Tanimoto et al., 2008]. Neste gráfico são mostrados os valores de profundidade e o seus respectivos valores de disparidade calculados. Pode-se observar que vários valores de profundidade vizinhos pertencem à um mesmo intervalo ADD, pois derivam uma mesma disparidade. Os valores de um intervalo podem ser manipulados, de forma semelhante a uma filtragem, na tentativa de aumentar a eficiência de codificação.

Após a implementação dos métodos citados, utilizou-se a Equação 2.15 [Tian et al., 2009] e as premissas abordadas na seção 2.3.3 para a elaboração do algoritmo de síntese de vistas.

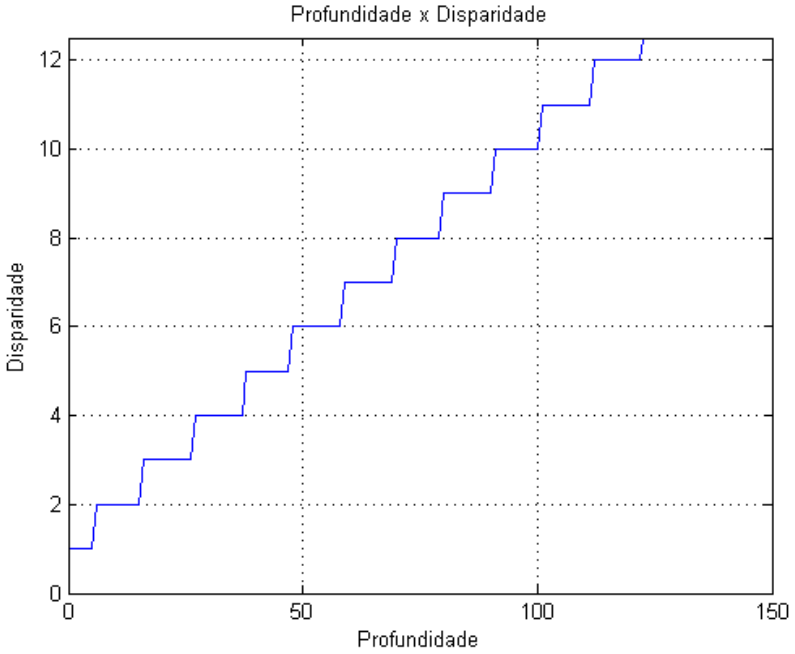


Figura 4.1: Agrupamento de valores de profundidade.

A síntese de vistas a partir dos mapas de profundidade pré-processados e original são comparados a fim de comprovar que nenhum erro de sintetização foi introduzido.

A Figura 4.2 ilustra o trabalho proposto de maneira geral por meio de um fluxograma, apresentando como este se encaixa no processo de sintetização de vistas virtuais.

Após o pré-processamento sobre os mapas de profundidades, utilizando as técnicas desenvolvidas, utilizou-se o codificador HEVC para a compressão dos dados referentes aos mapas de profundidade. O arquivo de configuração (encoder_lowdelay_P_main.cfg) utilizado no codificador foi modificado para que a compressão fosse realizada em um modo muito próximo ao modo sem perdas (*lossless*). Este arquivo pode ser verificado no Apêndice A. Os parâmetros alterados em relação aos arquivo padrão original foram: IntraPeriod, QP, TransquantBypassEnableFlag e CUTransquantBypassFlagForce.

4.2 ADD-M

Para realizar o pré-processamento ADD-M sobre os mapas de profundidades, utilizou-se a ideia explorada no artigo de [Zhang et al., 2016] que utiliza o conceito de Distorção de Profundidade Admissível (ADD) para modificar os mapas de profundidade. Além disso, é analisado nesta relação de "muitos-para-um"(processo que ocorre no momento da renderização baseada em imagem de profundidade) qual o melhor valor de substituição para um determinado intervalo ADD. No artigo de referência é proposto que o valor de subs-

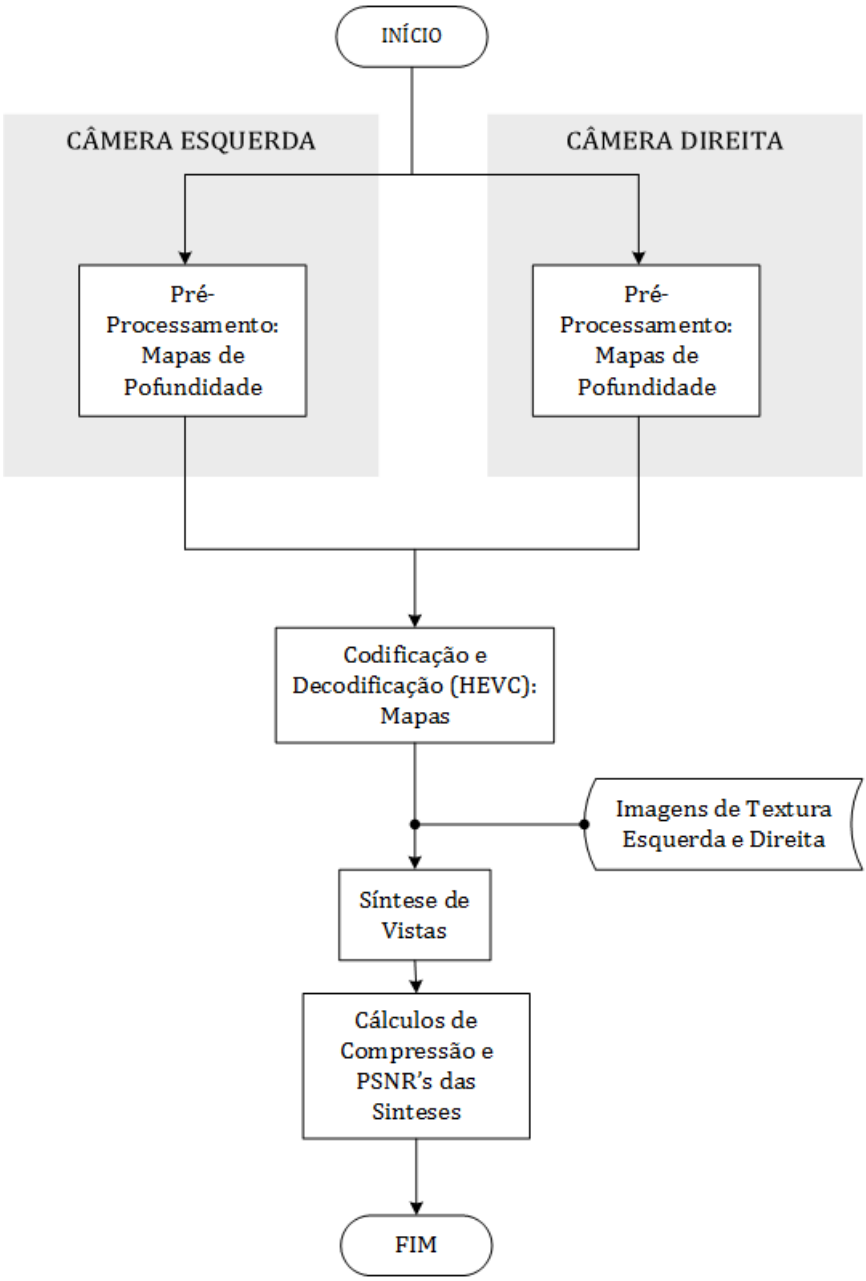
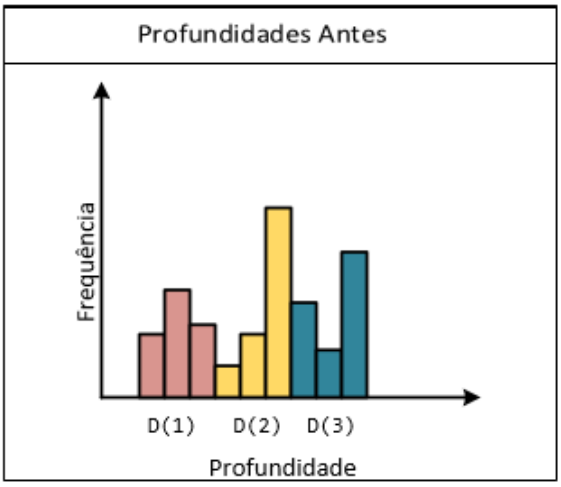
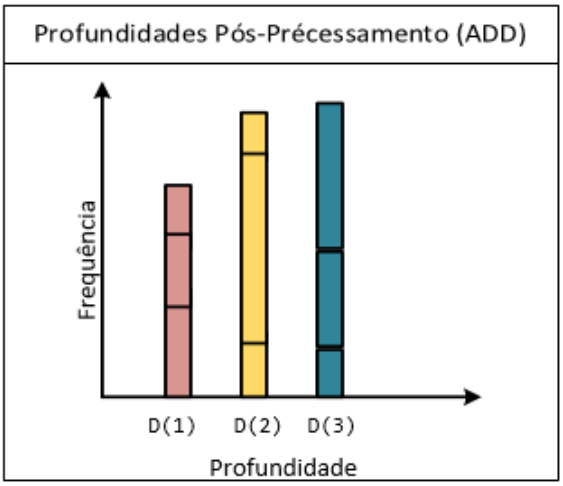


Figura 4.2: Fluxograma geral da metodologia proposta.

tituição seja a mediana do intervalo. Esta proposta visa minimizar o erro de sintetização para uma compressão com perdas. Realizou-se um estudo comparativo para determinar se esta escolha era válida também para uma compressão sem perdas, isto é, se alcançava as maiores taxas de compressão. Assim, deseja-se determinar se esta abordagem obteria maiores ganhos quando comparado ao método proposto neste trabalho de dissertação. A compressão sem perdas tem a vantagem de ser mais fidedigna à sintetização de vistas, pois não introduz nenhum erro pelo processo de compressão, uma vez que os mapas de profundidade descomprimidos são utilizados como informação para a técnica de projeção



(a) Intervalos ADD do mapa de profundidade original representados por diferentes cores.



(b) Profundidades agrupadas e assumindo um único valor no intervalo ADD.

Figura 4.3: Agrupamento conforme o conceito Distorção de Profundidade Admissível (ADD) Figura baseada em [Zhang et al., 2015]

de píxeis DIBR.

O método ADD-M tem como objetivo mudar os valores do mapa de profundidade baseado nos intervalos ADD, fazendo com que todos os píxeis que apresentam profundidades pertencentes a um mesmo intervalo assumam o valor mediano deste intervalo. Como ilustrado pelo gráfico da Figura 4.3a, cada uma das barras apresenta um valor de profundidade de um mapa de profundidade qualquer, em que cada cor diferente representa um grupo de valores de profundidade que fazem parte de um intervalo ADD distinto. Desta maneira, as profundidades podem ser agrupadas de forma a assumirem um único valor do intervalo como mostra a superposição das barras na Figura 4.3b. Dado

a ocorrência deste pré-processamento, prevê-se que os mapas de profundidade possuam menor entropia, pois vários valores de profundidade no intervalo assumem apenas um valor, diminuindo a variância no mapa de profundidade. O valor de profundidade λ_i' é, consequentemente, escolhido baseado no conjunto

$$\Lambda_i = \{\lambda^{min}, \lambda^{min} + 1, \ldots, \lambda^{max}\}, \tag{4.1}$$

conforme

$$\lambda_i' = \text{mediana}(\Lambda_i). \tag{4.2}$$

Outra forma de observar esta escolha de valores pode ser visto na ilustração da Figura 4.4. Nela há um objeto qualquer que está localizado em algum quadro de uma sequência de imagens e assume-se que os seus valores de profundidade estão todos situados em um mesmo intervalo ADD, representado pela cor do objeto. Após o pré-processamento ADD-M, estes valores de profundidade são todos substituídos por um único valor do intervalo que é representado na figura pelas linhas em diagonal. Esta abordagem não leva em consideração a localização do objeto e a possível divisão do quadro em blocos realizada pelo codificador.

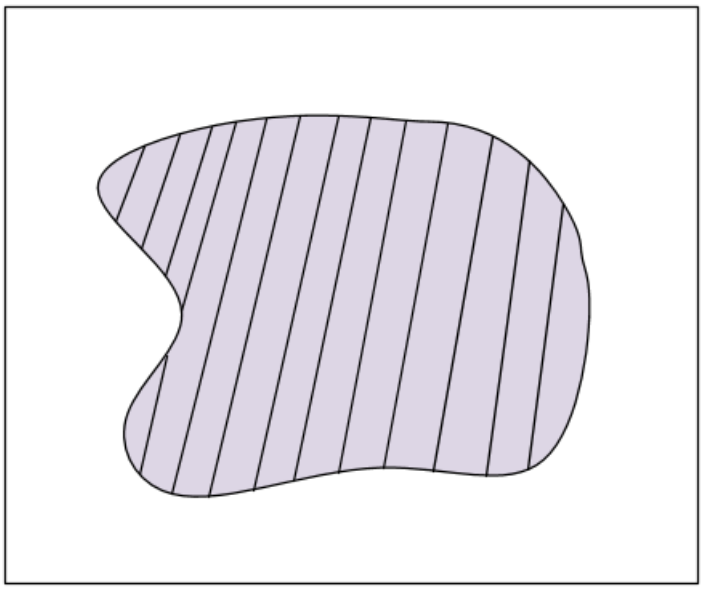


Figura 4.4: Objeto em uma imagem em que todos os seus valores pertencem a um mesmo intervalo ADD.

4.3 Mínima Variância no Bloco

A abordagem de Mínima Variância no Bloco-ADD (ADD-MVB), ao contrário da abordagem anterior, busca aproveitar a estrutura do codificador para alcançar um maior grau de compressão sobre os mapas de profundidade. Este algoritmo proposto foi apresentado no XXXIV Simpósio Brasileiro de Telecomunicações e Processamento de Sinais (SBrT) [Júlio et al., 2016].

Na codificação de imagens com o codec *High Efficiency Video Coding* (HEVC) [ITU-T, 2014], um menor valor de resíduo por Unidade de Codificação (UC) é desejável. Como explicado na seção 2.4 várias predições são realizadas sobre os quadros de uma sequência de imagens. Este trabalho está focado na predição intra-bloco que atua sobre as redundâncias espaciais no quadro sem levar em consideração as informações temporais (Inter-Bloco).

Propõe-se diminuir a variância dos valores de um mapa de profundidade dentro de uma mesma UC, tornando-a mais homogênea. Essa proposta é válida para qualquer codec baseado em codificação de blocos, não importando a escolha pelos codificadores específicos para múltiplas vistas, uma vez que este pré-processamento atua somente sobre os mapas de profundidade e tem como objetivo a otimização da codificação local destes. Utiliza-se a propriedade ADD para limitar a escolha do valor de profundidade em função dos seus valores vizinhos dentro do bloco a ser codificado.

Baseado nisso, elaborou-se um fluxograma do trabalho proposto que pode ser observado na Figura 4.5. Portanto, existe um mapa de profundidade e este é dividido em blocos de tamanho fixo e não sobrepostos sobre os quais serão analisadas as localizações dos píxeis e verificados os intervalos ADD aos quais os píxeis pertençam. Um estudo sobre o impacto da dimensão do tamanho do bloco também é realizado e mostrado na seção 5.

Desta forma, escolhe-se uma determinada medida estatística para o pré-processamento do bloco (UC) para a diminuição da variância. Foram testadas duas medidas: a média e a mediana. Ao respeitar os limites dos intervalos ADD para a substituição dos valores de profundidade, seleciona-se os valores mais próximos da medida estatística escolhida. Assim, dependendo do bloco em que o píxel está inserido, há uma escolha distinta do valor. Este método difere da Seção 4.2, pois aquele elege um valor de substituição para todos os valores do intervalo no intuito de diminuir a entropia total do mapa sem levar em consideração as divisões em blocos que ocorrem no momento da codificação e nem a localização dos píxeis.

Para ilustrar o funcionamento do algoritmo Mínima Variância no Bloco-ADD (ADD-MVB), pode-se observar na Figura 4.6 um objeto que tem suas partes localizadas em diferentes blocos, quando a imagem é dividida, sendo que todos os píxeis deste objeto pertencem à um mesmo intervalo ADD. Consequentemente, os valores que estes píxeis podem assumir dependem do intervalo ADD e do bloco no qual estão localizados.

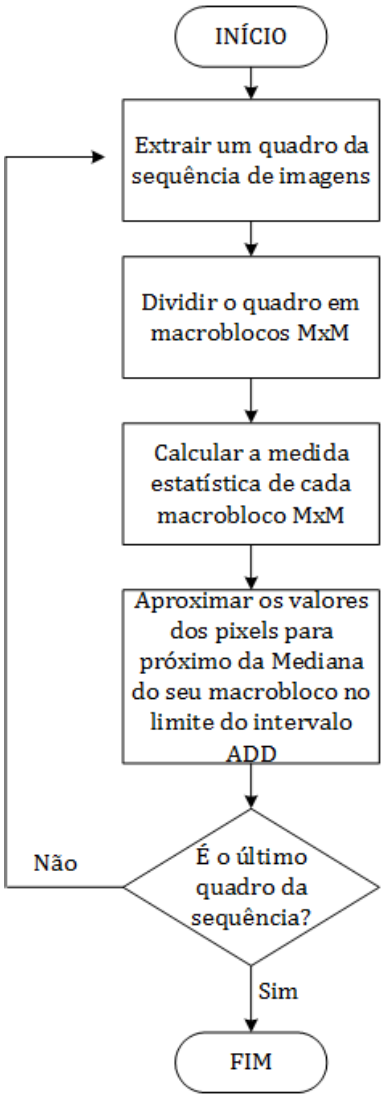


Figura 4.5: Fluxograma da Proposta Mínima Variância no Bloco-ADD (ADD-MVB)

Cada pixel verifica a média/mediana do seu bloco e assume um valor conforme os limites admitidos pelo seu intervalo ADD, diminuindo a variância dos valores de profundidade em seu bloco. Na Figura 4.6, as linhas em diferentes direções significam que para cada parte do objeto, vinculada a diferentes blocos, um valor diferente do intervalo foi escolhido e assumido, dependendo exclusivamente da medida estatística do bloco.

As medidas estatísticas escolhidas são consideradas medidas de tendência central pois representam um conjunto de valores que giram em torno deste. Assim, para a primeira abordagem do método ADD-MVB é testada a média como medida estatística a partir da qual os valores de um bloco são aproximados para a média deste. Denota-se o valor médio de um bloco como $\overline{m_i}$, calculado pela seguinte equação:

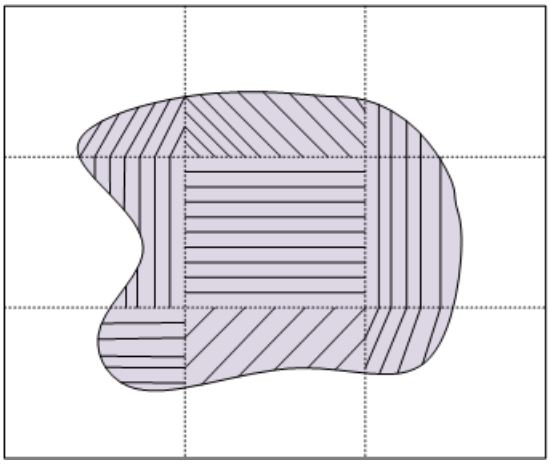


Figura 4.6: Objeto em uma imagem em que todos os seus valores pertencem a um mesmo intervalo ADD. Esolha para o método ADD-MVB.

$$\overline{m_i} = \frac{\sum\limits_{j \in \beta_i} \lambda_j}{M \times M}, \tag{4.3}$$

no qual β_i é o bloco i da imagem para o qual foi calculado o valor médio $\overline{m_i}$; j é a posição do pixel de profundidade λ_i dentro de um bloco β; M é o tamanho do bloco escolhido para o pré-processamento.

Para uma segunda abordagem do método ADD-MVB, calcula-se a mediana para os blocos da imagem. Assim é realizada a comparação entre a utilização da média e da mediana para determinar qual destas medidas apresenta melhor performance na compressão dos dados. A mediana é calculada se utilizando a seguinte equação:

$$\tilde{m_i} = \frac{1}{2}(\lambda_j + \lambda_{j+1}), \text{ tal que } j \in \beta_i \tag{4.4}$$

sendo $j = \frac{M \times M}{2}$, o elemento central de um conjunto de valores de profundidade ordenados, isto é, a mediana do conjunto. Como a quantidade de elementos deste conjunto de valores é par então o valor mediano do conjunto é dado pela média dos dois valores centrais, $\frac{\lambda_j + \lambda_{j+1}}{2}$. A mediana possui a vantagem de não ter seu valor afetado por valores extremos que estejam no conjunto de dados, como ocorre no uso da média.

Dado que se tem um conjunto Λ_i dentro do qual a profundidade λ_i poderá variar (intervalo ADD para a profundidade λ_i), o valor de substituição, λ_i'', baseado na menor variância será dado conforme

$$\lambda_i'' = \underset{a \in \Lambda_i}{\arg\min} \mid \overline{m_i} - a \mid \tag{4.5}$$

na qual a é o valor escolhido no intervalo ADD que se aproxima mais dos valor médio do bloco. Para a escolha do valor mais próximo da mediana do conjunto, equação é dada por

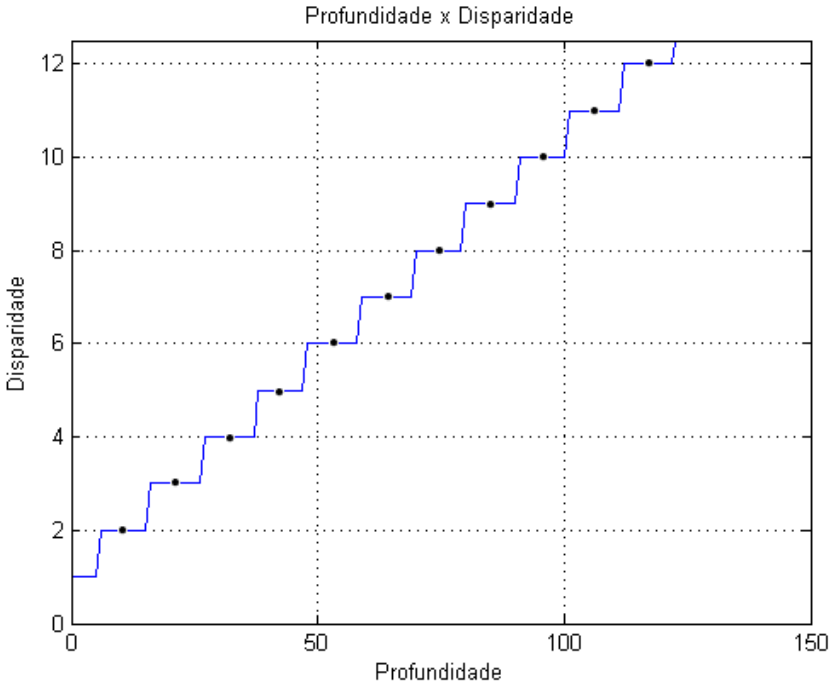
$$\lambda_i'' = \underset{a \in \Lambda_i}{\arg\min} \mid \tilde{m}_i - a \mid. \tag{4.6}$$

Logo, os valores de profundidade aproximam-se da medida estatística para um determinado bloco a depender da abordagem escolhida. Estes valores de profundidade variam no máximo que o intervalo ADD admita, pois se assumirem valores fora deste intervalo é possível a ocorrência de erros na projeção, causando uma distorção na síntese de vistas.

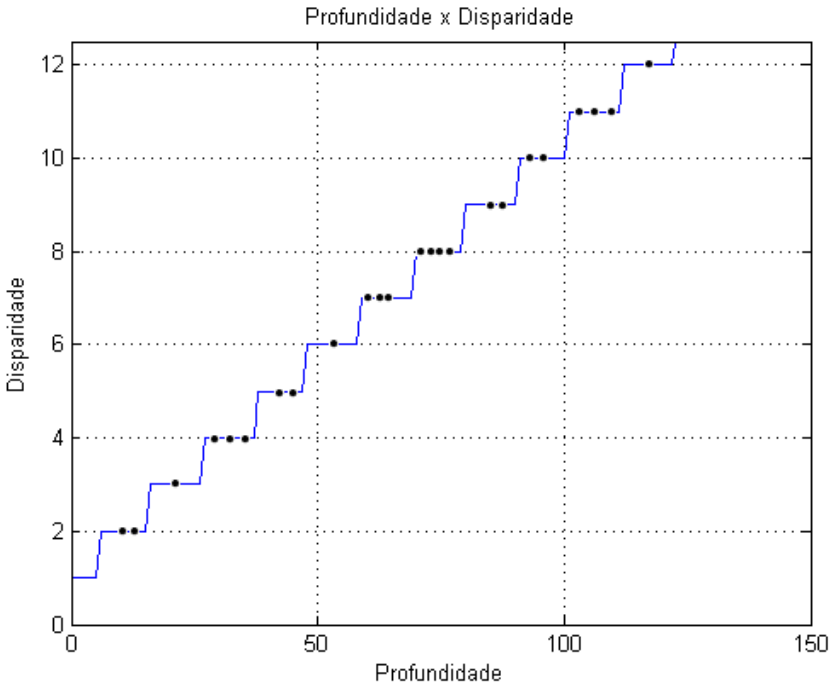
Na Figura 4.7 pode-se observar mais claramente as diferenças entre os dois métodos, ADD-M e ADD-MVB. Na abordagem ADD-M, mostrado no gráfico da Figura 4.7a escolhe-se o valor mediano do intervalo ADD para a substituição das profundidades que pertencem àquele intervalo, diminuindo a entropia total do mapa, pois menos valores de profundidade estão presentes no mapa de profundidade. Diferentemente, verifica-se na abordagem ADD-MVB representada pela Figura 4.7b que vários valores de um mesmo intervalo podem ser escolhidos e usados para a substituição das profundidades e isto dependerá exclusivamente da medida estatística para aquele bloco, acarretando uma maior entropia do mapa de profundidade. Apesar disso, como será visto no capítulo 5, este pré-processamento aumenta a compressão sobre os mapas de profundidade. Este ganho de compressão pode ser intuído pelo fato dos valores nos blocos estarem mais próximos, possuindo menor resíduo, principalmente entre blocos que representam uma mesma região. Como exemplo o *background* de uma imagem que possui valores de profundidade mais semelhantes.

Por último, investiga-se uma abordagem realizada sobre o método ADD-MVB que busca a definição do tamanho de bloco de tamanho não fixo em função do conteúdo dos mapas de profundidade. O tamanho é agora adaptado ao conteúdo, tentando englobar a maior homogeneidade dos valores de profundidade.

Esta abordagem foi proposta ao se notar uma melhor eficácia de compressão quando os blocos distinguiam regiões semelhantes, englobando-as. Este experimento tem como objetivo escolher blocos que separem os espaços que pertençam à regiões similares de profundidade, como por exemplo, separando o *background* do *foreground*. O tamanho destes blocos é variável e tenta causar a melhor separação das regiões em uma cena. Após a escolha dos blocos, é aplicada igualmente uma medida estatística e realizado o mesmo



(a) ADD-M: substituição dos valores de profundidade por um único valor (mediano) do intervalo.



(b) ADD-MVB: para a substituição dos valores de profundidade são assumidos diversos valores (indicado pelos pontos) dentro de um mesmo intervalo.

Figura 4.7: Agrupamento conforme o conceito Distorção de Profundidade Admissível (ADD) Figura baseada em [Zhang et al., 2015]

procedimento do método ADD-MVB. Este método foi testado em algumas sequências e seus resultados preliminares são apresentados no próximo capítulo.

4.4 Síntese de Vistas

A síntese de vistas foi elaborada conforme as Equações apresentadas na seção 2.3.3. Foi utilizado o método DIBR para a projeção das vistas de referência para a imagem virtual.

A síntese é realizada para comparação das imagens virtuais geradas a partir dos mapas originais e dos pré-processamentos. Dessa forma, verifica-se se houve a introdução de alguma distorção causada pelo pré-processamento.

Para a síntese foi usada a componente Y (luminância) do mapa de profundidade. A métrica utilizada para a medição de qualidade da imagem sintetizada (virtual) foi o PSNR sobre somente a componente Y.

O PSNR foi a métrica escolhida, pois esta verifica o quanto a imagem está diferente estruturalmente em relação à original. Outras métricas que são mais focadas na percepção do usuário não são muito válidas neste momento, pois o objetivo é medir exatamente o quanto um pixel diferiu de outro no momento da projeção, tendo como objetivo final a não distorção da imagem sintetizada. Logo, o resultado final esperado é que os PSNR's sejam idênticos entre as imagens sintetizadas com os mapas de profundidade original e com o pré-processado e desse modo verificar que não houve nenhuma adição de ruído na imagem.

Capítulo 5

Resultados

Neste capítulo são expostos os resultados alcançados com a utilização das técnicas explanadas no capítulo 4, mostrando mais detalhes na aplicação dos métodos mostrados nas seções 4.2 e 4.3.

O conjunto de teste foi composto por uma base de dados pública disponibilizada pela a Universidade de Nagoya. Desta base foram testadas as sequências: Kendo [Tanimoto et al., 2008], Balloons [Tanimoto et al., 2008], Poznan Street [Domanski et al., 2009], Pantomime [Tanimoto et al., 2005] [Tanimoto et al., 2008] e Lovebird1 [JTC1/SC29/WG11, 2008] e Akko-Kayo[Tanimoto et al., 2008], conforme ilustrado pela Figura 5.1. As sequências estão regularmente dispostas em um arranjo 1D, e apresentam diferentes resoluções e distâncias de *baseline*. Todas as sequências estão descritas na Tabela 5.1.

(a) Kendo Vista 1

(b) Balloons Vista 1

(c) Poznan Street Vista 3

(d) Pantomime Vista 37

(e) Lovebird1 Vista 4

(f) Akko-Kayo Vista 47

Figura 5.1: Primeiro quadro de cada uma das sequências utilizadas no teste.

Para se gerar os resultados foi utilizado apenas o primeiro quadro para cada uma das sequências, pois não existem relações temporais nos métodos utilizados, realizando-se um

pré-processamento Intra-Bloco. Para a codificação das imagens foi utilizado o codificador *High Efficiency Video Coding* (HEVC) [ITU-T, 2014] em modo sem perdas (*lossless*). Este codificador foi escolhido pois faz parte de um padrão já disseminado e amplamente utilizado na indústria.

Tabela 5.1: Descrição das sequências usadas para teste

Sequência	Resolução	Vistas Referência	Vista Sintetizada
Kendo	1024x768	1,3	2
Balloons	1024x768	1,3	2
Poznan St.	1920x1088	3,5	4
Pantomime	1280x960	37,39	38
Lovebird1	1024x768	4,6	5
Akko & Kayo	640x480	47,49	48

Os pré-processamentos foram aplicados para o primeiro quadro tanto da vista esquerda quanto da vista da direita para uma determinada sequência de teste. Após, a síntese de vistas foi realizada, utilizando os mapas pré-processados e verificando se houve alteração na imagem virtual produzida.

Para medir a eficiência das compressões foi utilizada a taxa média de bits gastas por pixel, por câmera (bppc). As comparações são feitas entre as compressões dos mapas de profundidade utilizando os métodos citados no capítulo anterior e a compressão do mapa de profundidade original, sem nenhum tipo de pré-processamento.

5.1 Aplicação do método ADD-M

A partir das premissas assumidas na seção 4.2, este método visa a agrupar as profundidade de um mapa de profundidades que produzem o mesmo valor no cálculo de disparidade, ou seja, que pertençam a um mesmo intervalo ADD. Dessa forma, o método substitui todos os valores de um mesmo intervalo por um único valor. Para este método é escolhida a mediana de cada intervalo ADD para a substituição de todos os valores que existem neste mesmo intervalo.

Dois histogramas, mostrados na Figura 5.2, comparam a diminuição da quantidade de profundidades representadas no mapa de profundidade depois do pré-processamento com o método ADD-M. Observa-se a diminuição da quantidade de profundidades presentes no histograma da Figura 5.2b em relação ao mapa original representado pelo histograma da Figura 5.2a. Estes histogramas apresentam no eixo das abcissas os valores de profundidades e no eixo vertical a quantidade dessas profundidades para cada imagem.

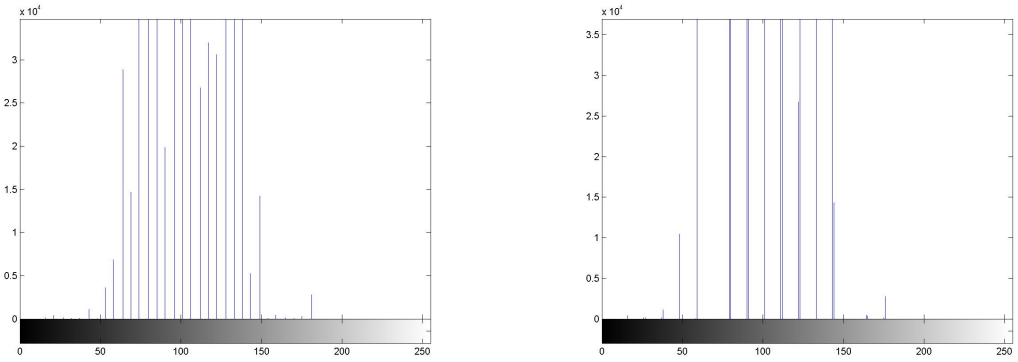
Tabela 5.2: Comparação da eficiência de compressão do método ADD-M em relação à compressão do mapa de profundidade original. Resultados mostrados em bpp/c e em percentual (%) em relação à compressão do mapa original.

Sequência	Original(bpp/c)	ADD-M (bpp/c)	ADD-M (%)
Kendo	0,1114	0,0850	-23,64
Balloons	0,1671	0,1145	-31,45
Poznan St.	0,4767	0,3315	-30,46
Pantomime	0,0990	0,0876	-11,47
Lovebird1	0,2924	0,1412	-51,70
Akko & Kayo	0,6315	0,4473	-29,16

Os resultados mostram que a eficiência de codificação aumenta à medida que a variação dos valores de profundidade diminui nos mapas, conforme observa-se na Tabela 5.2. Os resultados mostram que houve um grande ganho de compressão dos dados dos mapas de profundidades quando comparados à compressão de suas versões originais.

A Tabela 5.2 apresenta as compressões realizadas para seis sequências do banco de dados fornecido pela universidade de Nagoya e mostra na primeira coluna o nome das sequências utilizadas, seguida pela quantidade de bits gastos por câmera na compressão do mapa de profundidade original. A terceira coluna apresenta a quantidade de bits gastos para compressão dos mapas pré-processados pelo método ADD-M. A última coluna relaciona o nível de porcentagem de ganho do método ADD-M à compressão original. Os resultados negativos em (%) mostram que houve a compressão dos dados. Há a expansão dos dados quando os resultados são apresentados com valor de ganho positivo.

Os valores apresentados na Tabela 5.2 mostram níveis de compressões melhores quando



(a) Mapa de Profundidade Original (b) Pré-Processado com o método ADD-M

Figura 5.2: Histogramas dos Mapas de Profundidade da câmera 1, quadro 0 da sequência Kendo [Tanimoto et al., 2008].

realizado algum tipo de pré-processamento sobre os mapas de profundidade. A quantidade de bits gastos com a utilização da técnica de pré-processamento ADD-M é inferior à compressão do mapa de profundidade original. Isso mostra como a diminuição da entropia do mapa de profundidade possui influência sobre os níveis de compressão.

Na Figura 5.3 são mostradas as imagens referentes a textura e profundidade de duas sequências, Pantomime e Lovebird1. Em seguida, são apresentados os histogramas de seus mapas de profundidade. Essas duas sequências são mostradas em mais detalhes, pois foram as que apresentaram pior e melhor eficiência de compressão respectivamente, conforme a Tabela 5.2.

Identificou-se que a sequência Pantomime apresenta o menor ganho de compressão em relação a todas as outras sequências. Uma explicação possível é o fato desta sequência possuir poucos valores de profundidade presentes no seu mapa, como visto na Figura 5.3e. Isto faz com que o grau de eficiência de compressão do método de agrupamento ADD-M seja menor.

A Tabela 5.3 apresenta um resumo das informações dadas pelos histogramas de cada uma das seis sequências. Esta tabela apresenta, na segunda coluna, a quantidade de valores profundidade presentes nos respectivos histogramas. A terceira coluna mostra a quantidade de intervalos ADD usados durante o processo de agrupamento em relação à quantidade de intervalos possíveis para aquela sequência e a última coluna apresenta o comprimento médio destes intervalos ADD.

Pode-se constatar que a quantidade de valores de profundidade presentes no histograma (9) para a sequência Pantomime, é muito pouca e esparsa. Por conta disso, estes valores acabam por utilizar 5 dos 21 intervalos ADD possíveis, causando menos impacto na transformação da imagem pré-processada. Assim, o histograma da imagem que antes continha 9 valores passa a ter 5, apresentando pouca diferença quando comparado aos agrupamentos dos outros histogramas.

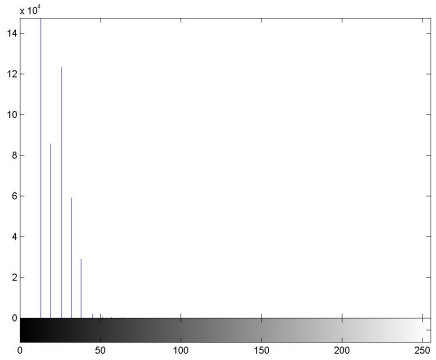
Em comparação à sequência Pantomime, a sequência Lovebird1 possui muitos valores profundidade em seu mapa (40), conforme é verificado pela Figura 5.3f. Estes valores estão muito próximos uns dos outros, e, portanto, muitos acabam pertencendo a um mesmo intervalo ADD, fazendo com que a entropia de primeiro grau deste mapa diminua significantemente. Desta forma, consegue agrupar mais valores por intervalo, aumentando a eficiência de compressão dessas informações. O seu histograma original que antes possuía 40 valores, agora apresenta somente 9 valores após o pré-processamento. A transformação que ocorre em Lovebird1 é muito maior do que a que ocorre com a sequência Pantomime. Este fato influencia a compressão de dados e explica a menor eficiência alcançada por esta sequência em relação às outras. Assim, pode-se verificar na Tabela 5.3 a relação da quantidade de intervalos ADD utilizados e a eficiência de compressão apresentada por

(a) Imagem de Textura-Pantomime

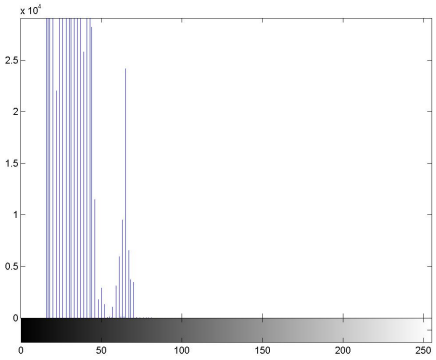
(b) Imagem de Textura-Lovebird1

(c) Mapa de Profundidade-Pantomime

(d) Mapa de Profundidade-Lovebird1



(e) Histograma-Pantomime



(f) Histograma-Lovebird1

Figura 5.3: (a) e (c) Textura e Mapa de Profundidade para o primeiro quadro da sequência Pantomime [Tanimoto et al., 2005], câmera 37. (b) e (d) Textura e Mapa de Profundidade para o primeiro quadro da sequência Lovebird1 [JTC1/SC29/WG11, 2008], câmera 4. Os histogramas dos mapas de profundidade são apresentados na figuras (e) e (f).

cada uma das sequências.

Tabela 5.3: Comparação entre a quantidade de agrupamento para cada sequência após a aplicação do método ADD-M. A coluna Profundidades representa a quantidade de profundidades presentes no histograma dos respectivos mapas de profundidade. Intervalos ADD apresentam a quantidade de intervalos que foram utilizados em relação à quantidade de intervalos ADD disponíveis. A coluna Comprimento apresenta o comprimento médio para do intervalo ADD e a coluna Agrupamento representa a quantidade em (%) da diminuição dos valores de profundidade presentes nos mapas.

Sequência	Profundidades	Intervalos ADD	Comprimento	Agrupamento
Kendo	32	16/25	10	50 %
Balloons	43	22/25	10	48.8 %
Poznan St.	174	55/80	3	68 %
Pantomime	9	5/21	12	44.5 %
Lovebird1	40	9/36	7	77.5 %
Akko & Kayo	19	8/8	32	57.9 %

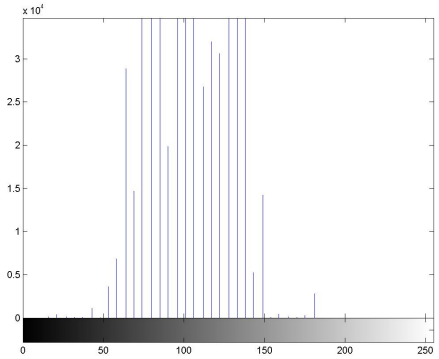
5.2 Aplicação do método ADD-MVB

O método proposto neste trabalho, que foi intitulado Mínima Variância no Bloco-ADD (ADD-MVB), é semelhante à abordagem definida como ADD-M que é baseada na Equação (4.2). A diferença no método ADD-MVB é a escolha dos valores de substituição, pois adota o valor do intervalo ADD que cause a mínima variância de profundidades em um determinado bloco ou UC no processo de codificação do mapa de profundidade de acordo com as Equações (4.5) e (4.6).

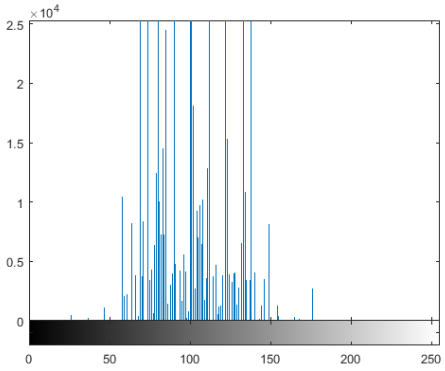
Como explanado na seção 4.3, os valores escolhidos no intervalo ADD dependem exclusivamente da localização do bloco e do respectivo valor da medida estatística escolhida: média ou mediana. Com isso, a escolha de valores dentro de um intervalo pode variar desde a escolha de um único valor até a escolha de todos os valores.

Como esta escolha de valores é variável isto faz com que a distribuição dos valores de profundidade diminua ou aumente no histograma do mapa de profundidade pré-processado. Um exemplo de histogramas de mapas de profundidade pré-processados são apresentados nas figuras 5.4b e 5.4c em comparação ao histograma do mapa de profundidade original presente na Figura 5.4a. O eixo das abcissas representa os valores de profundidade possíveis que podem existir no mapa de profundidade, enquanto que o eixo das ordenadas apresenta a quantidade de píxeis que possuem um determinado valor de profundidade.

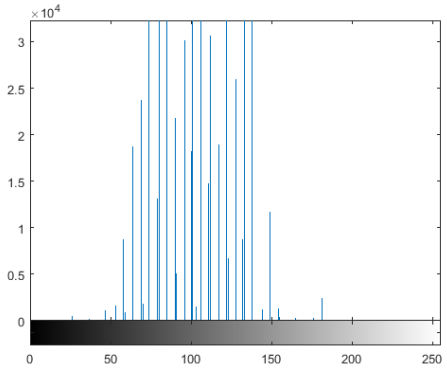
Por meio dos histogramas é possível perceber que há a escolha e a substituição de valores antigos de profundidade por novos valores diferentes dos de antes, isto é, nota-se que um mesmo valor de disparidade foi modificado para assumir diferentes valores



(a) Mapa de Profundidade Original



(b) Pré-Processado com o método ADD-MVB utilizando a Média



(c) Pré-Processado com o método ADD-MVB utilizando a Mediana

Figura 5.4: Histogramas dos Mapas de Profundidade da câmera 1 para o *frame* 0 da sequência Kendo [Tanimoto et al., 2008] com tamanho fixo de bloco igual a 64.

vizinhos e que fazem parte do seu intervalo ADD após o pré-processamento. Portanto, ocorre a diminuição da concentração de valores e observa-se um crescimento na entropia dos mapas de profundidade pré-processados com o método ADD-MVB tanto para uma medida estatística quanto para a outra.

Realizando uma comparação entre a análise do histograma apresentado na Figura 5.2b, o mapa de profundidade pré-processado com o método ADD-M possui menos valores de profundidade, o que acarreta em uma entropia igual a 3.0566, menor do que a entropia encontrada no mapa de profundidade original que é igual a 3.9172. Ao contrário do método ADD-M, o método ADD-MVB apresenta, para a média como medida estatística, entropia igual a 5.2112. O mesmo método utilizando a mediana possui entropia igual a 4.2291 após o pré-processamento. Ambas as abordagens para o o método ADD-MVB possuem uma maior entropia quando comparadas aos outros dois mapas de profundidades, o original e o pré-processado com o método ADD-M. Esta maior entropia pode ser explicada, como

citado anteriormente, pelo fato do algoritmo escolher qualquer valor dentro do intervalo ADD, podendo escolher inclusive todos.

Para se realizar uma análise mais abrangente sobre o método ADD-MVB, foram escolhidos diversos tamanhos de blocos para verificar se a eficiência de compressão mudaria conforme um certo tamanho escolhido. Investigou-se, conjuntamente aos diferentes tamanhos de blocos, diferentes medidas estatísticas para a verificação da mínima variância em um bloco. As medidas adotadas para teste, como já citado, foram a média e a mediana.

Para a escolha do tamanho de bloco não sobreposto ideal para a divisão da imagem, foram testados diferentes tamanhos: $M = 8 \times 8$, $M = 16 \times 16$, $M = 32 \times 32$, $M = 64 \times 64$, $M = 128 \times 128$ e $M = 256 \times 256$. Para cada sequência foi determinada uma das medidas estatísticas e testado cada um dos tamanhos de blocos. Assim, verifica-se qual a melhor medida e tamanho para uma compressão mais eficiente.

No gráfico da Figura 5.5 são mostrados os ganhos de compressão em porcentagem em relação ao resultado de compressão do mapa original para cada tamanho de bloco e para cada medida estatística. Neste gráfico pode-se observar a descrição dos tamanhos de blocos no eixo das abcissas e a eficiência de compressão no eixo das ordenadas. As duas medidas estatísticas foram comparadas. O resultado utilizando a mediana como fator para a mínima variância está marcado pela linha sólida, enquanto que o resultado utilizando a média é mostrado por linhas pontilhadas. Pode-se observar que o ganho em compressão cresce à medida que se aumenta o tamanho dos blocos.

Geralmente, a partir do tamanho $M = 128 \times 128$, a eficiência de compressão que antes crescia com o aumento do tamanho do bloco, passa a cair. É possível que, a depender do conteúdo do mapa de profundidade, tamanhos grandes de blocos passem a conter valores de profundidades muito diferentes e com intervalos ADD distantes, mesmo quando pré-processados para tornarem-se mais homogêneo, acabam por continuarem com um valor residual muito variante no momento da compressão, o que não ocasiona uma melhor eficiência de compressão. O mesmo ocorre com tamanhos de blocos muito pequenos como, por exemplo, os tamanhos $M = 8 \times 8$ e $M = 16 \times 16$. Em muitos casos, estes tamanhos acabam englobando regiões com pouca ou nenhuma diferença de valores de profundidade, fazendo com que o cálculo da mediana para aquele bloco não altere significantemente os valores de profundidade do mesmo.

Os resultados do gráfico são mostrados também em formato de tabela. Desta forma, as Tabelas 5.4 e 5.5 mostram o ganho em porcentagem das duas abordagens em relação ao original para o método ADD-MVB usando as medidas de média e mediana respectivamente. As tabelas mostram em suas colunas os diferentes tamanhos de blocos testados. Constata-se que o tamanho de bloco que alcança maior nível de compressão varia de acordo com a sequência e também com a medida estatística utilizada. Por exemplo, para

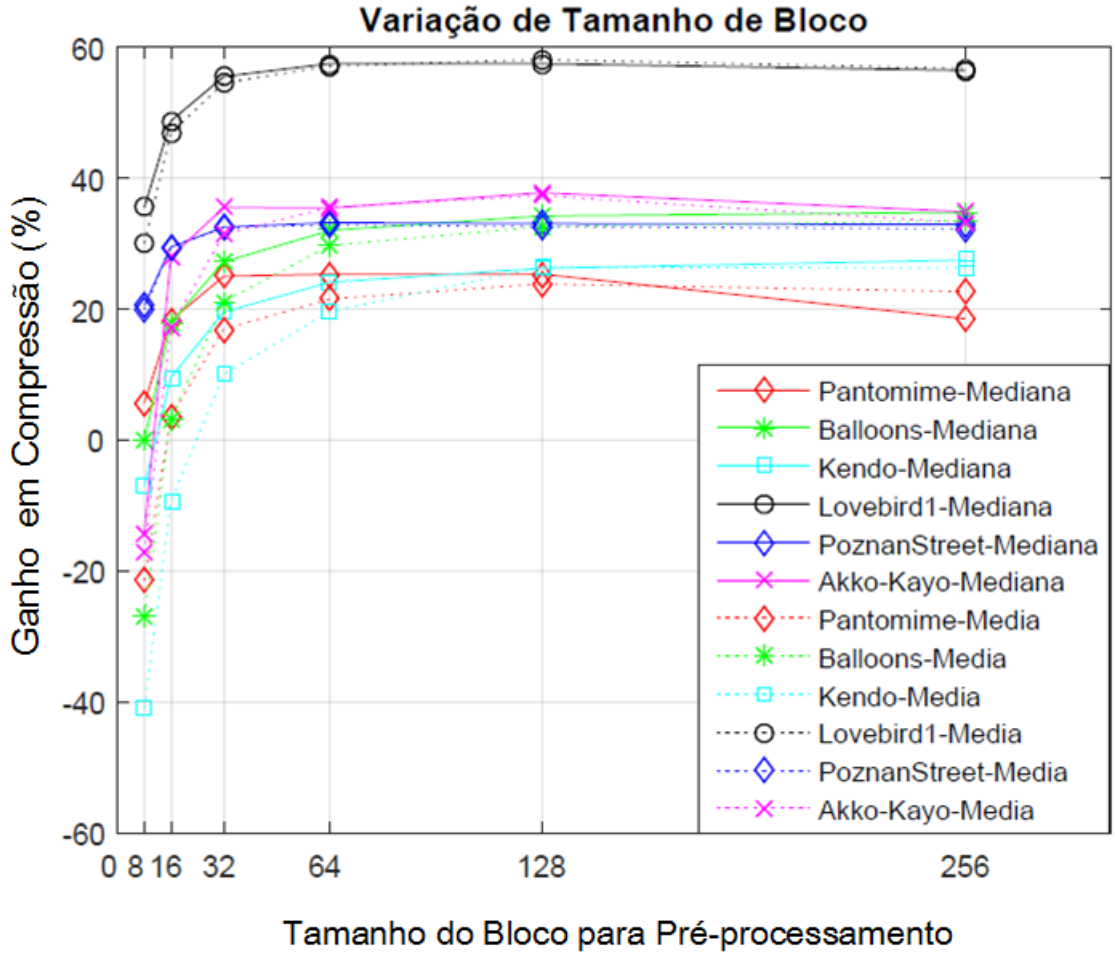


Figura 5.5: Variação do tamanho de bloco para o pré-processamento. As linhas pontilhadas apresentam os resultados do ganho em eficiência de compressão em (%) para o método ADD-MVB em relação ao método ADD-M, usando como medida estatística a média. As linhas sólidas apresentam o mesmo tipo de resultado para a medida estatística da mediana.

a medida estatística da média o tamanho ideal médio foi $M = 128 \times 128$, enquanto para a mediana o melhor tamanho médio foi $M = 64 \times 64$.

Na Tabela 5.6 são apresentados novamente os ganhos de compressão para cada um dos métodos abordados, ADD-M e ADD-MVB, em comparação à compressão dos mapas originais. Esta tabela facilita a comparação dos resultados encontrados para cada um dos métodos e apresenta em cada uma das linhas o melhor resultado para uma determinada sequência e para um determinado método. Ao lado dos resultados para as abordagens do método ADD-MVB, é mostrado um valor entre parênteses, este valor está associado ao resultado do melhor tamanho de bloco. Desta forma, torna-se mais fácil visualizar os melhores resultados encontrados.

Tabela 5.4: Eficiência de compressão dada em (%) em relação à compressão dos mapas de profundidade originais para diferentes tamanhos de blocos, usando o método ADD-MVB para a medida estatística da Média.

Sequência	8	16	32	64	128	256
Kendo	+40.91	+9.36	-10.04	-19.63	-26.46	-26.29
Balloons	+26.80	-3.29	-21.15	-29.74	-32.62	-33.49
Poznan St.	-19.96	-29.53	-32.43	-32.88	-32.62	-32.27
Pantomime	+21.42	-3.63	-16.96	-21.55	-23.87	-22.69
Akko & Kayo	+17.18	-17.22	-31.47	-35.58	-37.36	-33.19
Lovebird1	-30.08	-46.90	-54.55	-57.18	-58.12	-56.77

Tabela 5.5: Eficiência de compressão dada em (%) em relação à compressão dos mapas de profundidade originais para diferentes tamanhos de blocos, usando o método ADD-MVB para a medida estatística da Mediana.

Sequência	8	16	32	64	128	256
Kendo	+7.06	-9.57	-19.57	-24.12	-26.25	-27.50
Balloons	-0.12	-17.94	-27.25	-32.02	-34.25	-34.79
Poznan St.	-20.61	-29.49	-32.54	-33.23	-33.08	-32.96
Pantomime	-5.70	-18.29	-25.05	-25.36	-25.36	-18.54
Akko & Kayo	+14.19	-28.02	-35.55	-35.45	-37.79	-34.89
Lovebird1	-35.59	-48.76	-55.55	-57.52	-57.52	-56.48

Fazendo-se uma comparação entre os ganhos percentuais dos métodos ADD-M e ADD-MVB, percebe-se que houve uma diferença média de ganho de 5.72% a mais para a medida estatística da Média em relação aos ganhos do método ADD-M. A sequência Pantomime obteve a maior diferença de ganho percentual, sendo de 12.4% a mais em relação ao ganho ADD-M e a sequência Balloons a pior diferença de ganho, 2.04%. Já quando o método utiliza a medida estatística da mediana, têm-se que a diferença de ganho médio foi 6.38% em relação ao método ADD-M e que as sequências com melhor e o pior diferença de ganho são, respectivamente, Pantomime com 13.9% e a Poznan com 2.77%.

Por fim, é válido ressaltar que as sínteses de vistas geradas utilizando tanto os mapas originais quanto os mapas pré-processados não apresentam diferenças, logo conclui-

Tabela 5.6: Comparação da eficiência de compressão dos mapas de profundidade dada em (%) para os métodos ADD-M, ADD-MVB-média e ADD-MVB-mediana em relação à compressão original dos mapas de profundidade. Os resultados do método ADD-MVB apresentam valores entre parênteses que representam qual tamanho de bloco obteve o melhor resultado.

Sequência	Método ADD-M	Método ADD-MVB-média	Método ADD-MVB-mediana
Kendo	-23.64	-26.46 (128)	-27.50 (256)
Balloons	-31.45	-33.49 (256)	-34.79 (256)
Poznan St.	-30.46	-32.62 (128)	-33.23 (64)
Pantomime	-11.47	-23.87 (128)	-25.36 (64)
Akko & Kayo	-29.16	-37.36 (128)	-37.79 (128)
Lovebird1	-51.70	-58.12 (128)	-57.52 (64)

se que não houve nenhuma distorção introduzida pela aplicação dos métodos de pré-processamento.

Os PSNR's das imagens sintetizadas (virtuais) tanto pelos mapas originais quanto pelos mapas pré-processados, quando comparados à imagem *ground truth* [1], são idênticos.

5.2.1 Escolha de tamanho e posição de blocos adaptado ao conteúdo

Durante o processo de aferimento dos resultados apresentados na seção anterior, notou-se que a escolha do tamanho de blocos tinha grande influência sobre a eficiência de compressão. Observou-se que blocos escolhidos que englobavam grandes áreas similares obtinham maior eficiência de compressão em relação aos blocos que continham valores de profundidade muito distintos e, portanto, pertencentes a intervalos ADD muito distantes.

Com isso, elaborou-se uma abordagem para a escolha de blocos que atuassem na separação destas regiões mais semelhantes do mapa de profundidade. A fim de comparar a eficiência da abordagem, realiza-se algo análogo a uma segmentação de forma manual baseada na análise do conteúdo de profundidade dos mapas. Escolhe-se, para tanto, blocos não sobrepostos e de tamanho não fixos que contenham valores de profundidade com intervalos ADD próximos.

Depois de escolhido os blocos, estes passam pelo pré-processamento ADD-MVB, conforme explicado na seção 4.3. Assim, os valores de profundidade dos blocos são aproximados de uma medida estatística representativa para o mesmo, respeitando os limites

[1] Imagem real da câmera naquela posição

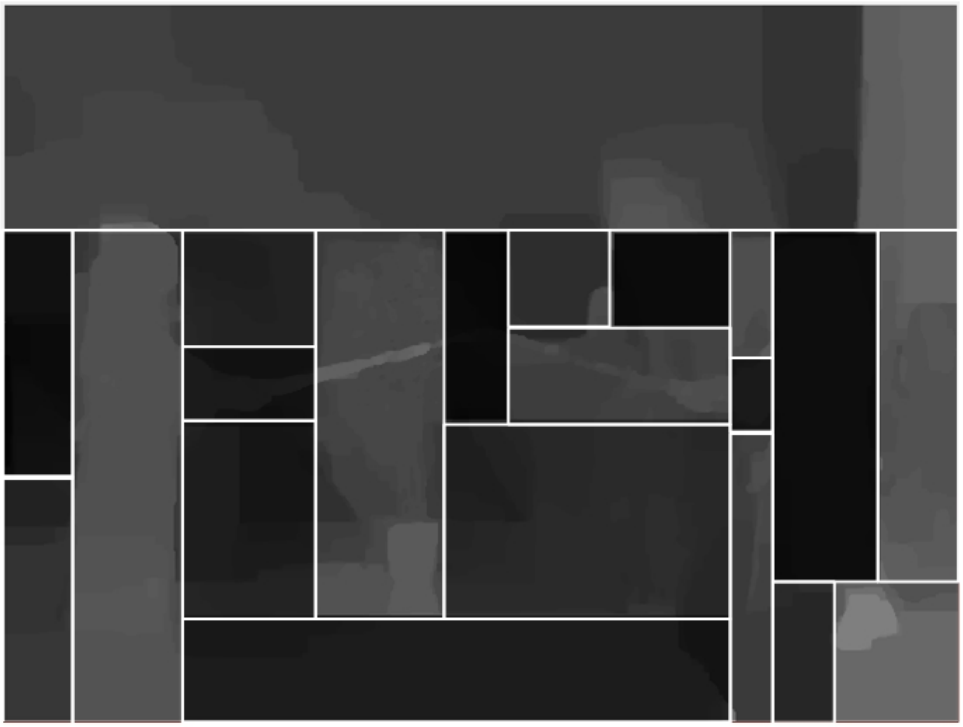
dos intervalos ADD para os píxeis. Os blocos são escolhidos no intuito de separar da melhor forma regiões semelhantes da imagem de profundidade. Foram testadas para esta abordagem, inicialmente, duas sequências: Kendo e Pantomime.

O resultado visual da escolha destes blocos pode ser verificado na Figura 5.6. A Figura 5.6a representa a escolha de blocos para o primeiro quadro da sequência Kendo e a Figura 5.6b para a sequência Pantomime. Os limites dos blocos são determinados pelas linhas brancas e buscam conter valores semelhantes de profundidade, por exemplo, causando em muitos casos uma separação entre regiões de *background* ou de *foreground*. Dessa forma, consegue-se obter uma certa segmentação das regiões do mapa de profundidade.

Essa abordagem, que se encontra em fase inicial, conseguiu obter ganhos superiores em relação aos melhores resultados para estas sequências no método ADD-MVB. O ganho para a sequência Kendo foi de 0.57% a mais que o melhor ganho com o método ADD-MVB, 27.50%, utilizando a medida estatística da mediana e tamanho de bloco igual a 256.

Para a sequência Pantomime os ganhos foram melhores, pois o mapa de profundidade desta sequência possui regiões que estão bem definidas, sem muito ruído. O ganho nessa abordagem foi 2.18% maior que o melhor resultado obtido nos métodos explorados anteriormente. Antes, o melhor resultado encontrado era de 25.36%.

Portanto, pode-se explorar ainda mais a eficiência de compressão dos mapas de profundidade ao se escolher blocos que separem da melhor forma regiões semelhantes do mapa, dando maior homogeneidade à estas regiões.



(a) Sequência Kendo [Tanimoto et al., 2008]

(b) Sequência Pantomime [Tanimoto et al., 2005]

Figura 5.6: Escolha de blocos não sobrepostos e de tamanho não fixo sobre a sequência Kendo e Pantomime. Esta abordagem visa a separar regiões de profundidades similares.

Capítulo 6

Conclusões

Neste trabalho é proposto um algoritmo que realiza um pré-processamento, utilizando o conceito Distorção de Profundidade Admissível (ADD) para realizar um processo semelhante à uma filtragem nos valores de profundidade dos mapas de profundidades das câmeras de referência em um sistema de múltiplas vistas. Os valores de profundidade que geram um mesmo valor de disparidade calculado são reunidos em um intervalo denominado Distorção de Profundidade Admissível (ADD).

Os métodos propostos são comparados. O método ADD-M é baseado em um agrupamento de valores de profundidade que assumem um único valor de acordo com [Zhang et al., 2016] e tem como objetivo diminuir a quantidade de níveis de intensidade nos mapas de profundidade para que haja um maior ganho de compressão em relação aos mapas de profundidade comprimidos originalmente. Este ganho já pode ser observado pela entropia dos mapas pré-processados que são menores em relação aos originais. O ganho médio desta abordagem é de 29.64% em relação ao ganho percentual da compressão dos mapas de profundidade originais.

A proposta ADD-MVB foi testada com duas medidas estatísticas e têm como objetivo, diferentemente do método ADD-M, não diminuir a quantidade de profundidades nos mapas de profundidade, mas produzir maior homogeneidade por bloco levando em consideração a quebra em blocos e a predição realizada no momento da codificação. A proposta apresentada intitulada ADD-MVB, que utiliza a medida estatística da mediana, consegue atingir um ganho médio de 36% em relação à compressão dos mapas de profundidade originais. O melhor ganho dessa abordagem obteve uma diferença de 13.9% superior em relação às taxas de compressão dos mapas utilizando o método ADD-M. Esta medida apresentou, em geral, resultados melhores quando comparada a medida estatística da média. A diferença do ganho médio para as sequências testadas na abordagem que utiliza mediana foi de 6.38 % em relação ao método ADD-M.

Os métodos propostos tinham também como objetivo alterar os valores do mapa de

profundidade sem que houvesse qualquer alteração na síntese de vistas realizada com estes mapas pré-processados. Portanto, não houve erros de sintetização na imagem virtual, pois a utilização de intervalos ADD assegurou esta propriedade.

Vale ressaltar que os resultados mostram que é possível alcançar maior eficiência de compressão mesmo havendo uma entropia maior no arquivo após o pré-processamento. Isto é, apesar de os mapas de profundidade apresentarem menor entropia quando pré-processados com o método ADD-M, os resultados do pré-processamento com o método ADD-MVB apresentou-se mais eficiente em relação à compressão dos dados, isto se deve ao fato de se explorar as vizinhanças dos píxeis, uma vez que é adotado um codificador de blocos, é possível tirar proveito das redundâncias espaciais que o mesmo utiliza no momento da compressão. Ao tornar o mapa mais homogêneo em relação aos seus blocos de codificação ganha-se mais em eficiência de compressão.

Além dos resultados dos métodos apresentados, elaborou-se uma última abordagem que visa a determinar blocos que contenham regiões com valores semelhantes de profundidade. Esta abordagem, que será objeto de futuro desenvolvimento, obteve melhores níveis de compressões quando comparado aos melhores resultados do método ADD-MVB para as sequências testadas. Isto ocorreu devido à maior homogeneidade adquirida nas regiões englobadas pelos blocos escolhidos manualmente, constatando-se que a eficiência de compressão aumenta ao se escolher blocos que contenham valores de profundidades vizinhos e portanto intervalos ADD próximos.

Como trabalho futuro propõe-se encontrar um parâmetro para a definição do tamanho e da posição do bloco ideal para pré-processar uma região. Tendo-se um parâmetro definido, é possível realizar a busca por blocos de forma automática. Um algoritmo recursivo foi desenvolvido inicialmente para realizar quebras sucessivas no tamanho do blocos, assim começa com $M = 256 \times 256$ e se um determinado parâmetro indicar a necessidade, 4 blocos de tamanho $M = 128 \times 128$ são criados e verificados e assim por diante, podendo quebrar um determinado bloco no tamanho máximo de $M = 8 \times 8$. Desta forma, o próximo passo é determinar um parâmetro capaz de indicar se um bloco já pode ser pré-processado ou se necessita ter seu tamanho quebrado, pois ter blocos que englobem regiões com uma certa variabilidade de valores de profundidade mas que tenham seus intervalos ADD próximos mostrou-se mais eficiente em termos de compressão de dados para os mapas de profundidade. Também, considera-se como trabalho futuro uma investigação na aplicação do método ADD-MVB levando em consideração os conceitos mais específicos (do uso da *quadtree*) do HEVC.

Todos os testes apresentados nesta dissertação foram realizados somente para o primeiro quadro dos vídeos, pois como não nenhum pré-processamento temporal não há motivação para a realização do pré-processamento do vídeo inteiro neste momento. Al-

guns mapas obtiveram maior compressão em razão de sua estrutura e distribuição de profundidades. Mapas que possuíam maior quantidade de fundo homogêneo ou grande quantidade de níveis de profundidade iguais foram os que tiveram melhor desempenho.

Conclui-se, portanto, que há diversos fatores que influenciam em uma codificação eficiente, pois não somente a entropia possui um papel significante, mas também as relações espacias e temporais podem atribuir ganhos no momento da compressão de dados. O método ADD-MVB prova que apesar da entropia ser maior no mapa de profundidade a sua compressão foi mais eficiente, pois a homogeneidade conferiu um impacto maior para a eficácia de compressão.

Referências

[Bradski and Kaehler, 2008] Bradski, G. and Kaehler, A. (2008). *Learning OpenCV: Computer vision with the OpenCV library*. "O'Reilly Media, Inc.". ix, 15, 16, 17

[Diniz et al., 2010] Diniz, P. S., Da Silva, E. A., and Netto, S. L. (2010). *Digital signal processing: system analysis and design*. Cambridge University Press. 27

[Domanski et al., 2009] Domanski, M., Grajek, T., Klimaszewski, K., Kurc, M., Stankiewicz, O., Stankowski, J., and Wegner, K. (2009). Poznan multiview video test sequences and camera parameters. In *ISO/IEC JTC1/SC29/WG11 MPEG 2009/M17050*, Xian, China. 53

[Fehn, 2004] Fehn, C. (2004). Depth-image-based rendering (dibr), compression, and transmission for a new approach on 3d-tv. In *Electronic Imaging 2004*, pages 93–104. International Society for Optics and Photonics. 3, 4, 19, 36

[Forsyth and Ponce, 2002] Forsyth, D. A. and Ponce, J. (2002). *Computer vision: a modern approach*. Prentice Hall Professional Technical Reference. 9

[Gonzalez and Woods, 2010] Gonzalez, R. C. and Woods, R. E. (2010). *Processamento Digital de Imagem*. Pearson, ISBN-10:8576054019, third edition. 10, 11, 27, 28

[Hartley and Zisserman, 2004] Hartley, R. I. and Zisserman, A. (2004). *Multiple View Geometry in Computer Vision*. Cambridge University Press, ISBN: 0521540518, second edition. 8

[Hore and Ziou, 2010] Hore, A. and Ziou, D. (2010). Image quality metrics: Psnr vs. ssim. In *Pattern recognition (icpr), 2010 20th international conference on*, pages 2366–2369. IEEE. 33

[ITU-T, 2014] ITU-T, I. (2014). *High efficiency video coding*. ITU-T Recommendation H.265. 4, 47, 54

[JTC1/SC29/WG11, 2008] JTC1/SC29/WG11, I. (2008). *Contribution for 3D Video Test-Material of Outdoor Scene*. x, 53, 57

[Júlio et al., 2016] Júlio, G., Dórea, C., and Machiavello, B. (2016). Otimização de distorção de profundidade admissível para codificação de mapas de profundidade. *XXXIV Simpósio Brasileiro de Telecomunicações*. 47

[Kang and Ho, 2012] Kang, M.-K. and Ho, Y.-S. (2012). Depth video coding using adaptive geometry based intra prediction for 3-d video systems. *IEEE Transactions on Multimedia*, 14(1):121–128. 36

[Kauff et al., 2007] Kauff, P., Atzpadin, N., Fehn, C., Müller, M., Schreer, O., Smolic, A., and Tanger, R. (2007). Depth map creation and image-based rendering for advanced 3dtv services providing interoperability and scalability. *Signal Processing: Image Communication*, 22(2):217–234. 2

[Kim et al., 2012] Kim, I.-K., Min, J., Lee, T., Han, W.-J., and Park, J. (2012). Block partitioning structure in the hevc standard. *IEEE transactions on circuits and systems for video technology*, 22(12):1697–1706. ix, 23, 24, 25

[Lee and Ho, 2009] Lee, C. and Ho, Y.-S. (2009). View synthesis using depth map for 3d video. In *Proceedings: APSIPA ASC 2009: Asia-Pacific Signal and Information Processing Association, 2009 Annual Summit and Conference*, pages 350–357. Asia-Pacific Signal and Information Processing Association, 2009 Annual Summit and Conference, International Organizing Committee. 1, 2, 4, 13, 29, 35

[Macchiavello et al., 2014] Macchiavello, B., Dorea, C., Hung, E. M., Cheung, G., and Bajic, I. (2014). Low-saliency prior for disocclusion hole filling in dibr-synthesized images. In *Acoustics, Speech and Signal Processing (ICASSP), 2014 IEEE International Conference on*, pages 579–583. IEEE. 14

[Mao et al., 2013] Mao, Y., Cheung, G., Ortega, A., and Ji, Y. (2013). Expansion hole filling in depth-image-based rendering using graph-based interpolation. In *ICASSP*, pages 1859–1863. 14

[Merkle et al., 2007] Merkle, P., Smolic, A., Müller, K., and Wiegand, T. (2007). Multi-view video plus depth representation and coding. In *Image Processing, 2007. ICIP 2007. IEEE International Conference on*, volume 1, pages I–201. IEEE. 1, 2, 29

[Richardson, 2011] Richardson, I. E. (2011). *The H. 264 advanced video compression standard*. John Wiley & Sons. 2, 11, 12, 21, 22, 25, 26, 27, 28, 32

[Sayood, 2012] Sayood, K. (2012). *Introduction to data compression*. Newnes. 22, 27, 32

[Sociedade Européia de Tecnologia da Informação, 2004] Sociedade Européia de Tecnologia da Informação, I. (2002-2004). Sistema tecnologia de televisão 3d avançado. http://cordis.europa.eu/project/rcn/62861$_e$*n.html. Accessadoem*25/11/2016. 3

[Sullivan and Wiegand, 2005] Sullivan, G. J. and Wiegand, T. (2005). Video compression-from concepts to the h. 264/avc standard. *Proceedings of the IEEE*, 93(1):18–31. 21, 28

[Tanimoto et al., 2008] Tanimoto, M., Fujii, M., and Fukushiuma, K. (2008). 1D parallel test sequences for MPEG-FTV. In *ISO/IEC JTC1/SC29/WG11 MPEG 2008/M15378*, Archamps, France. ix, x, 18, 19, 42, 53, 55, 59, 65

[Tanimoto et al., 2005] Tanimoto, M., Fujii, M., Senoh, T., Aoki, T., and Sugihara, Y. (2005). Test sequences with different camera arrangements for call for proposals on multiview video coding. In *ISO/IEC JTC1/SC29/WG11 MPEG 2005/M12338*, Poznan, Poland. x, 53, 57, 65

[Tech et al., 2016a] Tech, G., Chen, Y., Müller, K., Ohm, J.-R., Vetro, A., and Wang, Y.-K. (2016a). Overview of the multiview and 3d extensions of high efficiency video coding. *IEEE Transactions on Circuits and Systems for Video Technology*, 26(1):35–49. 1, 30, 31

[Tech et al., 2016b] Tech, G., Chen, Y., Müller, K., Ohm, J.-R., Vetro, A., and Wang, Y.-K. (2016b). Overview of the multiview and 3d extensions of high efficiency video coding. *Circuits and Systems for Video Technology, IEEE Transactions on*, 26(1):35–49. 29

[Tian et al., 2009] Tian, D., Lai, P.-L., Lopez, P., and Gomila, C. (2009). View synthesis techniques for 3d video. In *SPIE Optical Engineering+ Applications*, pages 74430T–74430T. International Society for Optics and Photonics. ix, 7, 8, 9, 17, 18, 19, 20, 35, 42

[Uhrina et al., 2014] Uhrina, M., Frnda, J., Sevcik, L., and Vaculik, M. (2014). Impact of h. 264/avc and h. 265/hevc compression standards on the video quality for 4k resolution. *Advances in Electrical and Electronic Engineering*, 12(4):368. 29

[Vetro et al., 2011] Vetro, A., Wiegand, T., and Sullivan, G. J. (2011). Overview of the stereo and multiview video coding extensions of the h. 264/mpeg-4 avc standard. *Proceedings of the IEEE*, 99(4):626–642. 36

[Wang and Bovik, 2009] Wang, Z. and Bovik, A. C. (2009). Mean squared error: Love it or leave it? a new look at signal fidelity measures. *IEEE signal processing magazine*, 26(1):98–117. 33

[Wiegand et al., 2003] Wiegand, T., Sullivan, G., Bjontegaard, G., and Luthra, A. (2003). Overview of the H.264/AVC video coding standard. *IEEE Transactions on Circuits and Systems for Video Technology*, 13(7). 28

[Zhang et al., 2014] Zhang, Y., Kwong, S., Hu, S., and Kuo, C.-C. J. (2014). Efficient multiview depth coding optimization based on allowable depth distortion in view synthesis. *Image Processing, IEEE Transactions on*, 23(11):4879–4892. 3, 36, 37, 38, 39, 40, 41

[Zhang et al., 2013] Zhang, Y., Kwong, S., Xu, L., Hu, S., Jiang, G., and Kuo, C.-C. J. (2013). Regional bit allocation and rate distortion optimization for multiview depth video coding with view synthesis distortion model. *IEEE Transactions on Image Processing*, 22(9):3497–3512. 36

[Zhang et al., 2015] Zhang, Y., Pan, Z., Zhou, Y., and Zhu, L. (2015). Allowable depth distortion based fast mode decision and reference frame selection for 3d depth coding. *Multimedia Tools and Applications*, pages 1–20. x, 3, 36, 38, 40, 45, 51

[Zhang et al., 2016] Zhang, Y., Zhu, L., Liu, X., and Jiang, G. (2016). Allowable depth distortion based depth filtering for 3d high efficiency video coding. In *Circuits and Systems (ISCAS), 2016 IEEE International Symposium on*, pages 2559–2562. IEEE. 3, 5, 36, 42, 43, 66

[Zhao et al., 2011] Zhao, Y., Zhu, C., Chen, Z., and Yu, L. (2011). Depth no-synthesis-error model for view synthesis in 3-d video. *Image Processing, IEEE Transactions on*, 20(8):2221–2228. 3, 36, 37

[Zitnick et al., 2004] Zitnick, C. L., Kang, S. B., Uyttendaele, M., Winder, S., and Szeliski, R. (2004). High-quality video view interpolation using a layered representation. In *ACM Transactions on Graphics (TOG)*, volume 23, pages 600–608. ACM. 18

Apêndice A

Arquivo de Configuração do Codificador HEVC

======== File I/O =====================
BitstreamFile : str.bin
ReconFile : rec.yuv

======== Profile ======================
Profile : main

======== Unit definition ==================
MaxCUWidth : 64 Maximum coding unit width in pixel
MaxCUHeight : 64 Maximum coding unit height in pixel
MaxPartitionDepth : 4 Maximum coding unit depth
QuadtreeTULog2MaxSize : 5 Log2 of maximum transform size for quadtree-based TU coding (2...6)
QuadtreeTULog2MinSize : 2 Log2 of minimum transform size for quadtree-based TU coding (2...6)
QuadtreeTUMaxDepthInter : 3
QuadtreeTUMaxDepthIntra : 3

======== Coding Structure ================
IntraPeriod : 1 Period of I-Frame (-1 = only first)
DecodingRefreshType : 0 Random Accesss 0:none, 1:CRA, 2:IDR, 3:Recovery Point SEI
GOPSize : 4 GOP Size (number of B slice = GOPSize-1)

Type POC QPoffset QPfactor tcOffsetDiv2 betaOffsetDiv2 temporal_id ref_pics_active ref_pics reference pictures predict deltaRPS ref_idcs reference idcs

Frame1: P 1 3 0.4624 0 0 0 4 4 -1 -5 -9 -13 0
Frame2: P 2 2 0.4624 0 0 0 4 4 -1 -2 -6 -10 1 -1 5 1 1 1 0 1
Frame3: P 3 3 0.4624 0 0 0 4 4 -1 -3 -7 -11 1 -1 5 0 1 1 1 1
Frame4: P 4 1 0.578 0 0 0 4 4 -1 -4 -8 -12 1 -1 5 0 1 1 1 1

========== Motion Search ==========
FastSearch : 1 0:Full search 1:TZ search
SearchRange : 64 (0: Search range is a Full frame)
BipredSearchRange : 4 Search range for bi-prediction refinement
HadamardME : 1 Use of hadamard measure for fractional ME
FEN : 1 Fast encoder decision
FDM : 1 Fast Decision for Merge RD cost

======== Quantization ==========
QP : 0 Quantization parameter(0-51)
MaxDeltaQP : 0 CU-based multi-QP optimization
MaxCuDQPDepth : 0 Max depth of a minimum CuDQP for sub-LCU-level delta QP
DeltaQpRD : 0 Slice-based multi-QP optimization
RDOQ : 1 RDOQ
RDOQTS : 1 RDOQ for transform skip
TransformSkip : 1 Transform skipping (0: OFF, 1: ON)
TransformSkipFast : 1 Fast Transform skipping (0: OFF, 1: ON)

========== Deblock Filter ==========
LoopFilterOffsetInPPS : 1 Dbl params: 0=varying params in SliceHeader, param = base_param + GOP_offset_param; 1 (default) =constant params in PPS, param = base_param)
LoopFilterDisable : 0 Disable deblocking filter (0=Filter, 1=No Filter)
LoopFilterBetaOffset_div2 : 0 base_param: -6 6
LoopFilterTcOffset_div2 : 0 base_param: -6 6
DeblockingFilterMetric : 0 blockiness metric (automatically configures deblocking parameters in bitstream). Applies slice-level loop filter offsets (LoopFilterOffsetInPPS and LoopFilterDisable must be 0)

========== Misc. ==========

InternalBitDepth : 8 codec operating bit-depth

========== Coding Tools ==========
SAO : 1 Sample adaptive offset (0: OFF, 1: ON)
AMP : 1 Asymmetric motion partitions (0: OFF, 1: ON)
SAOLcuBoundary : 0 SAOLcuBoundary using non-deblocked pixels (0: OFF, 1: ON)

========== Slices ==========
SliceMode : 0 0: Disable all slice options.
1: Enforce maximum number of LCU in an slice,
2: Enforce maximum number of bytes in an 'slice'
3: Enforce maximum number of tiles in a slice
SliceArgument : 1500 Argument for 'SliceMode'.
If SliceMode==1 it represents max. SliceGranularity-sized blocks per slice.
If SliceMode==2 it represents max. bytes per slice.
If SliceMode==3 it represents max. tiles per slice.

LFCrossSliceBoundaryFlag : 1
In-loop filtering, including ALF and DB, is across or not across slice boundary.
0:not across, 1: across

========== PCM ==========
PCMEnabledFlag : 0 0: No PCM mode
PCMLog2MaxSize : 5 Log2 of maximum PCM block size.
PCMLog2MinSize : 3 Log2 of minimum PCM block size.
PCMInputBitDepthFlag : 1 0: PCM bit-depth is internal bit-depth. 1: PCM bit-depth is input bit-depth.
PCMFilterDisableFlag : 0 0: Enable loop filtering on I_PCM samples. 1: Disable loop filtering on I_PCM samples.

========== Tiles ==========
TileUniformSpacing : 0
0: the column boundaries are indicated by TileColumnWidth array, the row boundaries are indicated by TileRowHeight array
1: the column and row boundaries are distributed uniformly
NumTileColumnsMinus1 : 0 Number of tile columns in a picture minus 1

TileColumnWidthArray : 2 3 Array containing tile column width values in units of CTU (from left to right in picture)
NumTileRowsMinus1 : 0 Number of tile rows in a picture minus 1
TileRowHeightArray : 2 Array containing tile row height values in units of CTU (from top to bottom in picture)
LFCrossTileBoundaryFlag : 1
In-loop filtering is across or not across tile boundary.
0:not across, 1: across
========== WaveFront ==========
WaveFrontSynchro : 0
0: No WaveFront synchronisation (WaveFrontSubstreams must be 1 in this case).
>0: WaveFront synchronises with the LCU above and to the right by this many LCUs.
========== Quantization Matrix ==========
ScalingList : 0 ScalingList 0 : off, 1 : default, 2 : file read
ScalingListFile : scaling_list.txt Scaling List file name. If file is not exist, use Default Matrix.
========== Lossless ==========
TransquantBypassEnableFlag : 1 Value of PPS flag.
CUTransquantBypassFlagForce: 1 Force transquant bypass mode, when transquant_bypass_enable is enabled
========== Rate Control ========== RateControl : 0 Rate control: enable rate control
TargetBitrate : 1000000 Rate control: target bitrate, in bps
KeepHierarchicalBit : 2 Rate control: 0: equal bit allocation; 1: fixed ratio bit allocation; 2: adaptive ratio bit allocation
LCULevelRateControl : 1 Rate control: 1: LCU level RC; 0: picture level RC
RCLCUSeparateModel : 1 Rate control: use LCU level separate R-lambda model
InitialQP : 0 Rate control: initial QP
RCForceIntraQP : 0 Rate control: force intra QP to be equal to initial QP