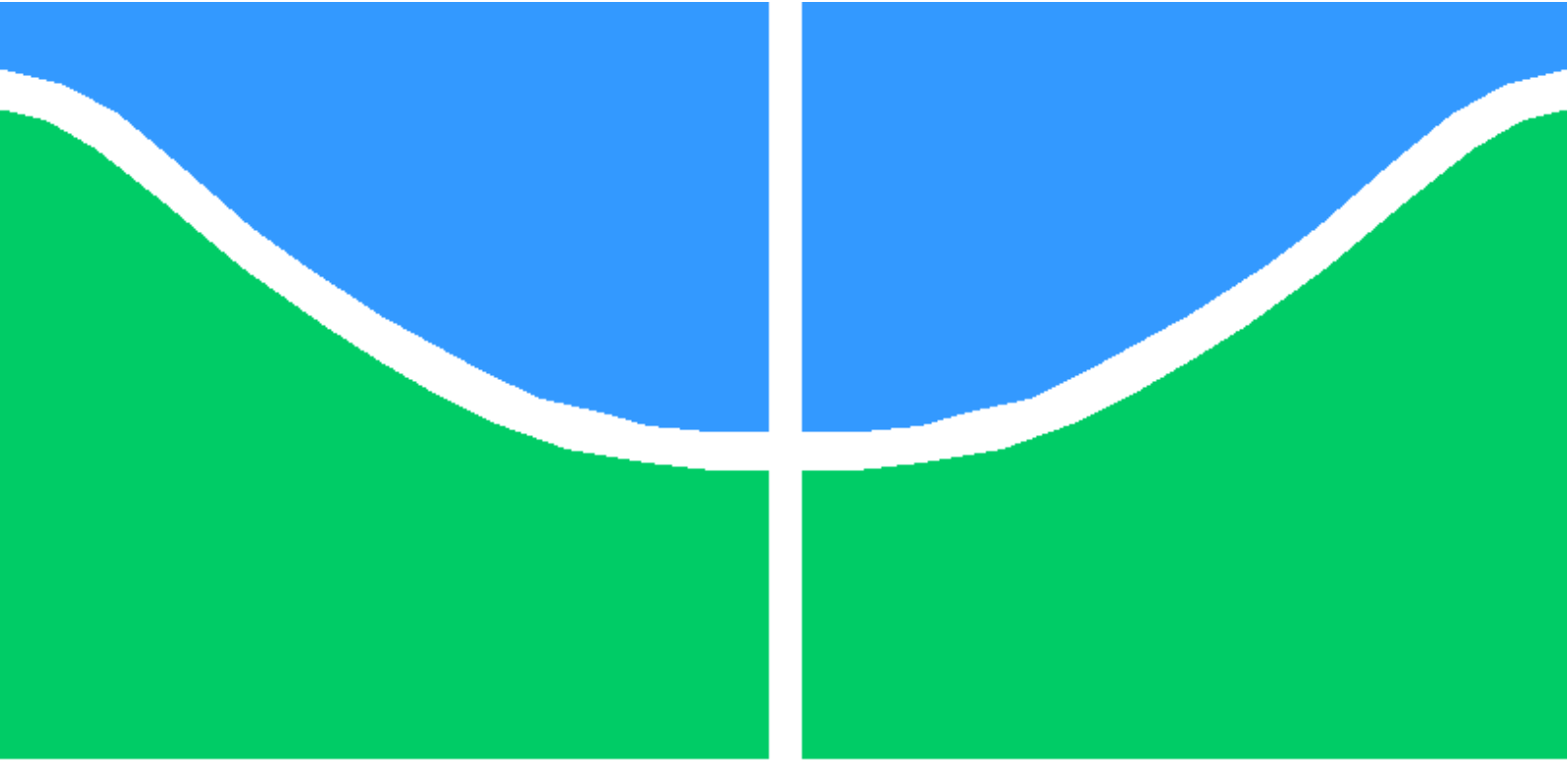


DISSERTAÇÃO DE MESTRADO

SÍNTESE E CODIFICAÇÃO DE VISTAS VIRTUAIS PARA SISTEMAS DE PONTO DE VISTA LIVRE

Thacio Garcia Scandaroli

Brasília, Maio de 2012

UNIVERSIDADE DE BRASÍLIA

FACULDADE DE TECNOLOGIA

UNIVERSIDADE DE BRASILIA
Faculdade de Tecnologia

DISSERTAÇÃO DE MESTRADO

SÍNTESE E CODIFICAÇÃO DE VISTAS VIRTUAIS PARA SISTEMAS DE PONTO DE VISTA LIVRE

Thacio Garcia Scandaroli

Relatório submetido ao Departamento de Engenharia Elétrica como requisito parcial para obtenção do grau de Mestre em Engenharia Elétrica

Banca Examinadora

Prof. Ricardo Lopes de Queiroz, CIC/UnB _____
Orientador

Prof. Camilo C. Dôrea, CIC/UnB _____
Examinador externo

Prof. João Luiz A. de Carvalho, ENE/UnB _____
Examinador interno

Dedicatória

À minha família, que sempre me apoiou.

Thacio Garcia Scandaroli

Agradecimentos

Primeiramente, agradeço aos meus pais por todo o suporte e ao meu irmão, que embora distante, esteve sempre presente neste período.
Agradeço também ao meu orientador Prof. Ricardo Queiroz por mais uma grande oportunidade e por mais um projeto concluído. Da mesma forma, sou grato aos companheiros do nosso grupo, que estiveram presente nesta jornada e o apoio foi essencial para a conclusão deste trabalho.
E por último, mas igualmente importantes, aos meu amigos que de uma forma ou de outra ajudaram ou me apoiaram nesta jornada.
Obrigado,

Thacio Garcia Scandaroli

RESUMO

Nos últimos anos, houve uma crescente tendência de desenvolvimento de novas tecnologias que possibilitaram novas formas de interação entre usuário e conteúdo. Sistemas de ponto de vista livre, que são sistemas que possibilitam ao usuário determinar qual ponto de vista da cena será exibido, se tornam cada vez mais próximos de serem concretizados. Este trabalho tem como objetivo investigar este tipo de sistema. Primeiro, é criado um sintetizador de vistas que, com a informação de diferentes câmeras de uma cena, gera uma nova imagem referente a um novo ponto de vista o qual não foi capturado por nenhuma câmera. Desta forma, possibilita a criação de sistemas de ponto de vista livre. Alguns métodos para suavização de contornos e interpolação de pixels para a melhora da qualidade da imagem gerada pela síntese de vista foram propostos. Dependendo de qual lado a síntese de vista for realizada (codificador ou decodificador), diferentes dados são transmitidos no sistema. Se houver um canal de retorno, pode ser melhor sintetizar os novos pontos de vista no lado do codificador, sendo este cenário adequado caso o decodificador possua baixa complexidade computacional ou o canal de transmissão tenha restrição de banda. Sem o canal de retorno, a síntese deve ser realizada no decodificador e todas as vistas capturadas devem ser transmitidas. É investigado a arquitetura do sistema para cada alternativa e seu custo-benefício.

ABSTRACT

In recent years, a trend to develop new technologies to enable human-content interaction arose. Free viewpoint television (FTV) enables the user to interactively control the viewpoint of the scene being displayed and is now becoming a viable technology. This work investigates this type of system. First, a view synthesizer is created that generates an image that corresponds to a viewpoint of the scene which was not captured by any camera, using the video captured by the existing cameras. With that is possible to create a FTV system. Some methods were proposed for edge smoothing and pixel interpolation to improve the overall image quality in view synthesis. Depending on which side of the system view synthesis is carried (encoder or decoder) different data should be transmitted to the receiver. Where a feedback channel is available it is perhaps better to synthesize new views at the encoder size and this is suited to low-complexity decoders and to channels with reduced bandwidth. Without a feedback channel, views are synthesized at the decoder and all the captured views are sent to the decoder side. We investigated the system architecture for each alternative and investigated their cost-effectiveness.

SUMÁRIO

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE SÍMBOLOS

Símbolos Latinos

Y	Intensidade da componente de Luminância
C	Intensidade da componente de Crominância
R	Intensidade da componente de Vermelho
G	Intensidade da componente de Verde
B	Intensidade da componente de Azul
QP	Parâmetro de quantização
Z	Profundidade de um ponto
$\mathbf{X}$	Posição de um ponto no espaço tridimencional
$\mathbf{x}$	Posição de um ponto no plano da imagem
$\mathbf{A}$	Matriz de parâmetros intrínsecos de uma câmera
$\mathbf{[R\|T]}$	Matriz de parâmetros exertrínsecos de uma câmera
$\mathbf{P}$	Matriz de projeção de uma câmera
I	Intesidade de um pixel
(u, v)	Posição de um pixel em uma imagem

Símbolos Gregos

λ	Parâmetro de interpolação
σ^2	Variância
μ	Média

Subscritos

r	vermelho ou referência
g	verde
b	azul
e	esquerda
d	direita
v	virtual
min	mínima
max	máxima
v	virtual

Siglas

RGB	Sistema de cores
YUV	Sistema de cores
YCbCr	Sistema de cores
H.264	Padrão de vídeo
MPEG2	Padrão de vídeo
FTV	*Free Viewpoint Television* - Televisão com ponto de vista livre
ITU-T	*Telecommunication Standardization Sector of the International Telecommunications Union*
IEC	
MPEG	
DCT	Transformada discreta de cosseno
PSNR	Relação Sinal-Ruído de pico
SAD	Soma das diferenças absolutas
MSE	Erro Médio Quadrático
AVC	Advanced Video Coding
MVC	Multiview Video Coding
MVD	Multiview Plus Depth

Capítulo 1

Introdução

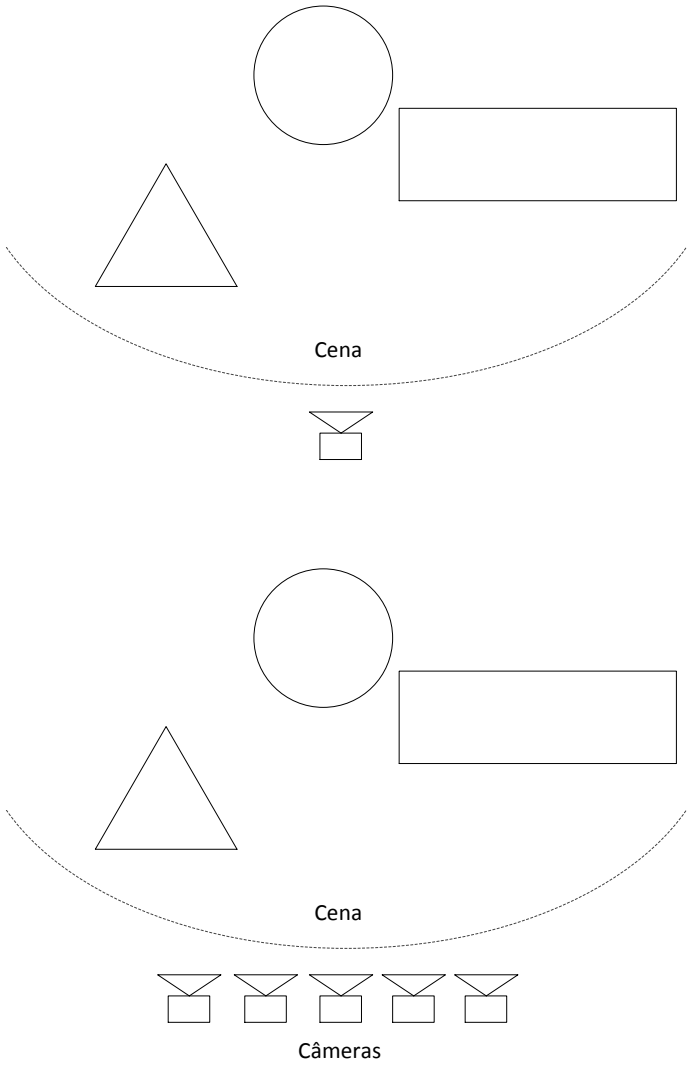
1.1 Contextualização

Com o avanço das tecnologias digitais, a compressão de vídeo se tornou uma área chave. Nela, as pesquisa têm seu maior foco em melhorar a taxa de compressão de um vídeo mantendo a sua qualidade visual.

Compressão de vídeo pode ser sem perdas ou com perdas. A compressão sem perdas de um vídeo digital consiste na representação total de suas informação utilizando um menor número de bits em relação ao vídeo original. A compressão com perdas refere-se a representação das informações de um vídeo utilizando menos bits em relação ao vídeo original, mas acarretando em uma perda de qualidade visual devido a perda de certas informações. Este é o método mais utilizado atualmente e há diversos esforços em pesquisa ao redor do mundo no desenvolvimento dele. O padrão mais avançado atualmente de codificação de vídeo é o H.264 [2]. O foco principal no desenvolvimento de tecnologias de compressão com perdas é a diminuição do número de bits que representam a informação de vídeos, mas mantendo uma qualidade visual elevada.

A utilização de vídeos digitais trouxe novos paradigmas e novos desafios para a tecnologia atual. Há uma demanda crescente por vídeos de melhor qualidade e que proporcionem novas interações entre conteúdo e usuário. O sistema de vista única é o mais comum na captura de vídeos digitais, onde apenas uma câmera captura uma determinada cena. Há também o sistema multi-vistas no qual é utilizado mais de uma câmera na captura de uma determinada cena, com isto, é possível a exibição do vídeo no formato 3D em que cada olho do usuário assiste à imagem de uma câmera diferente, tendo assim a percepção de profundidade. Sistemas multi-vistas também proporcionam novas possibilidades, como o sistema de ponto de vista livre ou FTV (*free viewpoint television*) [23] no qual o usuário escolhe de qual ponto de vista irá assistir ao vídeo. A Figura 1.1 ilustra a forma de captura de uma cena em sistemas de vista única e de sistemas multi-vistas. Com a popularização das televisões 3D e com o aumento do poder de processamento dos computadores, sistemas multi-vistas estão se tornando cada vez mais populares. Então há um grande interesse no desenvolvimento de métodos de compressão para sistemas multi-vistas.

Para um sistema FTV ilustrado na Figura 1.2, o plano representa os pontos de vista os quais o



(b)

Figura 1.1: Tipos de captura em sistemas de vídeo. (a) Sistema de vista única: o vídeo de uma determinada cena é capturado por uma única câmera. (b) Sistema Multi-Vista: São capturados vídeos de diferentes pontos de vista de uma mesma cena por câmeras em diferentes posições.

usuário pode assistir à cena gravada. Para a captura de tais pontos de vista, há duas possibilidades: vários pontos de vista de uma cena são capturados por uma única câmera, desde que a cena seja estática, ou a cena é capturada por diversas câmeras em diferentes posições, sendo possível sintetizar as vistas intermediárias entre tais câmeras. Este último caso pode também capturar cenas dinâmicas, as quais há movimento. Então, utilizando um determinado número de câmeras para gravar uma cena, é possível, com a síntese de novas vistas, criar vídeos da cena de diferentes ângulos que não foram gravados, possibilitando a implementação de um sistema FTV.

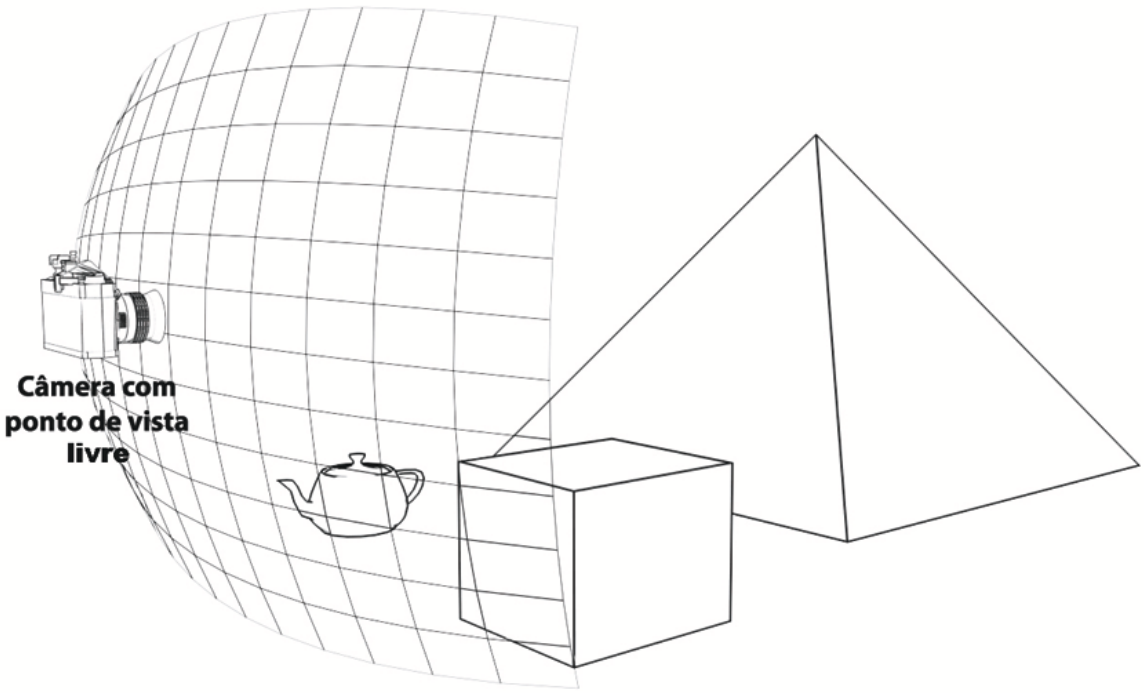


Figura 1.2: Sistema de ponto de vista livre. A câmera captura diversos pontos de vista de uma mesma cena. O plano ilustrado representa os diferentes pontos de vista que o sistema captura.

1.2 Definição do problema

Como pode ser visto na Figura 1.1(b), as várias câmeras do sistema capturam uma mesma cena de ângulos diferentes, havendo uma grande redundância nos dados capturados. Com o aumento do número de câmeras, maior é o tamanho da informação a ser codificada e comprimida, então é necessário o desenvolvimento de métodos mais eficientes para compressão de vídeos em sistemas multi-vistas. Cada vídeo desse sistema pode ser comprimido individualmente por um padrão de vídeo como o H.264/AVC. Neste caso definimos essa compressão e transmissão como *simulcast*, mas, devido a redundância de dados entre as câmeras, é possível utilizar a correlação entre elas para se obter um aumento de qualidade e compressão.

O mesmo pode ser feito em sistemas FTV devido ao fato das imagens sintetizadas possuírem redundâncias das imagens reais. Em um sistema FTV, se for de interesse de que a síntese de imagem seja feita no codificador, a imagem sintetizada deverá ser comprimida no próprio e então transmitida para o usuário. Em tal sistema, os maiores desafios são a síntese de vídeo, que precisa gerar uma imagem de alta qualidade e com baixa incidência de erros, e a compressão eficiente dos dados do sistema, que deve manter uma alta qualidade visual para as imagens descomprimidas, mas visando também a redução da taxa de bits do conteúdo para a transmissão.

1.3 Objetivos do projeto

Este trabalho tem como objetivo explorar possibilidades de sistemas FTV, tanto na geração de um sintetizador de vistas utilizando câmeras reais e mapas de profundidade como na investigação de métodos de compressão de sistemas multi-vista com vista sintetizada.

É de interesse criar um sintetizador de vistas que gere imagens de melhor qualidade reduzindo erros e artefatos devido a erros nos mapas de profundidade. Na compressão destas imagens e vídeos, têm-se como objetivo utilizar informações da síntese para melhorar a compressão das vistas sintetizadas em sistemas multi-vistas.

Neste trabalho, os principais pontos abordados são:

- Sintetização de vistas - Geração de vistas sintetizadas de alta qualidade, com correção de erros e eliminação de artefatos na imagem final.
- Compressão do conteúdo - Determinar as melhores formas de compressão para vídeos em sistemas FTV que os otimize.
- Transmissão dos dados - Identificar o melhor modelo para síntese de vistas e para a compressão do conteúdo em sistemas FTV.

1.4 Apresentação do manuscrito

No Capítulo 2 é feita uma revisão bibliográfica sobre codificação de vídeo e é descrito um modelo de câmera e alguns métodos de processamento de imagem utilizados no desenvolvimento deste trabalho. O Capítulo 3 apresenta sistemas multi-vistas e de ponto de vista livre, assim como os métodos de compressão para eles. A síntese de vista, sua teoria, implementação e resultados são discutidos no Capítulo 4. Diferentes arquiteturas e performances de sistemas FTV são desenvolvidos e testados no Capítulo 5, assim como métodos de compressão de vídeos em sistemas multi-vistas com síntese. No Capítulo 6, são apresentadas as conclusões deste trabalho.

Capítulo 2

Conceitos de Processamento e Codificação de Vídeo

2.1 Codificação de vídeo

2.1.1 Introdução

Codificação de vídeo é o processo de codificação de informações visuais de um vídeo [20]. Na codificação de vídeos digitais ocorre a compressão e descompressão deles. A compressão têm como objetivo a redução da quantidade de bits necessária para a representação de vídeos digitais e esta compressão pode ser com ou sem perdas. Na compressão sem perdas, após a descompressão, o sinal reconstruído é idêntico ao original enquanto que na compressão com perdas, o sinal reconstruído é degradado.

Padrões de codificação de vídeos são necessários para padronizar técnicas de codificação. Um par de codificador, que codifica e comprimi as informações originais, e decodificador, que decodifica a informações e reconstrói o vídeo, é chamado de CODEC. [20]

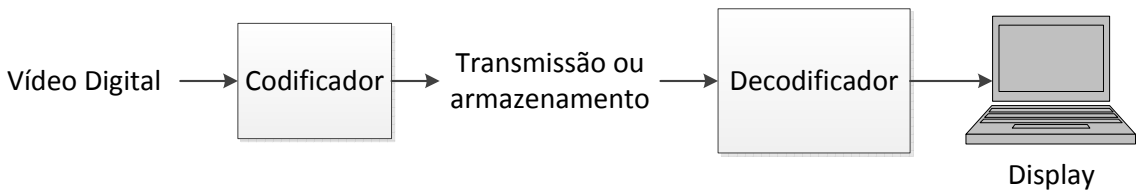


Figura 2.1: Representação de um sistema de vídeo, em que o vídeo digital é codificado, transmitido, decodificado e em seguida exibido ao usuário.

O H.264/AVC [2] é um padrão de vídeo desenvolvido conjuntamente pelo grupo *ITU-T Video Coding Experts Group (VCEG)* e pelo *International Organization for Standardization*(ISO)/*International Electrotechnical Commission*(IEC) *Moving Picture Experts Group* (MPEG). Em comparação ao padrão de vídeo MPEG-2/H.262, antecessor ao H.264/AVC e desenvolvido pelos mesmos grupos, o H.264/AVC é conhecido por antingir a mesma taxa de qualidade de um vídeo MPEG-2/H.262

utilizando apenas metade da taxa de bits.

Um codificador de vídeo tipicamente possui três etapas básicas: compressão utilizando modelo temporal, modelo espacial e codificador de entropia. [20] O modelo temporal, explicado na Seção 2.1.4, tenta reduzir redundâncias temporais do vídeo por meio da correlação entre quadros temporalmente vizinhos. Em seguida, o modelo espacial, explicado na Seção 2.1.5, é utilizado, tentando reduzir redundâncias entre pixels vizinhos de um mesmo quadro por meio da utilização de transformadas matemáticas. Os coeficientes resultantes das transformadas são quantizados, acarretando perdas, mas reduzindo significativamente a taxa de bits final. Por último, todas as informações resultantes do processo de codificação, como cabeçalhos e informações resultantes dos modelos anteriores, são comprimidas por um codificador de entropia (Seção 2.1.8), que reduz redundâncias estatísticas entre os bits dessas informações, gerando um arquivo de vídeo compactado.

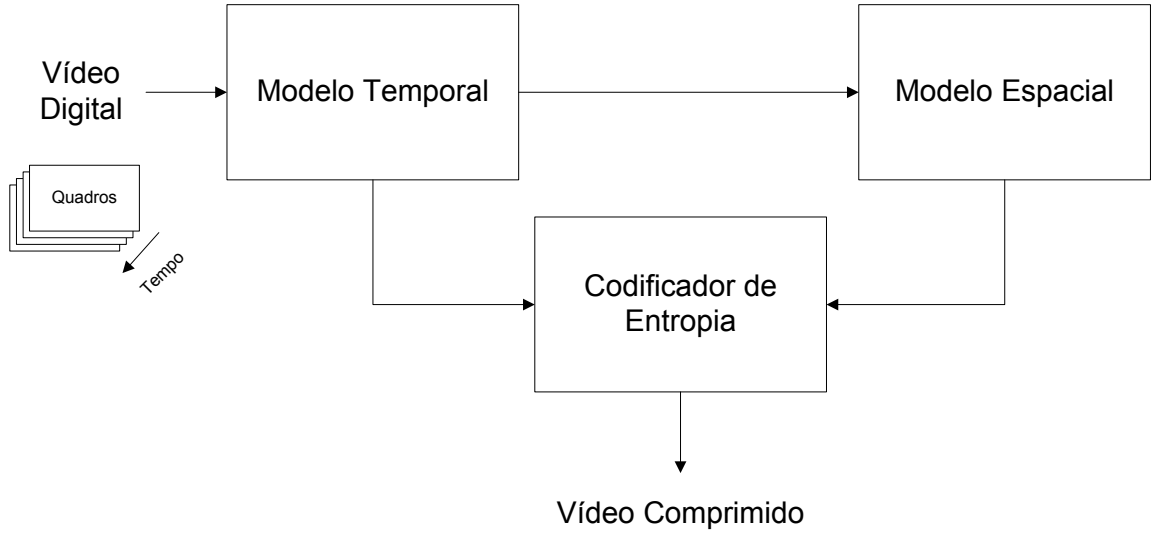


Figura 2.2: Estrutura de um codificador de vídeo.

O padrão de vídeo H.264/AVC utiliza a partição de imagem em macroblocos. [20] Um macrobloco é um bloco que agrupa pixels vizinhos em quadrados ou retângulos. No H.264/AVC, o macrobloco de maior tamanho compreende os pixels dentro de um quadrado de tamanho 16 pixels, sendo ilustrado na figura 2.3(b). Assim, esse macrobloco possui tamanho horizontal de 16 pixels e vertical de 16 pixels, tendo então tamanho 16x16 pixels. Ele pode ser dividido horizontalmente e verticalmente, resultando em blocos menores. Os tamanhos possíveis de macroblocos do padrão H.264/AVC são 16x16, 16x8, 8x16, 8x8, 8x4, 4x8 e 4x4 pixels. Deste modo, a codificação de vídeo utilizando o padrão H.264/AVC é orientada a blocos, sendo a predição intra-quadros e inter-quadros também baseada em blocos, e isto possibilita a codificação por transformada. A divisão de uma imagem em macroblocos assim como as diferentes partições de um macrobloco são ilustradas na Figura 2.3.

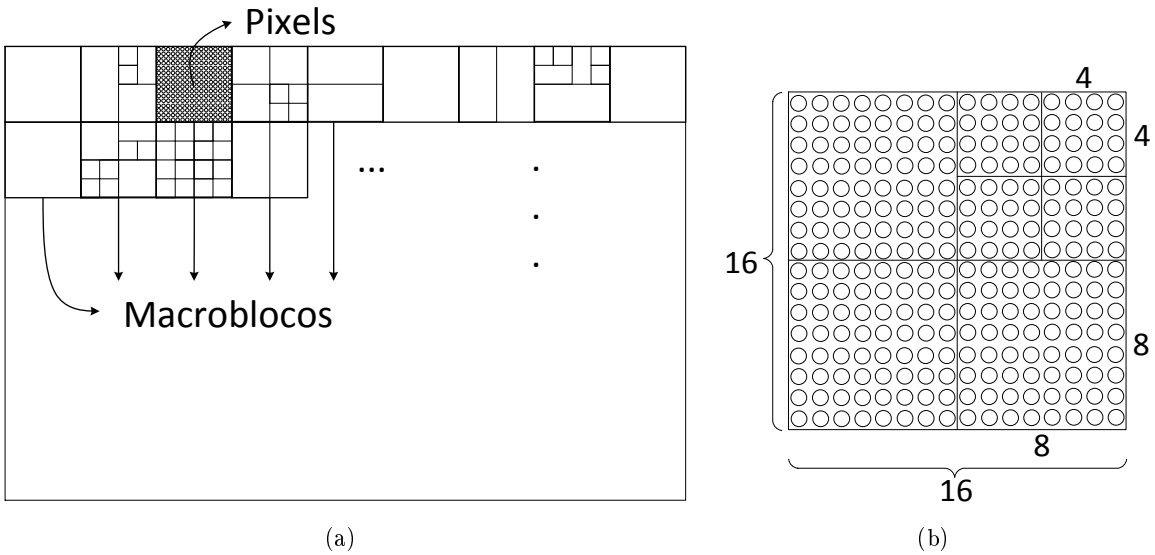


Figura 2.3: Partição de macroblocos. (a) Divisão de uma imagem em macroblocos. (b) Divisão de um macrobloco, o maior tamanho possível é de 16x16 pixels, e este pode ser divido em blocos de tamanho 16x8, 8x8, 8x4 e 4x4.

2.1.2 Espaço de cores

Para a representação de imagens e vídeos digitais é necessário um mecanismo que represente as informações de cores. Para imagens preto e branco, apenas é necessário obter o valor de luminância de cada pixel, que é a medida de densidade da intensidade de luz e varia em níveis de cinza, da cor preta até a branca. Para imagens coloridas, o espaço de cor mais utilizado é o RGB. É comum utilizar 8 bits para cada componente R, G e B, podendo assim assumir 256 diferentes valores que representam a intensidade de cada componente. O espectro visível da luz é aproximadamente separado em três componentes básicas: vermelho, verde e azul. As cores são formadas pela combinação linear dessas três componentes com diferentes intensidades. Assim, utiliza-se filtros na faixa de frequência dessas componentes na captura de vídeo, determinando a intensidade de cada componente separadamente e definindo cada pixel da imagem no sistema RGB.

O espaço de cores YCbCr e suas variações, como YUV, são de comum utilização na representação de imagens com cores [20], no qual Y é a luminância e Cb e Cr são as crôminancias, sendo estas duas as diferença entre a componente azul e luminância, e vermelho e luminância, respectivamente. Este espaço de cores é mais utilizado para a compressão devido ao fato de que o sistema visual humano é mais sensível à luminância do que à crominância, sendo possível sub-amostrar as componentes de crominância sem grandes danos a percepção de qualidade do usuário. No sistema RGB, todas as componentes possuem igual importância, sendo impossível a sub-amostragem sem grandes danos a percepção de qualidade da imagem.

A luminância (Y) é calculada como sendo uma média ponderada das componentes RGB, da seguinte forma:

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

Onde k_r, k_g e k_b são pesos multiplicadores das componentes R, G e B, respectivamente.

Cada crôminancia é a diferença entre vermelho, verde ou azul e a luminância:

$$\begin{aligned} Cb &= B - Y \\ Cr &= R - Y \\ Cg &= G - Y \end{aligned} \tag{2.2}$$

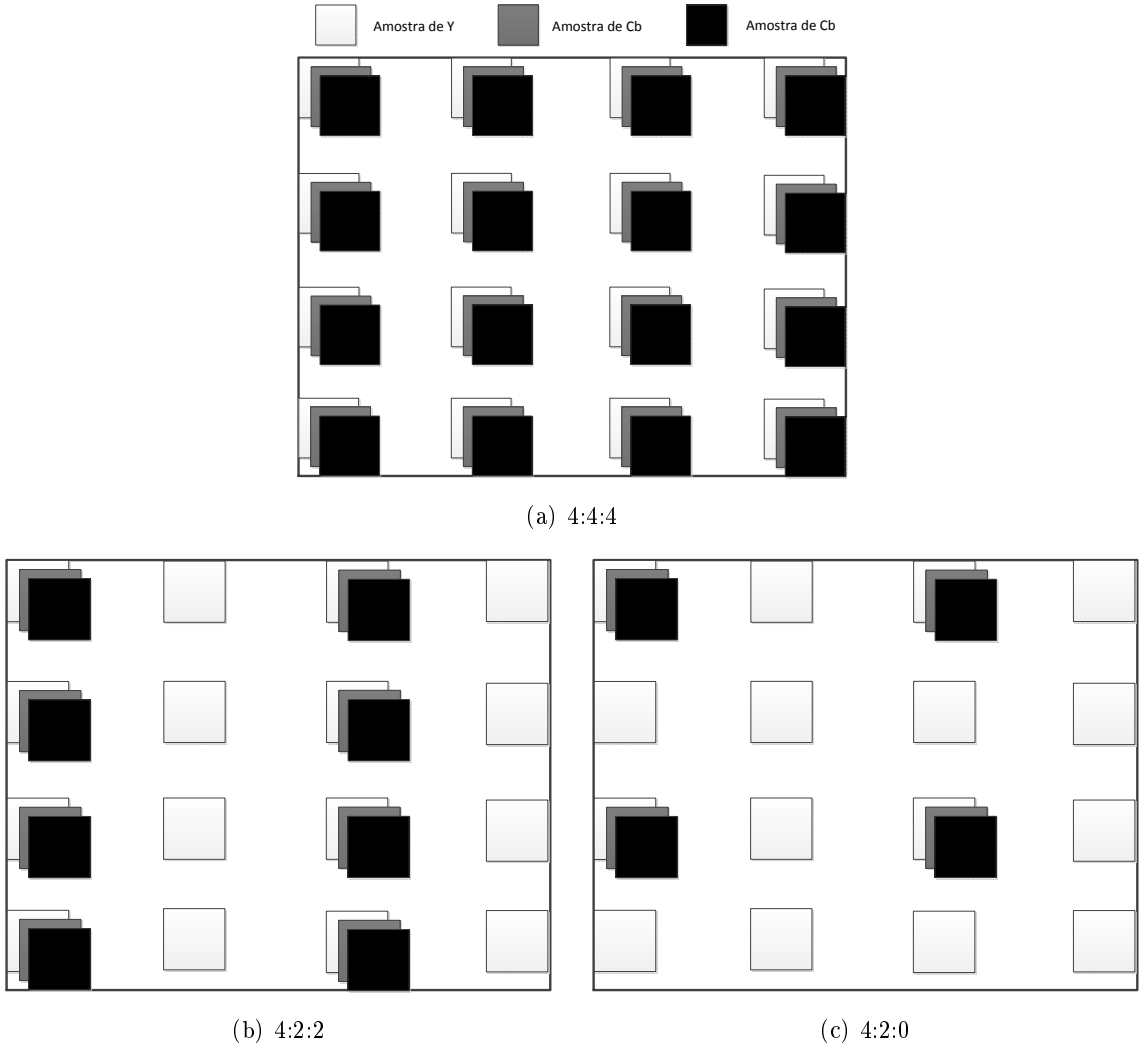
O sistema YCbCr não utiliza a crominância Cg, entre verde e luminância, pois é possível calculá-la utilizando as componentes Y, Cb e Cr.

Há diferentes tipos de amostragem para o sistema YCbCr [20], os mais comuns são: 4:4:4, 4:2:2 e 4:2:0. A amostragem 4:4:4 significa que cada pixel da imagem possui a total informação de suas três componentes, Y, Cb e Cr, assim a cada 4 componentes de luminância há 4 componentes de crominância. Já a amostragem 4:2:2 sub-amostra as componentes de crominância pela metade na direção horizontal, logo cada pixel possui informação de luminância, mas apenas 2 a cada 4 pixels na direção horizontal possuem informação de crominância. O sistema de amostragem 4:2:0 é o mais utilizado na compressão de vídeo e imagem, ele sub-amostra as componentes de crominância por um fator 4:1, assim, a cada 4 componentes de luminância há apenas 1 componente de cada crôminancia. Este tipo de amostragem não pode ser inferido diretamente de sua nomeclatura 4:2:0, já que esta não representa a escala entre as componentes, diferentemente dos outros tipos de amostragem citados. A Figura 2.4 mostra os diferentes tipos de amostragem do sistema YCbCr.

Embora o H.264/AVC suporte diferentes tipos de amostragem, o sistema de amostragem mais utilizado é o 4:2:0. [20] Como as componentes de crominância do sistema 4:2:0 são sub-amostradas em 4 vezes, este sistema necessita da metade da informação necessária para representar a imagem em relação ao RGB ou YCbCr 4:4:4. Um grupo de 4 pixels necessita de 12 componentes no sistema RGB, ou 3 componentes por pixel, já na amostragem 4:2:0, é necessário apenas 6 componentes para um grupo de 4 pixels, ou 1,5 componente por pixel. Em um sistema em que cada componente é representada por 8 bits, o sistema RGB necessita de 24 bits/pixel e o sistema YCbCr com sub-amostragem 4:2:0 de apenas 12 bits/pixel.

2.1.3 Codificação de imagem por predição

Devido à alta energia contida em uma imagem ou vídeo, a simples codificação sem eliminação de redundância pode consumir uma grande quantidade de bits, podendo tornar inviável a transmissão ou o armazenamento da imagem ou vídeo. Os padrões de codificação atuais realizam a codificação por predição, no qual cria-se uma predição da imagem a ser comprimida e subtrai-se a imagem predita da imagem original, restando apenas os valores das diferenças entre os pixels, esta sendo uma imagem de baixa energia em relação a imagem original, propícia a ser comprimida. Essa diferença entre a imagem predita e a original é chamada de resíduo. [20] Como os padrões de vídeo



(a) 4:4:4

(b) 4:2:2

(c) 4:2:0

Figura 2.4: Padrões de amostragem do sistema YCbCr. Cada quadrado representa uma componente de um pixel.

atuais são orientados a blocos, a predição da imagem é feita a partir de blocos, logo temos um macrobloco de resíduo. Assim, particiona-se a imagem em vários blocos e cria-se uma predição para cada bloco, em seguida é obtido o resíduo entre o bloco predito e o original e este resíduo é codificado por outras etapas do codificador.

Há dois tipos de predição no H.264/AVC, a predição intra-quadros e a inter-quadros. [20] A predição intra-quadros cria uma predição de cada macrobloco interpolando-os utilizando a informação dos pixels vizinhos a eles. A Figura 2.5 mostra a posição dos pixels vizinhos ao macrobloco que podem ser utilizados para interpolação do tipo intra-quadros. Existem diversos tipos de interpolação padronizados pelo H.264/AVC, como os pixels do macrobloco predito sendo a média dos pixels de 1 a 13 mostrados na Figura 2.5, ou a extrapolação horizontal dos pixels de 1 a 4. Outros métodos de predição intra-quadros utilizados pelo padrão H.264/AVC são discutidos em [20]. A Figura 2.6 ilustra um exemplo de predição intra-quadro e obtenção de resíduo de um macrobloco utilizando extrapolação vertical dos pixels vizinhos.

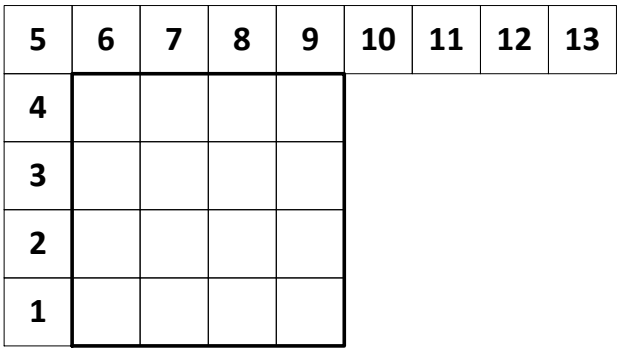


Figura 2.5: Os pixels de 1 a 13, vizinhos de um macrobloco, podem ser utilizados para predição tipo intra-quadro.

A predição inter-quadros utiliza da relação temporal de um vídeo e é discutida na Seção 2.1.4.

2.1.4 Modelo Temporal

O modelo temporal visa eliminar informações redundantes entre quadros de um vídeo, devido ao fato deles possuírem correlação temporal. Para a redução da redundância temporal, utiliza-se a predição inter-quadros que gera uma predição de um quadro a ser comprimido utilizando informações de outros, estes chamados de quadros de referência. Em seguida, a imagem predita é subtraída da imagem original, gerando uma imagem residual de menor energia e este resíduo é comprimido junto com as informações de predição do quadro. A Figura 2.7 ilustra a criação de resíduo entre dois quadros. O quadro 0 de referência é subtraído do quadro 1 a ser predito, obtendo assim o resíduo da predição. Como quadros de referência para a predição é possível utilizar tanto quadros temporalmente anteriores ao que será predito assim como quadros futuros, mas para isso, é preciso que os de referência sejam codificados antes de tal quadro. Assim como a predição intra-quadros, a predição inter-quadros utiliza partição de macroblocos. [20]

2.1.4.1 Estimação e Compensação de Movimento

A predição do modelo temporal utiliza a estimação de movimento [20] para determinar a melhor predição para um macrobloco. Em um vídeo, os objetos podem mudar de posição a cada quadro devido ao movimento, logo é possível estimar a correspondência de pixels entre diferentes quadros e este processo chama-se estimação de movimento. Nos padrões de vídeo orientados a blocos, estimação de movimento consiste em estimar a origem de cada macrobloco do quadro a ser predito para um quadro de referência. Tendo encontrado o macrobloco que mais se aproxima do bloco a ser predito, determina-se o vetor de movimento, este sendo o vetor que indica a distância, em pixels, entre a posição do macrobloco a ser predito e a sua correspondência na imagem de referência. A Figura 2.8 ilustra a estimação de movimento entre dois quadros. Com o vetor de movimento determinado é possível obter a predição do macrobloco, assim o macrobloco predito e o original são subtraídos para a obtenção do resíduo, este processo é chamado de compensação de movimento [20], e em seguida realiza-se a codificação utilizando os resíduos e os vetores de movimento dos

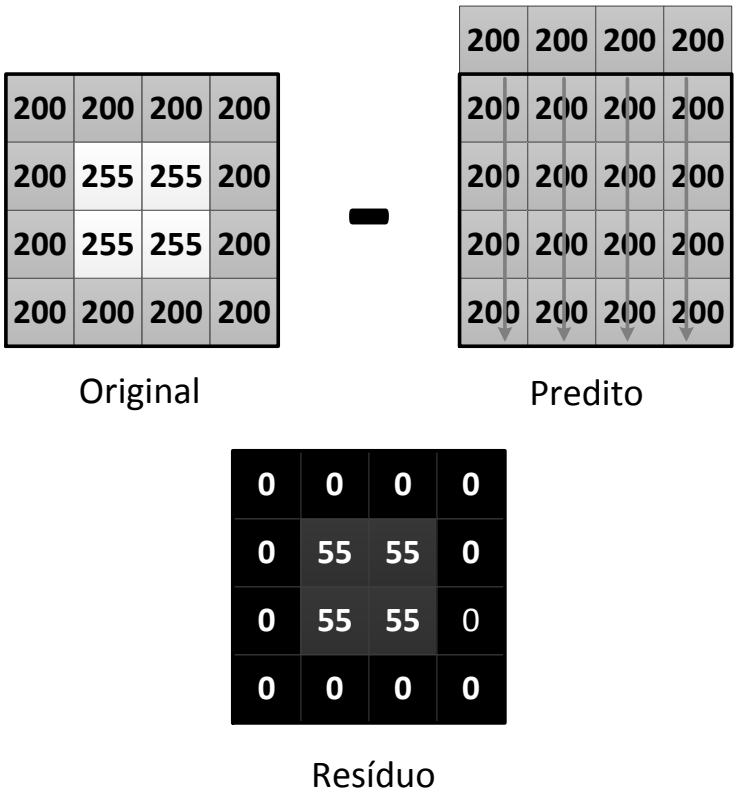


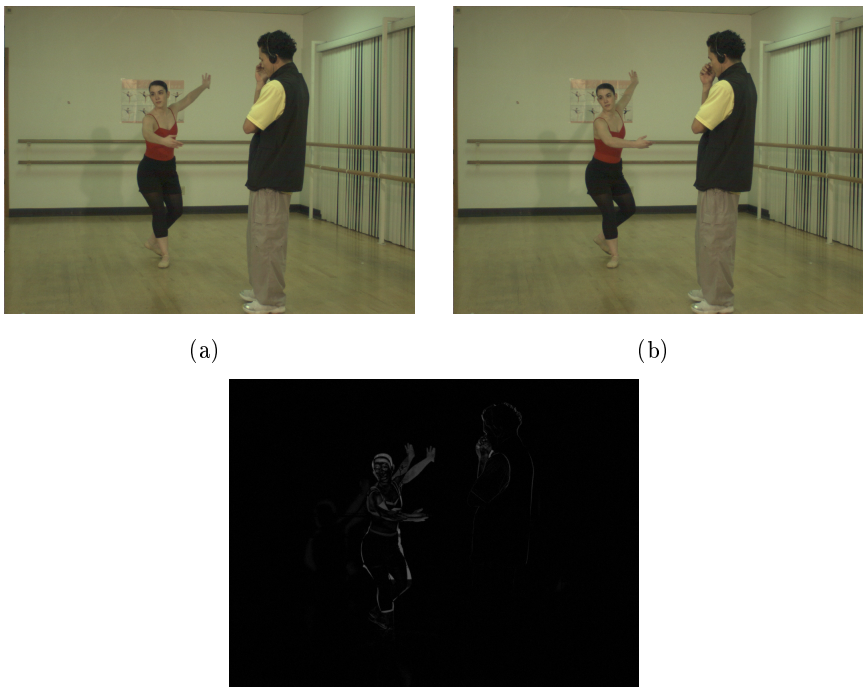
Figura 2.6: Exemplo de predição intra de um macrobloco com extrapolação vertical. O macrobloco é predito utilizando informações de pixels vizinhos localizados acima dele, e o macrobloco original é subtraído do predito, gerando o resíduo.

macroblocos, sendo possível a reconstrução do bloco original pelo decodificador utilizando essas informações. [20]

Na prática, a estimação de movimento é determinada pela diferença dos pixels de dois macroblocos. Para um determinado macrobloco a ser predito, o codificador procura por blocos nas imagens de referência que resultam na melhor predição e a busca é realizada dentro de uma determinada região de procura pré-estabelecida. Dentro da região de procura há diversos macroblocos e para cada macrobloco é calculada a SAD (*Sum of absolute differences*) [20] entre o bloco a ser predito e o original. Geralmente é utilizada a SAD da componente de luminância Y. SAD é a soma das diferenças absolutas, os pixels de mesma posição entre dois macroblocos são subtraídos e o módulo das diferenças é somado obtendo assim o resultado da SAD, como pode ser visto na equação 2.3, onde I corresponde a intensidade de um pixel de posição horizontal u e vertical v do macrobloco predito MB_{pred} ou de referência MB_{ref}.

$$SAD = \sum |I_{MB_{pred}}(u,v) - I_{MB_{ref}}(u,v)| \tag{2.3}$$

A SAD é determinante para a decisão da melhor predição para um macrobloco, pois quanto menor o valor da SAD, menor será o resíduo da estimação de movimento, logo menos bits serão necessários para a representação de um mesmo bloco. A Figura 2.8 ilustra a correspondência entre



(a) (b)

(c)

Figura 2.7: Resíduo da predição temporal sem estimação de movimento. O Quadro de referência é subtraído do quadro a ser predito, resultando no resíduo da predição. (a) Quatro 0 - Quadro de referência. (b) Quadro 1- Quadro a ser predito. (c) Resíduo - subtração do quadro original pelo quadro predito.

dois macroblocos em quadros diferentes, já a Figura 2.7(c) mostra o resíduo entre dois quadros distintos sem estimação de movimento, que equivale a vetores de movimento nulo. É possível observar que que o resíduo possui baixa energia, havendo resíduos de alta energia nas partes onde houve movimento entre os dois quadros.

2.1.5 Modelo Espacial

No modelo espacial de um codificador de vídeo, o objetivo é diminuir a redundância espacial de uma imagem utilizando a correlação entre pixels vizinhos. Isto é realizado pela codificação por transformada e quantização [20]. A codificação por transformada leva a informação de cores dos pixels (ou do resíduo) a um outro domínio propício a compressão e em seguida é feita a quantização dos dados neste novo domínio, sendo esta a etapa em que pode haver perdas na compressão de vídeo, mas em contrapartida há uma redução significativa na quantidade de bits necessária na representação o vídeo.

2.1.5.1 Transformada Discreta de Cosseno

A etapa da transformada tem como objetivo converter a imagem ou seu resíduo em outro domínio, visando a redução de bits da imagem comprimida. Para isso, os dados no domínio da

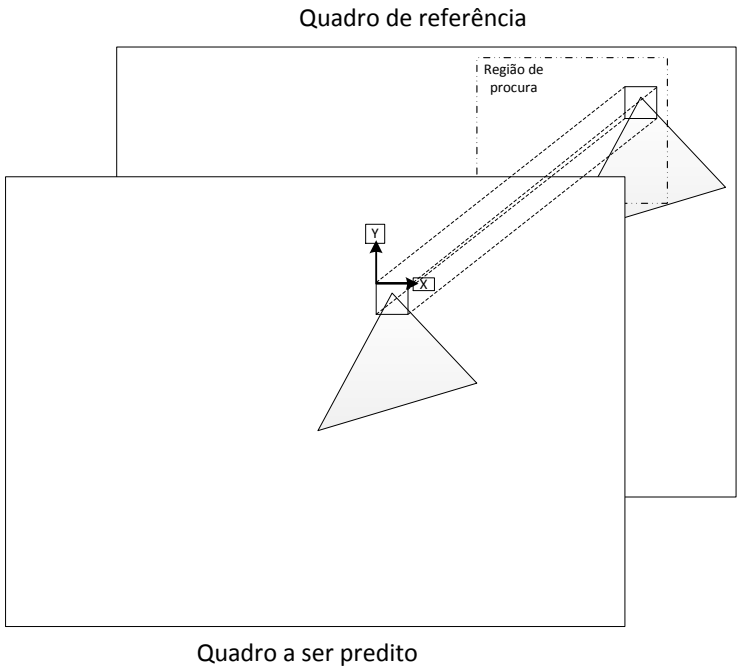


Figura 2.8: Estimação de movimento de um macrobloco. É determinado qual macrobloco da imagem de referência melhor prediz um determinado macrobloco do quadro a ser predito.

transformada devem ser descorrelacionados e compactos, onde a maior parte da energia do sinal deve estar concentrada em um pequeno intervalo de valores. A transformada deve ser reversível e de baixa complexidade computacional. Devido a estes fatores, a transformda mais popular na codificação de imagem e vídeo é a Transformada Discreta de Cosseno ou DCT (*Discrete Cosine Transform*) [19].

A DCT opera em uma matriz quadrada I, ou um bloco, de tamanho NxN e cria um outro bloco C de coeficientes de mesmo tamanho. A transformada pode ser representada por uma matriz A, onde a DCT direta é dada por:

$$I = ACA^T. \tag{2.4}$$

A transformada inversa, ou IDCT, por:

$$C = A^T IA, \tag{2.5}$$

onde a matriz de transformada A é dada pela equação:

$$\begin{aligned} A_{ij} &= C_i cos(\frac{(2j+1)i\pi}{2N}), \\ onde\, C_i &= \sqrt{\frac{1}{N}}\, para\, i = 0, \\ e\, C_i &= \sqrt{\frac{2}{N}}\, para\, i > 0. \end{aligned} \tag{2.6}$$

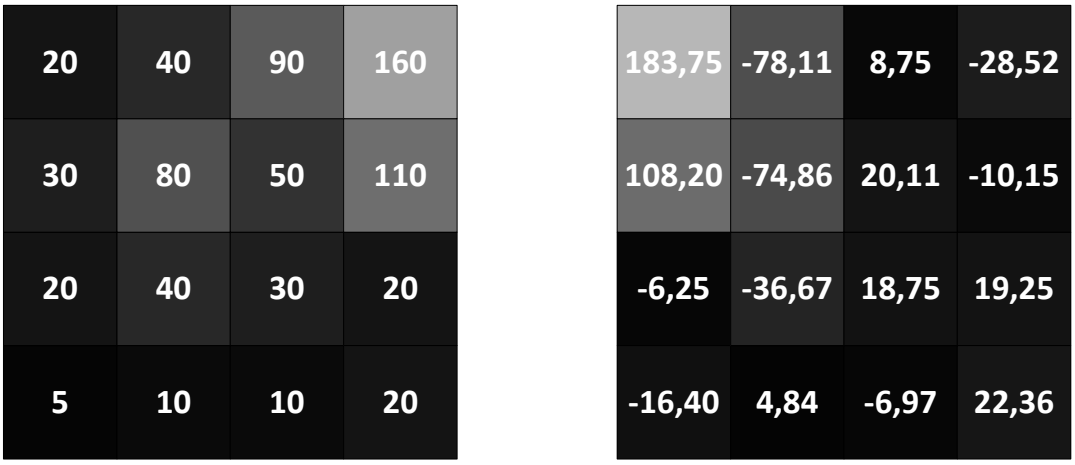


Figura 2.9: DCT de um bloco 4x4. (a) Bloco original. (b) Coeficientes DCT.

A Figura 2.9 mostra um bloco de 4x4 original e seus coeficientes da DCT.

O padrão H.264/AVC estabelece várias transformadas de diferentes tamanhos [2, 20]. A principal transformada utilizada é a DCT em blocos de tamanho 4x4, utilizada para codificação de resíduos. No H.264/AVC, a DCT 4x4 foi alterada para que haja apenas coeficientes inteiros na matriz A [2, 20], acarretando em uma redução de complexidade computacional e sem perdas de precisão durante a decodificação.

2.1.5.2 Quantização

Um quantizador mapeia um sinal de intervalo R para um sinal quantizado de menor intervalo Z. O padrão H.264/AVC utiliza um quantizador escalar para os resíduos [2, 20]. Um quantizador escalar mapeia um valor de um sinal em outro valor quantizado. Determinado um valor escalar para o quantizador, se o resíduo de um macrobloco na imagem for divido por ele, este resíduo será mapeado em um sinal de menor intervalo, possivelmente com alguns valores deste resíduo sendo fracionários. Caso seja feito o arredondamento desses valores para o inteiro mais próximo, ocorrerá perdas no processo de quantização, mas o sinal poderá ser representado com um número menor de bits do que a informação original. Assim, para um determinado valor X, a quantização pode ser generalizada por:

$$X_q = arredondamento(\frac{X}{QP}), \tag{2.7}$$

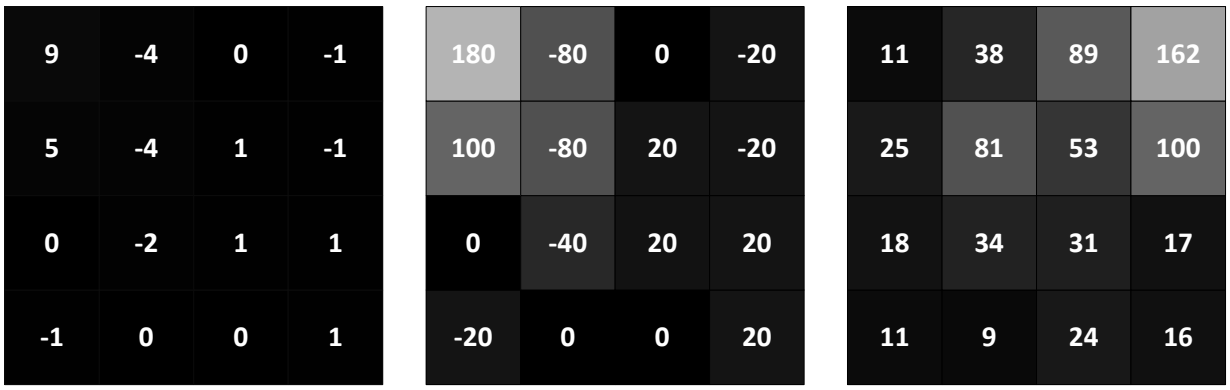
onde X é o valor original, X_q é o valor quantizado e QP é o passo de quantização. Desta forma, haverá perda de informação caso a divisão $\frac{X}{QP}$ não seja inteira. O valor pode ser recuperado por meio da equação:

$$Y = X_q QP, \tag{2.8}$$

onde Y é o valor recuperado. Como pode haver perdas na quantização, em grande parte dos casos o valor recuperado Y não será igual a X.

A Figura 2.10(a) mostra a quantização dos coeficientes da DCT da Figura 2.9(b) utilizando

QP de 20, a Figura 2.10(b) mostra a recuperação dos coefficientes da DCT devido a multiplicação dos coeficientes quantizados por $QP = 20$ e a Figura 2.10(c) mostra o bloco recuperado devido a aplicação da DCT inversa dos coeficientes recuperados.



(a) (b) (c)

Figura 2.10: Quantização e recuperação do bloco da figura 2.9(b). (a) Coeficientes de um bloco quantizados por $QP = 20$. (b) Coeficientes de (a) recuperados pela multiplicação por $QP = 20$. (c) Bloco recuperado após a aplicação da DCT inversa no bloco de (b).

2.1.6 Tipos de Quadros

É chamado de *slice* um grupo de macroblocos organizados na sequência do tipo *raster*. Sequência do tipo *raster* é a sequência em uma imagem que a percorre da esquerda para a direita e de cima para baixo, como mostrado na Figura 2.11. No padrão H.264/AVC, um *slice* pode conter um número variável de macroblocos, desde apenas um até o número de macroblocos contido em uma imagem [20]. A implementação mais comum do H.264/AVC é a de que cada *slice* seja um quadro inteiro, ou seja, a cada quadro há apenas um *slice* que compreende todos os macroblocos do quadro.

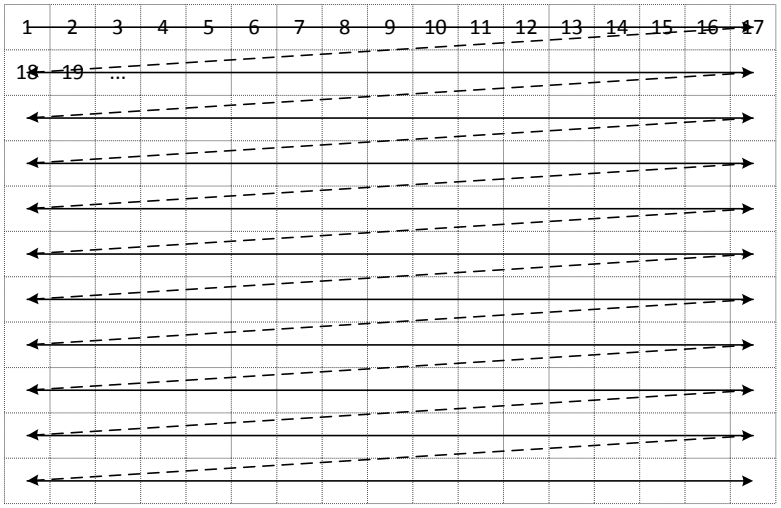


Figura 2.11: Escaneamento do tipo *raster*.

Há três tipos principais de *slices* no padrão H.264/AVC: tipo I, P e B. Os *slices* do tipo I são grupos de macroblocos nos quais só há predição do tipo intra-quadro. Já *slices* do tipo P podem ser preditos tanto por predição intra-quadro como predição inter-quadros, mas esta ocorre apenas utilizando um quadro de referência, logo cada macrobloco é predito apenas por um outro macrobloco. Os *slices* tipo B podem ser preditos por intra-quadro ou inter-quadros. Para um macrobloco do tipo B, a predição inter-quadros pode ser feita a partir de dois outros blocos de referência, diferentemente dos *slices* tipo P, ou seja, o tipo P utiliza apenas um quadro como referência para a predição do tipo inter e o slice tipo B pode utilizar dois quadros como referência, sendo o macrobloco predito a média dos macroblocos resultantes da estimação de movimento. Desta forma, haverá um vetor de movimento para cada referência nos slices tipo B.

A Figura 2.12 mostra um grupo de quadros no qual cada quadro é um tipo de *slice* diferente. As setas têm como partida os quadros que utilizam predição inter-quadros e têm como chegada os quadros utilizados como referência. Pode-se observar que os quadros tipo P utilizam tanto quadros anteriores como futuros a ele como referência, assim como quadros tipo B, mas estes utilizam até dois quadros como referência para uma mesma predição.

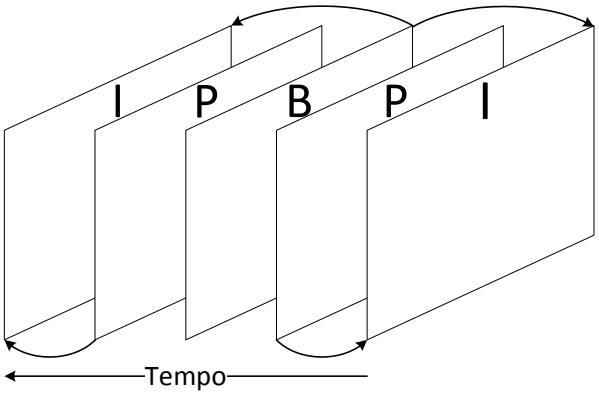


Figura 2.12: Exemplo de predição de um grupo de quadros. A seta parte do bloco a ser predito até o bloco de referência utilizado para a predição.

2.1.7 Decisão do tipo de Macrobloco - Rate-Distortion

A otimização taxa-distorção, RDO (*Rate-Distortion Optimization*), é um método utilizado para otimizar o processo de codificação de vídeo. Em suma, a RDO otimiza a seleção do tipo de predição de macroblocos, balanceando a distorção de qualidade em relação a quantidade de bits necessária para a codificação levando em consideração cabeçalhos, vetores de movimento e coeficientes. Desta forma, ela decide qual o tamanho do macrobloco e sua predição para a codificação, qual predição será utilizada, predição intra, inter, ou o macrobloco será do tipo *skip* ou *direct* [20]. Embora uma pesquisa em cima da RDO não seja feita neste trabalho, é necessário o entendimento de sua função para a compreenssão de decisões tomadas nos experimentos desta dissertação. Informações adicionais sobre RDO podem ser encontradas em [20].

2.1.8 Codificador de Entropia

O codificador de entropia [20] reduz a redundância estatística do sinal de saída. Após a redução de redundância temporal e espacial, os resultados dessas etapas que consistem em cabeçalhos, coefficientes de resíduos quantizados, vetores de movimento e informações suplementares, possuem sequências de bits a serem codificadas. O codificador de entropia comprimi estas sequências de bits em sequências menores e sem perda de dados. Isto ocorre devido ao fato de que certas sequências de bits possuem uma probabilidade maior de aparição e para estas sequências um código menor é definido. Para as sequências com menor probabilidade de aparição, um código maior é definido. Desta forma é possível a compressão das sequências de bits das etapas anteriores do codificador de vídeo.

O padrão H.264/AVC utiliza dois tipos de codificadores de entropia, CABAC(*Context-Adaptive Arithmetic Coding*) e CAVLC (*Context-Adaptive Variable length coding*). Como esta parte do codificador de vídeo não faz referência ao trabalho deste Mestrado, fica a critério do leitor um estudo aprofundado do tema [20]. Além do codificador de entropia, há outras etapas [20] não cobertas por este Capítulo, como a predição dos vetores de movimento, que utiliza a subtração dos vetores de movimento de macroblocos vizinhos para a redução de informação e a reorganização de dados dos coeficientes quantizados, para aproveitar do fato da quantização criar uma matriz esparsa de coeficientes.

2.1.9 Medidas de qualidade

Devido a redução de qualidade na codificação com perdas de um vídeo, é de interesse uma métrica que determine a qualidade visual de vídeos codificados. Esta qualidade visual pode ser definida tanto subjetivamente quanto objetivamente.

Diversos fatores influenciam na determinação da qualidade visual subjetiva, já que esta varia de pessoa para pessoa devido as diferentes percepções de cada uma para o mesmo vídeo. Para determiná-la, é necessário fazer experimentos com um vasto grupo de espectadores e avaliar a percepção de tal grupo sobre um determinado vídeo.

A qualidade visual objetiva pode ser determinada por métricas matemáticas. A métrica mais utilizada atualmente é a Relação Sinal-Ruído de Pico ou PSNR (*Peak signal-to-noise ratio*) [20], que mede a distorção entre o vídeo codificado e o vídeo original, e tem como unidade o decibel (dB). O erro médio quadrático MSE (*Mean Squared Error*) entre duas imagens I e K de tamanho $M \times N$ é dado por:

$$MSE = \frac{1}{MN} \sum_{i=0}^{M-1} \sum_{j=0}^{N-1} (I(i,j) - K(i,j))^2 \tag{2.9}$$

Assim, o erro médio quadrático é a média do somatório das diferenças entre os pixels de I e J ao quadrado. A PSNR utiliza o erro médio quadrático entre duas imagens e é dada pela fórmula:

$$PSNR_{db} = 10log_{10} \frac{(2^n - 1)^2}{MSE} \tag{2.10}$$

Onde n é o numero de bits que representa uma amostra da imagem. Assim, no caso em que cada componente de um pixel é representado por 8 bits, a PSNR de uma componente é calculada por:

$$PSNR_{db} = 10log_{10}\frac{255}{MSE} \tag{2.11}$$

Pela equação 2.11 é possível inferir que caso as duas imagens sejam idênticas, a divisão $\frac{255}{MSE}$ tenderá a infinito e a PSNR da imagem será aproximadamente 99.99 dB. Quanto maior for a diferença entre as duas imagens, menor será o valor de $\frac{255}{MSE}$ e menor será o valor da PSNR. Para duas sequências de vídeo o usual para a obtenção da PSNR total, de acordo com a recomendação da ITU-T [22], é calcular a PSNR a cada quadro e em seguida obter a média das PSNRs de cada quadro.

De fato, somente o valor da PSNR não determina a qualidade visual de uma imagem, mas ela têm se mostrado muito útil na comparação de qualidade de compressão de vídeos e imagens. Isto ocorre pois ela é comumente utilizada para uma mesma imagem de teste ou sequência de vídeo, mas com diferentes tipos de compressão para serem comparados. Para esta comparação, utiliza-se curvas PSNR × taxa de bits (*BitRate*) , onde a PSNR dos vídeos codificados são calculadas utilizando o vídeo original como referência e as curvas são traçadas utilizando a PSNR de cada vídeo com sua respectiva taxa de bits, que mede a quantidade de bits em um segundo do vídeo codificado e tem como unidade *bps* (*bits per seconds* ou bits por segundo). A Figura 2.13 mostra um exemplo de uma curva de PSNR × BitRate de uma mesma sequência de vídeo codificado utilizando os padrões H.264/AVC e MPEG-2. Observa-se que para uma mesma PSNR, a curva H.264 possui um valor menor de taxa de bits por segundo, significando que o H.264 consegue uma compressão maior de dados para uma mesma qualidade visual.

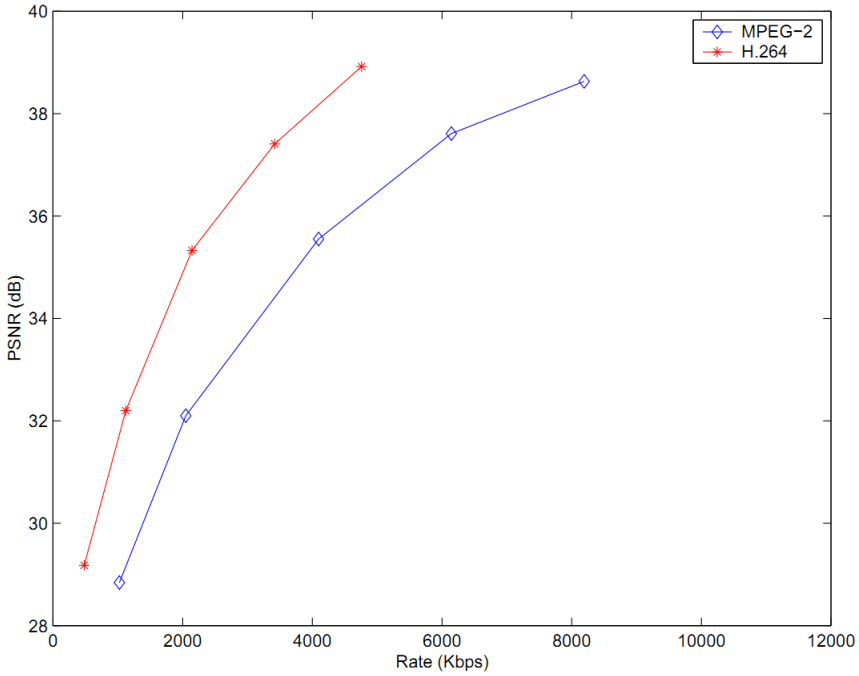


Figura 2.13: Exemplo de uma curva PSNR x Bitrate.

2.2 Sistemas de Vídeo e Imagens

É necessária a criação de um modelo matemático de câmeras para que se possa compreender e processar imagens e vídeos capturadas por elas. Desta forma, um modelo matemático de câmera do tipo *pinhole* é apresentado nas Seções 2.2.1 e 2.2.1.1. Este modelo é necessário para diversos tipos de processamento, como a calibração de câmeras [6], que corrige as distorções causadas por lentes, a estimação de mapas de profundidade de câmeras em sistemas multivistas (Seção 3.2.2) e a síntese de vista (Capítulo 4).

Na Seção 2.2.2 são apresentados alguns filtros de imagens e uma operação básica de imagem, a dilatação, utilizados neste trabalho.

2.2.1 Modelo de câmera

O modelo de câmera *pinhole*, ou "buraco de alfinete", é o modelo mais simples que relaciona pontos em um espaço real tridimensional e sua representação em uma imagem 2D. Neste modelo, a imagem capturada pela câmera é representada por um plano. Um ponto de entrada de luz, chamado de abertura, projeta um ponto de posição $[X \quad Y \quad Z]^T$ no espaço real tridimensional para um ponto de posição $[u \quad v \quad z]^T$ no plano que representa a imagem da câmera, como ilustrado na Figura 2.14. Nela, u e v representam a posição horizontal e vertical, respectivamente, do ponto projetado no plano da imagem e z corresponde a profundidade do pixel no espaço tridimensional, logo $z = Z$. Este modelo pode ser utilizado para câmeras que não possuem distorções de lente. Maior discussão sobre o modelo pode ser encontrada em [7].

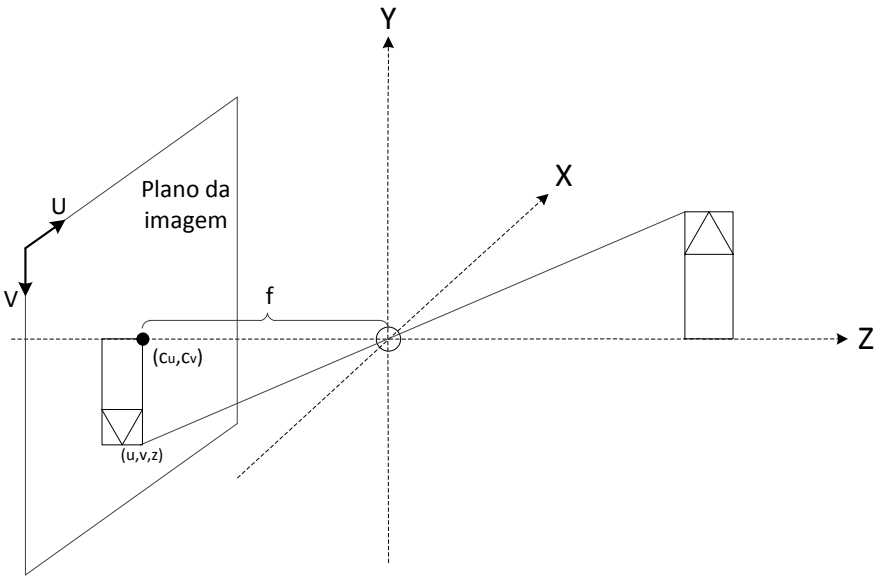


Figura 2.14: Modelo de câmera *pinhole*. Um objeto no espaço tridimensional é projetado ao plano de imagem da câmera.

2.2.1.1 Parâmetros de câmera

Uma câmera possui parâmetros intrínsecos, que descrevem suas características físicas, e extrínsecos, que descrevem sua relação a um sistema global de coordenadas localizado no espaço real tridimensional. Estes parâmetros são geralmente representados em matrizes, formando as matrizes intrínseca e extrínseca da câmera.

No modelo *pinhole*, a câmera possui uma distância focal, que é a distância do plano da imagem até sua abertura, e um ponto de projeção, que é o centro do plano da imagem onde os eixos ópticos intersectam o centro desse plano, como pode ser visto na Figura 2.14. A matriz intrínseca $\mathbf{A}$ é formada por tais parâmetros, sendo descrita como:

$$\mathbf{A} = \begin{pmatrix} f_u & \gamma & c_u \\ 0 & f_v & c_v \\ 0 & 0 & 1 \end{pmatrix}, \tag{2.12}$$

onde f_u e f_v são as distâncias focais da câmera em relação aos eixos u e v do plano 2D da câmera, (c_u, c_v) é o ponto principal de projeção e γ é a relação entre os eixos x e y, como normalmente este é de 90^o graus, logo $\gamma = 0$.

Os parâmetros extrínsecos da câmera representam a posição do centro de coordenadas global do espaço real tridimensional para o centro de coordenadas da câmera. Desta forma, esses parâmetros são compreendidos por uma matriz de rotação $\mathbf{R}$ e uma matriz de translação T, em que a matriz extrínseca $[\mathbf{R}|\mathbf{T}]$ é a concatenação delas. Estas matrizes podem ser obtidas por métodos de calibração de câmera e uma maior discussão pode ser encontrada em [7].

Com as matrizes extrínseca e intrínseca de uma câmera definidas, é possível determinar a relação entre um ponto do espaço e seu correspondente no plano da câmera pela relação:

$$\mathbf{x} = \mathbf{PX}, \tag{2.13}$$

onde $\mathbf{x} = [u \quad v \quad z]^T$ é o ponto no plano 2D da câmera, $\mathbf{X} = [X \quad Y \quad Z]^T$ é o ponto no sistema de coordenadas tridimensional e $\mathbf{P} = \mathbf{K}[\mathbf{R}|\mathbf{T}]$ é a matriz de projeção. Assim, utilizando a equação 2.13, é possível determinar a posição da imagem que a câmera capturará um ponto no espaço tridimensional, como visto na Figura 2.14.

2.2.2 Processamento de imagens

O processamento de imagens é necessário para diversas áreas como detecção de objetos, codificação de vídeos, navegação de robôs, entre outras. Nesta Seção será apresentado os filtro Canny, de mediana e a operação morfológica de dilatação, que serão utilizados neste trabalho.

2.2.2.1 Filtro de mediana

O filtro de mediana é um filtro espacial utilizado para suavizar imagens, normalmente utilizado na eliminação de ruído. Esse filtro consiste na substituição de um determinado pixel da imagem pela mediana dos seus pixels vizinhos. Para isso é necessário determinar a janela de pixels vizinhos a ser considerada. Como exemplo, no caso de uma janela de 3x3 pixels, é considerado um total de 9 pixels. Determina-se a mediana da intensidade dos pixels dessa janela e o pixel central é substituído por ela . Assim, o procedimento para a aplicação de um filtro de mediana em um determinado pixel de posição (u, v) é, primeiramente, determinar o tamanho da janela do filtro, em seguida organiza-se as intensidades dos pixels dessa janela em ordem crescente, por último, a mediana é determinada e ela é associada ao pixel de posição (u, v) em questão. A figura 2.15 mostra um exemplo da aplicação do filtro de mediana em um pixel com grande discrepância de intensidade dos pixels vizinhos. O filtro de mediana suaviza as altas frequências, sendo um filtro passa-baixas. Uma explicação detalhada é discutida em [5].

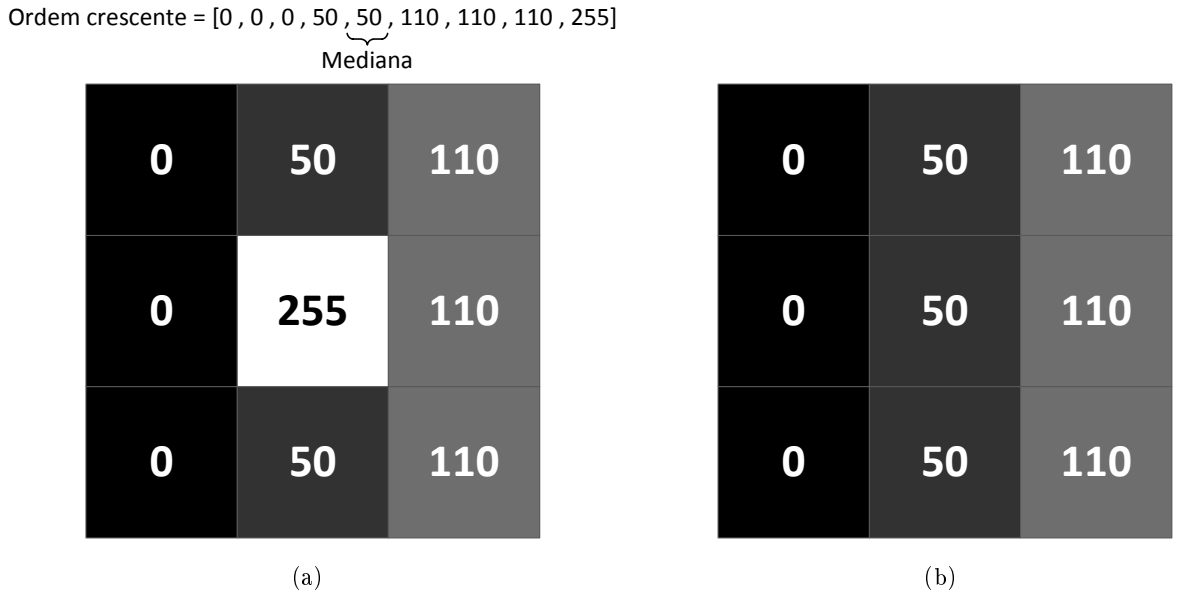


Figura 2.15: Exemplo de aplicação do filtro de mediana. (a) As intensidades dos pixels de uma janela de 3x3 pixels são organizadas em ordem crescente e a mediana é determinada. (b) O pixel central é substituído pela mediana determinada em (a).

2.2.2.2 Filtro Canny para detecção de contornos

O filtro Canny foi apresentado por John Canny e é detalhadamente discutido em [3]. O filtro Canny é um filtro de detecção de contornos (ou bordas) de convolução que usa a primeira derivada da função Gaussiana, $g'(k)$. É um filtro robusto a ruídos, havendo um tratamento destes por convolução com uma gaussiana para posteriormente ser feita a detecção de contorno. Esta é realizada por meio do cálculo de gradientes das intensidades da imagem. Caso o gradiente de um dado pixel seja maior que uma limiar estabelecido, ele é considerado como borda. Caso menor que outro limiar também estabelecido, é descartada a hipótese de ser borda. Os pixels que possuem

gradiente de valor entre esses limiares são considerados como borda dependendo dos pixels vizinhos a ele. Um exemplo de detecção de contorno do tipo Canny pode ser visto na Figura 2.16.

(a) (b)

Figura 2.16: Filtro Canny para detecção de contornos. (a) Imagem a ser aplicado o filtro de Canny. (b) Contornos obtidos pelo filtro de Canny em (a).

2.2.2.3 Dilatação

Dilatação é um conceito simples, embora útil. Para uma imagem preto e branco, considera-se uma janela de tamanho N×N pixels com N ímpar. A dilatação ocorre em um determinado pixel central dessa janela, de posição (u, v) na imagem, e ele será definido como preto caso haja apenas pixels pretos na janela, se houver um ou mais pixels brancos, ele será definido como branco. Assim, a dilatação tende a aumentar as áreas brancas da imagem. A Figura 2.17 mostra a imagem 2.16(b) dilatada com uma janela 5x5. Uma explicação mais extensa pode ser lida em [5].

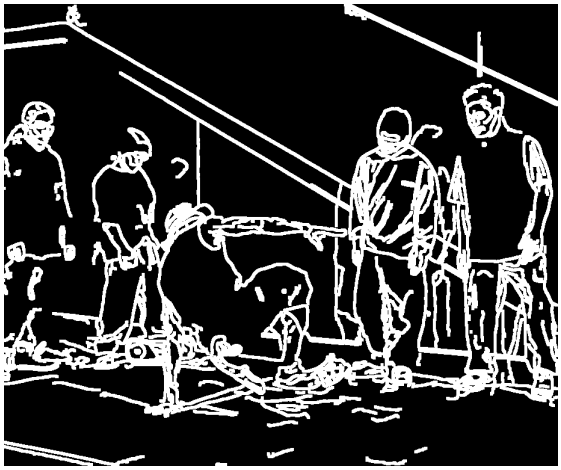


Figura 2.17: Dilatação da Figura 2.16(b)

Capítulo 3

Sistemas de Ponto de Vista Livre

3.1 Introdução

Como explicado na seção 1.1, um sistema de ponto de vista livre (FTV do inglês *Free-viewpoint television*) é um sistema interativo que possibilita ao usuário escolher de qual ponto de vista assistirá a cena.

O conteúdo exibido ao espectador pode ser gerado por computação gráfica ou pode ser gravado por câmeras. No primeiro caso, a cena é toda gerada por computadores, como em jogos eletrônicos, sendo possível a exibição da cena sob qualquer ponto de vista. No caso em que o conteúdo é gravado, para que haja uma experiência agradável ao usuário na escolha do ponto de vista, seria necessário a utilização de um número muito grande de câmeras na captura da cena, o que tornaria o sistema extremamente complexo.

Na gravação de um conteúdo em um sistema FTV, existem dois tipos de cena: a cena estática, na qual não há movimento, e a cena dinâmica, em que há movimento. A gravação do conteúdo pode ser feita por uma única câmera, que percorre uma trajetória horizontal ao redor da cena mantendo sua posição vertical constante, capturando assim diversos pontos de vista, como ilustrado na Figura 3.1. Este caso só pode ser aplicado para cenas estáticas pois todos os pontos de vistas diferentes precisam ser capturados simultaneamente, antes que ocorra o movimento da cena. Este método não pode ser utilizado em cenas dinâmicas. As animações em *stop-motion* se enquadram como cenas estáticas, que são animações quadro a quadro feitas normalmente com objetos inanimados, como massa de modelar. A cena é montada com os objetos e sua imagem é capturada por uma máquina fotográfica. Em seguida, a cena é alterada movendo os objetos de posição e sua imagem é capturada novamente. Cada nova imagem capturada é um quadro, e colocando-os em sequência e transformando-os em um vídeo há a criação de movimento. Assim, é possível obter o conteúdo necessário para a aplicação de FTV utilizando apenas uma única câmera na captura da cena e um dispositivo mecãnico que translada essa câmera ao redor do cenário.

As cenas dinâmicas e estáticas podem ser capturadas por um sistema multi-vista, que consiste na utilização de diversas câmeras para a captura simultânea de diversos pontos de vista da cena. Este método é necessário para a captura de conteúdos FTV de cenas dinâmicas. O restante das

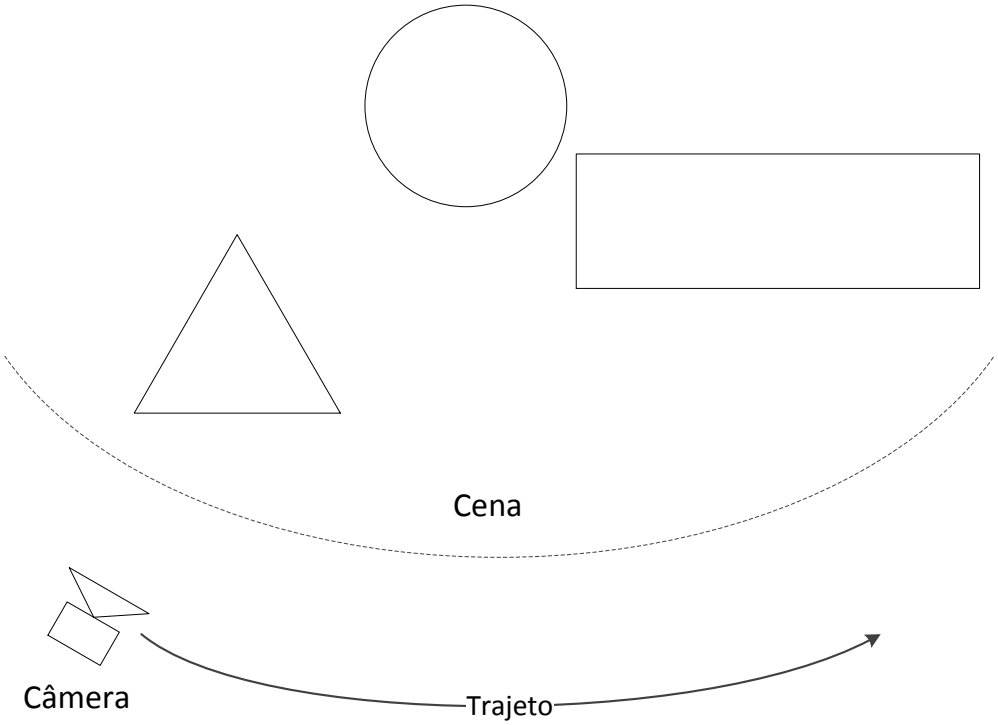


Figura 3.1: Gravação de uma cena estática por apenas uma câmera. A câmera captura diversos pontos de vista de uma cena seguindo uma trajetória ao redor desta.

vistas da cena podem ser, então, interpoladas utilizando as imagens capturadas pelas câmeras. Este método de interpolação, chamado de síntese de vista, é explicado no capítulo 4.

3.2 Sistema Multi-Vista

Um sistema multi-vista captura imagens de diversos pontos de vista de uma mesma cena. A Figura 3.2 mostra um exemplo de sistema multi-vista. Como objetos em uma mesma cena possuem profundidades diferentes em relação as câmeras, pode haver oclusão de objetos em diferentes regiões de suas imagens, como ilustrado na Figura 3.3. Devido as oclusões, cada câmera captura informações da cena que podem não haver sido capturadas por outras câmeras.

3.2.1 Codificação De Sistemas Multi-Vista

Para o funcionamento de um sistema que permita que o usuário possa sintetizar as vistas que deseja como o FTV, é necessária a codificação e transmissão eficiente dos vídeos capturados. Um método simples de codificação para um sistema multi-vista seria a codificação independente de todas as vistas utilizando o padrão H.264/AVC, como representado na Figura 3.4 que mostra a disposição dos quadros nessa codificação. Assim cada vista seria independentemente codificada e enviada para o destino. Este método, chamado de *simulcast*, não é eficiente pois não reduz a redundância das imagens entre câmeras adjacentes.

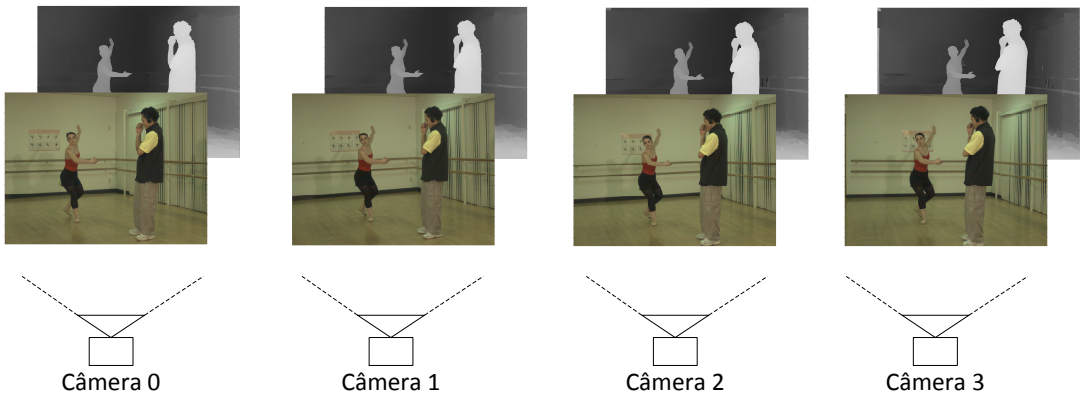


Figura 3.2: Sistema multi-vista. A cena é capturada por câmeras com diferentes pontos de vista.

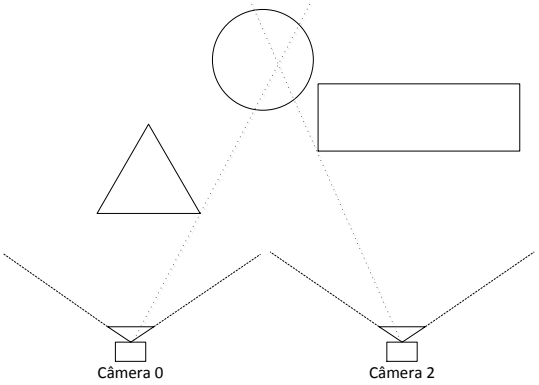


Figura 3.3: Exemplo de oclusão em sistema multi-vista. As duas câmeras capturam em suas imagens diferentes partes do objeto circular devido a oclusão causada pelos objetos triangular e retangular em cada câmera.

A codificação de vídeo multi-vista (MVC do inglês *Multiview Video Coding*) [13] utiliza o padrão H.264/MVC, que extende o padrão H.264/AVC, para compressão de vídeos de um sistema multi-vista, tirando proveito tanto da relação temporal quanto da relação espacial entre vistas adjacentes. Considerando três vista, 0, 1 e 2, onde a vista 1 é a central, o MVC codifica as vistas na ordem 0-2-1, sendo que as vistas codificadas posteriormente utilizam as já codificadas como referência para a predição do tipo inter-quadros. Como exemplo, a vista 0 é codificada independetemente das outras. Após isto, a vista 2 é codificada com predição inter temporal utilizando quadros anteriores da própria vista como referência, mas também utilizando quadros da vista 0 como referência. Em seguida a vista 1 é codificada, utilizando tanto o modelo temporal como utilizando quadros das vistas 0 e 1 como referências, assim diminuindo redundâncias espaciais entre as câmeras. A Figura 3.5 ilustra as operações de predição multi-vista, onde a vista direita é codificada independetemente das outras, a vista esquerda utiliza a vista direita para predição e a vista central utiliza as duas outras vistas para predição. Já a Figura 3.6 mostra a disposição dos tipos de quadros de um sistema *multicast*. Foi comprovado que o método MVC para compressão de sistemas multi-vistas consegue uma performance superior ao método *simulcast* [4]. Vários testes foram realizados para tal comprovação [25] e para sistemas com até 8 vistas houve um ganho médio de 20% de redução da taxa de bits. Uma discussão maior sobre formas de compressão e armazenamento de vídeos de

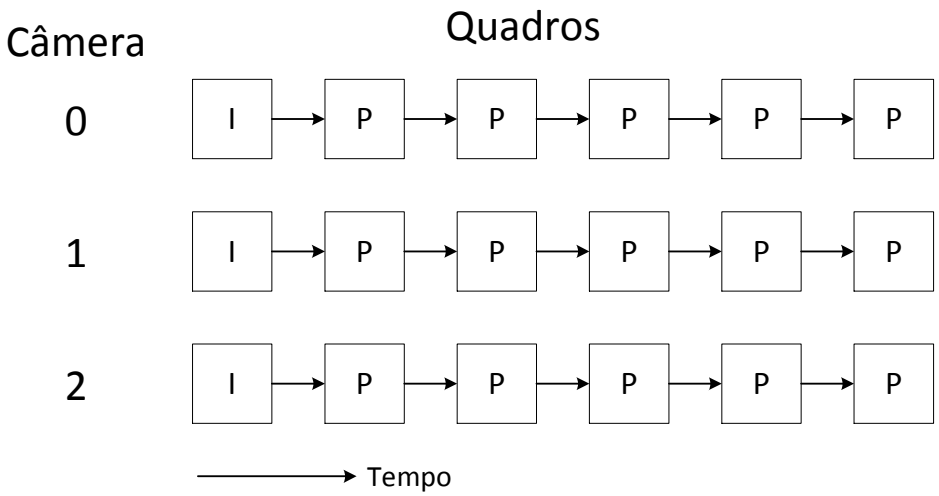


Figura 3.4: Codificação *simulcast*. Os vídeos capturados por cada câmera são codificados independentemente. A diferença entre os quadros de tipo I, P e B são apresentados na seção 2.1.6.

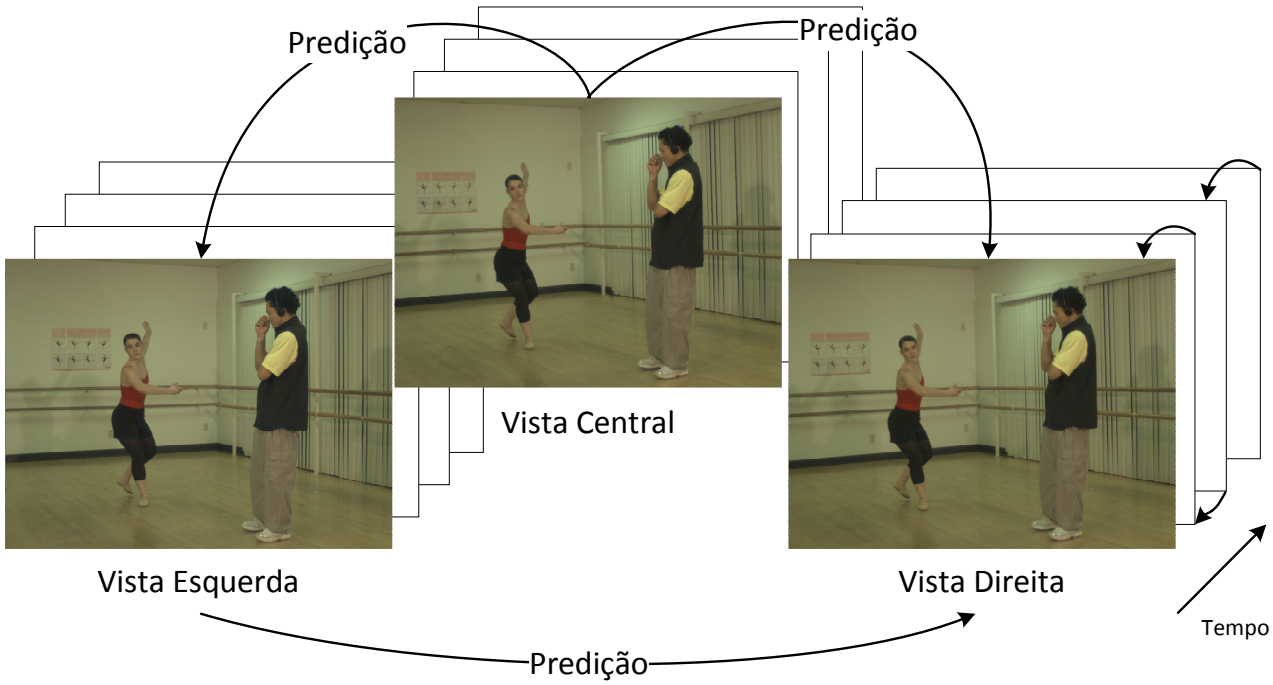


Figura 3.5: Predição do tipo inter utilizada para cada vista na codificação *multicast*, em que um quadro utiliza como referência para sua predição o quadro de referência apontado pela seta.

sistemas multi-vistas podem ser encontrada em [26].

O JMVC (*Joint Multiview Video Coding*) é o software de referência para o projeto de codificação de vídeos multi-vista (MVC) do grupo de vídeo JVT (*Joint Video Team*) do ISO/IEC MPEG. Ele é utilizado como padrão para testes e pesquisas em sistemas multi-vistas. Este programa será utilizado como codificador de vídeo para os testes de compressão deste trabalho.

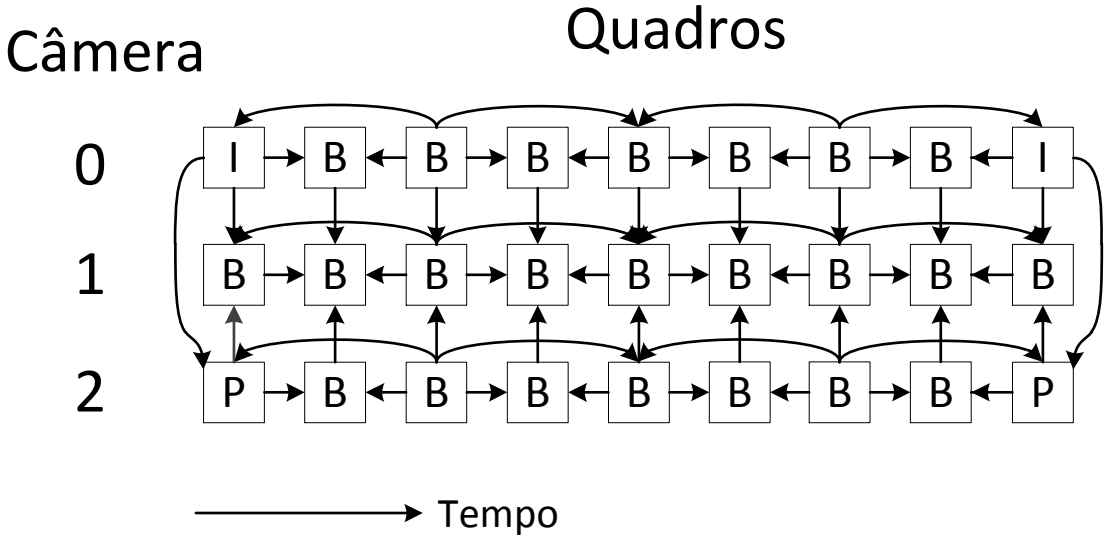


Figura 3.6: Codificação *multicast*. Para cada câmera do sistema multi-vista, é feita a predição espacial, em que elas utilizam as câmeras adjacentes para predição do tipo inter, e a temporal. A diferença entre os quadros de tipo I, P e B são apresentados na seção 2.1.6.

3.2.2 Mapas de profundidade

Os mapas de profundidade representam a informação de profundidade de imagens. Eles são normalmente representados por imagens em escala de cinza em que cada pixel é representado por 8 bits, ou seja, havendo 256 valores possíveis de profundidade. Eles são importantes em sistemas multi-vistas pois ajudam a determinar a relação entre os pixels das câmeras desse sistema, além de possibilitar a criação de vistas sintetizadas. Na determinação da profundidade de cada pixel, é determinada a profundidade em distância (em metros, centímetros, etc) até um sistema de referência. Após a determinação da profundidade completa da imagem, esses valores de profundidade são normalizados de acordo com a equação 3.1 para serem representados por imagens em escala de cinza.

São necessárias duas variáveis para definir a normalização do mapa de profundidade. A distância mínima Z_{min} dos pixels da imagem até um sistema de coordenadas e a distância máxima Z_{max} deles até esse sistema. Em sistemas multi-vistas, o centro do sistema de coordenadas em que os eixos se encontram, de posição (0,0,0), pode tanto ser diferente para cada câmera, como pode ser o mesmo para todos as câmeras. No primeiro caso, é comum a utilização do centro de coordenadas na mesma posição da câmera, assim o mapa de profundidade representa a distância dos objetos até cada câmera. No segundo caso, o centro do eixo de coordenadas é comumente definido na mesma posição de uma das câmeras do sistema. A normalização é feita a partir da fórmula:

$$D(x,y) = \left(\frac{1}{Z(x,y)} - \frac{1}{Z_{max}}\right)\frac{255}{\frac{1}{Z_{min}} - \frac{1}{Z_{max}}} \tag{3.1}$$

Onde D é a intensidade do pixel do mapa de profundidade, Z é a profundidade e os índices (x, y) representam a posição do pixel na imagem. O olho humano consegue definir melhor a

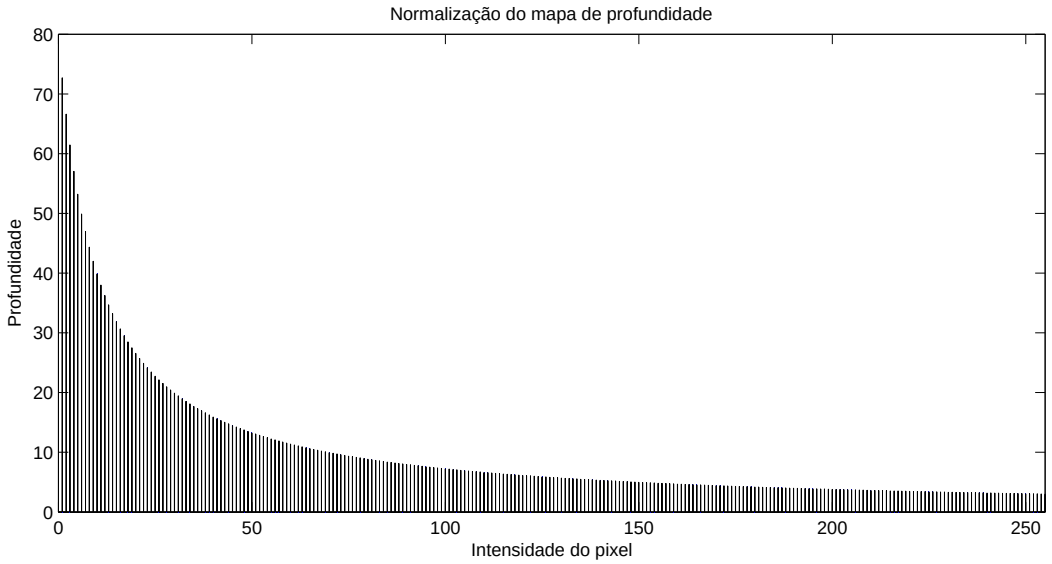


Figura 3.7: Ilustração da escala de normalização de um mapa de profundidade com $Z_{min} = 3$ e $Z_{max} = 80$.

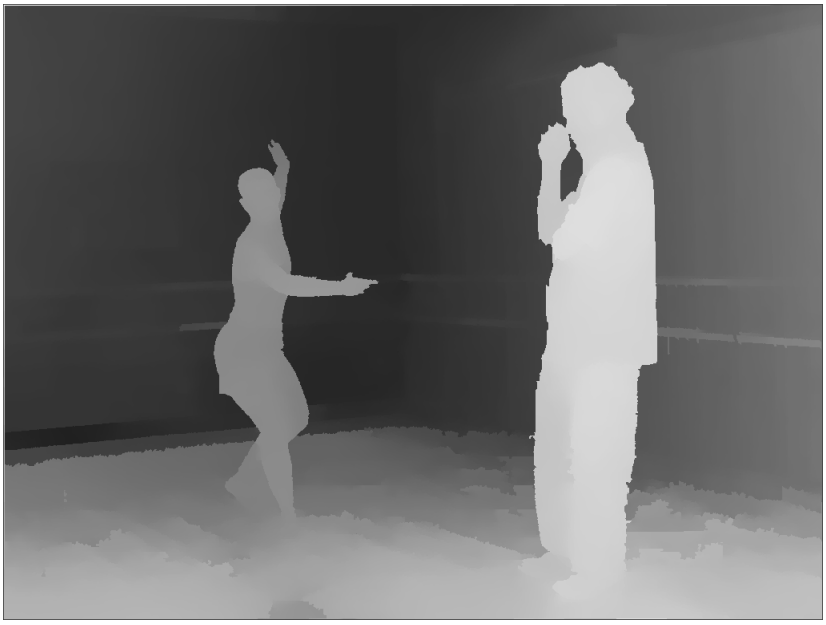


Figura 3.8: Exemplo de um mapa de profundidade.

profundidade para objetos mais próximos, da mesma forma, a fórmula faz com que haja uma resolução de profundidade maior para os menores valores de profundidades. Considerando um mapa de profundidade com $Z_{min} = 3$ e $Z_{max} = 80$, a normalização dele é representada na Figura 3.7. Um exemplo de mapa de profundidade pode ser visto na Figura 3.8.

A nomenclatura *Multiview-plus-Depth* (MVD), em português *multi-vista mais profundidade*, é utilizada para designar dados de sistemas multi-vistas que compreendam a informação das imagens da câmera e de seus mapas de profundidade.

3.3 Compressão de cenas estáticas capturadas por câmera única

A captura de cenas estáticas é explicada na seção 3.1 e é ilustrada na Figura 3.1. Se, por exemplo, uma única câmera percorrer a cena apenas horizontalmente, mantendo sua posição vertical constante, o resultado final é um vídeo em que cada quadro representaria a vista de uma perspectiva diferente da cena, pois a cena é estática. Desta forma, se este vídeo for codificado utilizando o padrão H.264/AVC, a predição do tipo inter-quadros irá utilizar informações de quadros temporalmente adjacentes do vídeo, logo informações de vistas adjacentes para diminuir redundâncias. Assim a compressão de vídeos FTV para cenas estáticas se torna uma compressão multi-vistas. No caso em que a câmera capture a cena pela trajetória horizontal em diversas alturas diferentes, os pontos de vistas capturados pertencem ao plano ilustrado na Figura 1.2 e neste caso, a cada X trajetórias percorridas pela câmera em diferentes altitudes, haverá X vídeos contendo diferentes perspectivas com diferenças horizontais. Para a compressão destes dados, é possível que a predição inter-quadros utilize o próprio vídeo como referência, como utilize os vídeos de alturas diferentes. Desta forma, a predição utiliza as vista horizontais e verticais para predição. Este método é ilustrado na Figura 3.9.

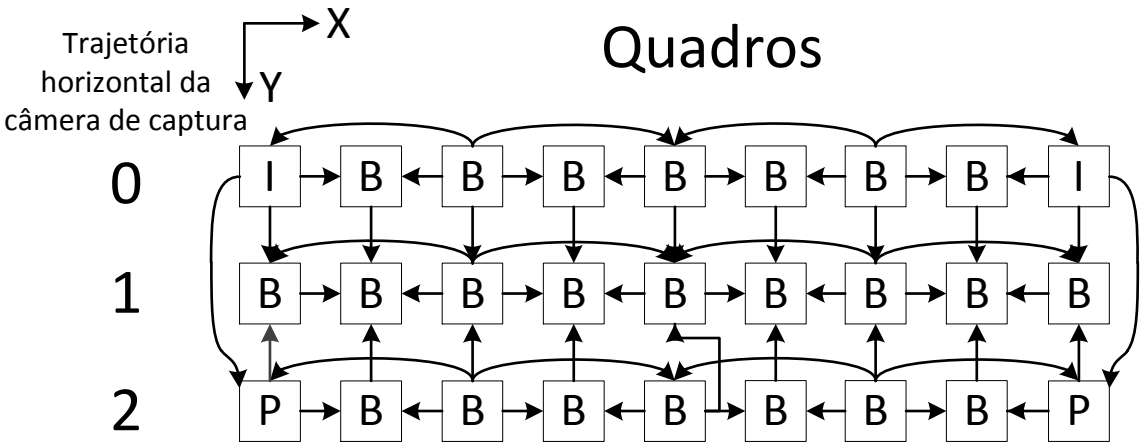


Figura 3.9: Codificação multi-vista. Cada trajetória da câmera, distanticadas verticalmente, captura a cena com diversas perspectivas horizontais, e a predição é feita espacialmente entre as vistas horizontalmente e verticalmente adjacentes. Os quadros são espacialmente distanciados.

Em cenas estáticas animadas, como em animações stop-motion, há também o fator temporal. Neste caso, a cena é capturada da forma explicada anteriormente a cada quadro de animação, que é quando a cena é alterada para criar a animação temporal, assim capturando todos os ângulos dessas cenas. Na compressão, a diferença será que, além da predição do tipo inter multi-vista, cada quadro de animação também utilizará os quadros de animação anteriores para a predição. Isto faz com que a codificação de cenas estáticas seja idêntica a codificação de sistemas multi-vistas, como explicado na seção 3.2.1. A diferença no resultado final do vídeo codificado está no fato de que, com o tipo de captura empregado nas cenas estáticas, há um número maior de vistas capturadas (o que permite a criação de um sistema FTV sem síntese de vista). Já no sistema que utiliza diversas câmeras para a captura da cena, é necessário a síntese de vista para interpolar as vistas entre as câmeras, a menos que os espaçamentos entre as câmeras sejam muito pequenos. Desta

forma, o resultado final dos dois cenários mencionados, de gravação da cena com câmera única ou com câmera múltipla, são similares.

Capítulo 4

Síntese de Vista em Sistemas Multivistas

4.1 Introdução

A síntese de vista consiste na síntese da imagem que representa um ponto de vista de uma cena. A Figura 4.1 exemplifica uma vista sintetizada a partir de duas câmeras adjacentes.

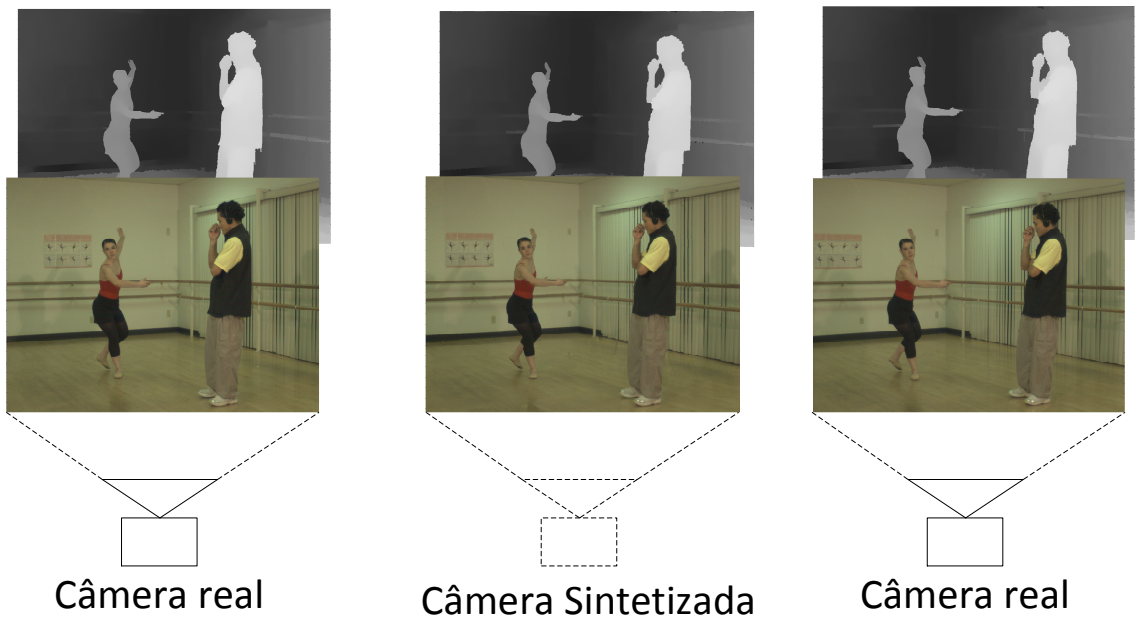


Figura 4.1: Exemplo de uma vista sintetizada utilizando informações de duas câmeras reais e adjacentes.

Para a síntese, é necessário que cada câmera utilizada na síntese possua um mapa de profundidade, ou seja, que a profundidade de cada pixel seja conhecida. Com isto, é possível determinar a posição dos pixels de cada câmera em um sistema de referência global tridimensional. Assim, pode-se projetar os pixels desse sistema 3D à esta nova câmera virtual, gerando uma imagem

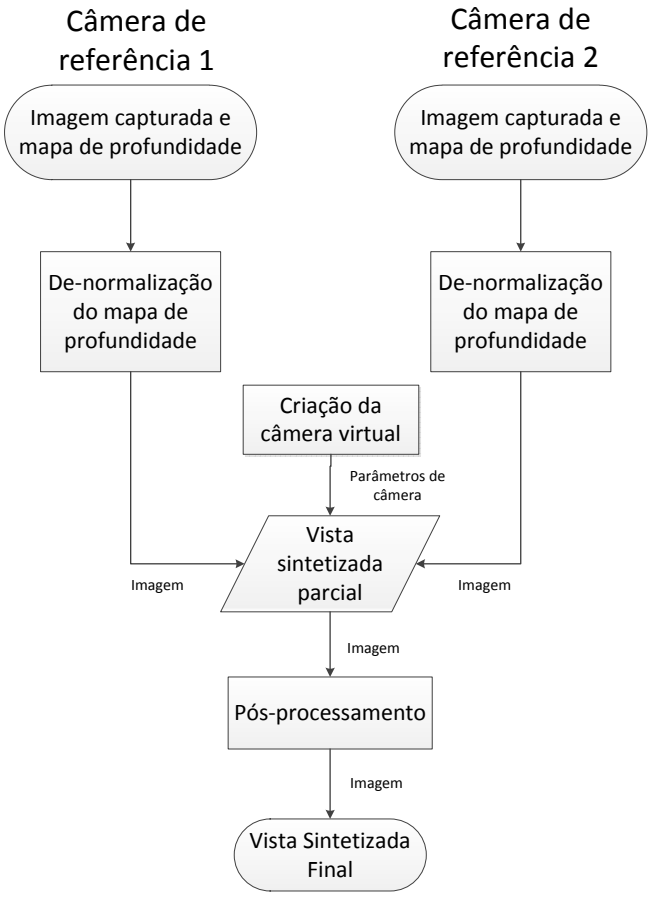
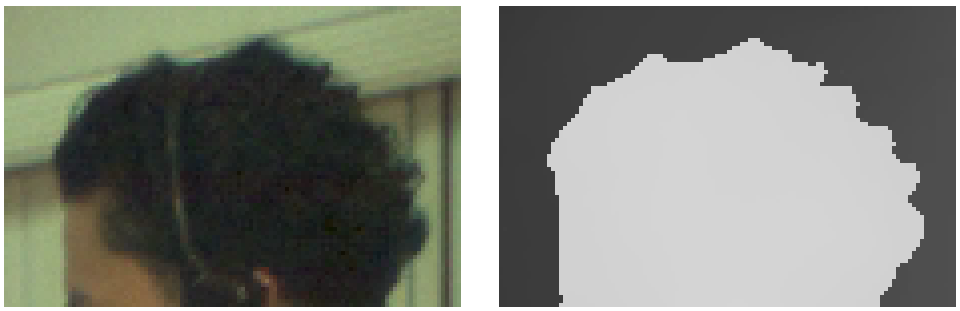


Figura 4.2: Etapas básicas da síntese de vista.

sintetizada.

A síntese de vista consiste em três etapas básicas: a de-normalização dos mapas de profundidade, a projeção dos pixels das imagens de referência à imagem a ser sintetizada e o pós processamento. A de-normalização do mapas de profundidade é o processo de transformar os valores normalizados dos mapas de profundidade em valores representando a profundidade de cada pixel. Após esta etapa, é possível a projeção dos pixels das imagens de referência à imagem sintetizada. Por último realiza-se o pós-processamento, removendo ruídos e preenchendo espaços vazios na imagem sintetizada. O diagrama com as etapas básicas da síntese de vista é ilustrado na Figura 4.2.

Um processo complexo da síntese de vista é a estimação de profundidade, que pode ser estimada utilizando informações de câmeras adjacentes com ou sem o auxílio de sensores que detectam a profundidade da cena [8, 12]. Nessa estimação, há diversos erros nos mapas gerados que acarretam em artefatos na imagem final da síntese. As Figuras 4.3(a) e 4.3(b) ilustram algums tipos de artefatos gerados na síntese de vista. Pode-se obversar em tal figura uma transição gradual de cor entre o cabelo da pessoa e o fundo da imagem, e tal transição não é refletida no mapa de profundidade. Desta forma, artefatos são gerados na síntese devido a tais erros. Outro problema decorre da diferença no histograma de cores das imagens de referência, então faz-se necessária a calibração de cores afim de diminuir as distorções na imagem sintetizada. Estes e outros problemas



(a) Imagem Capturada (b) Mapa de profundidade

Figura 4.3: Problemas de contornos em mapas de profundidade. Na imagem capturada (a) há uma transição gradual das cores do fundo da cena para o objeto em primeiro plano, o que não ocorre no mapa de profundidade (b) da imagem.

são tratados neste trabalho e são explicados nas seções seguintes.

4.2 Etapas Básicas

4.2.1 Criação da câmera virtual

Em geral, utiliza-se duas vistas de referência para a síntese de uma imagem e este trabalho foca nesta possibilidade, embora possam ser utilizadas diversas câmeras de diferentes vistas para sintetizar uma nova vista. Para a síntese, é necessária a criação de uma câmera virtual, sendo possível que esta seja criada em qualquer posição do sistema 3D com matrizes intrínseca e extrínseca quaisquer (seção 2.2.1.1). Caso a posição e as matrizes dessa câmera não sejam bem determinadas, é possível que oclusões de objetos na cena não sejam preenchidas e que não haja projeção de pixels para determinadas regiões da imagem, em consequência havendo partes da imagem não preenchidas. Para evitar tais problemas, duas câmeras são utilizadas como referência e a câmera virtual é criada no sistema 3D entre elas, como mostrado na Figura 4.4. Desta forma, é necessário para a câmera virtual a criação de suas matrizes instrínseca e extrínseca e estas são interpoladas a partir das matrizes das câmeras de referência. Define-se por λ o parâmetro de distância entre as duas câmeras, com λ pertencente ao intervalo [0,1], onde $\lambda = 0$ situa a câmera virtual na mesma posição da câmera de referência mais a esquerda da cena e $\lambda = 1$ a situa na mesma posição da câmera de referência mais a direita. Desta forma, o parâmetro λ é utilizado para vários tipos de interpolação durante a síntese de imagem. A interpolação das matrizes intrínseca e extrínseca da câmera virtual são feitas da seguinte forma:

$$\mathbf{A}_v = \lambda \mathbf{A}_e + (1 - \lambda)\mathbf{A}_d, \tag{4.1}$$

$$[\mathbf{R}|\mathbf{T}]_v = \lambda [\mathbf{R}|\mathbf{T}]_e + (1 - \lambda)[\mathbf{R}|\mathbf{T}]_d, \tag{4.2}$$

onde $\mathbf{A}$ é a matriz intrínseca, $[\mathbf{R}|\mathbf{T}]$ a matriz extrínseca, o subíndice e refere-se a parâmetros em relação à câmera mais à esquerda da cena e d à câmera mais à direita. Desta forma, os parâmetros

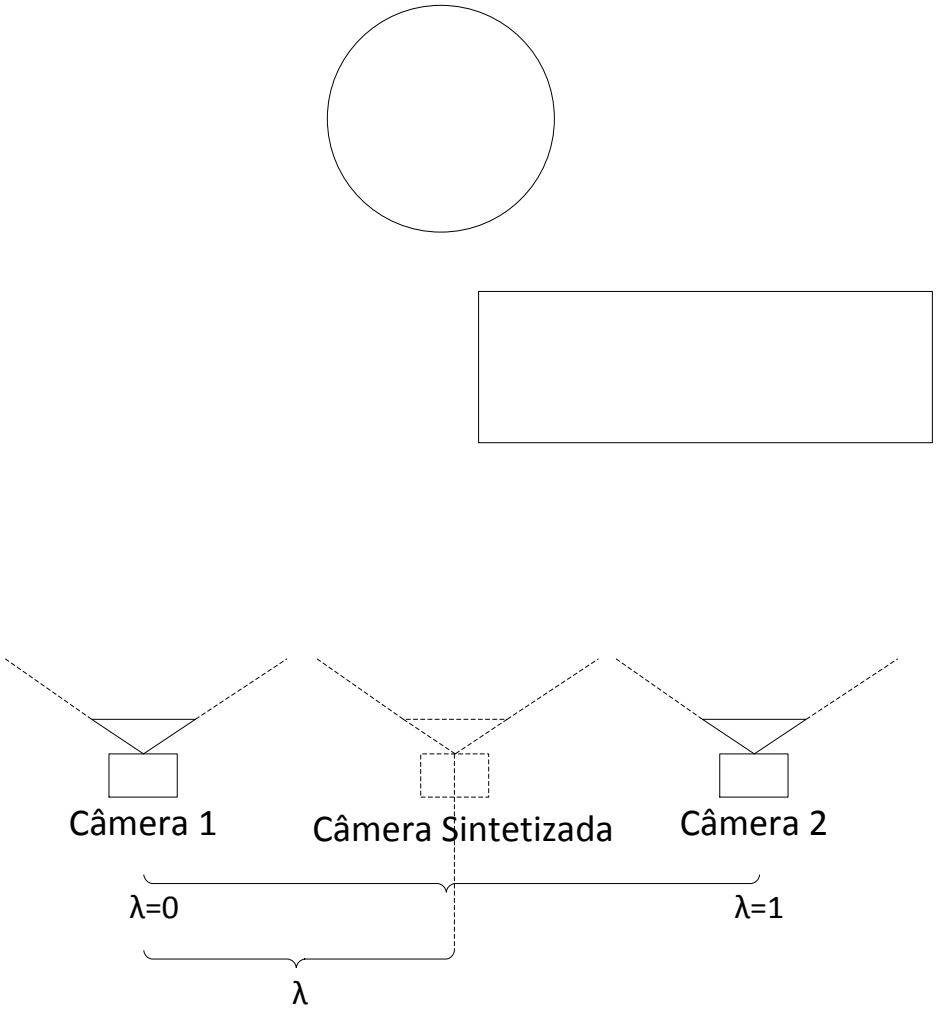


Figura 4.4: Posição de criação de uma câmera virtual. A câmera sintetizada é criada situada entre as câmeras de referência com parâmetro λ, onde $\lambda = 0$ situa aquela na mesma posição da câmera mais à esquerda da cena e $\lambda = 1$ situa na posição da mais à direita.

de translação da matrix extrínseca são interpolados linearmente, logo, se $\lambda = 0.5$, então a câmera virtual é localizada exatamente na metade da distância entre as duas câmeras de referência. Em [16], os autores utilizam interpolação esférica linear para a interpolação da matriz de rotação $\mathbf{R}$, interpolação linear para a matriz de translação $\mathbf{T}$ e para a matriz intrínseca. Neste trabalho, utiliza-se apenas a interpolação linear em vez da interpolação esférica linear.

4.2.2 De-normalização do Mapa de Profundidade

Para possibilitar a projeção dos pixels para a câmera virtual, é preciso obter primeiramente a informação real de profundidade dos pixels de cada câmera de referência. Como explicado na seção 3.2.2, os mapas de profundidade representam a profundidade de cada pixel normalizada em 256 valores. A De-normalização do mapa de profundidade é a transformação desse mapa representado em informações da profundidade Z no sistema de coordenadas de referência. Para isso, utiliza-se a fórmula:

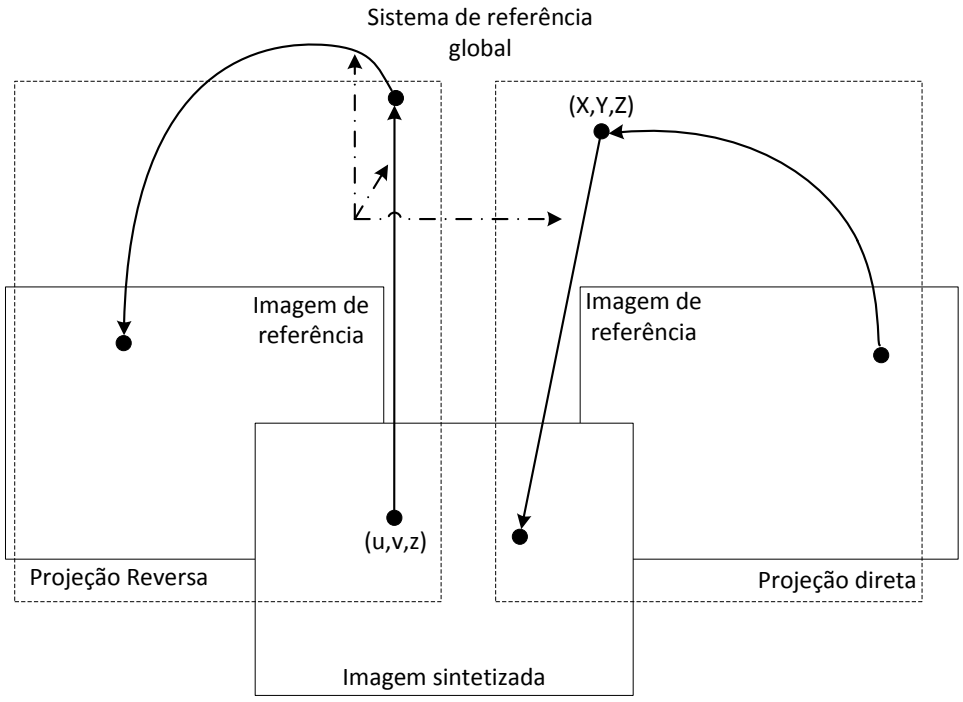


Figura 4.5: Projeção direta, na qual um pixel de uma imagem de referência é projetado na imagem sintetizada, e projeção reversa, na qual um pixel da imagem sintetizada é projetado na imagem de referência.

$$z_{i,j} = \frac{1}{\frac{d_{i,j}}{255}(\frac{1}{Z_{min}} - \frac{1}{Z_{max}}) + \frac{1}{Z_{max}}} \tag{4.3}$$

Na equação, (i, j) representam a posição do ponto na imagem, z é o valor da coordenada Z no sistema de coordenadas, d representa o valor de intensidade do pixel do mapa de profundidade, Z_{min} é a distância mínima e Z_{max} a distância máxima de um pixel do mapa de profundidade até a câmera.

Com a de-normalização dos mapas de profundidade, é possível fazer a projeção dos pixels à câmera virtual.

4.2.3 Projeção de pixels

A projeção dos pixels para a câmera virtual é feita em dois passos. Primeiro os pixels das imagens de referência são projetados para o sistema de coordenadas 3D e por último eles são projetados deste sistema para o plano de imagem da câmera virtual. Este método de projeção é chamado de projeção direta, em que a projeção é feita na direção da imagem de referência para a imagem sintetizada. Há também a projeção reversa em que o mapa de profundidade da imagem sintetizada é projetado no sistema de coordenada 3D e após isto é projetado para a imagem de referência. Este último procedimento é utilizado para o preenchimento de espaços vazios e necessita que já exista o mapa de profundidade para a câmera virtual, logo não é possível utilizá-lo nesta etapa da síntese devido a inexistência dele. Os métodos de projeção são ilustrados na Figura 4.5.

A projeção de um pixel de uma determinada câmera r de coordenadas na imagem $\mathbf{x} = \begin{pmatrix} u & v & 1 \end{pmatrix}^T$ é dada por:

$$\mathbf{x} = \mathbf{P}_r \mathbf{X} \tag{4.4}$$

Em que $\mathbf{X} = \begin{pmatrix} X & Y & Z \end{pmatrix}^T$ é a coordenada do pixel no sistema de coordenadas global e $\mathbf{P}_r$ é a matriz de projeção da câmera.

Desta forma, a projeção do sistema de coordenadas para uma câmera v com matriz de projeção $\mathbf{P}_v$ é dada por:

$$\mathbf{X} = \mathbf{P}_v^{-1} \mathbf{x} \tag{4.5}$$

Logo, a projeção, tanto direta quanto reversa, pode ser descrita por:

$$\mathbf{x} = \mathbf{P}_r \mathbf{P}_v^{-1} \mathbf{X} \tag{4.6}$$

Onde o subíndice r é referente à câmera para o qual o pixel está sendo projetado e o subíndice v é referente à câmera de projeção do pixel.

Para a etapa de projeção, primeiramente projeta-se de forma direta os pixels da imagem da vista esquerda para a câmera virtual. Em seguida, os pixels da outra câmera de referência são projetados. Assim, para uma determinada posição de pixel da câmera virtual, há as seguintes possibilidades de projeção.

Não há projeção de pixels naquela posição. Pode ser ocasionado por oclusão nas duas vistas. Buracos são gerados na imagem sintetizada, pois não há informação de pixels naquela posição.

1 ou mais pixels são projetados de apenas uma vista. Isto ocorre devido a oclusões. Como exemplo, um pixel do fundo da cena pode ser projetado na mesma posição de um pixel de um objeto em frente à câmera. O pixel a ser copiado naquela posição é o de menor distância em relação à câmera virtual.

1 ou mais pixels de diferentes vistas. Nesta possibilidade, diferentes pixels das vistas de referência podem ser projetados na mesma posição. O pixel de cada vista a ser inserido na posição é o de menor distância em relação a câmera virtual. Caso eles possuam distância muito próxima em relação à câmera virtual, significa que os pixels de cada cena pertencem ao mesmo objeto. Assim, o pixel a ser utilizado na imagem virtual será a interpolação dos dois pixels.

$$I_v(u,v) = \begin{cases} I_e(u,v) & \text{se } z(u,v) \text{ não existe ou } z_{e,p}(u,v) < z(u,v) \\ I_d(u,v) & \text{se } z(u,v) \text{ não existe ou } z_{d,p}(u,v) < z(u,v) \\ \lambda I_e(u,v) + (1-\lambda) I_d(u,v) & \text{se } z_{e,p}(u,v),\ z_{d,p}(u,v) \text{ existem} \\ & \text{e } |z_{e,p}(u,v) - z_{d,p}(u,v)| < e \end{cases} \tag{4.7}$$

Assim, para um pixel de posição u, v e profundidade $z(u, v)$ em uma determinada posição da imagem virtual, se houver uma nova projeção nessa posição que tenha profundidade z_λ, então o pixel a ser utilizado na imagem virtual é descrito pela equação 4.7. Nela, I representa a intensidade do pixel, (u, v) a posição do pixel na imagem e e é um limiar de baixo valor para identificar se duas profundidades podem ser consideradas iguais. O subíndice p significa que o pixel da imagem de referência foi projetado à imagem virtual e os subíndices v, e e d são referentes a imagem virtual, da vista equerda e da direita, respectivamente. Esta etapa de decisão do valor do pixel após a projeção é feito tanto para a projeção direta quanto para a projeção reversa.

As etapas da projeção de pixels na câmera virtual são mostradas na Figura 4.6.

4.2.4 Preenchimento de espaços vazios

Após a etapa de projeção, restam espaços vazios na imagem sintetizada onde pixels não foram projetados. Desta forma, é necessário preenchê-los utilizando interpolação de pixels. Existem vários métodos de interpolação, como a utilização do vizinho mais próximo do pixel a ser interpolado (*nearest neighbor*), interpolação linear, bilinear ou convolução cúbica [27]. Em [18], a autora utiliza interpolação linear de pixels para preencher espaços vazios na imagem sintetizada, assim como neste trabalho. Essa interpolação foi escolhida por se tratar de um método de baixa complexidade computacional.

A interpolação linear I_{interp} para um determinado pixel é calculada sendo a média da intensidade dos pixels vizinhos ponderada pela distância, onde o peso w é a distância em número de pixels até o pixel a ser interpolado e N é o número total de pixels a serem utilizados pela interpolação.

$$I_{interp}(u, v) = \frac{\sum_k^N w_k(u_k, v_k) I_k(u_k, v_k)}{\sum_k^N w_k(u_k, v_k)} \tag{4.8}$$

$$w_k(u_k, v_k) = (1 + |u_k - u|)(1 + |v_k - v|) \tag{4.9}$$

4.2.5 Resultado

A imagem sintetizada utilizando as etapas básicas pode ser vista na Figura 4.7. Na imagem final, podemos perceber vários artefatos devido às más projeções, mostradas na Figura 4.8, diferenças de iluminação nas imagens de referência e erros no mapa de profundidade. Como exemplo, há erros de contornos em torno da bailarina e do espectador, criando fantasma devido ao problema de contornos já mencionados. Embora estes erros sejam devido em sua maioria a problemas de casamento entre o mapa de profundidade e a imagem capturada, pois a imagem possui contornos borrados pela abertura dos sensores e a profundidade não pode ser borrada por motivos óbvios, então são necessários processos de tratamento de imagem para a diminuição de tais artefatos, que são apresentadas na seção 4.3.

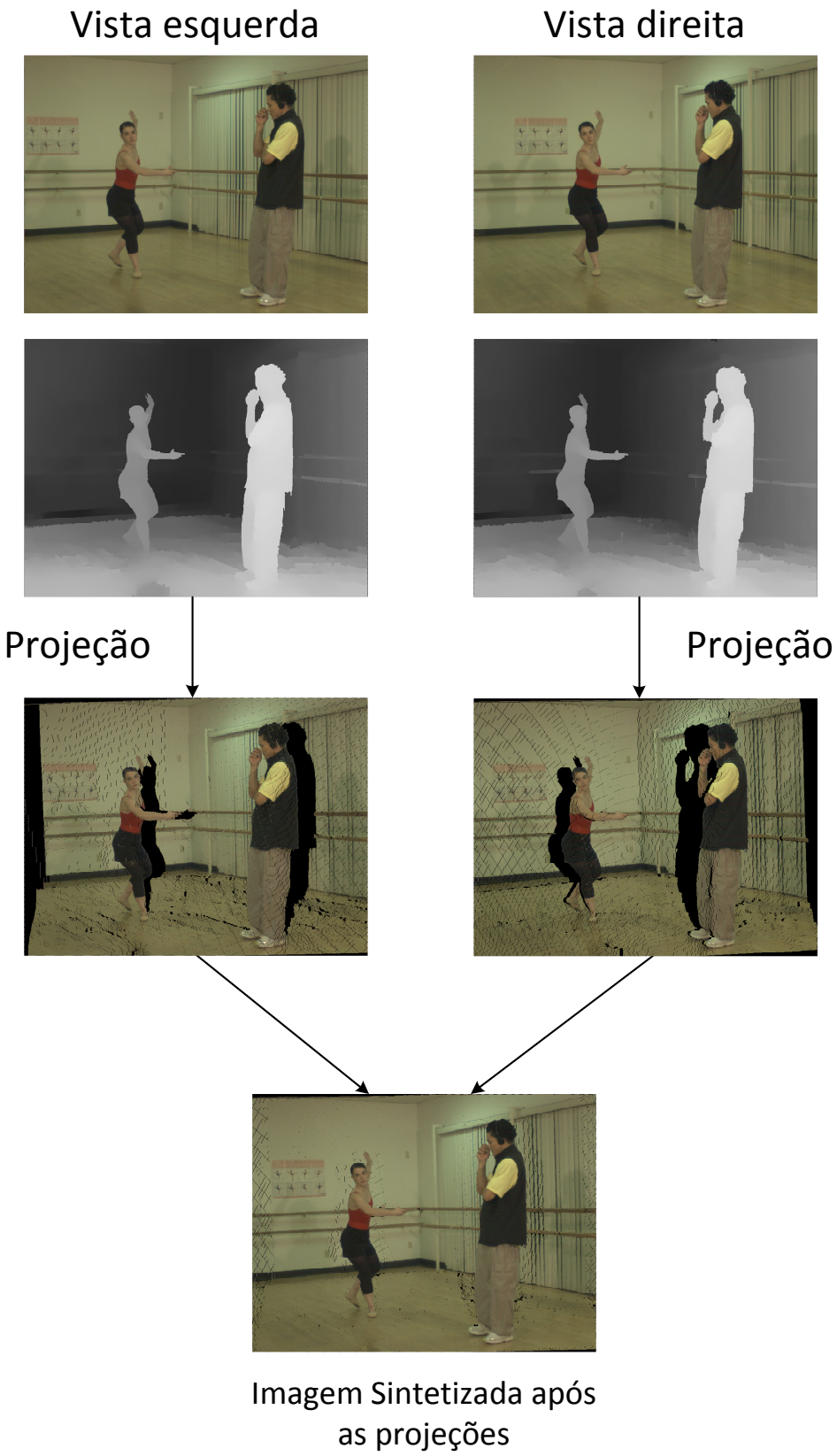
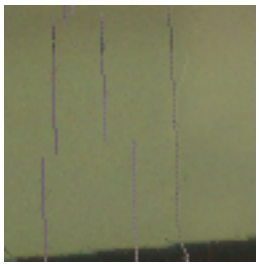
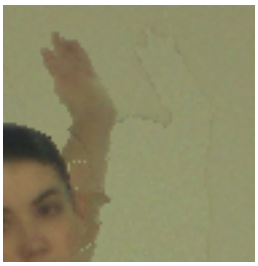


Figura 4.6: Projeção simples de pixels à uma câmera virtual. Os pixels das vistas esquerda e direita são projetados à câmera virtual, formando a imagem sintetizada.

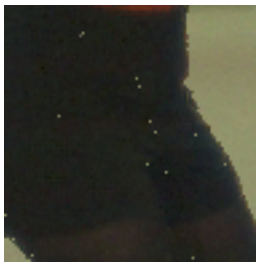
Figura 4.7: Imagem sintetizada final.



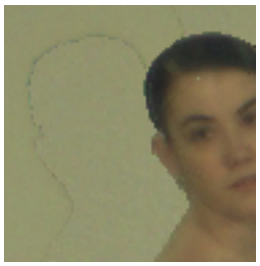
(a) Mal casamento entre o mapa de profundidade e a imagem capturada.



(b) Fantasmas devido a diferença de iluminação entre as imagens de referência.



(c) Erros no mapa de profundidade.



(d) Fantasmas gerados na imagem sintetizada.

Figura 4.8: Artefatos criados na imagem sintetizada devido a erros no mapa de profundidade.

4.3 Implementação

Diversas melhorias são necessárias no sintetizador para torná-lo robusto contra a geração de erros que possam diminuir a qualidade visual da imagem sintetizada. Este trabalho utiliza como base os sintetizadores de vista desenvolvidos em [18] e [16]. São propostas três etapas para suprimir alguns tipos de erros: tratamento de contorno, detecção de erros no domínio do mapa de profundidade e interpolação de pixels por projeção reversa.

Como mencionado anteriormente, há diferenças no histograma de cores das câmeras. Como

primeiro processo no sintetizador de vistas, é necessária a correção de luminância das imagens de referência para evitar distorções de cores na imagem sintetizada. O processo de correção é explicado na Seção 4.3.1.

Em seguida, ocorre a fase de projeção dos pixels à câmera virtual. Nesta etapa, ocorrerá o tratamento de contornos para evitar fanstamas como na Figura 4.8(d). Em [16] é feita a projeção dos pixels utilizando camadas, assim evitando a criação de fantasmas. Neste trabalho, é feita primeiramente a projeção dos pixels fora da região de contorno das imagens de referência. Em seguida há a detecção de erros, além da correção e preenchimento de espaços vazios por projeção reversa - explicado na seção 4.3.3. Na sequência, é feita a projeção dos pixels das regiões dos contornos das imagens de referência. Este método é útil pois cria-se um mapa verdade para a projeção dos pixels nas regiões de contorno, evitando a criação de fantasmas e sem ocorrer o degradamento dos contornos da imagem, que embora sutil, ocorre no método utilizado em [16]. O método utilizado neste trabalho é explicado em detalhes na seção 4.3.2.

Após a projeção dos pixels é feito o preenchimento dos espaços vazios restantes por interpolação. Neste trabalho, é utilizada a interpolação linear, escolhida devido à simplicidade de implementação. Um método mais robusto é discutido em [17], em que o *in-paint* é feito para preenchimento dos buracos da imagem levando em consideração a informação do mapa de profundidade. Neste caso, os buracos tendem a ser preenchidos com o fundo da cena, em vez de mesclarem informações do fundo da cena com informações do primeiro plano para a interpolação de pixels, assim diminuindo a criação de erros e artefatos devido a interpolação de pixels. Por último, é utilizado um filtro passa-baixas nos contornos da imagem sintetizada para uma transição gradual entre camadas da cena, este processo sendo relatado na seção 4.3.4. O diagrama de blocos do sintetizador de vistas completo pode ser visto na Figura 4.9.

4.3.1 Correção de Luminância

Uma cena em um sistema multi-vista é capturada por câmeras com calibrações diferentes, desta forma, as imagens de cada uma possuem diferenças em seus histogramas de cor. Caso não haja um pré-processamento, a vista sintetizada poderá ter distorções nas cores devido a mistura de pixels de diferentes distribuições de luminância e cor, como pode ser visto na Figura 4.10.

O método utilizado neste trabalho para correção de luminância foi proposto em [21]. Neste, considera-se um vídeo de uma cena externa na qual a iluminação global da cena varia a cada quadro, havendo quadros escuros e outros claros no decorrer do vídeo, e deseja-se corrigir a iluminação do vídeo de forma que esta se mantenha estável. Para isto, a média e variância da intensidade de luminância de cada quadro é corrigida para valores arbitrários pré-estabelecidos, sendo a correção sendo feita por regiões. Na correção de luminância para síntese de vista [18], pode-se transformar a distribuição de luminância de uma imagem de referência a compatibilizar com a média e variância da outra.

Considerando uma imagem I com um certa distribuição de intensidade de pixels de média μ e variância σ^2, é possível transformar esta distribuição para uma de média e variância arbitrária μ_0

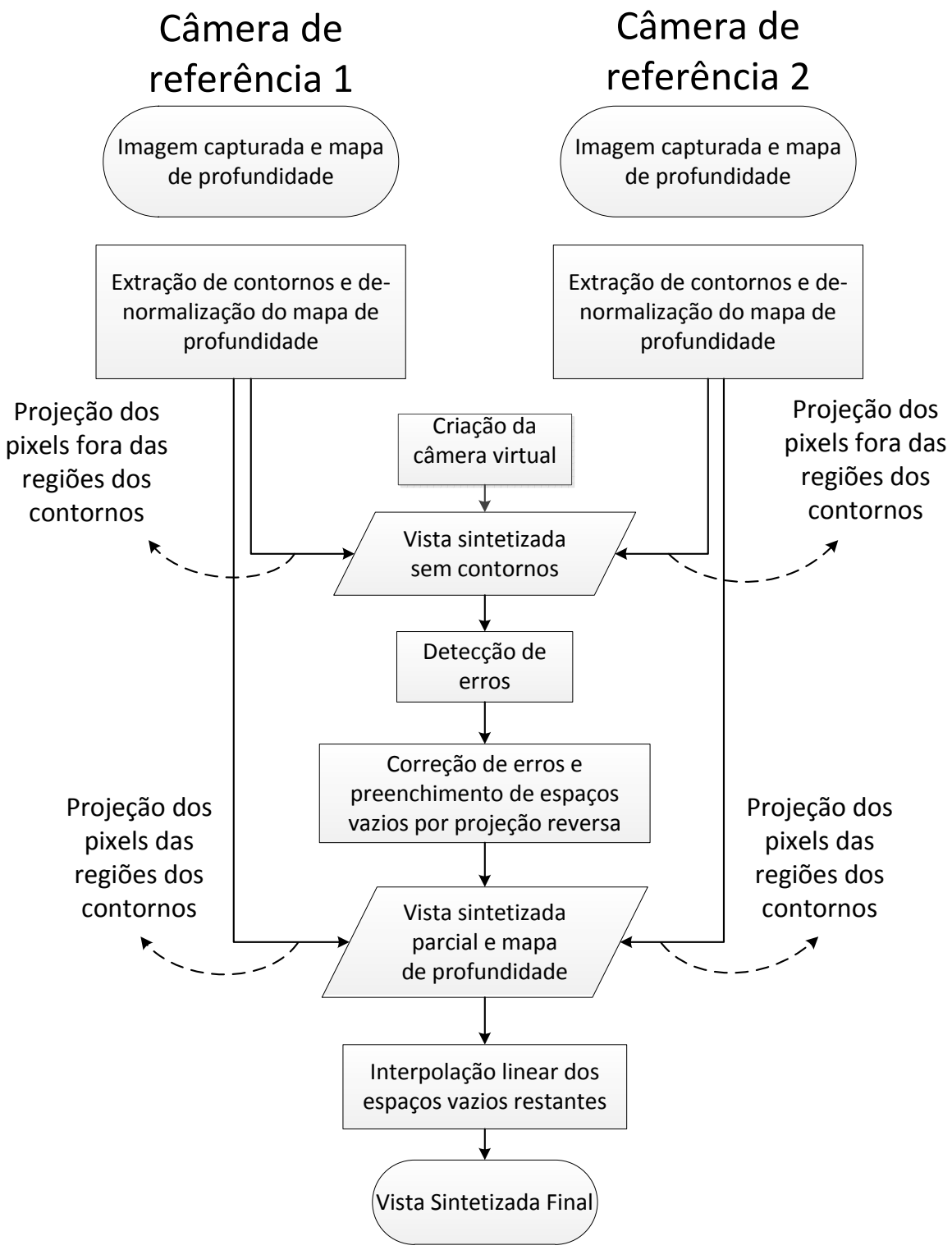
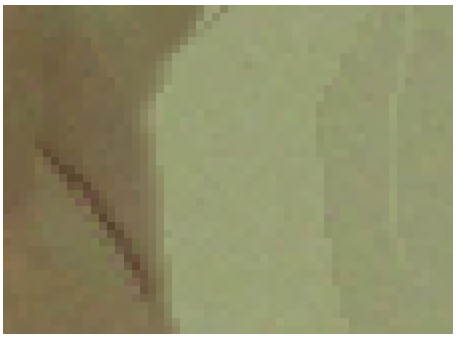


Figura 4.9: Esquemático de um sintetizador de vistas.

e σ_0^2 por meio da seguinte equação:

$$\dot{I} = (I - \mu)\frac{\sigma_0}{\sigma} + \mu_0, \tag{4.10}$$

(a)



(b)

Figura 4.10: Vista sintetizada sem correção de luminância. (a) Vista sintetizada. (b) Aproximação da área destacada em (a).

onde $\dot{I}$ é a imagem com distribuição de pixels alterada. Esta equação pode ser aplicada para distribuições gaussianas. Com a equação 4.10, é possível corrigir a distribuição de luminância de duas imagens de diferentes distribuições. Embora em [21] a correção tenha sido feita por regiões, a correção de luminância deste trabalho considera a média e variância das imagens inteiras por simplicidade de implementação. Assim, é calculada a média e variância das imagens das duas vistas, uma média μ_0 e variância σ_0 são interpoladas linearmente baseadas naquelas e em seguida suas distribuições de luminância são corrigidas para estes parâmetros interpolados. A interpolação é feita pelas fórmulas a seguir.

$$\mu_0 = \lambda\mu_e + (1-\lambda)\mu_d, \tag{4.11}$$

$$\sigma_0 = \lambda\sigma_e + (1-\lambda)\sigma_d, \tag{4.12}$$

onde μ_e e σ_e são a média e variância da luminância da imagem da vista esquerda e μ_d e σ_d da

(a)

(b)

Figura 4.11: Vista sintetizada com correção de luminância. (a) Vista sintetizada. (b) Aproximação da área destacada em (a).

vista direita.

A Figura 4.11 mostra a síntese de vista com correção de luminância das duas imagens, na qual já houve correção dos fantasmas referentes aos problemas de iluminação.

4.3.2 Tratamento de contornos

Os erros nos mapas de profundidade causam vários artefatos na imagem sintetizada final, assim, é de extrema importância um método que diminua tais efeitos. Em [29] e [16] os autores utilizam projeção de pixels por camadas. As regiões de contornos da imagem, regiões em que há descontinuidades nos valores de profundidade da cena, são as de maiores inconsistências nos mapas de profundidade. Os autores utilizaram a segmentação dos mapas de profundidade formando camadas que agrupam intervalos de profundidade e a projeção de pixels é feita por camada.

Neste trabalho, utiliza-se também a segmentação dos mapas de profundidade das câmeras de referência de modo a distinguir as regiões que possuem inconsistências. O objetivo é projetar os pixels das duas imagens de referência identificando quais projeções são inconsistentes e mantendo as projeções corretas. Para isso, primeiro identifica-se as regiões de inconsistências, extraindo os contornos dos mapas de profundidade das câmeras de referência utilizando o método Canny, assim, a projeção é feita em cinco etapas: (i) a projeção dos pixels das imagens de referência fora das regiões de contorno, pois são áreas com informações de profundidade consistentes; (ii) o preenchimento de espaços vazios e correções de artefatos na imagem sintetizada; (iii) a criação do mapa de profundidade da imagem sintetizada; (iv) a extração de contorno do mapa de profundidade da vista sintetizada utilizando o método de Canny; e por último, (v) a projeção dos pixels das regiões de contorno das imagens de referências para as regiões de contorno da imagem sintetizada.

Como primeiro passo é necessário extrair os contornos das imagens de referência. Isto é feito utilizando o filtro Canny de detecção de contornos, mostrado na seção 2.2.2.2, nos mapas de profundidade para segmentar as regiões de descontinuidade onde há inconsistências e criar máscaras que indiquem tais regiões. Nos experimentos realizados neste trabalho se utilizou um limiar de 100 para o filtro Canny. Em seguida, foi feita a dilatação, de acordo com a seção 2.2.2.3, para aumentar as regiões que incluem as inconsistências de forma a diminuir ainda mais a criação de artefatos.

Com as máscaras definidas, a projeção dos pixels fora das áreas de contorno são projetadas, evitando a criação de artefatos, mas em contrapartida perdendo informações dos contornos na imagem sintetizada, como pode ser vista na Figura 4.12. Após este passo, é feito o preenchimento de espaços vazios desta imagem utilizando projeção reversa, explicada na seção 4.3.3, podendo ser vista na Figura 4.13. Com os espaços vazios da imagem virtual preenchidos, utiliza-se o filtro Canny no mapa de profundidade da imagem virtual, detectando as regiões de contorno dela e criando uma nova máscara. Esta é considerada o mapa verdade para as projeções dos contornos das imagens de referência, então estes contornos são projetados para a imagem virtual e só serão utilizados caso sejam projetados dentro da região de contorno da imagem virtual, definido por sua máscara. A Figura 4.14 mostra a imagem virtual após a recuperação de contornos. Desta forma, a informação dos contornos da imagem foi utilizada sem a criação de artefatos devido as más projeções. O processo pode ser visto na Figura 4.15.

4.3.3 Preenchimento por projeção reversa

A interpolação de pixels utiliza informações dos pixels vizinhos para gerar uma nova informação e designa-lá a um determinado pixel. A síntese de vista utiliza informações de câmeras reais para sintetizar uma vista virtual, que não existe. É interessante utilizar as informações reais das câmeras de referência o máximo possível antes de utilizar interpolação de cores na imagem sintetizada, para isso, é proposto por este trabalho o processo de utilização da interpolação de profundidade associada com projeção reversa, explicada na seção 4.2.3, para a interpolação de pixels.

Após a etapa da projeção dos pixels fora das regiões de contorno há a etapa de detecção de erros. Após isso, há a correção deles e o preenchimento dos espaços vazios, sendo estas duas feitas

Figura 4.12: Imagem virtual após a projeção dos pixels fora das regiões onde há contorno.

Figura 4.13: Imagem virtual após o preenchimento de espaços vazios por projeção reversa.

por projeção reversa, assim as amostras das imagens de referência são coletadas para preencher e corrigir a imagem sintetizada.

Para este processo será criado uma máscara M contendo informações de quais pixels serão interpolados, $M(u, v)$ podendo ser 1, caso o pixel seja considerado para a interpolação, ou 0, caso contrário. Neste trabalho, a detecção de erros e a escolha dos pixels a serem interpolados são realizadas por meio da verificação do mapa de profundidade. Após a primeira projeção, o mapa de profundidade da vista sintetizada está preenchido, embora com vários espaços vazios. Os pixels considerados para interpolação são os pixels vazios, os quais estão em uma posição a qual não

Figura 4.14: Imagem virtual após a recuperação dos contornos.

houve projeção de pixels, e os pixels os quais a informação de profundidade distoe da informação de profundidade dos pixels vizinhos. Assim, a cada pixel da imagem, é calculado a média das profundidades dos pixels em sua vizinhança, pela seguinte fórmula:

$$z_m = \frac{1}{9} \sum_{i=-1}^{1} \sum_{j=-1}^{1} z(u+i, v+j) \quad \text{para } i \neq 0 \text{ e } j \neq 0, \tag{4.13}$$

onde z_m é a média da profundidade dos pixels em uma determinada região, (u, v) é a posição do pixel na imagem, i e j é a posição do novo pixel em relação a (u, v).

Sendo $z(u, v)$ a profundidade de um determinado pixel na imagem e τ um limiar arbitrário, caso $|z(u, v) - z_m(u, v)| > \tau$, significa que $z(u, v)$ é uma má projeção e não pertence a região, assim o pixel é apontando como erro e será interpolado por projeção reversa. A princípio, τ deve ser pequeno para não considerar diferenças discrepantes de profundidade sendo iguais e neste trabalho foi utilizado $\tau = 2$. A máscara criada após a projeção de pixels pode ser vista na Figura 4.16.

Como tenta-se evitar a criação de novas intensidades de pixel utilizando interpolação, também deseja-se evitar a criação de novos valores de profundidade. Assim, após a criação da máscara, para cada posição onde $M = 1$ será determinada uma nova profundidade $z^{'}$, em que esta é a mediana dos 8 pixels vizinhos a posição em questão.

$$z^{'}(u, v) = mediana(z(u, v)) \quad \text{para } M(u, v) = 1 \tag{4.14}$$

Tendo o novo valor de profundidade, será feita a projeção reversa do pixel da imagem sintetizada para a imagem de referência, obtendo assim a informação a ser utilizada. Caso algum pixel seja encontrado em uma das imagens de referência, este com profundidade z_{interp}, utiliza-se a equação

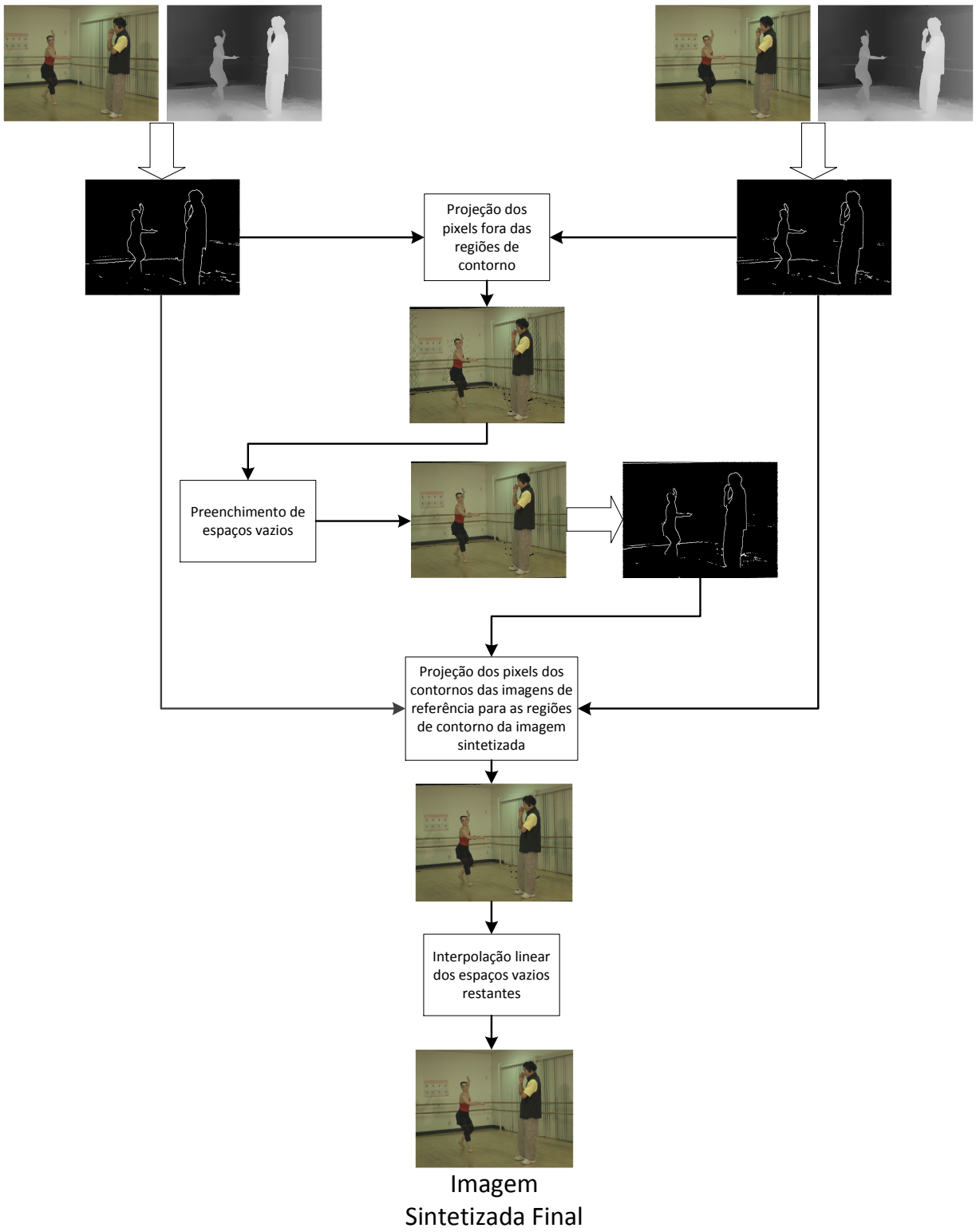


Figura 4.15: Processo do tratamento de contorno na síntese de vista.

4.7 para definir o novo pixel desde que $|z^{'}(u,v) - z_{interp}(u,v)| < e$, ou seja, possam ser consideradas iguais.

A Figura 4.17 mostra o resultado da vista sintetizada após a interpolação por projeção reversa.

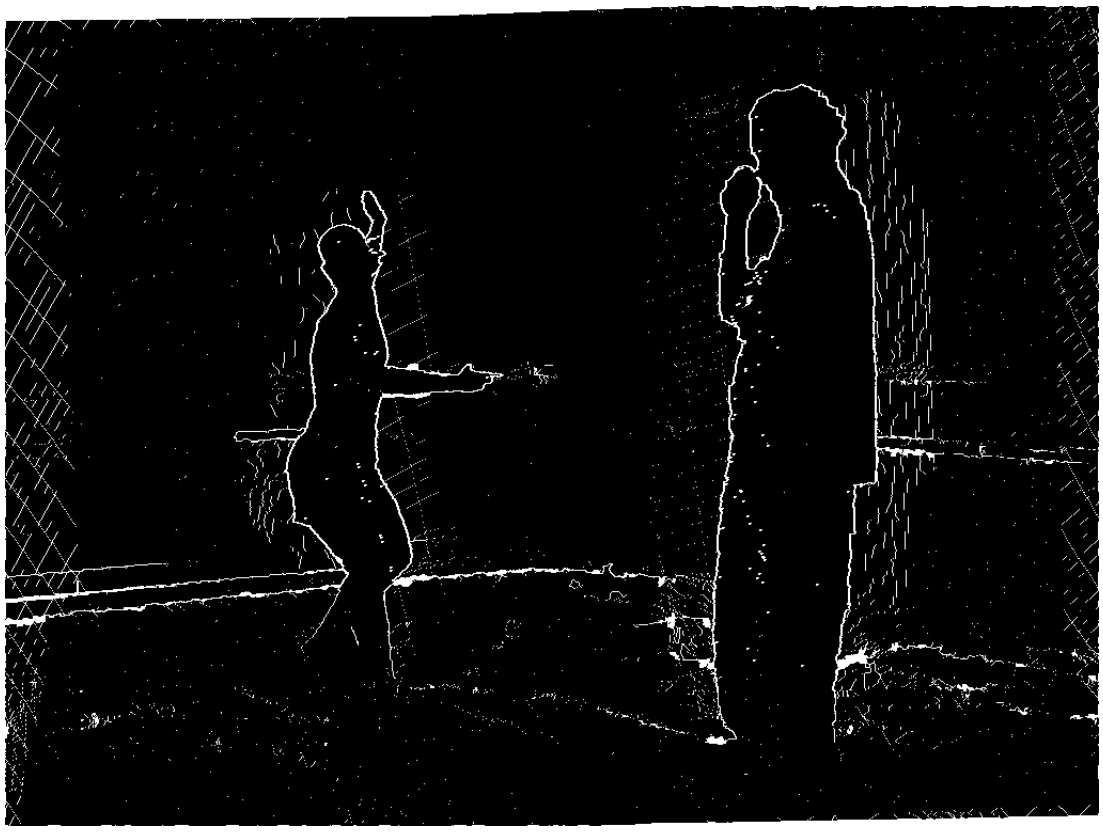


Figura 4.16: Mascára para interpolação de pixels. As regiões brancas da imagem são os pixels considerados para interpolação.

Note que esta etapa faz o preenchimento dos espaços vazios e a correção de erros da Figura 4.12.

A Figura 4.3.3 mostra uma comparação de resultados entre a interpolação linear e interpolação por projeção reversa. Após a interpolação por projeção reversa, ocorre a projeção das regiões onde há contornos nas imagens de referência e, em seguida, caso ainda haja espaços vazios na imagem ou pixels que foram detectados erros mas não cumpriram as condições para a interpolação por projeção reversa, é feita a interpolação linear como explicada na seção 4.2.4.

4.3.4 Suavização de contornos

Em uma imagem capturada por uma câmera, é natural a suave transição de cor entre objetos, como mostrado na Figura 4.3(a). Desta forma, é interessante que a imagem sintetizada também tenha uma suave transição de cor em seus contornos. Para atingir este efeito, utiliza-se a técnica aplicada em [16], que consiste na aplicação de um filtro passa-baixas nas regiões de contorno da imagem sintetizada. Assim, utiliza-se os contornos já obtidos da imagem sintetizada mostrada na seção 4.3.2. Na região onde há contorno, será feita a convolução com um filtro passa-baixas, o resultado da convolução sendo a média dos pixels vizinhos da posição a ser convoluída.

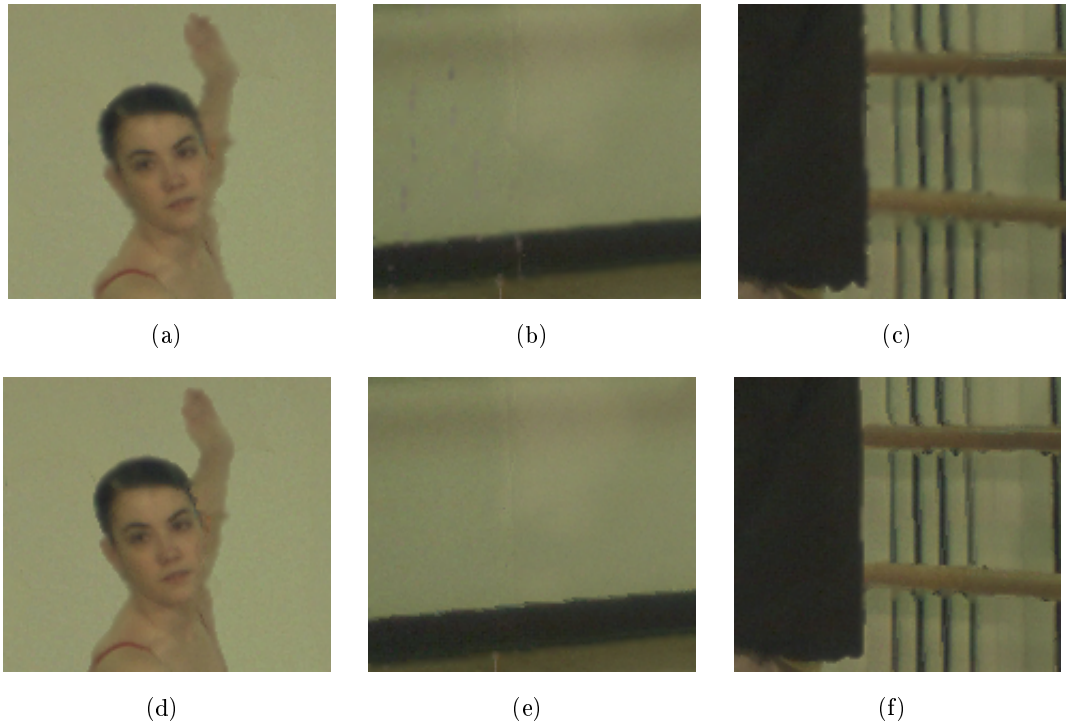
Figura 4.17: Vista sintetizada após a interpolação por projeção reversa.

$$I(u,v) = \frac{1}{9}\sum_{i=-1}^{1}\sum_{j=-1}^{1} I(u+i, v+j) \tag{4.15}$$

A equação 4.15 é referente ao filtro passa-baixas aplicado nos contronos da imagem, onde I representa a intensidade de uma componente de um pixel, e (u,v) é a posição do pixel na imagem. A Figura 4.19 um exemplo de resultado após a etapa de suavização de contorno.

4.3.5 Resultados

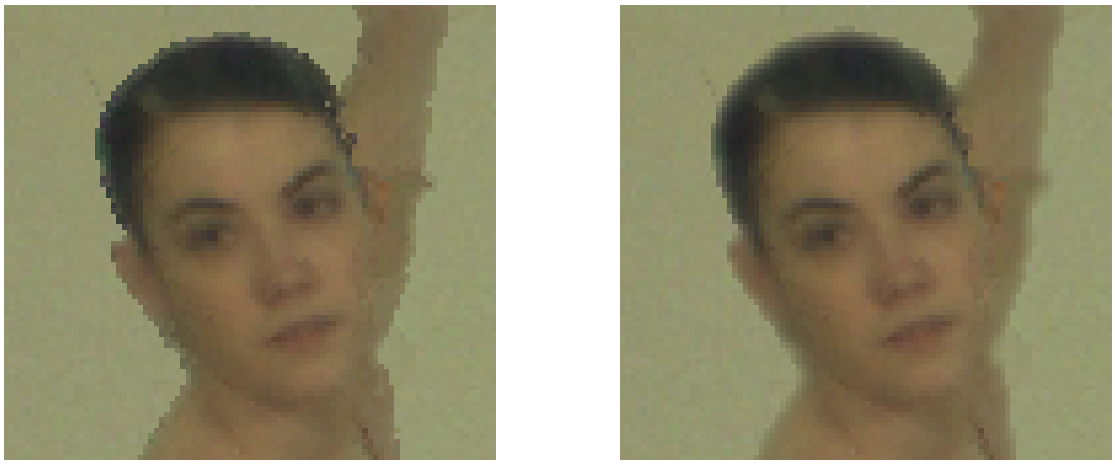
Utilizando as câmeras 3 e 5 da sequência *Ballet* [28] como referência e definindo a câmera virtual com os parâmetros da câmera 4, é possível sintetizá-la para a comparação de resultados, que são mostrados na Figura 4.20. A síntese resulta em uma imagem com alta qualidade visual, embora haja aterfatos nas partes de reflexões do ambiente. Este é um dos grande problemas da síntese de vista por projeção de pixels. Há também erros nos mapas de profundidade que resultam em artefatos como a má projeção do polegar da bailarina. Outros erros ocorrem devido a interpolação linear de pixels, que mescla informações do fundo da cena com informações de primeiro plano, gerando assim pixels com cores discrepantes em relação a cena. Para solucionar este último tipo de problema é necessário um refinamento mais robusto a erros na geração dos mapas de profundidade, ou utilizar o método de *in-paint* baseado no mapa de profundidade [17].



(a) (b) (c)

(d) (e) (f)

Figura 4.18: Comparação de resultado da interpolação linear e interpolação por projeção reversa. As Figuras da fileira (a)-(c) foram processadas apenas com interpolação linear, (d)-(f) por interpolação por projeção reversa e interpolação linear.



(a) Imagem sintetizada sem suavização.

(b) Imagem sintetizada com aplicação do filtro passa-baixas.

Figura 4.19: Suavização de contornos

(a) Imagem capturada pela câmera 4.

(b) Imagem sintetizada da câmera 4, utilizando as câmeras 3 e 5 como referência.

Figura 4.20: Comparação de resultado da síntese de vista.

Capítulo 5

Arquitetura de Sistemas FTV

5.1 Introdução

Para um sistema FTV ser completo, são necessários seis passos: captura, codificação, transmissão e decodificação do conteúdo multi-vistas, além da geração dos mapas de profundidade e síntese de novas vistas.

A geração dos mapas de profundidade e a síntese de vista podem ser feitas no lado do codificador ou no lado do decodificador. Caso a estimação de profundidade e a síntese sejam realizadas no codificador, é necessário um canal de retorno entre codificador e decodificador com a informação de qual vista está sendo exibida ao usuário e, assim, pode-se transmitir apenas a vista sintetizada. Caso a síntese seja feita no decodificador, há duas possibilidades diferentes. Na primeira, os mapas de profundidade são gerados no codificador e transmitidos juntos com as imagens das vistas capturadas. A segunda possibilidade é a geração dos mapas de profundidade e a síntese de vista no decodificador, em que as imagens das vistas capturadas são transmitidas. Logo, as 3 possibilidades para um sistema FTV em relação a geração das novas vistas e suas transmissões são:

Cenário (A) Geração dos mapas de profundidade e síntese de vista no codificador com transmissão da vista sintetizada. É necessário um canal de retorno entre codificador e decodificador.

Cenário (B) Geração dos mapas de profundidade no codificador e síntese de vista no decodificador, sendo necessária a transmissão das imagens das vistas de referência e de seus mapas de profundidade.

Cenário (C) Geração dos mapas de profundidade e síntese de vista no decodificador, com a transmissão das imagens das vistas de referência.

A Figura 5.1 ilustra essas três posibilidades, que também são detalhadas na Tabela 5.1.

Tabela 5.1: Tabela descritiva das possíveis arquiteturas de um sistema FTV.

	Cenário (A)	**Cenário (B)**	**Cenário (C)**
Geração dos mapas de profundidade	No codificador	No codificador	No decodificador
Síntese de vista	No codificador	No decodificador	No decodificador
Dados de transmissão	Vista sintetizada	Imagens e mapas de profundidade das vistas de referência	Imagens das vistas de referências
Canal de retorno	Sim	Não	Não

O cenário (A) é ideal para sistemas com baixa potência computacional. Desta maneira a geração dos mapas e da vista sintetizada serão feitas no codificador e apenas será necessária a transmissão das vistas sintetizadas. Logo, há uma restrição no número de vistas que podem ser sintetizadas e enviadas ao decodificador devido a limitação da banda de transmissão, mas diminuindo muito a complexidade computacional no decodificador. Já no cenário (B), a geração dos mapas é feita no codificador e a síntese no decodificador. Assim, há a necessidade de transmissão dos mapas de profundidade causando um aumento na banda de transmissão, mas possibilitando ao decodificador a síntese de vista de qualquer ponto de vista de interesse em troca de um aumento na complexidade computacional no decodificador. Por último, no cenário (C), a geração dos mapas e a síntese são feitas no decodificador, havendo apenas a necessidade de transmissão das imagens das vistas de referência, mas sendo necessário uma grande potência computacional no decodificador para a estimação de profundidade.

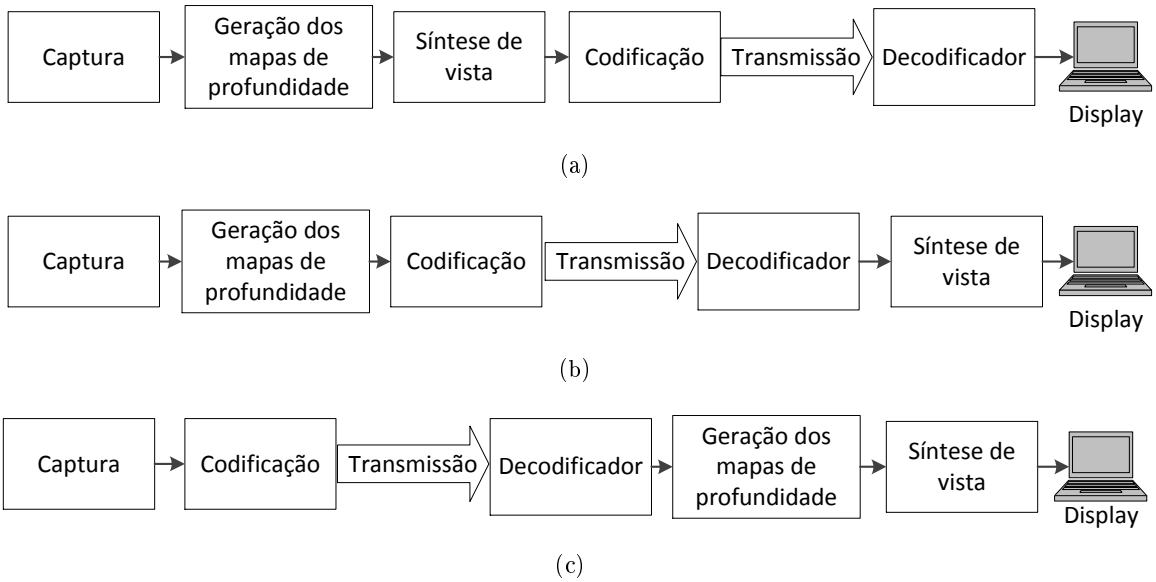


Figura 5.1: Possibilidades de arquitetura de um sistema FTV. (a) Geração dos mapas de profundidade e síntese de vista antes da transmissão; (b) geração dos mapas de profundidade antes da transmissão e síntese de vista após a transmissão; (c) geração dos mapas de profundidade quanto a síntese de vista são feitas após transmissão.

Tabela 5.2: Tabela de estrutura deste capítulo

Seção	Cenários Testados	Métodos de compressão utilizados para o experimento
5.2	(A), (B)	4 métodos de compressão são testado para o cenário (A) e 1 método de compressão para o cenário (B)
5.3	(A), (B), (C)	1 método de compressão é testado para cada cenário.

Neste Capítulo, serão apresentados experimentos de taxa de bits por distorção nos cenários apresentados. A Seção 5.2 apresenta experimentos realizados utilizando os cenários (A) e (B), para determinar o melhor tipo de compressão no cenário (A) fazendo comparações com o cenário (B). Para o cenário (A), em que a síntese de vista é realizada no codificador, serão apresentados quatro métodos de compressão para a vista sintetizada nas Seções 5.2.1.2, 5.2.1.3, 5.2.1.4 e 5.2.1.5. Para o cenário (B), será utilizado apenas um método de codificação (Seção 5.2.2). A Seção 5.3 apresenta uma arquiterua genérica que acomoda os três cenários mencionados e apresenta experimentos realizados nos cenários (A), (B) e (C). A Tabela 5.2 apresenta a estrutura deste capítulo.

5.2 Métodos de Compressão de Vistas Sintetizadas

Nesta Seção, têm-se como objetivo a compressão da vista sintetizada. É utilizada a sequência *Ballet* e os mapas de profundidade são gerados no codificador logo após a captura das vistas. Assim, nesta Seção, considera-se o cenário (A) fazendo comparações com o cenário (B), ambos apresentados na Seção 5.1. É considerado um sistema multi-vista com duas vistas e o foco é a compressão da vista sintetizada. As imagens das duas vistas capturadas são transmitidas nos dois cenários.

Para os testes de compressão, foi utilizado a sequência *Ballet*: duas vistas de referência correspondentes aos vídeos capturados das câmeras 3 e 4 e uma vista sintetizada usando estas duas vistas de referência. A câmera virtual para a síntese é criada situada entre as duas câmeras de referência, ou seja, com $\lambda = 0.5$ (Seção 4.2.1). O objetivo dos testes é comparar a compressão de vistas sintetizadas nos diferentes cenários mencionados. Para isso, foi utilizado o software de referência JMVC com os parâmetros de codificação apresentados na Tabela 5.3. A PSNR obtida nos resultados é a PSNR da vista sintetizada em relação à vista sintetizada em um sistema o qual não há perdas, ou seja, a vista é sintetizada com os vídeos em seus formatos originais, sem compressão.

A Seção 5.2.1 apresenta os métodos de codificação para o cenário (A) e a 5.2.2 para o cenário (B).

Tabela 5.3: Paramêtros de entrada utilizados para codificação de vídeo no software de referêmcia JMVC.

Parâmetro	**Valor**
Controle de Banda	Não
Otimização RD	Sim
Codificador de entropia	CABAC
Janela da região de procura	24

5.2.1 Cenário (A) - Síntese no codificador

Em sistemas com canal de retorno entre codificador e decodificador, é possível realizar a síntese de vista no codificador. Como mencionado na Seção 5.2, consideraremos nesta Seção que as imagens das vistas de referência são transmitidas para o decodificador, então são transmitidas as imagens das vistas de referência e a vista sintetizada. Note que, para o cenário (A), é apenas necessária a transmissão da vista sintetizada. Havendo a transmissão também das vistas de referência, a síntese também pode ser realizada no decodificador caso ocorra a estimação de profundidade, como no cenário (C). Assim, os usuários deste sistema podem tanto requerer a vista sintetizada por um canal de retorno ou podem estimar a profundidade e sintetizar a nova vista utilizando as vistas de referência, então este é um cenário híbrido entre (A) e (C). Como esta Seção pretende investigar a compressão da vista sintetizada, será considerado que o usuário requer a vista sintetizada, então este cenário híbrido será tratado como cenário (A).

As vistas de referência são codificadas por codificação multi-vista (MVC, Seção 3.2.1) e transmitidas para o usuário. O foco desta Seção é otimizar a codificação da vista sintetizada. Esta é sintetizada utilizando as imagens de vistas adjacentes, logo deve ser possível utilizar informações da síntese de vista para auxiliar a compressão e tais informações podem ser utilizadas de diferentes maneiras para auxiliar a compressão. Desta forma, há diferentes métodos de compressão que podem ser empregados na codificação da vista sintetizada, que podem ou não utilizar as informações da síntese de vista.

No caso em que as informações da síntese de vista não são utilizadas para auxiliar a compressão, há duas possibilidades de codificação da vista sintetizada. A primeira é a codificação independente dessa vista utilizando H.264/AVC. A outra é a codificação multi-vista MVC da vista sintetizada e das vistas de referência, aproveitando as correlações entre vistas adjacentes.

A codificação MVC pode ser realizada com codificação padrão H.264/MVC utlizando o codificador JMVC, e pode ser otimizada utilizando informações da síntese de vista. Os métodos de codificação que utilizam tais informações são apresentadas nas Seções 5.2.1.4 e 5.2.1.5. As abordagens de codificação de vista sintetizada com síntese no codificador que são apresentadas nesta Seção são:

- Método (A).1 - Codificação independente da vista sintetizada utilizando H.264/AVC, apresentada na Seção 5.2.1.2.
- Método (A).2 - Codificação multi-vista normal para a vista sintetizada, apresentada na Seção 5.2.1.3.
- Método (A).3 - Codificação utilizando vetores de movimento da síntese de vista para a região de procura, apresentada na Seção 5.2.1.4.
- Método (A).4 - Codificação dos vetores de movimento e resíduos para síntese de vista por blocos, apresentada na Seção 5.2.1.5.

Para o levantamente das curvas RD (*PSNR x taxa de bits*) do sistema simulcast para a comparação de eficiência dos métodos, só é necessário variar a qualidade da vista sintetizada e obter sua taxa, pois ela é codificada independentemente das outras vistas. Para o sistema multi-vista, há diversas possibilidades de codificação pois, como as três vistas são codificadas independentemente, é possível alterar o parâmetro de quantização QP (Seção 2.1.5.2) de cada vista. Para os testes, serão feitos dois tipos de codificação, um no qual as três vistas são codificadas utilizando os mesmos QPs para o levantamento da curva e na outra possibilidade as duas vistas de referência serão codificadas com parâmetros QPs iguais e fixos, enquanto o QP da vista sintetizada, ou seja, sua qualidade, varia. Nestes casos, a taxa de bits considerada é apenas a taxa da vista sintetizada.

5.2.1.1 Obtenção de informações da síntese de vista para codificação multi-vista

A vista sintetizada é criada a partir de duas vistas adjacentes. Logo, é possível utilizar informações da síntese de vista para auxiliar a compressão de vídeo. A utilização da síntese de vista para auxiliar a codificação de vídeos multi-vista já foi realizada em [14]. Nesse artigo, os autores consideram um sistema multi-vista de duas câmeras e os vídeos são codificados por H.264/MVC. Na codificação de vídeo do padrão H.264, um determinado quadro do vídeo é predito a partir de um quadro de referência reconstruído. Nesse trabalho, os autores em vez de utilizaram como referência para as predições inter-quadros entre câmeras os quadros reconstruídos do padrão H.264, eles utilizaram como referência quadros sintetizados por meio da síntese de vista. Os autores conseguiram ganhos consistentes com um aumento de até 3 dB para a PSNR do vídeo codificado em relação a codificação H.264/MVC. Ainda nesse trabalho, os autores utilizaram apenas câmeras reais para teste.

O grupo MPEG já definiu o padrão de codificação eficiente de vídeos multi-vista, sendo este a codificação H.264/MVC (Seção 3.2.1). Logo, não é de nosso interesse alterar o modo de codificação desse padrão, mas utilizar sua estrutura para o nosso objetivo.

Durante a síntese de vista, ocorre a projeção dos pixels das vistas adjacentes até a câmera virtual. Como ponto de partida, a idéia é transmitir informações para o decodificador sobre como sintetizar a nova vista. Durante a projeção dos pixels para a câmera virtual (Seção 4.2.3), é possível, a partir da posição da qual cada pixel da imagem sintetizada foi projetado, determinar os vetores de movimento dos pixels da imagem sintetizada para os pixels das imagens de referência, como

mostrado na Figura 5.2. Com isto, para a transmissão de informações para síntese no codificador, seria necessário a transmissão dos vetores de movimento de cada pixel da imagem sintetizada para as imagens de referência. Codificar um arquivo com as informações dos vetores de movimento de todos os pixels de uma imagem consome uma grande quantidade de bits. Pixels vizinhos da vista sintetizada possuem vetores de movimento próximos pois originaram de uma mesma região da imagem de referência. Assim, é possível agrupar pixels com vetores de movimento iguais para a diminuição da quantidade de informações na codificação. Uma divisão simples da imagem seria em blocos, enviando apenas um vetor de movimento para cada grupo de pixels que forma um bloco, mas neste caso podendo ocasionar artefatos na imagem devido a perda de informações. Este procedimento pode ser realizado utilizando um codificador H.264 e em vez de desenvolver um método de codificação para competir com ele, é interessante utilizá-lo pois ele é extremamente otimizado. Um modo de utilizá-lo é fornecendo os vetores de movimento da síntese de vista para o codificador de vídeo, sendo utilizados durante a estimação de movimento.

A abordagem deste método utiliza os vetores de movimento obtidos na síntese de vista para a codificação da vista sintetizada. Os vetores de movimento extraídos são fornecidos para o codificador JMVC, que os utiliza para determinar a posição da janela de procura, sendo a estimação de movimento feita em torno desses vetores. A decisão do melhor vetor de movimento para cada macrobloco é determinada por RDO (Seção 2.1.7). Embora os vetores de movimentos fornecidos são os de verdadeira origem dos pixels, não é incomum escolher-se outro macrobloco para a predição, como por exemplo o vetor de movimento nulo (0,0), através da RDO.

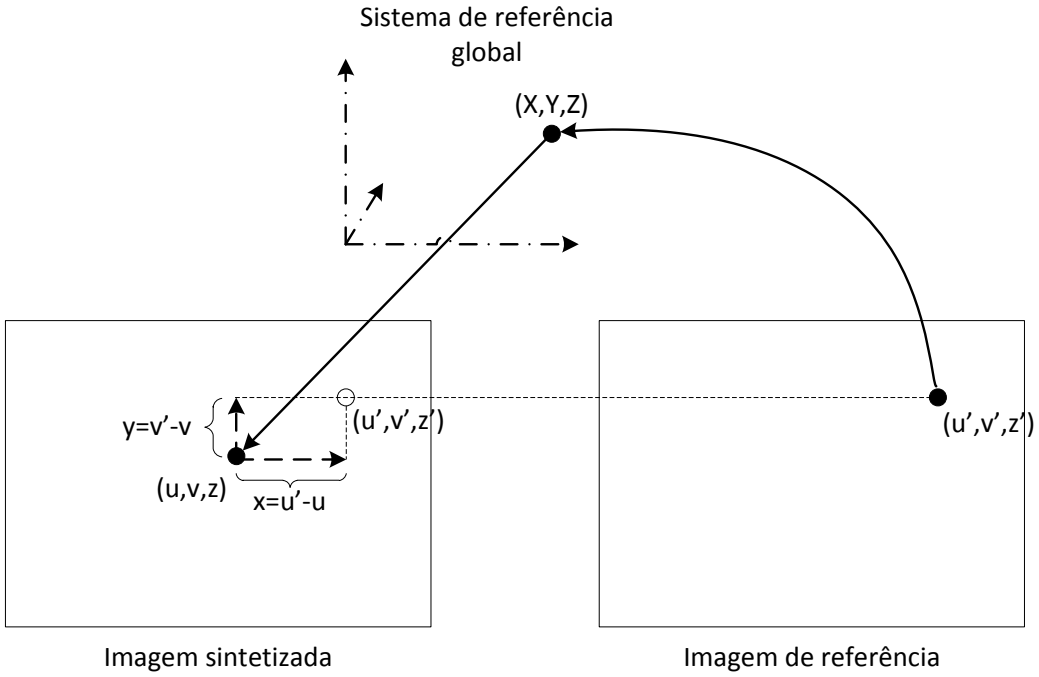


Figura 5.2: Vetores de movimento da imagem sintetizada em relação à vista de referência são obtidos durante a projeção de pixels durante a síntese de vista.

Para a determinação dos vetores de movimento durante síntese de vista, a imagem sintetizada completa é dividida em blocos de 16×16, 8×8 e 4×4 pixels. Para cada bloco B de $N \times N$, há N^2

pixels e como cada pixel possui um vetor de movimento para a imagem de referência, então há N^2 vetores de movimento **mv**. Para determinar o vetor de movimento de um macrobloco X, é calculada a SAD desse em relação aos macroblocos fornecidos pela projeção dos vetores de movimento **mv** contidos no macrobloco X à imagem de referência. O vetor de movimento escolhido para predizer o macrobloco X é o de menor SAD. Como para a síntese de uma vista são necessárias duas vistas de referência, este processo é realizado para cada da imagem de referência, sendo determinado dois vetores de movimento para cada macrobloco, representando seu deslocamente de cada uma das duas imagens de referência até a imagem sintetizada.

Após a determinação dos vetores de movimento, há duas possiblidades de utilização deles para a codificação, que são apresentadas nas seções 5.2.1.4 e 5.2.1.5.

5.2.1.2 Método (A).1 - Codificação independente da vista sintetizada

A codificação independente da vista sintetizada é realizada utilizando o padrão H.264/AVC, apresentado no Capítulo 2.

5.2.1.3 Método (A).2 - Codificação multi-vista normal para a vista sintetizada

A codificação multi-vista normal para a vista sintetizada consiste na codificação utilizando o padrão H.264/MVC (Seção 3.2.1) das vistas de referência e da vista sintetizada. A imagem da câmera 3 da sequência *Ballet* é considerada como sendo a vista esquerda e é codificada independentemente. A imagem da câmera 4 é considerada como a da vista direita e é predita a partir da câmera 3. A vista sintetizada, considerada como a vista central, é predita a partir das câmeras 3 e 4. O esquema de predição é ilustrado na Figura 3.5.

5.2.1.4 Método (A).3 - Codificação MVC com janela de busca alterada

Uma possibilidade de utilização dos vetores de movimento obtidos na síntese de vista, como apresentado na Seção 5.2.1.1, é fornecer tais vetores para o software JMVC. Na implementação padrão do H.264 e do JMVC, a janela de busca da estimação de movimento é posicionada de forma que a busca seja feita nos macroblocos em volta do macrobloco a ser predito. No método (A).3, a janela de busca é alterada de forma que seu centro tenha a mesma posição do macrobloco a ser predito, de forma que a busca seja feita nos macroblocos em volta da posição fornecida pelo vetor de movimento **mv**, obtido pelo processo da Seção 5.2.1.1. O processo de otimização RD continua o mesmo para a decisão do tipo de macrobloco. Apenas a janela de busca da estimação de movimento é alterada. Assim, os vetores de movimento serão apenas utilizados para a melhora da codificação.

5.2.1.5 Método (A).4 - Codificação e Síntese por Blocos

A segunda possibilidade de codificação utilizando as informações da síntese de vista consiste na idéia de, em vez de codificar e transmitir o vídeo da vista sintetizada para o decodificador, transmite-se as instruções necessárias para sua síntese. Após a determinação dos vetores pelo processo da Seção 5.2.1.1, é possível sintetizar a imagem apenas recuperando blocos das imagens de referência fornecidos pela projeção desses vetores de movimento. Assim, se esses vetores de movimento forem enviados para o decodificador e a imagem for rescontruída apenas com eles, será feita uma síntese de vista por blocos. Fazendo o processo de síntese de vista por blocos com blocos de tamanho 16×16 pixels, com as vistas de referência sem perdas por compressão, temos o resultado mostrado na Figura 5.3. Para conseguir uma taxa maior de compressão, a estimação de movimento temporal é realizada para diminuir a redundância temporal, já que após testes foi possível verificar que há uma maior redução de taxa na utilização de estimação temporal do que entre vistas. A utilização conjunta das duas estimações, temporal e entre-vistas, melhora ainda mais o resultado. Neste método é desligado a estimação do tipo *intra*, assim como não são utilizados macroblocos do tipo *skip/direct*. Se houver a compensação de movimento e o envio de resíduos na codificação, ocorre a melhora da imagem sintetizada, acarretando um aumento na taxa de bits.

Figura 5.3: Síntese de vista por blocos de tamanho 16x16.

5.2.2 Cenário (B) - Síntese no decodificador

No cenário (B), as imagens das vistas de referência e seus mapas de profundidade são codificados e transmitidos para que a síntese seja realizada no decodificador. Assim, também há diversas possibilidades de codificação para o levantamento de curvas desse sistema. Para os testes, foi utilizado um determinado parâmetro de quantização para as imagens de referência e outro para os mapas de profundidade. Logo, as possibilidades de compressão são formadas pela variação do Qp das imagens de referência em relação ao Qp dos mapas de profundidade. Para o levantamento de curvas, os vídeos são codificados, decodificados em seguida e é realizada a síntese da nova vista, esta podendo ser comparada à vista sintetizada do sistema sem perdas. A taxa de bits considerada é a soma da taxa de bits dos mapas de profundidade, já que este cenário tem como objetivo comparar apenas a possibilidade de criação da vista sintetizada, considerando a hipótese em que as vistas de referência são obrigatoriamente transmitidas.

5.2.3 Resultados

Como explicado na Seção 2.1.9, o levantamento de uma curva PSNR x Taxa de Bits para a comparação de métodos de compressão é feita codificando a sequência de teste com um dos métodos e variando a qualidade de compressão, ou seja, variando o parâmetro de quantização QP, assim obtendo diferentes valores de PSNR e taxas de bits para cada codificação. Os métodos de codificação já apresentados e utilizados para os experimentos são os seguintes.

Método (A).1 Codificação independente da vista sintetizada no padrão H.264/AVC. (Seção 5.2.1.2)

Método (A).2 Codificação da vista sintetizada no padrão H.264/MVC. (Seção 5.2.1.3)

Método (A).3 Codificação da vista sintetizada com o padrão H.264/MVC, com a janela de busca da estimação de movimento alterada para a posição fornecida pelos vetores de movimento obtidos na síntese de vista. (Seção 5.2.1.4)

Método (A).4 Codificação e envio os vetores de movimento obtidos pela síntese de vista para a síntese por blocos no decodificador. (Seção 5.2.1.5)

Cenário (B) Síntese no decodificador com transmissão dos mapas de profundidade. Os mapas de profundidade são codificados no padrão H.264/MVC e são transmitidos para que a síntese de vista seja feita no decodificador. (Seção 5.2.2)

Em um sistema multi-vista há diversas sequências de vídeos a serem codificadas referentes as diferentes vistas do sistema e cada uma pode ser codificada com um parâmetro QP diferente. Para comparar métodos de compressão, as curvas RD (taxas de bits x PSNR) são levantadas segundo dois tipos de experimento.

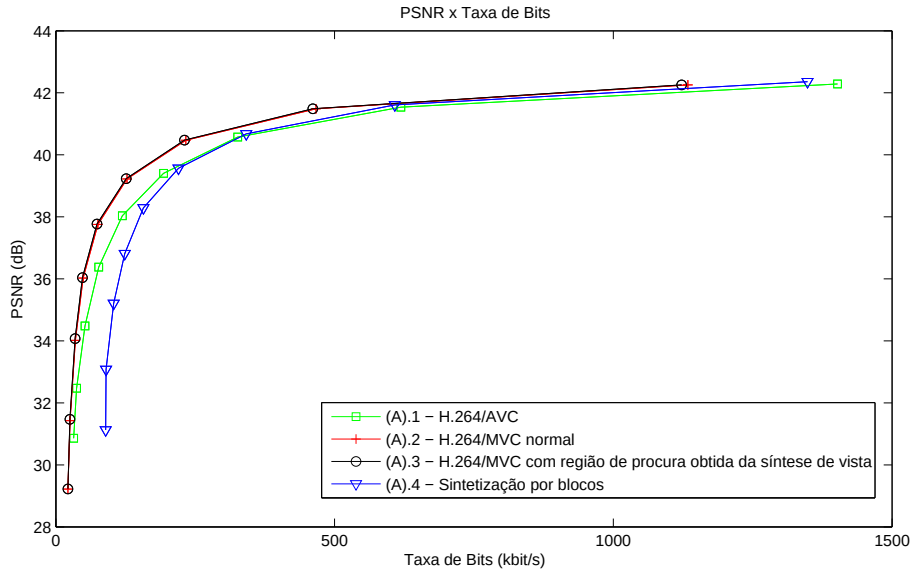
No primeiro, denomidado por experimento (1), todas as vistas são codificadas com um mesmo QP em comum. O parâmetro QP é alterado para variar a qualidade de compressão das sequências.

Este experimento é realizado apenas para os métodos do cenário (A) e a qualidade das vistas de referência variam da mesma forma que a qualidade da vista sintetizada. Assim, pode-se determinar qual o melhor método de compressão para o cenário (A) em sistemas que todas as vistas são codificadas com qualidade parecida. Os resultados deste experimento são apresentados na Seção 5.2.3.1.

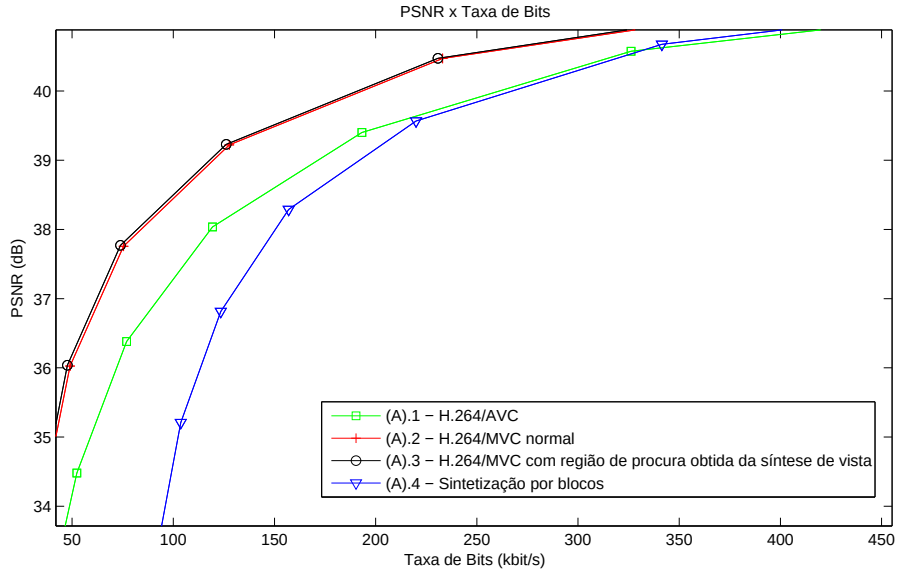
O segundo, denomidado por experimento (2), utiliza a codificação das vistas de referência com um QP fixo. Para todos os métodos de compressão em que a síntese de vista é feita no codificador, o QP da vista sintetizada é variado para obter as curvas RD. Para a síntese de vista no decodificador, os mapas de profundidade são codificados com um QP em comum. A curva RD é obtida variando este QP e realizando a síntese no decodificador utilizando as imagens das vistas de referência e os mapas de profundidade descomprimidos, simulando assim um sistema multi-vistas com codificação, transmissão e decodificação. Com este experimento é possível fazer a comparação dos cenários (A) e (B), já que as vistas de referência são codificadas com um QP fixo e apenas a qualidade da vista sintetizada é variada. Note que, no cenário (A), a qualidade da vista sintetizada é alterada pela variação do QP de sua compressão, e no cenário (B), sua qualidade é alterada variando o QP da codificação dos mapas de profundidade. Os resultados deste experimento são apresentado na Seção 5.2.3.2.

5.2.3.1 Experimento (1)

A Figura 5.4 representa os dados apresentados na Tabela 5.4, que são os resultados obtidos no experimento (1). Podemos observar as curvas obtidas pela compressão da vista sintetizada com síntese no codificador para uma região de procura de 24 pixels para a sequência ballet. Para o levantamento das curvas, considera-se que as vistas de referência estão sendo transmitidas para o usuário, assim sendo possível fazer a codificação H.264/MVC da vista sintetizada junto as vistas de referência. A taxa de bits considerada é somente a taxa de bits do vídeo da vista sintetizada. Isto é devido ao fato de que se deseja comparar a qualidade de compressão apenas da vista sintetizada em diferentes métodos de codificação, com a síntese dessa vista ocorrendo no codificador. Podemos observar que método (A).4 de síntese por blocos é a de pior desempenho para baixas taxas. Isto se dá pois a síntese por blocos utiliza apenas a estimação inter-quadros do H.264. Embora a síntese por blocos escolha a real origem dos pixels da imagem sintetizada, a implementação típica do H.264 decide o tipo de estimação que possua o melhor custo benefício entre a melhor predição da imagem original e o aumento da taxa de bits. Assim, muitas vezes o custo em bits é maior ao transmitir um vetor de movimento da real origem do bloco na imagem de referência em vez de utilizar macroblocos do tipo *skip* ou predição do tipo intra-quadros. Isto causa um grande aumento na taxa de bits do método (A).4 de síntese por blocos em comparação aos outros métodos. Para baixas taxas de compressão, o método H.264/AVC (A).1 possui melhor desempenho em relação a síntese por blocos, para maiores taxas de compressão, o contrário é verdadeiro. Para uma distorção PSNR de 41 dB, o método (A).4 consegue uma redução de 4% de taxa em relação ao (A).1. Já a codificação H.264/MVC normal (A).2 possui um ganho para PSNR de 41 dB de 18% de taxa em relação ao (A).4. Já a codificação (A).3 H.264/MVC com região de procura alterada, para uma



(a)



(b)

Figura 5.4: (a) Gráfico PSNR para o experimento (1). (b) Aproximação do gráfico em (a).

mesma PSNR de 41 dB, possui um ganho de 1,4% em relação a (A).2.

O método (A).3 foi o de melhor desempenho. Foi possível utilizar as informações da síntese de vista de forma a aumentar a taxa de compressão da codificação da vista sintetizada. Já o método (A).4 de síntese por blocos se mostrou ineficiente em relação ao outros, principalmente para baixas taxas. Para baixas taxas, isto ocorre principalmente pois os outros métodos também utilizam estimação do tipo intra-quadros, que tem uma grande importância para baixa taxas de vídeo.

Tabela 5.4: Tabela de resultados referente ao experimento (1). Ocorre a transmissão das vistas de referência para o decodificador, mas é considerado apenas a taxa de bits da vista sintetizada para a comparação dos métodos.

(A).1 - H.264/AVC		(A).2 - H.264/MVC normal		(A).3 - H.264/MVC com região de procura obtida da síntese de vista		(A).4 - Sintetização por blocos	
PSNR (dB)	Taxa de bits (kbit/s)	PSNR (dB)	Taxa de bits (kbit/s)	PSNR (dB)	Taxa de bits (kbit/s)	PSNR (dB)	Taxa de bits (kbit/s)
30,85	31,86	29,21	21,63	29,22	21,55	31,12	89,46
32,47	37,25	31,42	25,46	31,47	25,20	33,07	90,15
34,48	52,31	34,01	35,14	34,07	34,48	35,20	103,71
36,38	76,94	36,02	48,96	36,03	47,63	36,81	123,47
38,03	119,38	37,75	75,41	37,76	73,88	38,28	157,02
39,40	193,18	39,22	127,65	39,22	126,15	39,56	219,98
40,57	326,17	40,46	233,08	40,47	230,74	40,67	341,42
41,53	618,83	41,48	466,76	41,48	461,04	41,61	608,09
42,28	1402,20	42,25	1133,69	42,25	1122,50	42,35	1348,40

5.2.3.2 Experimento (2)

Uma comparação de síntese no codificador e no decodificador pode ser vista na Figura 5.5, que se refere ao experimento (2), sendo a codificação do sistema mantendo uma qualidade fixa para vistas de referência, mas variando a qualidade da codificação da vista sintetizada e dos mapas de profundidade. Neste caso, é considerado que as vistas de referência são transmitidas para o usuário, assim a taxa de bits considerada para os métodos com síntese no codificador é apenas a taxa de bits do vídeo da vista sintetizada, e no caso que a síntese ocorre no decodificador, cenário (B), a taxa de bits considerada é a taxa de bits dos vídeos dos mapas de profundidade. A síntese de vista por blocos (A).4 possui um melhor desempenho em relação a codificação H.264/AVC (A).1, para uma PSNR de 41 dB, há um ganho de 17,8%. Pode-se observar que o custo de transmissão em (B) dos mapas de profundidade para a síntese no decodificador é muito alto, isto ocorre porque, para a sequência *ballet*, a qualidade da imagem sintetizada é sensível a qualidade dos mapas de profundidade [15], assim ocorrendo a degradação da vista sintetizada. Para que haja uma qualidade elevada da vista sintetizada com a síntese realizada no decodificador, é necessário que as perdas na compressão dos mapas de profundidade não os distorçam, assim os vídeos codificados destes, com baixas perdas, possuem uma elevada taxa de bits. Devido a este problema, várias soluções estão sendo pesquisadas para melhorar a síntese de vista utilizando mapas de profundidade codificados. Em [11], os autores propõem um filtro para uma melhor codificação e reconstrução das bordas do mapa de profundidade, e em [24], os autores propõem um método de síntese baseado em treliças, que é mais robusto a erros nos mapas de profundidade. Neste trabalho, os mapas de profundidade

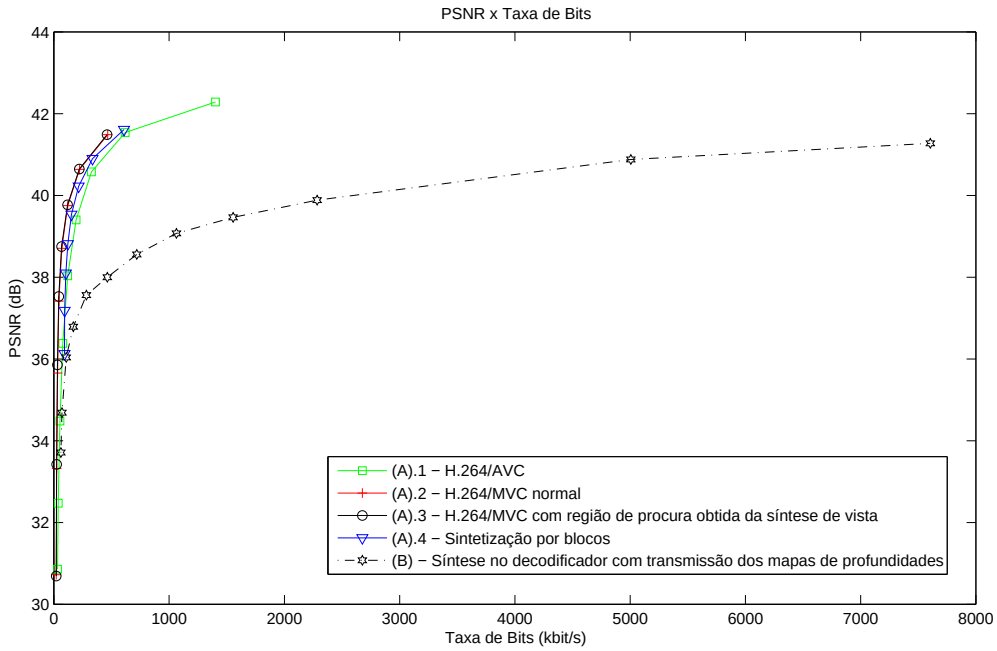
Tabela 5.5: Tabela de resultados das transmissões das vistas do sistema, referente ao experimento (2). As vistas de referência são transmitidas com qualidade fixa de 42,3 dB, com variação da qualidade de codificação da vista sintetizada e dos mapas de profundidade.

(A).1 - H.264/AVC		(A).2 - H.264/MVC normal		(A).3 - H.264/MVC com região de procura obtida da síntese de vista		(A).4 - Sintetização por blocos		(B) - Síntese no decodificador com transmissão dos mapas de profundidade	
PSNR (dB)	Taxa de bits (kbit/s)	PSNR (dB)	Taxa de bits (kbit/s)	PSNR (dB)	Taxa de bits (kbit/s)	PSNR (dB)	Taxa de bits (kbit/s)	PSNR (dB)	Taxa de bits (kbit/s)
30,85	31,86	30,72	20,10	30,69	20,08	36,12	91,71	33.71	60.95
32,47	37,25	33,31	23,46	33,42	23,27	37,18	92,66	34.68	69.61
34,48	52,31	35,65	31,56	35,85	30,89	38,09	103,50	36.78	168.69
36,38	76,94	37,41	43,15	37,52	42,06	38,81	120,37	37.99	463.62
38,03	119,38	38,70	68,35	38,74	66,50	39,53	151,82	38.55	719.90
39,40	193,18	39,75	119,81	39,77	118,59	40,22	214,78	39.46	1554.67
40,57	326,17	40,63	222,79	40,64	220,98	40,89	335,58	39.88	2284.23
41,53	618,83	41,48	466,76	41,48	461,04	41,61	608,09	40.87	5006.66
42,28	1402,20							41.27	7604.89

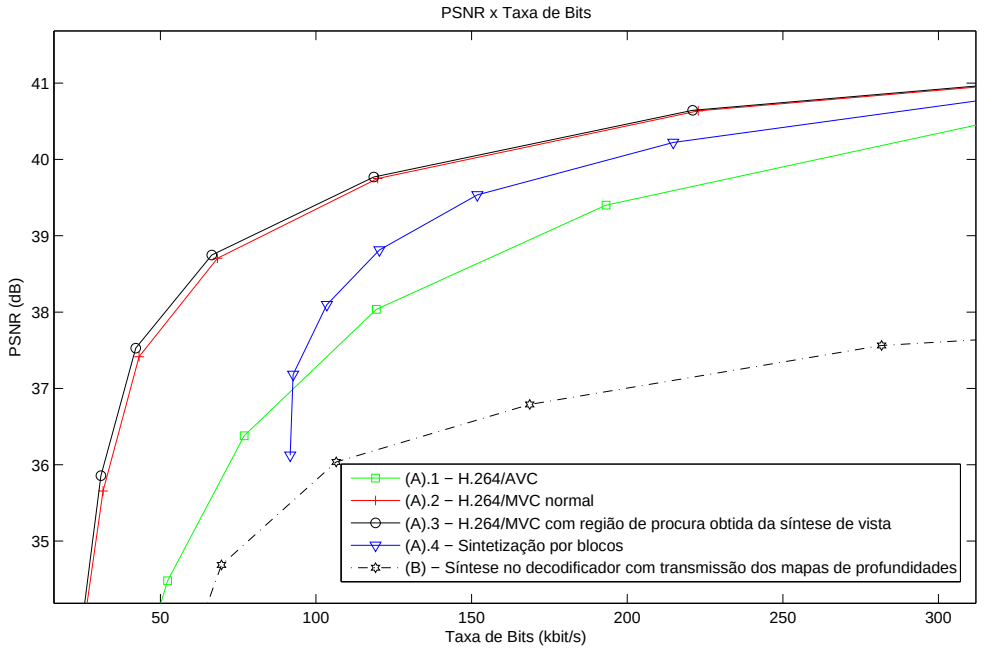
foram codificados utilizando H.264/MVC. Devido aos resultados obtidos neste experimento, pode-se estimar que para uma PSNR de 41 dB, pode-se comprimir e transmitir pelo menos 12 vistas sintetizadas na codificação (A).1 utilizando a mesma taxa de bits necessária para a transmissão dos mapas de profundidade. Assim, para sistemas FTV que possuam canal de retorno entre codificador/decodificador, é interessante que a síntese seja feita no codificador, podendo assim economizar em banda de transmissão. Caso contrário, ainda é necessário a transmissão dos mapas de profundidade para a síntese. O método de compressão (A).3 possui um desempenho melhor de taxa de bits em relação ao (A).2 de 1,37%.

Para este experimento, o método (A).3 novamente conseguiu melhor desempenho, tomando proveito das informações da síntese de vista para auxiliar a codificação. O método (A).4 de síntese por blocos se mostrou melhor que a codificação independente das vistas sintetizada, método (A).1, quando as vistas de referência possuem boa qualidade, mas de pior desempenho em relação aos métodos de codificação multi-vista (A).2 e (A).3. O método (A).4 só não alcançou melhor desempenho que o método (A).1 para baixas taxas, porque, como já mencionado, a síntese por blocos não utiliza estimação intra-quadros, que é eficaz para baixas taxas.

A Figura 5.6 mostra as curvas dos métodos em que a síntese é feita no codificador e no decodificador considerando a taxa de bits total de cada cenário, incluindo as imagens da vistas de referência, referentes a Tabela 5.6. Neste caso, o método (A).1 apenas transmite a vista sintetizada. No método (A).2, há a transmissão das vistas de referência e da vista sintetizada, e a taxa de bits total é a soma das taxas de bits dos vídeos dessas vistas. E por último, no caso (B) em



(a)



(b)

Figura 5.5: (a) Gráfico PSNR para o experimento (2). (b) Aproximação do gráfico em (a).

que a síntese é feita no decodificador, há a transmissão das vistas de referência e de seus mapas de profundidade, logo a taxa de bits total é a soma das taxas de bits de cada um desses vídeos. As duas vistas de referência são codificadas com uma PSNR de 42,3 dB e uma taxa de bits total de 836,9 kbit/s. Em sistemas com canal de retorno, é de interesse que a transmissão seja feita pelo

Tabela 5.6: Tabela de resultados das codificações da vista sintetizada com a síntese feita no codificador e no decodificador, contabilizando a taxa de bits total do sistema em cada método.

(A).1 - H.264/AVC		(A).2 - H.264/MVC normal		(B) - Síntese no decodificador	
PSNR (dB)	Taxa de bits (kbit/s)	PSNR (dB)	Taxa de bits (kbit/s)	PSNR (dB)	Taxa de bits (kbit/s)
30,85	31,86	30,72	857,00	33,71	897,85
32,47	37,25	33,31	860,33	34,68	906,52
34,48	52,31	35,65	868,46	36,78	1005,59
36,38	76,94	37,41	880,05	37,99	1300,52
38,03	119,38	38,70	905,25	38,55	1556,80
39,40	193,18	39,75	956,71	39,07	1900,53
40,57	326,17	40,63	1059,69	39,88	3121,14
41,53	618,83	41,48	1303,67	40,87	5843,56
42,28	1402,20			41,27	8441,80

método (A).1, pois o método (A).2 transmitirá também as vistas de referência sem necessidade. Assim, para uma PSNR de 41 dB, o método (A).1 reduz a taxa de bits do método (A).2 em 61%, pois as vistas de referência não são transmitidas. Para o ponto em que a PSNR é de 41 dB, para (B) em que a síntese seja feita no decodificador, é necessário o envio das vistas de referência e seus mapas de profundidade, e para essa PSNR a taxa de bits do sistema é de 6646 kbit/s, enquanto a do (A).1 é de 455 kbit/s e do (A).2 é de 1164 kbit/s. Assim, com a mesma taxa de bits utilizada pelo sistema de síntese no decodificador, é possível transmitir 14 vistas sintetizadas no método (A).1 ou transmitir as duas vistas de referência junto com pelo menos 12 vistas no método (A).2.

5.3 Transmissão de Dados em Sistemas FTV

Sistemas FTVs necessitam da geração de mapas de profundidade e de síntese de vistas, que são processos computacionalmente complexos. Um balanço entre taxa de transmissão e complexidade computacional é desejado. Em transmissões do tipo *broadcast*, em que dados são transmitidos para um grande número de usuários, um codificador computacionalmente potente pode transmitir as informações MVD (Seção 3.2.2) do sistema para vários receptores, cada um deles sintetizando seu próprio ponto de vista. Em um cenário contrastante, como por exemplo conferências entre duas pessoas, pode haver canal de retorno entre codificador e decodificador. Logo, há a possibilidade do decodificador informar ao codificador qual vista o usuário deseja assistir, assim o codificador pode sintetizar e transmitir apenas a vista informada pelo decodificador.

Nesta Seção, será apresentada uma arquitetura genérica que acomoda os diferentes cenários de sistemas FTV apresentados na Seção 5.1 e será estudado o balanço entre banda de transmissão

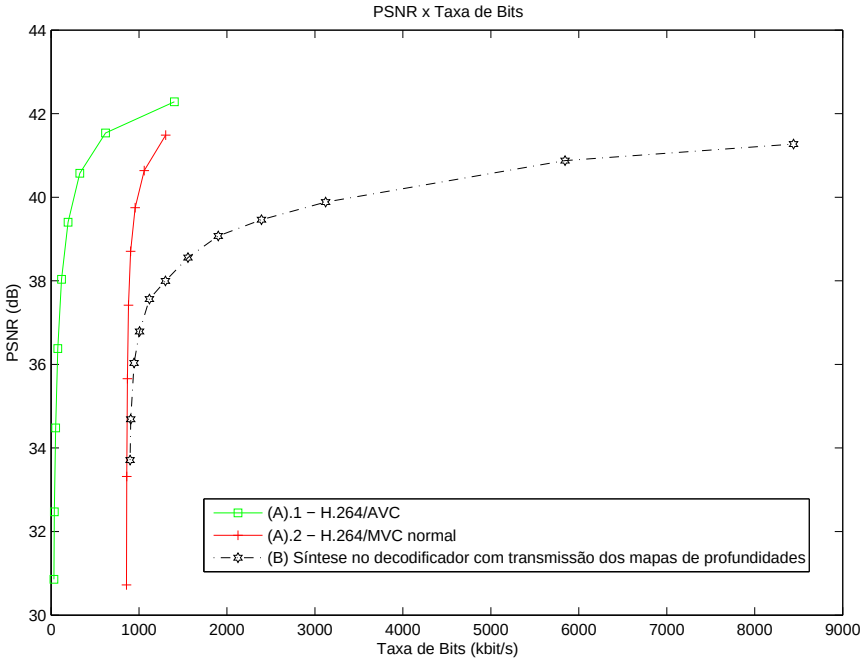


Figura 5.6: Comparação dos métodos de compressão de vista sintetizada considerando a taxa de bits total utilizada por cada método.

utilizada e qualidade de síntese, já que a geração dos mapas de profundidade e a síntese de vista podem ser realizados no codificador ou no decodificador. A Seção 5.3.1 apresenta a arquitetura genérica para sistemas FTV, a Seção 5.3.2 discute os cenários já apresentados levando em consideração essa arquitetura genérica e a Seção 5.3.3 desenvolve experimentos nos diferentes cenários FTV.

5.3.1 Arquitetura de sistemas FTV

Uma arquitetura genérica para sistemas FTV é ilustrada na figura 5.7. Ela é baseada em três blocos principais: a captura da cena, a rede e o *display*. O primeiro bloco captura o conteúdo multi-vista da cena. A rede é constituída por um bloco codificador e um bloco decodificador. A geração dos mapas de profundidade e síntese de vista podem ser realizadas em qualquer bloco com potência computacional disponível, normalmente, estes processos são realizados no bloco codificador ou decodificador. Note que o *display* pode ser utilizado como decodificador dependendo de sua potência computacional ou da aplicação do sistema. Cada equipamento *display* pode escolher o seu próprio ponto de vista e o conteúdo multi-vista pode estar disponível para todos os receptores. A princípio, o ponto de vista é decidido pelo *display* e o processo computacionalmente complexo de síntese de vista é realizado por algum equipamento ao alcance dele. Entretanto, se o *display* tiver baixa complexidade computacional, é improvável que ele seja capaz de estimar a profundidade e sintetizar uma nova vista utilizando as informações multi-vista recebidas. Se o codificador transmitir os mapas de profundidade de cada vista, a síntese de vista é simplificada

em troca de uma taxa de transmissão maior. É sugerida a introdução de um decodificador na rede capaz de receber conteúdos multi-vistas do codificador, com ou sem a transmissão dos mapas de profundidade, e sintetizar uma nova vista. Deste modo, equipamentos *display* se comunicam com o decodificador e recebem apenas a vista desejada. O decodificador da rede precisa re-codificar a vista sintetizada com uma qualidade proporcional a banda de transmissão disponível na rede local. Desta maneira, broadcast de conteúdos FTV podem ser usados em tables e outros equipamentos menores, transferindo os complexos processos de estimação de profundidade e síntese de vista para a rede.

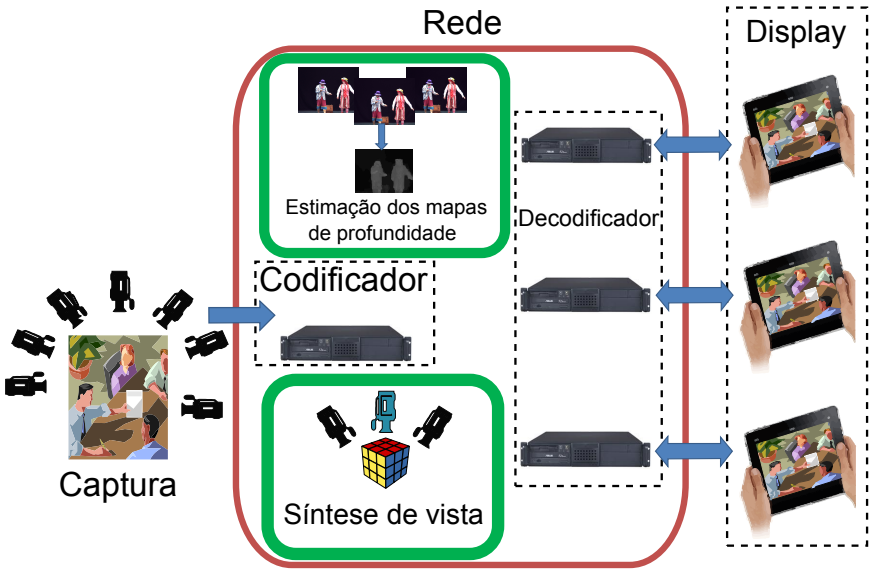


Figura 5.7: Arquitetura genérica para sistemas FTV. O conteúdo é capturado e enviado para o codificador. As informações dos vídeos são codificadas e transmitidas pela rede para um decodificador na rede, que decodifica e transmite para o *display*. A síntese de vista e estimação de profundidade podem ser realizadas no codificador ou no decodificador.

5.3.2 Cenários para sistemas FTV

As possibilidades de cenários FTV já foram explicados na Seção 5.1.

Sem o canal de retorno, a síntese deve ser realizada no decodificador ou no *display*, e o conteúdo multi-vista deve ser transmitido pela rede. Isto leva a dois casos distintos.

No primeiro, as imagens das vistas são transmitidas sem os mapas de profundidade. Logo, a estimação de profundidade e a síntese de vista são realizadas no decodificador ou no *display*.

No segundo caso, a profundidade é estimada no codificador e o conteúdo MVD é transmitido, assim a nova vista é sintetizada no decodificador ou no *display*.

Note que se a síntese de vista for realizada no decodificador da rede, a vista virtual deve ser codificada e enviada para o *display* por uma rede local, resultando em mais perdas de qualidade. Se na rede local existir uma banda de transmissão suficientemente grande, a máxima qualidade atingida com a síntese de vista no bloco decodificador será no caso em que a codificação da vista

sintetizada é realizada com compressão sem perdas. Neste caso, a síntese de vista no decodificador da rede ou no *display* resultam na mesma qualidade do vídeo final, sendo este o caso considerado neste capítulo.

5.3.3 Experimento

Foi realizado um experimento em um sistema FTV de cinco vistas para comparar o desempenho entre os cenários mencionados considerando a taxa total de bits utilizada por cada cenário e a qualidade da vista sintetizada entregue ao usuário. Foram usados 90 quadros das sequências de teste *Pantomime* e *Champagne* [1]. São consideradas as vistas 35, 37, 39, 41 e 43 destas duas sequências para compor o sistema. O sistema é capaz de sintetizar qualquer vista intermediária entre as câmeras 35 e 43. A estimação de profundidade necessita da imagem de duas câmeras adjacentes, logo, para gerar o mapa de profundidade da vista 35, as imagens das vistas 33 e 37 são usadas como referência. Da mesma maneira, a geração do mapa de profundidade da vista 43 ocorre utilizando as imagens das vistas 41 e 45. Embora este sistema use cinco vistas para sintetizar novas vistas, as vistas vizinhas 33 e 45 são necessárias apenas para o propósito de estimação de profundidade, então sete vistas são necessárias para o sistema. A síntese de vista é realizada com o software de referência VSRS 3.5 [9] e a estimação de profundidade é feita no modo automático do software de referência DERS 5.1 [9]. Os videos são codificados no padrão H.264 com o software de referência JMVC 8.3.1 para codificação do tipo AVC e MVC. Para codificação a codificação multi-vista, predição temporal e entre vistas são realizadas.

Para comparar os cenários, as vistas 35 e 37 são utilizadas para sintetizar a vista 36 das sequências de testes, e a PSNR entre a vista sintetizada e a imagem original da vista 36 é calculada. Esta métrica é utilizada para comparar a qualidade de síntese do sistema, utilizando a vista 36 como base. Por esta razão o sistema é apenas composto de vistas ímpares da sequência *Ballet*, sendo assim possível utilizar uma vista de número par como parâmetro de comparação. Este processo é diferente do realizado da Seção 5.2, em que a PSNR é calculada entre a vista sintetizada com os dados MVD originais e a vista sintetizada comprimida ou sintetizada com os dados MVD comprimidos. Note que não são necessárias todas as vistas para a síntese da vista 36, mas todas as vistas precisam ser transmitidas nos cenários os quais não há canal de retorno. A vista sintetizada com estimação de profundidade e síntese de vista com os vídeos originais sem perdas produz uma PSNR de 38.16 dB para a sequência *Pantomime* e 32.55 dB para a sequência *Champagne*. Neste experimento, os QPs utilizados para a codificação das imagens são: 6, 10, 14, 18, 22, 30, 38 e 46. Para os mapas de profundidade, os QPs usados são: 10, 18, 38, 46.

5.3.3.1 Cenário (A) - Síntese no Codificador

No cenário (A), apenas a vista 36 sintetizada é transmitida pela rede. Então, a PSNR entre a imagem original e descomprimida da vista 36 sintetizada é calculada.

5.3.3.2 Cenário (B) - Estimação de profundidade e Síntese no decodificador

No cenário (B), as imagens das sete vistas (33, 35, 37, 39, 41, 43, 45) são transmitidas pela rede. Como mencionado anteriormente, as imagens das vistas 33 e 45 são enviadas apenas pelo propósito de estimação de profundidade das vistas 35 e 43. As imagens das sete vistas são codificadas em MVC com o mesmo parâmetro de quantização (QP). A estimação de profundidade é realizada com os vídeos descomprimidos e em seguinda a vista 36 é sintetizada. A taxa de bits considerada é a taxa de bits total necessária para o envio das imagens das sete vistas.

5.3.3.3 Cenário (C) - Transmissão do Conteúdo MVD e Síntese no Decodificador

No cenário (C), o conteúdo MVD é transmitido pela rede para síntese no decodificador. As imagens e mapas de profundidade das vistas são codificados separadamente em MVC e elas podem ser codificados com diferentes QPs. Para obter os dados de PSNR por taxa de bits, é fixado um QP para a compressão dos mapas de profundidade enquanto é variado o QP das imagens das vistas. Variando o QP dos mapas de profundidade, é obtido uma de PSNR por taxa de bits para cada QP dos mapas de profundidade. A síntese da vista 36 é realizada utilizando as informações decodificadas. A taxa de bits considerada é a taxa de bits total necessária para enviar o conteúdo MVD das cinco vistas. Esta abordagem é útil na análise dos efeitos de compressão dos mapas de profundidade na síntese de vista.

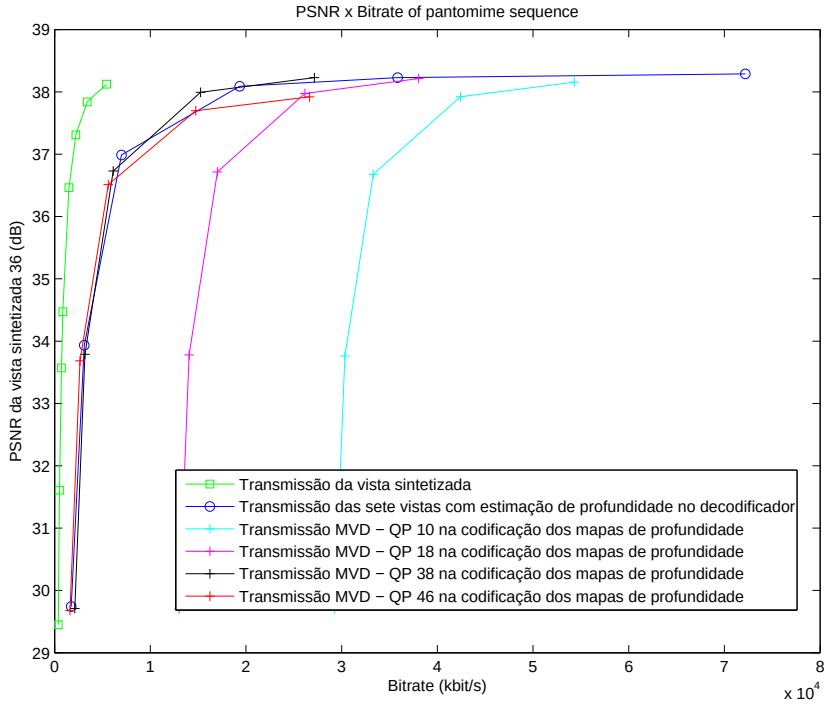
5.3.3.4 Resultados

A Figura 5.8 mostra a comparação entre os três cenários. Na Figura 5.8(a), o cenário (A) necessita, para uma PSNR de 38.1 dB, de uma taxa de bits de aproximadamente 74% menor que os outros dois cenários. Na Figura 5.8(b), a taxa necessária para o cenário (A) para uma PSNR de 32.6 dB é aproximadamente 80% menor se comparada com a melhor transmissão do conteúdo MVD do cenário (C). Este cenário só é adequado para aplicações FTV com um baixo número de usuários, como vídeo conferências. Caso o número de usuários que tenham escolhidos diferentes pontos de vista seja aproximadamente igual ou ultrapasse o número de vistas do sistema, é esperado que os outros cenários sejam mais eficientes que este.

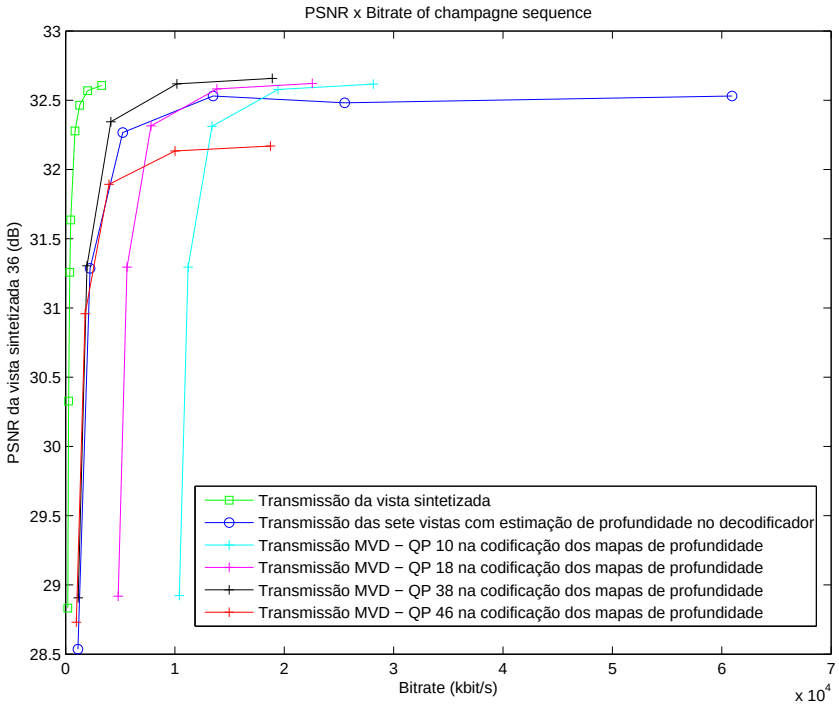
Para o cenário (C), é possível observar que a qualidade das imagens das vistas são mais importantes para a qualidade final da vista sintetizada do que a qualidade dos mapas de profundidade, como concluído em [10]. A PSNR da vista sintetizada com diferentes taxas de compressão para os mapas de profundidade apresentaram pequenas diferenças, embora a taxa de bits dos mapas de profundidade afetem significantemente a taxa de bits total do sistema. Na Figura 5.8(a), o cenário (B) teve uma performance parecida com o cenário (C) para baixas e médias taxas de bits, mas foi superado nos casos de alta taxa de bits para os mapas de profundidade. Na Figura 5.8(b), o cenário (B) teve sua performance superada pelo cenário (C) para altos valores de PSNR, isto ocorreu pelo fato de que, para a sequência Champagne, a estimação de profundidade utilizando imagens de referência descomprimidas causaram distorções na qualidade da vista sintetizada final. No geral, o cenário (B) alcançou uma melhor performance no sentido de PSNR por taxa de bits

em relação ao cenário (C).

Este capítulo considerou um sistema FTV de cinco vistas. Em sistemas FTV práticos, é esperado um número maior de câmeras, podendo ser utilizados na captura de 3 a 100 câmeras, ou mais. Neste caso, a estimação de profundidade no codificador deve obter uma performance melhor do que o cenário em que o conteúdo MVD é transmitido. Em um sistema FTV em que as câmeras são alinhadas e espaçadas apenas em um direção, a horizontal, assim como nas sequências de teste utilizadas neste capítulo, o cenário (B) necessitaria de apenas duas vistas a mais para estimação de profundidade no decodificador, enquanto no cenário (C), a taxa de bits pros mapas de profundidade aumentariam com o aumento do número de câmeras. As pesquisas atuais têm seu foco na melhora da compressão dos mapas de profundidade, reduzindo artefatos gerados na síntese de vista devidos a compressão. Trabalhos futuros podem focar na melhor estimação de profundidade utilizando imagens descomprimidas, pois isto levaria a ganhos no sistema podendo ter uma performance melhor do que cenário de transmissão do conteúdo MVD.



(a) Pantomime



(b) Champagne

Figura 5.8: Comparação de um sistema FTV de cinco vistas para as sequências Pantomime e Champagne em diferentes cenários. As curvas são obtidas com a taxa total de bits necessária para cada cenário e as informações PSNR são calculadas entre a imagem original e sintetizada da vista 36.

Capítulo 6

Conclusões

Este trabalho teve como objetivo investigar o funcionamento de sistemas de pontos de vista livre (FTV). Estes sistemas têm como característica principal a interface entre conteúdo-usuário, em que é possível escolher o ponto de vista a ser exibido. Para isto, é necessário que haja a informação do conteúdo multi-vista do sistema, em que a cena é capturada por diferentes pontos de vista. A capacidade de exibir ponto de vista livre pode ser alcançada capturando todos os pontos de vista de uma cena, ou capturando-a por um número discreto de câmeras e sintetizando as vistas intermediárias entre elas. Para a síntese de novas vistas, é necessária a geração dos mapas de profundidade das vistas capturadas por meio da estimação de profundidade. Assim, a síntese de uma nova vista arbitrária requer as informações das imagens capturadas pelas câmeras e de seus mapas de profundidade.

Optou-se por utilizar a captura por um número discreto de câmeras. Com isso, foi desenvolvido um sintetizador de vistas, em que a entrada são os vídeos da cena e seus mapas de profundidade. Com isto, é possível criar câmeras virtuais que representam pontos de vista não capturados da cena. Neste trabalho, apresentou-se algumas soluções e melhorias para os sintetizadores de vista, como tratamento de bordas e interpolação por projeção reversa.

Em um sistema FTV, é necessária a compressão e o envio das informações requeridas para a síntese de novas vista pelo usuário. Caso haja um canal de retorno no sistema, é possível para o decodificador informar ao codificador qual vista foi escolhida por ele, havendo a necessidade de transmissão da vista em questão. Neste caso, a estimação de profundidade e a síntese de vista são realizadas no codificador. Este sistema também pode ser realizado por meio da transmissão das imagens das câmeras e de seus mapas de profundidade, em que a estimação de profundidade ocorre no codificador e a síntese de vista no decodificador. No terceiro cenário, o codificador transmite as imagens das vistas capturadas pelo sistema, assim, a estimação de profundidade e a síntese de vista ocorrem no decodificador. O Capítulo 5 desenvolveu experimentos nestes três cenários.

Na Seção 5.2, comparou-se os dois primeiros cenários. Usou-se como métrica de comparação as perdas da imagem sintetizada final para a imagem sintetizada original, que é realizada com as imagens e mapas de profundidade sem perdas, e focou-se na compressão da vista sintetizada em sistemas os quais as imagens capturadas pelas câmeras são enviadas ao decodificador. Com isto,

é possível utilizar informações da síntese de vista para auxiliar a compressão da vista síntetizada. Curvas PSNR por taxa de bits foram levantadas para comparar tais métodos de síntese, no codificador e no decodificador. Com a síntese no codificador, os métodos de compressão utilizaram o padrão H.264 e o software de referência JMVC, sendo os métodos de compressão: AVC; MVC, que consiste na compressão da vista sintetizada utilizando as vistas capturadas como referência; MVC do mesmo modo, mas com janela de busca da estimação de movimento alterada para regiões determinadas pela síntese de vista; e síntese por blocos, que consiste somente na utilização dos vetores de movimento obtidos pela síntese de vista na compressão da vista sintetizada. Com a síntese no decodificador, as imagens das câmeras de referência e seus mapas de profundidade são transmitidas para a síntese. Caso haja um canal de retorno, é interessante que a síntese seja realizada no codificador pois há um alto custo de taxa de bits para o envio dos mapas de profundidade sem a redução da PSNR da vista sintetizada em relação a vista sintetizada original, podendo ser enviados ao mínimo 12 vistas sintetizadas em codificação AVC, sem a transmissão dos mapas de profundidade e mantendo um mesmo valor de PSNR. Se não houver canal de retorno, é necessário a transmissão das imagens das câmeras de referência e de seus mapas de profundidade para a realização da síntese.

Na Seção 5.3, os três cenários descritos foram comparados. Considerou-se um sistema FTV de cinco vistas, podendo ser sintetizada qualquer vista intermediárias entre elas. No experimento, utilizou-se como parâmetro de comparação a imagem da câmera 36, assim, o decodificador sintetiza a imagem da vista 36 e a PSNR entre a imagem sintetizada e original da vista 36 é obtida. Desta forma, o experimento comparou o custo de bits por qualidade da vista sintetizada. O cenário em que a síntese ocorre no codificador obteve a melhor performance. Como já descrito, é necessário um canal de retorno para sua realização e ele é adequado para aplicações com um número reduzido de usuário, como vídeo conferências, mas provavelmente terá sua performance superada pelos outros cenários em aplicações cujo número de usuários seja aproximadamente igual ou supere o número de câmeras no sistema. Os cenários os quais a síntese ocorre no codificador obtiveram performance parecida. Pesquisas recentes focaram na melhora da compressão dos mapas de profundidade com redução de artefatos gerados na síntese de vista devido as perdas de compressão. Isto abre novas possibilidades de sistemas em que a estimação de profundidade seja realizada no decodificador.

REFERÊNCIAS BIBLIOGRÁFICAS

[1] "http://www.tanimoto.nuee.nagoya-u.ac.jp/."

[2] "Itu-t recommendation h.264: Advanced video coding for generic audiovisual services," International Telecommunications Union, 2003.

[3] J. Canny, "A computational approach to edge detection," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 8, pp. 679–698, November 1986.

[4] M. Flierl and B. Girod, "Efficient prediction structures for multiview video coding," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 17, pp. 1461 – 1473, November 2007.

[5] R. C. Gonzalez and R. E. Woods, *Digital Image Processing (3rd Edition)*. Upper Saddle River, NJ, USA: Prentice-Hall, Inc., 2006.

[6] A. Gruen and T. S. Huang, Eds., *Calibration and Orientation of Cameras in Computer Vision*. Secaucus, NJ, USA: Springer-Verlag New York, Inc., 2001.

[7] R. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*, 2nd ed. New York, NY, USA: Cambridge University Press, 2003.

[8] C.-C. Jong Dae Oh; Siwei Ma; Kuo, "Disparity estimation and virtual view synthesis from stereo video," in *IEEE International Symposium on Circuits and Systems. ISCAS 2007.*, May 2007.

[9] I. JTC1/SC29/WG11, "Reference softwares for depth estimation and view synthesis," Doc. M15377, April 2008.

[10] K. Klimaszewski, K. Wegner, and M. Domanski, "Distortions of synthesized views caused by compression of views and depth maps," in *3DTV Conference: The True Vision - Capture, Transmission and Display of 3D Video, 2009*, may 2009, pp. 1 –4.

[11] A. Y.-S. H. Kwan-Jung Oh; Vetro, "Depth coding using a boundary reconstruction filter for 3-d video systems," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 21, pp. 350–359, March 2011.

[12] E.-K. Lee and Y.-S. Ho, "Generation of high-quality depth maps using hybrid camera system for 3-d video," *J. Vis. Comun. Image Represent.*, vol. 22, pp. 73–84, January 2011.

[13] B. G. Markus Flierl, "Multi-view video compression, exploiting inter-image similarities," *IEEE Signal Processing Magazine*, vol. 24, pp. 66–76, November 2007.

[14] E. Martinian, A. Behrens, J. Xin, and A. Vetro, "View synthesis for multiview video compression," in *IN PICTURE CODING SYMPOSIUM*, 2006.

[15] P. Merkle, Y. Morvan, A. Smolic, D. Farin, K. Müller, P. H. N. de With, and T. Wiegand, "The effects of multiview depth video compression on multiview rendering," *Image Communication*, vol. 24, pp. 73–88, January 2009.

[16] K. Müller, A. Smolic, K. Dix, P. Merkle, P. Kauff, and T. Wiegand, "View synthesis for advanced 3d video systems," *EURASIP Journal on Image and Video Processing*, vol. 2008, pp. 1–11, 2008.

[17] K.-J. Oh, S. Yea, and Y.-S. Ho, "Hole filling method using depth based in-painting for view synthesis in free viewpoint television and 3-d video," in *Proceedings of the 27th conference on Picture Coding Symposium*, ser. PCS'09. Piscataway, NJ, USA: IEEE Press, 2009, pp. 233–236.

[18] C. M. Ordinas, "Virtual view generation for free viewpoint applications," Master's thesis, Universitat Politècnica de Catalunya, July.

[19] K. R. Rao and P. Yip, *Discrete Cosine Transform: Algorithms, Advantages, Applications.* Academic Press, Boston, 1990.

[20] I. E. G. Richardson, *H.264 and MPEG-4 Video Compression: Video Coding for Next-generation Multimedia.* Wiley, John and Sons, Incorporated, 2003.

[21] X. Suau, J. Casas, and J. Ruiz-Hidalgo, "Multi-resolution illumination compensation for foreground extraction," in *Image Processing (ICIP), 2009 16th IEEE International Conference on*, November 2009, pp. 3225 –3228.

[22] T. K. Tan, G. Sullivan, and T. Wedi, "Recommended simulation common conditions for coding efficiency experiments," ITU-T Q.6/SG16 (VCEG), 2005.

[23] M. Tanimoto, "Ftv (free viewpoint television) creating ray-based image engineering," in *IEEE International Conference on Image Processing, 2005. ICIP 2005.*, September 2005, pp. II – 25 – 8.

[24] A. B. M. Tian, D.; Vetro, "A trellis-based approach for robust view synthesis," in *IEEE International Conference on Image Processing (ICIP)*, September 2011.

[25] D. Tian, P. Pandit, P. Yin, and C. Gomila, "Study of mvc coding tools," *Joint Video Team, Doc. JVT-Y044*, 2007.

[26] A. Vetro, A. M. Tourapis, K. Muller, and T. Chen, "3d-tv content storage and transmission," *IEEE Transactions on Broadcasting*, vol. 57, pp. 384 – 394, June 2011.

[27] G. Wolberg, *Digital Image Warping*, 1st ed. Los Alamitos, CA, USA: IEEE Computer Society Press, 1994.

[28] C. L. Zitnick, S. B. Kang, M. Uyttendaele, S. Winder, and R. Szeliski, "High-quality video view interpolation using a layered representation," *ACM Trans. Graph.*, vol. 23, pp. 600–608, August 2004.

[29] L. C. Zitnick, S. B. Kang, M. Uyttendaele, S. Winder, and R. Szeliski, "High-quality video view interpolation using a layered representation," *ACM Trans. Graph.*, vol. 23, no. 3, pp. 600–608, 2004.