CODIFICAÇÃO DE TEXTURA E PROFUNDIDADE DE IMAGENS 3-D UTILIZANDO RECORRÊNCIA DE PADRÕES MULTIESCALAS

Anderson Vinícius Corrêa de Oliveira

Dissertação de Mestrado apresentada ao Programa de Pós-graduação em Engenharia Elétrica, COPPE, da Universidade Federal do Rio de Janeiro, como parte dos requisitos necessários à obtenção do título de Mestre em Engenharia Elétrica.

Orientador: Eduardo Antônio Barros da Silva

Rio de Janeiro
Março de 2011

CODIFICAÇÃO DE TEXTURA E PROFUNDIDADE DE IMAGENS 3-D UTILIZANDO RECORRÊNCIA DE PADRÕES MULTIESCALAS

Anderson Vinícius Corrêa de Oliveira

DISSERTAÇÃO SUBMETIDA AO CORPO DOCENTE DO INSTITUTO ALBERTO LUIZ COIMBRA DE PÓS-GRADUAÇÃO E PESQUISA DE ENGENHARIA (COPPE) DA UNIVERSIDADE FEDERAL DO RIO DE JANEIRO COMO PARTE DOS REQUISITOS NECESSÁRIOS PARA A OBTENÇÃO DO GRAU DE MESTRE EM CIÊNCIAS EM ENGENHARIA ELÉTRICA.

Examinada por:

Prof. Eduardo Antônio Barros da Silva, Ph.D.

Prof. Hae Yong Kim, D.Sc.

Prof. Weiler Alves Finamore, Ph.D.

Prof.ª Carla Liberal Pagliari, Ph.D.

Prof. Gelson Vieira Mendonça, Ph.D.

RIO DE JANEIRO, RJ – BRASIL
MARÇO DE 2011

Oliveira, Anderson Vinícius Corrêa de

Codificação de Textura e Profundidade de Imagens 3-D Utilizando Recorrência de Padrões Multiescalas/Anderson Vinícius Corrêa de Oliveira. – Rio de Janeiro: UFRJ/COPPE, 2011.

XIV, 88 p.: il.; 29, 7cm.

Orientador: Eduardo Antônio Barros da Silva

Dissertação (mestrado) – UFRJ/COPPE/Programa de Engenharia Elétrica, 2011.

Referências Bibliográficas: p. 72 – 74.

1. MMP. 2. Imagens Estéreo. 3. 3-D. I. Silva, Eduardo Antônio Barros da. II. Universidade Federal do Rio de Janeiro, COPPE, Programa de Engenharia Elétrica. III. Título.

"Aquele que quiser ser o maior
que seja o menor e aquele que
serve a todos."
Marcos, 9-35

Agradecimentos

A Deus, o SENHOR, pela fidelidade, graça e amor na minha vida, e por me honrar e abençoar em tudo aquilo que eu me proponho a fazer em louvor ao Seu nome. Por estar comigo em todos os lugares, e por nunca deixar que me faltasse nada.

Aos meu pais, Izaura e Armínio, pelo exemplo de vida e dedicação aos filhos; por me darem a vida, e por serem anjos de Deus que cuidaram de mim.

Aos meus irmãos Hugo, Igor e Sílvia: os meus melhores amigos... as pessoas que estão sempre presentes, e com quem eu sempre vou poder contar (e contem sempre comigo também, meus irmãos). E também a nossa "sobrinha-filha" Agnes.

Ao meu orientador e amigo, professor Eduardo, por ter acreditado em mim; pelos esforços e contratempos que teve me ajudando em situações que, por muitas vezes, foram além da relação professor-aluno.

À Fundação de Amparo à Pesquisa do Estado do Amazonas, FAPEAM, pelo financiamento.

Aos amigos Henrique e Igor por abrirem as portas de casa a mim durante vários meses assim que cheguei à cidade (ainda sem bolsa de estudos). Sem vocês, o início deste mestrado não teria sido possível, e ao meu amigo-irmão Zé Maria por ser, mesmo do outro lado do Brasil, o amigo mais presente neste período.

Ao professor Eddie, que foi a primeira pessoa que acreditou e me encaminhou a fazer um curso de mestrado; pela ótima carta de referência que me foi um passaporte para ingressar na COPPE, e ao professor Waldir, que, durante o curso, não economizou em apoio, ajuda, broncas e conselhos sempre bem dados.

À família LPS; aos "nerds-gênios" deste laboratório que, sempre bem solícitos, compartilham os seus conhecimentos com todos que aqui chegam; pelos esclarecimentos e ajudas de toda hora, que levam a um crescimento profissional enorme, e por proporcionarem a cada dia o melhor ambiente de trabalho do mundo... não só com esforço e dedicação, mas com muita descontração também. Aos amigos de toda vida que fiz aqui, em especial Lucas, Tadeu, Gabriel e Geórgio.

Aos amigos da Primeira Igreja Batista de Manaus, Fernando (padrão espiritual), Gleydson e Luíz pela amizade íntegra que só agrega valores, bem como aos amigos da Igreja Metodista do Jardim Botânico, Lucas (de novo), Sarah, Thiago, Marcos, Guto (e Sabrina), e ao meu pastor de jovem, Douglas Marins.

Por fim, gostaria de dedicar este trabalho a minha imagem 3-D favorita: a você Aline.

Resumo da Dissertação apresentada à COPPE/UFRJ como parte dos requisitos necessários para a obtenção do grau de Mestre em Ciências (M.Sc.)

CODIFICAÇÃO DE TEXTURA E PROFUNDIDADE DE IMAGENS 3-D UTILIZANDO RECORRÊNCIA DE PADRÕES MULTIESCALAS

Anderson Vinícius Corrêa de Oliveira

Março/2011

Orientador: Eduardo Antônio Barros da Silva

Programa: Engenharia Elétrica

O MMP (Multi-dimensional Multiscale Parser) é um algoritmo de compressão que possui um comportamento universal. O sinal de entrada é dividido em vetores menores, e cada vetor do sinal original é aproximado por vetores de um dicionário adaptativo, que é atualizado através de concatenações de versões expandidas e contraídas dos vetores previamente codificados, que se tornam recorrentes durante a codificação. Os casamentos dos vetores de entrada com os vetores do dicionário são feitos seguindo um critério de controle.

Este trabalho faz uma investigação do desempenho do MMP aplicado a pares de imagens estereoscópicas: imagens de textura e imagens de profundidade.

Primeiramente, foi feita uma modificação no algoritmo de maneira que o par de imagens de profundidade e textura fosse codificado conjuntamente, utilizando o mesmo dicionário final da codificação da vista de profundidade como dicionário inicial da codificação da vista de textura. Em seguida, foi feita uma investigação sobre a melhor relação de compressão entre imagem de textura e imagem de profundidade, utilizando o MMP, com o objetivo de obter a melhor alocação de bits visando a reconstrução de vistas virtuais sintetizadas.

Os resultados mostram que o MMP atinge resultados comparáveis com o estado da arte na compressão de pares textura-profundidade.

Abstract of Dissertation presented to COPPE/UFRJ as a partial fulfillment of the requirements for the degree of Master of Science (M.Sc.)

CODING OF TEXTURE AND DEPTH OF 3-D IMAGES BASED ON RECURRENT MULTISCALE PATTERNS

Anderson Vinícius Corrêa de Oliveira

March/2011

Advisor: Eduardo Antônio Barros da Silva

Department: Electrical Engineering

The MMP (Multi-dimensional Multiscale Parser) is a signal compression algorithm that has a universal behavior. The input signal is segmented in smaller vectors, and each of these vectors is approximated by an adaptive dictionary, that is updated using expansions and contractions of previously encoded vectors, that become recurrebt during coding. The matchings of the input vectors with the dictionary vectors are performed according to a control criterion.

This work performs an investigation of the performance of the MMP when applied to pairs of stereoscopic images: texture and depth images.

First, the MMP algorithm has been modified so that the texture-depth pair could be jointly encoded, using the final dictionary of the depth encoded as the initial dictionary for encoding the texture. Then, it has been investigated the best compression ration between the depth and texture images using the MMP, with the aim of obtaining the optimum bit allocation regarding virtual view synthesis. Results show that MMP can achieve state-of-the-art results in the compression of texture-depth pair.

Sumário

Lista de Figuras **x**

Lista de Tabelas **xiv**

1 Introdução **1**

1.1 Introdução 1

1.2 Organização da dissertação 2

2 Compressão de dados **4**

2.1 Compressão sem perdas 5

2.2 Compressão com perdas 6

3 MMP **9**

3.1 Introdução 9

3.1.1 Inicialização do dicionário 10

3.1.2 Processo de codificação 14

3.1.3 Atualização do dicionário 16

3.1.4 Transformação de escala 18

3.2 Melhorias adicionadas ao MMP 20

3.2.1 Uso de técnicas de predição no MMP 20

3.2.2 Controle de crescimento do dicionário 22

3.2.3 Dicionário com segmentação flexível 23

4 Fundamentos de imagens estereoscópicas **25**

4.1 Introdução 25

4.1.1 Fundamentos da visualização estéreo 26

4.1.2 Modelo de câmera puntiforme 26

4.1.3 Geometria da representação do sistema 3-D 27

4.1.4 A geometria epipolar 29

4.2 Mapa de profundidade 31

4.3 Reconstrução com base na imagem de profundidade 33

5 Codificação conjunta das vistas de textura e profundidade 34
5.1 Motivação 34
5.2 Imagem de textura com dicionário do mapa de profundidades 35
5.2.1 Resultados 36
5.3 Mapa de profundidades com dicionário da imagem de textura 43
5.3.1 Resultados 44

6 Alocação ótima de bits entre textura e profundidade 50
6.1 Introdução 50
6.2 Descrição do experimento 51
6.3 Resultados 53

7 Alocação ótima de bits utilizando regiões de interesse no mapa de profundidades 64
7.1 Introdução 64
7.2 Descrição do experimento 65
7.3 Resultados 65

8 Conclusões 70
8.1 Trabalhos futuros 71

Referências Bibliográficas 72

A Pseudo-Código do MMP 75

B Imagens originais utilizadas 78
B.1 Imagens de textura 79
B.2 Imagens de profundidade 84

Lista de Figuras

2.1 Relação entre entropia e informação mútua. 7
2.2 Função taxa-distorção R(D) típica. 8

3.1 Representação do dicionário inicial com nove níveis, utilizando segmentação vertical. 11
3.2 Representação do dicionário inicial com nove níveis, utilizando segmentação horizontal. 12
3.3 *Elemento 0 do dicionário de nível 6.* 13
3.4 *Elemento 100 do dicionário de nível 6.* 13
3.5 *Elemento 250 do dicionário de nível 6.* 13
3.6 Divisão da imagem em blocos de dimensão 16 x 16 pixels. 14
3.7 Segmentação vertical de um bloco de dimensão 4 x 4. 15
3.8 Bloco de entrada de dimensão 8 x 8. 16
3.9 Blocos codificados. 17
3.10 Árvore binária associada aos blocos codificados. 17
3.11 Atualização dos dicionários. 18
3.12 Modos de predição. 20
3.13 Hiperesferas de raio d. 22
3.14 Exemplo de segmentação flexível para um bloco 16x16. 24

4.1 Modelo de camera puntiforme. 27
4.2 Sistema estéreo simples (rep. de [15], p. 143).. 28
4.3 Geometria epipolar (rep. de [15], p. 151).. 29
4.4 Imagem Champagne Tower, câmera 39, *frame* 0: a) Imagem de textura; b) Mapa de profundidades. Fonte: *Nagoya University* [29]. . . . 32

5.1 Esquema da codificação conjunta. 35
5.2 Ballet (img. de textura), cam. 03, *frame* 0, a partir da img. de profundidade, cam. 03, *frame* 0. 37
5.3 Breakdancers (img. de textura), cam 03, *frame* 0, a partir da img. de profundidade, cam. 03, *frame* 0. 37

5.4 Book Arrival (img. de textura), cam 08, *frame* 0, a partir da img. de profundidade, cam. 08, *frame* 0. 38
5.5 Champagne Tower (img. de textura), cam 39, *frame* 0, a partir da img. de profundidade, cam. 39, *frame* 0. 38
5.6 Pantomime - frame 0 - (img. de textura), cam 37, *frame* 0, a partir da img. de profundidade, cam. 37, *frame* 0. 39
5.7 Segmentação do mapa de profundidade em uma região da imagem book arrival, câmera 08, *frame* 0, codificado com Lambda=10. 40
5.8 Segmentação do mapa de profundidade em uma região da imagem book arrival, câmera 08, *frame* 0, codificado com Lambda=10, sobreposta sobre a imagem de textura. 40
5.9 Segmentação do mapa de profundidade em uma região da imagem pantomime, câmera 37, *frame* 0, codificado com Lambda=10. 41
5.10 Segmentação do mapa de profundidade em uma região da imagem pantomime, câmera 37, *frame* 0, codificado com Lambda=10, sobreposta sobre a imagem de textura. 41
5.11 Esquema da codificação conjunta (textura seguida de profundidade). . 43
5.12 Ballet (img. de profundidade), cam. 03, *frame* 0, a partir da img. de textura, cam. 03, *frame* 0. 44
5.13 Breakdancers (img. de profundidade), cam. 03, *frame* 0, a partir da img. de textura, cam. 03, *frame* 0. 44
5.14 Book Arrival (img. de profunidade), cam. 08, *frame* 0, a partir da img. de textura, cam. 08, *frame* 0. 45
5.15 Champagne Tower (img. de profundidade), cam. 39, *frame* 0, a partir da img. de textura, cam. 39, *frame* 0. 45
5.16 Pantomime - frame 0 - (img. de profundidade), cam. 37, *frame* 0, a partir da img. de textura, cam. 37, *frame* 0. 46
5.17 Segmentação de uma região de textura da imagem book arrival, câmera 08, *frame* 0, codificado com Lambda=10. 47
5.18 Segmentação de uma região de textura da imagem book arrival, câmera 08, *frame* 0, codificado com Lambda=10, sobreposta sobre o mapa de profundidade. 47
5.19 Segmentação de uma região de textura da imagem pantomime, câmera 37, *frame* 0, codificado com Lambda=10. 48
5.20 Segmentação de uma região de textura da imagem pantomime, câmera 37, *frame* 0, codificado com Lambda=10, sobreposta sobre o mapa de profundidade. 48

6.1 Representação de um cenário de múltiplas câmeras. 51

6.2 Esquema de reconstrução da vista central a partir de dois pares de imagens estéreo. 51
6.3 Sequências de teste. 52
6.4 Ballet, câmera 04, *frame* 0, (imagem virtual). 53
6.5 Breakdancers, câmera 04, *frame* 0, (imagem virtual). 54
6.6 Champagne tower, câmera 38, *frame* 0, (imagem virtual). 54
6.7 Pantomime, câmera 38, *frame* 0, (imagem virtual). 55
6.8 Ballet, câmera 04, *frame* 0, (imagem virtual). 56
6.9 Breakdancers, câmera 04, *frame* 0, (imagem virtual). 56
6.10 Champagne tower, câmera 38, *frame* 0, (imagem virtual). 57
6.11 Pantomime, câmera 38, *frame* 0, (imagem virtual). 57
6.12 Ballet, câmera 04, *frame* 0, (imagem virtual). 58
6.13 Breakdancers, câmera 04, *frame* 0, (imagem virtual). 59
6.14 Champagne tower, câmera 38, *frame* 0, (imagem virtual). 59
6.15 Pantomime, câmera 38, *frame* 0, (imagem virtual). 60
6.16 Ballet, câmera 04, *frame* 0, (imagem virtual). 61
6.17 Breakdancers, câmera 04, *frame* 0, (imagem virtual). 61
6.18 Champagne tower, câmera 38, *frame* 0, (imagem virtual). 62
6.19 Pantomime, câmera 38, *frame* 0, (imagem virtual). 62

7.1 Máscaras de bordas 65
7.2 Mapas de profundidades (vista esquerda) codificados com e sem *edge aware*: a) ballet; b) breakdancers; c) champagne tower; d) pantomime. 66
7.3 Ballet, câmera 04, com edge aware. 67
7.4 Breakdancers, câmera 04, com edge aware. 67
7.5 Champagne tower, câmera 38, com edge aware. 68
7.6 Pantomime, câmera 38, com edge aware. 68
7.7 Ballet, câmera 04 (imagem virtual); a) 0,3 bpp (menor taxa avaliada para esta imagem); b) 0,8 bpp (maior taxa avaliada para esta imagem) 69
7.8 Breakdancers, câmera 04 (imagem virtual); a) 0,5 bpp (menor taxa avaliada para esta imagem); b) 1,1 bpp (maior taxa avaliada para esta imagem) 69

B.1 Ballet, câmera 03, *frame* 0. Fonte: [27]. 79
B.2 Ballet, câmera 05, *frame* 0. Fonte: [27]. 79
B.3 Breakdancers, câmera 03, *frame* 0. Fonte: [27]. 80
B.4 Breakdancers, câmera 05, *frame* 0. Fonte: [27]. 80
B.5 Book arrival, câmera 08, *frame* 0. Fonte: [28]. 81
B.6 Book arrival, câmera 10, *frame* 0. Fonte: [28]. 81
B.7 Champagne tower, câmera 37, *frame* 0. Fonte: [29]. 82

B.8 Champagne tower, câmera 39, *frame* 0. Fonte: [29]. 82
B.9 Pantomime, câmera 37, *frame* 0. Fonte: [29]. 83
B.10 Pantomime, câmera 39, *frame* 0. Fonte: [29]. 83
B.11 Ballet, câmera 03, *frame* 0. Fonte: [27]. 84
B.12 Ballet, câmera 05, *frame* 0. Fonte: [27]. 84
B.13 Breakdancers, câmera 03, *frame* 0. Fonte: [27]. 85
B.14 Breakdancers, câmera 05, *frame* 0. Fonte: [27]. 85
B.15 Book arrival, câmera 08, *frame* 0. Fonte: [28]. 86
B.16 Book arrival, câmera 10, *frame* 0. Fonte: [28]. 86
B.17 Champagne tower, câmera 37, *frame* 0. Fonte: [29]. 87
B.18 Champagne tower, câmera 39, *frame* 0. Fonte: [29]. 87
B.19 Pantomime, câmera 37, *frame* 0. Fonte: [29]. 88
B.20 Pantomime, câmera 39, *frame* 0. Fonte: [29]. 88

Lista de Tabelas

5.1 Modos de predição usados por cada dicionário após a codificação da primeira imagem (profundidade) na sequência book arrival. 42
5.2 Modos de predição usados por cada dicionário após a codificação da primeira imagem (textura) na sequência book arrival. 49

6.1 Relação entre lambda de textura e lambda de profundidade 60

Capítulo 1

Introdução

1.1 Introdução

O desenvolvimento cada vez mais rápido de tecnologias da informação proporcionou o aumento do uso de conteúdos em formato digital. Os computadores pessoais e serviços de telefonia celular tornaram-se acessíveis a praticamente todos os setores da sociedade, permitindo a criação e troca de informações através de grandes redes.

Em particular, os conteúdos visuais são utilizados em muitas áreas; da ciência ao entretenimento. O desenvolvimento de tecnologias relacionadas a conteúdos audiovisuais torna viável o uso comercial e doméstico de câmeras digitais com resoluções cada vez maiores. Neste tipo de conteúdo, o volume de dados é muito grande, assim como a banda necessária para a transmissão deste tipo de informação. A preocupação em reduzir a quantidade de informação visual antecede esse aumento do uso de conteúdos digitais, e os primeiros trabalhos nesse sentido relacionam-se com o desenvolvimento de métodos para representação de vídeos analógicos para transmissão em larguras de bandas definidas pelos sistemas de televisão usados. Assim, os algoritmos de compressão de dados são de grande importância em aplicações envolvendo imagens digitais, tanto para transmissão através de um canal como também para armazenamento em discos.

O estado da arte em métodos de compressão de imagem e vídeo geralmente usam transformadas matemáticas, e por isso esses métodos pressupõe que as imagens possuem baixa frequência espacial, e, com a aplicação da transformada, a maior parte da informação está nos coeficientes de baixa frequência.

O Multidimensional Multiscale Parser (MMP) é um método de codificação utilizado inicialmente em imagens que apresenta comportamento universal, no sentido de que não é necessário nenhum conhecimento estatístico prévio para a fonte que se deseja codificar. O algoritmo é baseado no casamento aproximado de padrões recorrentes multiescalas, que codifica segmentos de um dado sinal utilizando versões

contraídas e expandidas de padrões armazenados em um dicionário, que é constantemente atualizado durante o processo de codificação. O MMP pode ser estendido para n dimensões e é adaptativo (já que o dicionário é atualizado conforme o sinal de entrada vai sendo codificado). Sendo assim, é possível aplicá-lo a vários tipos de sinais, indicando um grande potencial para compressão de dados.

No contexto de conteúdos visuais, destacam-se as imagens 3-D (ou imagens estéreo). Este tipo de imagem pressupõe pelo menos o dobro de informação em relação a uma imagem convencional, pois a percepção de profundidade causada por esse tipo de imagem ocorre porque a cena vista é formada por, pelo menos, duas imagens da mesma cena, vista de ângulos diferentes. O uso crescente de conteúdo 3-D no cinema e HDTV sugerem um uso ainda maior da tecnologia 3-D no futuro. Esse progresso e expansão da tecnologia 3-D leva à exploração de técnicas de codificação existentes, entre elas o algoritmo MMP, cujos resultados com imagens de textura apresentam resultados satisfatórios (comparado com outros métodos de compressão), independente das características da imagem. A aplicação do algoritmo MMP em imagens estéreo é um campo de investigação que permanece aberto.

Esta dissertação se propõe a fazer uma investigação sobre compressão de imagens 3-D utilizando o algoritmo MMP, explorando as dependências entre imagem de textura e imagem de profundidade, e generalizando por aproximação a alocação ótima de bits na síntese de imagens virtuais.

1.2 Organização da dissertação

Esta dissertação é formada por 8 capítulos, que estão organizados da seguinte forma:

No capítulo 2, é feita uma revisão bibliográfica sobre compressão de sinais, assunto onde esta dissertação está inserida, estabelecendo os principais conceitos e classificações sobre compressão, em especial a descrição dos conceitos de compressão sem perdas e compressão com perdas.

No capítulo 3, é feita uma revisão bibliográfica detalhada sobre o algoritmo MMP, o qual faz parte dos algoritmos de compressão desenvolvidos nesta dissertação. Primeiramente, é descrito o MMP original em sua primeira versão, que já utilizava a otimização taxa-distorção, já que a descrição desta proporciona didaticamente um entendimento melhor das principais modificações inseridas posteriormente, as quais serão descritas ainda neste capítulo.

Em seguida, no capítulo 4, são estabelecidos os principais conceitos sobre imagens estereoscópicas, abordando os modelos matemáticos que descrevem a geometria do sistema estéreo e variáveis matemáticas que relacionam as duas vistas.

No capítulo 5, são apresentadas as modificações propostas no algoritmo para aplicação específica em imagens estéreo de textura e profundidade. Nesta parte,

são descritas as motivações que levaram à realização de uma codificação conjunta de textura e profundidade no mesmo algoritmo, e são apresentados os resultados decorrentes desta modificação.

No capítulo 6, é descrita uma investigação experimental sobre a relação de codificação entre imagens de textura e profundidade utilizando o MMP que proporcione a melhor relação taxa-distorção da vista virtual sintetizada.

No capítulo 7, o experimento do capítulo 6 é refeito utilizando-se imagens de profundidade codificadas com o MMP com tratamento especial das bordas dessas imagens, e o efeito deste uso será então avaliado na criação da vista virtual.

Finalmente, no capítulo 8, são feitas as conclusões dos resultados obtidos nos experimentos propostos, e ainda são apresentadas sugestões para trbalhos futuros.

Os apêndices incluem o pseudo-código do MMP e as imagens utilizadas.

Capítulo 2

Compressão de dados

O conceito de *informação* abrange uma diversidade de significados, que vão do uso cotidiano ao uso como um termo técnico. O primeiro estudo desenvolvido no sentido de "medir" e avaliar a informação foi feita por *Claude Shannon* em [1], cujas pesquisas deram origem ao campo da ciência conhecido como *Teoria da Informação*, o qual estuda e permite formular modelos matemáticos que descrevem o processamento, manipulação e organização da informação.

Com o advento da "era da informação" (ou "era do conhecimento"), caracterizada primeiramente pela popularização dos computadores pessoais e acesso à *internet* por praticamente todas as classes sociais, cada vez mais informações estão sendo geradas e utilizadas em formato digital. O uso irrestrito destas informações, no entanto, ainda não é possível devido a limitações como espaço de armazenamento em dispositivos e tamanho da banda do canal utilizado para a transmissão destas informações. Desta situação, surge a necessidade da redução dos dados armazenados e trafegados, e criou-se um campo da teoria da informação chamado *compressão de dados*.

Dada uma entrada X, um algoritmo de codificação (chamado *codificador*) gera uma versão compacta de X, definida como $\hat{X}$, de maneira que uma quantidade menor de bits seja necessária para a sua representação. O algoritmo de decodificação (chamado *decodificador*) reconstrói $\hat{X}$, gerando uma saída Y, como uma versão aproximada (ou exata) de X.

De um modo geral, os códigos podem ser casados com uma fonte em particular (atingem o seu desempenho máximo de compressão apenas para aquela fonte) ou podem ser universais, quando o seu desempenho independe da fonte. Quando a distorção D entre a fonte original, X, e a saída reproduzida pelo decodificador, Y, é igual a zero, dizemos que há uma compressão sem perdas. Por outro lado, quando a distorção é diferente de zero, dizemos que há uma compressão com perdas.

2.1 Compressão sem perdas

Neste tipo de processo, a fonte original X pode ser recuperada completamente, sendo exatamente igual à saída Y do decodificador. Como exemplo da aplicação de compressão sem perdas estão os compressores de textos.

Dentre os métodos utilizados na compressão sem perdas, por exemplo, estão o código de Huffman [2], o codificador aritmético [3] e o algoritmo Lempel-Ziv [4] e [5].

Em [1], Shannon definiu o termo *auto-informação* da seguinte forma, para medir a quantidade de informação agregada a um evento A, associado a uma probabilidade de ocorrência $P(A)$:

$$i(A) = \log_2 \frac{1}{P(A)} \tag{2.1}$$

Pela equação 2.1, observa-se que a quantidade de informação de um evento é alta se a sua probabilidade de ocorrência for baixa, e vice-versa. Neste caso, a auto-informação $i(A)$ é medida em bits devido à base do logaritmo.

Seja S o espaço amostral de um conjunto de eventos disjuntos A_i, então a *auto-informação média* de S é dada por:

$$H(S) = \sum_{i=1}^{n} P(A_i) i(A_i)$$

que, substituindo-se pela equação 2.1, resulta em:

$$H(S) = -\sum_{i=1}^{n} P(A_i) \log_2 P(A_i) \tag{2.2}$$

O espaço amostral S é chamado *fonte*, os eventos A_i são chamados *símbolos*, e o conjunto de símbolos $\mathcal{A} = \{A_i\}$ é chamado *alfabeto* da fonte. À auto-informação média $H(S)$, definida na equação 2.2, damos o nome de *entropia*, e é assim definida para um alfabeto $\mathcal{A}$, cujas probabilidades dos símbolos A_i sejam independentes. A demosntração do cálculo da entropia para qualquer fonte S pode ser vista em detalhes em [6].

A quantidade $H(S)$ da equação 2.2 define o limite teórico que qualquer código que utiliza métodos sem perda pode alcançar ao codificar uma fonte, de maneira que, nos métodos desenvolvidos, busca-se obter códigos cuja taxa se aproxime ao máximo da entropia da fonte. Teoricamente, a taxa R tende à entropia quando o número de símbolos codificados tende ao infinito.

2.2 Compressão com perdas

Existem aplicações onde permite-se que a saída Y do decodificador apresente perdas de informação em relação à fonte X como, por exemplo, na codificação de imagens, vídeos e arquivos de áudio, onde a qualidade de aceitação está relacionada às limitações dos sistemas visual e auditivo humano. Neste tipo de processo, a compactação obtida para uma fonte é maior do que os métodos de compressão sem perdas.

Nos métodos de compressão com perdas, aparece outro fator importante além da taxa alcançada na compressão, que é a *distorção* D entre a saída Y do decodificador e a fonte X.

A teoria taxa-distorção estuda a compressão de uma fonte sujeita a um critério de fidelidade, de maneira que a função $R(D)$ estabelece a relação entre a taxa de codificação alcançada e a distorção resultante.

Uma vez que se permita algum tipo de perda de informação entre a fonte X e a saída Y, os alfabetos de entrada e saída podem ser diferentes. Seja X formada pelo alfabeto $\mathcal{X} = \{x_1, x_2, ..., x_n\}$, e Y formada pelo alfabeto $\mathcal{Y} = \{y_1, y_2, ..., y_m\}$, então para estabelecer uma relação entre X e Y, consideremos primeiramente a medida de entropia de cada uma delas, que, pela equação 2.2, são dadas por:

$$H(X) = -\sum_{i=1}^{n} P(x_i) \log_2 P(x_i)$$

$$H(Y) = -\sum_{j=1}^{m} P(y_j) \log_2 P(y_j) \tag{2.3}$$

A *entropia conjunta* de X e Y é dada pela equação:

$$H(X,Y) = H(X) + H(Y|X) = H(Y) + H(X|Y), \tag{2.4}$$

onde os termos $H(X|Y)$ e $H(Y|X)$ recebem o nome de *entropia condicional*.

Para qualquer valor fixo $X = x_i$, dada a distribuição condicional de probabilidade $P(Y|X = x_i)$ de Y, dado que X é conhecido, a entropia condicional é dada por:

$$H(Y|X) = \sum_{i=1}^{n} P(x_i) H(Y|X = x_i)$$

$$H(Y|X) = -\sum_{i=1}^{n} \sum_{j=1}^{m} P(y_j|x_i) P(x_i) \log_2 P(y_j|x_i) \tag{2.5}$$

Analogamente,

$$H(X|Y) = \sum_{j=1}^{m} P(y_j) H(X|Y = y_j)$$

$$H(X|Y) = -\sum_{i=1}^{n}\sum_{j=1}^{m} P(x_i|y_j)P(y_j)\log_2 P(x_i|y_j) \tag{2.6}$$

A *entropia mútua* $I(X;Y)$ entre X e Y é definida como:

$$I(X,Y) = H(Y) - H(Y|X) = H(X) + H(X|Y) \tag{2.7}$$

As demonstrações das equações 2.4, 2.5 e 2.7 podem ser vistas com detalhes em [6], e o significado dos conceitos de entropia conjunta, entropia condicional e informação mútua podem ser visualizadas no esquema representado na figura 2.1:

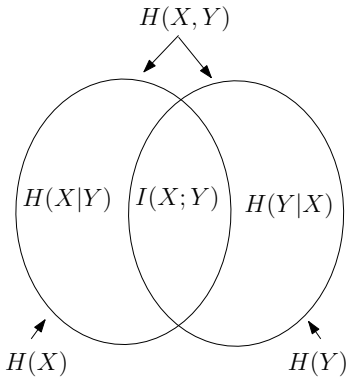


Figura 2.1: Relação entre entropia e informação mútua.

A função taxa-distorção, $R(D)$, estabelece o número médio de bits necessários para representar qualquer saída produzida pela fonte com distorção média menor ou igual a D.

Em [1], foi demonstrado que, em um esquema de compressão ótimo para uma fonte X, com distribuição de probabilidade $P(x_i)$, com uma saída Y que assume valores do alfabeto de saída $\{y_i\}$, e uma medida de distorção $d(x_i, y_i)$ entre X e Y, então a taxa mínima alcançada para uma distorção alvo D^* será igual ao mínimo da informação mútua entre X e Y, de maneira que:

$$R(D) = \min_{P(y_j|x_i)\,|\,D\leq D^*} I(X;Y) \tag{2.8}$$

onde D é a distorção média, dada por

$$D = \sum_{i}^{n}\sum_{j}^{m} P(x_i)P(y_i|x_i)d(x_i, y_j)$$

A função $R(D)$, definida na equação 2.8, estabelece o limite de desempenho de qualquer código de compressão de dados, ou seja, qualquer taxa alcançada pelo código será maior ou igual a $R(D)$.

O gráfico mostrado na figura 2.2 exemplifica o comportamento da função $R(D)$:

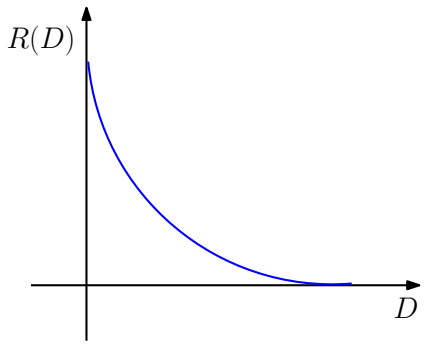


Figura 2.2: Função taxa-distorção R(D) típica.

Na função R(D), há dois extremos: o primeiro é definido para taxa igual a zero, ou seja, quando não se transmite informação nenhuma, o segundo ocorre quando a distorção D é igual a zero, ou seja, quando a informação é transmitida sem perdas. A teoria taxa-distorção estuda o comportamento entre estes dois extremos, e estabelece o limite de desempenho para a relação entre taxa e distorção.

Capítulo 3

MMP

3.1 Introdução

O algoritmo MMP (*Multidimensional Multiscale Parser*) é um esquema de compressão de dados com perdas, introduzido em [7], e seu funcionamento tem por base o casamento aproximado de padrões recorrentes multiescalas. Este algoritmo parte de um conjunto de dicionários iniciais de vetores, $\mathcal{D}$, que inicialmente possuem amostras uniformes com todas as dimensões de blocos possíveis de ocorrer durante o processo de codificação. O algoritmo opera particionando o sinal de entrada em vetores menores, e cada um destes é codificado sequencialmente a partir de um casamento aproximado do vetor de entrada com um dos elementos do dicionário, seguindo um critério de casamento estabelecido.

Quando um casamento é realizado, o dicionário é atualizado, e um novo elemento é então introduzido ao dicionário. Em seguida, são usadas transformações de escalas, e este novo elemento é então adicionado a todos os dicionários com diferentes dimensões.

O algoritmo de codificação gera uma sequência de saída com *flags* que contém as informações sobre o elemento do dicionário utilizado para aproximar o bloco de entrada, e esta sequência de saída é processada por um codificador aritmético. O decodificador do MMP constrói o mesmo dicionário inicial do codificador. A partir das informações dos *flags*, o decodificador constrói, bloco a bloco, a imagem aproximada, de maneira que não é necessário o envio do dicionário gerado pelo codificador.

O algoritmo MMP é adaptativo, e o dicionário "aprende" dinamicamente à medida que o sinal de entrada é codificado. Por isso, o MMP possui um comportamento universal, pois não necessita de nenhum conhecimento prévio do sinal de entrada. Diferente de outros métodos de compressão, o MMP não utiliza transformadas matemáticas, como a FFT (discrete Fourier transform) e a DCT (discrete cosine trans-

form), e por isso a sua aplicação não está restrita a imagens suaves (com baixa frequência espacial), onde a maior parte da energia fica acumulada em poucos coeficientes após a aplicação da transformada, sendo que os codificadores convencionais fazem uso dessa propriedade para obter boas taxas de compressão.

Além disso, por ser um algoritmo multidimensional, o uso do MMP não está relacionado apenas a imagens, mas o seu uso pode ser estendido a qualquer sinal n-dimensional. As suas aplicações incluem imagens estáticas, vídeos e sinais de áudio.

3.1.1 Inicialização do dicionário

Inicialmente, o algoritmo cria uma *família* padrão de dicionários, independente do tipo de imagem a ser processada, o qual contém vetores de todas as dimensões possíveis de ocorrer no processo de codificação. Após a leitura do cabeçalho da imagem original, o algoritmo define o valor mínimo de luminância possível de um pixel ($min\_pixel \in \mathbb{Z}$). Este será sequencialmente incrementado por um valor fixo ($step$) até que se atinja o valor máximo de luminância de um pixel ($max\_pixel \in \mathbb{Z}$) que pode ser utilizado.

A família inicial de dicionários, $\mathcal{D}_0$, é definida por:

$$\mathcal{D}_0 = \{S_0, S_1, ..., S_{M-1}\},$$

onde:

- $M = 2\log_2(Lb) + 1$ é o número de dicionários $\mathcal{S}_i$ que compõe o dicionário inicial, e L_b é o tamanho do bloco de entrada.

- $S_i = \{s_i^0, s_i^1, ..., s_i^{P-1}\}$ é o dicionário de ordem i que contém P vetores s_i^j de dimensão $N \times M$, e $P = max\_pixel - min\_pixel + 1$.

A figura 3.1 mostra o esquema do dicionário inicial para blocos entrada de dimensão Lb 16×16. O dicionário tem nove níveis (de 0 a 8).

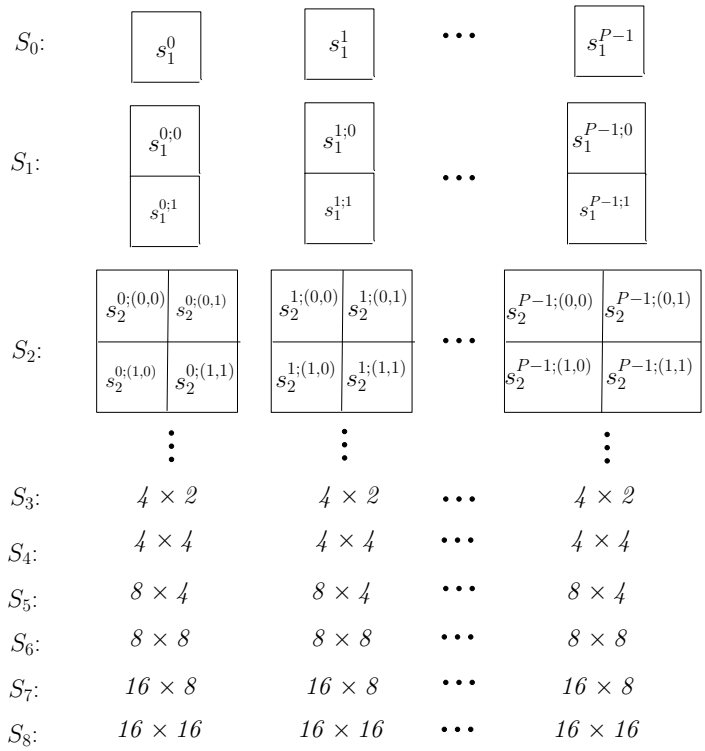


Figura 3.1: Representação do dicionário inicial com nove níveis, utilizando segmentação vertical.

Na figura 3.1, os elementos do dicionário de nível 8 têm a dimensão máxima Lb do bloco de entrada, neste caso 16×16. Nesta representação, o dicionário foi construído partindo de uma segmentação vertical, por isso os elementos do dicionário de ordem ímpar possuem dimensão retangular cuja proporção é $2:1$. A segmentação inicial, entretanto, também pode ser realizada na direção horizontal. Neste caso, os elementos do dicionário de ordem ímpar possuirão dimensão cuja proporção é $1:2$. Uma vez estabelecida a direção de segmentação inicial, prossegue-se dividindo o bloco em direções alternadas (vertical seguida de horizontal ou vice-versa) até a chegada ao nível 0, constituído de elementos de tamanho 1×1, ou seja, um único pixel. Assim, os elementos do dicionário S_0 são uma coleção de pixels com todos os valores possíveis na imagem.

A figura 3.2 mostra o mesmo esquema partindo de uma segmentação horizontal do bloco 16×16.

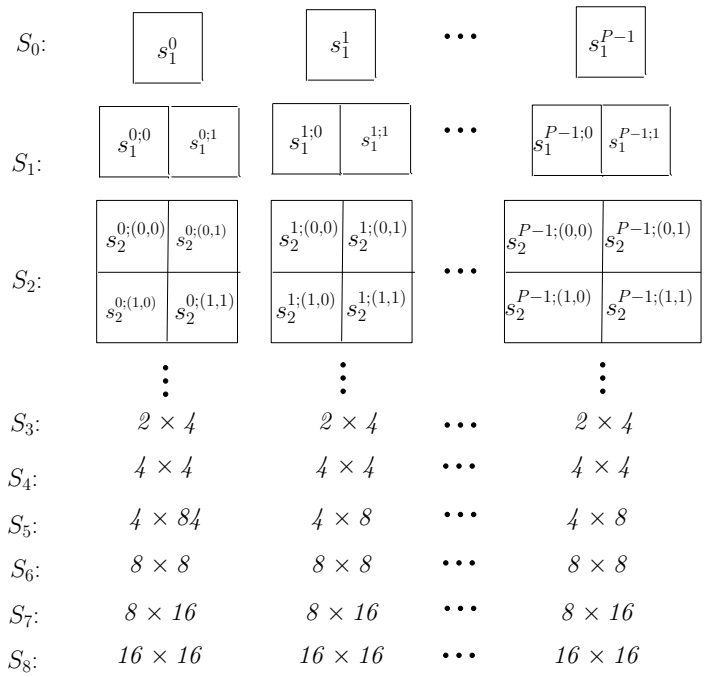


Figura 3.2: Representação do dicionário inicial com nove níveis, utilizando segmentação horizontal.

Em cada dicionário $\mathcal{S}_i$, o primeiro elemento possui todos os pixels com valores iguais a min_pixel, o segundo elemento com pixels iguais a $min\_pixel + step$, até o último elemento s_i^{P-1}, com pixels iguais a max_pixel.

As figuras 3.3, 3.4 e 3.5 ilustram, respectivamente, os elementos de índices 0, 100 e 250, pertencentes ao dicionário de nível 6 dos esquemas represetados nas figuras 3.1 e 3.2.

Como parâmetro de entrada, definido no início da codificação, é escolhida a dimensão Lb dos blocos em que a imagem será pariticionada. Cada bloco $Lb \times Lb$ da imagem original será o vetor de entrada da codificação.

0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0

(a)

(b)

Figura 3.3: *Elemento 0 do dicionário de nível 6.*

100	100	100	100	100	100	100	100
100	100	100	100	100	100	100	100
100	100	100	100	100	100	100	100
100	100	100	100	100	100	100	100
100	100	100	100	100	100	100	100
100	100	100	100	100	100	100	100
100	100	100	100	100	100	100	100
100	100	100	100	100	100	100	100

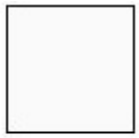
(a)

(b)

Figura 3.4: *Elemento 100 do dicionário de nível 6.*

250	250	250	250	250	250	250	250
250	250	250	250	250	250	250	250
250	250	250	250	250	250	250	250
250	250	250	250	250	250	250	250
250	250	250	250	250	250	250	250
250	250	250	250	250	250	250	250
250	250	250	250	250	250	250	250
250	250	250	250	250	250	250	250

(a)



(b)

Figura 3.5: *Elemento 250 do dicionário de nível 6.*

3.1.2 Processo de codificação

Assim que o dicionário inicial é construído, o algoritmo divide a imagem em blocos quadrados, com lados de dimensão Lb pixels. Cada bloco é codificado individualmente de maneira sequencial, da esquerda para a direita, até que a imagem esteja totamente codificada. A Figura 3.6 mostra uma imagem de textura dividida em blocos 16×16, com destaque para o processamento do bloco referente à região de um dos olhos.

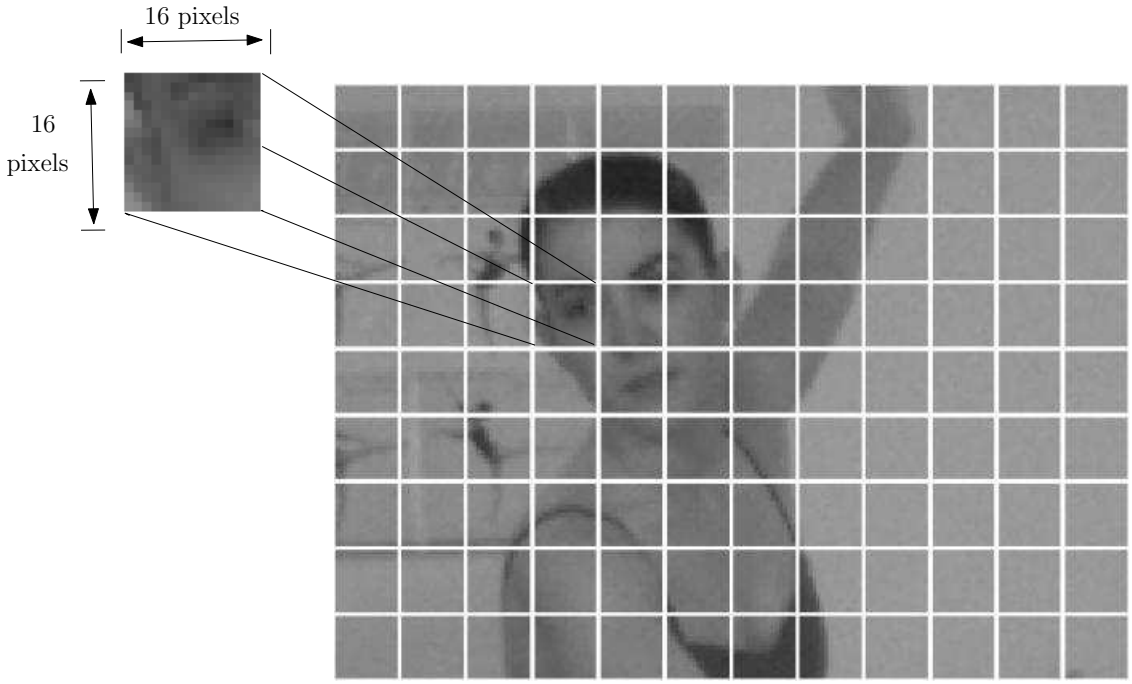


Figura 3.6: Divisão da imagem em blocos de dimensão 16 x 16 pixels.

Para codificar um bloco de dimensão $N \times M$, o MMP procura no dicionário de mesma dimensão $N \times M$ pelo elemento que proporcione a melhor aproximação com o bloco, de acordo com o critério de controle estabelecido. Neste projeto, este controle representa a relação entre taxa e distorção, que são controladas pelo multiplicador de Lagrange λ, de acordo com a função custo, definida como:

$$J = D + \lambda R, \tag{3.1}$$

onde D é a distorção entre a aproximação do elemento do dicionário e o bloco que está sendo codificado. É obtida calculando-se o erro quadrático entre o elemento do dicionário e o bloco da imagem. R é a taxa que corresponde ao número de bits gastos para representar o índice do vetor no dicionário juntamente com o *flag* que indica se houve ou não divisão do bloco, e λ é uma constante definida no início do processo de codificação, sendo um parâmetro de entrada do algoritmo.

Como o objetivo é minimizar o custo J, os valores elevados de λ irão priorizar

casamentos que resultem em taxas baixas (e consequentemente distorções maiores). Por outro lado, os valores baixos de λ irão priorizar casamentos que resultem em distorções pequenas (e assim uma quantidade maior de bits).

Dado o primeiro bloco, de dimensão $Lb \times Lb$, o algoritmo irá procurar no dicionário S_i de maior dimensão pelo elemento s_i^j que minimiza o custo Lagrangeano da representação do bloco $\mathbf{x^0}$ por s_i^j. O bloco $\mathbf{x^0}$ é então segmentado em dois sub-blocos $\mathbf{x^1}$ e $\mathbf{x^2}$ com a mesma dimensão entre si e com a metade do tamanho de $\mathbf{x^0}$, de maneira que $\mathbf{x^1} \bigcup \mathbf{x^2} = \mathbf{x^0}$. O algoritmo tenta agora aproximar $\mathbf{x^1}$ e $\mathbf{x^2}$ por elementos do dicionário S_{i-1} que proporcionem a melhor aproximação com cada um dos dois sub-blocos, $\mathbf{x^1}$ e $\mathbf{x^2}$. Este procedimento é aplicado recursivamente a cada um dos sub-blocos obtidos até a chegada ao nível 0 da figura 3.1, o qual é representado por elementos de dimensão 1×1. A figura 3.7 exemplifica este procedimento para segmentação vertical de um bloco de entrada de dimensão 4×4.

As partições dos blocos podem ser associadas a uma árvore binária de segmentação, percorrida da esquerda para a direita, onde um bloco maior é representado como um "nó pai", e dois blocos menores, resultantes do bloco maior, representados como dois "nós filho".

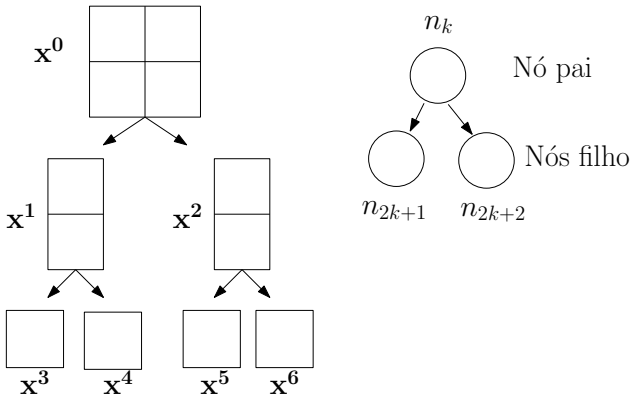


Figura 3.7: Segmentação vertical de um bloco de dimensão 4 x 4.

Uma vez obtida a árvore cheia, o algoritmo irá agora encontrar a árvore ótima, que melhor representa o bloco original.

Seja J_P o custo associado ao nó pai e J_F o custo total dos nós filho, incluindo a taxa que indica se houve segmentação, A codificação de n_k é feita se a soma do custo J_F associada aos dois nós filho n_{2k+1} e n_{2k+2} for maior do que o custo J_P da representação do nó n_k. Neste caso, a árvore é cortada, e o bloco $\mathbf{x^k}$ é representado pelo elemento s_i^j, de mesma dimensão que $\mathbf{x^k}$. Um *flag* de valor "1" é enviado a saída do codificador para indicar que houve casamento, juntamente com um *flag* representando o índice j do dicionário. Caso contrário, decide-se por não cortar a árvore, já que para este nó a divisão é a melhor escolha, significando que o bloco representado, $\mathbf{x^k}$, deve ser segmentado. É, então, enviado um *flag* de valor "0" para

indicar que houve segmentação.

Este processo é realizado a partir dos "nós folha", aqueles pertencentes ao nível com a menor dimensão (blocos 1×1), e prossegue até que a árvore esteja completamente percorrida, no nó raiz.

3.1.3 Atualização do dicionário

Uma particularidade do MMP é o fato de que o dicionário inicial, descrito na seção 3.1.1, não é estático. Uma vez realizado o casamento de um bloco, o dicionário é atualizado e novos elementos são incorporados aos dicionários. Uma consequência deste processo é que o algoritmo é capaz de "aprender" padrões que já ocorreram durante o processo de codificação, explorando as redundâncias da própria imagem uma vez que estes se tornem recorrentes.

Para ilustrar o procedimento de atualização, vamos exemplificar a codificação do bloco de entrada de dimensão 8×8, representado na figura 3.8, partindo de uma segmentação inicial vertical. O algoritmo procura no dicionário $\mathcal{S}_6$ da figura 3.1 pelo elemento s_6^j que proporcione o melhor casamento com este bloco, de acordo com a função custo definida na equação 3.1.

$\mathbf{x_{00}}$	$\mathbf{x_{01}}$	$\mathbf{x_{02}}$	$\mathbf{x_{03}}$	$\mathbf{x_{04}}$	$\mathbf{x_{05}}$	$\mathbf{x_{06}}$	$\mathbf{x_{07}}$
$\mathbf{x_{10}}$	$\mathbf{x_{11}}$	$\mathbf{x_{12}}$	$\mathbf{x_{13}}$	$\mathbf{x_{14}}$	$\mathbf{x_{15}}$	$\mathbf{x_{16}}$	$\mathbf{x_{17}}$
$\mathbf{x_{20}}$	$\mathbf{x_{21}}$	$\mathbf{x_{22}}$	$\mathbf{x_{23}}$	$\mathbf{x_{24}}$	$\mathbf{x_{25}}$	$\mathbf{x_{26}}$	$\mathbf{x_{27}}$
$\mathbf{x_{30}}$	$\mathbf{x_{31}}$	$\mathbf{x_{32}}$	$\mathbf{x_{33}}$	$\mathbf{x_{34}}$	$\mathbf{x_{35}}$	$\mathbf{x_{36}}$	$\mathbf{x_{37}}$
$\mathbf{x_{40}}$	$\mathbf{x_{41}}$	$\mathbf{x_{42}}$	$\mathbf{x_{43}}$	$\mathbf{x_{44}}$	$\mathbf{x_{45}}$	$\mathbf{x_{46}}$	$\mathbf{x_{47}}$
$\mathbf{x_{50}}$	$\mathbf{x_{51}}$	$\mathbf{x_{52}}$	$\mathbf{x_{53}}$	$\mathbf{x_{54}}$	$\mathbf{x_{55}}$	$\mathbf{x_{56}}$	$\mathbf{x_{57}}$
$\mathbf{x_{60}}$	$\mathbf{x_{61}}$	$\mathbf{x_{62}}$	$\mathbf{x_{63}}$	$\mathbf{x_{64}}$	$\mathbf{x_{65}}$	$\mathbf{x_{66}}$	$\mathbf{x_{67}}$
$\mathbf{x_{70}}$	$\mathbf{x_{71}}$	$\mathbf{x_{72}}$	$\mathbf{x_{73}}$	$\mathbf{x_{74}}$	$\mathbf{x_{75}}$	$\mathbf{x_{76}}$	$\mathbf{x_{77}}$

Figura 3.8: Bloco de entrada de dimensão 8 x 8.

O bloco 8×8 é particionado em dois blocos 8×4, e, para cada um destes, o algoritmo procura no dicionário $\mathcal{S}_5$ da figura 3.1 pelo elemento s_5^j que melhor represente cada bloco. O procedimento é repetido até que a árvore de segmentação alcance os blocos 1×1. A árvore ótima é obtida de acordo com o procedimento descrito na seção 3.1.2.

Suponhamos que a obtenção da árvore ótima gere os seguintes blocos particionados, representados na figura 3.9.

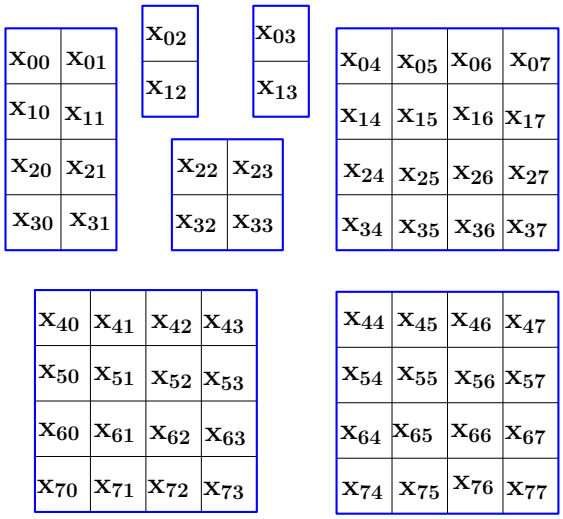


Figura 3.9: Blocos codificados.

Estes blocos foram codificados com sucesso por elementos do dicionário de dimensão correspondente a cada um deles. Esta partição gera a árvore binária representada na figura 3.10, onde os nós preenchidos de preto correspondem aos blocos codificados da figura 3.9.

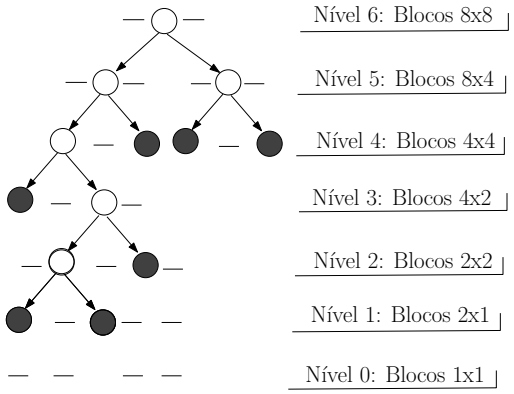


Figura 3.10: Árvore binária associada aos blocos codificados.

Quando dois nós correspondentes a dois blocos de mesma dimensão, e filhos do mesmo nó pai, forem codificados, dizemos que estes dois nós estão disponíveis, e os dois blocos, pertencentes ao dicionário $\mathcal{S}_i$, são concatenados, gerando um bloco maior de nível $i+1$. Este bloco resultante é então incorporado ao dicionário $\mathcal{S}_{i+1}$. Em seguida, o novo bloco é adicionado aos elementos de todos os outros dicionários com diferentes dimensões do novo bloco através de uma transformação de escala, que será descrita na seção 3.1.4. A figura 3.11 ilustra a atualização do dicionário com a concatenação dos dois blocos 2×1 da figura 3.9 (os quais, na árvore binária da figura 3.10, representam os nós de nível 1). No exemplo, os blocos 2×1 são concatenados, formando um bloco 2×2, e, em seguida, um bloco 4×2 e outro 4×4.

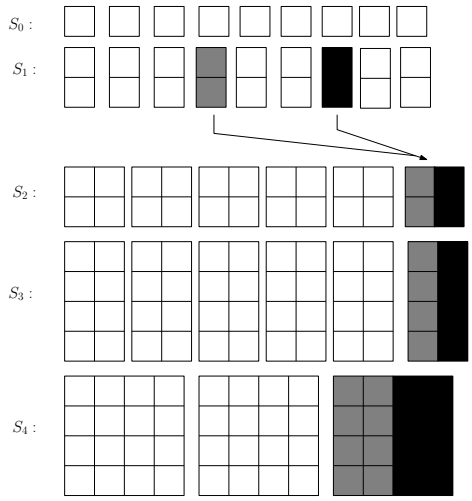


Figura 3.11: Atualização dos dicionários.

3.1.4 Transformação de escala

Os blocos codificados, depois de concatenados, são adicionados aos elementos do dicionário de dimensão correspondente, e, em seguida, são adicionados a todos os outros dicionários. Para isso, é necessária a realização de uma transformação de escala, que contrai ou dilata o novo bloco adicionado a partir de um operador que associa dois operadores de tamanhos distintos.

A transformação de escala bidimensional $T_{N.M}[X]$ de um vetor X de dimensão $N \times M$ usada na atualização do dicionário é realizada aplicando-se uma transformação de escala unidimensional $T_M[\cdot]$ a todas as linhas de X, e em seguida outra transformação unidimensional $T_N[\cdot]$ às M colunas do vertor resultante da primeira transformação.

As transformações unidimensionais são operações de mudanças de taxas de amostragem. Seja S um vetor de tamanho N_0, pode-se obter um vetor S^S de tamanho N a partir das seguintes operações, descritas em [8], quando:

- $N > N_0$

$$S_n^s = \left\lfloor \frac{\alpha_n \left(S_{m_n^1} - S_{m_n^0} \right)}{N} \right\rfloor + S_{m_n^0}, \quad n = 0, 1, ..., N-1 \tag{3.2}$$

onde:

$$\alpha_n = n\left(N_0 - 1\right) - N m_n^0$$

$$m_n^0 = \left\lfloor \frac{n(N_0-1)}{N} \right\rfloor$$

$$m_n^1 = \begin{cases} m_n^0 + 1 & , \quad m_n^0 < N_0 - 1 \\ m_n^0 & , \quad m_n^0 = N_0 - 1 \end{cases}$$

A equação 3.2 muda o tamanho de $N_0 N$ usando um interpolador linear como filtro. Em seguida, é feita a redução da taxa de amostragem por N_0.

- $N < N_0$

$$S_n^s = S_{m^0_{n,k=0}} + \frac{1}{N_0+1} \sum_{k=0}^{N_0} \left\lfloor \frac{\alpha_{n,k} \left(S_{m^1_{n,k}} - S_{m^0_{n,k}} \right)}{N} \right\rfloor, \quad n = 0, 1, ..., N-1 \quad (3.3)$$

onde:

$$\alpha_{n,k} = n\left(N_0 - 1\right) + k - N m^0_{n,k}$$

$$m^0_{n,k} = \left\lfloor \frac{n(N_0-1)+k}{N} \right\rfloor$$

$$m^1_{n,k} = \begin{cases} m^0_{n,k} + 1 & , \quad m^0_{n,k} < N_0 - 1 \\ m^0_{n,k} & , \quad m^0_{n,k} = N_0 - 1 \end{cases}$$

A equação 3.3 muda o tamanho de $N_0 N$ usando um interpolador linear como filtro. Em seguida, é aplicado um filtro de média de tamanho $N_0 + 1$, e então é feita a redução da taxa de amostragem por N_0.

3.2 Melhorias adicionadas ao MMP

A seguir, serão descritas as principais modificações adicionados ao algoritmo MMP original, com o objetivo de melhorar o seu desempenho. Os métodos adicionados, descritos a seguir, compõe o algoritmo base utilizado como ponto de partida para os testes realizados neste trabalho.

3.2.1 Uso de técnicas de predição no MMP

Com o objetivo de melhorar o desempenho do MMP para a codificação de imagens suaves, foi adicionado em [9] um método de predição intra-frame tal como utilizado no padrão H.264/AVC [10]. O uso de técnicas de predição possui a característica de gerar resíduos, cuja função de distribuição de probabilidade apresenta picos em torno de zero [11].

Os modos de predição utilizados são os mesmos usados no padrão H.264/AVC, com excessão do modo de predição 2 (modo DC), e do modo 9, introduzido em [12]. O modo 2 foi substituído pelo modo MFV (*Most Frequent Value*), o qual seleciona entre os pixels vizinhos aquele que ocorre com maior freqência, e o modo 9 é chamado LSP (*least-squares prediction*).

A figura 3.12 mostra, respectivamente, os esquemas dos modos de predição 0, 1, 3 e 4, e a representação direcional dos modos usados.

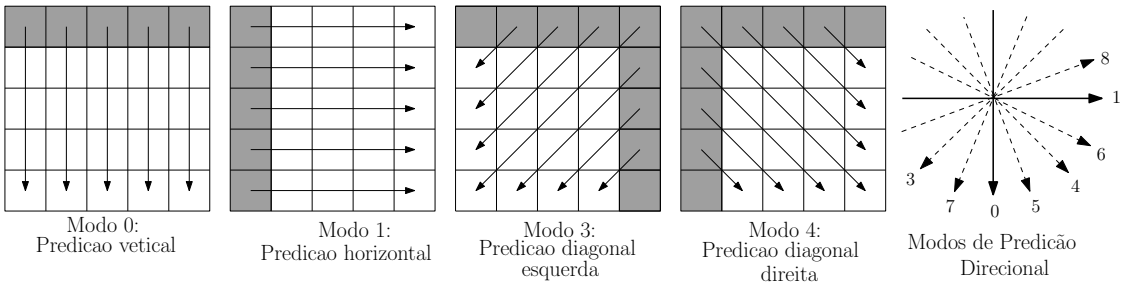


Figura 3.12: Modos de predição.

Ao codificar um bloco de entrada $\mathbf{x}^i$, o algoritmo escolhe o modo de predição que apresente o menor custo J, utilizando pixels de blocos vizinhos já codificados (acima e/ou a esquerda do bloco atual), para gerar um bloco de predição $\mathbf{p}_i^M$, onde i é o nível do dicionário e M é o modo de predição. O bloco de predição gerado é, então, subtraído do bloco de entrada original $\mathbf{x}^i$, gerando um resíduo $\mathbf{r}_i^{PM}$, o qual será codificado pelo MMP.

O modo de predição escolhido é enviado ao decodificador através do *flag de predição*. Dessa forma, o decodificador é capaz de determinar o mesmo bloco de predição $\mathbf{p}_i^M$. Após isso, o bloco de resíduo é decodificado, e a reconstrução $\hat{\mathbf{x}}^i$ do bloco de entrada $\mathbf{x}^i$ é determinada a partir de:

$$\hat{\mathbf{x}}^i = \mathbf{p}_i^M + \mathbf{r}_i^{PM} \tag{3.4}$$

No procedimento descrito na seção 3.1.2, o casamento era feito entre os blocos da imagem e os elementos do dicionário. Agora o algoritmo realiza um casamento entre os blocos de resíduo da imagem original e os elementos do dicionário.

Tal como ocorria no processo de codificação descrito na seção 3.1.2, a codificação de resíduos também é realizada de forma hierárquica. Primeiramente, o processo é feito para o bloco de entrada maior (geralmente 16×16), escolhendo-se o modo de predição cujo bloco de resíduo gerado apresente o melhor compromisso taxa-distorção, sendo que agora o custo J inclui os *flags de predição*. Em seguida, o bloco é segmentado em dois, e o procedimento é repetido para cada sub-bloco gerado, construindo-se a árvore de segmentação binária, e determinando-se a árvore ótima, assim como era realizado no algoritmo original.

O uso de técnicas de predição para a codificação de imagens no MMP traz uma consequência imediata, que é a existência de blocos com valores negativos de pixel, uma vez que o bloco de resíduo é obtido pela subtração de dois blocos. Por isso, o dicionário deve ter elementos que sejam capazes de representar tais valores, significando que a faixa de valores que vai do maior ao menor pixel é o dobro da faixa de valores descrita na seção 3.1.1.

Os blocos de resíduos têm a tendência de serem constituídos por valores de pixels próximos de zero. Por isso, é importante que agora existam muito mais elementos do dicionário com valores próximos de zero do que elementos com valores afastados. Por isso, o valor do *step* fixo, referido na seção 3.1.1, utilizado para a inicialização do dicionário, foi adaptado de maneira que apresentasse valores baixos, próximos de zero, aumentando conforme ocorre o afastamento em relação a esse valor.

Por heurística, determinou-se em [13] os seguintes valores para o *step* na inicialização do dicionário:

$$\begin{aligned}
&step = 2, && \text{se } |pixel| \leq 2 \\
&step = 4, && \text{se } 10 < |pixel| \leq 22 \\
&step = 8, && \text{se } 22 < |pixel| \leq 86 \\
&step = 13, && \text{se } |pixel| > 86
\end{aligned}$$

3.2.2 Controle de crescimento do dicionário

No exemplo da figura 3.10, dois blocos 2×1 são concatenados e adicionados ao dicionário S_2 da figura 3.1 no processo de atualização do MMP. Suponhamos que o casamento tenha sido feito em um dos blocos maiores. No processo de atualização, quando o bloco maior for "contraído" para a atualização de um dicionário com escala menor, é possível que o novo elemento inserido já esteja presente neste dicionário.

Além disso, como cada bloco codificado é atualizado em todos os dicionários S_i, como descrito na seção 3.1.3, o número final de blocos presentes em cada dicionário é muito maior do que o número de blocos utilizados. Cada novo elemento adicionado requer um novo índice no dicionário correspondente, e o crescimento desordenado do dicionário aumenta a entropia média dos símbolos e prejudica o desempenho do algoritmo, uma vez que aumenta a taxa de codificação.

Em [13], foi desenvolvido um método de controle de redundância entre os elementos do dicionário. Neste método, cada novo bloco $\hat{\mathbf{x}}^i$, de escala i, é testado com todos os elementos do dicionário S_i, e é adicionado a este somente se a distorção entre $\hat{\mathbf{x}}^i$ e todos os outros elementos de S_i for maior do que um limiar d^2, definido no início do algoritmo, de modo que:

$$d^2 = \sum_{m,n} = (\hat{\mathbf{x}}^i(m,n) - s_i^j(m,n))^2 \tag{3.5}$$

Este método de controle de redundância pode ser entendido como um empacotamento de hiperesferas de raio d que garantem uma condição de distorção mínima entre os elementos de um dicionário.

A figura 3.13 mostra o espaço particionado em hiperesferas de raio d, que garantem a condição de distância mínima entre os vetores do dicionário.

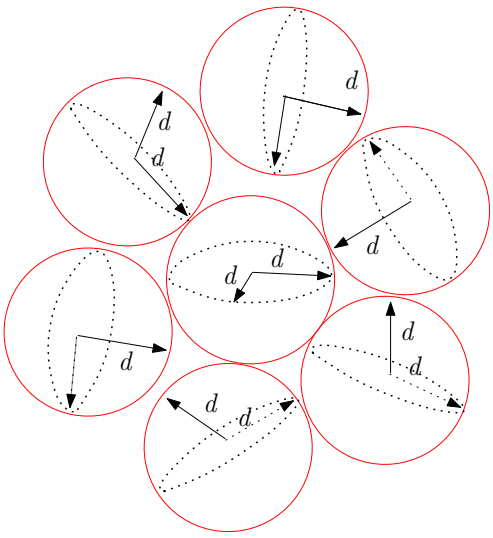


Figura 3.13: Hiperesferas de raio d.

Valores elevados de d criam grandes regiões vazias no espaço entre as esferas da figura, e essas regiões correspondem a padrões que não serão representados com boa qualidade. Por outro lado, quando o raio da hiperesfera é muito pequeno, os vetores do dicionário serão muito próximos um ao outro e o controle de crescimento do dicionário pode não ser satisfatório.

A melhor escolha de d está relacionada à taxa desejada. Para taxas altas (onde se obtém imagens com boa qualidade), deve-se escolher valores pequenos para d, e assim permitir-se uma maior quantidade de elementos no dicionário a fim de que os diversos padrões sejam representados da melhor maneira possível. Para baixas taxas de compressão, onde se permite representações com perdas maiores, o ideal é escolher-se valores altos de d, que proporcionará menos índices novos no dicionário, e assim menor taxa de codificação.

Por heurística, determinou-se a seguinte relação entre os valores de d e λ da equação 3.1, usada como parâmetro de entrada da relação taxa-distorção:

$$\begin{aligned} &5, && \lambda < 25 \\ &10, && 25 \leq \lambda < 75 \\ &20, && 75 \leq \lambda \leq 500 \\ &50, && \lambda > 500 \end{aligned}$$

Esta foi a relação usada neste projeto e é diferente daquela descrita em [13].

3.2.3 Dicionário com segmentação flexível

No algoritmo MMP original, a imagem era dividida em blocos de dimensão $Lb \times Lb$, e, no processo de codificação, a divisão dos blocos podia ser realizada tanto na direção vertical como na horizontal, sendo que a direção inicial era definida em um parâmetro de entrada no início do algoritmo. Este processo de partição dos blocos usava um esquema fixo onde cada bloco quadrado (pertencente a um dicionário S_i, com nível i par) era sempre dividido na direção vertical (para escolha de segmentação inicial vertical), enquanto os blocos de nível i ímpar eram segmentados horizontalmente. Esta característica do processo de partição dos blocos fazia com que o MMP usasse somente blocos quadrados e retangulares (de proporções 1:2 e 2:1). Quando a divisão inicial da imagem fosse feita em blocos 16×16 (como utlizado na prática normalmente), o dicionário formado possuía 9 (nove) escalas diferentes (níveis de 0 a 8). O uso de tal esquema fixo de partição limitava a capacidade do MMP de adaptar-se à estrutura da imagem.

Em [14], foi proposto um novo esquema de segmentação, onde qualquer bloco pode ser segmentado tanto na direção horizontal como na direção vertical durante o processo de codificação, sendo que a escolha é decidida com base em um critério taxa-distorção.

No novo esquema, cada bloco de entrada $\mathbf{x}^i$ da imagem original é segmentado em ambas as direções, e aquela com menor custo J é escolhida para a representação do bloco na árvore de segmentação. O processo de segmentação de cada bloco obtido, assim como ocorria no esquema anterior, descrito na seção 3.1.2, é feito recursivamente até que a árvore de segmentacao alcance os blocos de dimensão 1×1. A figura 3.14 mostra um exemplo da partição de um bloco de entrada 16×16 utilizando segmentação flexível.

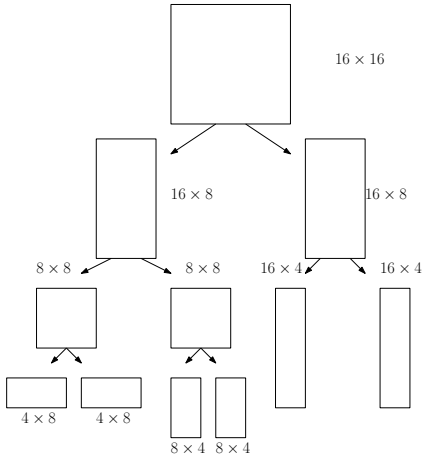


Figura 3.14: Exemplo de segmentação flexível para um bloco 16x16.

Este novo esquema de segmentação gera uma consequência imediata que é a existência de uma quantidade maior de dimensões distintas de padrões disponíveis para a codificação. Para um bloco de entrada 16×16, tem-se agora todos os blocos $2^m \times 2^n$ $(m, n = 1, 2, 3$ e $t4)$. Sendo assim, para representar todas as dimensões possíveis da arvore binária, o dicionário $\mathcal{D}$ possuirá agora 25 níveis ao invés de 9, representando as dimensões: 1×1, 1×2, 2×1, 2×2, 1×4, 4×1, 2×4, ..., 8×16, 16×8 e 16×16.

No esquema anterior de segmentação, era necessário enviar um *flag* apenas informando se um bloco foi ou não particionado. A informação sobre a direção não era necessária, uma vez que definido o sentido inicial, o algoritmo alternava as direções em cada segmentação sucessiva. No esquema de segmentação flexível, no entanto, é necessário informar ao decodificador não apenas se um bloco foi particionado, mas ainda em que direção a segmentação ocorreu.

Capítulo 4

Fundamentos de imagens estereoscópicas

4.1 Introdução

Em imagens digitais 2-D, a informação de profundidade dos objetos está ausente. Pode existir apenas a idéia intuitiva de profundidade, tendo como referência o tamanho de um objeto em relação a outro na imagem, além de propriedades que utilizam informações da imagem como contraste, sonbreamento, nitidez e geometria dos objetos da cena.

O uso de tecnologias 3-D tem aumentado bastante, e o conteúdo assim exibido tem-se popularizado bastante, principalmente no cinema e em jogos de computador, entre outros. No entanto, os conteúdos exibidos em 3-D presentes nestas aplicações são, muitas vezes, computação gráfica ou conteúdos 3-D a partir de uma escolha muito limitada, já que os custos de criação de conteúdos em 3-D ainda são muito mais altos do que os custos para conteúdos em 2-D tradicionais. Estes custos podem incluir, por exemplo, a utilização de um display especial que evite o uso de óculos.

Uma das soluções para esta situação é a utilização dos conteúdos em 2-D existentes, ou seja, uma conversão 2-D para 3-D, que pode abranger diversas técnicas que utilizam informações de geometria de duas ou mais imagens 2-D que apresentem o conteúdo da mesma cena. Estas informações incluem relações matemáticas de vetores de posição e o conteúdo de profundidade da cena em relação a cada câmera. Esta abordagem apresenta algumas vantagens como baixa complexidade numérica, e não há necessidade de ajuste de parâmetros para os diferentes tipos de imagens.

Em aplicações de vídeos 3-D, as cenas capturadas por cada uma das câmeras são polarizadas e sobrepostas na mesma tela. Em seguida, cada um dos olhos vê a cena através de um filtro polarizado, e assim cada olho é capaz de visualizar a cena de perspectivas diferentes entre si, causando a sensação de três dimensões.

4.1.1 Fundamentos da visualização estéreo

Em Visão Computacional, busca-se estimar ou tornar explícitas as propriedades dinâmicas e geométricas do mundo 3-D em imagens digitais 2-D a partir de um cenário de técnicas matemáticas. As ferramentas necessárias incluem *hardware* para a aquisição e armazenamento de imagens digitais, e algoritos de processamento de imagens que explorem as propriedades geométricas dos vetores de posicionamento dos objetos da cena em relação à câmera.

Uma imagem 3-D é uma imagem onde há a sensação de profundidade dos objetos presentes na cena. Tal como ocorre no sistema visual humano, onde vemos o mundo em três dimensões a partir da combinação das imagens formadas pelos dois olhos, assim também o sistema digital 3-D captura a mesma cena por pelo menos duas câmeras. Ou seja, o uso de duas ou mais câmeras em diferentes origens causa uma disparidade binocular que fornece informações importantes sobre a profundidade (distâncias relativas dos objetos em relação ao observador).

Uma vez que o conteúdo 3-D é obtido pelo processamento e combinação das propriedades do conteúdo 2-D, será feita uma abordagem sobre o modelo de câmera puntiforme (utilizado para aquisição de imagens 2-D), e a seguir será abordado o modelo geométrico do sistema estéreo, destacando as relações existentes entre as variáveis que compõe dois sistemas puntiformes, utilizados para aquisição da mesma cena, para a formação do sistema epipolar.

A abordagem utilizada nas descrições dos modelos a seguir tem como base as referências [15] e [16], de maneira que uma descrição detalhada e demonstrações dos modelos descritos a seguir podem ser encontradas nestas obras.

4.1.2 Modelo de câmera puntiforme

O modelo de câmera puntiforme é o esquema usado no processo de aquisição de uma imagem. Basicamente, este sistema relaciona matematicamente os parâmetros externos (também chamados *extrínsecos*) e internos (*intrínsecos*) da câmera. Os parâmetros extrínsecos descrevem a localização e orientação da câmera em relação a um ponto no espaço, enquanto os parâmetros intrínsecos relacionam as coordenadas dos pixels de uma imagem com as coordenadas da câmera.

Seja o modelo de câmera puntiforme representado na figura, 4.1, tem-se um sistema de coordenadas X, Y, Z, com origem em O, chamado *centro de projeção*.

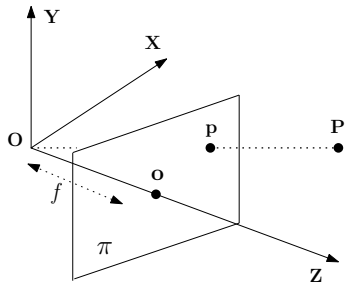


Figura 4.1: Modelo de camera puntiforme.

A imagem será formada no plano π, o qual é ortogonal ao eixo Z. A distância f entre o centro de projeção O e o centro da imagem $\mathbf{o}$, onde o eixo Z intercepta o plano π, é chamada *distância focal*.

Seja $\mathbf{P}$ um ponto no espaço cujas coordenadas são $[X, Y, Z]^T$, então define-se a representação de $\mathbf{P}$ na imagem como o ponto $\mathbf{p}$, de coordenadas $[x, y, z]^T$, onde a reta $O - \mathbf{P}$ intercepta o plano π, assim como mostrado na figura 4.1.

A posição (x, y) do ponto $\mathbf{p}$ na imagem 2-D, formada no plano π, é definida pelas seguintes equações:

$$x = f\frac{X}{Z}$$

$$y = f\frac{Y}{Z} \tag{4.1}$$

Neste sistema, o valor de z para o ponto $\mathbf{p}$ será sempre igual a f.

4.1.3 Geometria da representação do sistema 3-D

A representação do sistema 3-D é formada pelo modelo geométrico de duas câmeras puntiformes. Em cada uma delas, a posição (x, y) do ponto $\mathbf{p}$ no plano da imagem, que representa o ponto $\mathbf{P}$ no espaço, é determinada da mesma maneira que foi descrita na seção 4.1.2. Uma questão importante do sistema estereo é o problema da correspondência. Uma vez determinada a posição (x_e, y_e) do ponto $\mathbf{p_e}$ na imagem esquerda, correspondente ao ponto $\mathbf{P}$, o próximo passo é determinar o ponto homólogo $\mathbf{p_d}$ na imagem direita que corresponde ao mesmo ponto $\mathbf{P}$ no espaço.

A figura 4.2 mostra a visão superior de um sistema estéreo simples, composto por duas câmeras puntiformes, esquerda e direita, com centros de preojeção O_e e O_d respectivamente.

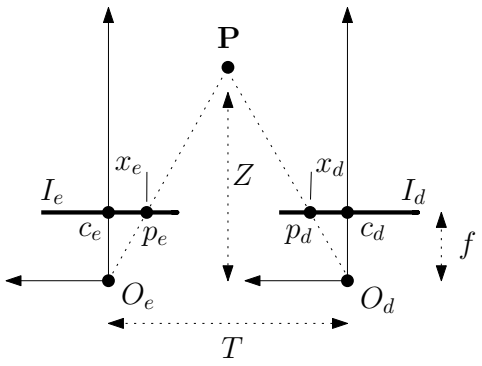


Figura 4.2: Sistema estéreo simples (rep. de [15], p. 143)..

Na representação, as imagens são formadas nos planos I_e e I_d, que neste exemplo são coplanares, e os eixos ópticos (linhas que interceptam cada um dos centros de projeção e o centro da imagem correspondente) são paralelos. As duas câmeras possuem a mesma distância focal f. A distância T entre os centro de projeção O_e e O_d é chamada de *linha base* do sistema estéreo.

Seja o ponto $\mathbf{P}$ no espaço, a sua representação na imagem esquerda é o ponto $\mathbf{p_e}$ onde a linha $O_e - \mathbf{P}$ intercepta o plano I_e. Supondo, neste exemplo, que o problema da correspondência já foi resolvido, então $\mathbf{p_d}$ é a representação de $\mathbf{P}$ no plano da imagem direita, I_d. A distância entre $\mathbf{p_e}$ e o centro da imagem esquerda, c_e é a coordenada de distância x_e. Da mesma forma, x_d é a distância entre $\mathbf{p_d}$ e o centro da imagem direita.

Observa-se na figura 4.2 uma semelhança de triângulos entre $\mathbf{p_e}$-$\mathbf{P}$-$\mathbf{p_d}$ e O_e-$\mathbf{P}$-O_d. Obtém-se daí a seguinte relação:

$$\frac{T}{Z} = \frac{T - (-x_e + x_d)}{Z - f}$$

$$\frac{T}{Z} = \frac{T + x_e - x_d}{Z - f} \tag{4.2}$$

À relação $x_e - x_d$, dá-se o nome especial de *disparidade*, e é definida como a diferença de posição entre pontos correspondentes nas duas imagens [15].

Assim, a profundidade Z é definida como:

$$Z = f\frac{T}{d} \tag{4.3}$$

onde $d = x_e - x_d$ é a disparidade. Da equação 4.3, observa-se que a profundidade é inversamente proporcional à disparidade.

4.1.4 A geometria epipolar

A *geometria epipolar* é uma representação do sistema estéreo utilizada para facilitar a correspondência entre os parâmetros de duas câmeras. Esta representação é mostrada na figura 4.3.

O sistema é formado por duas câmeras puntiformes, esquerda e direita, cada uma com seus respectivos centros de projeção, O_e e O_d, e distâncias focais f_e e f_d.

O nome *geometria epipolar* é usado porque os pontos $\mathbf{e}_e$ e $\mathbf{e}_d$ onde a linha que liga os centros de projeção (linha base) intercepta os planos π_e e π_d são chamados *epipólos*. O epipólo esquerdo $\mathbf{e}_e$ é a imagem do centro de projeção da câmera direita e vice-versa [15].

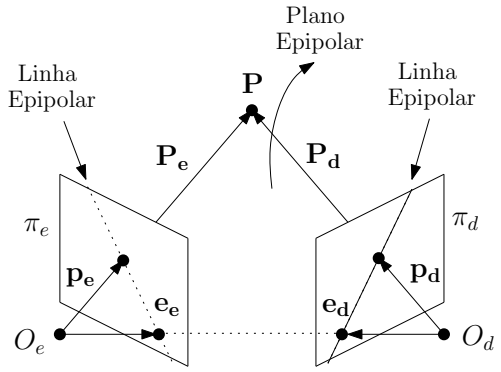


Figura 4.3: Geometria epipolar (rep. de [15], p. 151)..

As imagens serão formadas nos planos π_e e π_d das câmeras esquerda e direita. Os planos são ortonormais aos seus eixos ópticos, e estes coincidem com os eixos de profundidade Z, assim como descrito na seção 4.1.2.

Na figura 4.3, considerando que cada câmera identifica o mesmo ponto P no espaço, os vetores $\mathbf{P}_e = [X_e, Y_e, Z_e]^T$ e $\mathbf{P}_d = [X_d, Y_d, Z_d]^T$ referem-se a este mesmo ponto P, e os vetores $\mathbf{p}_e = [x_e, y_e, z_e]^T$ e $\mathbf{p}_d = [x_d, y_d, z_d]^T$ são as projeções do ponto P nos planos π_e e π_d respectivamente. A geometria do sistema garante que a correspondência entre $\mathbf{p}_e$ e $\mathbf{p}_d$ esteja sobre as linhas epipolares representadas na figura, e a busca por pontos homólogos é restrita a essas regiões das duas imagens. Ou seja, o ponto na imagem direita que corresponde ao ponto $\mathbf{p}_e$ na imagem esquerda estará sobre a linha epipolar representada na imagem direita.

Assim como no modelo de câmera puntiforme, todos os pontos projetados nos planos π_e e π_d terão suas componentes de profundidade iguais a $z_e = f_e$ para a câmera esquerda e $z_d = f_d$ para a câmera direita.

Os sistemas de coordenadas associados a cada câmera estão relacionados entre si através de uma transformação, definida por um vetor de translação $\mathbf{T} = (O_d - O_e)$, de dimensão 3×1, que descreve as posições relativas das origens das duas câmeras, e por uma matriz de rotação R. A matriz R é ortogonal ($RR^T = R^T R = I$), de

dimensão 3×3, e descreve a rotação de uma câmera na posição de centro óptico O_e para a posição de centro óptico O_d.

A matriz de rotação R e o vetor de translação $\mathbf{T}$ são chamados *parâmetros extrínsecos* do sistema estéreo, pois descrevem a localização e orientação da câmera em relação a um ponto no espaço.

No sistema eipolar, os parâmetros extrínsecos são relacionados com a restrição epipolar através de uma matriz E, de dimensão 3×3 e posto 2, chamada *matriz essencial*. A matriz E pode ser interpretada como um mapeamento entre pontos em um plano e linhas epipolares no outro plano.

Estas relações utilizam as coordenadas no domínio da câmera (ou seja, os parâmetros extrínsecos). Os parâmetros intrínsecos das câmaras esquerda e direita são representados pelas *matrizes de projeção* M_e e M_d. A transformação para as coordenadas de pixels (parâmetros intrínsecos) são feitas a partir da matriz F, chamada *matriz fundamental*, de acordo com a equação:

$$\bar{\mathbf{p}}_d^T F \bar{\mathbf{p}}_e = 0 \tag{4.4}$$

onde $\bar{\mathbf{p}}_d$ e $\bar{\mathbf{p}}_e$ são as coordenadas no domínio dos pixels, e a matriz F é encontrada a partir de $F = M_d^{-T} E M_e^{-1}$.

A matriz essencial E estabelece relações apenas com os parâmetros extrínsecos, e relaciona os pontos correspondentes em ambos os planos a partir de rotações e translações em seus sistemas de coordenadas, enquanto a matriz fundamental F é a representação algébrica da geometria epipolar, estabelecendo relações com os parâmetros extrínsecos e intrínsecos das câmeras, relacionando as coordenadas dos pixels nas duas vistas. A obtenção de todas as matrizes e demonstrações podem ser vistas com detalhes em [15] e [16].

Estas relações são bastante úteis para fazer a síntese de vistas a partir de métodos baseados em DIBR [17]. Para concluir esta abordagem sobre conceitos de estereoscopia, exemplificamos abaixo as matrizes de projeção M_i, os centros de projeção O_i, a linha base T e a matriz de rotação R, correspondentes às câmeras 37 (considerada neste caso como a câmera esquerda) e 39 (câmera direita) da sequëncia *Champagne Tower*, figuras B.7 e B.8 do Apêndice B. (Fonte: Nagoya University [29]).

$$M_e^{37} = \begin{bmatrix} 2969 & 0 & -163,48 \\ 0 & 2969 & 457,7 \\ 0 & 0 & 1 \end{bmatrix} \qquad O_e^{37} = \begin{bmatrix} -125 \\ 0 \\ 0 \end{bmatrix} \qquad R^{37} = R^{39} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$M_d^{39} = \begin{bmatrix} 2969 & 0 & -127,34 \\ 0 & 2969 & 457,7 \\ 0 & 0 & 1 \end{bmatrix} \qquad O_d^{39} = \begin{bmatrix} -25 \\ 0 \\ 0 \end{bmatrix} \qquad T = O_d^{39} - O_e^{37} = \begin{bmatrix} 100 \\ 0 \\ 0 \end{bmatrix}$$

4.2 Mapa de profundidade

Um Mapa de Profundidade é uma imagem formada por uma matriz cujos valores são as medidas normalizadas da distância da origem de um sistema de coordenadas de referência até os pontos na superfície dos objetos da cena observada. Esta informação é utilizada para recuperar a estrutura 3-D, sendo ainda muito útil para realizar o casamento entre pontos correspondentes das duas vistas.

De acordo com a equação 4.3, a profundidade de um ponto da cena é inversamente proporcional a disparidade, que é a grandeza que define a diferença de posição entre pontos correspondentes de duas imagens. Esta característica pode ser verificada observando-se objetos em movimento: objetos distantes parecem mover-se mais lentamente.

Uma estimação precisa do mapa de disparidades, para a reconstrução posterior da cena 3-D, é crucial para a análise da imagem estereoscópica em visão estéreo porque o valor de disparidade corresponde à distância entre as câmeras e o objeto da cena 3-D. Assim, muitos estudos têm sido feitos para se obter uma estimativa precisa da disparidade. Porém não vale a pena calcular um mapa de disparidades preciso se o custo para transmitir ou armazenar este mapa for muito alto uma vez que, do ponto de vista de codificação, mais importante do que a reconstruçãp perfeita da cena 3-D é o compromisso entre taxa e distorção [18].

Dentre os métodos para obtenção do mapa de disparidades podemos citar o algoritmo *area- based* [19], que utiliza uma janela adaptativa, obtendo um mapa preciso. As técnicas de correspondência com blocos de tamanho fixo (*fixed size block matching*, FSBM), exploram a redundância no mapa de disparidades com uma estrutura regular, sendo assim mais efetivos em termos de compromisso taxa-distorção ao custo de um mapa de disparidade pouco consistente, comprometendo a qualidade da imagem reconstruída. Nos métodos baseados em FSBM, pode-se utilizar blocos maiores para regiões homogêneas e blocos menores para regiões co muitos detalhes, chamados métodos *quadtrees* [20] e [21].

Em [22] é proposto um método de obter o codificar o mapa de disparidade que é baseado em objetos , apresentando vantagens em relação aos métodos baseados em blocos, resultando, porém, em mapas mais esparsos, e exigindo uma etapa de segmentação da cena em objetos.

A figura 4.4(a) mostra uma imagem estereoscópica de textura, e a figura 4.4(b) o seu respectivo mapa de profundidade.

(a) (b)

Figura 4.4: Imagem Champagne Tower, câmera 39, *frame* 0: a) Imagem de textura; b) Mapa de profundidades. Fonte: *Nagoya University* [29].

Na figura 4.4(b), os tons de cinza mais claros representam pontos de objetos mais próximos da câmera, enquanto os mais escuros representam objetos mais distantes. Assim, os mapas devem apresentar valores de profundidade muito próximos dentro do mesmo objeto, enquanto que as bordas caracterizam as transições de profundidade.

Os mapas de profundidade são fornecidos pela câmera juntamente com as respectivas imagens de textura, sendo que, neste projeto, as informações de profundidade das imagens de textura utilizadas já eram conhecidas a priori.

4.3 Reconstrução com base na imagem de profundidade

Nas seções 4.1.3 e 4.1.4, foram apresentados os fundamentos da representação geométrica de um sistema estéreo. As equações mostradas nestas seções basicamente modelam a projeção de um ponto 3-D no espaço em um plano 2-D da imagem. Em aplicações de FTV e 3DTV, vistas virtuais de uma cena podem ser sintetizadas a partir das vistas de referência, permitindo que o usuário possa escolher o ângulo de visão. Para isso, o modelo matemático mostrado anteriormente deve fazer o caminho inverso: estimar a posição 3-D de um ponto (profundidade) utilizando as imagens 2-D [17].

O estado da arte para a síntese de vistas virtuais usam geralmente métodos relacionados a *Depth Image-Based Rendering* (DIBR). A síntese da vista virtual é realizada a partir da informação presente nos mapas de profundidade das imagens de referência. Por isso, a precisão das informações presentes nos mapas de profundidades provocam um impacto direto na qualidade da imagem reconstruída. Quando convertidas em tons de cinza, estas informações podem perder a precisão se quantizadas de maneira inadequada.

A estimativa da profundidade visa ao cálculo da estrutura e profundidade dos objetos presentes na cena a partir de um cenário de múltiplas imagens. A aplicação destes métodos implicam, assim, no conhecimento das matrizes R, T, E e F, descritas na seção 4.1.4.

Uma descrição detalhada da estimativa de profundidade a partir de múltiplas vistas, bem como os fundamentos matemáticos de métodos DIBR podem ser conferidos na referência [23].

Neste trabalho, foi usado o *software* VSRS (*View Synthesis Reference Software*) [24], no capítulo 6, para avaliação da compressão de imagens estéreo (utilizando o MMP) sobre a síntese de vistas virtuais.

Capítulo 5

Codificação conjunta das vistas de textura e profundidade

5.1 Motivação

Em [25], o MMP foi aplicado para codificar mapas de profundidade, e os resultados obtidos superaram o JPEG2000 [26] e o H.264/AVC [10]. Estes resultados podem ser justificados pelo fato do MMP não fazer uso de transformadas para o domínio da frequência, mas apenas casamentos aproximados, o que preserva as informações de profundidade nos mapas. No entanto, o algoritmo usado para codificação de imagens de profundidade é o mesmo algoritmo MMP usado na codificação de qualquer outra imagem. Por isso, resolveu-se desenvolver, a partir da então versão atual do algoritmo MMP, um algoritmo que fizesse uma codificação "conjunta" das imagens de textura e profundidade, de maneira que a similaridade que há entre as duas imagens pudesse ser explorada.

Uma primeira tentativa na busca de tentar explorar estas similaridades entre as imagens de textura e profundidade é usar o mesmo dicionário obtido na codificação de uma das vistas do par para a codificação da outra vista. Uma vez que o MMP é adaptativo e "aprende" os padrões codificados, é razoável deduzir que os padrões aprendidos com uma das imagens possa ser útil para a codificação da próxima, haja vista a semelhança visual existente entre elas.

5.2 Imagem de textura com dicionário do mapa de profundidades

O algoritmo MMP foi adaptado para realizar uma codificação conjunta, seguindo o esquema da figura 5.1:

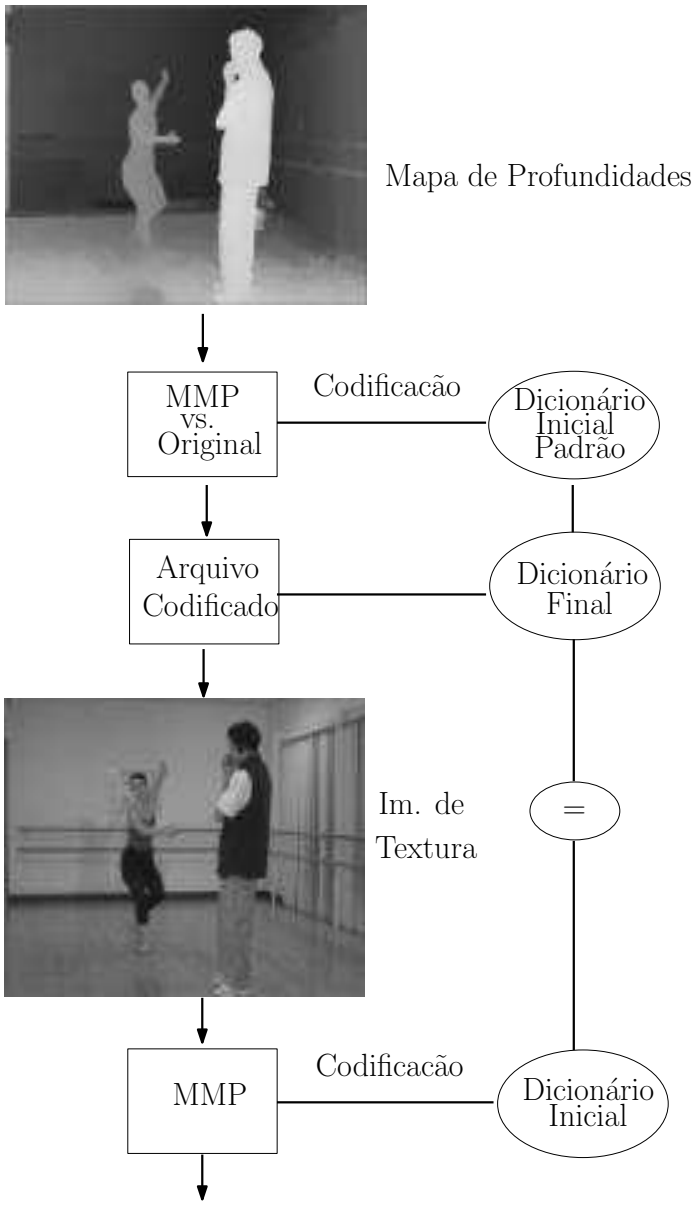


Figura 5.1: Esquema da codificação conjunta.

Na figura 5.1, uma imagem de profundidade é codificada com a versão original do algoritmo MMP. Ao final desta codificação, teremos um dicionário final com novos padrões adicionados de acordo com as características da imagem de profundidade. Este novo dicionário é usado como ponto de partida para a codificação da imagem de textura correspondente ao mapa de profundidade codificado primeiramente, sendo então o dicionário inicial da codificação da imagem de textura. Em seguida, esta é

codificada normalmente, de maneira que o dicionário continua sendo atualizado.

Para a realização de experimentos com o algoritmo esquematizado na figura 5.1, resolveu-se usar a imagem de profundidade como a primeira imagem a ser codificada, uma vez que essas imagens possuem características de apresentar mais regiões uniformes, e por isso o dicionário final será menor, considerando que haverá menos segmentação de blocos durante a codificação.

Os resultados das simulações para a codificação da ordem inversa das imagens (primeiro textura, e depois profundidade) podem ser observados na seção 5.3

Neste experimento, o algoritmo MMP descrito no capítulo 3 será chamado de *versão original*, e a adaptação feita para a codificação conjunta de pares estéreo será chamada *versão estéreo*.

Nas simulações do algoritmo mostrado na seção 5.2, foram usados 5 (cinco) pares de sequências estéreo (mapa de profundidades e imagem de textura): Ballet (câmera 03, *frame* 0) e Breakdancers (câmera 03, *frame* 0), estas duas obtidas de [27], Book Arrival (câmera 08, *frame* 0) [28], Champagne Tower (câmera 39, *frame* 0) e Pantomime (câmera 37, *frame* 0) [29]. As imagens originais utilizadas podem ser vistas no apêndice B.

5.2.1 Resultados

A seguir são mostrados os gráficos taxa × PSNR apresentando a comparação entre os resultados do algoritmo original e a versão esquematizada na figura 5.1 para os 5 (cinco) pares de imagens testadas.

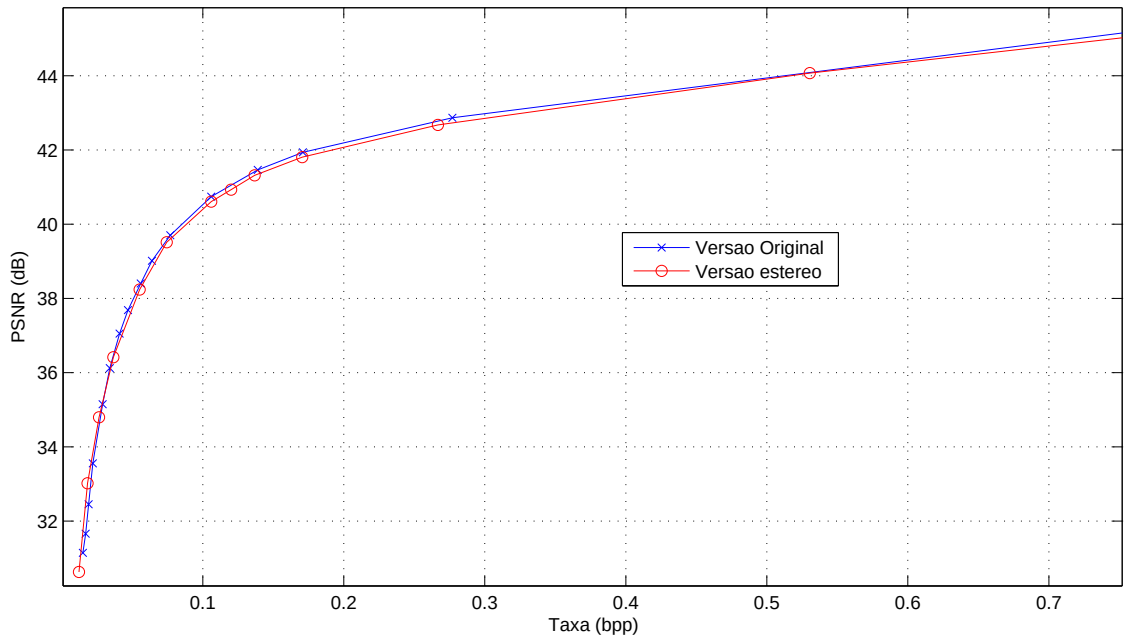


Figura 5.2: Ballet (img. de textura), cam. 03, *frame* 0, a partir da img. de profundidade, cam. 03, *frame* 0.

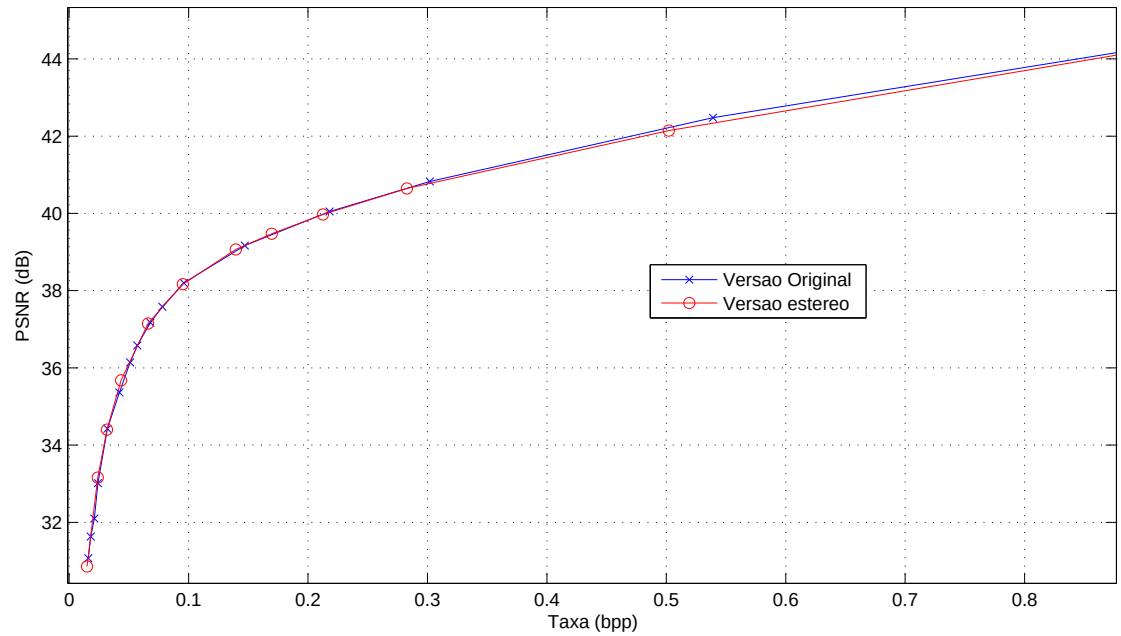


Figura 5.3: Breakdancers (img. de textura), cam 03, *frame* 0, a partir da img. de profundidade, cam. 03, *frame* 0.

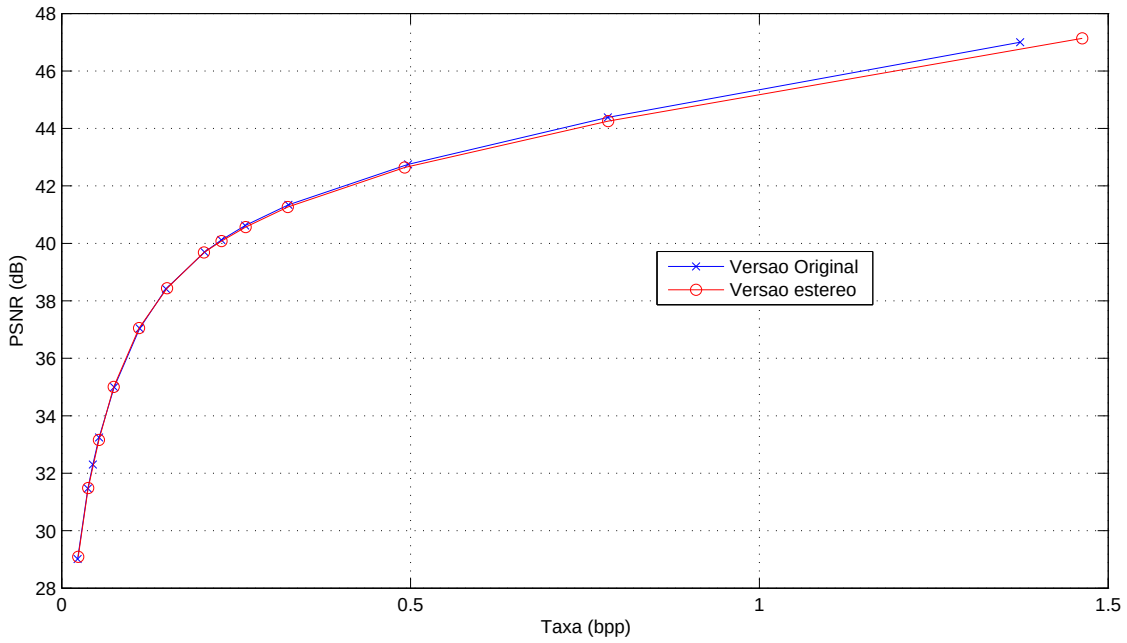


Figura 5.4: Book Arrival (img. de textura), cam 08, *frame* 0, a partir da img. de profundidade, cam. 08, *frame* 0.

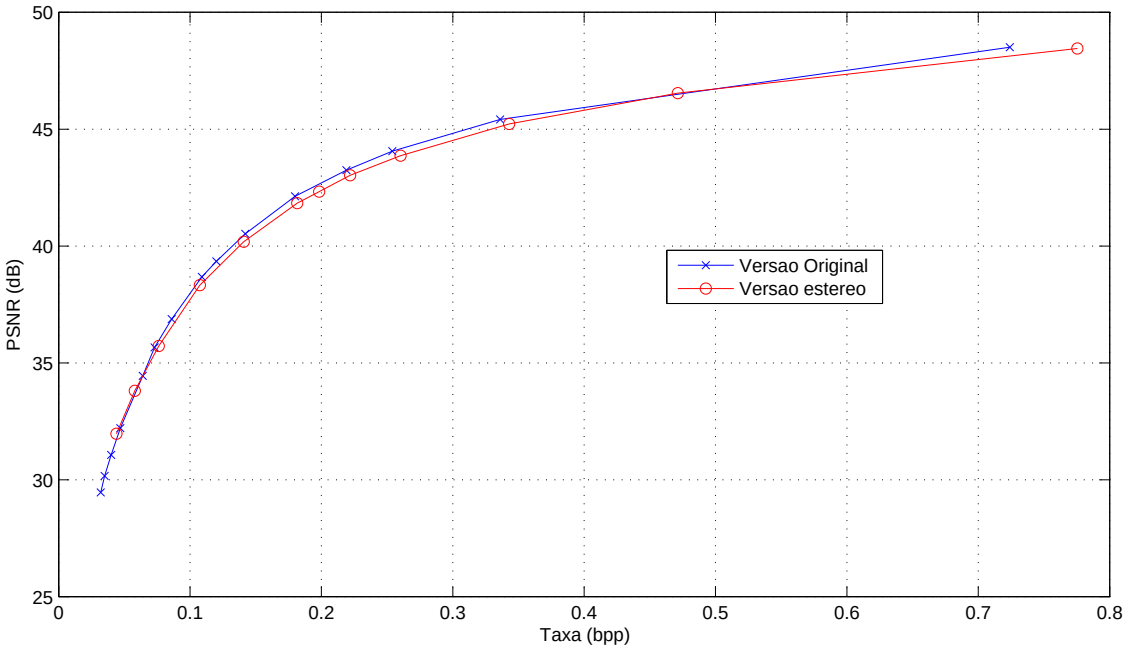


Figura 5.5: Champagne Tower (img. de textura), cam 39, *frame* 0, a partir da img. de profundidade, cam. 39, *frame* 0.

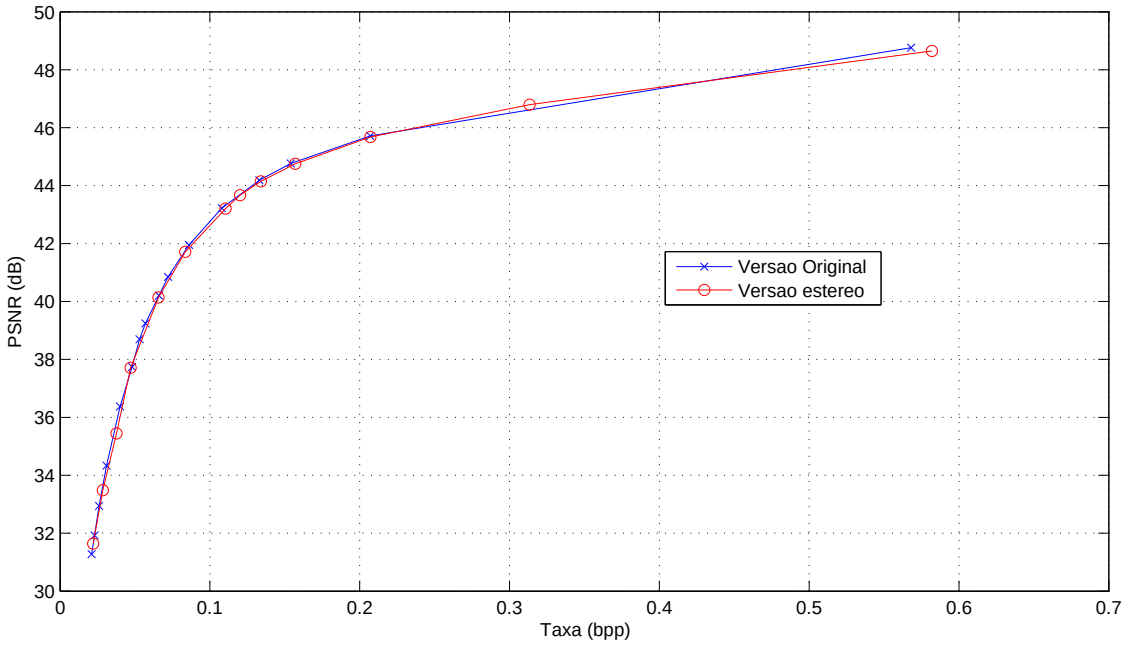


Figura 5.6: Pantomime - frame 0 - (img. de textura), cam 37, *frame* 0, a partir da img. de profundidade, cam. 37, *frame* 0.

A tabela 5.1 mostra a quantidade de vezes que cada dicionário utilizou cada modo de predição ao final da codificação do mapa de profundidade para a imagem *book arrival* (câmera 08, *frame* 0). Na tabela, podemos observar que, na maioria dos dicionários, hove uma tendência de se utilizar bastante os modos de predição vertical e horizontal (modos 0 e 1) em relação aos modos de predição diagonais.

As figuras 5.7 e 5.8 mostram respectivamente a segmentação obtida no mapa de profundidade da imagem *book arrival* (câmera 08, *frame* 0), e a seguir a mesma segmentação sobreposta sobre a imagem de textura. Podemos notar que, na imagem de textura, esta segmentação não está bem adaptada, e padrões complexos (como por exemplo, a região da orelha) está representado por um único grande bloco ao invés do bloco ser segmentado para a obtenção de um casamento melhor. A segmentação se concentrou nas bordas do mapa de profundidade. A mesma interpretação pode ser dada ao par de imagens da sequência *pantomime* (câmera 37, *frame* 0), mostrado nas figuras 5.9 e 5.10.

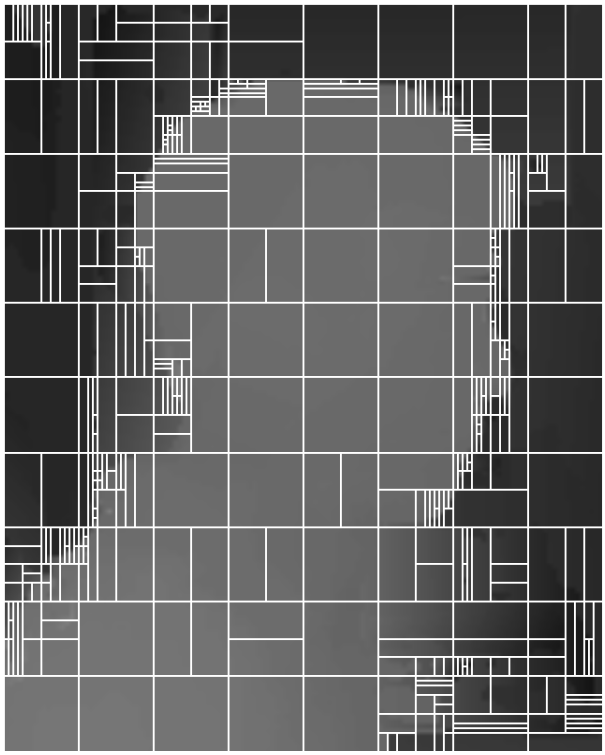


Figura 5.7: Segmentação do mapa de profundidade em uma região da imagem book arrival, câmera 08, *frame* 0, codificado com Lambda=10.

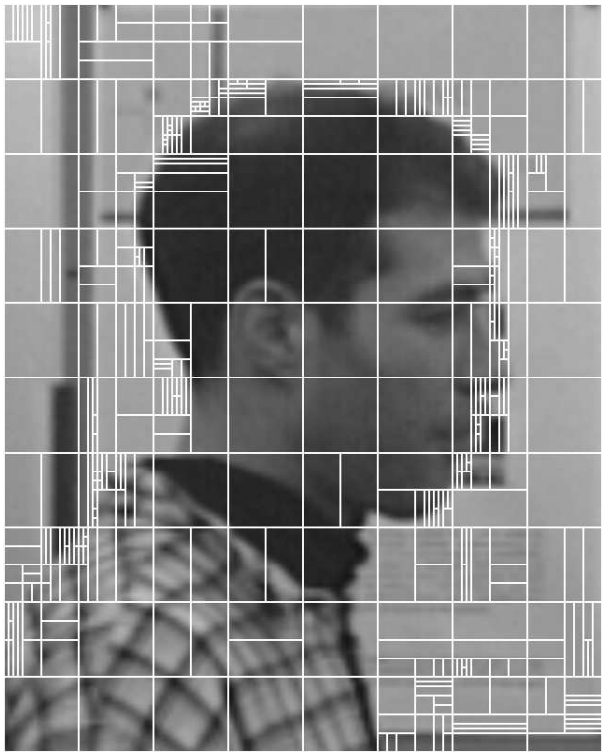


Figura 5.8: Segmentação do mapa de profundidade em uma região da imagem book arrival, câmera 08, *frame* 0, codificado com Lambda=10, sobreposta sobre a imagem de textura.

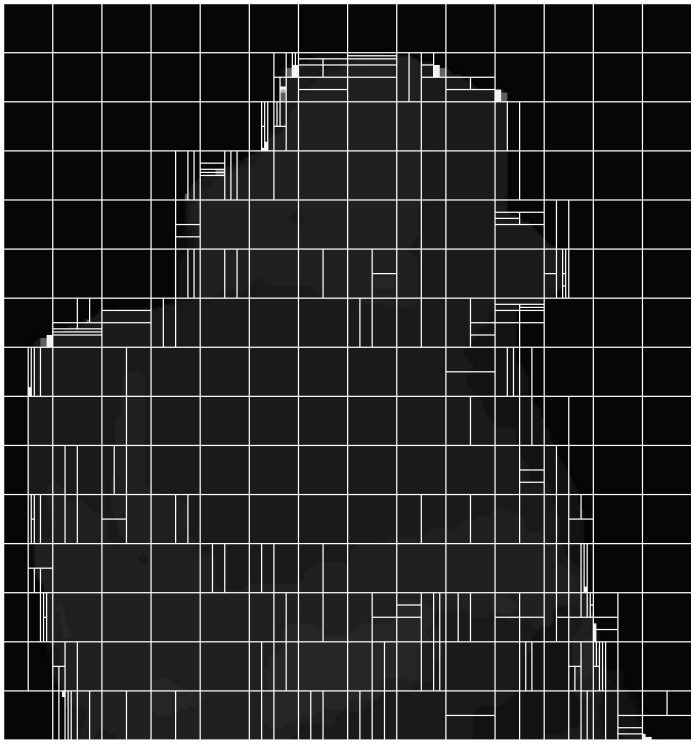


Figura 5.9: Segmentação do mapa de profundidade em uma região da imagem pantomime, câmera 37, *frame* 0, codificado com Lambda=10.

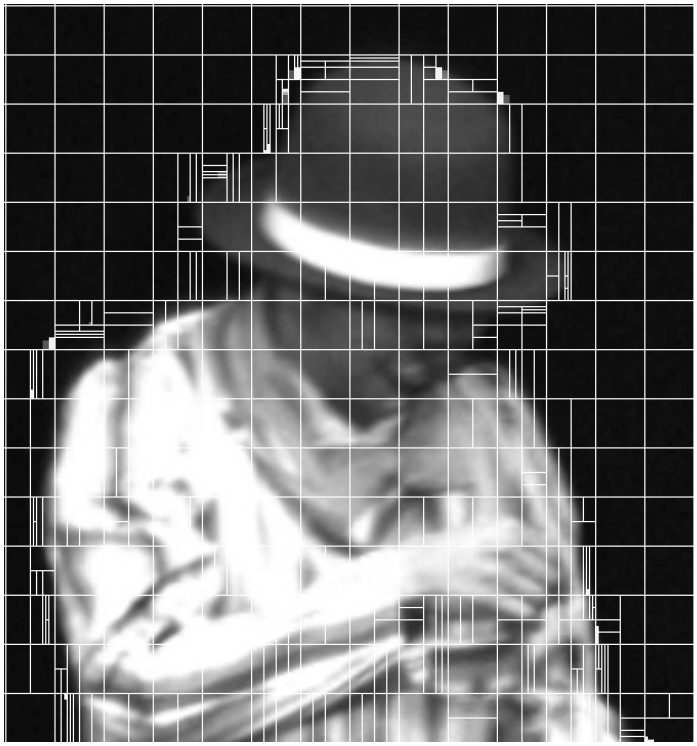


Figura 5.10: Segmentação do mapa de profundidade em uma região da imagem pantomime, câmera 37, *frame* 0, codificado com Lambda=10, sobreposta sobre a imagem de textura.

Tabela 5.1: Modos de predição usados por cada dicionário após a codificação da primeira imagem (profundidade) na sequência book arrival.

	Modos de Predição									
Dic.	0: vert.	1: hor.	2: MFV	3: diag. inf. esq.	4: diag. inf. dir.	5: vert. dir.	6: hor. inf.	7: vert. esq.	8: hor. sup.	9: LSP
0	0	0	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0	0	0
3	0	0	0	0	0	0	0	0	0	0
4	0	0	0	0	0	0	0	0	0	0
5	0	0	0	0	0	0	0	0	0	0
6	0	0	0	0	0	0	0	0	0	0
7	0	0	0	0	0	0	0	0	0	0
8	84	8	120	5	12	14	11	9	37	10
9	15	50	33	1	1	9	2	2	11	6
10	19	11	2	0	0	1	0	1	0	0
11	121	117	59	3	14	10	11	1	3	31
12	35	28	85	1	11	7	5	1	5	3
13	117	227	51	3	30	40	23	9	13	38
14	273	14	135	8	16	29	18	1	47	53
15	173	23	37	4	1	12	42	20	120	58
16	28	60	17	16	4	1	6	3	1	3
17	9	5	7	0	4	0	1	0	0	6
18	31	27	192	22	26	14	23	9	29	26
19	37	29	37	0	8	8	5	0	0	5
20	74	172	57	4	17	27	32	14	14	44
21	74	46	4	3	8	31	6	0	10	68
22	100	234	35	45	83	86	96	36	25	30
23	229	71	18	12	39	26	25	5	11	67
24	133	868	204	653	0	0	0	0	0	0

5.3 Mapa de profundidades com dicionário da imagem de textura

Na seção 5.2, foram apresentados os resultados das simulações para a codificação conjunta da imagem de textura e profundidade, de maneira a codificar-se primeiramente o mapa de profundidade para evitar o um crescimento muito grande do dicionário, o qual será utilizado como dicionário inicial da codificação da segunda imagem.

O teste inverso (codificação da imagem de textura, seguida do mapa de profundidades) utiliza o esquema da figura 5.11.

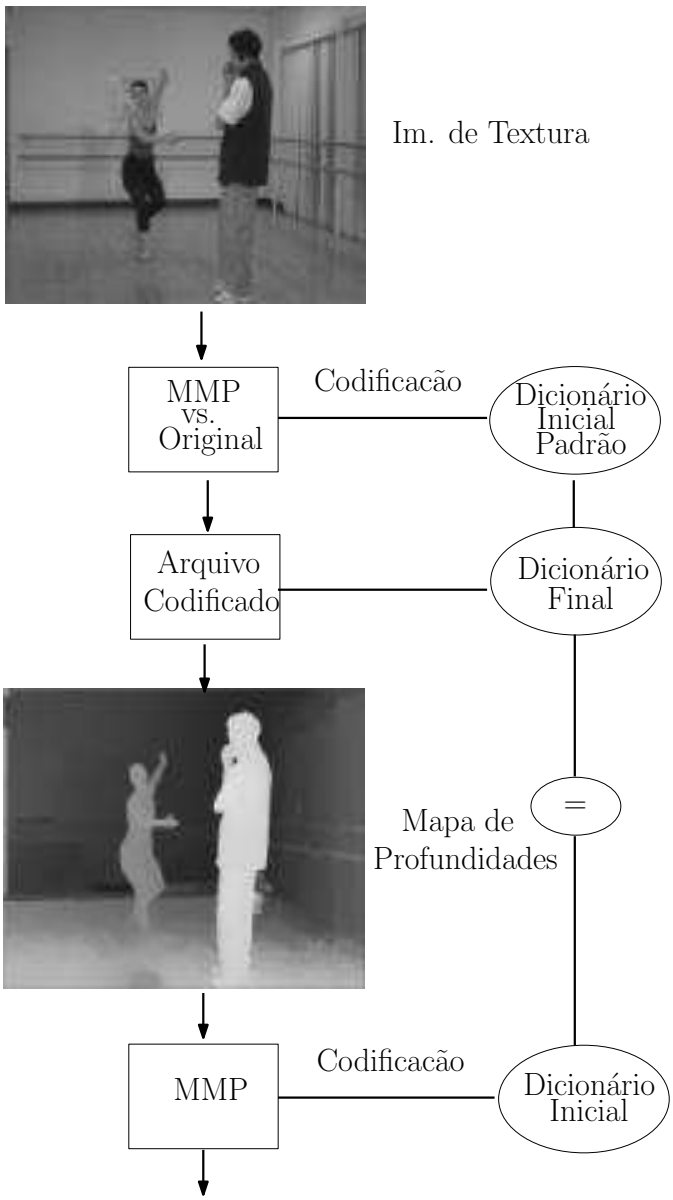


Figura 5.11: Esquema da codificação conjunta (textura seguida de profundidade).

5.3.1 Resultados

Os resultados para o teste inverso daqueles apresentados na seção 5.2 são apresentados a seguir. Os gráficos mostram a relação taxa × PSNR dos mapas de profundidades codificados utilizando como dicionário inicial o dicionário final da codificação da imagem de textura correspondente.

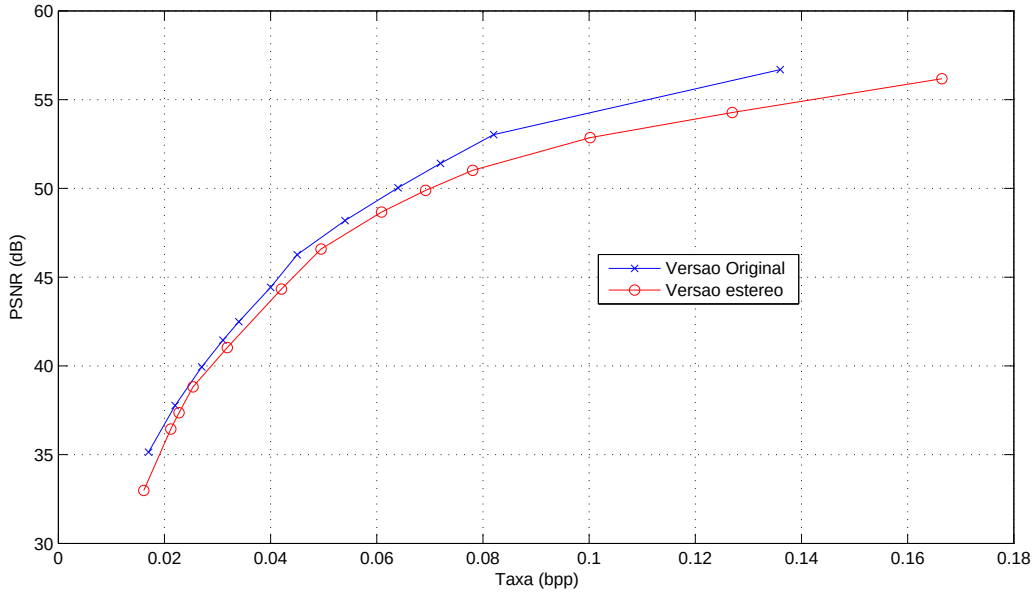


Figura 5.12: Ballet (img. de profundidade), cam. 03, *frame* 0, a partir da img. de textura, cam. 03, *frame* 0.

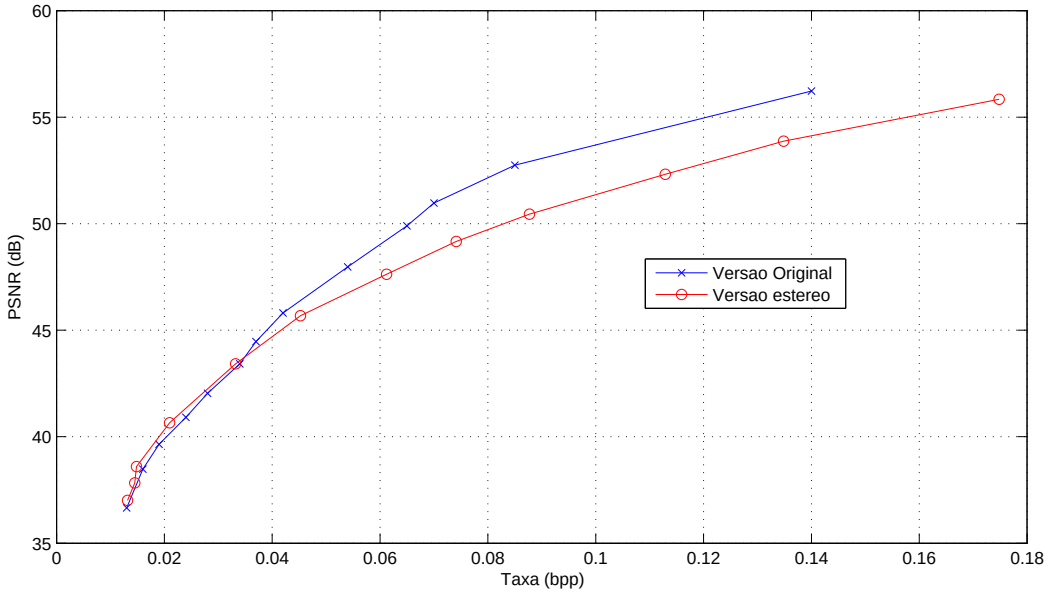


Figura 5.13: Breakdancers (img. de profundidade), cam. 03, *frame* 0, a partir da img. de textura, cam. 03, *frame* 0.

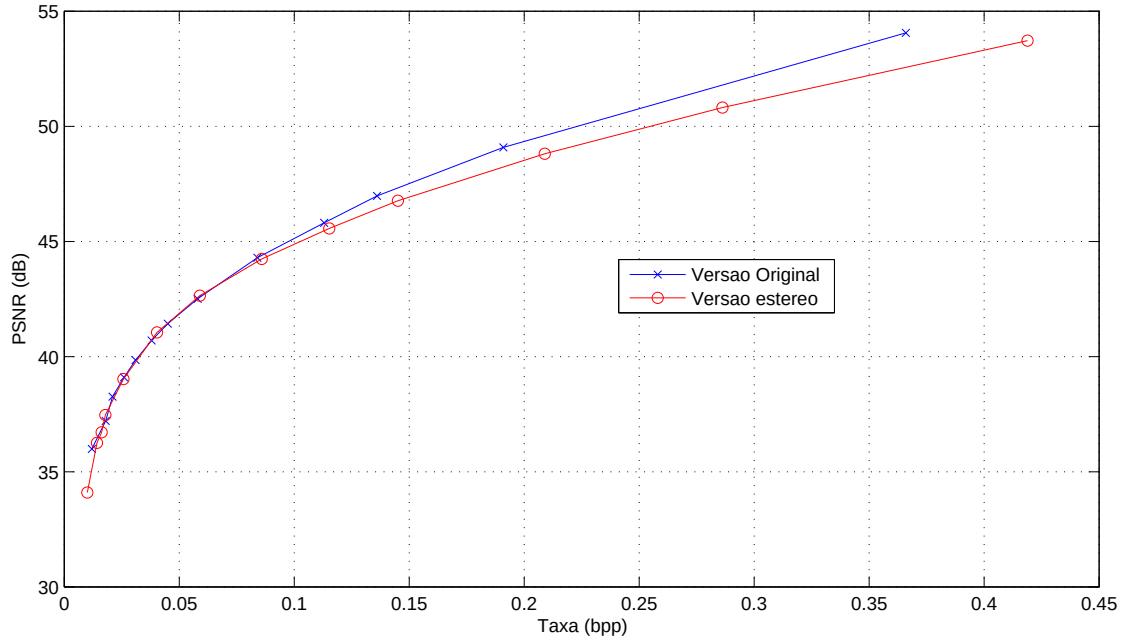


Figura 5.14: Book Arrival (img. de profunidade), cam. 08, *frame* 0, a partir da img. de textura, cam. 08, *frame* 0.

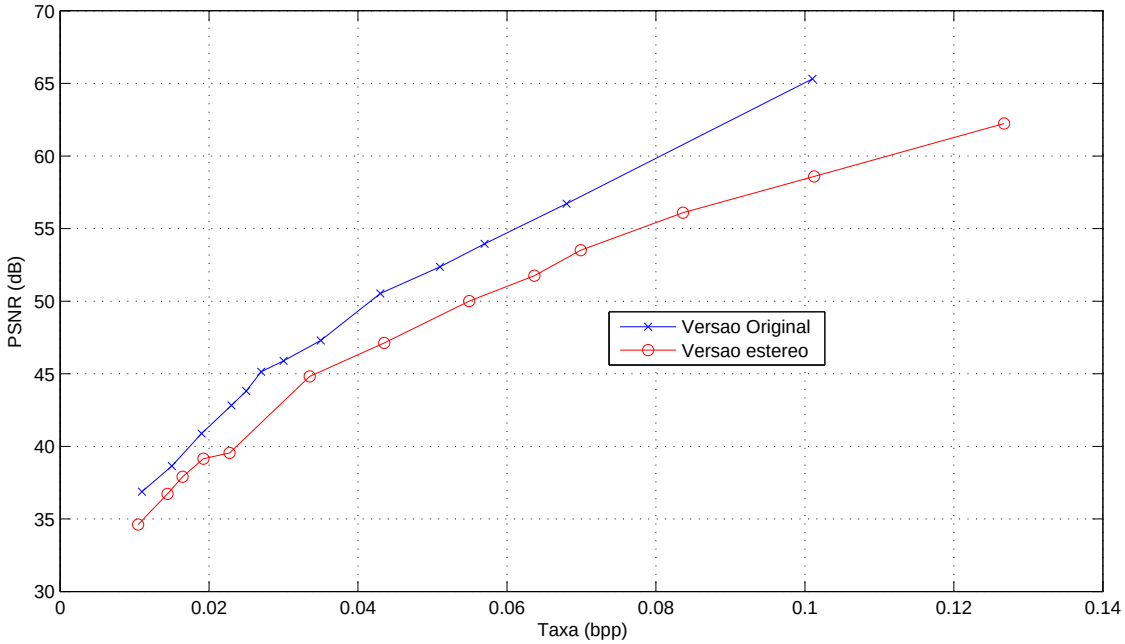


Figura 5.15: Champagne Tower (img. de profundidade), cam. 39, *frame* 0, a partir da img. de textura, cam. 39, *frame* 0.

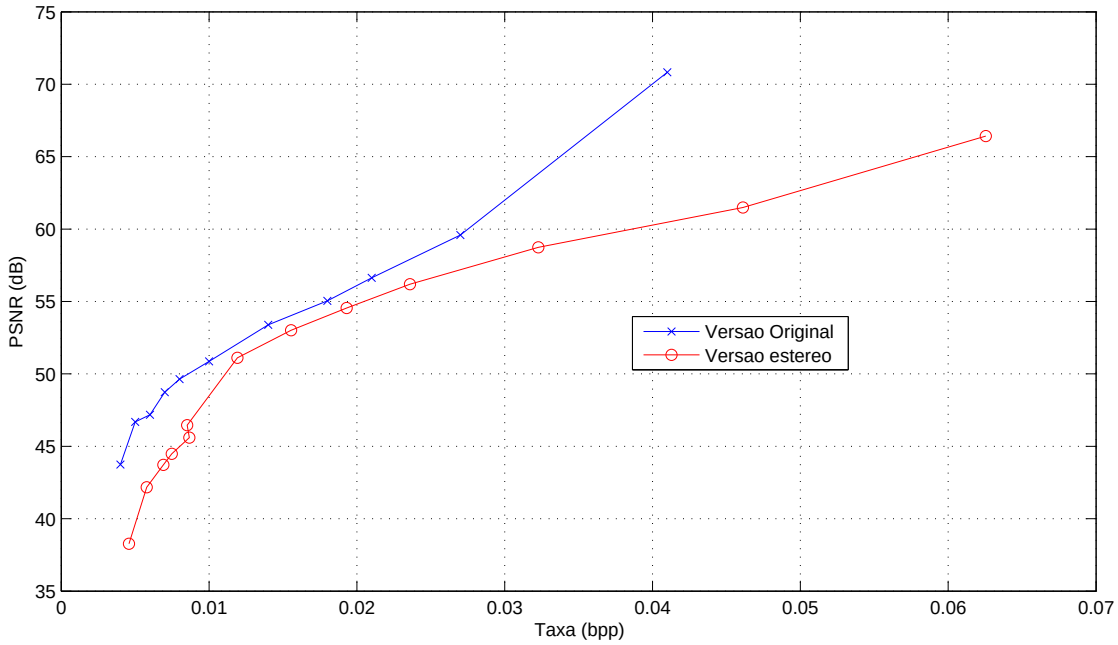


Figura 5.16: Pantomime - frame 0 - (img. de profundidade), cam. 37, *frame* 0, a partir da img. de textura, cam. 37, *frame* 0.

Nas figuras 5.17 e 5.19, podemos observar a segmentação em segmentos das imagens de textura *book arrival* (câmera 08, *frame* 0) e *pantomime* (câmera 37, *frame* 0). Há muito mais segmentações no dicionário final da primeira etapa de codificação do que nos resultados da seção 5.2 devido a quantidade de detalhes das imagens de textura.

A tabela 5.2 mostra a quantidade de vezes que cada dicionário utilizou cada modo de predição ao final da codificação da imagem de textura para a imagem *book arrival* (cãmera 08). Nesta tabela, podemos observar pelo número de modos usados que houve muito mais segmentações de blocos do que nos resultados vistos na tabela 5.1. Dessa forma, o dicionário inicial da segunda imagem já terá muitos padrões (dentre eles, bastante diagonais) e assim terá que se adaptar novamente para a codificação do mapa de profundidade.

Nas figuras 5.18 e 5.20, podemos ver a segmentação obtida ao final da codificação das imagens de textura *book arrival* e *pantomime* sobrepostas sobre os seus mapas de profundidade correspondentes. Percebemos que a segmentação ficou mal adaptada, uma vez que regiões uniformes do mapa de profundidade estão representados com blocos pequenos, resultantes de muitas segmentações.

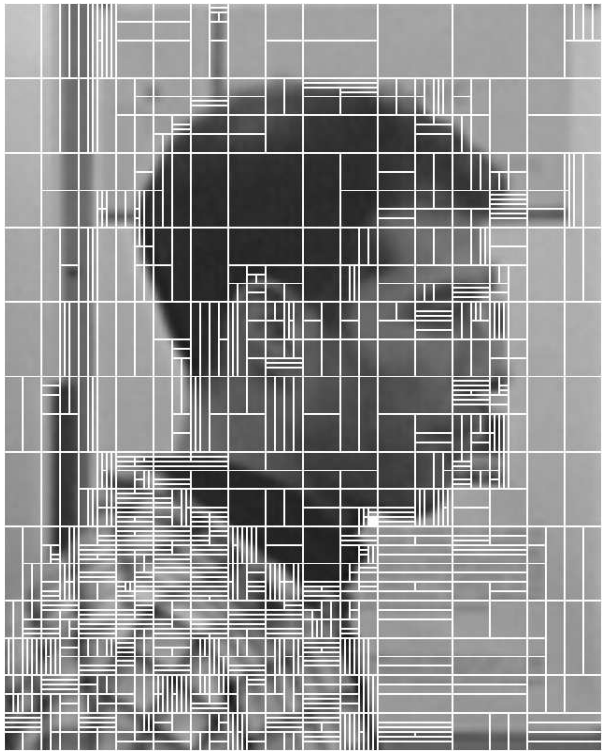


Figura 5.17: Segmentação de uma região de textura da imagem book arrival, câmera 08, *frame* 0, codificado com Lambda=10.

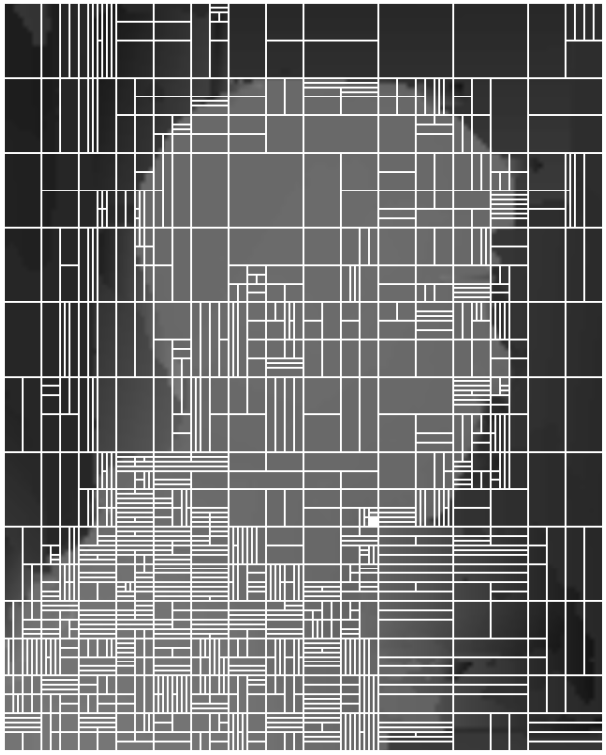


Figura 5.18: Segmentação de uma região de textura da imagem book arrival, câmera 08, *frame* 0, codificado com Lambda=10, sobreposta sobre o mapa de profundidade.

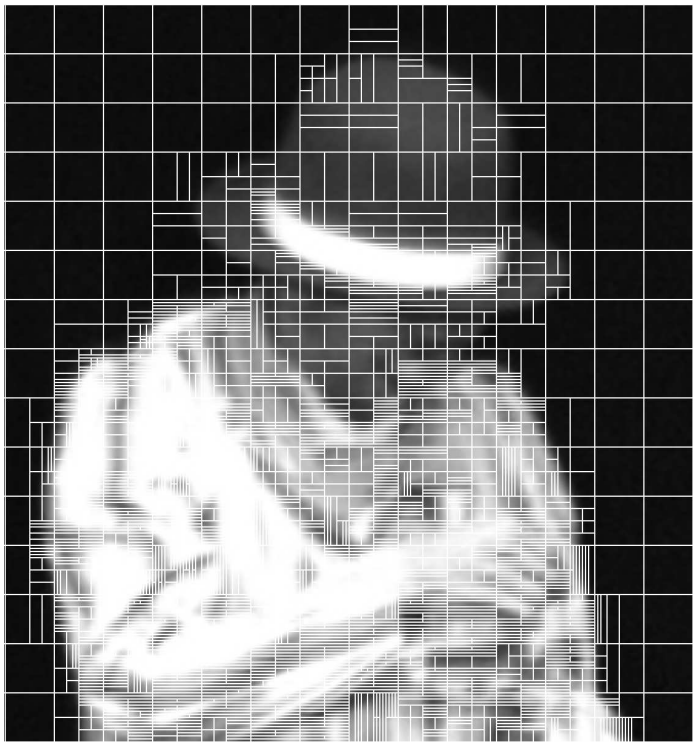


Figura 5.19: Segmentação de uma região de textura da imagem pantomime, câmera 37, *frame* 0, codificado com Lambda=10.

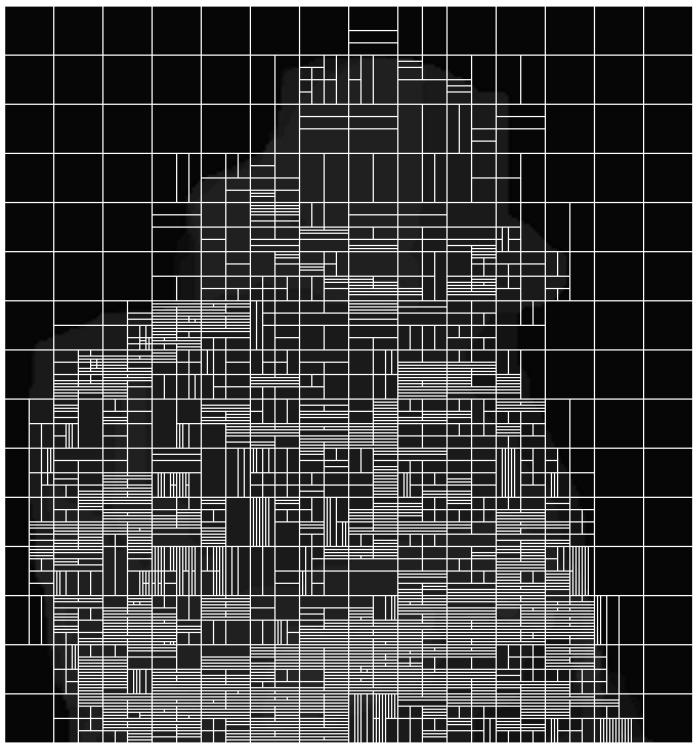


Figura 5.20: Segmentação de uma região de textura da imagem pantomime, câmera 37, *frame* 0, codificado com Lambda=10, sobreposta sobre o mapa de profundidade.

Tabela 5.2: Modos de predição usados por cada dicionário após a codificação da primeira imagem (textura) na sequência book arrival.

	Modos de Predição									
Dic.	0: vert.	1: hor.	2: MFV	3: diag. inf. esq.	4: diag. inf. dir.	5: vert. dir.	6: hor. inf.	7: vert. esq.	8: hor. sup.	9: LSP
0	0	0	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0	0	0
3	0	0	0	0	0	0	0	0	0	0
4	0	0	0	0	0	0	0	0	0	0
5	0	0	0	0	0	0	0	0	0	0
6	0	0	0	0	0	0	0	0	0	0
7	0	0	0	0	0	0	0	0	0	0
8	323	235	15	45	53	58	85	50	136	482
9	81	49	4	6	47	22	14	8	213	312
10	119	43	8	1	23	41	18	11	0	562
11	48	114	53	5	60	50	39	25	99	193
12	94	119	6	8	46	52	39	11	13	352
13	269	307	23	14	79	26	32	32	80	291
14	82	137	33	18	83	54	46	23	102	330
15	100	71	191	23	62	48	55	17	69	308
16	120	17	3	2	19	25	0	1	26	67
17	14	43	0	0	10	4	17	1	0	94
18	125	7	3	24	15	10	6	4	127	180
19	26	171	15	3	6	10	8	5	35	152
20	390	9	0	0	23	47	48	22	187	209
21	36	365	4	4	10	3	33	0	59	178
22	133	142	104	11	65	40	31	23	34	203
23	134	157	45	6	23	62	38	7	57	265
24	76	551	19	359	0	0	0	0	0	0

Capítulo 6

Alocação ótima de bits entre textura e profundidade

6.1 Introdução

Como descrito na seção 3.1.2, o MMP utiliza o critério taxa distorção para a codificação dos blocos de entrada, de acordo com o menor custo J, definido na equação 3.1. A variável λ é quem controla a proporção entre taxa e distorção na função custo, e esta função garante a alocação ótima de bits na codificação de uma imagem utilizando o MMP.

No entanto, quando uma vista virtual é estimada a partir de dois pares de imagens estereoscópicas codificadas (um par esquerdo e um par direito), não se sabe ainda o quanto se deve comprimir a imagem de textura e o mapa de profundidades de maneira a se obter a vista virtual com o melhor resultado.

Assim, decidiu-se fazer uma investigação sobre a relação de codificação entre a imagem de textura e o mapa de profundidade para obter uma vista reconstruída ótima. Uma vez que a qualidade da reconstrução de vistas virtuais é afetada pela qualidade da preservação das informações presentes nos mapas de profundidade, então, na reconstrução a partir e imagens de referência já comprimidas, o MMP não fica vulnerável aos efeitos de quantização nas regiões de alta frequência (bordas), já que não se baseia em transformadas de domínio de frequência.

6.2 Descrição do experimento

Consideremos primeiramente uma cena capturada por 3 (três) câmeras idênticas, cujos sistemas de coordenadas estejam alinhados, diferenciando-se apenas na posição de suas origens. Esse esquema é representado na figura 6.1.

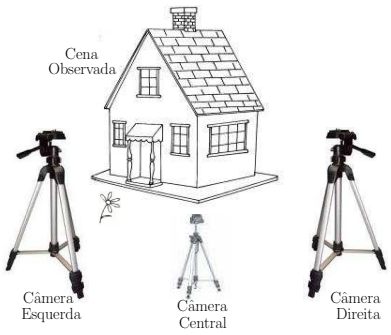


Figura 6.1: Representação de um cenário de múltiplas câmeras.

Cada câmera captura uma imagem de textura e um mapa de profundidade da cena observada. A partir da vista esquerda e da vista direita (imagem de textura e profundidade), pode-se "reconstruir" a vista central como uma vista virtual estimada pelas imagens de referência (laterais). A precisão dos mapas de profundidades das imagens de referência são fundamentais na síntese da imagem virtual, uma vez que os mapas farão o mapeamento de cada ponto dos objetos da cena no ponto virtual.

As imagens de textura da vista esquerda e da vista direita são codificadas com o MMP, determinando a função custo Lagrangeano com λ_T, e as imagens de profundidade da vista esquerda e da vista direita são codificadas usando a função custo Lagrangeano com λ_D. Este esquema é representado na figura 6.2.

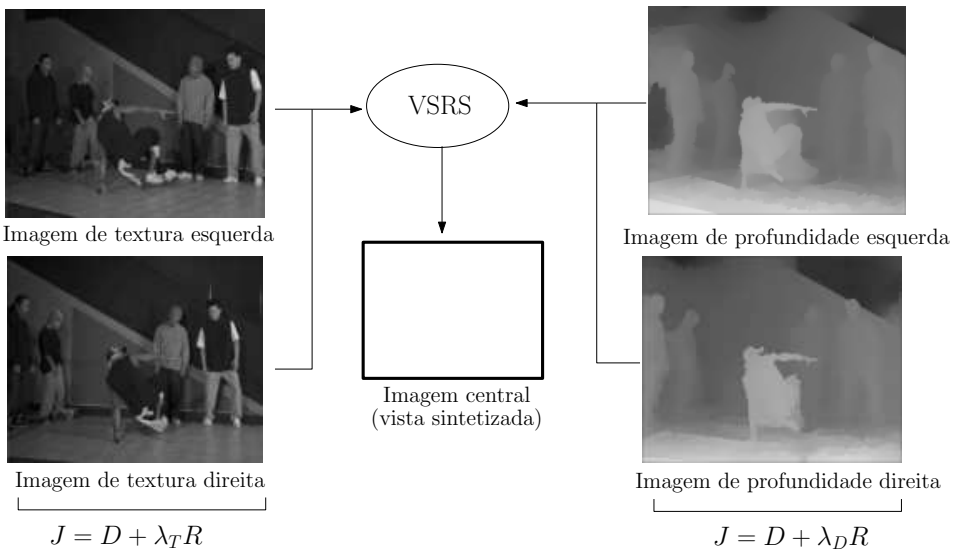


Figura 6.2: Esquema de reconstrução da vista central a partir de dois pares de imagens estéreo.

A imagem da vista central é reconstruída a partir dos dois pares de imagens estereoscópicas codificados com o MMP, e a imagem obtida é comparada com a imagem reconstruída a partir dos dois pares não codificados.

As sequências utilizadas para os testes são mostradas na figura 6.3.

Ballet [27]
Câm. 03 e 05, frame 0
(reconstói-se a cam. 04)

Champagne Tower [29]
Câm. 37 e 39, frame 0
(reconstói-se a cam. 38)

Breakdancers [27]
Câm. 03 e 05, frame 0
(reconstói-se a cam. 04)

Pantomime [29]
Câm. 37 e 39, frame 0
(reconstói-se a cam. 38)

Figura 6.3: Sequências de teste.

Logo, definindo:

$$\alpha = \frac{\lambda_D}{\lambda_T}, \tag{6.1}$$

o objetivo é encontrar α que fornece o melhor compromisso taxa $\times$ distorção ($R \times D$) na vista reconstruída.

Notas:

- O valor do PSNR é a medida de fidelidade da imagem reconstruída a partir das vistas codificadas em relação à imagem reconstruída a partir das vistas originais.
- A taxa R é a soma das taxas de todas as imagens necessárias para a reconstrução, ou seja, é a taxa de codificação da vista de textura esquerda + textura direita + vista de profundidade esquerda + profundidade direita.
- O *software* de reconstrução utilizado foi o VSRS (*View Synthesis Reference Software*) [24].

6.3 Resultados

Os gráficos preliminares das figuras 6.4, a 6.7 mostram que os melhores resultados da construção das vistas centrais, a partir de dois pares laterais de imagens estéreo codificados, são obtidos quando se comprime mais as imagens de textura do que as imagens de profundidade, uma vez que os mapas de profundidades fazem o mapeamento dos pontos virtais, e por isso deve-se preservar mais estas imagens utilizando-se um λ menor na compressão dos mapas de profundidade do que os valores de λ utilizados nas imagens de textura.

Em cada um dos gráficos seguintes, a curva do meio representa a reconstrução da vista central a partir de dois pares laterais cujo λ_D usado nas imagens de profundidade tem o mesmo valor do λ_T usado nas imagens de textura. Abaixo dela, há duas curvas, que representam respectivamente $lambda_D$ cinco e dez vezes maior do que λ_T. E acima da curva $\lambda_T = \lambda_D$, estão duas curvas, onde λ_T é cinco e em seguida dez vezes maior do que λ_D.

A notação LT significa "*lambda de textura*" e LD "*lambda de profundidade*".

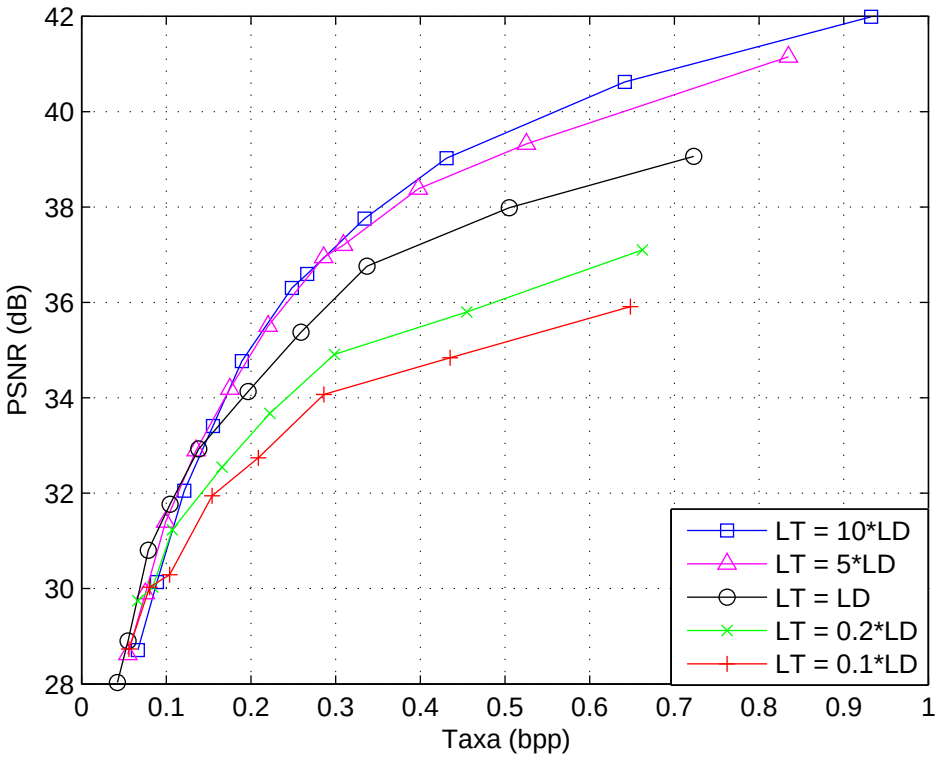


Figura 6.4: Ballet, câmera 04, *frame* 0, (imagem virtual).

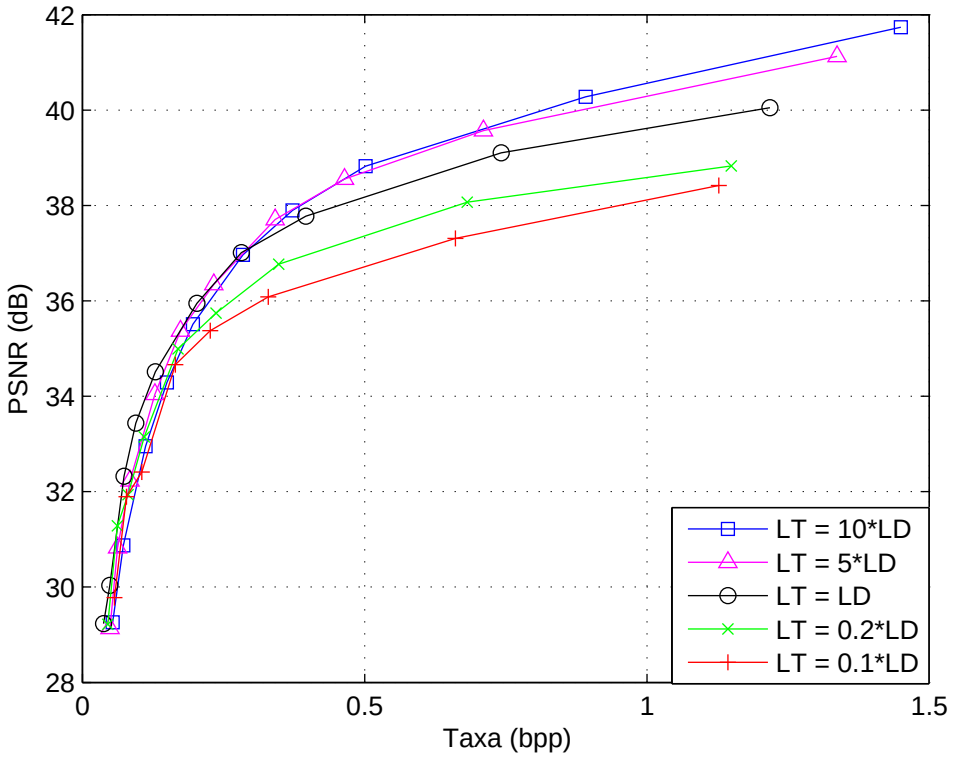


Figura 6.5: Breakdancers, câmera 04, *frame* 0, (imagem virtual).

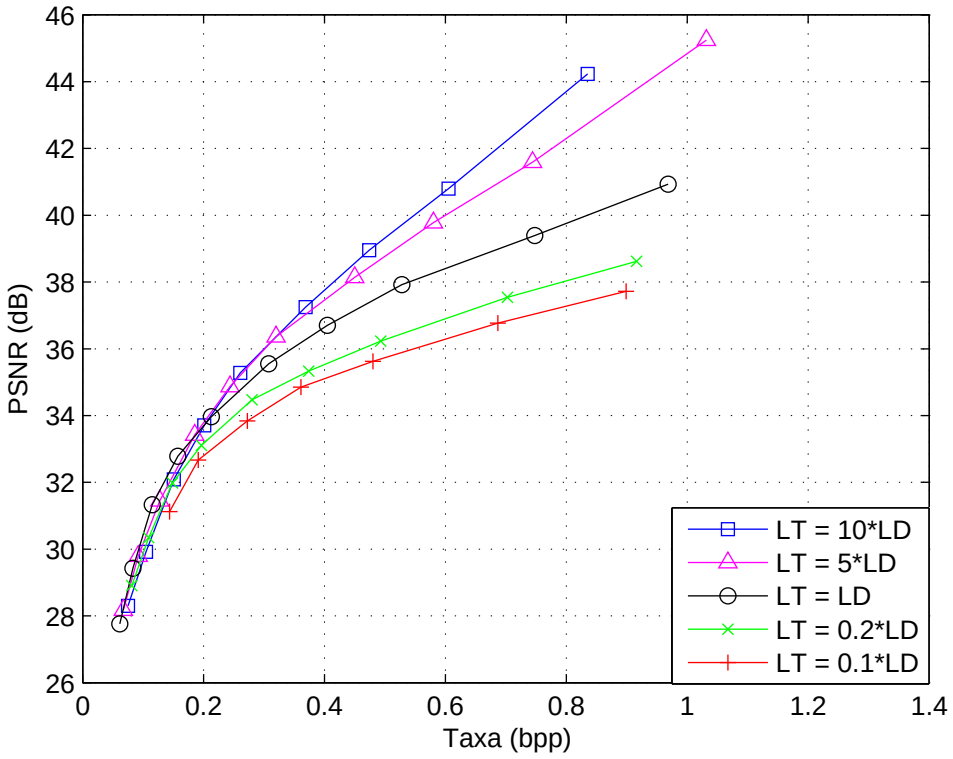


Figura 6.6: Champagne tower, câmera 38, *frame* 0, (imagem virtual).

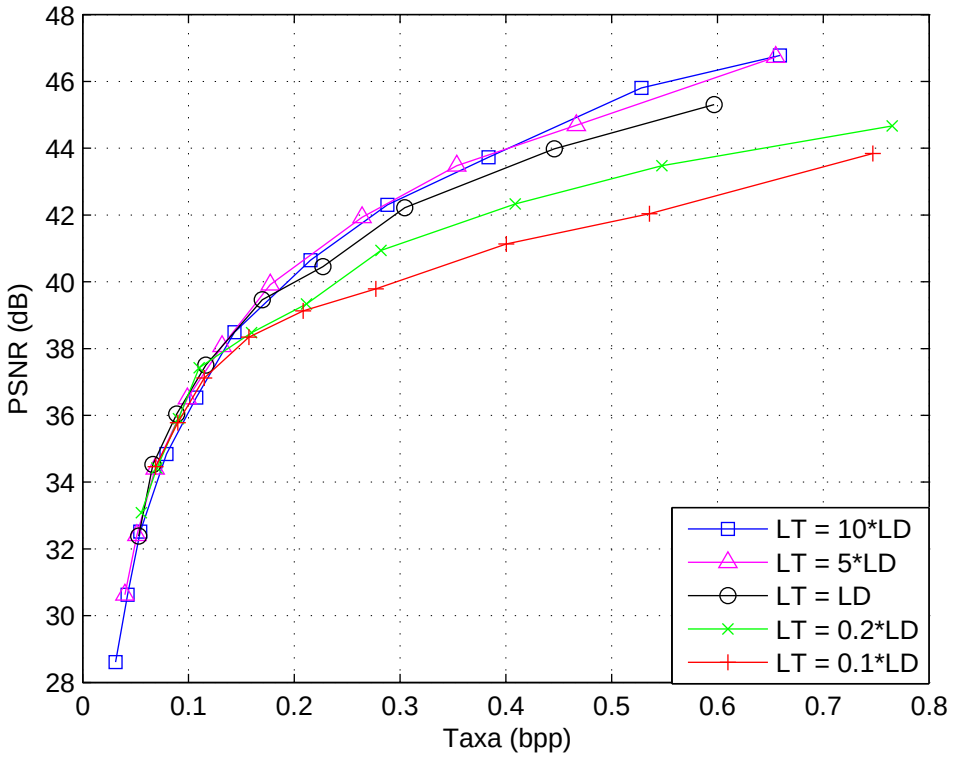


Figura 6.7: Pantomime, câmera 38, *frame* 0, (imagem virtual).

Então, para cada sequência de teste, foi construído um gráfico estabelendo várias curvas taxa × PSNR, onde λ_T e λ_D tenham a mesma proporção em toda curva, assim como os gráficos das figuras 6.4 a 6.7. A proporção, denominada α, é a mesma definida na equação 6.1.

Para cada sequência, foi feito um gráfico taxa × PSNR com 10 (dez) curvas, apresentando as seguintes proporções:

$$\begin{array}{ll} \lambda_T = \lambda_D & \lambda_T = 25\lambda_D \\ \lambda_T = 5\lambda_D & \lambda_T = 30\lambda_D \\ \lambda_T = 10\lambda_D & \lambda_T = 35\lambda_D \\ \lambda_T = 15\lambda_D & \lambda_T = 40\lambda_D \\ \lambda_T = 20\lambda_D & \lambda_T = 50\lambda_D \end{array}$$

Para facilitar a interpretação dos resultados, foram escolhidas algumas taxas em um intervalo apropriado para cada imagem, e, a partir destes intervalos, montou-se um outro gráfico mostrando o comportamento do PSNR, em dB, conforme a relação α varia de 1 a 50.

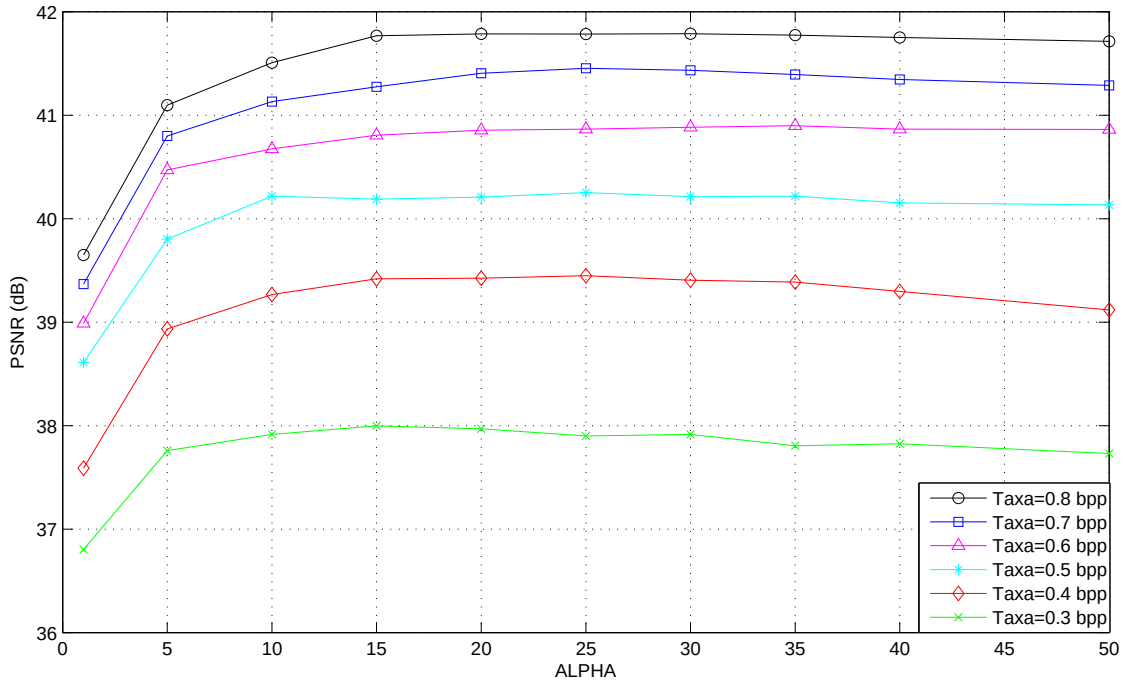


Figura 6.8: Ballet, câmera 04, *frame* 0, (imagem virtual).

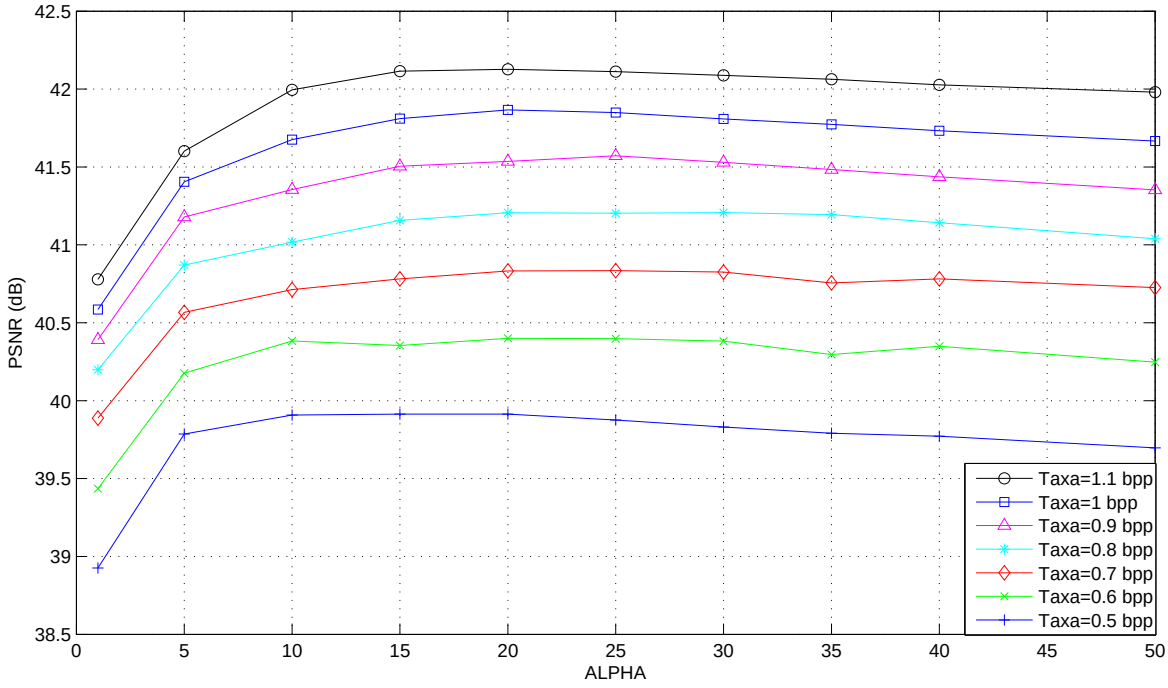


Figura 6.9: Breakdancers, câmera 04, *frame* 0, (imagem virtual).

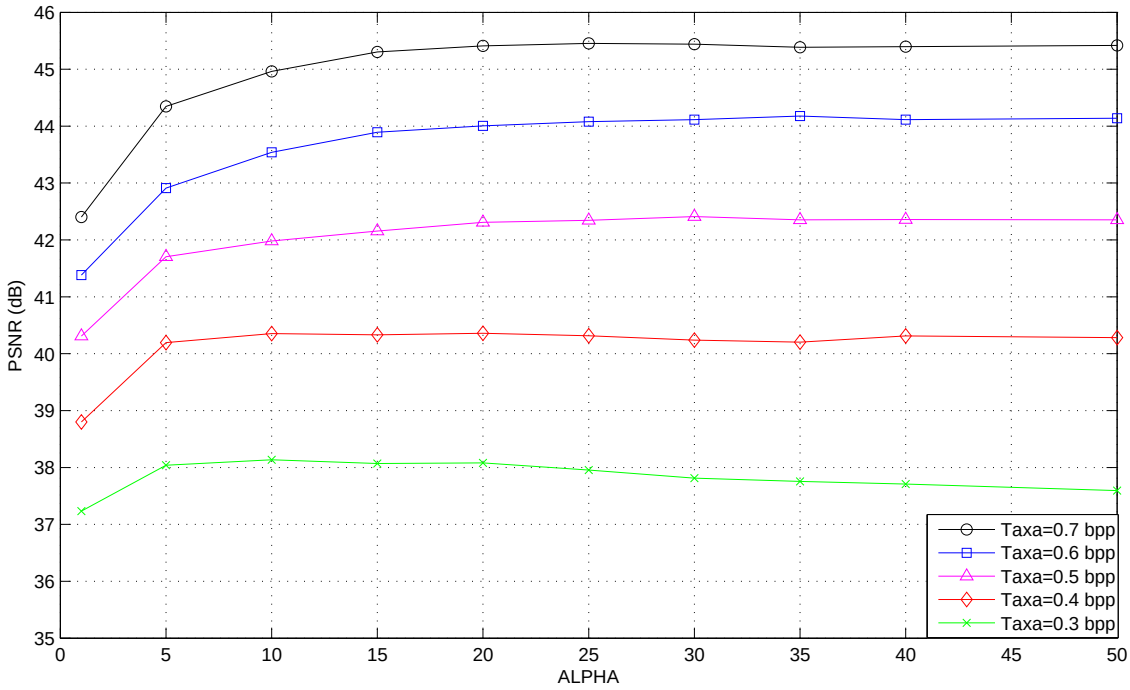


Figura 6.10: Champagne tower, câmera 38, *frame* 0, (imagem virtual).

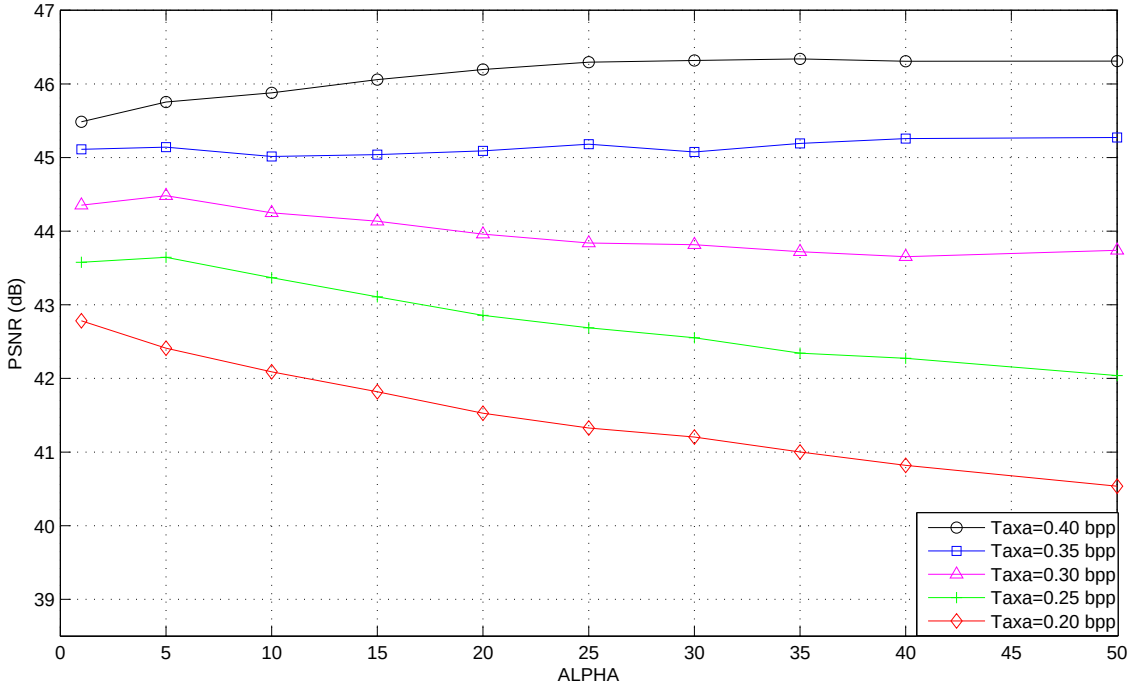


Figura 6.11: Pantomime, câmera 38, *frame* 0, (imagem virtual).

Como se pode observar nos gráficos apresentados nas figuras de 6.8 a 6.11, a relação α da equação 6.1 apresenta algumas mudanças de comportamento entre as diferentes imagens testadas, e apresenta ainda variações de acordo com a taxa.

Logo, para pesquisar qual é a melhor relação entre λ_T e λ_D para a obtenção de vistas virtuais, codificamos em seguida cada par de imagem estéreo mencionada na figura 6.3 com diversos valores de λ.

Para cada mapa de profundidade, foram realizadas 28 (vinte e oito) simulações com valores de λ_D entre 1 e 5000. Para cada imagem de textura, foram realizadas 60 (sessenta) simulações com valores de λ_T entre 1 e 5000. Os resultados de todas as combinações possíveis foram lançados no mesmo gráfico taxa $\times$ PSNR. Então é encontrada a curva ótima, localizando os pontos com melhor compressão taxa $\times$ distorção do gráfico (fecho côncavo). Finalmente, foi estabelecida uma relação empírica entre λ_T e λ_D que se aproximou da curva ótima para todas as sequências do experimento.

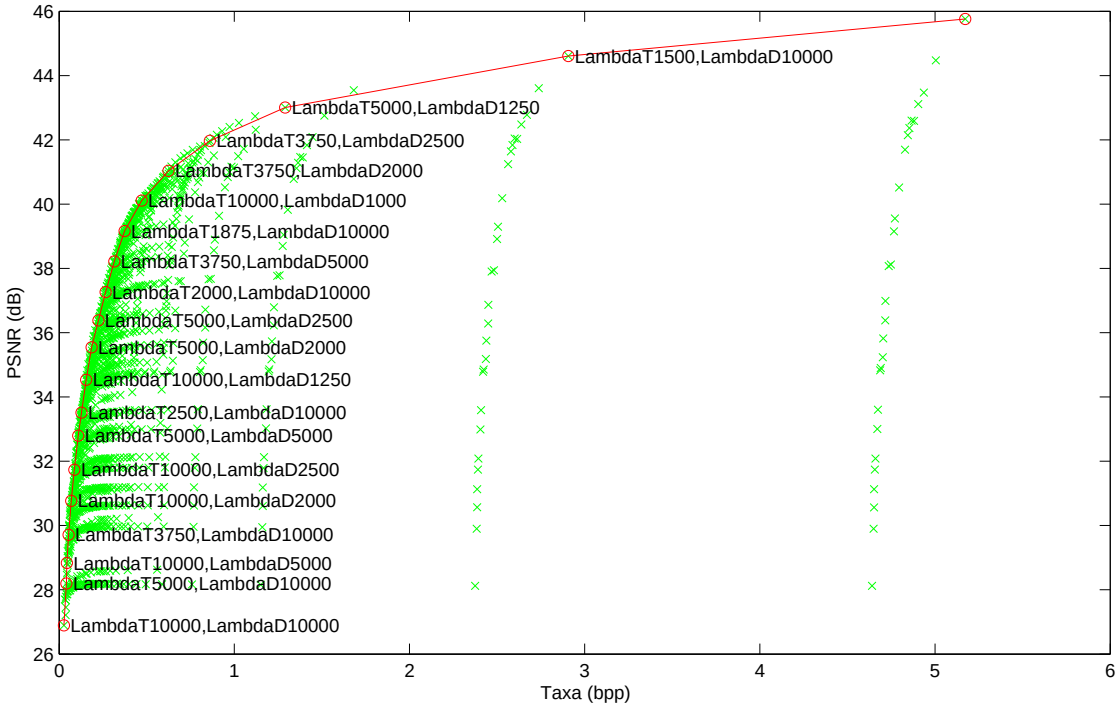


Figura 6.12: Ballet, câmera 04, *frame* 0, (imagem virtual).

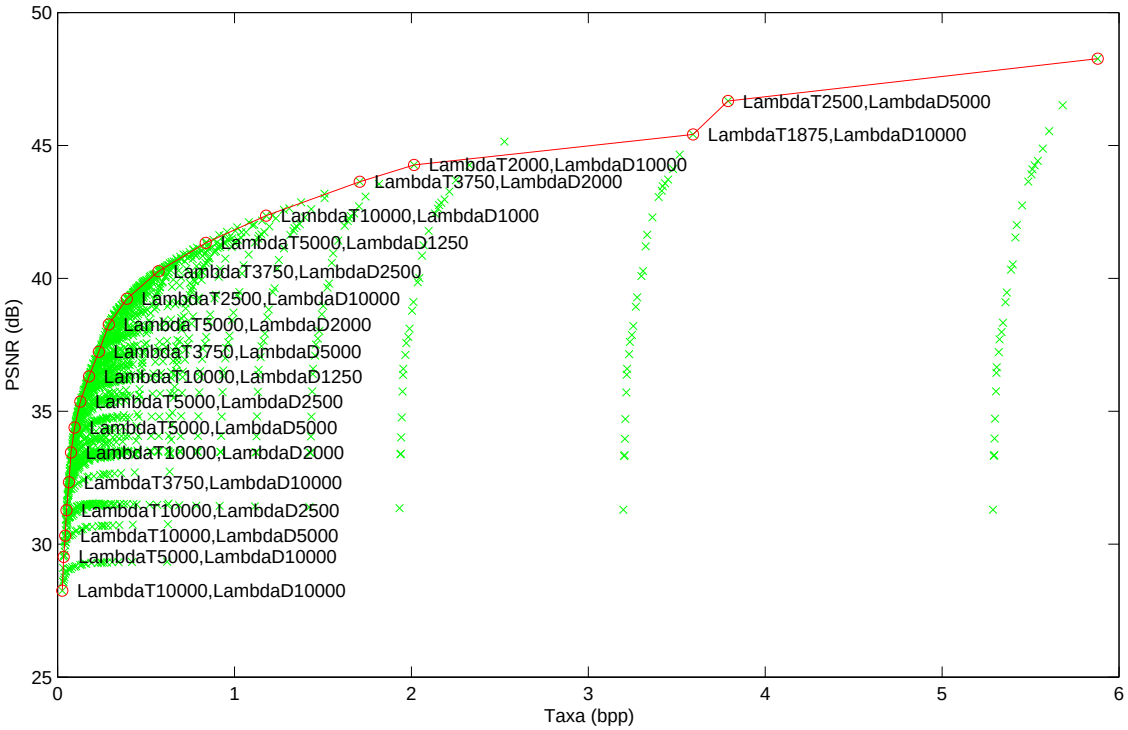


Figura 6.13: Breakdancers, câmera 04, *frame* 0, (imagem virtual).

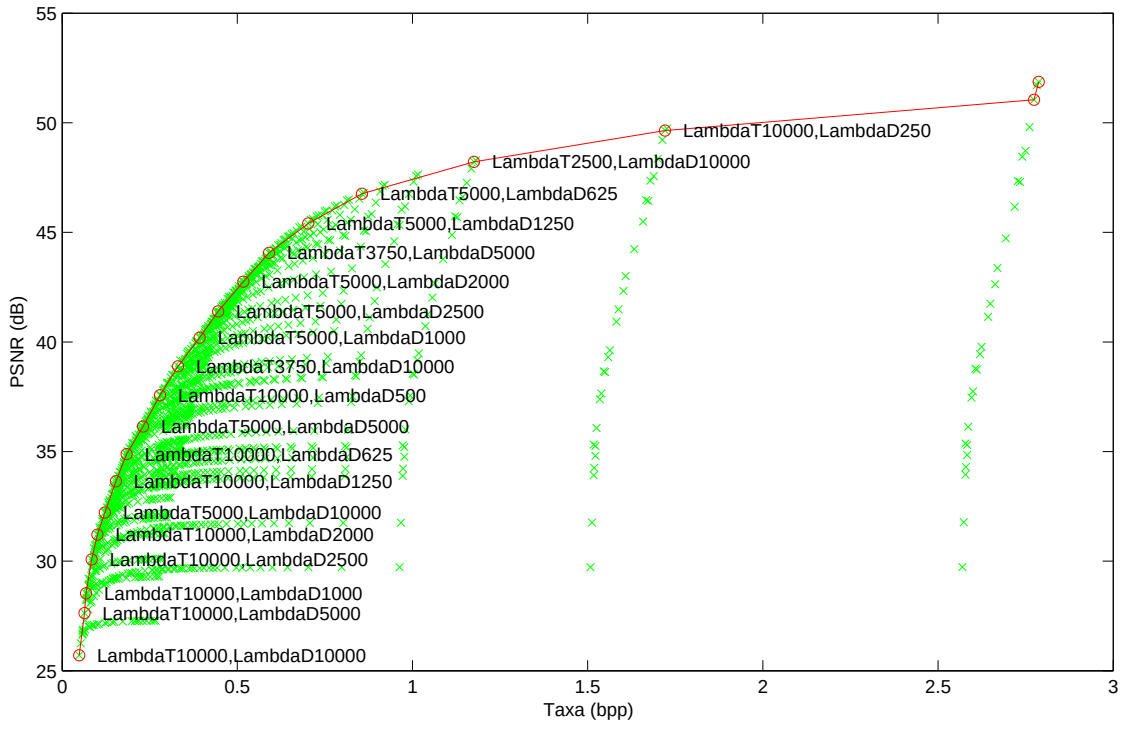


Figura 6.14: Champagne tower, câmera 38, *frame* 0, (imagem virtual).

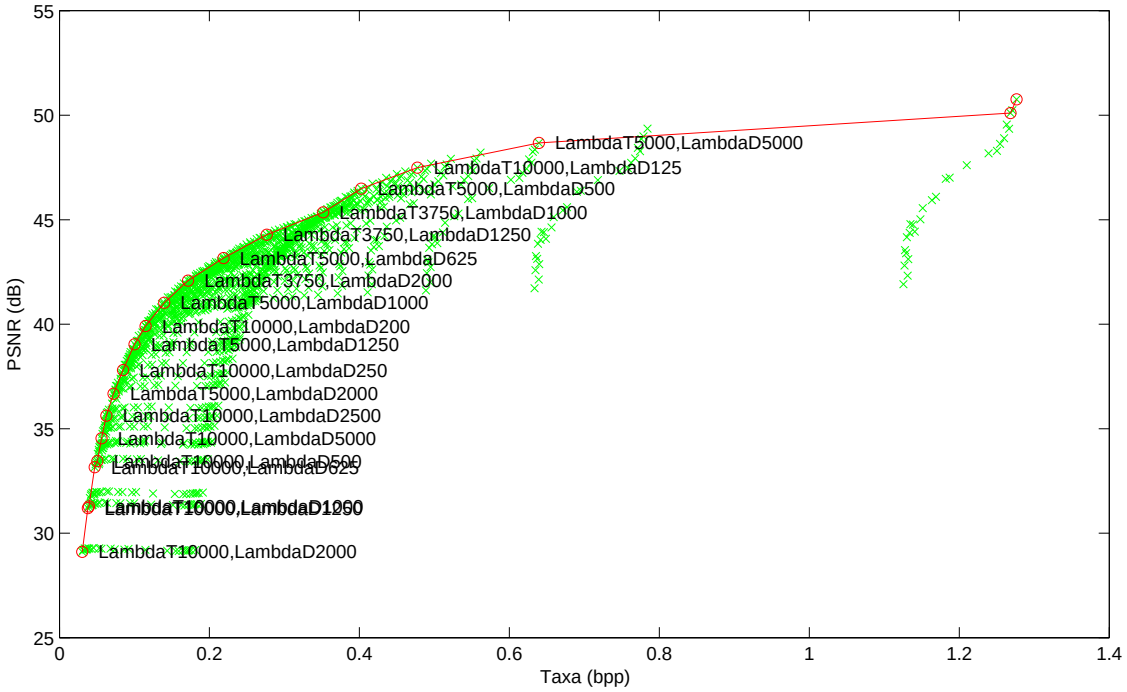


Figura 6.15: Pantomime, câmera 38, *frame* 0, (imagem virtual).

A partir dos pontos traçados nos gráficos das figuras 6.12, 6.13, 6.14 e 6.15, determinou-se experimentalmente a seguinte relação entre λ_T e λ_D:

Quando o Lambda de textura for:	Usar Lambda de profundidade:
$0 < LT \leq 10$	$LD = 0,25$
$10 < LT \leq 35$	$LD = 0,75$
$35 < LT \leq 70$	$LD = 10$
$70 < LT \leq 250$	$LD = 50$
$250 < LT \leq 500$	$LD = 100$
$500 < LT \leq 1000$	$LD = 500$
$1000 < LT \leq 5000$	$LD = 1000$

Tabela 6.1: Relação entre lambda de textura e lambda de profundidade

Os valores da tabela 6.1 proporcionam e relação taxa × distorção da vista reconstruída muito próximos das curvas ótimas obtidas nas figuras 6.12, 6.13, 6.14 e 6.15. A seguir apresentaremos gráficos das mesmas imagens, cujas curvas foram feitas a partir da heurística apresentada na tabela 6.1, comparados com os pontos ótimos das figuras 6.12 a 6.15.

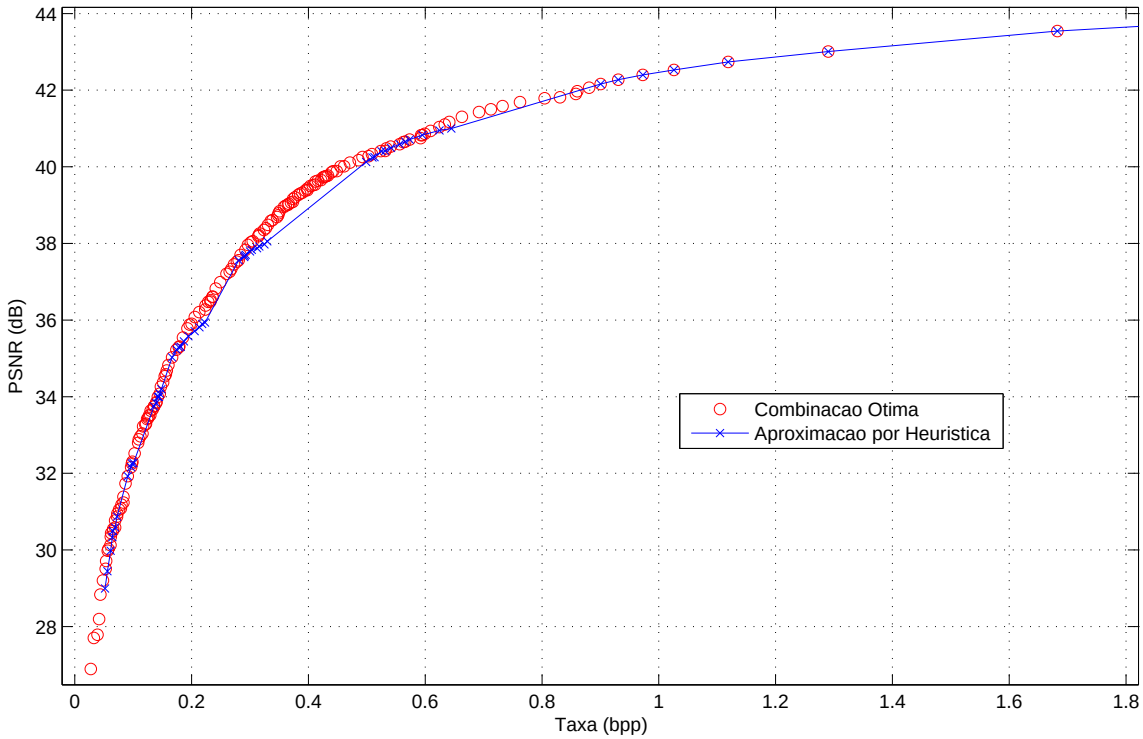


Figura 6.16: Ballet, câmera 04, *frame* 0, (imagem virtual).

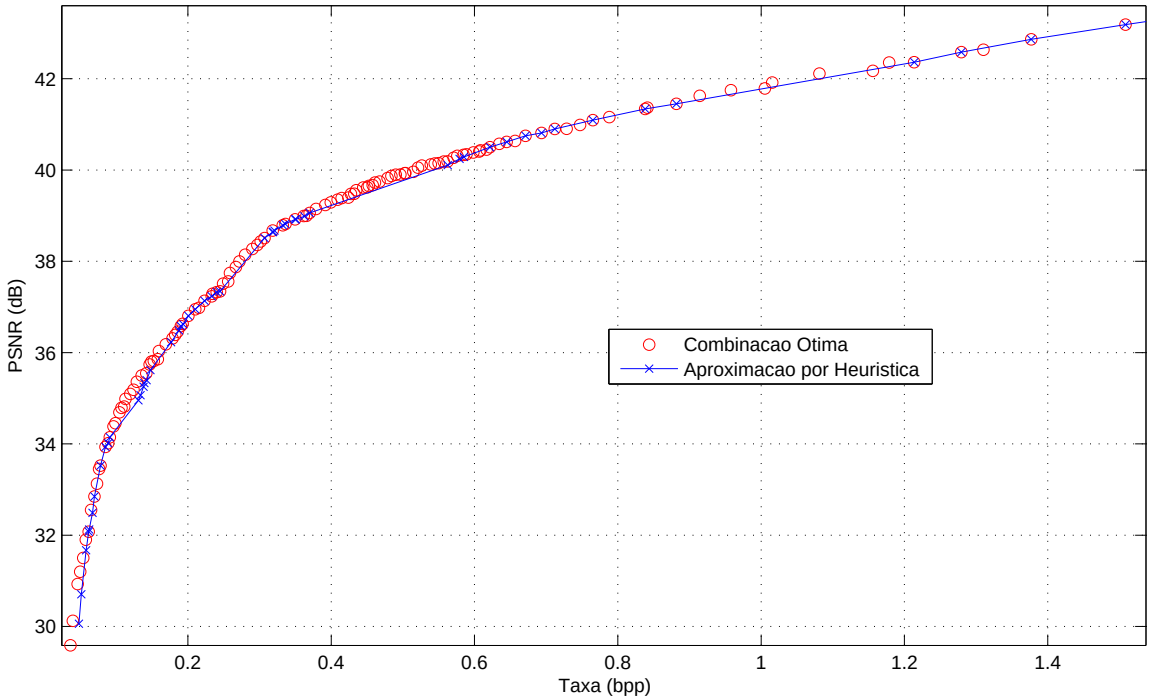


Figura 6.17: Breakdancers, câmera 04, *frame* 0, (imagem virtual).

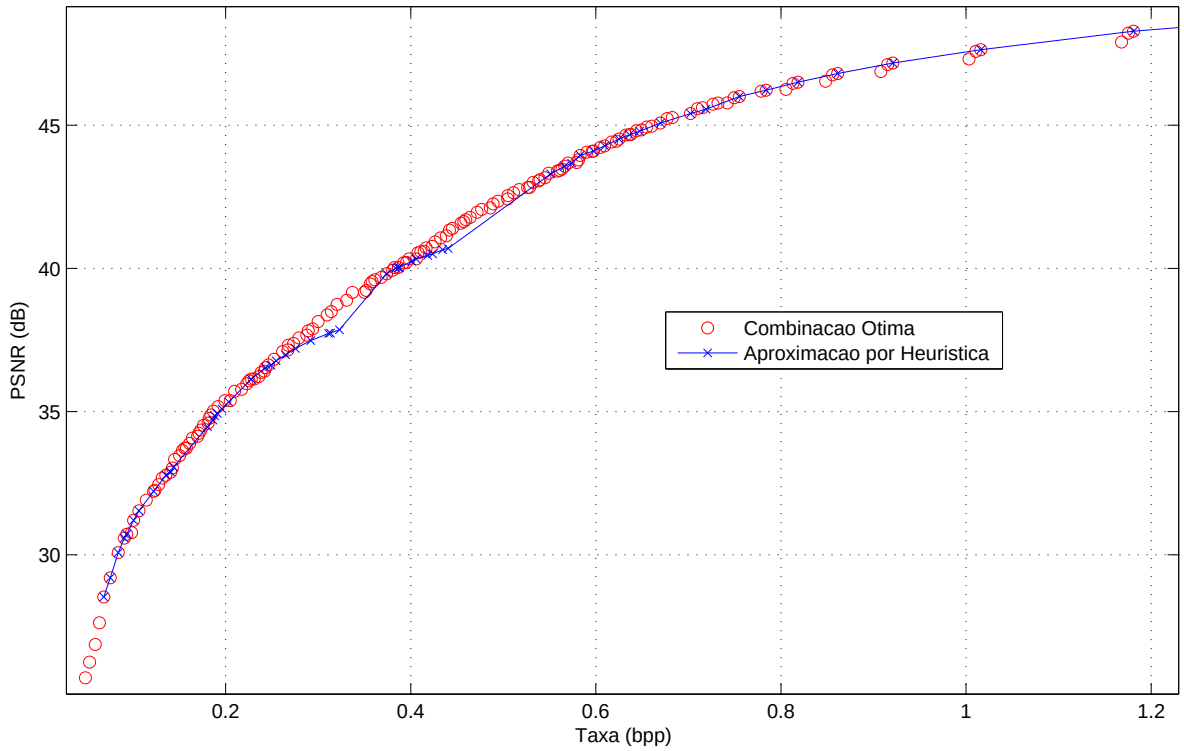


Figura 6.18: Champagne tower, câmera 38, *frame* 0, (imagem virtual).

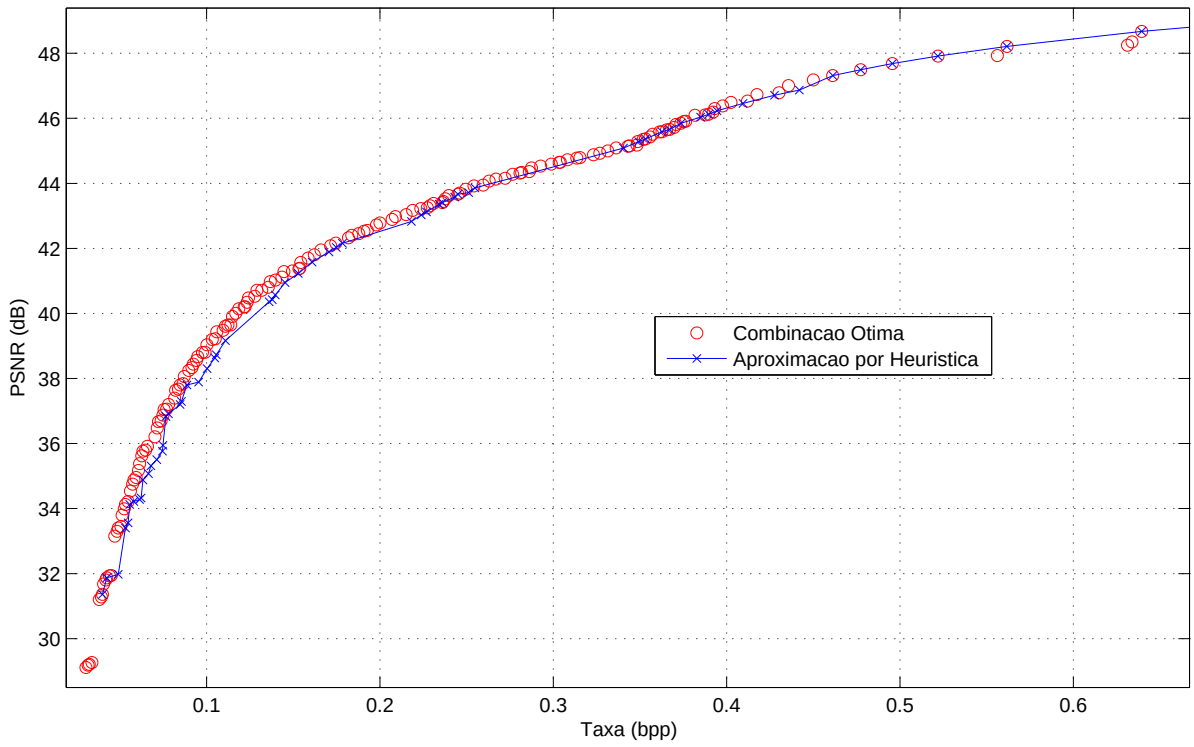


Figura 6.19: Pantomime, câmera 38, *frame* 0, (imagem virtual).

Na relação apresentada na tabela 6.1, os valores de λ_D são sempre menores do que os valores de λ_T, o que significa dizer que o mapa de profundidades deve ser codificado com mais bits para que as informações sejam preservadas, e assim a imagem virtual reconstruída apresente uma qualidade melhor.

Nos gráficos apresentados nas figuras 6.16 a 6.19 cada curva ótima é formada pelos melhores pontos, em termos de taxa $\times$ PSNR, dentre mais de 1.600 combinações de λ_T e λ_D em cada imagem reconstruída, que são expressas na "nuvem" de pontos mostradas nas figuras 6.12 a 6.15. Na aproximação por heurística, cada lambda de profundidade λ_D, com valores distribuídos entre 1 e 500, foi combinado (dentre sessenta opções de combinação) com um lambda de textura λ_T de acordo com a tabela 6.1 para a reconstrução da imagem virtual, e o resultado da reconstrução destas imagens se aproximou bastante em todas as imagens e em muitos pontos até mesmo coincidiu com a curva ótima, conforme observado nos gráficos.

Capítulo 7

Alocação ótima de bits utilizando regiões de interesse no mapa de profundidades

7.1 Introdução

No capítulo 6, foi observado que uma vista virtual sintetizada a partir de dois pares de imagem estéreo codificadas apresenta melhor resultado quando os mapas de profundidade das imagens de referência estão mais "intactos" da codificação do que a imagem de textura correspondente. Ou seja, é obtido um resultado melhor quando se comprime mais a imagem de textura e se preserva mais o mapa de profundidades.

Assim, é intuitivo que a preservação de informações importantes do mapa de profundidade durante o processo de codificação melhore a qualidade das imagens virtuais que são estimadas a partir das imagens de referência codificadas.

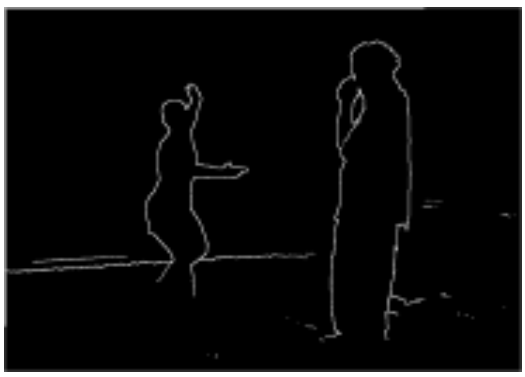
Em [30][1], foi introduzido no algoritmo MMP um detector de bordas (*edge aware*), com o propósito específico de codificar mapas de profundidade, gastando mais bits nas regiões das bordas, de maneira a preservar essa informação. Embora haja um aumento na taxa de codificação, é também viável supor que a melhora obtida na sintetização da vista virtual a partir do novo mapa de profundidade codificado compense esse aumento de taxa.

[1]Tese com defesa prevista para abril de 2011

7.2 Descrição do experimento

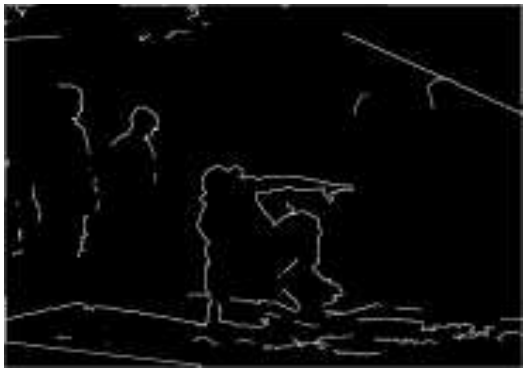
Assim como os experimentos realizados no capítulo 6, aqui deseja-se também sintetizar uma vista virtual a partir de dois pares de imagens estéreo (textura e profundidade, esquerda e direita). Porém, na codificação do mapa de profundidade, o algoritmo MMP foi adaptado colocando-se além da imagem de entrada (mapa de profundidade), uma imagem correspondente a ela representando as bordas. Esta nova imagem é definida como *máscara*. O MMP lê ambas as imagens e nas regiões correspondentes às bordas, o algoritmo faz um tratamento especial, priorizando a codificação com mais bits.

Então, para a definição das máscaras, foi construído em [30], um algoritmo matlab para detecção de bordas pelo método do limiar para *binarização* da imagem. Assim, foram criadas as seguintes máscaras:



Ballet

Champagne Tower



Breakdancers

Pantomime

Figura 7.1: Máscaras de bordas

7.3 Resultados

Primeiramente, apresentamos a seguir os gráficos de taxa × PSNR da codificação dos mapas de profundidade (vista esquerda) para as sequências ballet, breakdancers, champagne tower e pantomime com e sem *edge aware*. Podemos observar nos

gráficos das figuras 7.2 que, em todos os gráficos, o uso de *edge aware* apresenta um comportamento inferior em baixas taxas do que a codificação sem *edge aware*. A partir de certo ponto em cada uma das imagens, o uso de *edge aware* apresenta vantagens em termos de taxa × PSNR, como podemos ver nos gráficos correspondentes. Para estas taxas, as vistas reconstruídas associadas a cada mapa podem apresentar resultados melhores do que a reconstrução sem uso de *edge aware*.

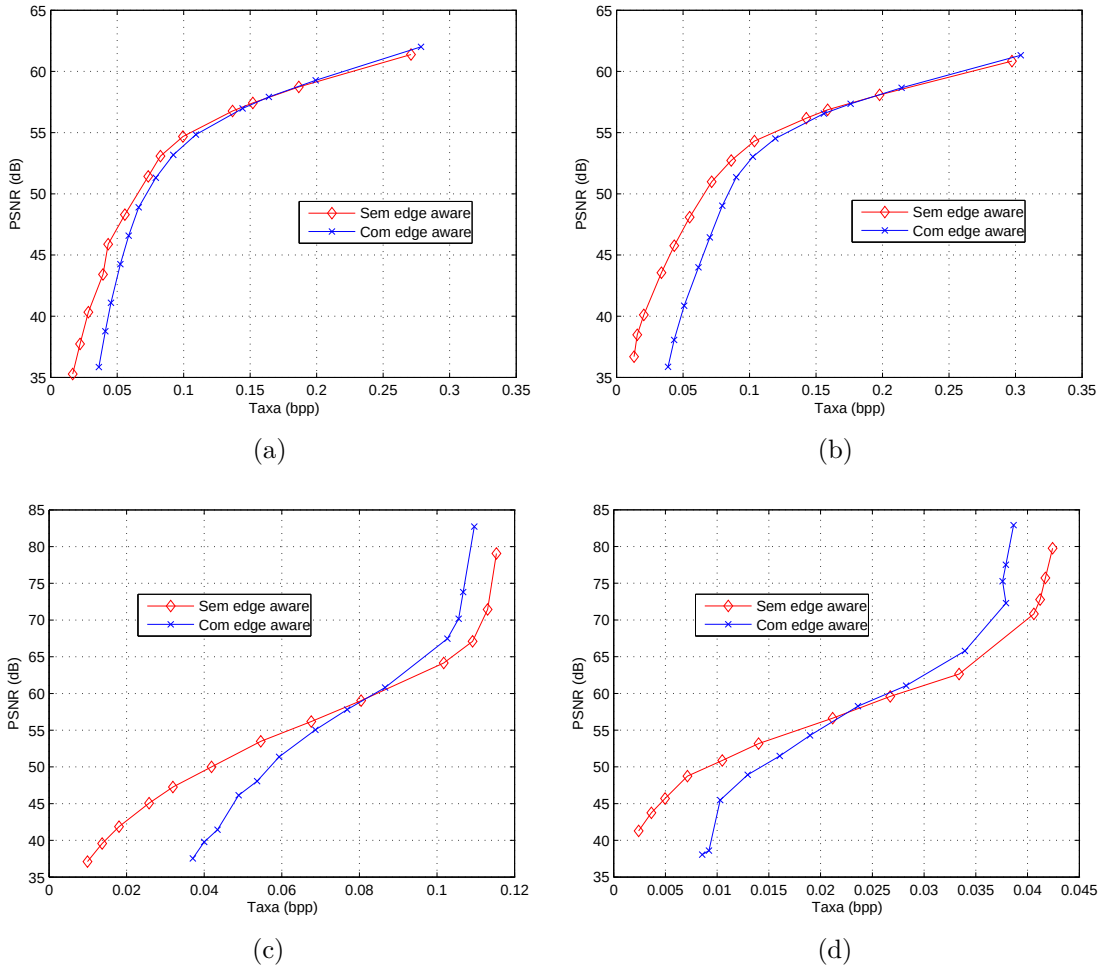


Figura 7.2: Mapas de profundidades (vista esquerda) codificados com e sem *edge aware*: a) ballet; b) breakdancers; c) champagne tower; d) pantomime.

Uma vez conhecido que as vistas sintetizadas a partir de pares estéreo codificados apresentam resultado melhor quando o mapa de profundidades está mais preservado do que a imagem de textura (que no caso do MMP significa dizer que $\lambda_T > \lambda_D$), as simulações realizadas nesta etapa se consistem nas mesmas apresentadas no capítulo 6, construindo-se curvas taxa × PSNR, onde λ_T e λ_D tenham a mesma proporção α em toda curva. A seguir, foram escolhidas as mesmas das figuras taxas escolhidas nos gráficos de 6.8 a 6.11, e, a partir dos intervalos, montou-se um outro gráfico para cada imagem, mostrando o comportamento do PSNR, em dB, conforme a relação α varia de 1 a 50.

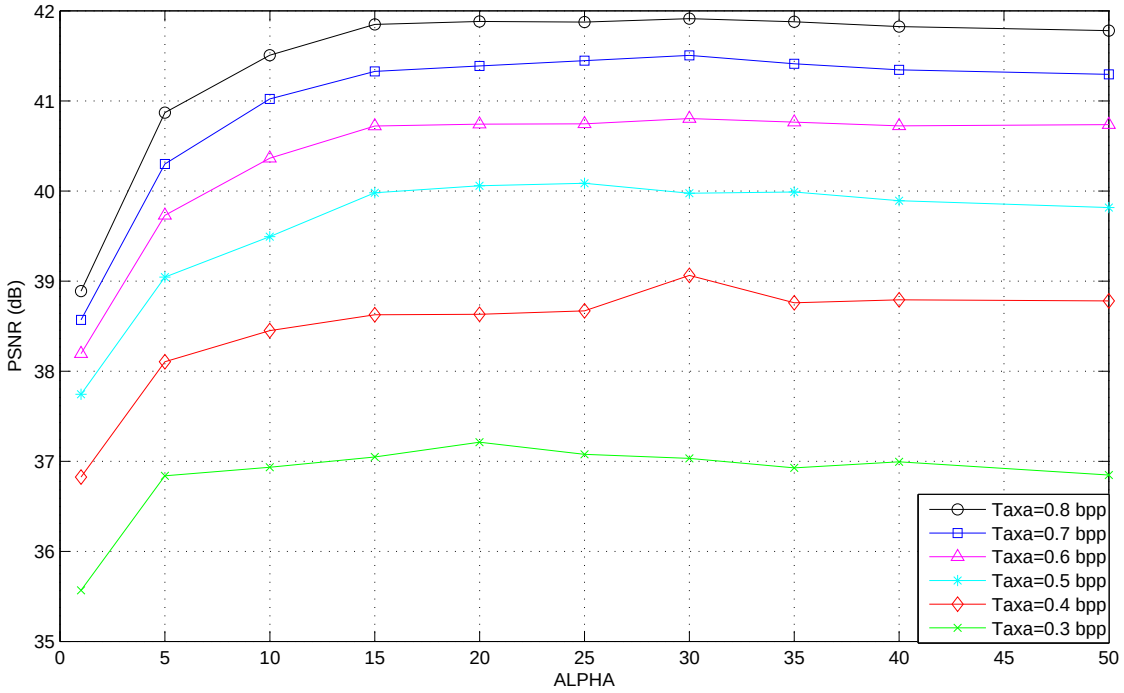


Figura 7.3: Ballet, câmera 04, com edge aware.

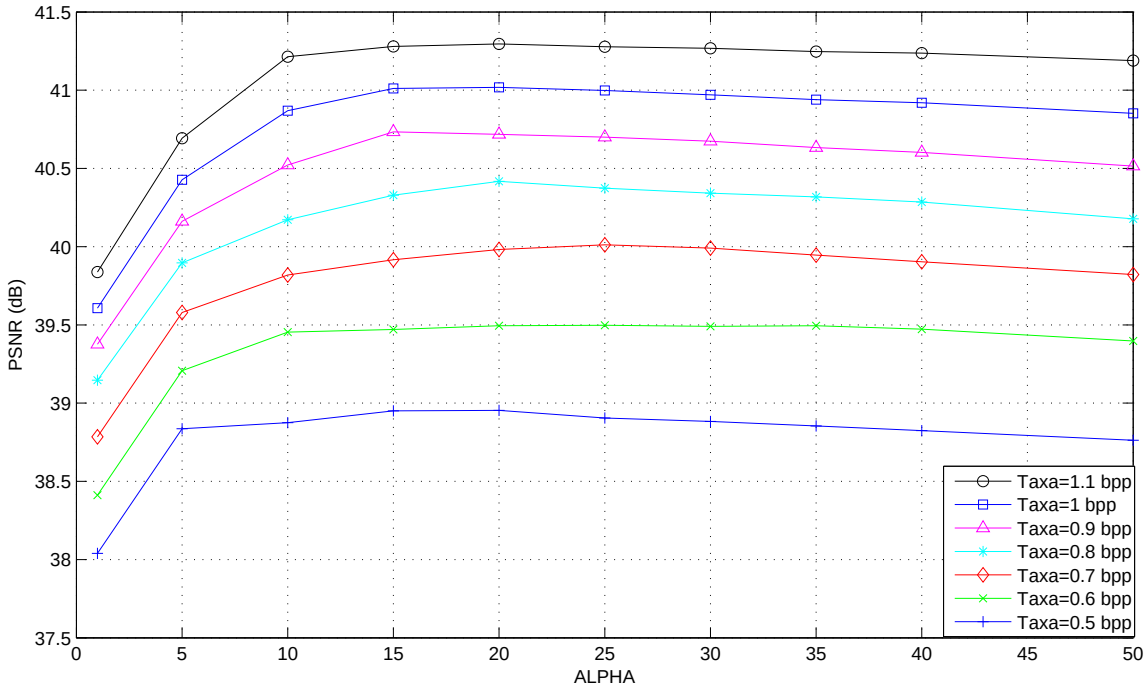


Figura 7.4: Breakdancers, câmera 04, com edge aware.

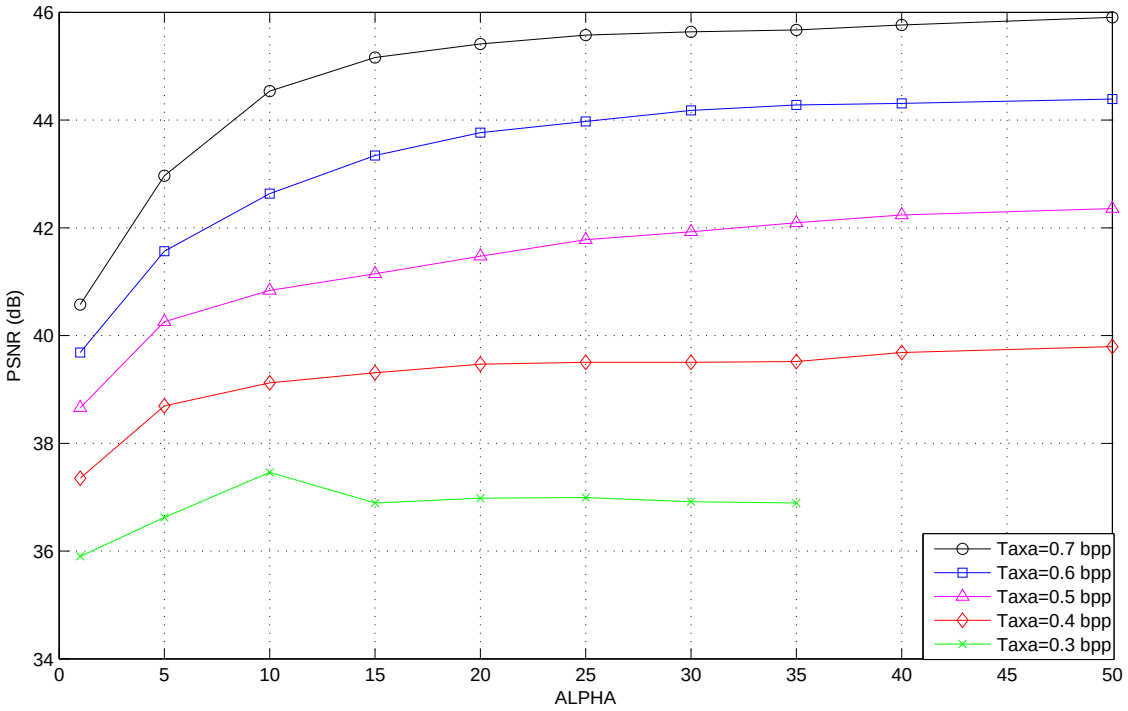


Figura 7.5: Champagne tower, câmera 38, com edge aware.

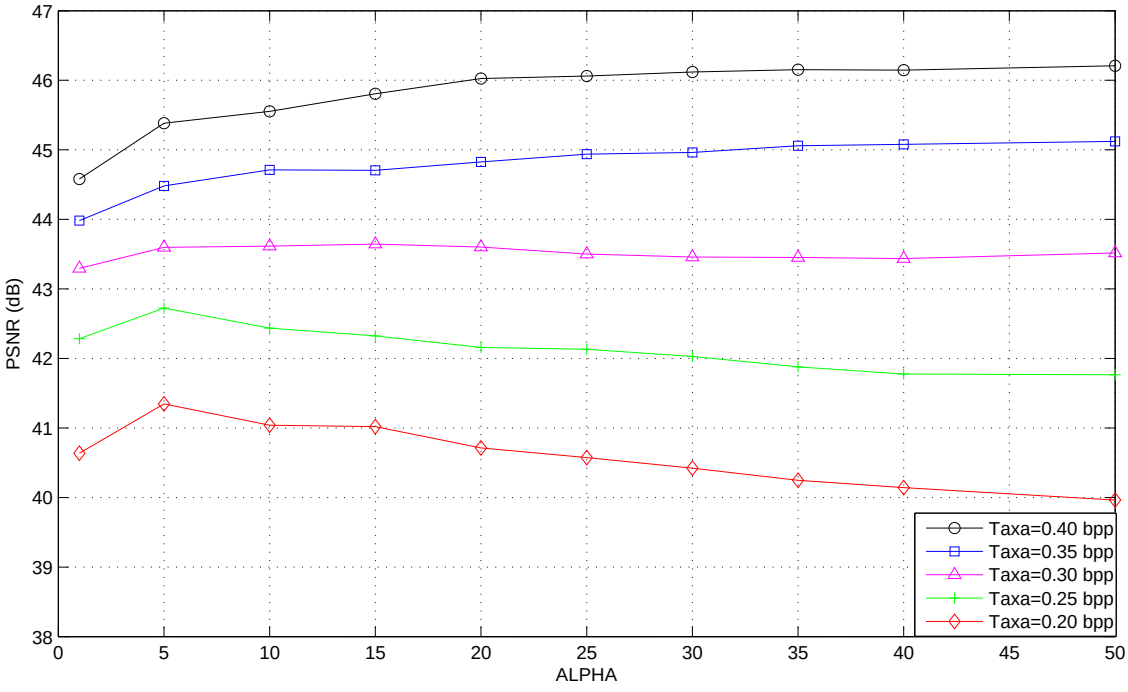
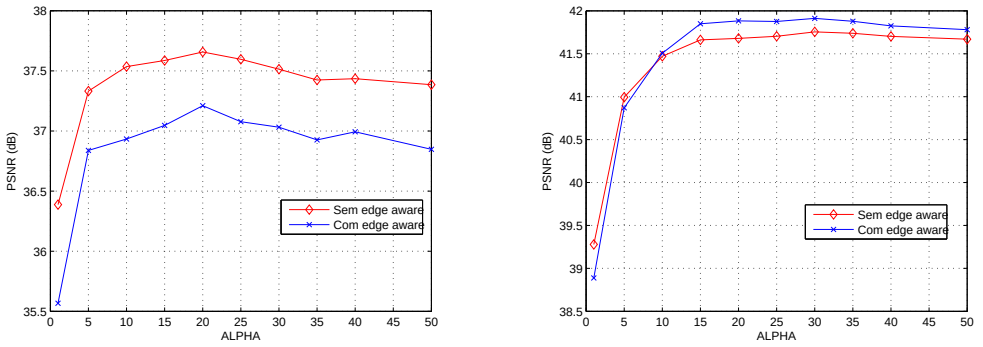


Figura 7.6: Pantomime, câmera 38, com edge aware.

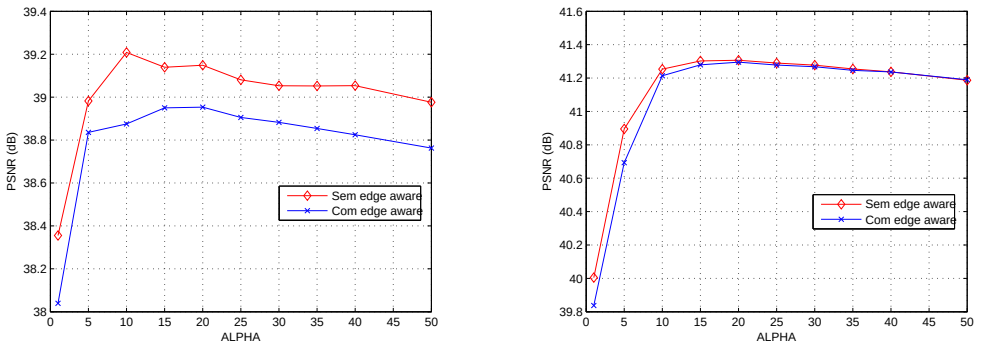
Os gráficos das figuras 7.3 a 7.6 apresentam o comportamento de taxa × PSNR para as mesmas taxas analisadas no capítulo 6, nas figuras de 6.8 a 6.11. Comparando o resultado desses dois procedimentos, podemos observar que, de uma maneira geral, para todas as taxas plotadas de reconstrução com e sem *edge aware*, a alocação ótima do MMP, que ocorre quando é usado o mesmo λ para codificar todos os blocos (ou seja, sem o uso de *edge aware*), é melhor em termos de taxa × PSNR do que priorizar regiões do mapa de profundidade quando se deseja reconstruir uma imagem virtual.

As figuras 7.7 e 7.8 mostram respectivamente, para as imagens ballet e breakdancers, o comportamento do PSNR para a menor e a maior taxas testadas para essas imagens. Para a maioria das taxas, os resultados sem o uso de *edge aware* foram melhores. A figura 7.7(b) mostra um caso pontual onde o uso de *edge aware* foi melhor, porém esse comportamento não foi observado nas demais imagens, mesmo para altas taxas, daí não é possível generalizar um comportamento.



(a) (b)

Figura 7.7: Ballet, câmera 04 (imagem virtual); a) 0,3 bpp (menor taxa avaliada para esta imagem); b) 0,8 bpp (maior taxa avaliada para esta imagem)



(a) (b)

Figura 7.8: Breakdancers, câmera 04 (imagem virtual); a) 0,5 bpp (menor taxa avaliada para esta imagem); b) 1,1 bpp (maior taxa avaliada para esta imagem)

Capítulo 8

Conclusões

Os resultados apresentados para a codificação conjunta de imagens de textura e profundidade, na seção 5.2, mostram que, embora no início da codificação da imagem de textura se tenha mais padrões no dicionário (provenientes do mapa de profundidade), os blocos obtidos apresentam poucos padrões de textura, já que as imagens de profundidade têm a característica de apresentar grandes regiões uniformes. Para os blocos de entrada da imagem de profundidade, então, os modos de predição mais usados serão os modos vertical, horizontal e DC. Uma vez que a imagem seguinte apresenta muitos detalhes (textura), os blocos da segunda imagem podem ser melhor preditos com outros modos de predição (como diagonais), ou ainda com outros tamanhos de bloco (blocos menores).

Dessa forma, ao invés de termos uma probabilidade bem adaptada, teremos uma probabilidade mal adaptada, o que irá fazer com que o codificador aritmético tenha o trabalho de adaptar novamente a probabilidade para a imagem de textura. Além disso, todos os elementos adicionados, vindos do mapa de profundidade (e que na sua maioria não foram usados na codificação da imagem de textura), requerem um novo índice, aumentando a entropia média e a taxa de codificação da imagem de textura.

Assim também na seção 5.3, a codificação conjunta cria um cenário onde é obtido um dicionário mal adaptado para a segunda imagem. Como a imagem de textura apresenta muitos detalhes, é normal que haja muitas segmentações dos blocos de entrada. O número de elementos inseridos aumenta muito a taxa de codificação para a segunda imagem (mapa de profundidade).

No capítulo 6, observamos pelos gráficos apresentados nas figuras 6.16 a 6.19 que a relação empírica encontrada e apresentada na tabela 6.1 se aproximou satisfatoriamente da curva ótima estabelecida para cada imagem reconstruída em todas as sequências testadas.

Por fim, no capítulo 7, embora tenha sido observado que a preservação do mapa de profundidade em relação a imagem de textura resulte na melhor síntese de vistas

virtuais, o uso de *edge aware* na codificação de mapas de profundidade, priorizando a região das bordas, gasta mais bits para codificar essas regiões, aumentando bastante a taxa de codificação. A melhora na preservação das bordas dos mapas de profundidades não se reflete numa melhora da vista virtual sintetizada.

8.1 Trabalhos futuros

Para trabalhos futuros, são sugeridas duas aplicações que visam a investigação da performance do algoritmo MMP. Estas são:

- Otimização da relação entre o desempendo do algoritmo MMP e o tempo de codificação, uma vez que, embora a performance do MMP supere outros algoritmos de compressão, o tempo de execução do programa ainda é muito alto.
- Aplicação do MMP a vídeos 3-D.

Referências Bibliográficas

[1] SHANNON, C. E. "A Mathematical Theory of Communication", *The Bell System Technical Journal*, v. 27, pp. 379–423, out. 1948.

[2] HUFFMAN, D. A. "A Method for the Construction of Minimum Redundancy Codes", *Proceedings of the IRE*, v. 40, pp. 10989–1101, 1951.

[3] ABRANSOM, N. *Information Theory and Coding*. New York, McGraw-Hill, 1963.

[4] J. ZIV, A. LEMPEL "A Universal Algorithm for Data Compression", *IEEE Trans. Inf. Theory*, v. 23, pp. 337–343, 1977.

[5] J. ZIV, A. LEMPEL "A Compression of Individual Sequences Via Variable-Rate Coding", *IEEE Trans. Inf. Theory*, v. 24, pp. 530–536, 1978.

[6] COVER, T. M., THOMAS, J. A. *Elements of Information Theory*. 2 ed. New Jersey, John Wiley and Sons Inc, 2006.

[7] DE CARVALHO, M. B. *Compression of Multidimensional Signals based on Recurrent Multiscale Patterns*. Tese de D.Sc., COPPE/UFRJ, Rio de Janeiro, RJ, Brasil, 2001.

[8] DA SILVA JÚNIOR, W. S. *Compressão de Imagens Utilizando Recorrência de Padrões Multiescalas com Segmentacão Flexível*. Tese de M.Sc., COPPE/UFRJ, Rio de Janeiro, RJ, Brasil, 2004.

[9] N. M. M. RODRIGUES, E. A. B. DA SILVA, M. B. DE CARVALHO, S. M. M. DE FARIA, V. M. M. SILVA "Universal image coding using multiscale recurrent patterns and prediction", *IEEE International Conference on Image Processing*, pp. II–245–II–248, set. 2005.

[10] "Joint Video Team (JVT) of ISO/IEC MPEG & ITU-T VCEG (ISO/IEC JTC1/SC29/WG11 and ITU-T SG16 Q.6), Draft of Version 4 of H.264/AVC". ITU-T Recommendation H.264 and ISO/IEC 14496-10 (MPEG-4 part 10) Advanced Video Coding, March 2005.

[11] N. C. FRANCISCO, N. M. M. RODRIGUES, E. A. B. DA SILVA, M. B. DE CARVALHO, S. M. M. DE FARIA, V. M. M. DA SILVA "Scanned Compound Document Encoding Using Multiscale Recurrent Patterns", *IEEE Transactions on Image Processing*, v. 19, n. 10, pp. 2712–2724, out. 2010.

[12] D. B. GRAZIOSI, N. M. M. RODRIGUES, E. A. B. DA SILVA, S. M. M. DE FARIA, M. B. DE CARVALHO "Improving Multiscale Recurrent Pattern Image Coding with Least-Squares Prediction Mode", *IEEE International Conference on Image Processing*, pp. 2813–2816, nov. 2009.

[13] N. M. M. RODRIGUES, E. A. B. DA SILVA, M. B. DE CARVALHO, S. M. M. DE FARIA, V. M. M. SILVA "On Dictionary Adaptation for Recurrent Pattern Image Coding", *IEEE Transactions on Image Processing*, v. 17, n. 9, pp. 1640–1653, set. 2008.

[14] N. C. FRANCISCO, N. M. M. RODRIGUES, E. A. B. DA SILVA, M. B. DE CARVALHO, S. M. M. DE FARIA, V. M. M. DA SILVA, M. J. C. S. REIS "Multiscale Recurrent Pattern Image Coding With a Flexible Partition Scheme", *IEEE International Conference on Image Processing*, pp. 141–144, out. 2008.

[15] TRUCCO, E. & VERRI, A. *Introductory Techniques for 3-D Computer Vision*. Prentice Hall, 2002.

[16] FAUGERAS, O. *Three-dimensional computer vision: a geometric viewpoint*. 1 ed. London, The MIT Press, 1993.

[17] S. ZINGER, L. DO, P. H. N. DE WITH "Free-viewpoint depth image based rendering", *Journal of Visual Communication and Image Representation*, v. 21, n. 5, pp. 533–541, jul. 2010.

[18] WOO, W. *Rate Distortion Based Dependent Coding for Stereo Images and Video: Disparity Estimation and Dependent Bit Alloocation*. Phd. thesis, University of Southern California, dec. 1998.

[19] T. KANADE, M. OKUTOMI "A Stereo Matching Algorithm with an Adaptive Window: Theory and Experiment", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, v. 16, n. 9, pp. 920–932, sep. 1994.

[20] S. SENBEL, H. ABDEL-WAHAB "Scalable and Robust Image Compression using Quadtrees", *Signal Processing: Image Communication*, v. 14, pp. 425–442, 1999.

[21] C. J. TSAI, A. K. KATSAGGELOS "Dense Disparity Estimation with a Divide-and-Conquer Disparity Space Image Technique", *IEEE Transactions on Multimedia*, v. 1, n. 1, pp. 18–29, mar. 1999.

[22] M. G. STRINTZIS, S. MALASSIOTIS "Object-Based Coding of Stereoscopic and 3D Image Sequences", *IEEE Signal Processing Magazine*, pp. 14–28, maio 1999.

[23] MORVAN, Y. *Acquisition, Compression and Rendering of Depth and Texture for Multi-View Video*. Ph.d. Thesis, Technische Universiteit Eindhoven, Eindhoven, Nederland, 2009.

[24] "ISO/IEC JTC1/SC29/WG11, Report on Experimental Framework for 3D Video Coding, Doc. N11631". Guangzhou, China, October 2010.

[25] D. B. GRAZIOSI, N. M. M. RODRIGUES, C. L. PAGLIARI, E. A. B. DA SILVA, M. B. DE CARVALHO "Compressing Depth Maps using Multiscale Recurrent Pattern Image Coding", *Electronics Letters*, v. 46, n. 5, pp. 340–341, mar. 2010.

[26] TAUBMAN, D. S., MARCELIN, M. W. *JPEG2000: Image Compression Fundamentals, Standards and Practice*. Kluwer Academic Publishers, 2001.

[27] "Microsoft Research 3D Video. Test images set by Interactive Visual Media Group". Disponível em http://research.microsoft.com/en-us/um/people/sbkang/3dvideodownload. Acessado em agosto de 2010.

[28] "ISO/IEC JTC1/SC29/WG11, "HHI Test Material for 3D Video," M15413". FhG-HHI 3DV data sequence by Heinrich Hertz Institute, April 2008.

[29] "Nagoya University. FTV test sequences, Tanimoto Laboratory". Disponível em http://www.tanimoto.nuee.nagoya-u.ac.jp/MPEG-FTVProject.html. Acessado em agosto de 2010.

[30] GRAZIOSI, D. B. *Contribuicões à compressão de imagens com e sem perdas utilizando recorrência de padrões multiescalas*. Tese de D.Sc., COPPE/UFRJ, Rio de Janeiro, RJ, Brasil, 2011.

Apêndice A

Pseudo-Código do MMP

A seguir apresentaremos o pseudo-código do algoritmo MMP, elaborado em [30]. O procedimento codifica um bloco de entrada $X(i)$, com o custo J^l numa árvore binária $\mathcal{T}$.

Procedimento 1 PRINCIPAL

Inicializar o dicionário $\mathcal{D}^{1\times1}$ com todos os valores $[-255, 255]$ e os outros dicionários com parâmetros uniformes.

for $i = 0$ até o último bloco de entrada **do**
$(J^l, \mathcal{T}, m) = otimizar(X(i))$
$\hat{X}(i) = analizar(\mathcal{T})$
$\mathcal{D} \leftarrow \hat{X}(i)$
end for

Procedimento 2 $(J^{m,n}, \mathcal{T}, modo) = otimizar(X^{m,n})$

$\{P_1^{m,n}, \ldots, P_M^{m,n}\} = predição(X^{m,n})$
for $i = 0$ to M **do**
    $R^{m,n} = X^{m,n} - P_i^{m,n}$
    $(J^{m,n}, \mathcal{T}) = otimizar\_resíduo(R^{m,n})$
    $J = J^{m,n} + \lambda taxa(i)$
    if $J < minJ$ **then**
        $melhor\_modo = i$
        $minJ = J$
    end if
end for
if $X^{m,n}$ pode ser dividido verticalmente **then**
    $X^l = [X_{esquerdo} : X_{direito}]$
    $(J_{esquerdo}, \mathcal{T}_{esquerdo}, mode_{esquerdo}) = otimizar(X_{esquerdo})$
    $(J_{direito}, \mathcal{T}_{direito}, mode_{direito}) = otimizar(X_{direito})$
    $J_{hor} = J_{esquerdo} + J_{direito} + \lambda taxa(Flag(2))$
    if $J_{hor} < minJ$ **then**
        $minJ = J_{hor}$
        $\mathcal{T} = [\mathcal{T}_{left} : \mathcal{T}_{right}]$ é o nó da árvore
    end if
end if
if $X^{m,n}$ pode ser dividido horizontalmente **then**
    $X^{m,n} = [X_{acima} : X_{abaixo}]$
    $(J_{acima}, \mathcal{T}_{acima}, mode_{acima}) = otimizar(X_{acima})$
    $(J_{abaixo}, \mathcal{T}_{abaixo}, mode_{abaixo}) = otimizar(X_{abaixo})$
    $J_{vert} = J_{up} + J_{abaixo} + \lambda taxa(Flag(3))$
    if $J_{vert} < minJ$ **then**
        $minJ = J_{vert}$
        $\mathcal{T} = [\mathcal{T}_{acima} : \mathcal{T}_{abaixo}]$ é o nó da árvore
    end if
end if
return $(minJ, \mathcal{T}, melhor\_modo)$

Procedimento 3 $(J, \mathcal{T}) = otimizar\_resíduo(R^{m,n})$

if $m, n = 1 \times 1$ **then**
  $\mathcal{T} \leftarrow S$ folha da árvore
  Onde S é a palavra código de $\mathcal{D}^{1,1}$ que minimiza $minJ = ||R^{1,1} - S||^2 + \lambda taxa(S))$
  return $(minJ, \mathcal{T}^{1,1})$
end if
for $i = 0$ to $|\mathcal{D}^{m,n}|$ **do**
  $J(i) = ||R^{m,n} - S^{m,n}(i)||^2 + \lambda taxa(S^{m,n}(i)))$
  if $minJ < J(i)$ **then**
    $\mathcal{T} \leftarrow S^{m,n}(i)$ é a folha da árvore
    $minJ = J^l(i) + \lambda taxa(Flag(1))$
  end if
end for
if $R^{m,n}$ pode ser dividido verticalmente **then**
  $R^{m,n} = [R_{esquerdo} : R_{direito}]$
  $(J_{esquerdo}, \mathcal{T}_{esquerdo}) = otimizar\_residuo(R_{esquerdo})$
  $(J_{direito}, \mathcal{T}_{direito}) = otimizar\_residuo(R_{direito})$
  $J_{vert} = J_{esquerdo} + J_{direito} + \lambda rate(Flag(4))$
  if $J_{vert} < minJ$ **then**
    $minJ = J_{vert}$
    $\mathcal{T} \leftarrow [\mathcal{T}_{esquerdo} : \mathcal{T}_{direito}]$ é o nó da árvore
  end if
end if
if $R^{m,n}$ pode ser dividido horizontalmente **then**
  $R^{m,n} = [R_{acima} : R_{abaixo}]$
  $(J_{acima}, \mathcal{T}_{acima}) = otimizar\_residuo(R_{acima})$
  $(J_{abaixo}, \mathcal{T}_{abaixo}) = otimizar\_residuo(R_{abaixo})$
  $J_{hor} = J_{acima} + J_{abaixo} + \lambda rate(Flag(5))$
  if $J_{hor} < minJ$ **then**
    $minJ = J_{hor}$
    $\mathcal{T} \leftarrow [\mathcal{T}_{acima} : \mathcal{T}_{abaixo}]$ é o nó da árvore
  end if
end if
return $(minJ, \mathcal{T})$

Apêndice B

Imagens originais utilizadas

A seguir, serão mostradas as imagens originais utilizadas nesta dissertação.

- As sequências *Ballet* e *Breakdancers* [27] foram feitas com o uso de 7 (sete) câmeras, sendo que cada uma delas obtém 100 *frames* da cena a uma taxa de 15 fps. A resolução da câmera é 1024x768.
- A sequência *Book Arrival* [28] utiliza 16 (dezesseis) câmeras, cada uma obtendo 100 *frames* com tempo total de 6 segundos de duração da sequência. A resolução da câmera é 1024x768.
- As sequências *Champagne Tower* e *Pantomime* [29] utilizam 80 (oitenta) câmeras, e cada uma obtém 300 *frames* com taxa de 30 fps. A resolução da câmera é 1280x960.

B.1 Imagens de textura

Figura B.1: Ballet, câmera 03, *frame* 0. Fonte: [27].

Figura B.2: Ballet, câmera 05, *frame* 0. Fonte: [27].

Figura B.3: Breakdancers, câmera 03, *frame* 0. Fonte: [27].

Figura B.4: Breakdancers, câmera 05, *frame* 0. Fonte: [27].

Figura B.5: Book arrival, câmera 08, *frame* 0. Fonte: [28].

Figura B.6: Book arrival, câmera 10, *frame* 0. Fonte: [28].

Figura B.7: Champagne tower, câmera 37, *frame* 0. Fonte: [29].

Figura B.8: Champagne tower, câmera 39, *frame* 0. Fonte: [29].

Figura B.9: Pantomime, câmera 37, *frame* 0. Fonte: [29].

Figura B.10: Pantomime, câmera 39, *frame* 0. Fonte: [29].

B.2 Imagens de profundidade

Figura B.11: Ballet, câmera 03, *frame* 0. Fonte: [27].

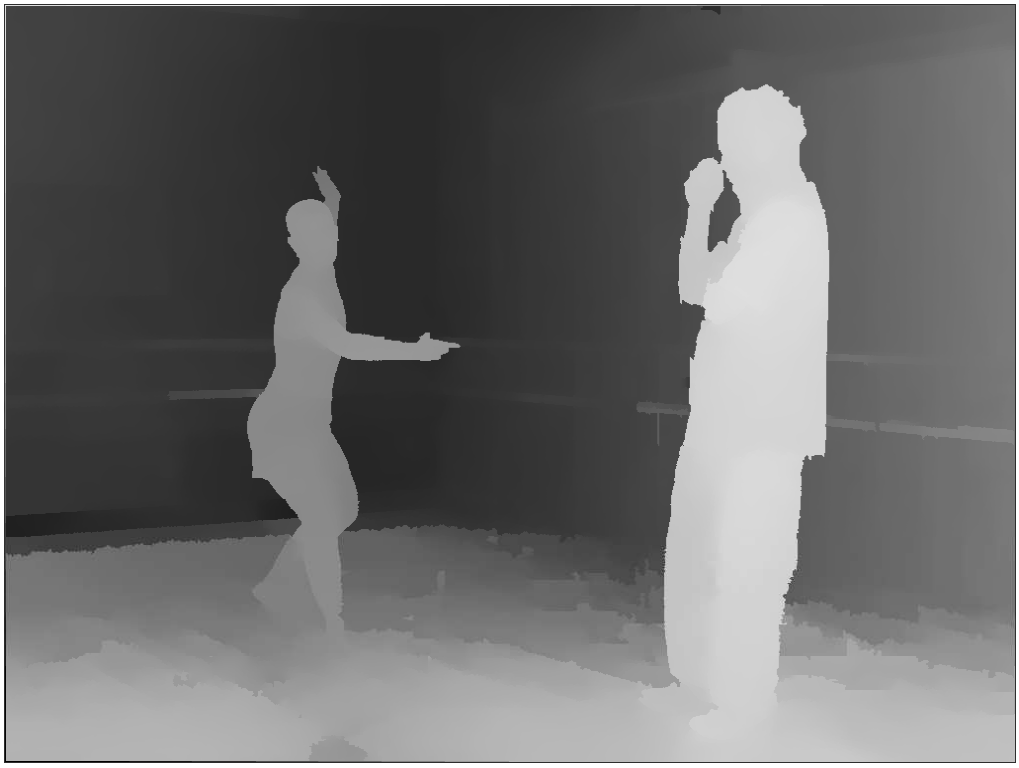


Figura B.12: Ballet, câmera 05, *frame* 0. Fonte: [27].

Figura B.13: Breakdancers, câmera 03, *frame* 0. Fonte: [27].

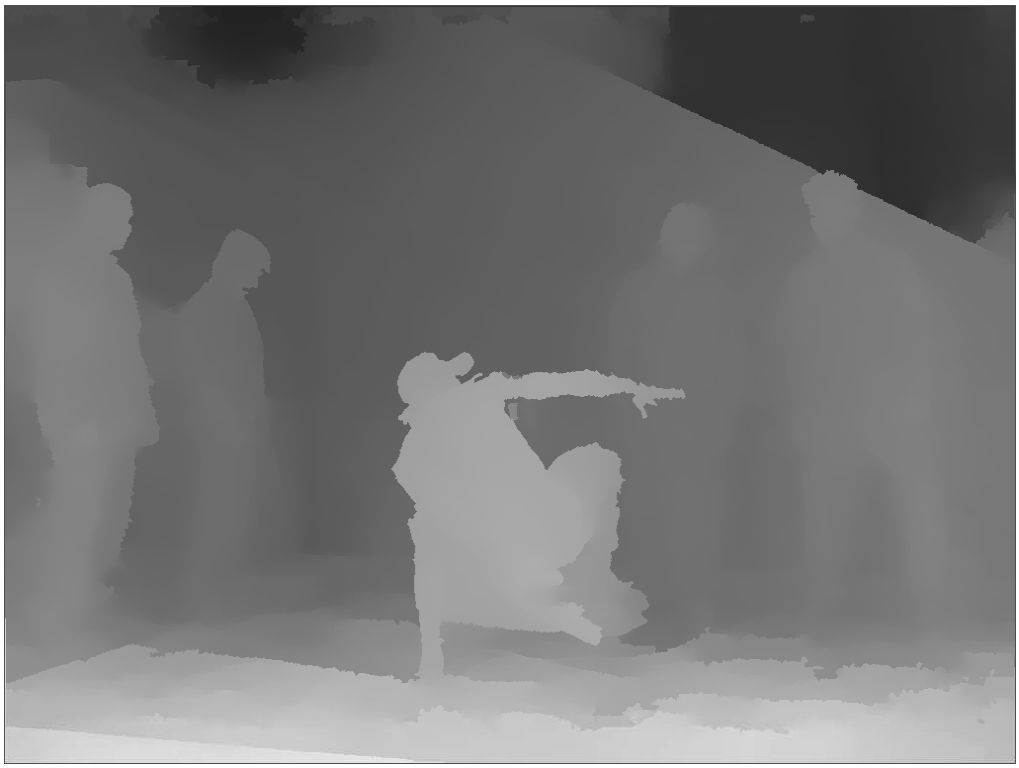


Figura B.14: Breakdancers, câmera 05, *frame* 0. Fonte: [27].

Figura B.15: Book arrival, câmera 08, *frame* 0. Fonte: [28].

Figura B.16: Book arrival, câmera 10, *frame* 0. Fonte: [28].

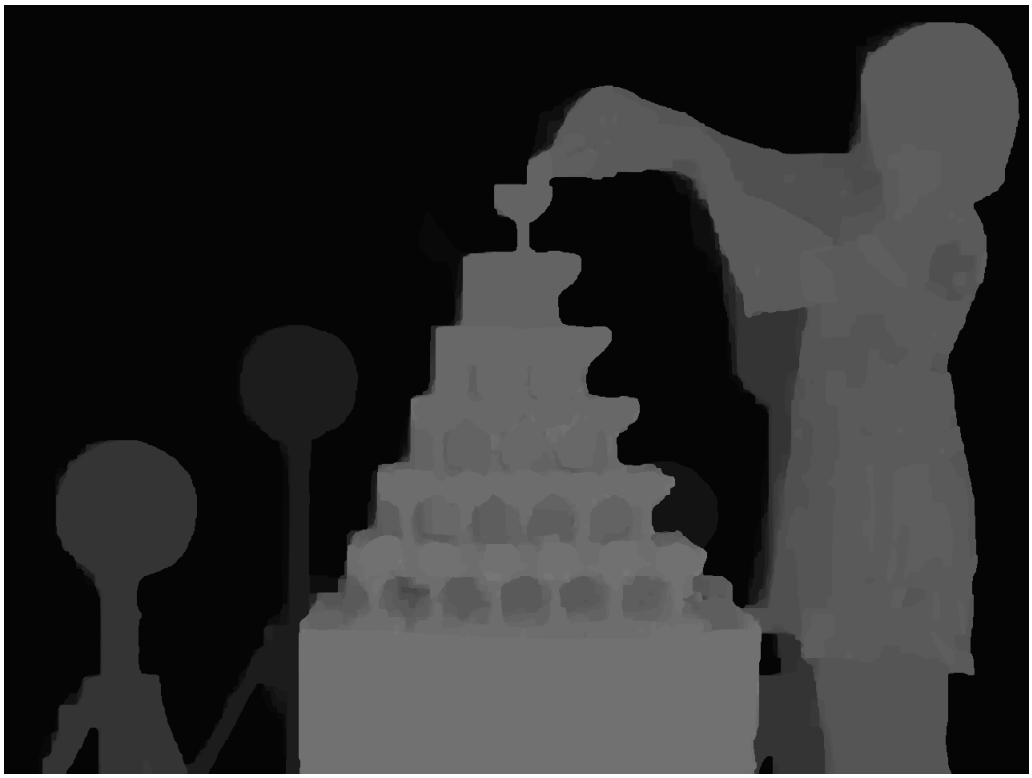


Figura B.17: Champagne tower, câmera 37, *frame* 0. Fonte: [29].

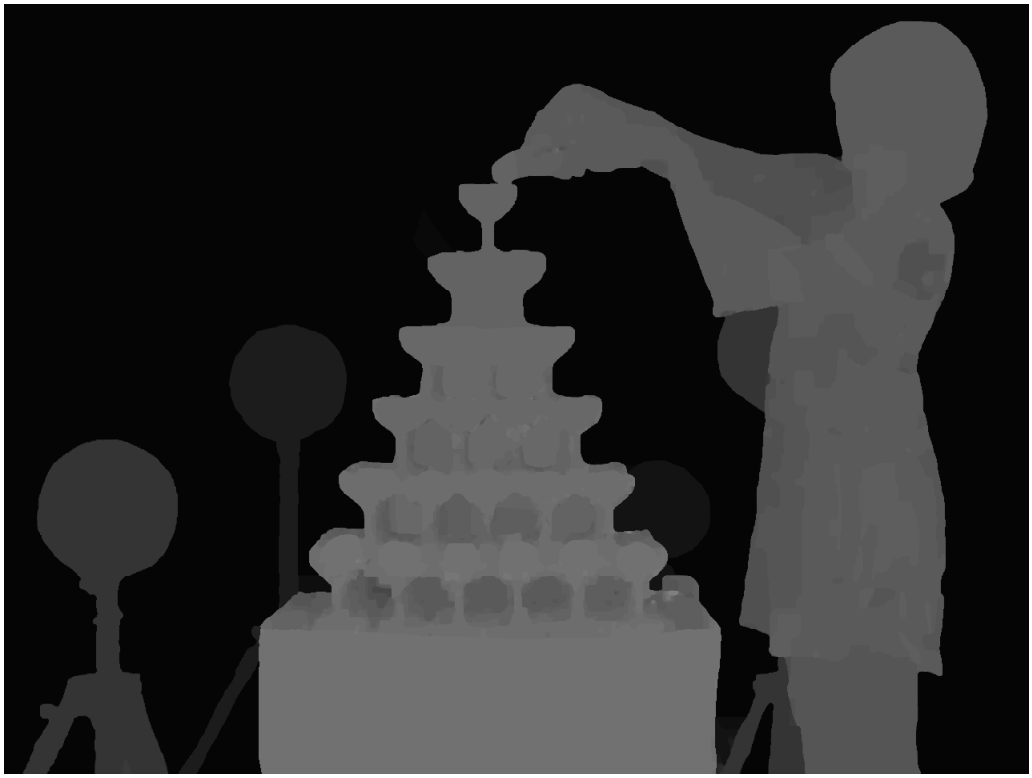


Figura B.18: Champagne tower, câmera 39, *frame* 0. Fonte: [29].

Figura B.19: Pantomime, câmera 37, *frame* 0. Fonte: [29].

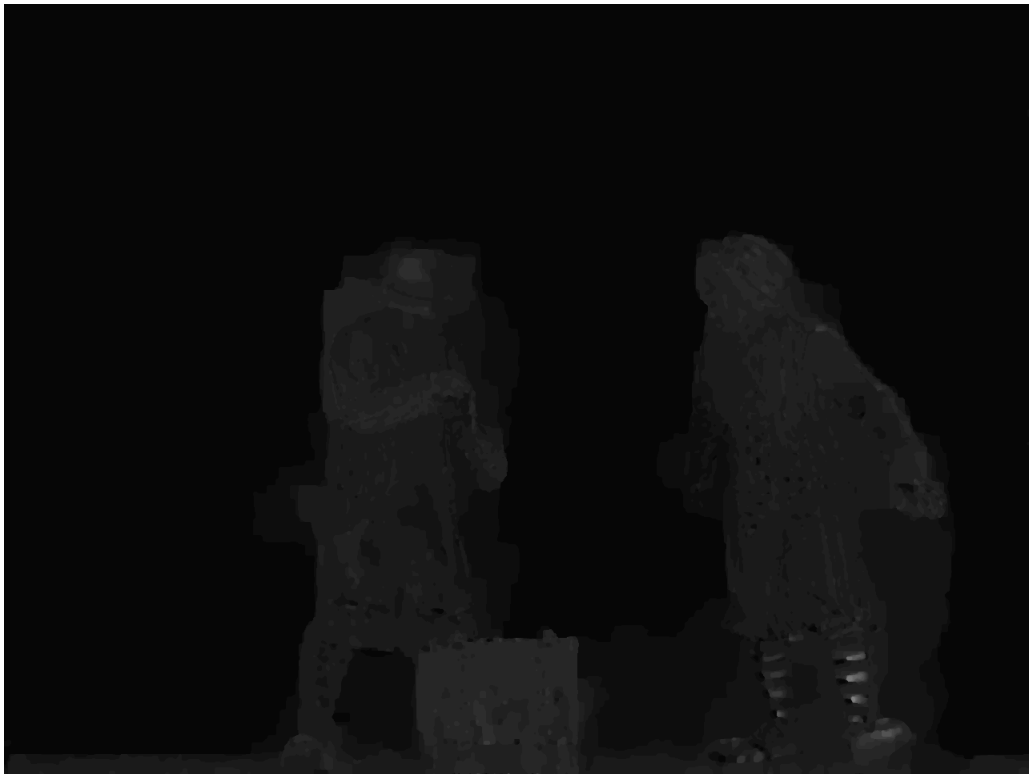


Figura B.20: Pantomime, câmera 39, *frame* 0. Fonte: [29].