CODIFICAÇÃO DE IMAGENS ESTÉREO USANDO RECORRÊNCIA DE PADRÕES

Maria Heveline Vieira Duarte

TESE SUBMETIDA AO CORPO DOCENTE DA COORDENAÇÃO DOS PROGRAMAS DE PÓS-GRADUAÇÃO DE ENGENHARIA DA UNIVERSIDADE FEDERAL DO RIO DE JANEIRO COMO PARTE DOS REQUISITOS NECESSÁRIOS PARA A OBTENÇÃO DO GRAU DE DOUTOR EM CIÊNCIAS EM ENGENHARIA ELÉTRICA.

Aprovada por:

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

Prof. Carla Liberal Pagliari, Ph.D.

Prof. Murilo Bresciani de Carvalho, D.Sc.

Prof. Gelson Vieira Mendonça, Ph.D.

Prof. Sergio Lima Netto, Ph.D.

Prof. Abraham Alcaim, Ph.D.

Prof. Weiler Alves Finamore, Ph.D.

RIO DE JANEIRO, RJ - BRASIL

AGOSTO DE 2002

DUARTE, MARIA HEVELINE VIEIRA

Codificação de Imagens Estéreo Usando Recorrência de Padrões [Rio de Janeiro] 2002

XIX, 188 pp., 29,7 cm (COPPE/UFRJ, D.Sc., Engenharia Elétrica, 2002)

Tese - Universidade Federal do Rio de Janeiro, COPPE

1.Codificação de Imagens 2.Imagens Estéreo 3.Recorrência de Padrões

I.COPPE/UFRJ II.Título (série)

Agradecimentos

Dedico este trabalho ao Marco Antônio. Meu amor, meu companheiro de todas as horas, meu maior incentivador.

Dedico à memória dos meus pais, Sebastião e Neusa, meus modelos de amor, força, trabalho e vida.

Agradeço ao meu orientador e amigo Eduardo, por ter acreditado em mim desde o início, por ter me incentivado nos momentos mais difíceis, e por ter me feito desistir de desistir.

Agradeço ao Murilo, o "pai" do MMP, pela ajuda, pelas sugestões, pela amizade.

Obrigada Prof. Gelson, por ter aceitado me receber na Engenharia Elétrica como sua aluna junto com o Eduardo.

Agradeço a Carla Pagliari pelas sugestões sempre relevantes ao longo da tese.

Agradeço ao Professor Diniz pelas aulas no curso de Filtros Adaptativos, pelo seu entusiamo contagiante, que resgatou meu orgulho de ser engenheira e me fez ter vontade de retornar à Engenharia Elétrica. Obrigada por ter me recebido de braços abertos, por manter o grupo de Processamento de Sinais unido, por nos proporcionar este ambiente de trabalho tão agradável no LPS, pelas brincadeiras, e principalmente pela amizade.

Agradeço ao Willian, pelo incentivo e ajuda no início do doutorado, por estudar comigo Filtros Digitais, Multitaxas, e tantas outras disciplinas. Obrigada por organizar os churrascos, e obrigada pela amizade.

Agradeço aos colegas de Morfologia Matemática pela difícil tarefa de estudar, entender e (isto é incrível) dar aulas desta disciplina: Emanuel, Cabelo, Lisandro, Ciancio, Esquef, Caetano, Fábio, Felipe. Que dureza, heim ?

Obrigada Décio, por me ajudar nos exercícios de Visão por Computador, pela amizade de verdade, por estar disponível quando parecia que tudo ia desabar. Obrigada Ana Paula por ter me amparado num momento tão delicado. Admiro muito você, seu alto astral. Gostaria de ser assim quando eu crescer.

Agradeço ao Lisandro, pela companhia no café, pelas conversas inteligentes, pelos assuntos sempre interessantes, pelo carinho e atenção.

Agradeço à minha querida amiga Bené, pelo apoio, pela amizade constante. Descobri a grande amiga que você é, com quem pude contar em todas as horas. Desculpe o trabalho que te dei, você foi uma procuradora e tanto.

Agradeço ao CEFET-CE e ao CNPq pelo suporte financeiro.

Agradeço às minhas irmãs tão queridas. Lúcia, Margarida, Paezinha e Vilani. Obrigada por cuidarem da sua irmãzinha caçula tão trabalhosa. Aos meus sobrinhos queridos. Ao Lima Filho e as minhas "irmãzinhas" caçulas Gilmara e Lia, vocês foram grandes incentivos pra eu não desistir.

Obrigada Lara, por tudo. Pela atenção, pela dedicação, por trazer o Miguel pra me dar sorte nas defesas.

Obrigada L.W.

Obrigada Esquef, por você ter me recitado Milton Nascimento: *"Nada a temer senão o correr da luta. Nada a fazer senão esquecer o medo. Abrir o peito à força numa procura. Fugir às armadilhas da mata escura. Longe se vai sonhando demais..."*

Obrigada Stefan, seu maluco, sinto sua falta.

Obrigada Fábio, pela atenção, por abdicar do seu tempo para discutir o MMP comigo, por me ajudar a decifrar o que o Eduardo pedia. Pelo apoio em todos os aspectos.

Obrigada à minha psicóloga Maria Lúcia por me ajudar a superar algumas barreiras, a entender certos comportamentos, a desfazer nós, a ser mais flexível com os outros e menos dura comigo mesma. Obrigada à minha outra psicóloga Maria Cecília, que me ensinou a confiar mais na minha força.

Obrigada Caetano, pelas dicas de processamento de imagens, no início, quando trabalhar com Linux e linguagem C para imagens ainda eram tarefas tão árduas. Obrigada por estar sempre disposto a nos ajudar.

Agradeço aos colegas do LPS2, pelo carinho, pelo companheirismo, pelas brincadeiras do dia a dia: Fábio, Lisandro, Caetano, Ailton, Cassio Barbosa, Felipe.

Agradeço às secretárias do LPS: Vanessa e Michelle. Vanessa, obrigada pela iniciativa do "lanchinho" das 16:00h.

Obrigada Calôba pela valiosa listinha de queijos e vinhos.

Obrigada Maria, pelo café com amor, e pelo sorriso de todos os dias.

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

CODIFICAÇÃO DE IMAGENS ESTÉREO USANDO RECORRÊNCIA DE PADRÕES

Maria Heveline Vieira Duarte

Agosto/2002

Orientadores: Eduardo Antônio Barros da Silva
Carla Liberal Pagliari

Programa: Engenharia Elétrica

O algoritmo *Multidimensional Multiscale Parser* – MMP, utilizado como ponto de partida para esta tese, é uma nova classe de algoritmos de compressão, que se baseia em casamento aproximado de padrões recorrentes com escalas, ou seja, o algoritmo tenta representar um bloco de dados de entrada usando versões dilatadas ou contraídas de blocos anteriores. Esta tese tem como objetivo utilizar o MMP explorando eficientemente as características das imagens estéreo, de forma a se obter uma melhor compressão do par estéreo. A imagem de entrada é segmentada em blocos de tamanho variável. Cada segmento é codificado fazendo uso de contrações, expansões e deslocamentos de elementos de um dicionário. A segmentação obedece a um critério de taxa e distorção e o dicionário é atualizado com a concatenação de elementos previamente codificados. A novidade deste trabalho é a não transmissão do mapa de disparidades no processo de codificação. Isto é, os transtornos envolvidos no seu cálculo preciso, as fases de pré e pós processamento, a geração de imagens de erro, bem como sua codificação e transmissão não são necessários. Em resumo, o método proposto evita todo o processo de estimação de disparidades e ainda apresenta resultados de alta qualidade na imagem reconstruída, compararáveis com os resultados dos métodos de codificação de imagens estéreo que transmitem mapa de disparidades.

Abstract of Thesis presented to COPPE/UFRJ as a partial fulfillment of the requirements for the degree of Doctor of Science (D.Sc.)

STEREO IMAGE CODING USING MULTISCALE RECURRENT PATTERNS

Maria Heveline Vieira Duarte

August/2002

Advisors: Eduardo Antônio Barros da Silva
Carla Liberal Pagliari

Department: Electrical Engineering

The *Multidimensional Multiscale Parser* – MMP algorithm, used as start point for this thesis, is a new class of compression algoritms, using multiscale recurrent patterns. The method represents a present image block using dilated and contracted versions of blocks already processed. The objective of this thesis is, using the MMP algorithm, exploit efficiently the stereo characteristics of the stereo pair, for obtaining a better rate×distortion trade-off. The input image is segmented in variable-sized blocks. Each segment is coded by making use of contractions, expansions and displacements of elements in a dictionary. The segmentation is performed according to a rate×distortion criterion and the dictionary is updated with the concatenation of previously coded elements. The novelty of this work is that the disparity map is not transmitted. That is, the burden of its calculation, the pre- and post-processing phases, the generation of the error images, as well as their coding and transmission, are not necessary. In brief, the proposed method cuts off the load of the whole disparity estimation process yet presenting high-quality results at the decoder end, compared to another coding methods for stereo images that transmit the disparity map.

Sumário

1 Introdução **1**

1.1 Percepção 3D e geometria estéreo 1

1.1.1 Determinação da estrutura 3D 3

1.2 Codificação de imagens estéreo 4

1.2.1 Obtenção do mapa de disparidades 6

1.3 *Multidimensional Multiscale Parser* – MMP 8

1.4 Objetivo desta tese 9

1.5 Descrição dos capítulos desta tese 9

2 *Multidimensional Multiscale Parser* - MMP **14**

2.1 Introdução 14

2.2 Descrição do algoritmo MMP 15

2.2.1 Exemplo de aplicação do MMP para representação de um sinal unidimensional 15

2.2.2 Descrição do MMP para representação de um sinal bidimensional 34

2.2.3 A atualização do dicionário 38

2.2.4 O decodificador 41

2.3 Conclusão 41

3 Avaliação do desempenho do MMP aplicado a imagens estéreo **42**

3.1 Introdução 42

3.2 Descrição suscinta do algoritmo *Optimal Dependent Quantisation* – ODQ 43

3.3 Características das imagens estéreo utilizadas 52

3.4 Teste 1: MMP versus ODQ aplicados à imagem esquerda e ao quadro DCD 56
3.5 Teste 2: MMP versus ODQ, aplicados diretamente às imagens esquerda e direita, sem cálculo de quadro DCD 61
3.6 Teste 3: MMP aplicado às imagens esquerda e direita, versus ODQ aplicado à imagem esquerda e ao quadro DCD 63
3.7 Conclusão 66

4 MMP estéreo **69**
4.1 Introdução 69
4.2 MMP aplicado a linhas de blocos (ROBs) correspondentes 70
4.3 Versão 1 do MMP estéreo 72
4.4 Versão 2 do MMP estéreo 77
4.5 Versão 3 do MMP estéreo 80
4.6 Versão 4 do MMP estéreo 88
4.7 Versão 5 do MMP estéreo 88
4.8 Versão 6 do MMP estéreo 90
4.9 Comparação entre MMP versão 6 e ODQ 97
4.10 Comparação entre MMP versão 6 e ODQ melhorado 99
4.11 Conclusão 101

5 MMP estéreo com cálculo do mapa de disparidades **102**
5.1 Introdução 102
5.2 Como utilizar (sem transmitir) a informação dos vetores de disparidade 103
5.3 Versão 7 do MMP estéreo 108
5.4 Versão 8 do MMP estéreo 115
5.5 Versão 9 do MMP estéreo 119
5.6 Versão 10 do MMP estéreo 123
5.7 Versão 11 do MMP estéreo 126
5.8 Versão 12 do MMP estéreo 130
5.9 Comparação entre as versões 6 e 12 do MMP estéreo 134
5.10 Comparação entre as versões 1 e 12 do MMP estéreo 136
5.11 Comparação entre MMP versão 12 e ODQ 142

5.12 Comparação entre MMP versão 12 e ODQ melhorado (HQBM) . . . 144
5.13 Imagens reconstruídas . . . 146
5.14 MMP estéreo versão 12: Tabelas com valores de taxa e PSNR . . . 148
5.15 Versão 12 do MMP estéreo: itens implementados . . . 149
5.16 Conclusão . . . 150

6 Conclusões **152**

Referências Bibliográficas **155**

A Descrição do algoritmo *Optimal Dependent Quantisation* – ODQ 159
A.1 Introdução . . . 159
A.2 Sistemas de codificação preditivos . . . 159
A.3 Formulação do problema: codificação dependente . . . 160
A.3.1 Estrutura loop fechado . . . 161
A.3.2 Estrutura loop aberto . . . 162
A.4 Quantização dependente ótima por blocos . . . 162
A.5 Solução usando o algoritmo de Viterbi . . . 164
A.5.1 Definições . . . 164

B Imagens estéreo originais **166**

C Descrição das versões do MMP estéreo **170**

D Descrição dos testes realizados **176**
D.1 Testes realizados no Cap. 3 . . . 176
D.2 Testes realizados no Cap. 4 . . . 176
D.3 Testes realizados no Cap. 5 . . . 178

E Exemplo de programação dinâmica **180**
E.1 O Problema da viagem . . . 180
E.2 Os elementos de programação dinâmica. O algoritmo . . . 184
E.2.1 Convenção . . . 186
E.2.2 Observações . . . 186

Lista de Figuras

1.1 Representação geométrica do sistema de câmeras estéreo [1]. 3

2.1 Exemplo de um sinal de entrada unidimensional. 16
2.2 Vetores do dicionário inicial para o sinal unidimensional u exemplificado 16
2.3 Divisões possíveis no MMP para um vetor do dicionário inicial, criado a partir da primeira amostra do sinal unidimensional u exemplificado. 17
2.4 Divisões possíveis no MMP para um vetor do dicionário inicial, criado a partir da quinta amostra do sinal unidimensional u exemplificado. . 18
2.5 Divisão do sinal unidimensional u em duas partes de tamanhos iguais u_E e u_D. 19
2.6 Custos associados à representação de um sinal por um determinado vetor do dicionário num nível mais alto (custo J), ou por dois vetores do dicionário num nível mais baixo (custos J_E e J_D). 20
2.7 Divisão dos sinais u_E e u_D em duas partes de tamanhos iguais. 21
2.8 Divisão dos sinais u_E1, u_E2, u_D1 e u_D2 em duas partes de tamanhos iguais. 22
2.9 Árvore binária que irá representar um sinal de entrada, e suas possíveis divisões. Os nós estão denominados por letras para facilitar o entendimento da decodificação da seqüência de bits. 23
2.10 Dicionário no início da decodificação da seqüência de bits. 24
2.11 Seqüência de decodificação da seqüência de bits exemplificado. Ao lado de cada nó temos os *flags*. *Flag=1* indica divisão. *Flag=0* indica nó terminal. Para cada nó terminal temos um índice, indicando o vetor do dicionário que representa aquela parte do sinal. 25

2.12 Seqüência de decodificação da seqüência de bits exemplificada. Ao lado dos nós terminais mostramos os vetores correspondentes do dicionário. 26
2.13 Concatenação de dois nós terminais (F e G), originando um novo vetor (que corresponde ao nó E) a ser inserido no dicionário, num nível acima do nível dos nós F e G. 26
2.14 Dicionário após a inserção do vetor correspondente ao nó E, resultante da concatenação dos nós F e G. Este novo vetor foi inserido na posição 16, no nível 1 do dicionário. 27
2.15 Expansões do vetor correspondente ao nó E, a serem inseridas nos níveis superiores do dicionário. 28
2.16 Dicionário após a inserção das expansões do vetor correspondente ao nó E. A primeira expansão resultou no vetor 23 inserido no nível 2 do dicionário, e a outra expansão resultou no vetor 11 inserido no nível 3 do dicionário. 29
2.17 Concatenação de dois nós (D e E) para inserir um novo vetor no nível 2 do dicionário. 29
2.18 Dicionário após a inserção do vetor correspondente ao nó B, resultante da concatenação dos vetores correspondentes aos nós D e E. Este vetor foi inserido na posição 24, no nível 2 do dicionário. 30
2.19 Expansão e contração do vetor correspondente ao nó B, resultante da concatenação dos vetores correspondentes aos nós D e E, a serem inseridas no dicionário. 30
2.20 Dicionário após as inserções da contração (vetor 17, nível 1) e da expansão (vetor 12, nível 3) do vetor 24, do nível 2, correspondente ao nó B. 31
2.21 Concatenação dos vetores correspondentes aos nós B e C, gerando um novo vetor que corresponde ao nó A. Este novo vetor será inserido no nível 3 do dicionário. 31
2.22 Dicionário após a inserção do vetor correspondente ao nó A, (posição 13, nível 3) resultante da concatenação dos vetores correspondentes aos nós B e C. 32

2.23 Dicionário após a inserção do vetor resultante da contração do vetor 13, nível 3 (correspondente ao nó A), inserido na posição 25, nível 2). 33
2.24 Diagrama de uma árvore binária que representa um bloco 8x8 e suas possíveis divisões. A divisão dos blocos é vertical. 35
2.25 Elemento E[6][0] (nível 6, índice 0) do dicionário. 36
2.26 Elemento E[6][1] (nível 6, índice 1) do dicionário. 36
2.27 Diagrama representando um nó de uma árvore binária, e suas possíveis divisões (nós filhos). Os custos associados ao nó pai, e aos nós filhos esquerdo e direito são, respectivamente, J_P, J_E e J_D. 37
2.28 Árvore ótima após o processo de segmentação (podagem da árvore). Esta árvore é a que melhor representa um determinado bloco da imagem. 39
2.29 Atualização do dicionário. 39
2.30 Exemplo de contrações e expansões de um bloco da imagem. 40

3.1 Diagrama em blocos do algoritmo ODQ. V são os vetores de disparidade por bloco, $I_E(Q1)$ é a imagem esquerda quantizada com o conjunto de quantizadores Q_1, e DCD(Q2) é a imagem DCD quantizada com o conjunto de quantizadores Q_2. 45
3.2 Diagrama em blocos da quantização de uma ROB da imagem esquerda com o algoritmo ODQ. Obtenção da taxa R_1 e distorção D_1. . 46
3.3 Preparação para achar dependência entre os blocos da imagem esquerda e direita. 47
3.4 Dependência entre os blocos das imagens esquerda e direita. 47
3.5 Diagrama em blocos da quantização de uma ROB da imagem direita com o algoritmo ODQ. Obtenção da taxa R_2. 48
3.6 Diagrama em blocos da quantização de uma ROB da imagem direita com o algoritmo ODQ. Obtenção da distorção D_2. 49
3.7 Otimização dos quantizadores do ODQ utilizando algoritmo de Viterbi. 50
3.8 Diagrama em blocos da reconstrução da imagem direita, utilizando o algoritmo ODQ, a partir da imagem esquerda quantizada por Q_1, dos vetores de disparidade V e do frame DCD quantizado com Q_2. 51
3.9 Pares de imagens stereo: (a) CORRIDOR (256x256), (b) FRUIT (256x256), (c) MAN (384x384). 53

3.10 Quadro DCD das imagens estéreo: (a) CORRIDOR, (b) FRUIT, (c) MAN. 55
3.11 Diagrama em blocos do teste 1: Comparação entre os algoritmos ODQ e MMP aplicados à imagem esquerda e ao quadro DCD do par estéreo. 56
3.12 Resultados do teste 1 para a imagem estéreo CORRIDOR: (a) Imagem esquerda, (b) Quadro DCD, (c) média entre a imagem esquerda e o quadro DCD. 58
3.13 Resultados do teste 1 para a imagem estéreo FRUIT: (a) imagem esquerda, (b) quadro DCD, (c) média entre imagem esquerda e DCD. 59
3.14 Resultados do teste 1 para a imagem estéreo MAN: (a) Imagem esquerda , (b) quadro DCD, (c) média entre imagem esquerda e DCD. . 60
3.15 Diagrama em blocos do teste 2: Comparação entre os algoritmos ODQ e MMP aplicados diretamente às imagens esquerda e direita do par estéreo. O quadro DCD não é utilizado. 61
3.16 Resultados do teste 2 para as imagens estéreo (a) CORRIDOR, (b) FRUIT, e (c) MAN. 62
3.17 Diagrama em blocos do teste 3: Comparação entre o algoritmo ODQ aplicado à imagem esquerda e ao quadro DCD, com algoritmo MMP aplicado diretamente às imagens esquerda do par estéreo, sem uso do quadro DCD. 64
3.18 Resultados do teste 3 para as imagens estéreo (a) CORRIDOR, (b) FRUIT e (c) MAN. 65

4.1 Diagrama em blocos de um par estéreo (imagens esquerda e direita) com suas linhas de blocos (ROBs) correspondentes. 71
4.2 MMP aplicado às linhas de blocos esquerda e direita (MMP ROB a ROB), comparado com MMP aplicado à imagem esquerda completa e depois à imagem direita completa do par estéreo CORRIDOR. 72
4.3 Elementos deslocados a partir de dois blocos vizinhos pertencentes a uma ROB da imagem esquerda reconstruída. 72
4.4 Comparação entre o MMP estéreo (versão 1) e MMP ROB a ROB para a imagem estéreo CORRIDOR (média das imagens esquerda e direita. 74

4.5 Percentual de ocorrências dos elementos do dicionário por tipo, para cada nível, (a) para a imagem esquerda, e (b) para a imagem direita do par estéreo CORRIDOR. 78
4.6 Percentual de ocorrências dos elementos do dicionário por tipo, para cada nível, (a) para a imagem esquerda, e (b) para a imagem direita do par estéreo MAN. 79
4.7 Comparação entre as versões 2 e 1 do MMP estéreo, para a imagem CORRIDOR. (a) imagem esquerda, (b) imagem direita. 81
4.8 Dicionário original, com elementos comuns e deslocados da ROB. Dicionário original separado em dois: necessidade de um *flag*. 83
4.9 Comparação entre as versões 3 e 2 do MMP estéreo, para a imagem CORRIDOR. (a) imagem esquerda, (b) imagem direita. 86
4.10 Comparação entre as versões 3 e 1 do MMP estéreo, para a imagem CORRIDOR. (a) imagem esquerda, (b) imagem direita. 87
4.11 Comparação entre as versões 4 e 3 do MMP estéreo, para a imagem direita do par estéreo CORRIDOR. 89
4.12 Comparação entre as versões 5 e 3 do MMP estéreo, para a imagem direita do par estéreo CORRIDOR. 89
4.13 Comparação entre usar deslocamentos de 1/4 e 1/2 de pixel, para a imagem direita do par CORRIDOR. 90
4.14 Comparação entre as versões 6 e 1 do MMP estéreo. (a) imagem esquerda, e (b) imagem direita do par estéreo CORRIDOR. 92
4.15 Comparação entre as versões 6 e 1 do MMP estéreo. (a) imagem esquerda, e (b) imagem direita do par estéreo AQUA. 93
4.16 Comparação entre as versões 6 e 1 do MMP estéreo. (a) imagem esquerda, e (b) imagem direita do par estéreo MAN. 94
4.17 Comparação entre as versões 6 e 1 do MMP estéreo. (a) imagem esquerda, e (b) imagem direita do par estéreo SAXO. 95
4.18 Comparação entre as versões 6 e 1 do MMP estéreo. (a) imagem esquerda, e (b) imagem direita do par estéreo TRAIN. 96
4.19 Comparação entre MMP estéreo versão 6 e ODQ, para o par estéreo CORRIDOR. (a) imagem esquerda, (b) imagem direita. 97

4.20 Comparação entre MMP estéreo versão 6 e ODQ, para o par estéreo AQUA. (a) imagem esquerda, (b) imagem direita. 98
4.21 Comparação entre MMP estéreo versão 6 e HQBM, para o par estéreo (a) CORRIDOR e (b) AQUA. 100

5.1 Para um determinado pixel da imagem direita, procura-se numa região da imagem esquerda o pixel correspondente. 104
5.2 Reconstrução da imagem direita a partir da imagem esquerda e do mapa de disparidades. 105
5.3 Cálculo e transmissão do mapa de disparidades apenas para o primeiro par de ROBs correspondentes. 106
5.4 Como são gerados os elementos "deformados". 107
5.5 Cálculo do mapa de disparidades entre duas ROBs correspondentes, que será utilizado para estimar a próxima ROB direita. 107
5.6 Codificamos o primeiro par de ROBs usando um dicionário comum. Reconstruímos este par de ROBs e calculamos o mapa de disparidades, que será utilizado para codificar a próxima ROB direita. . . . 108
5.7 Uso do próprio mapa de disparidades calculado entre duas ROBs correspondentes, para gerar elementos "deformados" para codificar a próxima ROB direita. 110
5.8 Repetição da última linha do mapa de disparidades calculado entre as ROBs anteriores. 111
5.9 Média simples das colunas do mapa calculado entre as ROBs anteriores. 112
5.10 Média gradual, dando um peso maior aos valores mais próximos da vizinhança das ROBs e um peso menor aos valores mais distantes. . . 113
5.11 Média móvel (de dois pontos) das colunas das disparidades obtidas das ROBs anteriores. 114
5.12 Distribuição de ocorrências dos elementos deformados para as imagens (a) CORRIDOR e (b) MAN. 116
5.13 Número de elementos deformados do dicionário reduzido a um único elemento, aquele correspondente ao bloco da imagem direita. 117
5.14 Comparação entre as versões 8 e 7 do MMP estéreo. Resultados para a imagem direita dos pares estéreo (a) CORRIDOR e (b) MAN. . . . 118

5.15 Comparação entre as versões 9 e 8 do MMP estéreo. Resultados para a imagem direita dos pares estéreo (a) CORRIDOR e (b) MAN. . . . 120
5.16 Distribuição das ocorrências dos elementos deslocados, limitados pela janela de busca (25 a 128) da imagem MAN. 121
5.17 Uso da informação de disparidades para encontrar janela de deslocados.122
5.18 Distribuição das ocorrências dos elementos deslocados, que foram gerados baseados na informação de disparidade. Resultados para a imagem MAN. 123
5.19 Comparação entre as versões 10 e 9 do MMP estéreo. Resultados para a imagem direita dos pares estéreo (a) CORRIDOR, (b) MAN e (c) SAXO. 125
5.20 Comparação entre as versões 11 e 10 do MMP estéreo. (a) imagem esquerda, e (b) imagem direita do par estéreo CORRIDOR 127
5.21 Comparação entre as versões 11 e 10 do MMP estéreo. (a) imagem esquerda, e (b) imagem direita do par estéreo MAN. 128
5.22 Comparação entre as versões 11 e 10 do MMP estéreo. (a) imagem esquerda, e (b) imagem direita do par estéreo SAXO. 129
5.23 Como são formados os elementos rotacionados. 130
5.24 Como são formados os elementos simétricos. 131
5.25 Comparação entre as versões 12 e 11 do MMP estéreo. Resultados para a imagem direita dos pares estéreo (a) CORRIDOR e (b) MAN. 132
5.26 Comparação entre as versões 12 e 6 do MMP estéreo, para o par estéreo CORRIDOR. (a) imagem esquerda, (b) imagem direita. . . . 134
5.27 Comparação entre as versões 12 e 6 do MMP estéreo, para o par estéreo MAN. (a) imagem esquerda, (b) imagem direita. 135
5.28 Comparação entre as versões 12 e 1 do MMP estéreo, para o par estéreo CORRIDOR. (a) imagem esquerda, (b) imagem direita, (c) média entre as imagens esquerda e direita. 137
5.29 Comparação entre as versões 12 e 1 do MMP estéreo, para o par estéreo AQUA. (a) imagem esquerda, (b) imagem direita, (c) média entre as imagens esquerda e direita. 138

5.30 Comparação entre as versões 12 e 1 do MMP estéreo, para o par estéreo MAN. (a) imagem esquerda, (b) imagem direita, (c) média entre as imagens esquerda e direita. 139
5.31 Comparação entre as versões 12 e 1 do MMP estéreo, para o par estéreo SAXO. (a) imagem esquerda, (b) imagem direita, (c) média entre as imagens esquerda e direita. 140
5.32 Comparação entre as versões 12 e 1 do MMP estéreo, para o par estéreo TRAIN. (a) imagem esquerda, (b) imagem direita, (c) média entre as imagens esquerda e direita. 141
5.33 Comparação entre MMP estéreo versão 12 e ODQ, para o par estéreo CORRIDOR. (a) imagem esquerda, (b) imagem direita. 142
5.34 Comparação entre MMP estéreo versão 12 e ODQ, para o par estéreo AQUA. (a) imagem esquerda, (b) imagem direita. 143
5.35 Comparação entre MMP estéreo versão 12 e HQBM, para o par estéreo (a) CORRIDOR e (b) AQUA. 145
5.36 (a) CORRIDOR original (256x256), 131110 bytes (inclui cabeçalho) (b) CORRIDOR reconstruído ($\lambda = 32$), img esq.: taxa=0,701 PSNR=37,01. img dir.: taxa=0,246 PSNR=36,94. Número bytes imagem compactada: 7765. 146
5.37 (a) MAN original (384x384), 294950 bytes (inclui cabeçalho) (b) MAN reconstruído ($\lambda = 32$), img esq.: taxa=0,159 PSNR=40,31. img dir.: taxa=0,111 PSNR=40,27. Número bytes imagem compactada: 4991. (c) MAN reconstruído ($\lambda = 8$), img esq.: taxa=0,298 PSNR= 43,14. img dir.: taxa=0,199 PSNR=42,67. Número bytes imagem compactada: 9182. 147

B.1 (a) CORRIDOR (256x256), (b) FRUIT (256x256). 166
B.2 (a) MAN (384x384), (b) AQUA (360x288). 167
B.3 (a) SAXO (700x576), (b) TRAIN (700x576). 168

E.1 Gráfico com as trajetórias possíveis para ir da cidade **A** para a cidade **J**. 181

E.2 Círculos acima das cidades **H** e **I** indicam os custos para ir destas cidades para a cidade **J**. 182
E.3 Círculos acima das cidades **E**, **F** e **G** indicam os custos para ir destas cidades para a cidade **J**. 183
E.4 Círculos acima das cidades **B**, **C** e **D** indicam os custos para ir destas cidades para a cidade **J**. 183
E.5 Círculos acima das cidades **B**, **C** e **D** indicam os custos para ir destas cidades para a cidade **J**. 184
E.6 Elementos de programação dinâmica. 185
E.7 Algoritmo de programação dinâmica. 187

Lista de Tabelas

3.1 Valores de taxa e distorção resultantes da codificação da imagem estéreo FRUIT com o ODQ. 54
3.2 Valores de taxa e distorção resultantes da codificação da imagem estéreo MAN com o ODQ. 56

5.1 Valores percentuais de ocorrência dos elementos do dicionário utilizado pelo MMP estéreo (versão 12) para codificar a imagem esquerda (a) CORRIDOR e (b) AQUA. 133
5.2 Valores de taxa e PSNR resultantes do MMP estéreo versão 12 aplicado às imagens estéreo (a) CORRIDOR e (b) AQUA. 148

Capítulo 1

Introdução

O uso prático de sistemas de vídeo estereoscópicos pode trazer mais realismo a muitas aplicações, dentre elas cinema 3D, operações cirúrgicas à distância e vídeo-conferência, devido à percepção de profundidade fornecida por estes sistemas. No entanto, seu uso irrestrito ainda não é possível devido a alguns desafios que precisam ser superados: a utilização de um *display* especial, que dispense o uso de inconvenientes óculos, e a aplicação de técnicas de compressão eficientes dado que imagens estéreo pressupõem no mínimo o dobro de informação a ser armazenada ou transmitida.

1.1 Percepção 3D e geometria estéreo

D. Marr em seu livro *Vision* [2], diz que no sistema de visão humano os dois olhos formam imagens ligeiramente diferentes do mundo. A diferença relativa nas posições dos objetos nas duas imagens é chamada *disparidade*, que é causada pelas diferenças em suas distâncias em relação ao observador. Nosso cérebro é capaz de medir esta disparidade e de usá-la para estimar as distâncias relativas dos objetos ao observador.

Já há muito tempo que imagens/vídeo vêm sendo intensivamente estudados na área de visão por computador, porque a visão estereocópica é um modo popular e básico para perceber o ambiente em 3D. As duas imagens em um par estéreo são chamadas imagens estéreo. A seqüência de imagens estereoscópicas é chamada vídeo estéreo. A principal limitação de imagens/vídeo estéreo é que a posição de visão do

observador é limitada pela posição das câmeras. Em geral, sistemas de imagem 3D permitem maior liberdade para a posição do observador do que sistemas estereoscópicos. Porém, os termos "estereoscópico" e "3D" são usados para a mesma coisa porque imagens/vídeo estereoscópicos podem ser facilmente estendidos para 3D. Em geral, a percepção 3D é baseada em vários efeitos como luz, sombra, tamanho relativo, movimento, oclusão, gradiente de textura, perspectiva geométrica, disparidade, etc. Porém, um dos fatores que contribuem mais efetivamente para a percepção 3D é o fato de que a percepção de profundidade pode ser obtida à partir da observação de uma cena a partir de posições ligeiramente diferentes. A informação de profundidade pode ser obtida baseada em técnicas de visão estéreo, sendo calculada por semelhança de triângulos, à partir do deslocamento relativo (disparidade) e da geometria da câmera.

A geometria de um sistema de imagens estéreo convencional envolve um par de câmeras, que devem ter a mesma distância focal f, e devem ter seus eixos ópticos mutuamente paralelos e separados por uma distância horizontal chamada linha de base (*baseline*) (Fig. 1.1) [1]. As câmeras devem ter seus eixos ópticos perpendiculares à linha de base estéreo, e suas linhas de varredura paralelas à linha de base (horizontal).

Visto que o deslocamento entre os centros ópticos das duas câmeras é puramente horizontal, a posição dos pontos correspondentes nas duas imagens pode diferir somente na componente horizontal. Existem sistemas de imagens estéreo que não possuem câmeras paralelas, e portanto não se pode fazer a consideração de disparidade puramente horizontal. Nestes casos um procedimento mais complexo é necessário para obtenção dos vetores de disparidades, dado que eles podem existir também no eixo vertical.

Na Fig. 1.1 as duas câmeras estéreo esquerda (E) e direita (D) são representadas por uma câmera com orifício puntiforme (*pinhole camera model*), com seus planos de imagem, I_E e I_D, refletidos em torno dos seus eixos de projeção, O_E e O_D, respectivamente. A origem do sistema de coordenadas real é em O_E, a distância focal efetiva de cada câmera é f, e a linha de base estéreo (*baseline*) é b. Os eixos do sistema de coordenadas real X, Y, e Z coincidem com os eixos de coordenadas da câmera esquerda, X_E, Y_E, e Z_E, respectivamente.

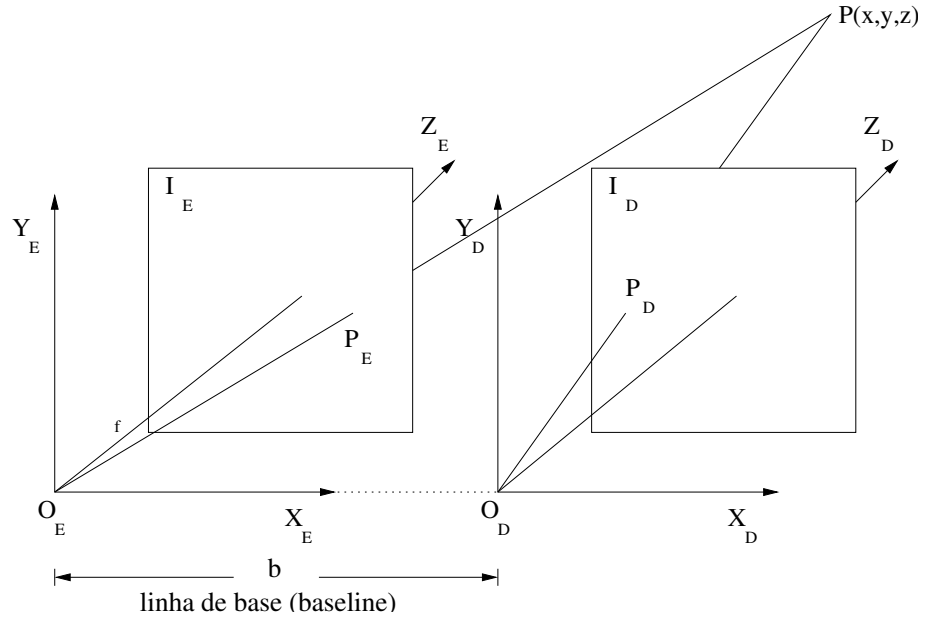


Figura 1.1: Representação geométrica do sistema de câmeras estéreo [1].

Sejam $P_E(x_E, y_E, z_E)$ e $P_D(x_D, y_D, z_D)$ as projeções do ponto $P(x, y, z)$ da cena 3D. Os raios de projeção $\overline{PO_E}$ e $\overline{PO_D}$ definem o plano de projeção do ponto P da cena 3D, chamado *plano epipolar*. Os pontos O_E e O_D, juntamente com qualquer ponto $P(x, y, z)$ no seu campo de visão, definem um plano epipolar. Para um dado ponto P_E na imagem esquerda, seu ponto de correspondência P_D na imagem direita deve estar na linha de interseção entre o plano epipolar e o plano da imagem, que é chamada *linha epipolar*. A linha epipolar define o espaço de busca dentro do qual o ponto correspondente deve ser encontrado. Assim, a restrição epipolar é obtida como um resultado da geometria do sistema de câmeras, e ajuda a limitar o espaço de busca no problema de obtenção de correspondência para análise estéreo. Na geometria convencional de eixos paralelos, todos os planos epipolares interceptam os planos da imagem ao longo de linhas horizontais. Maiores detalhes sobre a geometria estéreo podem ser encontrados em [3, 4].

1.1.1 Determinação da estrutura 3D

A geometria convencional de eixos paralelos fornece um valor de disparidade d para cada par de pontos casados $P_E(x_E, y_E)$ e $P_D(x_D, y_D)$ (veja a Fig. 1.1) como $d = x_E - x_D$. Considerando semelhança entre triângulos, as coordenadas do sistema

real do ponto $P(x, y, z)$ podem ser obtidas por:

$$x = b\frac{x_E}{d}, \quad y = b\frac{y_E}{d}, e \quad z = b\frac{f}{d}$$

onde b é a linha de base estéreo e f a distância focal da câmera.

1.2 Codificação de imagens estéreo

Em 1982, Barnard e Fischler [5] definiram seis passos necessários para a análise estéreo:

- Aquisição da imagem;
- Modelagem da câmera;
- Aquisição de características (*feature primitives*);
- Obtenção da correspondência ou casamento (*matching*);
- Reconstrução da profundidade;
- Interpolação.

Destes passos, a obtenção da correspondência é considerado o mais difícil de resolver e é claramente dependente da escolha das características. Dadas duas vistas de uma cena, a correspondência deve ser estabelecida entre aqueles pontos que são visíveis em ambas as vistas.

As estratégias de casamento podem ser baseadas em área (*area-based*) ou em características pré-adquiridas da imagem (*feature-based*).

No primeiro caso, a maneira mais comumente usada é tomar-se um pixel, e para uma área em torno deste pixel, calcular o menor erro médio quadrático em relação a uma área de mesmo tamanho na outra imagem. As estratégias de obtenção de correspondência baseadas em área têm sido aplicadas com sucesso à análise de imagens aéreas de terrenos, onde a superfície varia suavemente e continuamente. Estas oferecem a vantagem de gerar diretamente um mapa denso, mas são sensíveis a ruído, e funcionam mal em regiões de baixa textura, áreas oclusas e não fornecem boa precisão na localização de descontinuidades.

No caso de (*feature-based*), as características mais usadas são pontos localizados em descontinuidades de intensidade. Estes pontos, denominados *edgels*[6], são úteis porque eles representam pontos de alta confiança para correspondência sem ambiguidades. Porém, métodos baseados em características fornecem um mapa esparso, e requerem uma fase de interpolação bastante complexa.

Existem alguns problemas que tornam mais difícil o processo de casar as características selecionadas na imagem:

- **Variação Fotométrica:** A luz refletida pela cena e gravada pelas câmeras depende da posição da câmera em relação à cena. Assim, quando uma câmera é movida para uma nova posição, ou quando duas câmeras gravam uma cena de dois pontos diferentes, a intensidade nos pontos correspondentes pode ser diferente;

- **Oclusão:** Oclusão ocorre quando um objeto da cena (ou parte dele) é visto apenas por uma das câmeras, e assim aparece em uma vista mas não aparece totalmente na outra.

- **Repetição de textura:** Quando a imagem possui padrões periódicos é possível ocorrer múltipla correspondência. Este problema é enfatizado quando ocorre variação fotométrica e pode levar a uma correspondência errada.

- **Falta de textura:** A maioria dos objetos têm uma superfície com textura. Se os padrões forem escolhidos baseados na textura dos objetos da cena, ocorrerá um problema quando parte da cena não possuir nenhuma textura, pois neste caso a correspondência não será encontrada.

- **Distorção de perspectiva:** Quando a distância entre as câmeras é muito grande, o efeito de distorção de perspectiva é mais perceptível. Partes de um objeto aparecem com um determinado formato em uma das vistas, e aparecem com outro formato na outra vista. Podem ocorrer deformações do objeto, como expansão ou encolhimento de algumas partes, o que torna muito mais difícil encontrar a correspondência entre as duas vistas.

Tanto nos métodos baseados em área, quanto nos métodos baseados em características, quase sempre é necessário interpolar valores para aquelas regiões onde nen-

huma disparidade foi encontrada. Saint-Marc e Medioni [7] apresentam um método para suavizar a superfície enquanto preservam as descontinuidades, o que facilita a detecção de descontinuidades. Isto também tem sido feito recentemente utilizando filtros morfológicos: Gomila e Meyer [8] usando *Levelings* e Bosworth e Acton [9] usando *Lomo Filter*.

1.2.1 Obtenção do mapa de disparidades

A determinação de pontos homólogos da imagem dadas as vistas esquerda e direita é um passo de importância crucial para a análise da imagem estereoscópica. Muitos esforços têm sido feitos para se obter uma estimativa precisa da disparidade. Kanade e Okutomi [10] apresentam um algoritmo *area-based* para a obtenção de um mapa de disparidades preciso, utilizando uma janela adaptativa.

Muitas técnicas de estimação de disparidade desenvolvidas na comunidade de visão computacional podem ser aplicadas à codificação de imagens estéreo, porém a adoção direta destas técnicas pode não ser efetiva por várias razões.

Por exemplo, a principal ênfase de visão estéreo é a estimação precisa do mapa de disparidade, para reconstrução posterior da cena 3D. Uma estimação de deslocamento precisa é crucial em visão estéreo porque o vetor de disparidades corresponde à distância entre as câmeras e o objeto da cena 3D. No entanto, do ponto de vista de codificação, mais importante do que a reconstrução perfeita da cena 3D é o compromisso entre taxa e distorção (*rate-distortion, RD*). Sendo assim, o objetivo principal da codificação de imagens/vídeo estéreo não é estimar a disparidade verdadeira, e sim alcançar uma alta taxa de compressão. Tendo em vista este argumento, alguns autores consideram que não vale a pena calcular um mapa de disparidades preciso se o custo para transmitir ou armazenar este mapa for muito alto [11]. Para estes casos, têm-se usado muito técnicas de correspondência com blocos de tamanho fixo (*fixed size block matching*, FSBM), apesar do campo de disparidades não ser sempre constante dentro do bloco. Métodos baseados em FSBM são simples de implementar e são efetivos em termos de RD porque eles exploram a redundância no mapa de disparidades com uma estrutura regular, que não requer nenhuma informação adicional sobre o campo de disparidades.

Uma maneira simples de descorrelacionar duas imagens é codificar uma im-

agem e a diferença entre as duas imagens. Porém, este método não é eficiente porque cada objeto na cena pode ter disparidade diferente. Assim, os métodos evoluiram para codificação preditiva, onde um mapa de disparidades e um quadro diferenças compensado são codificados.

Devido à similaridade entre imagens estéreo e seqüências de vídeo monoculares, muitas técnicas usadas na codificação de vídeo são aplicadas a codificação de imagens estéreo [12]. A codificação preditiva com estimação de movimento em vídeo aumenta o ganho na codificação explorando a dependência temporal. Isto é possível porque imagens consecutivas em uma seqüência de vídeo tendem a ser similares. Em geral, estimação de disparidades é similar à estimação de movimento no sentido de que ambas exploram a similaridade entre duas (ou mais) imagens de forma a reduzir a taxa de bits. Porém, os esquemas de estimação de movimento desenvolvidos para codificação de vídeo podem não ser eficientes quando aplicados a imagens estéreo, a não ser que restrições geométricas sejam consideradas. Comparando com os vetores de movimento, que podem variar em qualquer direção no plano 2D, o mapa de disparidades terá uma região de busca mais restrita. Deve ser notado também que, diferente de seqüências de vídeo, imagens estéreo são projetadas em duas câmeras, e assim os níveis de intensidade entre as imagens tendem a ser ligeiramente diferentes. Além disso, áreas de oclusão podem ser geradas por qualquer objeto na cena e não apenas por aqueles em movimento, como ocorre na estimação de movimento. Por causa disto, o quadro de diferenças de disparidade compensadas (*disparity compensated difference*, DCD) pode ter uma energia residual maior, e conseqüentemente gastar mais bits comparado com o quadro de diferenças da estimação de movimento (*displaced frame difference*, DFD). Moellenhoff e Maier [13] fazem um estudo bastante detalhado sobre as características do mapa de disparidades, e conseqüentemente apontam quais os melhores métodos para codificá-lo. Pagliari [14] também faz uma análise estatística do mapa de disparidades e dos quadros de diferenças compensada (*disparity compensated residue*, DCR), e conclui que os mapas de disparidade são altamente correlacionados, enquanto que as diferenças compensadas apresentam descontinuidades, causadas por casamento ruim ou devido a oclusões, tendendo assim a serem menos correlacionadas do que os mapas de disparidade.

Métodos baseados em blocos são mais simples, porém podem resultar em um

mapa de disparidades não consistente, o que compromete a qualidade da imagem reconstruida. Pode-se melhorar esta limitação dos métodos baseados em FSBM usando interpolação ou superposição de blocos para se obter um mapa mais preciso. Pode-se usar também blocos de tamanhos diferentes para determinadas regiões da imagem (blocos maiores para regiões mais homogêneas, e blocos menores para regiões com mais detalhes), como por exemplo a abordagem *quadtree* [15, 16, 17, 18].

Uma abordagem mais recente que tem sido proposta para obter e codificar o mapa de disparidade é baseada em objetos. Strintzis e Malassiotis [19] propõem um método baseado em objetos, e afirmam que este tem vantagem em relação aos métodos baseados em blocos, resultando em um mapa mais robusto. As desvantagens porém são mapas esparsos, sem disparidades corretas associadas a áreas uniformes (baixa textura), e a exigência de uma fase de análise complexa para segmentar a cena em objetos e estimar seus movimentos e estrutura.

1.3 *Multidimensional Multiscale Parser* – MMP

O algoritmo *Multidimensional Multiscale Parser* (MMP) foi desenvolvido por Carvalho [20] e trata-se de uma proposta de uma nova classe de algoritmos de compressão. Este algoritmo se baseia em casamento aproximado de padrões recorrentes com escalas, isto é, o algoritmo tenta representar um bloco de dados de entrada usando versões dilatadas ou contraídas de blocos anteriores já processados. No MMP o dicionário é adaptativo: ele é inicializado com valores de luminâncias que compõem a imagem, e, à medida que os blocos da imagem vão sendo processados, novos elementos vão sendo adicionados. Podemos afirmar que o dicionário "aprende" os padrões existentes na imagem. Dentre as principais vantagens podemos citar que não precisamos de um processamento prévio para determinar o dicionário inicial, e principalmente, o dicionário teoricamente pode crescer à vontade, o que torna as possibilidades de representação do sinal mais ricas. Além disso, as atualizações do dicionário não precisam ser enviadas ao decodificador, já que este é capaz de construir o dicionário adaptativamente, tal e qual o dicionário construído pelo codificador, à medida que vai recebendo e decodificando os blocos. Este algoritmo foi comparado com o JPEG para codificar imagens e mostrou-se melhor. Para imagens

mistas, que contêm texto e imagem, ele superou o SPIHT[21]. Uma descrição mais detalhada do MMP encontra-se no Cap. 2 desta tese.

1.4 Objetivo desta tese

Esta tese tem como principal objetivo utilizar o MMP para codificar imagens estéreo, explorando eficientemente as características das mesmas, e, desta forma alcançar melhores resultados de compressão. Um par de imagens estéreo envolve duas imagens similares. Assim, o uso de um método de padrões recorrentes [22, 23, 24], onde os padrões aprendidos na codificação da imagem de referência podem ser usados para codificar a outra imagem, parece ser uma boa idéia.

Além dos elementos iniciais, concatenados, contraídos e expandidos presentes no MMP original, novos elementos foram introduzidos no dicionário. Estes novos elementos são diretamente relacionados com as características intrínsecas às imagens estéreo, e pudemos comprovar uma redução considerável na taxa de codificação da imagem direita em todos os pares estéreo utilizados, após a inserção destes novos elementos.

A novidade deste trabalho é a não transmissão do mapa de disparidades no processo de codificação de imagens estéreo. Em uma primeira fase desta tese conseguimos bons resultados de compressão sem os transtornos envolvidos em calcular, bem como codificar e transmitir o mapa de disparidades. Numa segunda fase, porém, introduzimos o seu cálculo. Apesar disso, no entanto, conseguimos fazer uso desta informação para obtermos melhores resultados de codificação sem ser necessário transmitir o mapa de disparidades. O método se mostrou bastante coerente e promissor, fornecendo melhores resultados de compressão à medida que novas idéias de explorar as características estéreo das imagens eram introduzidas.

1.5 Descrição dos capítulos desta tese

O MMP é uma nova classe de codificadores de imagens, diferente de todas as propostas de codificadores já apresentadas. Exatamente por este motivo faz-se necessário uma descrição detalhada deste algoritmo original no qual nos baseamos. Esta descrição é devidamente apresentada no Cap. 2 desta tese.

O MMP mostrou-se bastante eficaz na codificação de imagens mistas, que incluem imagens e texto [20, 25, 26]. Os testes realizados com o MMP até agora só levaram em consideração imagens monoculares, portanto, a proposta inicial do MMP só explora a similaridade existente dentro de uma mesma imagem. Pelo menos por enquanto ainda não foi verificada a eficiência do MMP para explorar eficientemente as similaridades existentes entre duas ou mais imagens, como uma seqüência de vídeo, ou imagens estéreo. Numa tentativa de verificar a eficiência do MMP aplicado a imagens estéreo, fizemos uma seqüência de nove testes, apresentados no Cap. 3, comparando o desempenho do MMP com um outro método proposto exclusivamente para codificar imagens estéreo. O algoritmo utilizado para efeito de comparação com o MMP foi o *optimal dependent quantisation* (ODQ) [11], que faz a codificação de imagens estéreo através da quantização ótima de cada bloco da imagem. A descrição detalhada deste algoritmo pode ser encontrada no Apêndice A. O objetivo destes testes é tentar obter uma direção a seguir para, utilizando o MMP, explorar eficientemente as características das imagens estéreo.

Uma vez definida a direção, realizamos as primeiras implementações explorando a similaridade estéreo no algoritmo que chamamos de MMP estéreo, apresentadas no Cap. 4. Nesta primeira fase do desenvolvimento desta tese foram introduzidos os elementos deslocados no dicionário, como uma tentativa inicial de explorar as similaridades existentes entre as duas imagens do par estéreo. Nesta fase inicial não calculamos o mapa de disparidades. Neste capítulo descrevemos seis versões do MMP estéreo. Na primeira versão utilizamos um único dicionário para codificar as duas imagens do par estéreo. O processamento é feito por linhas de blocos (*row of blocks - ROBs*) correspondentes. Após processamos uma ROB da imagem esquerda, geramos deslocamentos de blocos desta ROB e inserimos no dicionário que será utilizado para processar a ROB direita correspondente. Os elementos deslocados são descartados após o processamento de cada ROB direita, ou seja, eles só são aproveitados de uma ROB esquerda para a sua ROB direita correspondente. Isto está de acordo com uma das restrições de processamento de imagens estéreo que estabelece que existe correspondencia apenas entre ROBs de uma mesma linha epipolar.

Fizemos um estudo estatístico sobre a ocorrência de cada tipo de elemento

presente no dicionário, e verificamos que os deslocados ocorriam muito na imagem direita, como esperávamos, mas também ocorriam na imagem esquerda. Isto nos levou a deixar os elementos deslocados no dicionário, acumulados de uma ROB para a outra. Esta foi a segunda versão do MMP estéreo. O fato de utilizar os elementos deslocados acumulados melhorou os resultados da codificação da imagem esquerda, mas piorou os da imagem direita. Isto nos levou à idéia fundamental deste capítulo (e desta tese), que foi separar os dicionários. Para codificar a imagem esquerda utilizamos um dicionário com elementos iniciais, concatenados, contraídos e expandidos, que chamamos de "elementos comuns", e também os deslocados acumulados; para codificar a imagem direita, temos a opção de escolher entre usar este dicionário comum, ou usar um dicionário com elementos deslocados da ROB esquerda correspondente. O MMP escolhe o dicionário cujo elemento foi aquele que gerou o menor custo. Isto resultou em um ganho muito significativo, porque ganhamos na codificação da imagem esquerda com o uso dos elementos deslocados acumulados, e ganhamos na codificação da imagem direita, com o uso dos elementos deslocados somente da ROB esquerda correspondente. Utilizamos um *flag* para indicar qual dos dois dicionário foi utilizado para codificar o bloco da imagem direita. Mostramos que a ocorrência dos elementos deslocados é tão grande que compensa o gasto de bits para o *flag*. Ainda neste capítulo mostramos algumas melhorias nesta versão, como gerar elementos deslocados limitados a uma janela de busca dada pela faixa de disparidades da imagem estéreo (versão 4), e fazer os deslocamentos de 1/2 pixel (versão 5). A versão 6 congrega todas as implementações descritas neste capítulo. Neste capítulo não calculamos os vetores de disparidade entre as duas imagens estéreo. Mostramos que mesmo assim conseguimos bons resultados de codificação.

A segunda fase de desenvolvimento desta tese é mostrada no Cap. 5. Nesta fase introduzimos o cálculo dos vetores de disparidade apenas como uma maneira de gerar elementos mais precisos para o dicionário, que serão usados para a codificação da imagem direita (*target*). Um estudo estatístico detalhado mostrou a importância de utilizar estes novos elementos, que chamamos de "deformados", para a codificação eficiente do par estéreo, principalmente da imagem direita. A principal vantagem deste método é que, apesar de calcularmos o mapa de disparidades, resultando assim em uma imagem reconstruída com menos distorção, não enviamos este mapa ao

decodificador, evitando assim um aumento desnecessário (*overhead*) na taxa. Neste capítulo descrevemos as versões 7 a 12 do MMP estéreo.

Na versão 7 introduzimos os elementos "deformados". Nesta versão temos dois dicionários diferentes. O primeiro dicionário, chamado de dicionário comum, possui elementos iniciais, concatenados, contraídos e expandidos, e é utilizado para codificar a imagem esquerda. O segundo dicionário possui somente elementos "deformados". Para codificarmos a imagem direita utilizamos o dicionário comum ou o dicionário de "deformados". A escolha do melhor dicionário é indicada por um *flag*. Nesta versão removemos os elementos deslocados tanto do dicionário comum (deslocados acumulados) quanto do dicionário de "deformados" (deslocados da ROB). Fizemos isto para observar isoladamente os benefícios de utilizar os elementos "deformados" na codificação da imagem direita.

Na versão 8 do MMP estéreo realizamos uma melhoria na versão 7. Reduzimos o número de elementos deformados a uma janela que possui como centro o elemento deformado correspondente ao bloco que está sendo processado. Este elemento deformado correspondente é dado pelos vetores de disparidades calculados para o par de ROBs anteriores. Esta janela é um parâmetro dado pelo usuário. Fizemos testes para janelas de vários tamanhos, mas a que deu melhores resultados foi uma janela com um único elemento deformado, exatamente aquele bloco equivalente ao bloco da imagem direita que está sendo processado. Este procedimento equivale a um refinamento na estimativa do modelo, onde reduzimos o número de elementos e utilizamos um elemento mais significativo para representar o bloco da imagem direita.

A versão 9 traz de volta os elementos deslocados para codificar a imagem direita. Estes elementos deslocados são gerados para todos os blocos da ROB, e não só para a janela de busca correspondente ao bloco da imagem direita que está sendo processado.

Na versão 10 nós restringimos os deslocados a uma janela dada pelo usuário, cujo centro é dado pelos vetores de disparidade, da mesma forma que fizemos com os elementos "deformados" na versão 8.

Na versão 11 nós trouxemos de volta os elementos deslocados acumulados que melhoram a codificação da imagem esquerda. E finalmente, na versão 12 nós

inserimos elementos rotacionados para codificar a imagem esquerda. Isto foi feito porque observamos ao longo desta tese que aumentar o número de elementos para codificar a imagem esquerda sempre melhora os resultados de compressão desta imagem.

A cada versão do MMP estéreo conseguimos melhorar os resultados de compressão, tanto da imagem esquerda, quanto (e principalmente) da imagem direita, que era nosso maior alvo. No final do Cap. 5 mostramos os resultados comparativos entre a primeira e a última versão do MMP estéreo, para várias imagens estéreo. Nestes gráficos podemos observar uma melhoria significativa dos resultados de compactação destas imagens.

Mostramos os resultados de taxa e distorção para várias imagens estéreo, e o MMP estéreo se mostrou bastante coerente e promissor.

No Cap. 6 apresentamos as conclusões gerais. Conseguimos atingir nosso objetivo principal, que era explorar com eficiência as características das imagens estéreo, utilizando o MMP. A novidade deste trabalho é que conseguimos atingir bons resultados de compressão mesmo sem a transmissão dos vetores de disparidade.

O Apêncice A traz uma descrição detalhada do algoritmo ODQ, que foi utilizado no Cap. 3 e no Cap. 4, para efeito de comparação com o desempenho do MMP.

O Apêndice B mostra as imagens estéreo originais que foram utilizadas nesta tese.

O Apêndice C enumera as versões do MMP estéreo, com uma descrição suscinta de cada uma delas. Estas informações podem fornecer ao leitor uma referência rápida sobre o que faz uma determinada versão do MMP estéreo.

O Apêndice E mostra um exemplo de programação dinâmica (algoritmo de Viterbi) que é um método utilizado para otimizar os quantizadores do algoritmo ODQ (que utilizamos no Cap. 3 e no Cap. 4 para comparar com o MMP). Descrevemos o funcionamento do algoritmo de Viterbi através de um exemplo simples, para facilitar seu entendimento.

Capítulo 2

Multidimensional Multiscale Parser - MMP

2.1 Introdução

Neste capítulo faremos uma descrição detalhada do algoritmo *multidimensional multiscale parser*(MMP) [20, 27, 25]. Este algoritmo representa uma nova classe de algoritmos de compressão com perdas, baseado em casamento aproximado de padrões recorrentes com escalas. O termo "recorrente" vem da idéia de representar um sinal de entrada por aproximações de partes previamente processadas deste sinal. O casamento aproximado com escalas é uma extensão do casamento de padrões comum, no qual existe a possibilidade de casar vetores de diferentes tamanhos. Para isto utilizam-se transformações em escalas que ajustam o tamanho dos vetores antes de fazer o casamento.

O sinal de entrada pode ser multidimensional. Para facilitar o entendimento do MMP, neste capítulo começaremos por descrevê-lo utilizando um sinal de entrada unidimensional.

O MMP possui um dicionário inicial formado por vetores de tamanho variável, que serão usados para representar o sinal de entrada. Ao longo do processamento novos vetores são acrescentados ao dicionário. Estes novos vetores são representações mais aproximadas dos padrões presentes nas partes do sinal de entrada que já foram processadas. Por este motivo dizemos que o dicionário é adaptativo, porque ele vai aprendendo com os padrões existentes no sinal de entrada à medida que o sinal

vai sendo processado. O objetivo é que, ao longo do processamento, o dicionário contenha vetores cada vez mais representativos do sinal de entrada, sem no entanto aumentar a quantidade de bits utilizados na codificação, o que irá resultar em uma melhor compressão do restante do sinal.

A representação do sinal por vetores de tamanho variável é uma das principais características deste algoritmo. A princípio escolhem-se vetores maiores no dicionário para representar o sinal de entrada. Estes vetores no entanto podem não atender ao requisito mínimo de qualidade da imagem reconstruída (distorção final). Então tenta-se representar o sinal de entrada por vetores menores, o que pode comprometer o requisito de taxa máxima permitida. Para evitar este problema, o algoritmo otimiza os custos envolvidos na codificação utilizando um critério de taxa e distorção (*rate* $\times$ *distortion* – RD) [28, 29], garantindo a melhor representação possível para o sinal de entrada atendendo tanto aos requisitos de taxa quanto aos requisitos de distorção exigidos.

A principal vantagem deste algoritmo é que o mesmo dicionário construído pelo codificador pode ser construído também pelo decodificador, não havendo assim necessidade de transmiti-lo. A única informação a ser transmitida para o decodificador é uma seqüência de bits (*bitstream*) informando se houve ou não divisões na representação daquele sinal, e quais os índices dos elementos do dicionário que foram escolhidos para representar aquele sinal.

A seguir descreveremos o MMP com mais detalhes.

2.2 Descrição do algoritmo MMP

2.2.1 Exemplo de aplicação do MMP para representação de um sinal unidimensional

Iniciaremos a descrição do MMP de uma forma intuitiva, utilizando como exemplo um sinal unidimensional para tornar mais fácil o entendimento do algoritmo.

Seja um sinal de entrada unidimensional u com oito amostras, mostrado na Fig. 2.1.

Construímos um dicionário inicial com vetores cujas amplitudes contenham toda a faixa dinâmica desejada, que possa representar bem as amplitudes presentes

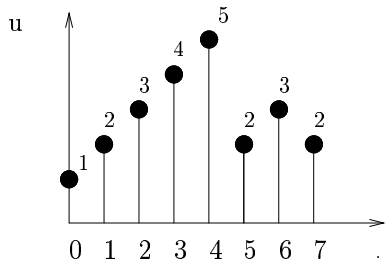


Figura 2.1: Exemplo de um sinal de entrada unidimensional.

no sinal de entrada original. Neste nosso exemplo, considerando que o sinal de entrada só possui amplitudes entre 0 e 5, a faixa dinâmica será de 0 a 5 também, mas isto pode variar conforme deseje o usuário. Aqui o nosso primeiro vetor será composto por oito amostras (tamanho do sinal de entrada inicial) iguais à amplitude do primeiro valor da faixa dinâmica do sinal (vetor 0 na Fig. 2.2).

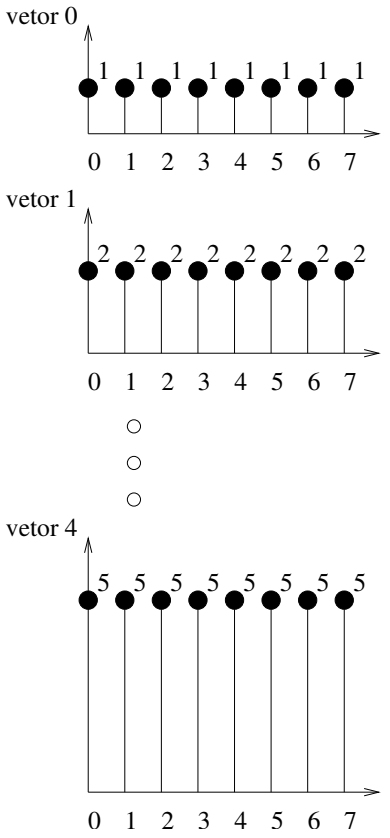


Figura 2.2: Vetores do dicionário inicial para o sinal unidimensional u exemplificado

O segundo vetor será composto por oito amostras iguais ao segundo valor da faixa dinâmica de amplitudes (vetor 1), e assim por diante, até o vetor 4.

Cada um dos vetores iniciais do dicionário são divididos em dois vetores menores, cada um com metade do número de amostras do vetor inicial. Esta divisão é mostrada na Fig. 2.3 para o primeiro vetor do dicionário, e na Fig. 2.4 para o quinto vetor. O processo de divisão é similar para os demais vetores. As divisões dos vetores são realizadas até que o vetor tenha somente uma amostra. Estas divisões são agrupadas por nível. Um dado nível do dicionário possui vetores com um mesmo tamanho. Assim construímos um dicionário com vetores de tamanho variável, onde cada nível corresponde a uma escala. Cada vetor do dicionário possui um índice que o identifica dentro daquele nível. No nível mais alto deste exemplo temos os vetores iniciais com índices 0 a 4.

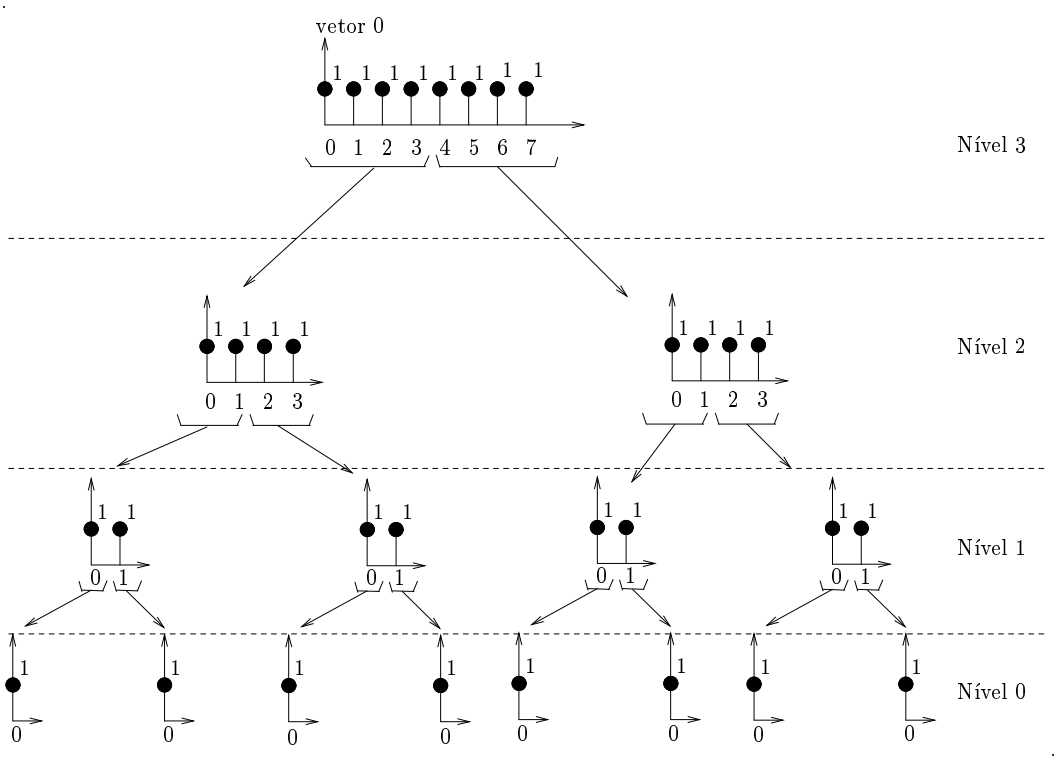


Figura 2.3: Divisões possíveis no MMP para um vetor do dicionário inicial, criado a partir da primeira amostra do sinal unidimensional u exemplificado.

O MMP inicia o processamento tentando representar o sinal de entrada pelos vetores iniciais do dicionário que possuem o mesmo tamanho do sinal de entrada.

A busca pelo melhor vetor é feita segundo um critério de taxa e distorção. A distorção D é calculada pelo erro médio quadrático entre o sinal de entrada e um vetor do dicionário. A taxa R é calculada pela quantidade de bits gastos para

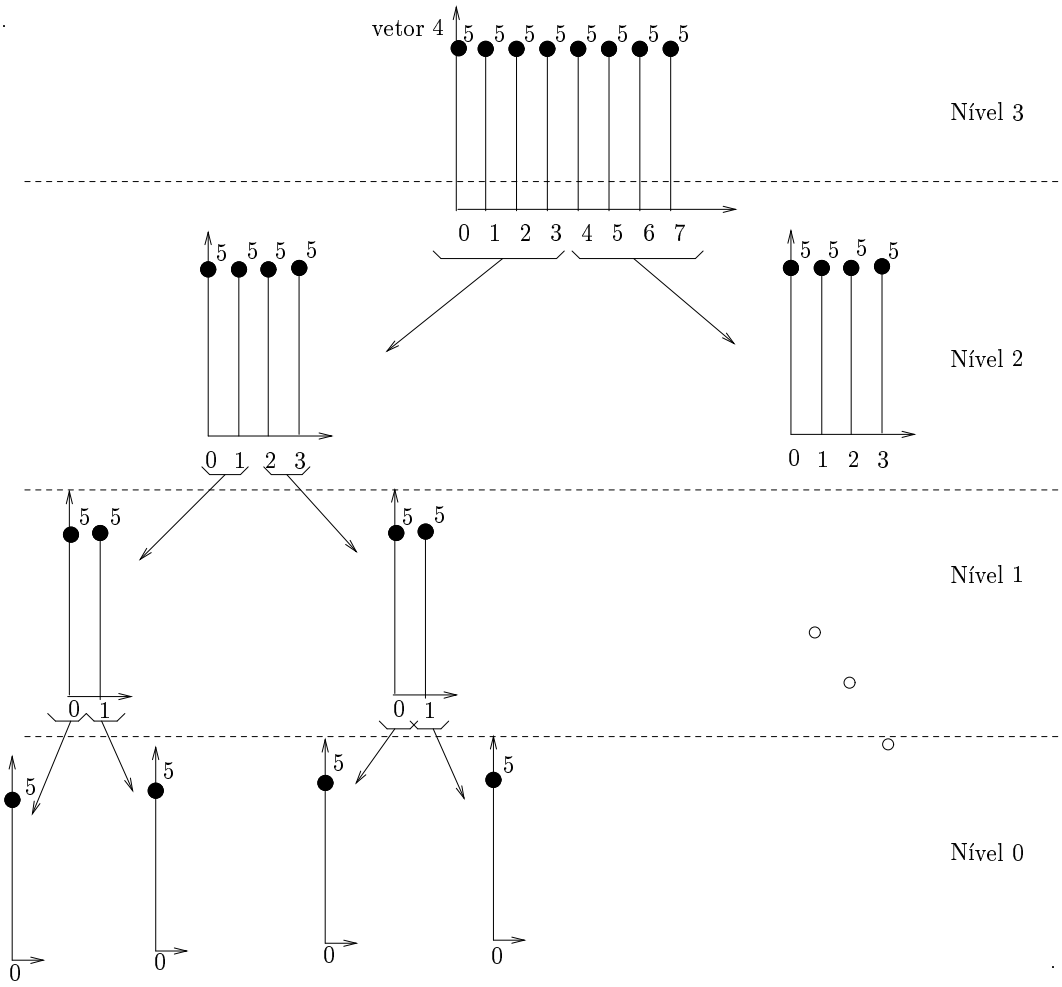


Figura 2.4: Divisões possíveis no MMP para um vetor do dicionário inicial, criado a partir da quinta amostra do sinal unidimensional u exemplificado.

codificar o índice do vetor do dicionário juntamente com os *flags* que indicam se houve ou não divisão de um vetor em um determinado nível. O custo total J é dado por $D + \lambda R$, onde λ é uma constante que prioriza a distorção ou a taxa, conforme exija a aplicação do usuário. Quando se deseja uma codificação sem perdas, coloca-se $\lambda = 0$, pois assim não importando a taxa, o processo de codificação só termina quando o objetivo de distorção mínima igual a zero for alcançado. Da mesma forma, se colocarmos um valor de λ alto, estaremos priorizando a taxa. Para alcançar a taxa mínima determinada, a distorção será penalizada, e o sinal reconstruído não terá boa qualidade.

Procura-se dentre os vetores existentes no nível mais alto do dicionário qual o vetor que gerou o menor custo J. Este vetor é o escolhido para representar o sinal

de entrada neste nível do dicionário.

Observando os vetores do nível mais alto do dicionário (no caso nível 3 das figuras 2.3 e 2.4), percebemos que eles são aproximações grosseiras do sinal de entrada. Para melhorar a aproximação do sinal de entrada pelos vetores do dicionário podemos então dividir o sinal de entrada em duas partes de quatro amostras, como mostra a Fig. 2.5, e tentar representar cada parte por vetores iniciais do dicionário que tenham o mesmo tamanho (4 amostras). Isto ocorre no nível 2 do dicionário. Esta divisão, bem como também as divisões dos vetores do dicionário mostradas nas figuras 2.3 e 2.4), seguem o modelo de uma árvore binária.

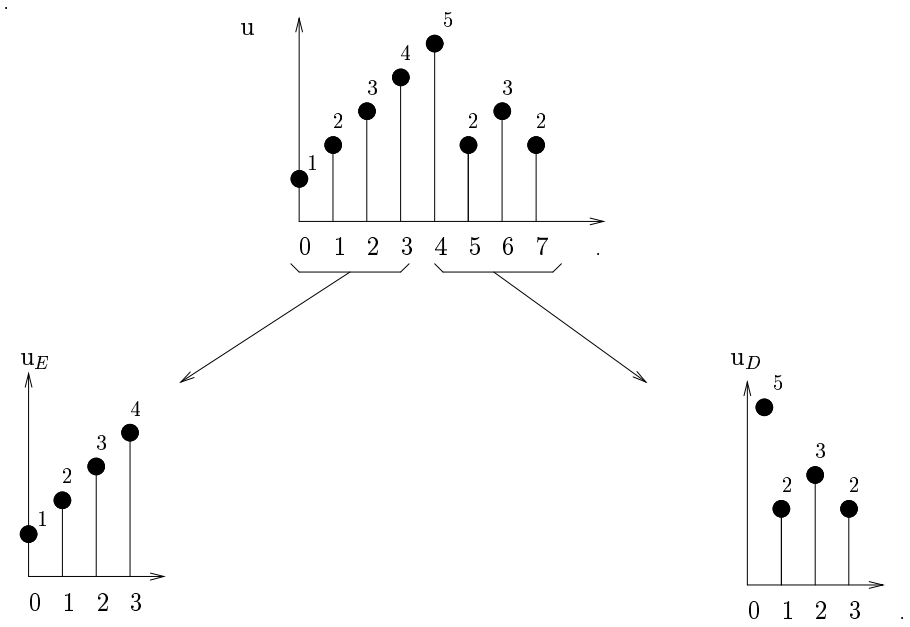


Figura 2.5: Divisão do sinal unidimensional u em duas partes de tamanhos iguais u_E e u_D.

Da mesma forma que a busca foi realizada anteriormente no nível mais alto do dicionário, agora também procura-se dentre os vetores existentes neste nível quais aqueles que proporcionam os menores custos para aproximar cada parte separada do sinal de entrada.

Considere a Fig. 2.6. Existe um custo associado a um vetor do nível mais alto (nível 3) se o escolhermos para representar o sinal de entrada. Chamamos este custo de J. Existe também um custo associado a cada vetor no nível imediatamente abaixo (nível 2) se escolhermos dois vetores de 4 amostras pertencentes a este nível, para representarem cada um uma parte do sinal de entrada. Chamamos estes custos

de J_E e J_D respectivamente. Se $(J_E + J_D) < J$, então significa que, dentre os vetores presentes no dicionário, representar o sinal por estes dois vetores menores é melhor do que representar por aquele vetor maior. Neste caso, indicamos através de um *flag* que houve divisão na árvore binária, e continuamos descendo a árvore, procurando melhores representações em níveis mais baixos. Quando encontramos dois vetores menores que juntos aproximam melhor o sinal do que um vetor maior já presente no dicionário, concatenamos estes dois vetores e inserimos um novo vetor no dicionário. A este processo damos o nome de "aprendizado" do dicionário, porque estamos introduzindo um conhecimento específico a respeito dos padrões presentes no sinal de entrada que poderão ser aproveitados para codificar outras partes do sinal que tenham padrões parecidos com estes.

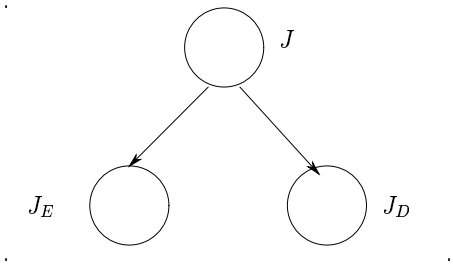


Figura 2.6: Custos associados à representação de um sinal por um determinado vetor do dicionário num nível mais alto (custo J), ou por dois vetores do dicionário num nível mais baixo (custos J_E e J_D).

Encontramos os dois vetores no nível 2 que apresentaram o menor custo para aproximar o sinal u_E e o sinal u_D. Como estamos procurando a melhor representação para o sinal, continuamos o processo e dividimos o sinal u_E novamente em duas partes de tamanhos iguais, agora com duas amostras cada (Fig. 2.7). Fazemos o mesmo para o sinal u_D. Procuramos agora o melhor casamento para estas quatro partes do sinal de entrada no nível 1 do dicionário.

Encontramos os quatro vetores no nível 1 que apresentaram o menor custo para aproximar os sinais u_E1, u_E2, u_D1 e u_D2. Como estamos procurando a melhor representação para o sinal, continuamos o processo e dividimos o sinal u_E1 novamente em duas partes de tamanhos iguais, agora com uma amostra cada (Fig. 2.8). Fazemos o mesmo para os sinais u_E2, u_D1 e u_D2. Procuramos agora o melhor casamento para estas oito partes do sinal de entrada no nível 0 do dicionário.

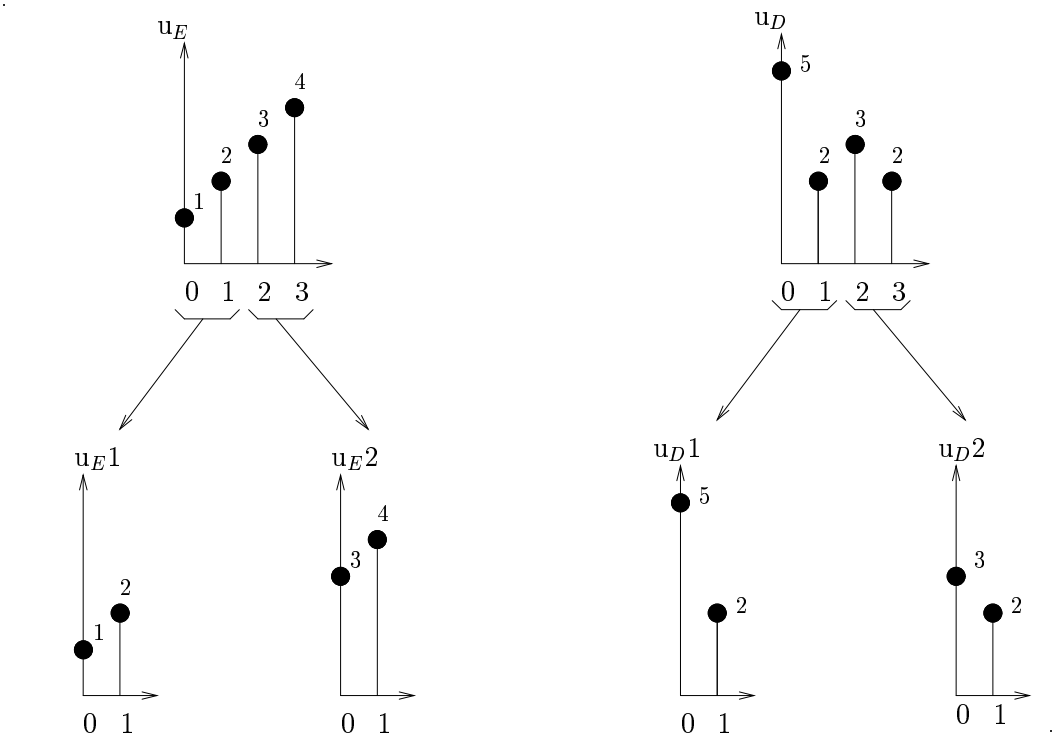


Figura 2.7: Divisão dos sinais u_E e u_D em duas partes de tamanhos iguais.

Neste nível é possível encontrar casamentos perfeitos para as amostras do sinal de entrada original. Isto porque escolhemos uma faixa dinâmica inicial que cobria todas as amplitudes presentes no sinal. Para cada amostra encontrada, concatenamos os dois vetores que melhor se aproximaram e inserimos no dicionário um novo vetor, que representa melhor o sinal de entrada do que qualquer outro vetor de mesmo tamanho já existente no dicionário.

A informação resultante deste processamento, iniciando-se no nível mais alto do dicionário, é uma seqüência de bits informando se houve ou não divisão da árvore. Em algum nó da árvore em que não houve divisão, a seqüência de bits inclui o código do índice do vetor do dicionário, no dado nível, que melhor representou a parte do sinal de entrada correspondente. Esta informação é enviada ao decodificador. O decodificador inicia o processamento com o mesmo dicionário inicial construído pelo codificador. Isto é possível porque o codificador antes de iniciar o processo envia ao decodificador os valores mínimo e máximo das amostras presentes no sinal, bem como também o passo de representação destas amostras no dicionário. Por exemplo, para o sinal unidimensional mostrado aqui o mínimo valor de amostra é 1, e o máximo é 5. O passo é 1. Então temos 5 vetores iniciais, onde cada um representa uma

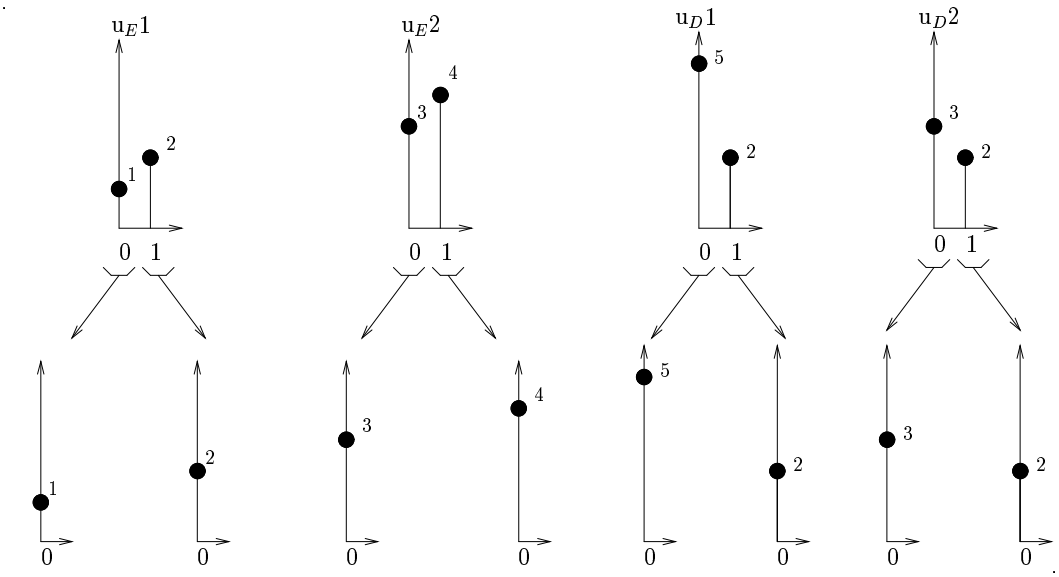


Figura 2.8: Divisão dos sinais u_E1, u_E2, u_D1 e u_D2 em duas partes de tamanhos iguais.

amostra do sinal. Uma vez montado o dicionário inicial, o decodificador comeca a decodificar a árvore ótima recebida. Cada bit representando um processo de divisão faz com que o MMP desça um nível da árvore. Quando não há divisão, a seqüência de bits informa o código do vetor do dicionário que melhor representa aquela parte do sinal. Neste caso, é só reconstruir a parte do sinal pegando o vetor correspondente do dicionário.

Para entendermos melhor como é feita a decodificação, consideremos a seguinte seqüência de bits recebida pelo decodificador:

$$1, 1, 0, 15, 1, 12, 10, 0, 23.$$

O decodificador começa o processamento no nível mais alto do dicionário (no caso deste exemplo no nível 3, onde o tamanho dos vetores é 8). Os vetores iniciais foram compostos da mesma forma que no codificador, pois o decodificador recebeu no início do processamento a faixa dinâmica de amplitudes presentes no sinal original. Para facilitar o acompanhamento do processo, denominamos os nós da árvore por letras. O nó pai é a letra A. Os nós filhos de A são B e C. Os nós filhos de B são D e E. Os nós filhos de E são F e G. Veja na Fig. 2.9 a árvore completa.

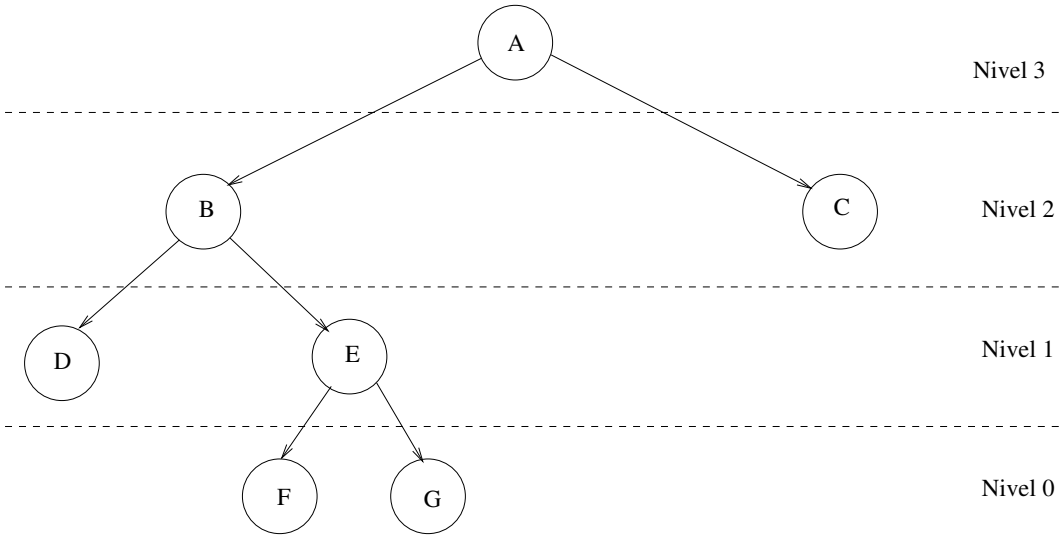


Figura 2.9: Árvore binária que irá representar um sinal de entrada, e suas possíveis divisões. Os nós estão denominados por letras para facilitar o entendimento da decodificação da seqüência de bits.

O dicionário no início da decodificação do seqüência de bits é composto pelos vetores mostrados na Fig. 2.10.

O primeiro valor da seqüência de bits é 1. Começamos pelo nó A, no nível 3 do dicionário. Este valor 1 corresponde a um *flag* indicando que houve uma divisão. Descemos então um nível na árvore, à esquerda. Sempre descemos primeiro pela esquerda, depois retornamos ao nó pai e descemos à direita. Esta seqüência é a mesma seguida no codificador.

Vamos então para o nó B. No nível 2, recebemos de novo um 1. Este valor indica outra divisão. Descemos novamente um nível na árvore, à esquerda, e vamos para o nó D.

No nível 1, recebemos um 0 da seqüência de bits. Este valor é um *flag* indicando que não há mais divisões, ou seja, este é um nó terminal. Sendo assim, o próximo valor da seqüência de bits será necessariamente um índice de um vetor do dicionário, no nível 1, que melhor representa esta parte do sinal. Recebemos um 15 da seqüência de bits correspondendo ao vetor 15, com amplitudes 100 e 200, mostrado na Fig. 2.11, ao lado do nó terminal D.

Continuando a nossa decodificação, retornamos ao nó B, e descemos pela direita, para o nó E. (Fig. 2.12). Recebemos um 1 da seqüência de bits. Este valor 1

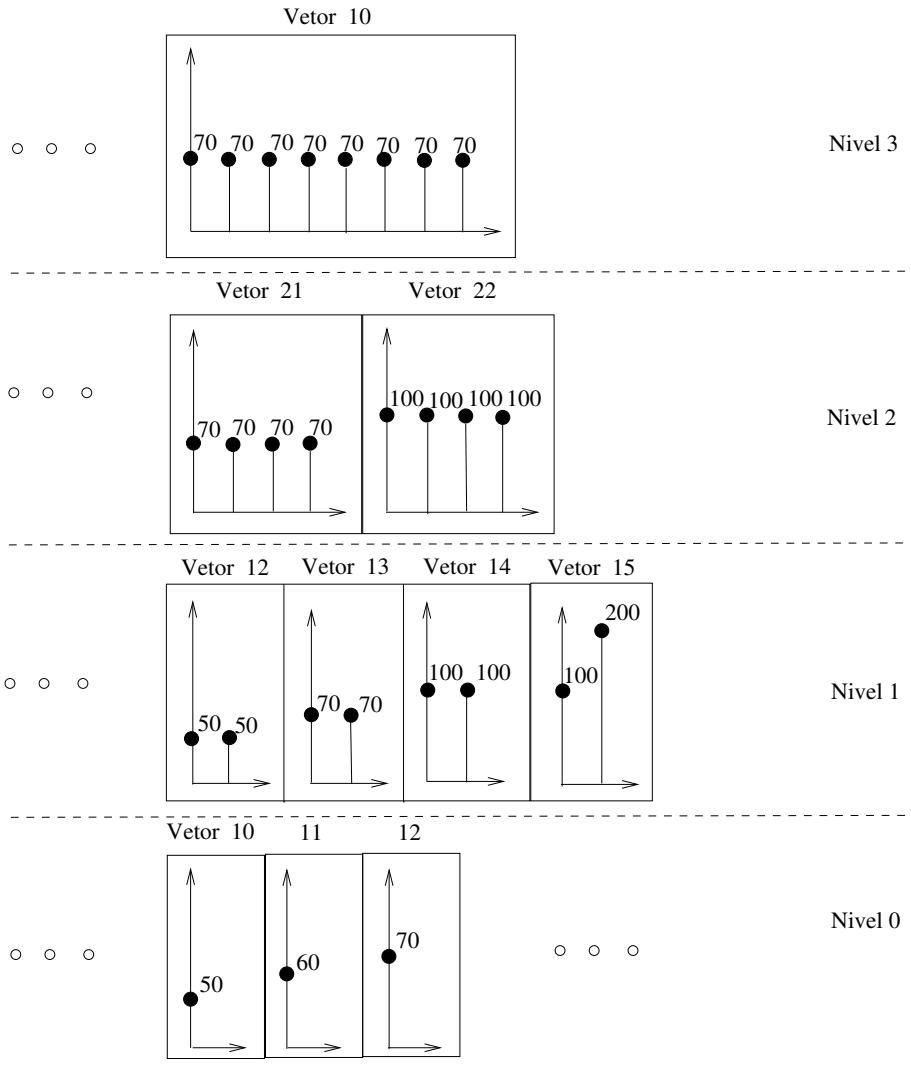


Figura 2.10: Dicionário no início da decodificação da seqüência de bits.

corresponde a um *flag* indicando que houve uma divisão. Descemos então um nível na árvore, à esquerda, e vamos para o nó F. O nó F é um nó terminal. Sabemos disso porque estamos no último nível da árvore, onde não podem mais ocorrer divisões. Sendo assim, o próximo valor da seqüência de bits é necessariamente um índice do vetor do dicionário, no nível 0, que melhor representa esta parte do sinal. Recebemos um valor 12, indicando que devemos pegar o vetor 12 do dicionário (este vetor contém uma amostra com amplitude 70, como mostra a Fig. 2.12, ao lado do nó terminal F).

Retornamos ao nó E, e descemos pela direita para o nó G. Sendo este nó também um nó terminal, o próximo valor da seqüência de bits também corresponde a um índice. Notem que, o fato de conhecermos o nível da árvore permite que econo-

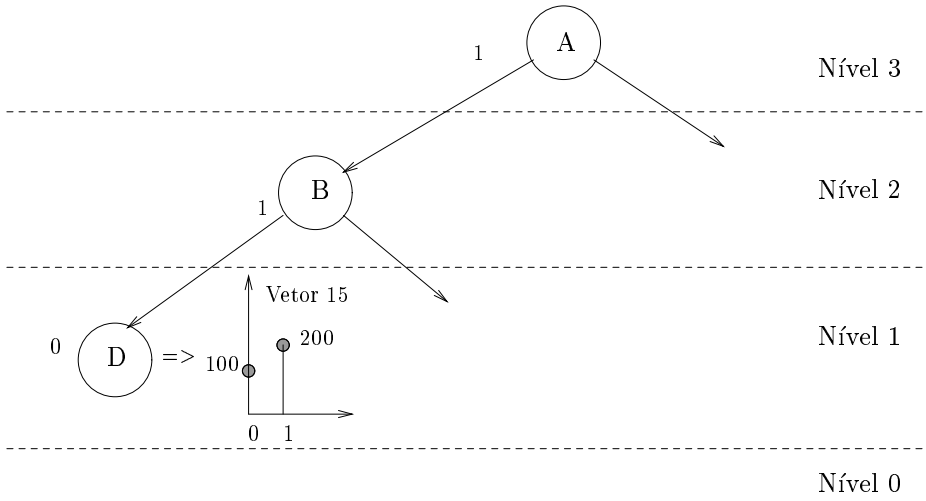


Figura 2.11: Seqüência de decodificação da seqüência de bits exemplificado. Ao lado de cada nó temos os *flags*. *Flag=1* indica divisão. *Flag=0* indica nó terminal. Para cada nó terminal temos um índice, indicando o vetor do dicionário que representa aquela parte do sinal.

mizemos alguns bits para indicar *flag=0* quando estivermos no nível 0 do dicionário. Quando se trata de um nó terminal que não seja do nível 0, esta informação é imprescindível, para podermos saber que o próximo valor da seqüência de bits será um índice. Recebemos um valaor 10 da seqüência de bits, indicando que devemos pegar o vetor 10 do dicionário para representar esta parte do sinal (este vetor contém uma amostra com amplitude 50, como mostra a Fig. 2.12, ao lado do nó terminal G) .

Visitados os dois nós filhos, F e G, retornamos ao nó pai E. Neste momento, concatenamos os dois vetores do dicionário no nível 0 e inserimos este novo vetor na posição 16, no nível 1 do dicionário. Este novo vetor corresponde às amplitudes 70 do nó F, e 50 do nó G, juntas num mesmo vetor. O dicionário aprendeu este padrão do sinal (veja Fig. 2.13).

O dicionário após a inserção deste novo elemento correspondente ao nó E (resultante da concatenação dos nós F e G) é mostrado na Fig. 2.14.

Além deste elemento concatenado, também geramos expansões deste elemento, e os inserimos em todos os níveis possíveis do dicionário, no caso nos níveis 2 (vetor 23) e 3 (vetor 11) (veja Fig. 2.15). As contrações neste caso geram os mesmos vetores correspondentes ao nó F e nó G, portanto não é necessário inserir no

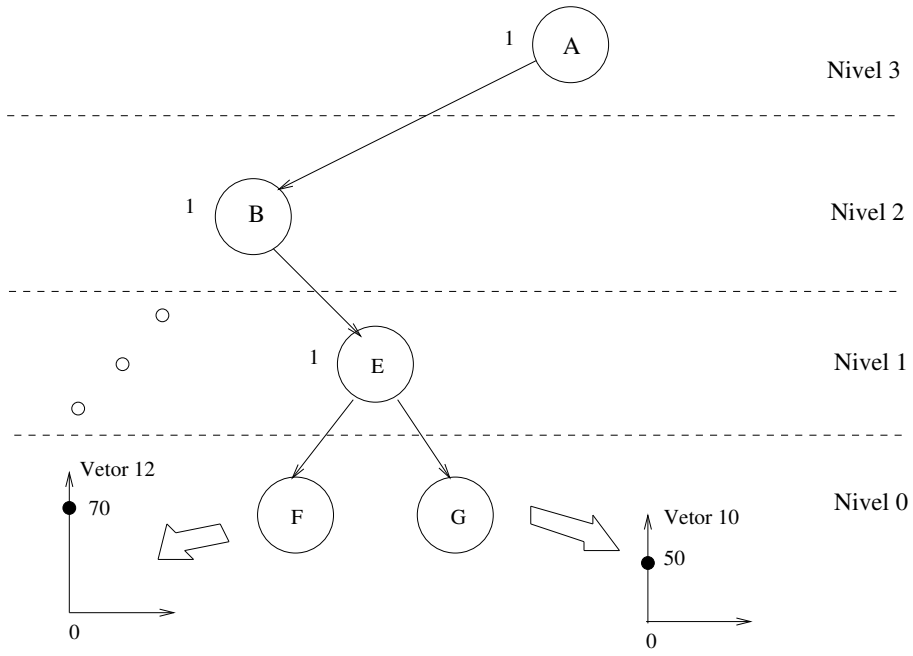


Figura 2.12: Seqüência de decodificação da seqüência de bits exemplificada. Ao lado dos nós terminais mostramos os vetores correspondentes do dicionário.

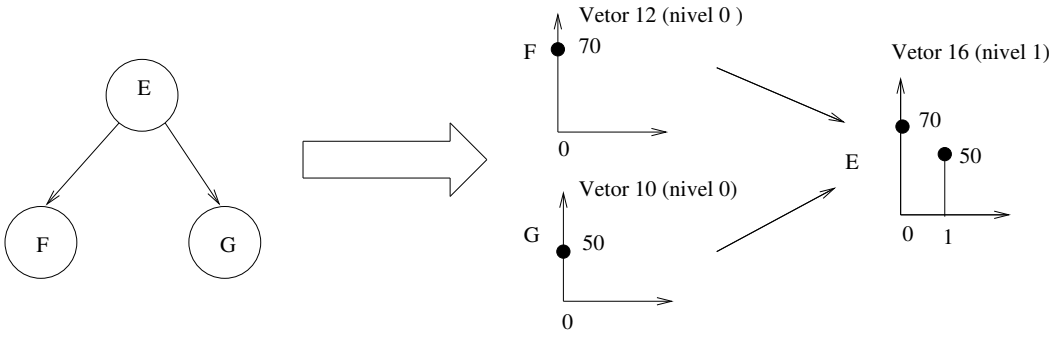


Figura 2.13: Concatenação de dois nós terminais (F e G), originando um novo vetor (que corresponde ao nó E) a ser inserido no dicionário, num nível acima do nível dos nós F e G.

dicionário vetores repetidos.

O dicionário após a inserção das expansões do vetor correspondente ao nó E, é mostrado na Fig. 2.16. A expansão do vetor 16 do nível 1, correspondente ao nó E, gerou um vetor que foi inserido no nível 2 do dicionário, na posição 23. Este, por sua vez, foi expandido e gerou um novo vetor que foi inserido na posição 11, no nível 3 do dicionário.

Retornamos agora ao nó pai B. Como foram visitados os dois nós filhos do nó

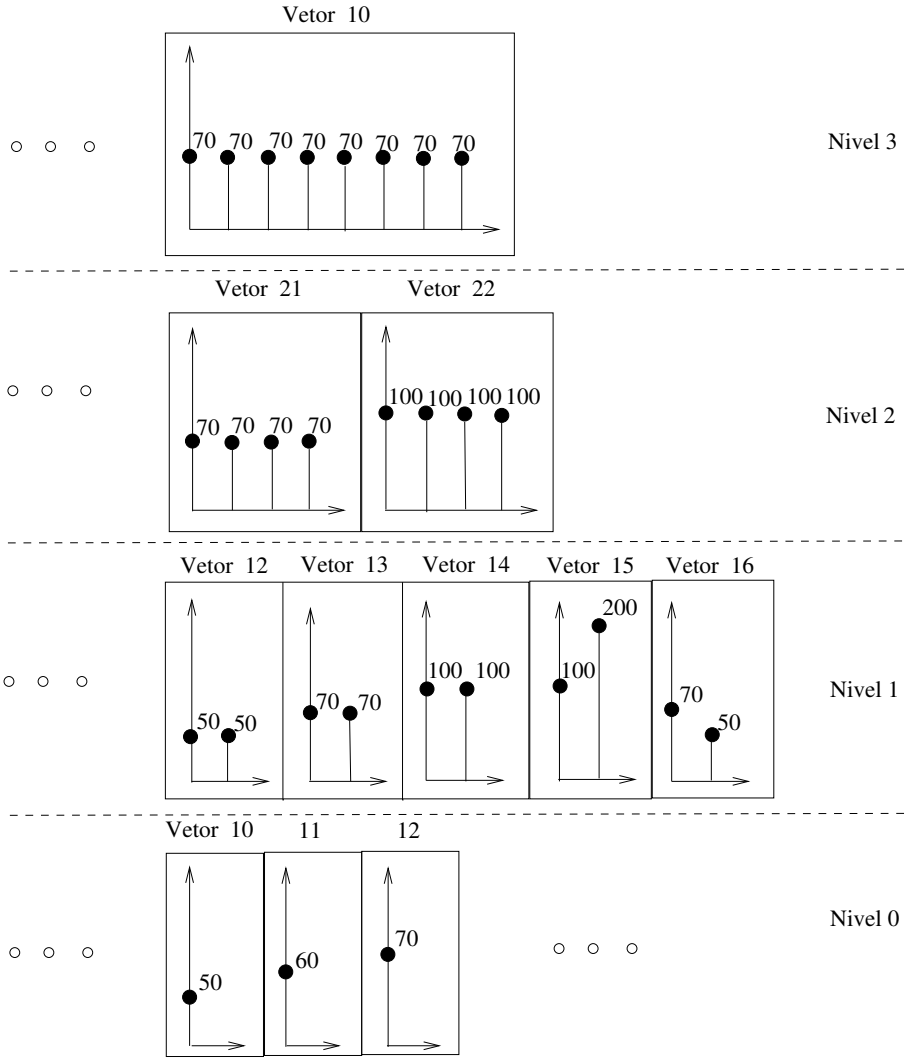


Figura 2.14: Dicionário após a inserção do vetor correspondente ao nó E, resultante da concatenação dos nós F e G. Este novo vetor foi inserido na posição 16, no nível 1 do dicionário.

B, concatenamos os dois vetores correspondentes a estes nós e formamos um novo vetor. Este novo vetor corresponde às amplitudes 100 e 200 do nó D, e 70 e 50 do nó E (veja Fig. 2.17). No dicionário estes vetores correspondem às posições 15 e 16 no nível 1, e o vetor resultante da concatenação foi inserido na posição 24, no nível 2 (veja Fig. 2.18). O dicionário aprendeu mais este padrão do sinal.

Geramos uma expansão deste elemento concatenado (veja Fig. 2.19), e o inserimos no nível 3 do dicionário, na posição 12. Geramos também uma contração deste elemento concatenado e a inserimos no nível 1, na posição 17 (veja Fig. 2.20). A contração gerada para o nível 0 (vetor com uma amostra com amplitude 100) já

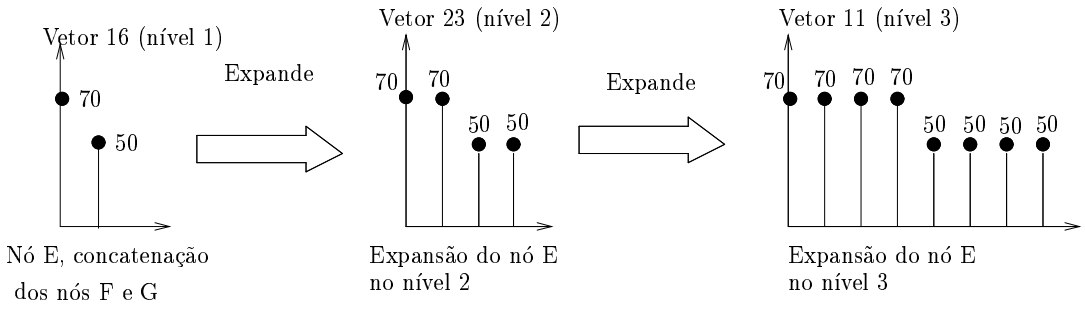


Figura 2.15: Expansões do vetor correspondente ao nó E, a serem inseridas nos níveis superiores do dicionário.

existe no dicionário, por isto não será inserida.

Retornamos ao nó pai A, e descemos pela direita, para o nó C. Recebemos um 0 da seqüência de bits indicando que este é um nó terminal. O próximo valor da seqüência de bits é portanto um índice do vetor do dicionário, no nível 2, que melhor representa esta parte do sinal. Recebemos o valor 23. Este vetor foi resultante de uma expansão do nó E, realizada anteriormente. Portanto, como podemos ver, o dicionário já está utilizando vetores baseados em partes da imagem já processadas.

Voltamos ao nó pai A, e como visitamos seus dois nós filhos, concatenamos os vetores correspondentes a estes nós (vetores 24 e 23) (veja Fig. 2.21), e inserimos este novo vetor no nível 3 do dicionário (vetor 13) (veja Fig. 2.22).

Um vetor correspondente à contração do nó A será gerado e inserido no dicionário, no nível 2, posição 25. Veja Fig. 2.23. As outras contrações geram vetores já existentes no dicionário, por isto não há necessidade de repeti-las.

Até este momento, o dicionário construiu uma representação mais precisa do sinal de entrada, formada pelas concatenações de todos os nós inferiores da árvore. Se um outro sinal, parecido com este, for codificado utilizando este mesmo dicionário, os novos vetores inseridos no dicionário provavelmente poderão ser utilizados. A distorção será reduzida sem que haja um aumento na taxa. Com isso haverá um ganho total na compressão do sinal.

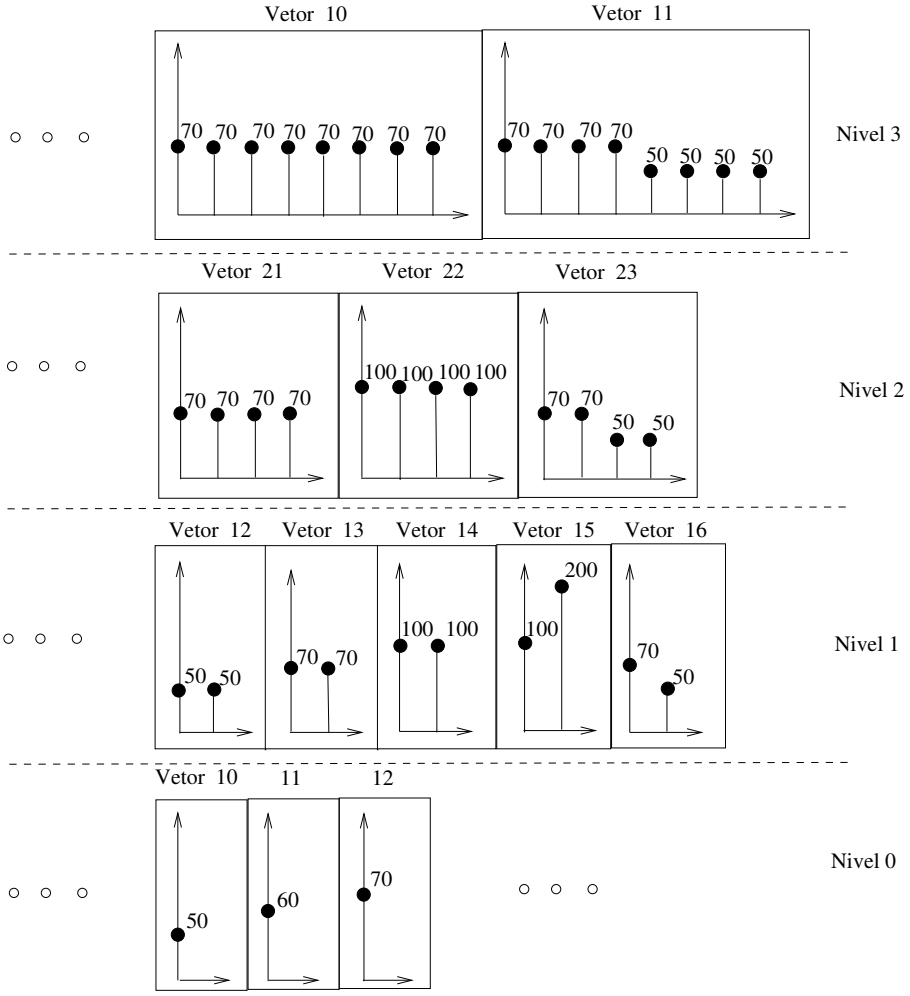


Figura 2.16: Dicionário após a inserção das expansões do vetor correspondente ao nó E. A primeira expansão resultou no vetor 23 inserido no nível 2 do dicionário, e a outra expansão resultou no vetor 11 inserido no nível 3 do dicionário.

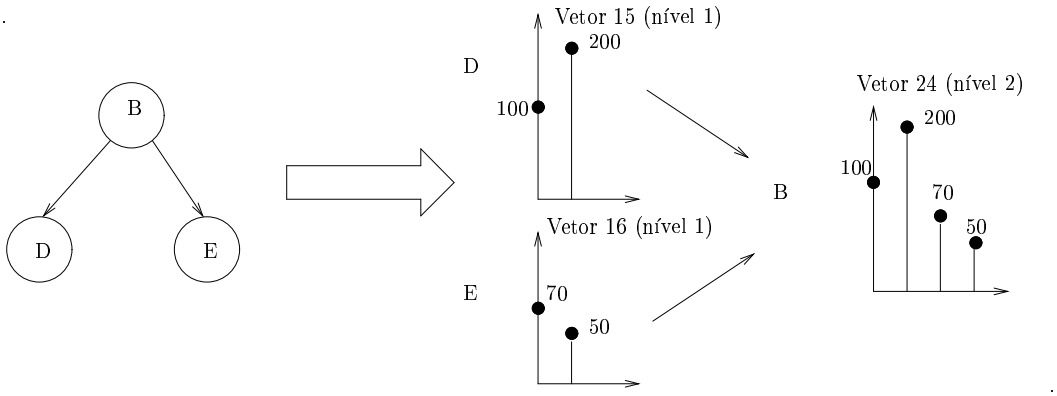


Figura 2.17: Concatenação de dois nós (D e E) para inserir um novo vetor no nível 2 do dicionário.

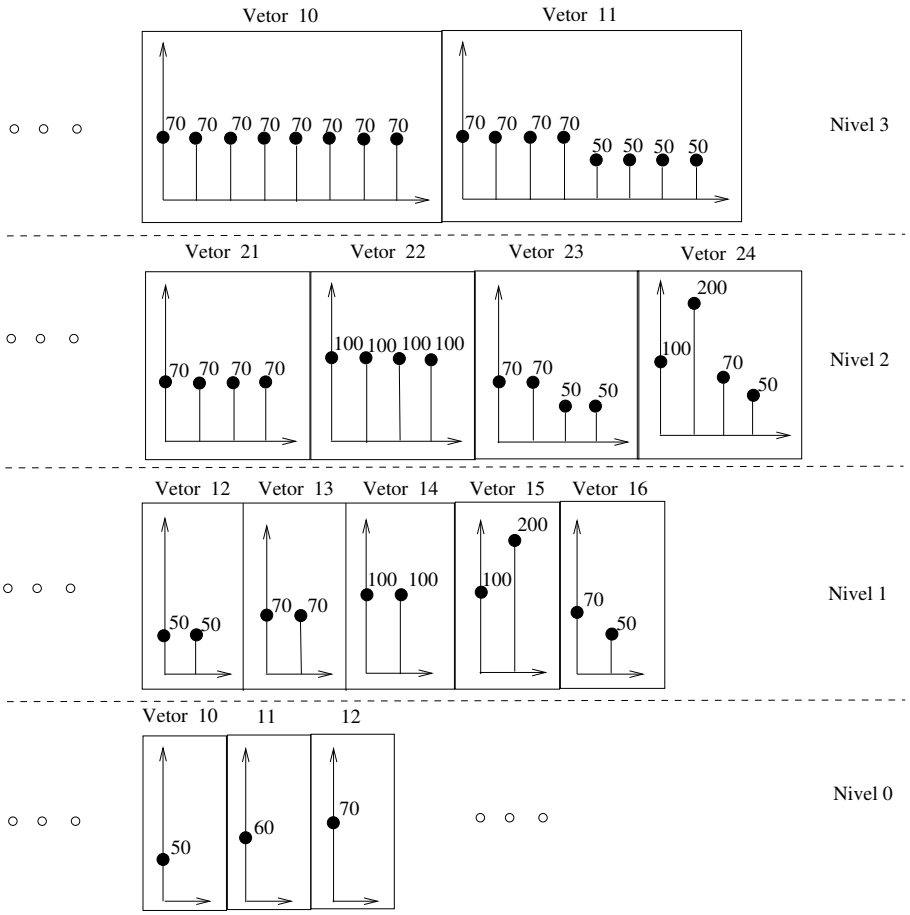


Figura 2.18: Dicionário após a inserção do vetor correspondente ao nó B, resultante da concatenação dos vetores correspondentes aos nós D e E. Este vetor foi inserido na posição 24, no nível 2 do dicionário.

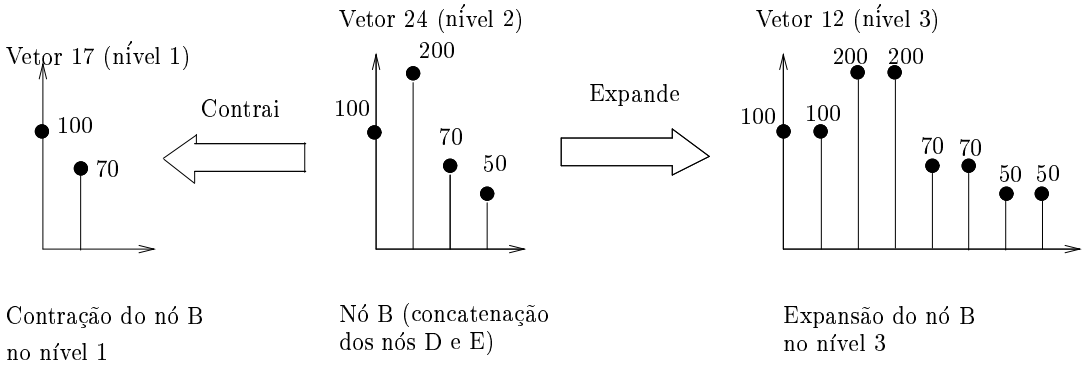


Figura 2.19: Expansão e contração do vetor correspondente ao nó B, resultante da concatenação dos vetores correspondentes aos nós D e E, a serem inseridas no dicionário.

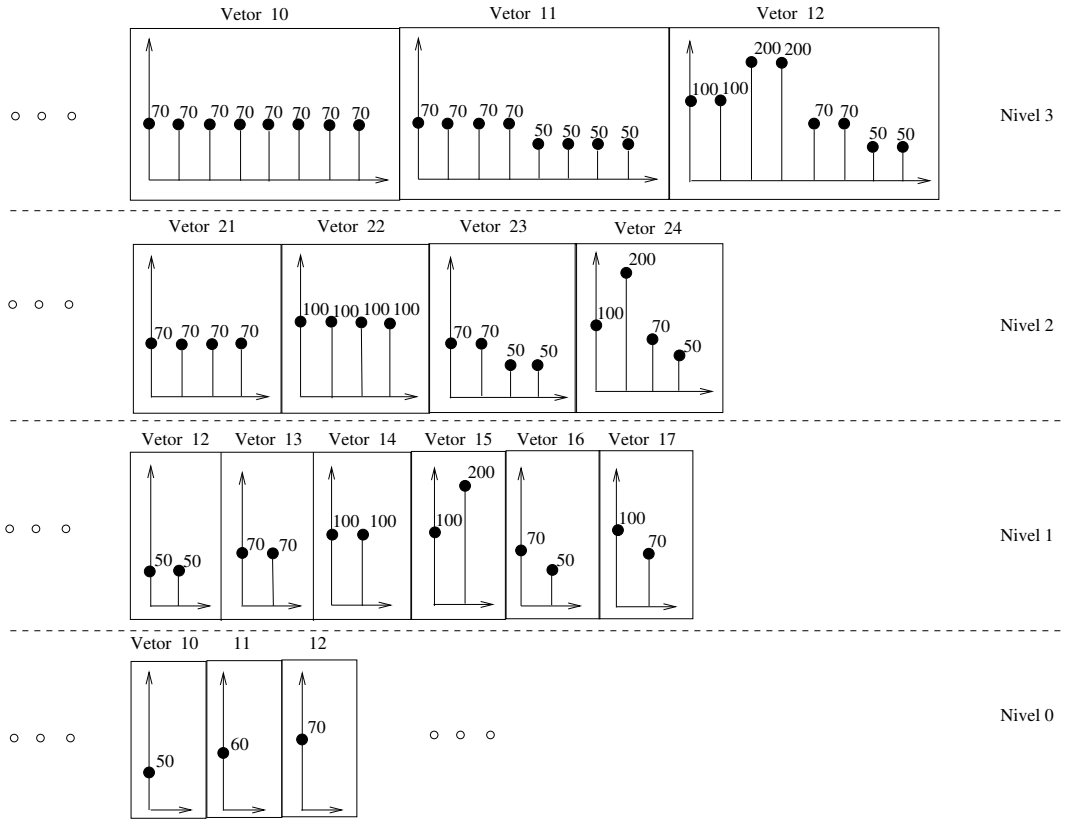


Figura 2.20: Dicionário após as inserções da contração (vetor 17, nível 1) e da expansão (vetor 12, nível 3) do vetor 24, do nível 2, correspondente ao nó B.

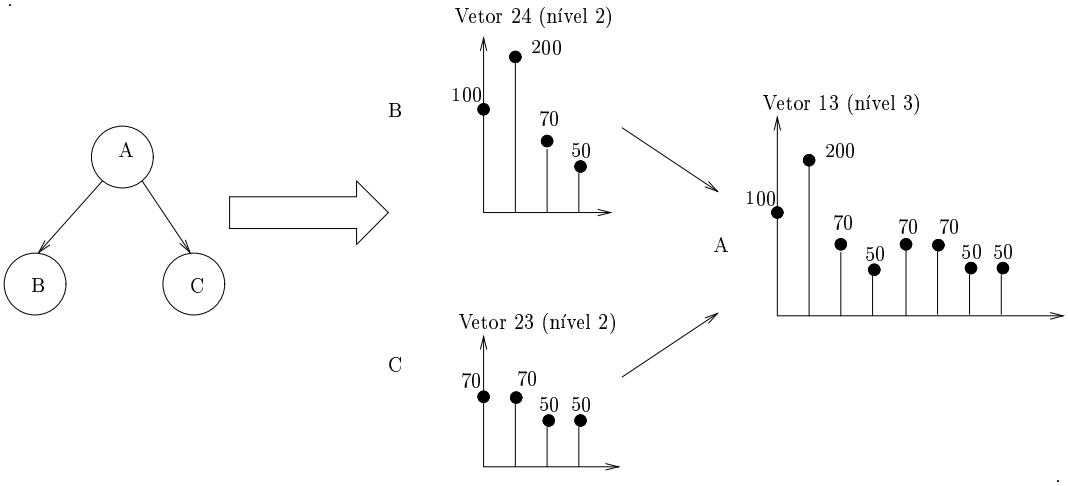


Figura 2.21: Concatenação dos vetores correspondentes aos nós B e C, gerando um novo vetor que corresponde ao nó A. Este novo vetor será inserido no nível 3 do dicionário.

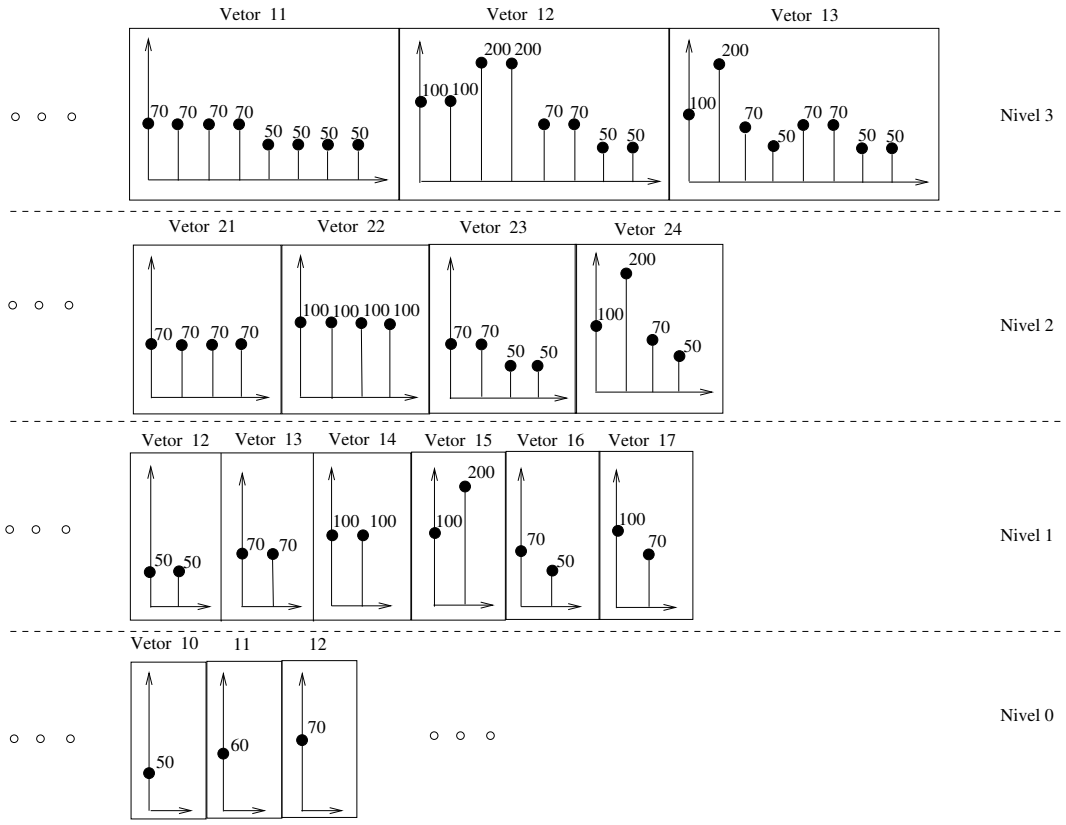


Figura 2.22: Dicionário após a inserção do vetor correspondente ao nó A, (posição 13, nível 3) resultante da concatenação dos vetores correspondentes aos nós B e C.

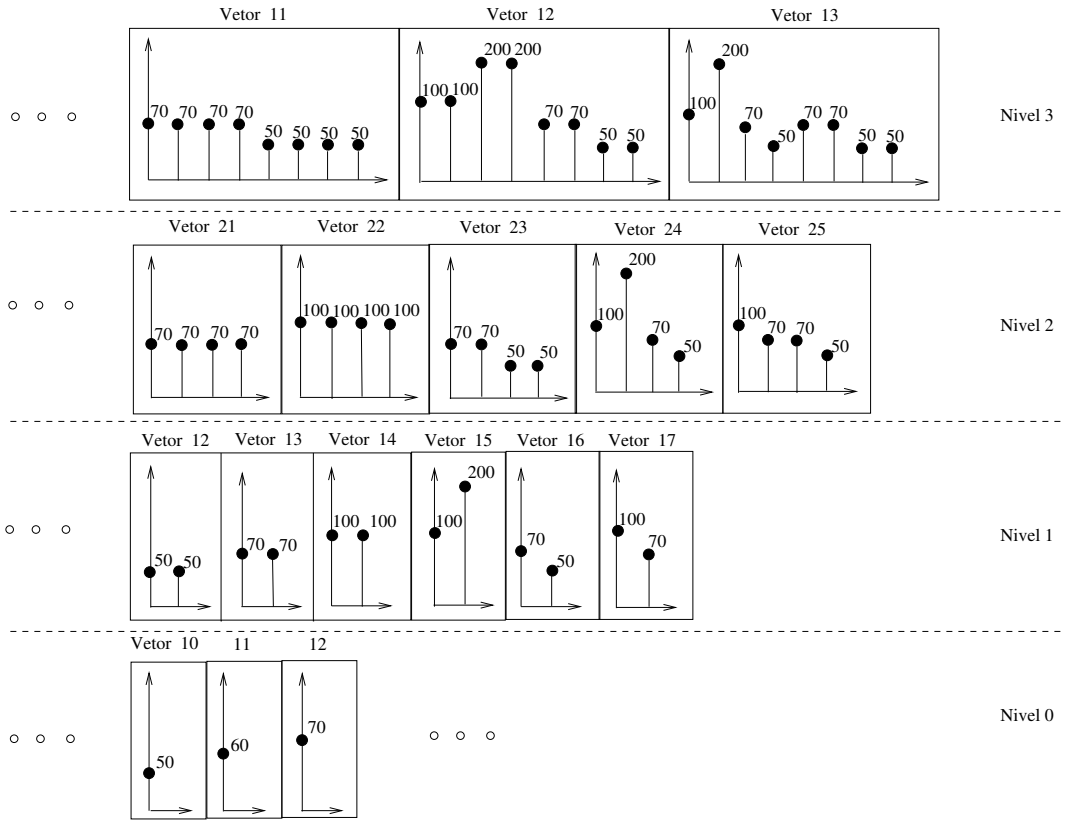


Figura 2.23: Dicionário após a inserção do vetor resultante da contração do vetor 13, nível 3 (correspondente ao nó A), inserido na posição 25, nível 2).

2.2.2 Descrição do MMP para representação de um sinal bidimensional

O algoritmo MMP possui basicamente quatro estágios:

- Cálculo dos custos;
- Obtenção da árvore ótima;
- Codificação dos *flags* e índices;
- Atualização do dicionário.

Inicialmente a imagem é dividida em blocos de tamanho NxN. Cada bloco será processado pelo MMP seqüencialmente, da esquerda para a direita. Chamamos de *dicionário* uma matriz de blocos, onde as linhas representam o nível do dicionário, e as colunas o elemento daquele nível do dicionário. Por exemplo, $E[6][2]$ se refere ao elemento de índice 2 do nível 6 do dicionário. Os elementos de cada nível do dicionário serão usados para representar os blocos da imagem. O objetivo é que o dicionário contenha elementos os mais próximos possíveis dos blocos da imagem, pois com isso a distorção entre a imagem original e a imagem reconstruída será menor. O MMP basicamente encontra os elementos que melhor representam o bloco da imagem que está sendo processado e envia os índices (codificados) destes elementos para o decodificador. Como não é necessário enviar o dicionário, pois o decodificador é capaz de construí-lo tal qual o codificador o faz, este pode crescer bastante, o suficiente para conter elementos cada vez mais representativos.

O dicionário tem vários níveis. O número de níveis está relacionado com as dimensões do bloco e com o número de divisões possíveis que este bloco pode ter de acordo com uma árvore binária.

Veja na Fig. 2.24 a estrutura em árvore de um bloco 8x8. No nível mais alto (nível 6), temos um elemento do tamanho do bloco. Este elemento pode ser dividido em dois elementos menores mostrados no nível logo abaixo (nível 5). A divisão do bloco pode ser horizontal ou vertical. Nesta figura temos divisão vertical. As divisões podem ir até o nível 0, onde cada elemento é do tamanho de um pixel. No MMP, dividir a árvore pode reduzir a distorção porque pode-se achar uma melhor representação para o bloco, mas quando isto acontece, aumenta-se a taxa, pois mais

índices e flags serão enviados. Assim, o algoritmo consiste em representar o bloco por uma árvore ótima que possua o melhor compromisso entre taxa e distorção (critério RD).

Para blocos $N \times N$, o número de níveis N_NIVEIS é dado por $1+2\log_2(N)$. Assim, para um tamanho de bloco 8x8 o número de níveis é 7 (nível 0 a nível 6). O número de pixels dos elementos do maior nível é $2^{N_NIVEIS-1}$, que é igual ao número de pixels do bloco da imagem. Para blocos 8x8 o número de pixels do elemento do nível 6 é igual a 64. O número de elementos no nível 0 é 64, com cada elemento sendo do tamanho de 1 pixel.

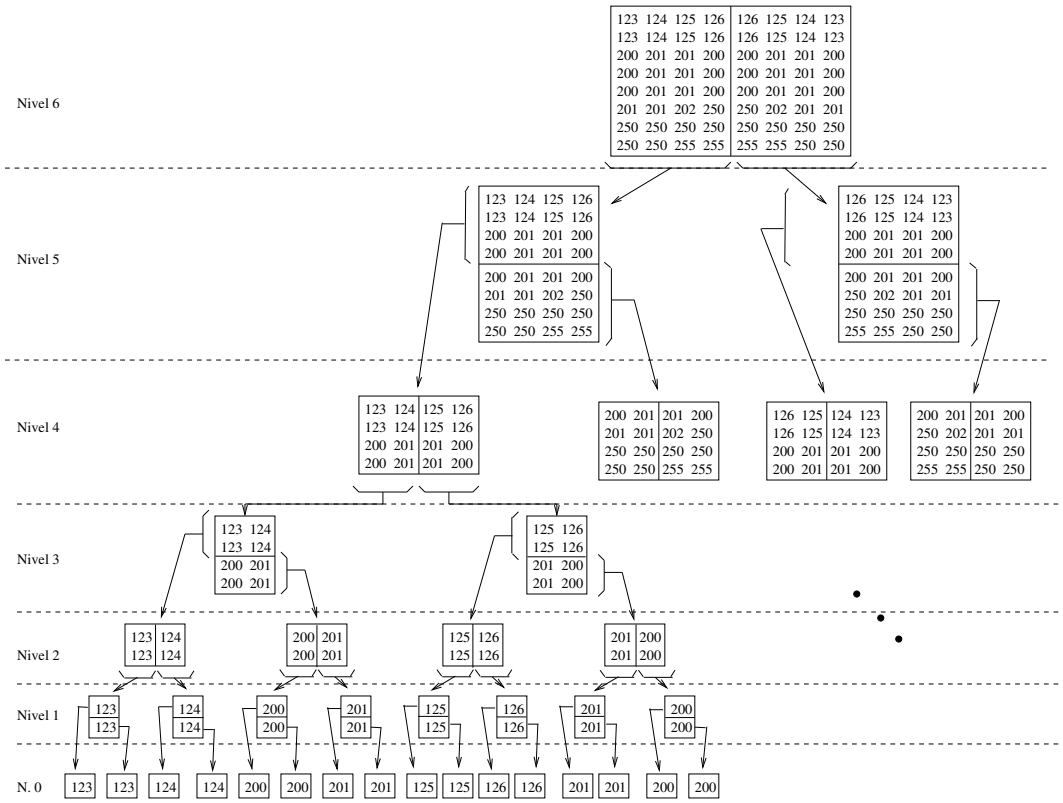


Figura 2.24: Diagrama de uma árvore binária que representa um bloco 8x8 e suas possíveis divisões. A divisão dos blocos é vertical.

Antes de começar o processamento do bloco, o MMP monta um dicionário com elementos iniciais cujos valores são baseados nos valores de luminância dos pixels da imagem. Por exemplo, seja *minpixel* o menor valor de luminância da imagem, e *maxpixel* o maior valor. Seja *n_elem_inicial* o número de elementos iniciais do dicionário. Os valores dos pixels dos elementos iniciais do dicionário

serão distribuídos de *passo* em *passo*, começando de *minpixel* até *maxpixel*. O *passo* é dado por:

$$passo = \frac{maxpixel - minpixel}{n_elem_inicial}$$

Se *n_elemen_inicial* for igual a 256, o *passo* será igual a 1. Para este caso especial, o dicionário inicial terá elementos que contêm todos os pixels presentes na imagem. (Obs.: as imagens usadas nesta tese contêm valores de luminância variando entre 0 e 255).

A Fig. 2.25 e a Fig. 2.26 mostram os elementos E[6][0] e E[6][1] do dicionario inicial, quando $minpixel = 0$, $maxpixel = 255$ e $n_elemen_nicial = 64$ ($passo = 4$), para blocos 8x8.

E[6][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	0

Figura 2.25: Elemento E[6][0] (nível 6, índice 0) do dicionário.

E[6][1]

4	4	4	4	4	4	4	4
4	4	4	4	4	4	4	4
4	4	4	4	4	4	4	4
4	4	4	4	4	4	4	4
4	4	4	4	4	4	4	4
4	4	4	4	4	4	4	4
4	4	4	4	4	4	4	4
4	4	4	4	4	4	4	4

Figura 2.26: Elemento E[6][1] (nível 6, índice 1) do dicionário.

Dado um bloco da imagem de tamanho NxN, o algoritmo procura dentre os elementos do dicionário de nível mais alto, aquele que dá o melhor casamento segundo o critério de otimização RD. Neste critério tenta-se encontrar o menor custo

J dado por:

$$J = D + \lambda R$$

onde D é a distorção, R é a taxa de bits por pixel, e λ é uma constante (multiplicador de Lagrange).

A distorção é obtida calculando-se o erro médio quadrático entre o bloco da imagem e um dado elemento do dicionário. A taxa é o número de bits por pixel gastos para transmitir o índice deste elemento do dicionário, codificado com codificador aritmético. Para o cálculo da taxa também consideram-se os bits gastos para enviar um *flag* que indica se houve divisão ou não na árvore.

O início da busca se dá no nível mais alto do dicionário. Obtém-se neste nível qual o índice do elemento que gerou o menor custo J. O algoritmo desce um nível na árvore, dividindo o bloco original em duas partes. A divisão pode ser horizontal ou vertical.

A busca agora é feita no dicionário de nível abaixo do maior nível, onde o número de pixels de cada elemento é metade do número de pixels do bloco original. O cálculo do custo J é realizado da mesma forma que no nível anterior. O mesmo é realizado para a outra metade do bloco.

Uma vez calculados todos os custos da árvore, para um determinado bloco da imagem, passamos então ao segundo processo do algoritmo, que é a podagem da árvore, ou seja, a obtenção da árvore ótima que melhor representa o bloco.

A podagem é feita da seguinte forma (veja a Fig. 2.27):

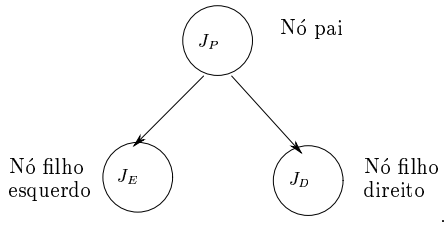


Figura 2.27: Diagrama representando um nó de uma árvore binária, e suas possíveis divisões (nós filhos). Os custos associados ao nó pai, e aos nós filhos esquerdo e direito são, respectivamente, J_P, J_E e J_D.

Seja J_P o custo do nó pai. Seja J_E o custo do nó filho à esquerda da árvore, e J_D o custo do nó filho à direita da árvore. Compara-se J_P com $J_E + J_D$. Se a

soma dos custos dos nós filhos for menor do que o custo do nó pai, a árvore não será podada. Para este nó pai é atribuído $flag = 1$, indicando que, para este nó, o melhor é que ele seja dividido. Isto significa que representar o bloco da imagem por dois blocos menores separadamente é melhor do que representá-lo por um bloco grande. Substitui-se então na árvore, o custo do nó pai pela soma dos custos dos nós filhos.

Caso contrário, se a soma dos custos dos nós filhos for maior ou igual ao custo do nó pai, a árvore será podada, ou seja, não é necessário dividir o nó, pois o algoritmo já encontrou uma melhor representação para o bloco da imagem usando um elemento maior do dicionário. Um $flag = 0$ é então atribuído ao nó pai da árvore, e o índice do elemento do dicionário que gerou este custo é guardado.

O processo de podagem continua até que toda a árvore seja percorrida. Um exemplo de uma árvore ótima resultante do processo de segmentação é mostrada na Fig. 2.28.

Após a obtenção da árvore ótima, inicia-se o processo de codificação propriamente dito. Percorre-se a árvore ótima, começando pelo nível mais alto (nó raiz) até o nível mais baixo (nós terminais ou folhas), percorrendo primeiro o nó esquerdo e depois o direito. Sempre que o algoritmo encontra um $flag = 1$, ele envia este *flag* (codificado com o codificador aritmético) para o decodificador, e desce um nível na árvore. Ao encontrar um nó terminal na árvore ótima, ele envia para o decodificador um $flag = 0$ e também o índice do elemento do dicionário que gerou aquele nó. Os índices são codificados com um codificador aritmético, usando um modelo para cada nível do dicionário.

2.2.3 A atualização do dicionário

O dicionário do MMP é adaptativo. Começa com os elementos iniciais e vai incorporando elementos obtidos durante o processo. O dicionário vai "aprendendo" com os padrões que ocorreram em blocos já processados da imagem.

Na árvore ótima, sempre que se chega a dois nós terminais, estes são concatenados e inseridos no dicionário num nível imediatamente superior ao dos nós terminais (veja a Fig.2.29).

Este mesmo elemento concatenado é expandido e inserido em níveis supe-

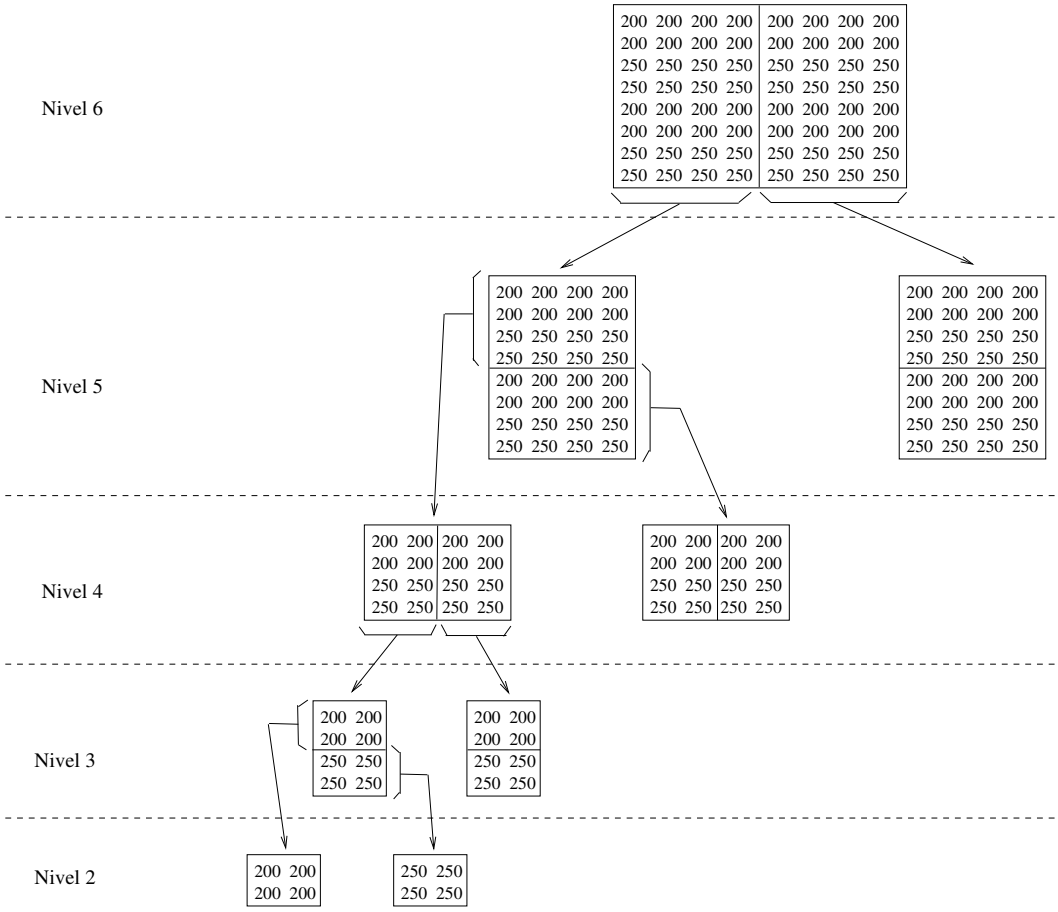


Figura 2.28: Árvore ótima após o processo de segmentação (podagem da árvore). Esta árvore é a que melhor representa um determinado bloco da imagem.

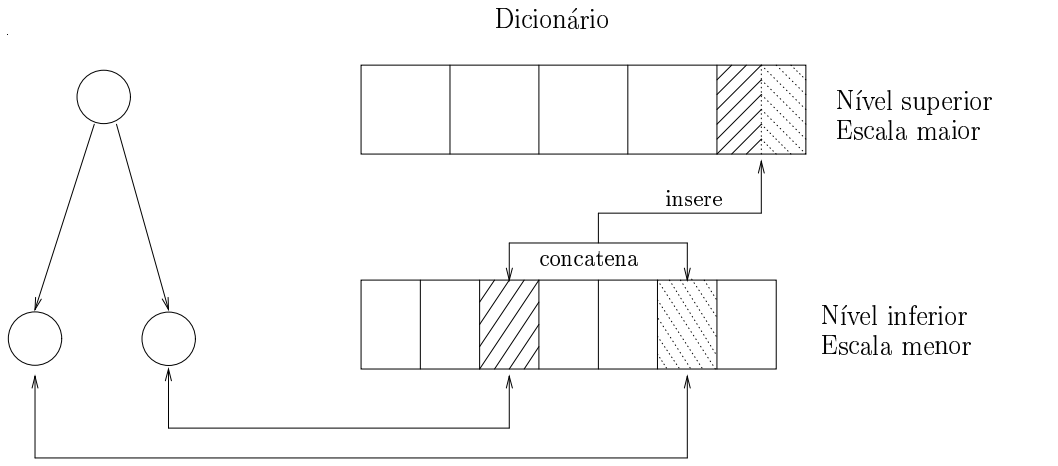


Figura 2.29: Atualização do dicionário.

riores do dicionário. A expansão de um bloco pode ser feita de várias maneiras, uma delas é simplesmente repetir os pixels daquele bloco. Outra maneira é fazer uma média de dois pixels vizinhos para obter um pixel intermediário (ver Fig. 2.30). O bloco concatenado também é contraído e inserido em níveis mais baixos do dicionário. A contração pode ser feita simplesmente eliminando pixels intercalados, ou substituindo dois pixels vizinhos por um pixel que seja a média dos dois (ver Fig. 2.30).

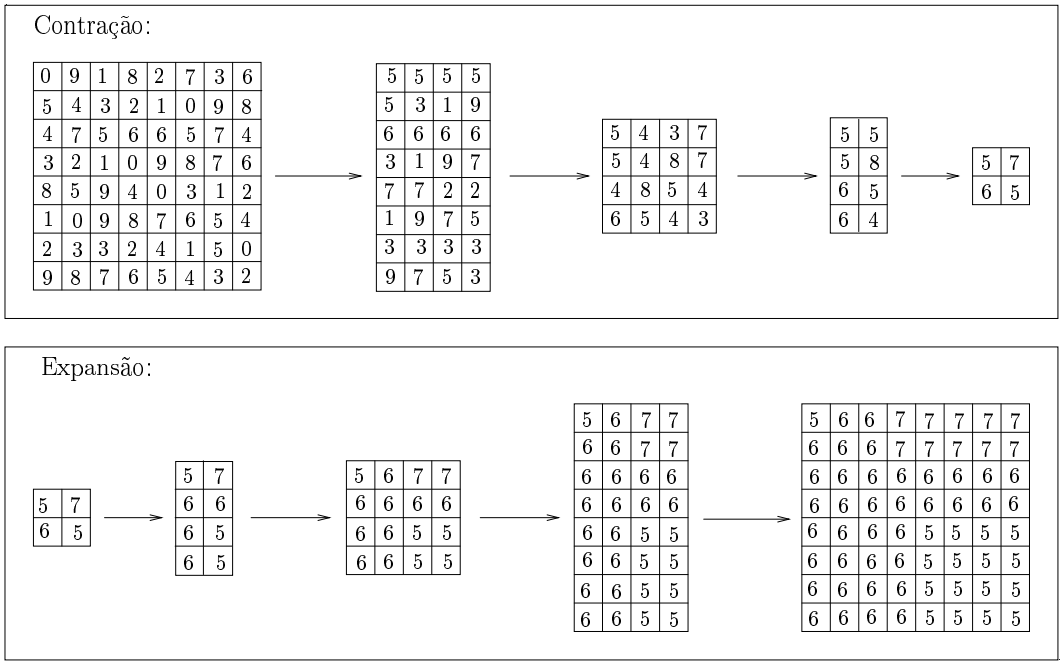


Figura 2.30: Exemplo de contrações e expansões de um bloco da imagem.

As concatenações, expansões e contrações são essenciais para um aprendizado mais rápido do dicionário. A tendência é que, no decorrer do processamento da imagem, os níveis mais altos do dicionário contenham elementos cada vez mais parecidos com os blocos que ocorreram na imagem, e assim menos divisões serão necessárias na árvore. Conseqüentemente, menos índices e menos *flags* serão gerados, o que diminui a taxa e melhora a compressão.

O tamanho máximo de elementos no dicionário, para cada nível, é 32760 (valor limitado por causa do número máximo de simbolos do codificador aritmético). Quando as inserções no dicionário atingem este limite, inicia-se o processo de "podagem" do dicionário, onde o elemento que foi acessado há mais tempo é substituido pelo novo elemento. Com isso tenta-se priorizar os padrões aprendidos mais recentemente

em detrimento daqueles que ocorreram há mais tempo na imagem. Devido a esta "podagem" é que se dá o "esquecimento" do dicionário.

2.2.4 O decodificador

O decodificador começa com o mesmo dicionário inicial do codificador. Para isso, ele só precisa receber inicialmente os valores de *minpixel*, *maxpixel* e o *passo*. O decodificador começa o processamento do bloco pelo nó raiz da árvore, tal qual o codificador. Ao receber os *flags* e os índices da árvore ótima obtida no codificador, ele vai compondo a imagem reconstruída a partir de elementos do seu dicionário. O processo de atualização do dicionário é feito da mesma maneira que no codificador. Quando o decodificador recebe dois nós terminais, ele concatena estes nós e inclui o nó resultante, bem como suas expansões e contrações no seu dicionário. A decodificação é bem mais rápida do que a codificação, pois as etapas de obtenção dos custos e podagem da árvore não são necessárias.

2.3 Conclusão

Neste capítulo fizemos uma descrição detalhada do MMP, utilizando como exemplo inicial um sinal unidimensional, para facilitar a compreensão do método, e depois estendemos a descrição do funcionamento do MMP aplicado a uma imagem.

A descrição deste algoritmo se fez necessária por este se tratar de uma proposta nova de codificação. O MMP apresenta algumas vantagens em relação aos métodos já conhecidos de codificação de imagens, como os métodos baseados no padrão JPEG. Dentre elas podemos citar o uso automático de blocos de tamanho variável para representar a imagem, a flexibilidade em definir o tamanho do bloco inicial (8x8, 16x16, 32x32, 64x64, etc.), e o fato de sempre encontrar a melhor representação para a imagem original, independentemente da precisão da estimação de movimento. O MMP apresenta resultados de compressão comparáveis aos resultados do algoritmo SPIHT [21] para uma larga classe de imagens; mais especificamente, o seu desempenho é ligeiramente inferior nos casos de imagens suaves, mas supera no caso de imagens que incluem textos e/ou gráficos (imagens mistas) [20].

Capítulo 3

Avaliação do desempenho do MMP aplicado a imagens estéreo

3.1 Introdução

No capítulo anterior fizemos uma descrição detalhada sobre o funcionamento do MMP aplicado a um sinal unidimensional, e depois à uma imagem.

O MMP é uma nova classe de codificadores de imagens, diferente de todas as propostas de codificadores já apresentadas. Os testes realizados com o MMP até agora só levaram em consideração imagens monoculares, isto é, a proposta inicial do MMP só explora a similaridade existente dentro de uma mesma imagem. Pelo menos por enquanto ainda não foi verificada a eficiência do MMP na exploração das similaridades existentes entre duas ou mais imagens, como uma seqüência de vídeo, ou imagens estéreo.

Neste capítulo iremos aplicar o MMP a algumas imagens estéreo para verificarmos sua eficácia. Fizemos uma seqüência de testes comparando o seu desempenho com um outro método proposto exclusivamente para codificar imagens estéreo. O algoritmo utilizado aqui para efeito de comparação com o MMP foi o *optimal dependent quantisation* (ODQ) [11]. Este algoritmo foi escolhido porque ele representa o estado da arte em codificação de imagens estéreo. O ODQ faz a codificação de imagens estéreo através da quantização ótima de cada bloco da imagem. Uma descrição suscinta do funcionamento deste algoritmo é dada neste capítulo, na Seção 3.2, e uma descrição mais detalhada pode ser encontrada no Apêndice A.

Existem outros métodos de codificação estéreo, citados a seguir, que não foram usados aqui para efeito de comparação com o MMP porque não forneciam resultados de compressão melhores do que o ODQ, ou porque abordavam aspectos muito diferentes dos do MMP, como por exemplo, o uso de esquemas de codificação baseados em objetos, que tornariam muito difícil o processo de comparação.

Em [30], um estimador de disparidades baseado em blocos é usado em conjunto com um esquema de compressão estéreo. Um sistema de codificação é proposto, utilizando a característica de escalabilidade temporal do MPEG-2. Também se propõe um esquema de quantização adaptativa do par estéreo, baseado no sistema de percepção visual humano. Vários esquemas de codificação [31] baseados no padrão MPEG-2 usam predição com movimento compensado acoplado com DCT sempre que isto for vantajoso para a codificação da imagem residual. Um esquema de codificação baseado em objetos é proposto em [32] para a codificação de seqüências de imagens estereoscópicas usando dados de movimento e disparidade. E mais recentemente, utilizando o critério de taxa e distorção, temos os codificadores propostos por [33], baseado no padrão MPEG-2, e [34], baseado em wavelets.

O objetivo principal do conjunto de testes realizados e descritos neste capítulo é obter uma direção na qual possamos prosseguir no desenvolvimento desta tese, no sentido de melhorar o desempenho do MMP para codificar imagens estéreo.

3.2 Descrição suscinta do algoritmo *Optimal Dependent Quantisation* – ODQ

Faremos agora uma descrição suscinta do ODQ, visando adiantar a compreensão por parte do leitor a respeito do seu funcionamento. Para aqueles que desejarem maiores detalhes, temos uma descrição mais completa no Apêndice. A.

O algoritmo ODQ, proposto por Woontack Woo [11], é um método para codificação de imagens estéreo, baseado no JPEG, que tem como principal característica a quantização ótima de cada bloco da imagem. Esta quantização é chamada de "dependente" porque a escolha dos quantizadores ótimos da imagem direita é atrelada à escolha dos quantizadores ótimos da imagem esquerda. A otimização dos quantizadores é feita utilizando algoritmo de Viterbi [35, 36]. Mostramos um exemplo do

algoritmo de Viterbi, também chamado de programação dinâmica, no Apêndice E. Este exemplo simples dá uma noção do funcionamento deste algoritmo, facilitando assim o entendimento da aplicação deste método utilizado no ODQ.

Numa estrutura de codificação preditiva, como no ODQ, uma imagem é selecionada como referência (I_E) e a imagem dependente ou destino ($\hat{I_D}$) é estimada a partir da imagem de referência. Similar a outros cenários de codificação preditiva, a estimação/compensação de deslocamento reduz a redundância entre as duas imagens do par estéreo. Ao invés de se codificar a imagem destino original (I_D), codifica-se o mapa de disparidades (V), obtido entre as duas imagens, e o quadro de diferenças de disparidade compensadas (*disparity compensated difference* – DCD), que é calculado pela diferença pixel a pixel entre a imagem destino original (I_D) e a imagem destino estimada ($\hat{I_D}$).

Sejam R e D a taxa e a distorção respectivamente. Para otimizar a eficiência da codificação, os bits dados têm que ser distribuídos entre as duas imagens do par estéreo, minimizando a distorção total. Distribuir os bits considerando as duas imagens conjuntas, e não cada uma independente, é chamada alocação de bits dependente. Na codificação de disparidade compensada, a imagem destino no par estéreo é substituída pelo mapa de disparidades (V) e pelo quadro DCD, e assim os bits dados são distribuídos entre a imagem de referência (I_E), o mapa V e o quadro DCD.

O custo J_1 para codificar a imagem esquerda é dado pela taxa R_1 e pela distorção D_1. R_1 e D_1 dependem dos quantizadores do conjunto Q_1 utilizados para quantizar a imagem esquerda.. O custo J_2 para codificar o quadro DCD é dado pela taxa R_2 e pela distorção D_2. R_2 e D_2, por sua vez, dependem tanto dos quantizadores do conjunto Q_2, quanto dos quantizadores do conjunto Q_1. O custo total é dado por $J = J_1 + J_2$. A otimização do custo total J é um problema de otimização com restrição que pode ser transformado em um problema sem restrição usando o método do multiplicador de Lagrange:

$J(\lambda) = J_1(Q_1) + J_2(Q_2|Q_1) = D_1(Q_1) + \lambda R_1(Q_1) + D_2(Q_2|Q_1) + \lambda R_2(Q_2|Q_1)$

onde o multiplicador de Lagrange (λ) é uma constante não negativa.

Os valores de λ utilizados para o ODQ neste capítulo foram dentro de uma faixa de 0.5 a 256. Esta faixa foi escolhida observando-se a qualidade final das

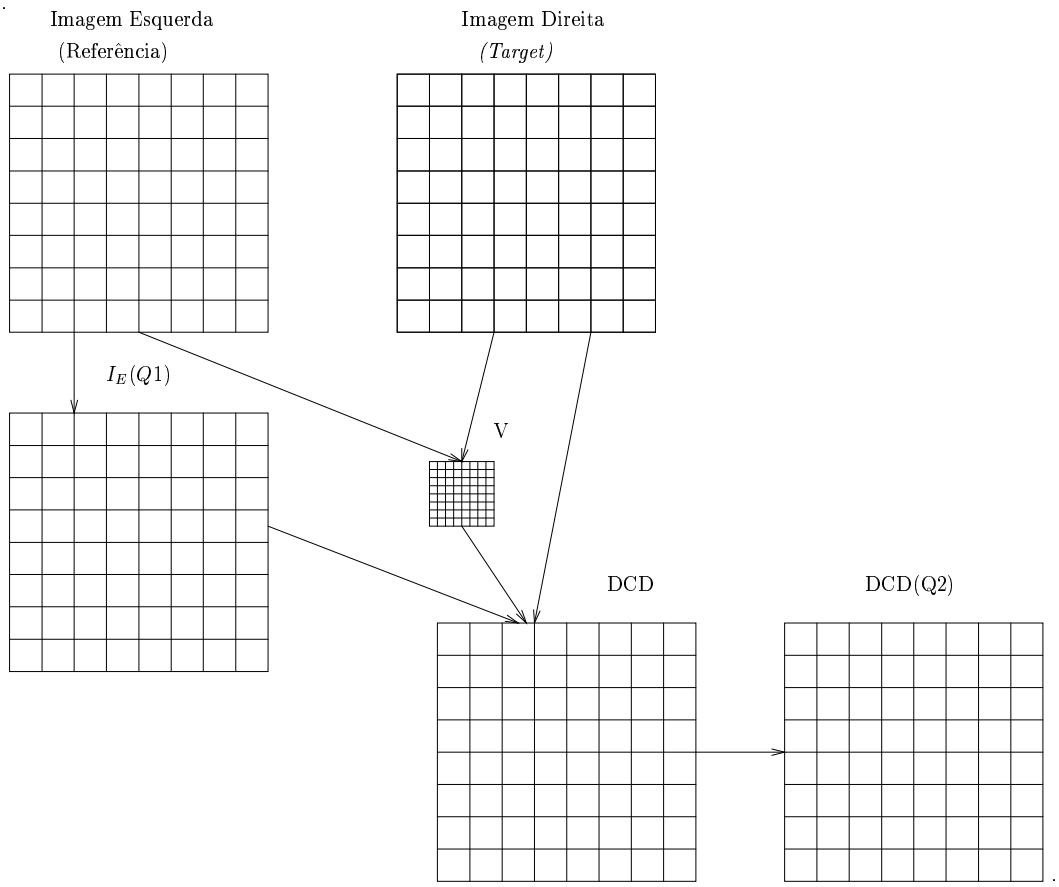


Figura 3.1: Diagrama em blocos do algoritmo ODQ. V são os vetores de disparidade por bloco, $I_E(Q1)$ é a imagem esquerda quantizada com o conjunto de quantizadores Q_1, e DCD(Q2) é a imagem DCD quantizada com o conjunto de quantizadores Q_2.

imagens reconstruídas, segundo um critério subjetivo.

A Fig. 3.1 mostra o diagrama em blocos do algoritmo ODQ.

No ODQ a imagem esquerda (imagem de referência) é codificada primeiro. O algoritmo encontra o melhor quantizador, dentro de um conjunto de possíveis quantizadores (Q_1), para cada bloco da imagem. A quantização do bloco é feita utilizando-se a matriz de quantização do JPEG e um quantizador (para refinar a quantização) escolhido dentro de um conjunto de quantizadores dado pelo usuário. A imagem esquerda quantizada com o conjunto de quantizadores (Q_1) é chamada de $I_E(Q_1)$. Logo após a quantização, o processo de codificação é similar ao utilizado pelo JPEG. Para codificar a imagem direita (*target*), calcula-se primeiro os vetores de disparidade (por bloco) entre as imagens esquerda e direita, para então obter-se o quadro DCD. Os vetores de disparidade e o quadro DCD é que são codificados, ao

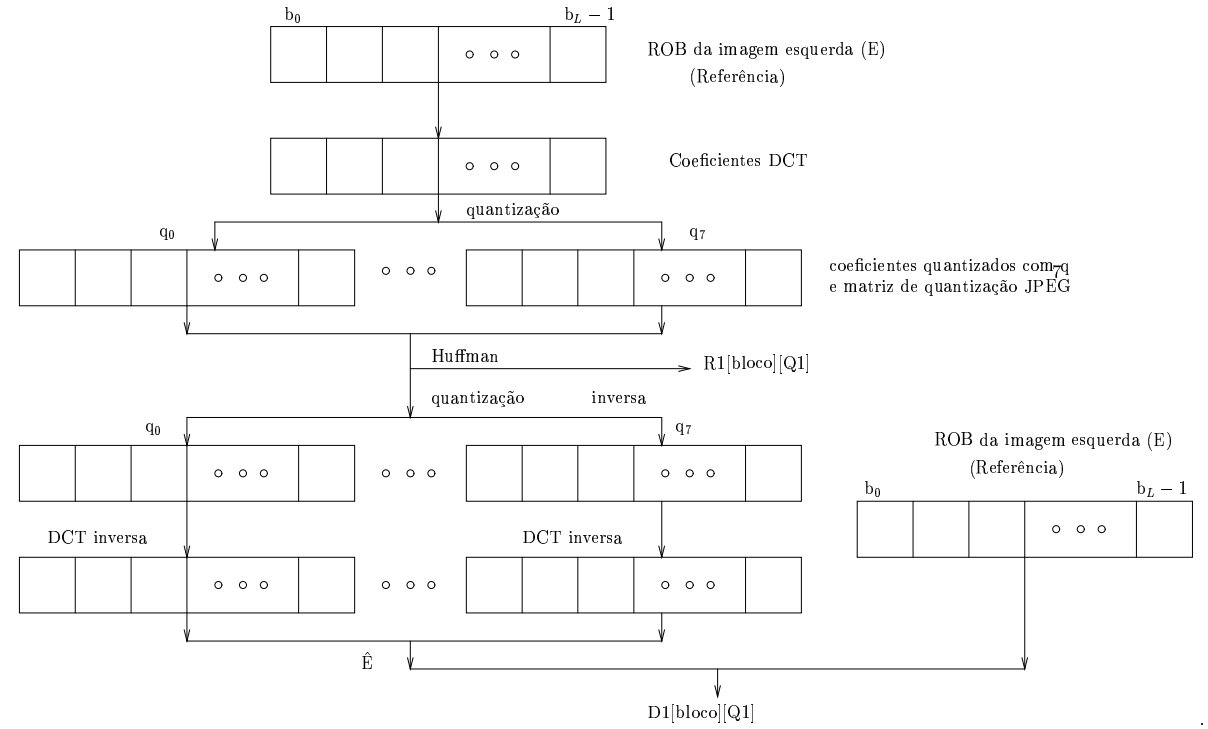


Figura 3.2: Diagrama em blocos da quantização de uma ROB da imagem esquerda com o algoritmo ODQ. Obtenção da taxa R_1 e distorção D_1.

invés da imagem direita.

O processo de encontrar os quantizadores ótimos para os blocos do frame DCD é dito dependente porque leva-se em consideração os quantizadores do conjunto Q_1 que foram utilizados para codificar a imagem esquerda. Os quantizadores encontrados para o frame DCD são escolhidos dentro de um conjunto de quantizadores Q_2 (diferente do conjunto Q_1).

A Fig. 3.2 mostra um diagrama em blocos da quantização de uma linha de blocos (*Row-of-Blocks* – ROB) da imagem esquerda com o algoritmo ODQ. No final deste processo obtemos a taxa R_1 e a distorção D_1 de cada bloco da ROB. O quantizador ótimo escolhido para um determinado bloco será aquele que gerou o menor custo J, dado por $D_1 + \lambda R_1$.

Para encontrar os quantizadores ótimos da imagem direita o processo é um pouco mais complexo. Primeiro encontramos a correspondência entre um bloco da imagem direta e um bloco da imagem esquerda, através dos vetores de disparidade. Esta correspondência pode ocorrer entre dois blocos vizinhos da imagem esquerda.

Quais os blocos da imagem esquerda que correspondem a um de-

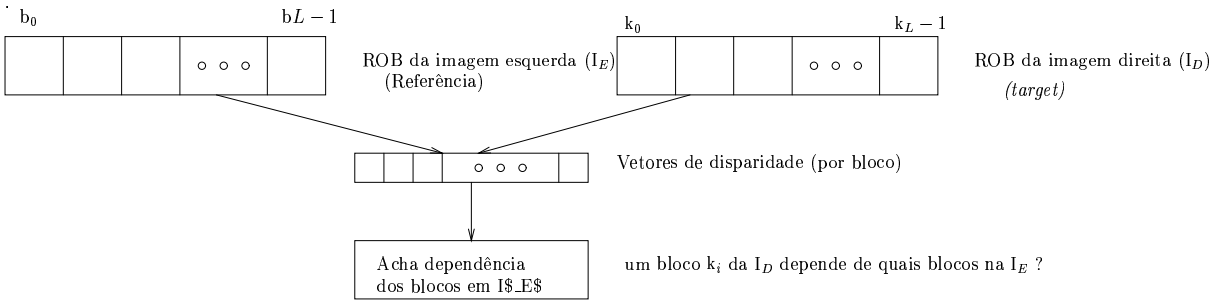


Figura 3.3: Preparação para achar dependência entre os blocos da imagem esquerda e direita.

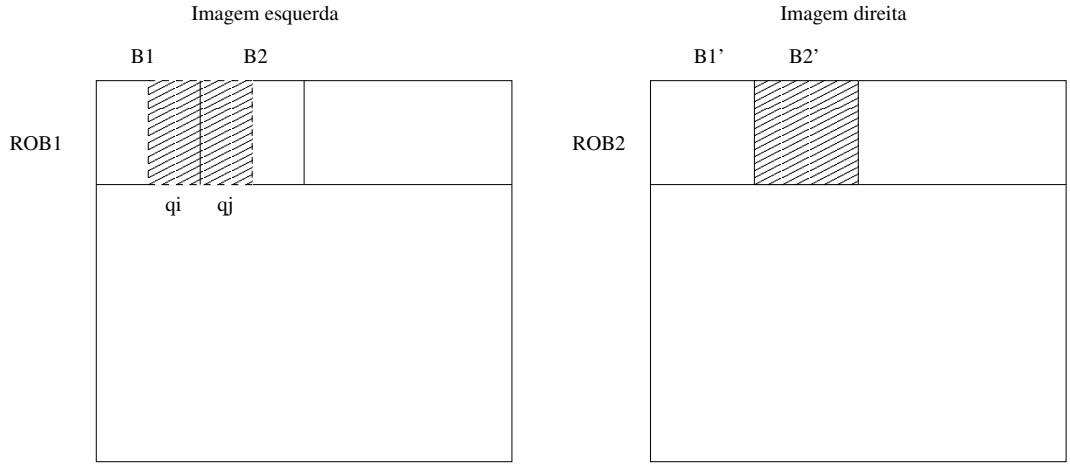


Figura 3.4: Dependência entre os blocos das imagens esquerda e direita.

terminado bloco da imagem direita?

Na Fig. 3.3 mostramos que um determinado bloco k_i da imagem direita tem uma correspondência com um bloco da imagem esquerda. A posição deste bloco da imagem esquerda é dada pelo vetor de disparidade do bloco, e pode se localizar entre dois blocos vizinhos da imagem esquerda. A Fig. 3.4 mostra isto melhor. Ainda nesta figura podemos ver que uma parte do bloco correspondente a $B2'$ pertence a um bloco $B1$ e outra parte pertence ao bloco vizinho de $B1$ ($B2$). Além disso, $B1$ pode ter sido quantizado com um quantizador q_i diferente do quantizador q_j utilizado em $B2$.

Portanto, vimos que dois blocos vizinhos da imagem esquerda podem ter sido cada um quantizado com um valor diferente. Isto influencia na obtenção do quadro DCD. Para obtermos o quadro DCD correspondente a um bloco da imagem direita temos que considerar todas as combinações possíveis de quantizadores para os dois

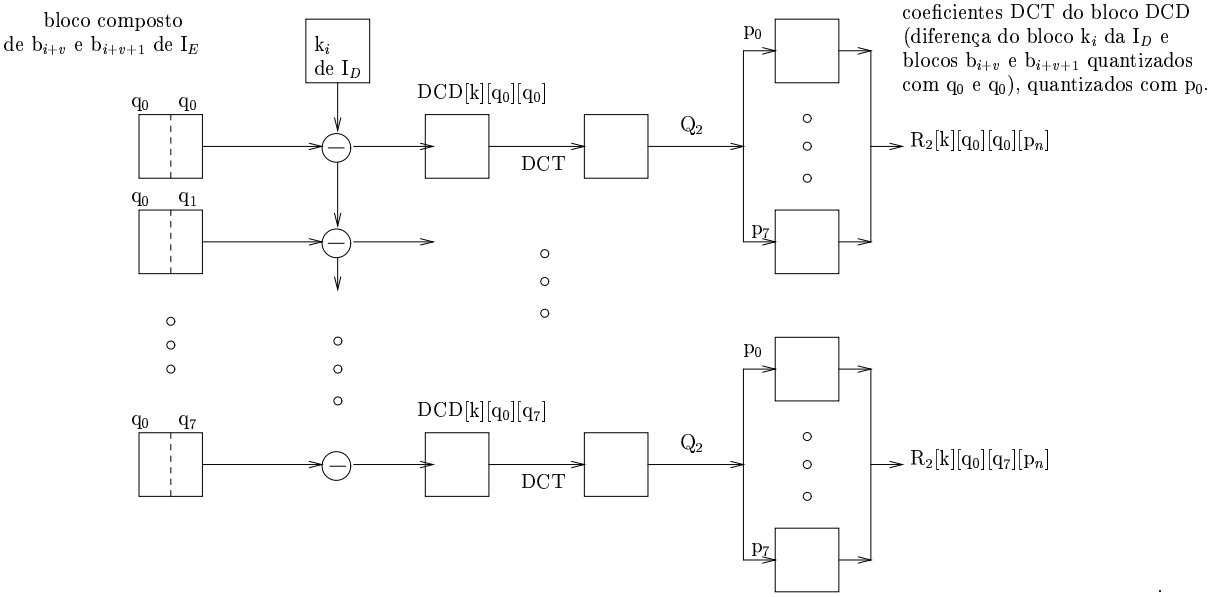


Figura 3.5: Diagrama em blocos da quantização de uma ROB da imagem direita com o algoritmo ODQ. Obtenção da taxa R_2.

blocos vizinhos dos quais depende o bloco da imagem direita. Podemos observar melhor estas combinações na Fig. 3.5. A obtenção de um determinado bloco DCD, correspondente a um bloco da imagem direita, depende de dois blocos vizinhos da imagem esquerda, que por sua vez foram quantizados com valores do conjunto Q_1. Se tivermos oito valores de quantizadores no conjunto Q_1 (q_0, q_1, q_2, ... , q_7), teremos 64 combinações para um único bloco da imagem direita. Este bloco DCD é então quantizado com um dos quantizadores de Q_2. Se o conjunto de quantizadores Q_2 também tiver oito valores diferentes (p_0, p_1, p_2, ... , p_7), totalizamos $64 \times 8 = 256$ combinações para um único bloco da imagem direita. Para cada bloco destes teremos uma taxa R_2 e uma distorção D_2 associados ao quantizador utilizado.

A Fig. 3.5 mostra a obtenção da taxa R_2, e a Fig. 3.6 mostra a obtenção da distorção D_2. Para obtermos a distorção precisamos fazer o processo de codificação inverso, para obtermos o bloco da imagem reconstruida, dadas os possíveis valores de quantização utilizados. Teremos, pois, ao final, 256 blocos DCD possíveis para um único bloco da imagem direita. Temos que escolher qual destes blocos é aquele que dará o menor custo. Isto é, escolheremos quais os quantizadores utilizados que geraram o menor custo J_2.

Como são muitas as combinações, utiliza-se programação dinâmica (algoritmo

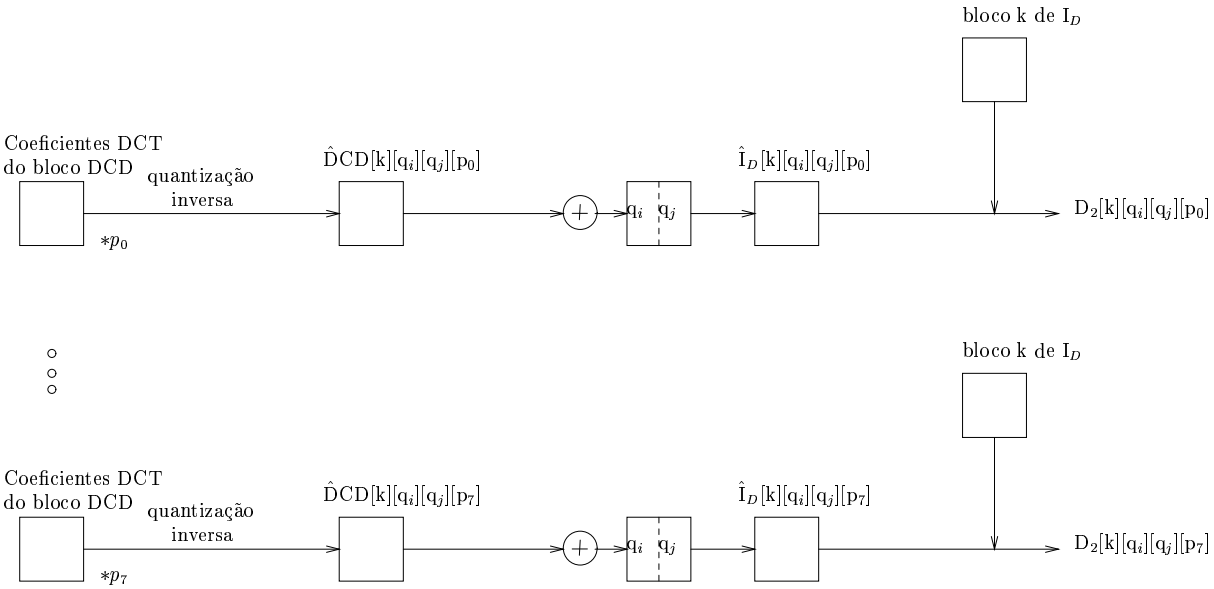


Figura 3.6: Diagrama em blocos da quantização de uma ROB da imagem direita com o algoritmo ODQ. Obtenção da distorção D_2.

de Viterbi) para otimizar os custos. No Apêndice E descrevemos um exemplo típico de programação dinâmica que irá facilitar o seu entendimento. Para aqueles que já conhecem o algoritmo de Viterbi, a Fig. 3.7 mostra sua aplicação ao problema de otimização dos custos no ODQ. Neste diagrama, os estados são os blocos vizinhos da imagem esquerda. Cada nó da treliça de Viterbi representa o custo J_1 associado ao bloco quantizado com um valor pertencente a Q_1. O custo J_2 de um ramo ligando dois blocos (b_i e b_{i+1}) da imagem esquerda vai ser o somatório dos custos de todos os blocos da imagem direita que dependem (estão associados através dos vetores de disparidade) dos blocos b_i e b_{i+1}.

A otimização de Viterbi encontra o caminho que dá o menor custo final, encontrando assim os melhores quantizadores Q_1 para a imagem esquerda, e Q_2 para a imagem direita.

A reconstrução da imagem direita, utilizando o algoritmo ODQ, pode ser feita a partir da imagem esquerda quantizada com Q_1, dos vetores de disparidade V e do frame DCD quantizado com Q_2. A Fig. 3.8 mostra o diagrama em blocos deste procedimento.

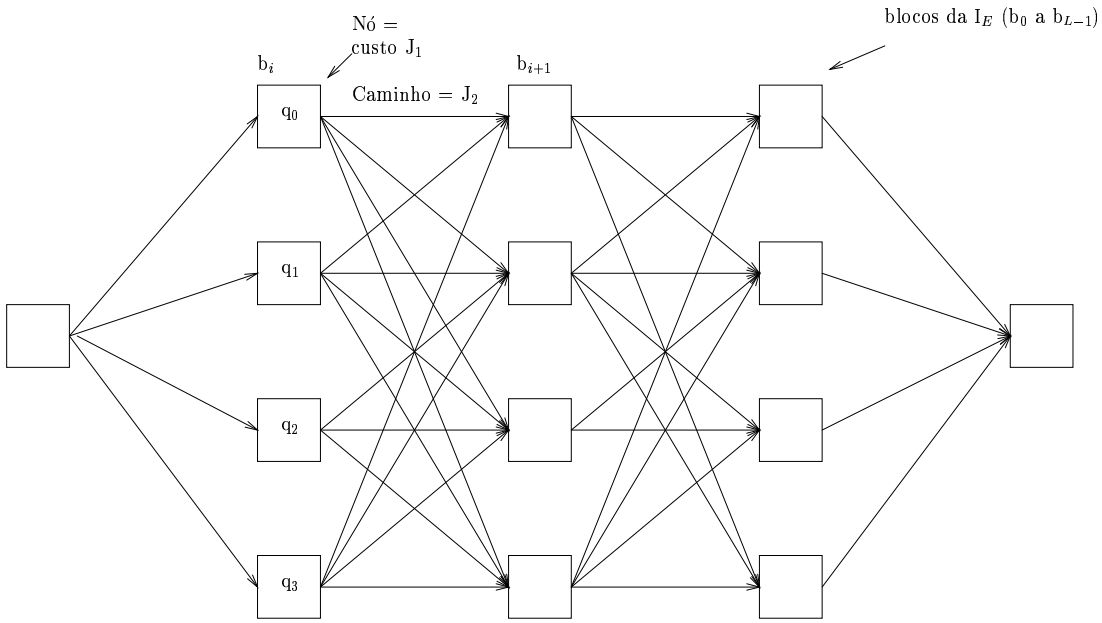


Figura 3.7: Otimização dos quantizadores do ODQ utilizando algoritmo de Viterbi.

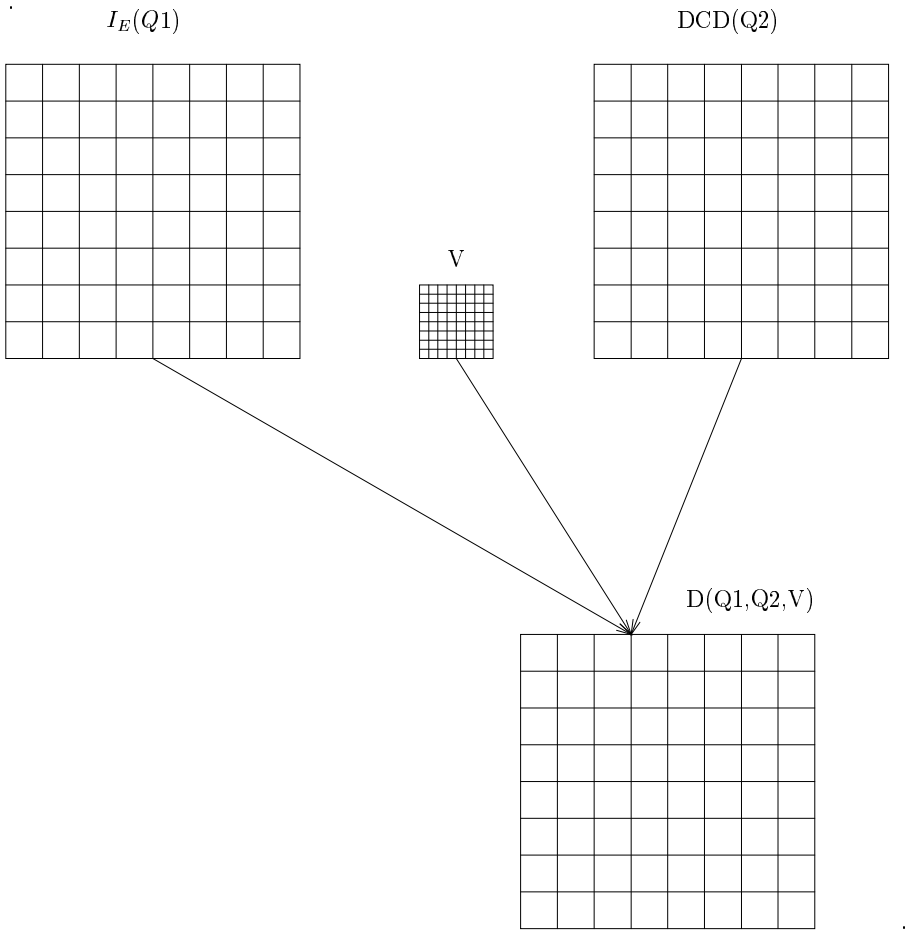


Figura 3.8: Diagrama em blocos da reconstrução da imagem direita, utilizando o algoritmo ODQ, a partir da imagem esquerda quantizada por Q_1, dos vetores de disparidade V e do frame DCD quantizado com Q_2.

3.3 Características das imagens estéreo utilizadas

As imagens utilizadas neste capítulo foram CORRIDOR, FRUIT e MAN. Estas imagens são mostradas na Fig. 3.9.

Apesar de aumentarmos consideravelmente o número de gráficos resultantes, fizemos questão de mostrar os resultados para estas três imagens porque elas possuem caraterísticas muito diferentes, tanto do ponto de vista de padrões presentes nas imagens, como do ponto de vista de características estereoscópicas.

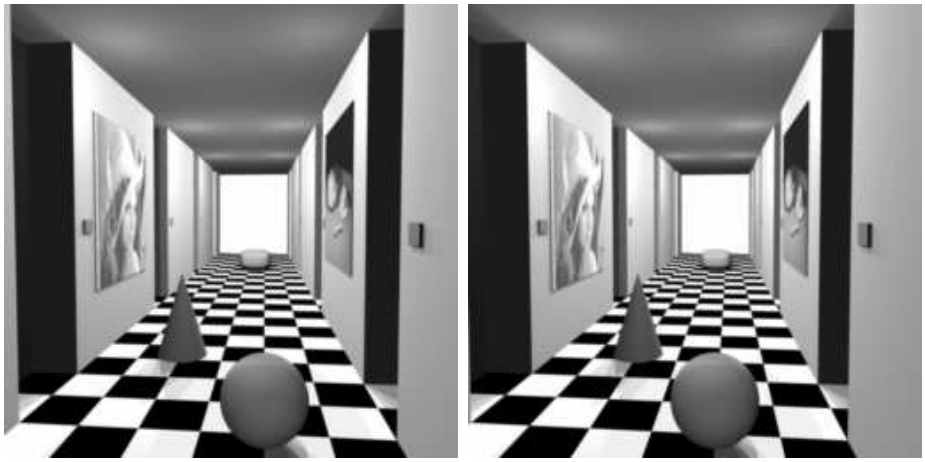
Começaremos analisando o primeiro fator, que diz respeito aos padrões presentes na imagem. Podemos observar na Fig. 3.9, as imagens CORRIDOR, FRUIT e MAN. A CORRIDOR é uma imagem sintética, que possui muita textura. A imagem FRUIT é mista, possuindo algumas regiões com baixa textura e outras regiões com alta textura. Esta imagem possui muita repetição de textura (toalha quadriculada), o que dificulta o processo de encontrar a correspondência entre as duas imagens.

A imagem MAN possui uma região de fundo (*background*) e também a região da face com baixa textura. Nestas regiões o processo de encontrar a correspondência entre as duas imagens se torna mais difícil.

Analisando o segundo fator, os valores de disparidades para a imagem CORRIDOR são pequenos [0 a 16] e puramente horizontais. Esta é uma imagem fácil de compactar. A imagem FRUIT tem valores de disparidades horizontais também nesta faixa, porém não é tão fácil de compactar porque apresenta também disparidade vertical.

A imagem MAN é extremamente difícil de compactar porque os valores de disparidades são muito grandes [25 a 128] e esta imagem possui muitas regiões de oclusão. Nestas regiões não é possível encontrar a correspondência entre alguns pontos, como uma das orelhas, que aparece numa das vistas e não aparece na outra.

Codificamos a imagem FRUIT com o ODQ e geramos a Tabela 3.1. A taxa é dada em bits por pixel (bpp) e a distorção é o erro médio quadrático medido entre a imagem original e a imagem reconstruída. A distorção é mostrada em termos de razão sinal ruído de pico - PSNR (em dB). As colunas taxa1 e PSNR1 se referem aos resultados da imagem esquerda, e taxa2 e PSNR2 se referem aos resultados da imagem direita. Taxa e PSNR referem-se à média entre a imagem esquerda e o quadro DCD. Os valores de λ utilizados foram escolhidos experimentalmente,



(a)

(b)

(c)

Figura 3.9: Pares de imagens stereo: (a) CORRIDOR (256x256), (b) FRUIT (256x256), (c) MAN (384x384).

Tabela 3.1: Valores de taxa e distorção resultantes da codificação da imagem estéreo FRUIT com o ODQ.

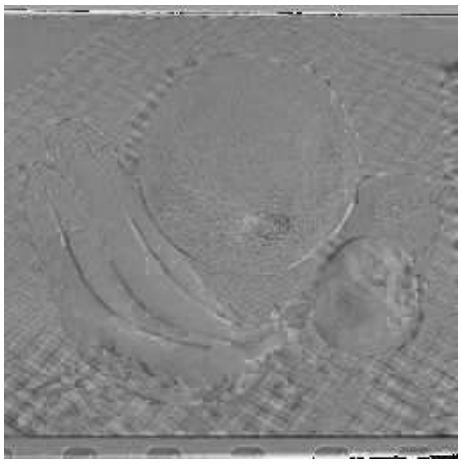
lambda	taxa1	PSNR1	taxa2	PSNR2	taxa	PSNR
0.5	3.603	47.361	4.093	47.401	3.888	47.381
1.0	3.150	46.139	3.888	46.957	3.569	46.528
2.0	2.362	43.658	3.010	44.059	2.744	43.853
4.0	1.712	41.321	2.133	41.256	1.983	41.288
8.0	1.115	38.943	1.344	38.459	1.290	38.694
16.0	0.808	37.350	0.803	36.326	0.864	36.808
32.0	0.610	35.866	0.521	34.863	0.620	35.335
64.0	0.486	34.695	0.402	33.868	0.493	34.262
128.0	0.401	33.580	0.370	33.272	0.433	33.423
256.0	0.353	32.756	0.359	32.650	0.404	32.703

segundo um critério subjetivo, observando a qualidade da imagem final reconstruída. Para os valores de λ na faixa de 0.5 a 8 verificamos que o total de bits gastos para enviar o quadro DCD desta imagem (Tabela 3.1, coluna taxa2) é um pouco maior do que o total de bits gastos (coluna taxa1) para codificar a imagem esquerda (referência).

Codificamos a imagem MAN com o ODQ e geramos a Tabela 3.2. Observamos que também para esta imagem ocorreu o fato da taxa2 ser maior do que a taxa1 para alguns valores de λ (0.5 e 1).

A Fig. 3.10 mostra o quadro DCD de cada par estéreo. O quadro DCD de cada par foi calculado com base na imagem direita (*target*) e na imagem esquerda reconstruída, que foi obtida a partir dos quantizadores ótimos Q_1 do ODQ, com lambda=0.5. O quadro DCD não levou em consideração a imagem esquerda original, e sim a imagem esquerda reconstruída porque é esta imagem que o decodificador vai ter disponível.

(a)



(b)

(c)

Figura 3.10: Quadro DCD das imagens estéreo: (a) CORRIDOR, (b) FRUIT, (c) MAN.

Tabela 3.2: Valores de taxa e distorção resultantes da codificação da imagem estéreo MAN com o ODQ.

lambda	taxa1	PSNR1	taxa2	PSNR2	taxa	PSNR
0.5	1.248	49.106	1.337	49.165	1.364	49.135
1.0	0.867	47.651	0.899	47.862	0.959	47.756
2.0	0.552	46.121	0.526	46.146	0.619	46.133
4.0	0.397	45.100	0.339	44.880	0.445	44.989
8.0	0.306	44.195	0.253	43.901	0.355	44.046
16.0	0.247	43.294	0.195	42.810	0.294	43.045
32.0	0.205	42.302	0.160	41.704	0.253	41.992
64.0	0.178	41.365	0.144	40.829	0.228	41.089
128.0	0.163	40.555	0.139	40.300	0.217	40.426
256.0	0.151	39.702	0.139	39.662	0.211	39.682

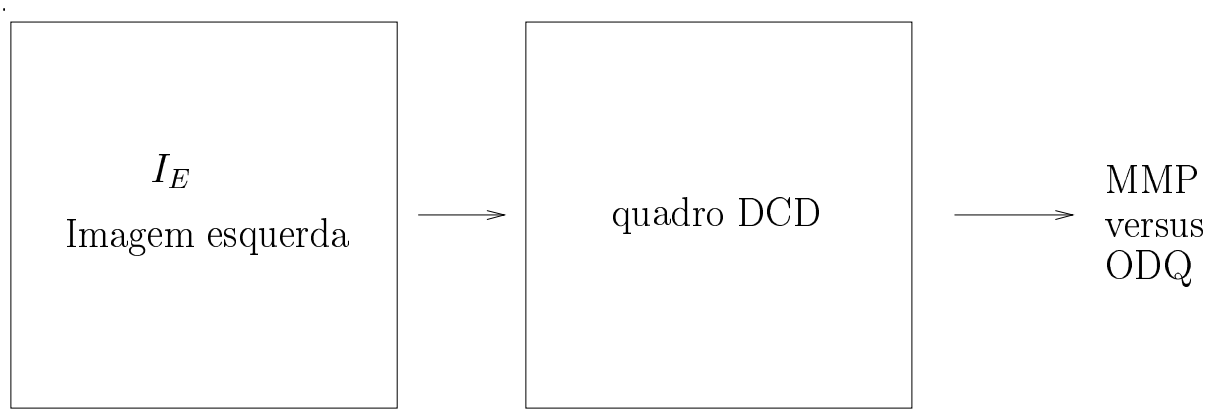


Figura 3.11: Diagrama em blocos do teste 1: Comparação entre os algoritmos ODQ e MMP aplicados à imagem esquerda e ao quadro DCD do par estéreo.

3.4 Teste 1: MMP versus ODQ aplicados à imagem esquerda e ao quadro DCD

A Fig. 3.11 mostra um diagrama em blocos deste primeiro teste, que compara os algoritmos ODQ e MMP aplicados à imagem esquerda e ao quadro DCD do par estéreo.

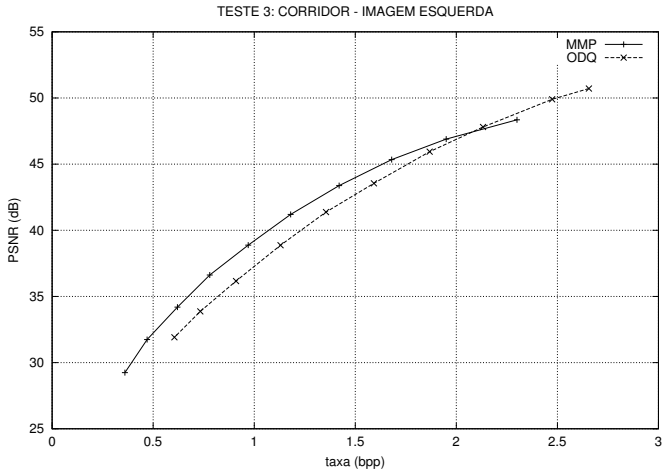
Os valores de λ utilizados tanto para o MMP quanto para o ODQ foram $(0.5, 1, 2, 4, 8, 16, 32, 64, 128, 256)$.

A Fig. 3.12 mostra os resultados do teste 1 para a imagem estéreo CORRIDOR (imagem esquerda, quadro DCD e média dos dois).

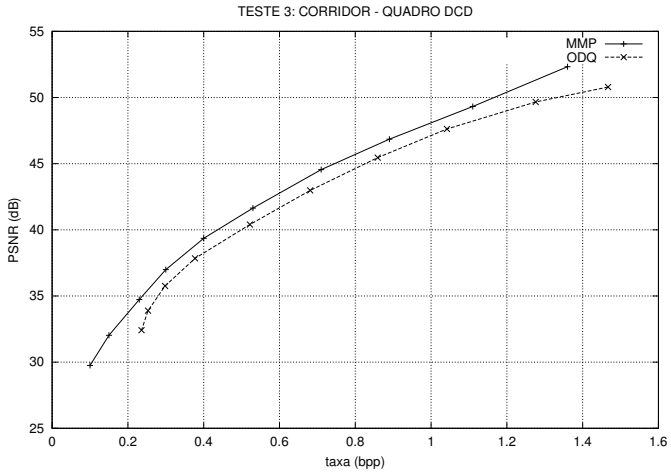
A Fig. 3.13 mostra os resultados do teste 1 para a imagem estéreo FRUIT, e a Fig. 3.14 para a imagem estéreo MAN.

O ODQ foi melhor do que o MMP para os pares de imagens que possuem disparidades maiores, como a imagem MAN. Este resultado já era esperado porque o ODQ foi proposto exatamente para codificar imagens estéreo. Este método encontra os quantizadores ótimos tanto para a imagem esquerda quanto para o quadro de diferenças DCD (que explora a similaridade obtida entre as duas imagens), e isto representa uma especificidade para a qual o MMP não foi projetado. O MMP inicialmente codifica a imagem esquerda, e o dicionário final, contendo os padrões aprendidos desta imagem não é de forma alguma aproveitável para codificar o quadro de diferenças DCD, porque este quadro possui informações muito diferentes das informações presentes na imagem esquerda (não há nenhuma similaridade entre os padrões destas duas imagens). Verificamos que obtínhamos melhores resultados de taxa e PSNR quando inicializávamos o dicionário antes de codificar o quadro DCD. Assim, o MMP codifica duas imagens independentemente (a imagem esquerda e o quadro DCD), ou seja, não usa a sua principal estratégia para conseguir comprimir a imagem, que é usar partes da imagem já processadas para representar outras partes semelhantes da imagem. Como o dicionário se adapta ao longo do processamento, a codificação do quadro DCD fica prejudicada quando esta contém padrões internos muito diferentes, difíceis de codificar. O dicionário não tem tempo hábil para aprender os padrões da imagem. E este é o caso da imagem MAN.

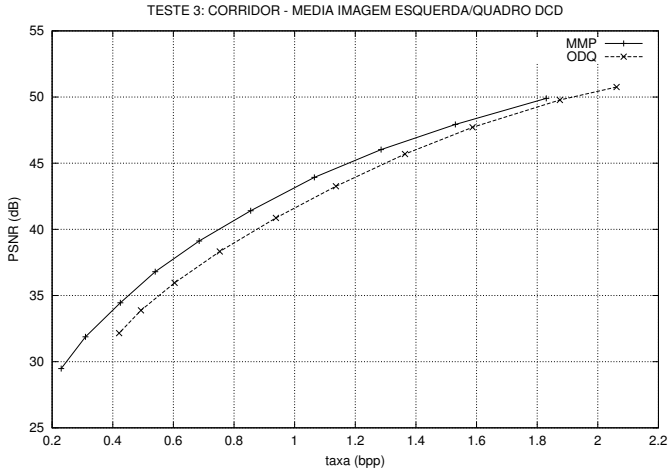
O MMP fornece melhores resultados do que o ODQ para imagens com mais textura, como o CORRIDOR, e o FRUIT (em algumas partes da imagem). Isto ocorre porque o ODQ é baseado no padrão JPEG, e portanto, devido ao uso da transformada DCT, consegue melhor compressão para imagens com baixa textura. Além disso, estas duas imagens possuem um quadro de diferenças menos ruidoso do que o do MAN, e assim o dicionário do MMP tem tempo de aprender os padrões desta imagem. Por estes dois motivos, conseguimos melhores resultados de compressão com o MMP do que com o ODQ para estas duas imagens. Na imagem FRUIT observamos que o ODQ ganha do MMP em taxas mais baixas (abaixo de 1,5 bpp),



(a)

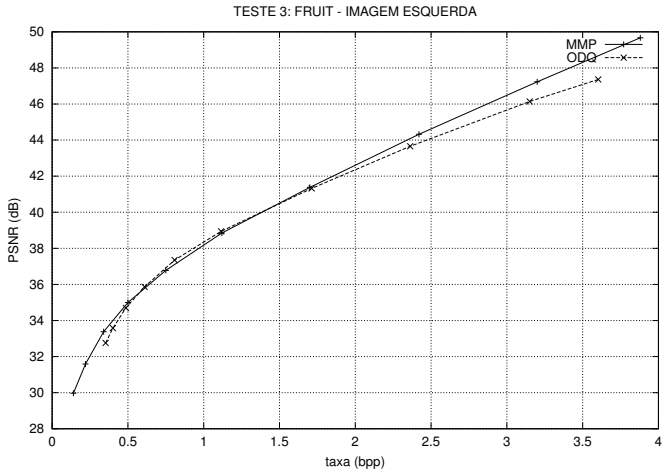


(b)

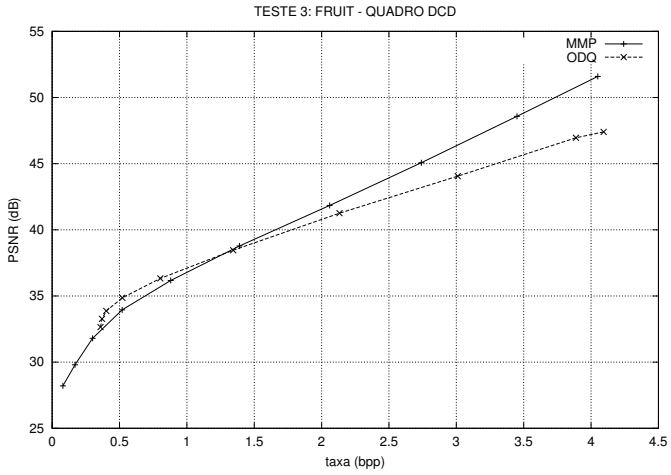


(c)

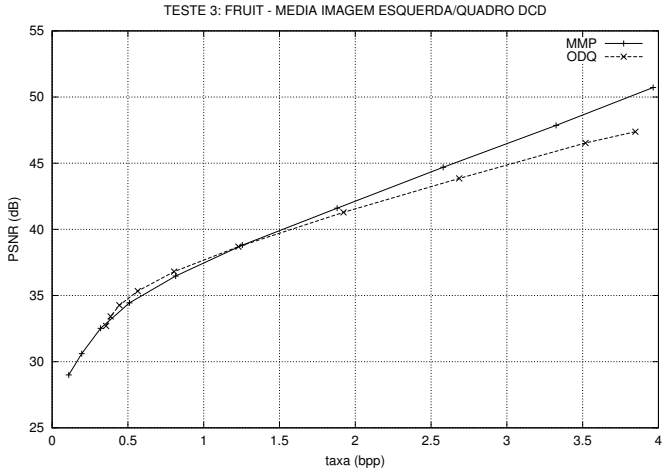
Figura 3.12: Resultados do teste 1 para a imagem estéreo CORRIDOR: (a) Imagem esquerda, (b) Quadro DCD, (c) média entre a imagem esquerda e o quadro DCD.



(a)

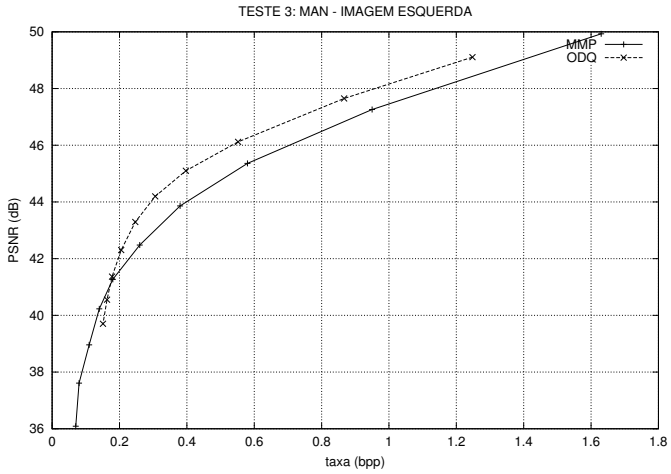


(b)

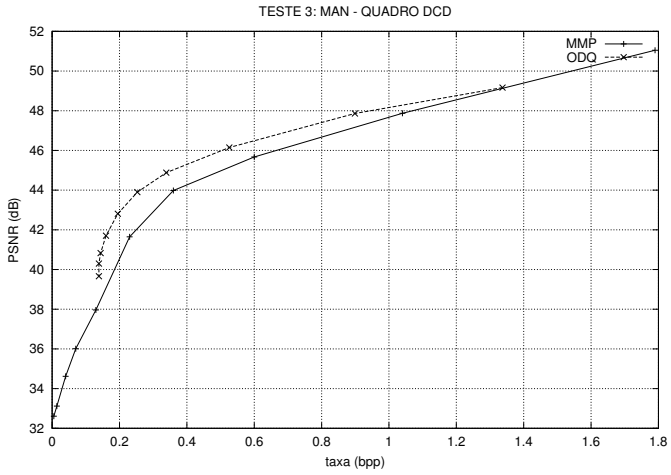


(c)

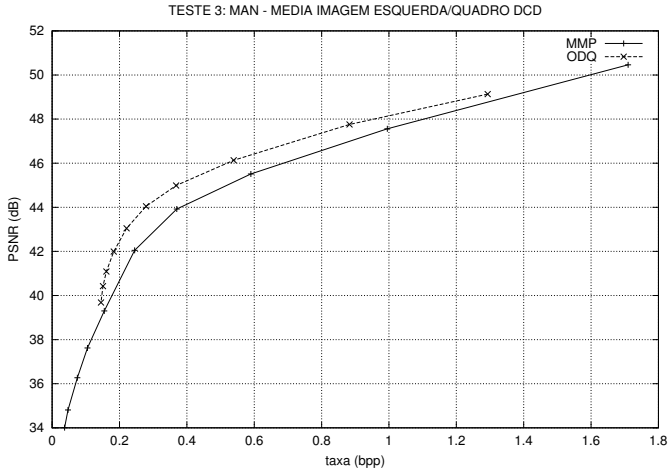
Figura 3.13: Resultados do teste 1 para a imagem estéreo FRUIT: (a) imagem esquerda, (b) quadro DCD, (c) média entre imagem esquerda e DCD.



(a)



(b)



(c)

Figura 3.14: Resultados do teste 1 para a imagem estéreo MAN: (a) Imagem esquerda , (b) quadro DCD, (c) média entre imagem esquerda e DCD.

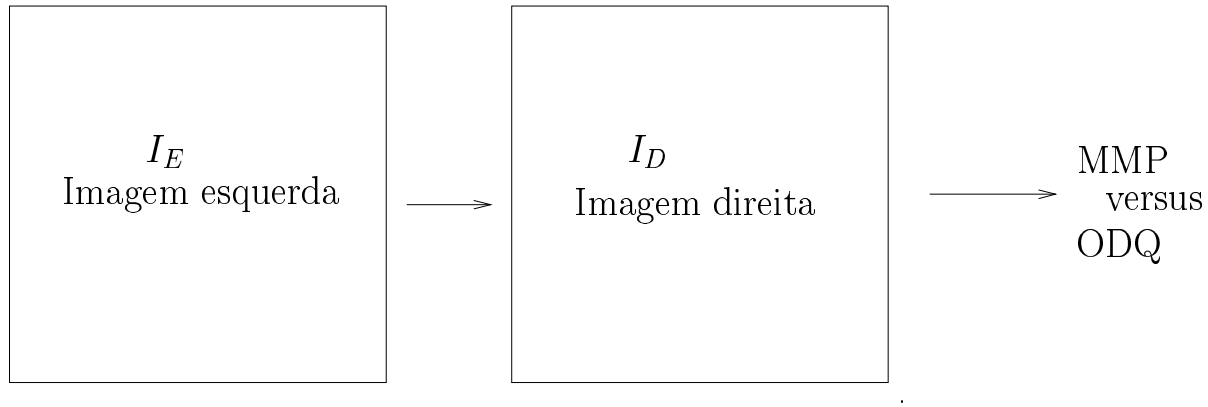


Figura 3.15: Diagrama em blocos do teste 2: Comparação entre os algoritmos ODQ e MMP aplicados diretamente às imagens esquerda e direita do par estéreo. O quadro DCD não é utilizado.

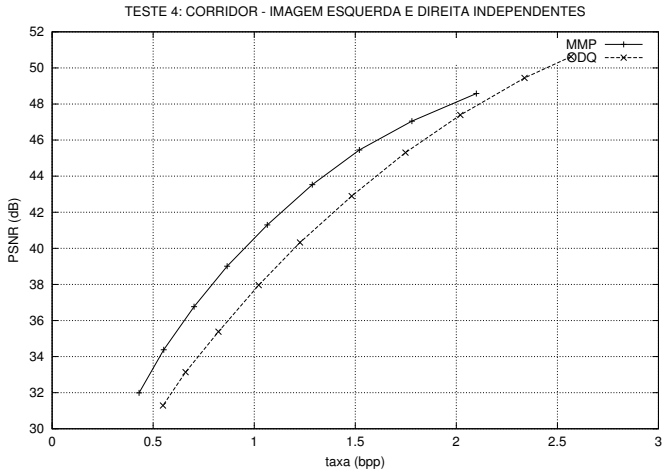
que deve corresponder exatamente àquelas regiões da imagem que possuem baixa textura, nas quais o ODQ leva vantagem em relação ao MMP.

3.5 Teste 2: MMP versus ODQ, aplicados diretamente às imagens esquerda e direita, sem cálculo de quadro DCD

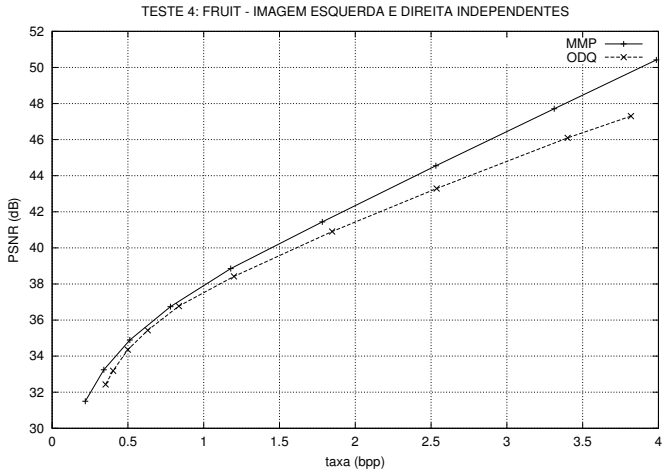
Este teste consiste em comparar os dois métodos MMP e ODQ para as imagens esquerda e direita independentes, sem cálculo de quadro de diferenças DCD em nenhum dos dois métodos (Fig. 3.15).

Neste caso a comparação é injusta com o ODQ, porque o MMP aproveita o dicionário construído durante a codificação da imagem esquerda para codificar a imagem direita. O ODQ não está sendo aplicado para o seu propósito, que é otimizar os quantizadores da imagem esquerda e do quadro DCD de forma dependente. Portanto espera-se que o MMP forneça melhores resultados do que o ODQ. A Fig. 3.16 mostra os resultados do teste 2 para as imagens estéreo CORRIDOR, FRUIT e MAN.

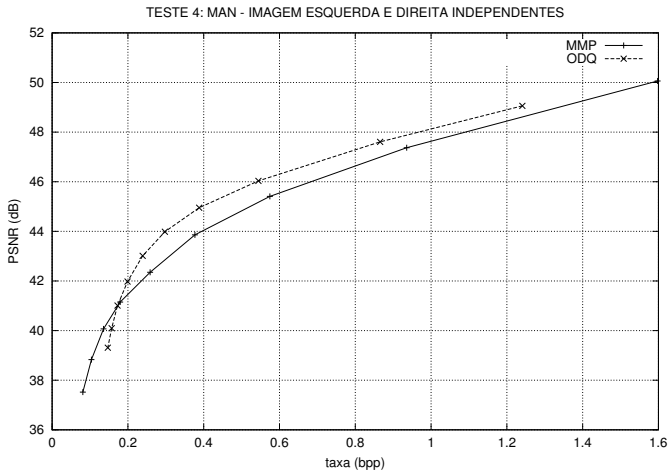
Como era esperado, o MMP apresentou melhores resultados do que o ODQ, tanto para a imagem FRUIT quanto para a imagem CORRIDOR. Porém, contrar-



(a)



(b)



(c)

Figura 3.16: Resultados do teste 2 para as imagens estéreo (a) CORRIDOR, (b) FRUIT, e (c) MAN.

iando a nossa previsão, o ODQ continuou dando melhores resultados para a imagem MAN. Isto provavelmente se deve ao fato de haver pouca correlação entre as imagens esquerda e direita deste par estéreo, fazendo com que o aprendizado do dicionário do MMP não seja útil de uma imagem para a outra. Como já havíamos mencionado, este par estéreo é considerado muito difícil de codificar devido aos grandes valores de disparidade. Esta imagem possui também várias regiões de oclusão, o que dificulta ainda mais encontrar blocos similares entre as duas imagens.

3.6 Teste 3: MMP aplicado às imagens esquerda e direita, versus ODQ aplicado à imagem esquerda e ao quadro DCD

Aproveitando os resultados obtidos no teste 1 e no teste 2, fazemos agora uma comparação entre imagem esquerda e quadro DCD codificados pelo ODQ, com imagem esquerda e imagem direita codificados pelo MMP. Estaremos assim comparando o desempenho dos dois algoritmos da forma como cada um é melhor aproveitado (Fig. 3.17).

A Fig. 3.18 mostra os resultados do teste 3 para as imagens CORRIDOR, FRUIT e MAN.

Para estas imagens o ODQ levou vantagem em relação ao MMP. Este resultado de certa forma já era esperado porque o ODQ foi otimizado para codificar o par estéreo, enquanto que o MMP foi projetado inicialmente para codificar uma imagem monocular, não existindo mecanismos para explorar a similaridade estéreo como existem no ODQ.

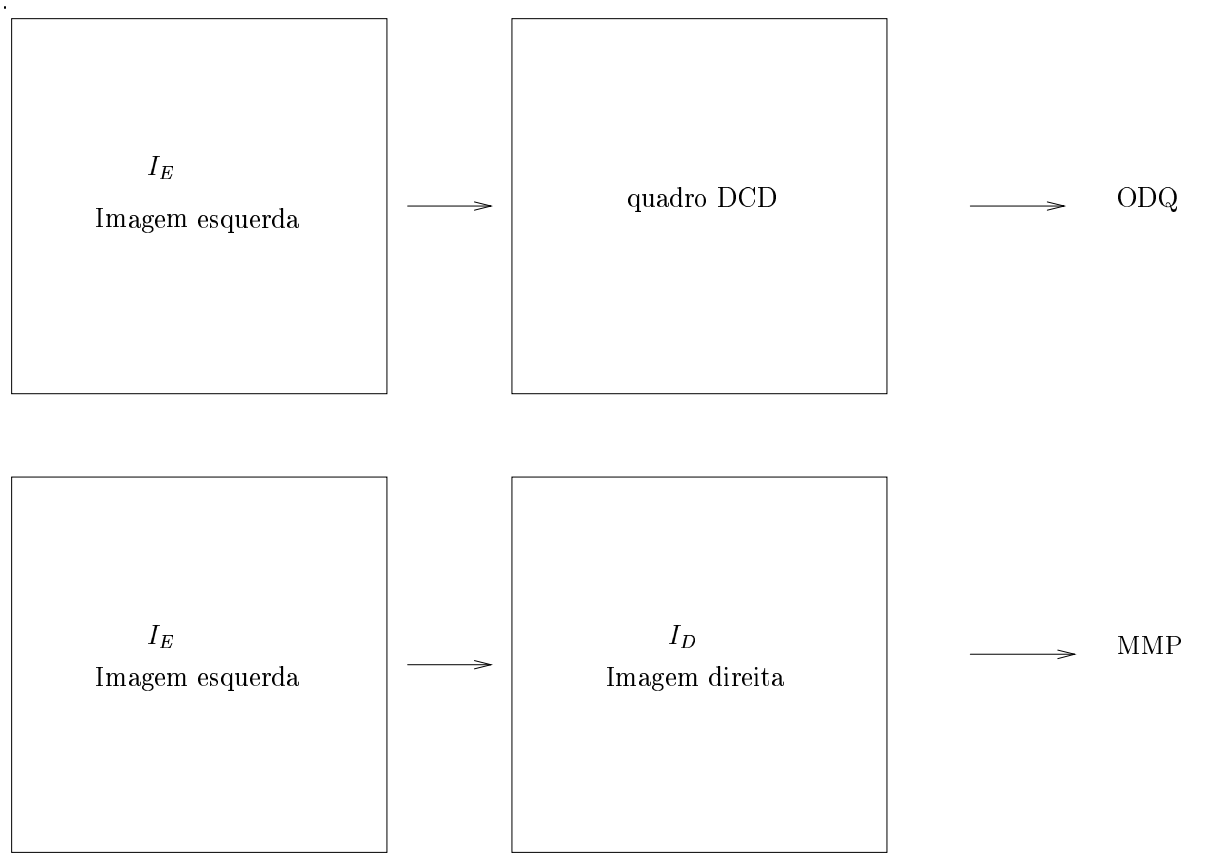
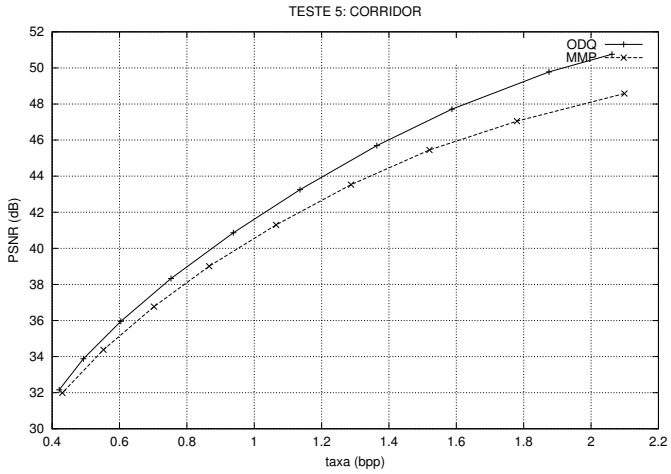
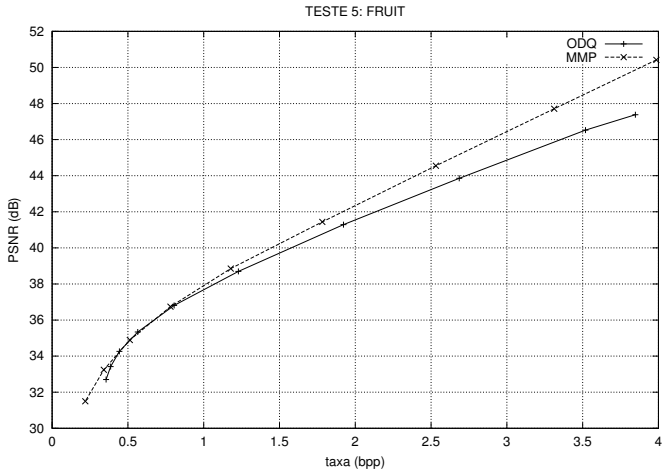


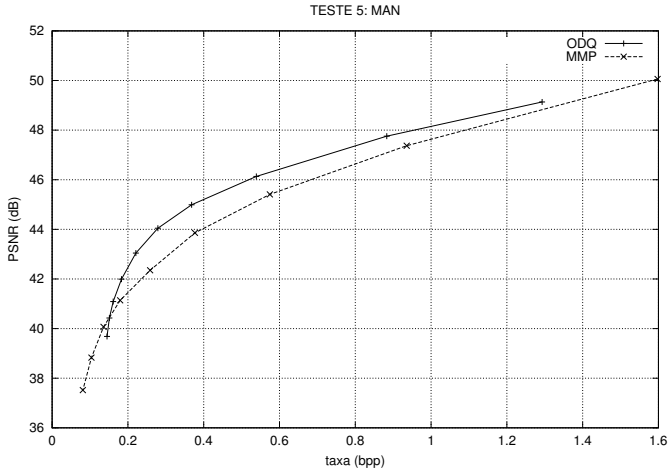
Figura 3.17: Diagrama em blocos do teste 3: Comparação entre o algoritmo ODQ aplicado à imagem esquerda e ao quadro DCD, com algoritmo MMP aplicado diretamente às imagens esquerda do par estéreo, sem uso do quadro DCD.



(a)



(b)



(c)

Figura 3.18: Resultados do teste 3 para as imagens estéreo (a) CORRIDOR, (b) FRUIT e (c) MAN.

3.7 Conclusão

Neste capítulo realizamos testes comparando o desempenho dos algoritmos MMP e ODQ aplicados a imagens estéreo. O primeiro algoritmo é uma proposta de uma nova classe de codificadores, que utiliza recorrência de padrões. O MMP não foi proposto inicialmente para codificar imagens estéreo, e portanto, não possui mecanismos que explorem eficientemente as características destas imagens. O segundo algoritmo é uma proposta recente para codificar imagens estéreo eficientemente, que encontra quantizadores ótimos para codificar a imagem esquerda e o quadro de diferenças DCD.

O nosso objetivo é saber se podemos investir na melhoria do MMP para codificar eficientemente imagens estéreo, e em que direção devemos seguir para conseguir estas melhorias.

No primeiro teste realizado nós aplicamos o ODQ e o MMP à imagem esquerda do par estéreo, e ao quadro de diferenças DCD. O MMP apresentou melhor desempenho do que o ODQ para o par estéreo CORRIDOR. Para o par estéreo FRUIT, o MMP apresenta melhores resultados em taxas mais altas (a partir de 1.5 bpp), e para o par estéreo MAN, o ODQ mostrou-se bem melhor do que o MMP. Considerando-se que neste teste o MMP não explorou nenhuma característica intrínseca à uma imagem estéreo, o MMP é um método promissor (podemos investir neste método para obter melhores resultados de compressão de imagens estéreo) porque ficou dentro da faixa de comparação com o ODQ, que é um método otimizado para imagens estéreo.

Existem dois aspectos a serem considerados na análise do comportamento destes dois algoritmos:

1. Tipos de padrões presentes na imagem;

2. Faixa de valores de disparidade.

O primeiro aspecto, que não diz respeito somente às imagens estéreo, mas sim a qualquer tipo de imagem, são os padrões presentes na imagem. Eles podem ser homogêneos (baixa textura), o que leva a um espectro suave, com freqüências predominantemente baixas, ou podem ter muitos detalhes (alta textura), o que leva

a um espectro com altas freqüências. No primeiro caso, o ODQ se beneficia porque ele é baseado no padrão JPEG, que apresenta melhores resultados de compressão quando o espectro resultante da transformada DCT tem em sua maioria freqüências baixas. Para o segundo caso o MMP apresenta melhor desempenho porque tenta representar o bloco da imagem por sub-blocos de tamanho variável, o que leva a uma melhor representação de regiões da imagem com mais detalhes. Este primeiro aspecto pode ser observado na codificação da imagem esquerda, na qual ambos os algoritmos são aplicados diretamente.

O outro aspecto a ser considerado, que diz respeito exclusivamente às imagens estéreo, são os valores de disparidade obtidos entre as duas imagens do par estéreo. Estes valores são diretamente relacionados à distância entre as câmeras (*baseline*), e à distância da cena 3D às câmeras (profundidade). Para valores grandes de disparidades a codificação é mais difícil, principalmente quando existe oclusão. No ODQ os problemas de oclusão podem comprometer a qualidade da imagem reconstruída, porque não se encontra correspondência nas regiões de oclusão. No MMP este problema pode ser contornado porque este algoritmo divide o bloco até encontrar uma representação ótima, segundo o critério de taxa e distorção.

Analisando-se o primeiro aspecto, observado principalmente na codificação da imagem esquerda, o MMP apresentou melhores resultados do que o ODQ para aquelas imagens com mais textura, como o par estéreo CORRIDOR. Para a imagem FRUIT, a vantagem devido a este aspecto não fica muito clara porque esta imagem possui algumas regiões com alta textura e outras com baixa textura. Para o MAN, que possui área muito extensas de baixa textura, por exemplo o fundo (*background*) e a face, o ODQ apresenta melhor desempenho do que o MMP.

Analisando-se o segundo aspecto (valores de disparidades), observado principalmente na codificação da imagem direita (do frame DCD, no caso do ODQ), o MMP apresenta bons resultados para aquelas imagens cujos valores de disparidade não são altos, como o CORRIDOR (faixa de disparidade entre 0 e 15) e o FRUIT (faixa de disparidade entre 0 e 16). Nestes casos a similaridade entre as duas imagens é explorada eficientemente pelo MMP apenas pelo uso do mesmo dicionário, que usa os padrões aprendidos na imagem esquerda para codificar a imagem direita. Mas no caso do MAN, cuja faixa de disparidade é muito alta (entre 25 e 128),

simplesmente usar o mesmo dicionário não é suficiente. Isto nos motiva a continuar a investigação na direção de usar técnicas que explorem melhor as características estéreo das imagens.

Se conseguirmos explorar adequadamente a similaridade existente entre as imagens esquerda e direita somente inserindo no dicionário elementos que representem esta correlação, poderemos evitar o cálculo do mapa de disparidades bem como o uso do quadro de diferenças DCD, evitando assim o gasto de bits extras para transmitir o mapa de disparidades ao decodificador. Esta já será uma vantagem considerável do MMP em relação ao ODQ, caso consigamos obter resultados de compressão pelo menos na mesma faixa comparando os dois algoritmos.

Capítulo 4

MMP estéreo

4.1 Introdução

No Cap. 3 fizemos alguns testes comparando o desempenho do MMP com o ODQ (ver Cap. 3, seção 3.2 e Apêndice A). O objetivo principal deste conjunto de testes foi obter uma direção na qual pudéssemos prosseguir no desenvolvimento desta tese, no sentido de melhorar o desempenho do MMP para codificar imagens estéreo.

De fato, baseado nos resultados dos testes do Cap. 3, pudemos conhecer melhor os problemas e tirar conclusões importantes que nos guiaram no desenvolvimento do presente capítulo. A partir de agora iremos investigar o uso do MMP de modo a explorar eficientemente as características das imagens estéreo. Com isto esperamos melhorar os resultados da compressão das imagens estéreo utilizadas.

Os pares de imagens estéreo utilizadas neste capítulo, além das imagens CORRIDOR e MAN já utilizadas anteriormente, são AQUA, SAXO e TRAIN (ver Apêndice B). A imagem AQUA possui muita textura, e como não são padrões repetitivos, é fácil encontrar a correspondência entre as duas imagens. A janela de busca desta imagem é [-40, 16]. Esta imagem possui disparidade vertical induzida, mas como é muito pequena (0,5), pode ser negligenciada. A imagem TRAIN também possui muita textura e sua faixa de disparidades (puramente horizontal) é [-28,17]. A imagem SAXO é uma imagem difícil de compactar porque a região de fundo com baixa textura dificulta encontrar a correspondência, possui muitas áreas de oclusão devido à grande distância entre as câmeras, e sua faixa de disparidades é [-40, 17].

4.2 MMP aplicado a linhas de blocos (ROBs) correspondentes

A primeira idéia de utilizar o MMP para codificar um par de imagens estéreo foi aplicá-lo diretamente a uma linha de blocos (*row of blocks, ROB*) da imagem esquerda, e logo depois, utilizando o mesmo dicionário, aplicá-lo à ROB correspondente da imagem direita. Como existe uma similaridade entre ROBs correspondentes das duas imagens do par estéreo, esperamos que os padrões da imagem esquerda armazenados no dicionário permitam uma melhor compressão da imagem direita. Por enquanto nenhuma outra característica estéreo será explorada, a não ser esta similaridade entre ROBs.

A Fig. 4.1 mostra um diagrama das imagens esquerda e direita do par estéreo e suas ROBs correspondentes, onde o MMP será aplicado. Neste primeiro teste não calculamos mapa de disparidades nem usamos nenhum outro método para obter a correlação existente entre as duas imagens. O MMP foi aplicado diretamente sem nenhum processamento prévio das imagens. À medida que os blocos da ROB esquerda são processados, acrescenta-se ao dicionário os elementos que melhor representam estes blocos. Este mesmo dicionário é utilizado para processar a ROB correspondente da imagem direita. Como existe similaridade entre ROBs correspondentes do par estéreo, no momento da codificação da ROB direita, os elementos do dicionário que representam blocos da ROB esquerda podem também representar blocos da ROB direita que forem parecidos com blocos da ROB esquerda, e assim esperamos obter uma melhor compressão da imagem direita.

As imagens de teste utilizadas neste capítulo foram as mesmas do Cap. 3.

Novamente comparamos o desempenho do MMP com o ODQ (ver Cap. 3, seção 3.2 e Apêndice A).

Este é somente um teste preliminar para termos uma idéia inicial a respeito do desempenho do MMP aplicado diretamente a um par estéreo. É importante enfatizarmos a diferença entre este teste e o que fizemos no Cap. 3. Lá aplicamos o MMP à imagem esquerda toda, e depois usamos o mesmo dicionário para codificar a imagem direita. Aqui o MMP é utilizado para codificar uma ROB da imagem esquerda e usa o mesmo dicionário para codificar a ROB correspondente da imagem

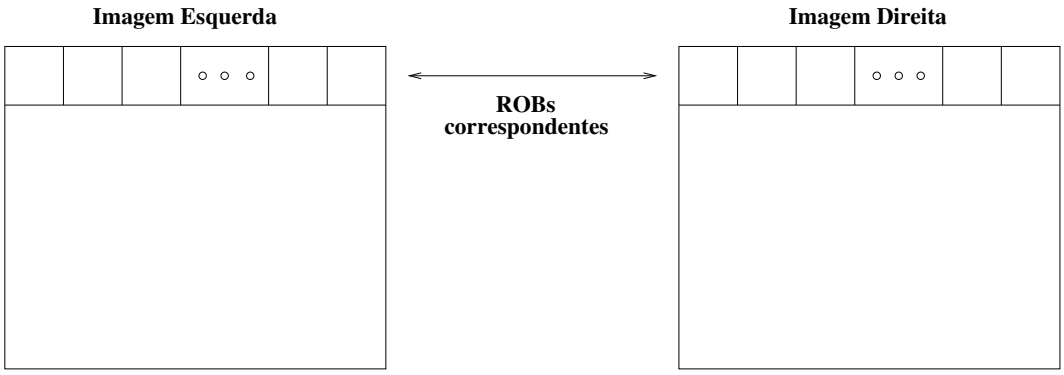


Figura 4.1: Diagrama em blocos de um par estéreo (imagens esquerda e direita) com suas linhas de blocos (ROBs) correspondentes.

direita.

Neste teste a única característica estéreo explorada (indiretamente, pois não calculamos os vetores de disparidade) é a similaridade existente entre duas linhas de blocos correspondentes das imagens esquerda e direita do par estéreo.

A figura 4.2 mostra os resultados de taxa × PSNR do MMP aplicado à imagem estéreo CORRIDOR. Este resultado é uma média dos valores de taxa e PSNR das imagens esquerda e direita. Podemos verificar que houve um ganho de aproximadamente 2 dB somente devido ao fato de codificar as ROBs correspondentes. Este resultado indica que realmente existe similaridade entre as ROBs, e que podemos conseguir melhorar os resultados de compressão de imagens estéreo utilizando adequadamente o MMP.

A partir de agora vamos descrever uma seqüência de versões do MMP estéreo desenvolvidas ao longo desta tese. Cada versão representa uma ou mais idéias de implementação que resultaram em melhorias consideráveis na codificação de imagens estéreo. A sinopse destas versões encontra-se no Apêndice C desta tese. A primeira parte do desenvolvimento desta tese, que é marcada pelo não uso de mapa de disparidades, é representada pelas versões 1 a 6, descritas neste capítulo. A segunda parte, na qual introduzimos o uso dos vetores de disparidades, representada pelas versões 7 a 12, encontra-se no Cap. 5 desta tese.

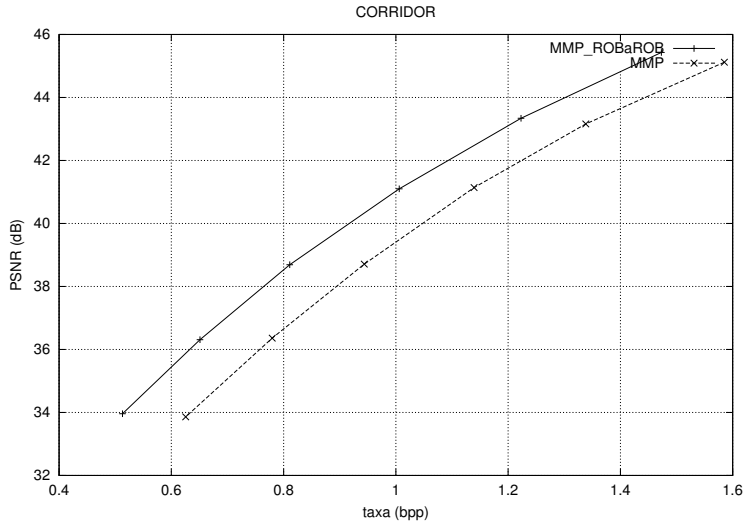


Figura 4.2: MMP aplicado às linhas de blocos esquerda e direita (MMP ROB a ROB), comparado com MMP aplicado à imagem esquerda completa e depois à imagem direita completa do par estéreo CORRIDOR.

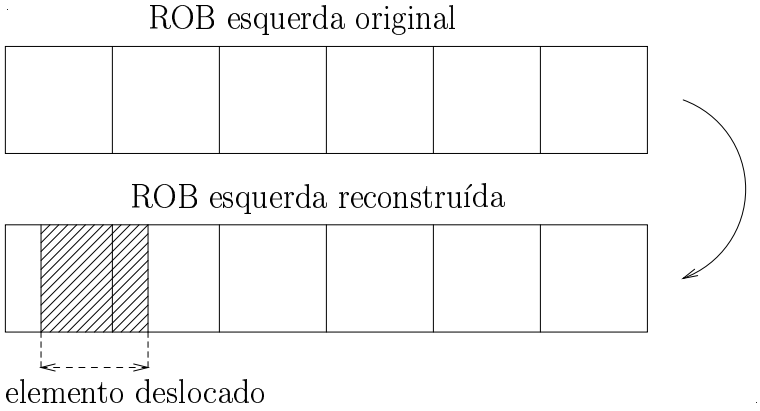


Figura 4.3: Elementos deslocados a partir de dois blocos vizinhos pertencentes a uma ROB da imagem esquerda reconstruída.

4.3 Versão 1 do MMP estéreo

A primeira tentativa no sentido de explorar as características estéreo das imagens foi introduzir elementos deslocados no dicionário. Estes elementos são deslocamentos entre dois blocos vizinhos de uma ROB da imagem esquerda reconstruída (ver Fig. 4.3).

Os deslocamentos representam (indiretamente) valores de disparidade existentes entre duas imagens estéreo. Esta porém é uma consideração grosseira, que

representa apenas um caso particular, porque a disparidade só corresponde a um deslocamento simples quando o objeto da cena 3D se encontra a uma distância suficientemente grande das duas câmeras estéreo. Quando este é o caso, deformações do objeto devido às distorções de perpectiva não são percebidas.

A princípio esperava-se que os elementos deslocados só fossem ocorrer na codificação da ROB correspondente na imagem direita, porém verificou-se que eles são utilizados na codificação de blocos da própria ROB esquerda, provavelmente devido a similaridades de padrões dentro da mesma imagem. Estas ocorrências podem ser observadas nos gráficos de barras da Fig 4.5 (a) e da Fig 4.6 (a), mostrados mais adiante nesta seção. Inicialmente estes deslocamentos foram inseridos em todos os níveis do dicionário, mas após alguns testes observou-se que o uso destes só apresentava resultados relevantes em níveis mais altos (até 3 níveis quando se usa blocos 8x8, no caso, níveis 6, 5 e 4).

A partir daqui chamamos o MMP de **MMP estéreo** porque ele já se diferencia do MMP original, incorporando procedimentos diretamente relacionados com as características das imagens estéreo.

Nesta primeira versão usamos o mesmo dicionário para codificar a imagem esquerda e a imagem direita. Os elementos deslocados inseridos no dicionário durante o processamento de uma ROB esquerda foram utilizados para codificar blocos posteriores da própria ROB, e foram utilizados para codificar blocos da ROB direita correspondente. Porém, durante o processamento da ROB esquerda seguinte, os elementos deslocados foram substituídos pelos novos elementos deslocados gerados pela ROB atual. Isto foi feito de forma proposital, porque considerou-se a restrição de linha epipolar quando se usam câmeras paralelas, que pressupõe a existência de correspondência apenas entre as linhas de blocos na mesma linha horizontal, ou seja, uma ROB da imagem esquerda só tem correspondência com a mesma ROB da imagem direita, e não com as outras ROBs. Nos testes realizados verificamos que usar o mesmo dicionário para codificar as duas imagens dá melhores resultados do que usar um dicionário diferente para cada imagem, mesmo que os deslocamentos da imagem esquerda sejam inseridos no dicionário da imagem direita.

Também testamos utilizar um dicionário só, mas com modelos de atualização diferentes no codificador aritmético (que registram as ocorrências de elementos de

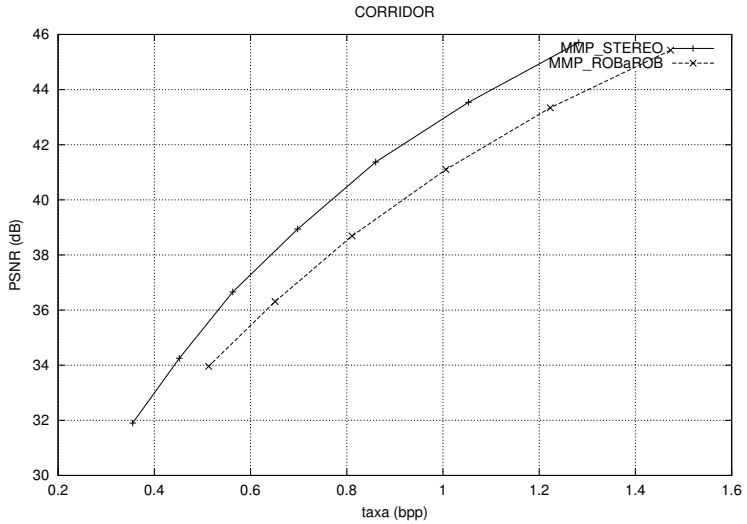


Figura 4.4: Comparação entre o MMP estéreo (versão 1) e MMP ROB a ROB para a imagem estéreo CORRIDOR (média das imagens esquerda e direita.

cada imagem). Verificamos que utilizar um só modelo de codificador aritmético para o dicionário é melhor do que utilizar um modelo para cada imagem.

Aplicamos o MMP estéreo à imagem CORRIDOR e comparamos com o MMP ROB a ROB mostrado anteriormente na Fig. 4.2. Podemos observar pelo gráfico que a inserção de elementos deslocados no dicionário melhorou os resultados de taxa e PSNR em relação ao MMP ROB a ROB mostrado anteriormente na primeira seção deste capítulo.

Fizemos um estudo estatístico baseado no número de ocorrências de cada tipo de elemento do dicionário durante a codificação da imagem esquerda e da imagem direita. Para isto continuamos utilizando um só dicionário para codificar as duas imagens, mas utilizamos dois modelos, um para a imagem esquerda e outro para a imagem direita, para podermos contabilizar as ocorrências dos elementos em cada imagem separadamente. Já havíamos concluido que usar um só modelo para as duas imagens era mais vantajoso, mas neste caso, para o estudo estatístico é necessário separar os modelos. Os tipos de elementos presentes no dicionário do MMP estéreo são:

1. Elementos Iniciais;

2. Elementos Concatenados;

3. Elementos Contraídos;

4. Elementos Expandidos;

5. Elementos Deslocados.

A Fig. 4.5 mostra o gráfico de ocorrências dos elementos do dicionário por nível e por tipo de elemento, para a imagem estéreo CORRIDOR. O valor de λ utilizado foi 32 (um valor intermediário dentro da faixa de 2 a 128). Para valores de λ menores que 32 observamos uma grande ocorrência de elementos nos níveis mais baixos do dicionário. Valores de λ pequenos priorizam a distorção em detrimento da taxa. O MMP então otimiza o custo priorizando a distorção, procurando dentre os elementos do dicionário uma melhor representação para o bloco. Sendo assim, como ocorrem mais divisões, os níveis mais baixos do dicionário são mais utilizados. Pelo mesmo motivo, para valores de λ maiores que 32, observamos que aumentam as ocorrências em níveis mais altos do dicionário.

Na parte (a) desta figura, que corresponde à imagem esquerda, verificamos que no nível 0 ocorrem 100% de elementos iniciais. Isto acontece porque no início da codificação o dicionário é pobre, e assim o MMP não encontra um bom casamento entre o bloco da imagem e os blocos maiores do dicionário. O MMP então divide os blocos até o nível de pixels para atender o critério RD. Convém mencionarmos aqui que o algoritmo MMP antes de inserir um elemento no dicionário, verifica se este elemento já existe. Se existir ele não é inserido novamente. No nível 0, os elementos iniciais já cobrem quase toda a faixa de pixels existentes na imagem. Ou seja, o dicionário para este nível já começa bastante representativo, principalmente se o *passo* é igual a 1 (ver Cap.2). Quando o *passo* é grande, alguns valores de pixel intermediários serão inseridos no nível 0 do dicionário como contrações de elementos de outros níveis. Portanto, a maior parte das ocorrências no nível 0 são de elementos iniciais. No nível 1 já começam a aparecer as ocorrências de elementos concatenados e contraídos. A ocorrência de elementos iniciais e elementos concatenados se mantém praticamente constante para esta imagem, desde o nível 1 até o nível 5. Os elementos expandidos foram pouco utilizados nesta imagem se compararmos com o número de ocorrência dos outros elementos, principalmente dos elementos deslocados. Poderíamos questionar a sua importância para a codificação

estéreo, porém é importante considerar que o número de elementos expandidos (bem como contraídos, concatenados e iniciais) é bem menor do que o número de elementos deslocados incluídos no dicionário. O estudo estatístico realizado aqui obteve um número de ocorrências absoluto, e não relativo ao número de elementos que foi incluído no dicionário. Portanto é prematuro descartar a utilidade dos elementos expandidos nesta etapa inicial de testes.

Nos níveis 4, 5 e 6 já observamos a ocorrência de elementos deslocados, ou seja, eles foram gerados durante o processamento desta imagem, e estão sendo utilizados para codificar blocos da própria imagem. Isto tem a ver com o tipo de imagem, que possui correlação entre regiões da própria imagem, e não com alguma característica estéreo, que foi o que deu origem a esta idéia de usar elementos deslocados. Talvez por causa disso o número de ocorrências deste tipo de elemento na imagem esquerda seja pequeno, comparado com o número de ocorrências na imagem direita. A Fig. 4.5 (b) mostra o gráfico para a imagem direita do mesmo par estéreo. Os resultados apresentados neste gráfico são bastante reveladores. Observamos nitidamente o aumento das ocorrências de elementos deslocados na codificação desta imagem. Este resultado é animador, pois nos indica que o MMP realmente explorou a característica estéreo de disparidade representada por este elemento. Os elementos iniciais do nível 0 continuam com um percentual muito grande de ocorrência (90%), porém para esta imagem já se observa a ocorrência também de elementos contraídos, mostrando que a contração foi importante para gerar pixels representativos da imagem que não estavam presentes no dicionário inicial. As concatenações são mais utilizadas nesta imagem do que na imagem esquerda, indicando uma maior correlação entre regiões dentro da mesma imagem. Os elementos iniciais são cada vez menos usados à medida que crescem os níveis, sendo superados pelo número cada vez maior de ocorrência dos elementos deslocados. Isto é bastante significativo, pois este indício explica o resultado de melhor compressão. Esta melhoria se deve à redução considerável na taxa de codificação, visto que menos índices e menos flags são gerados para codificar elementos de níveis mais altos. Neste sentido, a inserção de elementos deslocados no dicionário foi crucial para a codificação da imagem direita.

A Fig. 4.6 mostra o gráfico de ocorrências dos elementos do dicionário por nível e por tipo de elemento, para a imagem estéreo MAN. Esta imagem estéreo

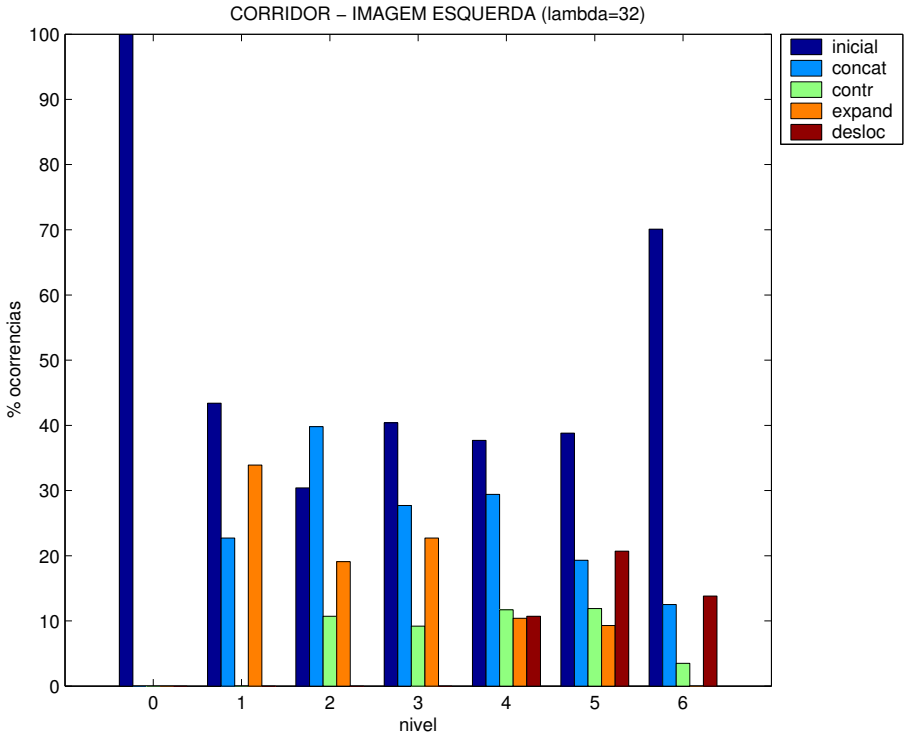
é muito difícil do ponto de vista de codificação. Não é diferente portanto para o MMP. Ela não possui correlação entre as regiões da imagem, portanto as inserções no dicionário ao longo do processamento não são suficientes para codificá-la eficientemente. Observamos a ocorrência muito alta de elementos iniciais, em todos o níveis, e principalmente no nível 6. Isto é realmente um indício de que o dicionário não aprendeu quase nada significativo em relação à imagem. Há uma ocorrência também alta de elementos contraídos, o que reforça a afirmativa anterior. Estes elementos supriram a falta de elementos representativos da imagem no dicionário inicial, mas representam um aprendizado muito pequeno comparado com outros tipos de elementos. A ocorrência de expansões, novamente, é muito pequena, mas, como foi observado anteriormente, é muito cedo para descartarmos a relevância deste tipo de elemento na codificação de imagens estéreo. E, finalmente, a ocorrência de deslocamentos é muito pequena para este par, mesmo na imagem direita, indicando que a grande disparidade entre as duas imagens foi um grande impecilho para que o MMP conseguisse compactar este par estéreo com sucesso. Se conseguirmos melhorar a qualidade dos elementos deslocados, ou de outro tipo de elemento a ser inserido no dicionário, de forma que se explore mais as características estéreo, poderemos melhorar ainda mais a compressão do par estéreo com o MMP.

Este estudo estatístico foi importante para nos mostrar o caminho a ser seguido no sentido de explorar as características estéreo para se obter uma melhor compressão da imagem direita. Pudemos observar a importância dos elementos deslocados, apesar de não serem suficientes para a codificação de imagens cujos valores de disparidade são muito grandes, como no caso da imagem MAN.

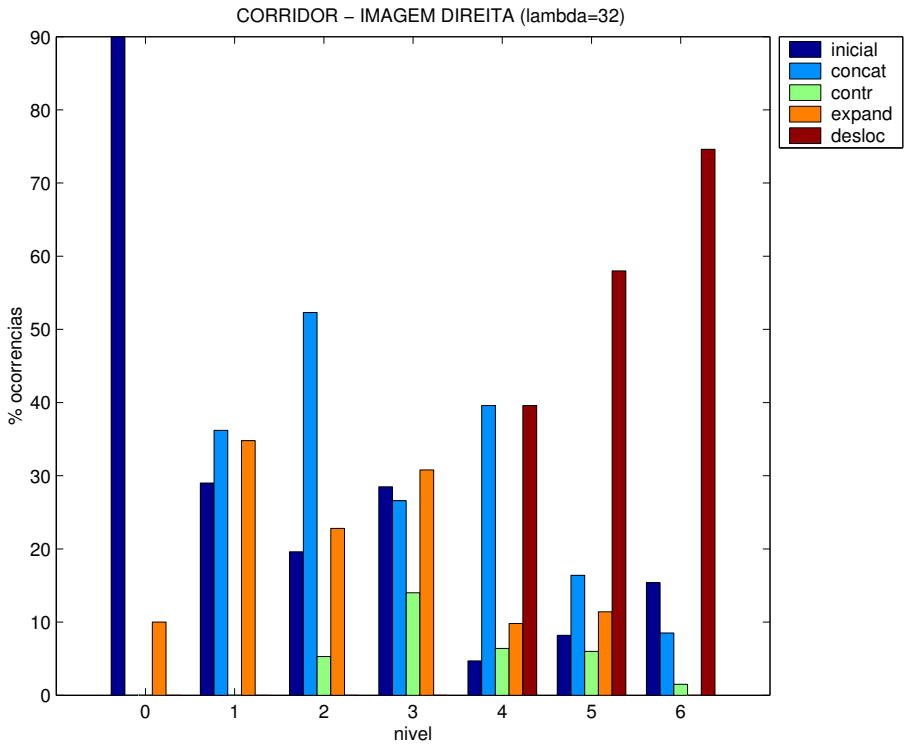
4.4 Versão 2 do MMP estéreo

Na versão 1 do MMP estéreo, os elementos deslocados gerados no processamento de uma ROB só eram utilizados para o processamento da ROB direita correspondente, ou seja, não eram aproveitados para outras ROBs.

Como já mencionado anteriormente, esperava-se a princípio que os elementos deslocados só fossem ocorrer na codificação da imagem direita, porém verificou-se que eles são utilizados na codificação de blocos da imagem esquerda, provavelmente

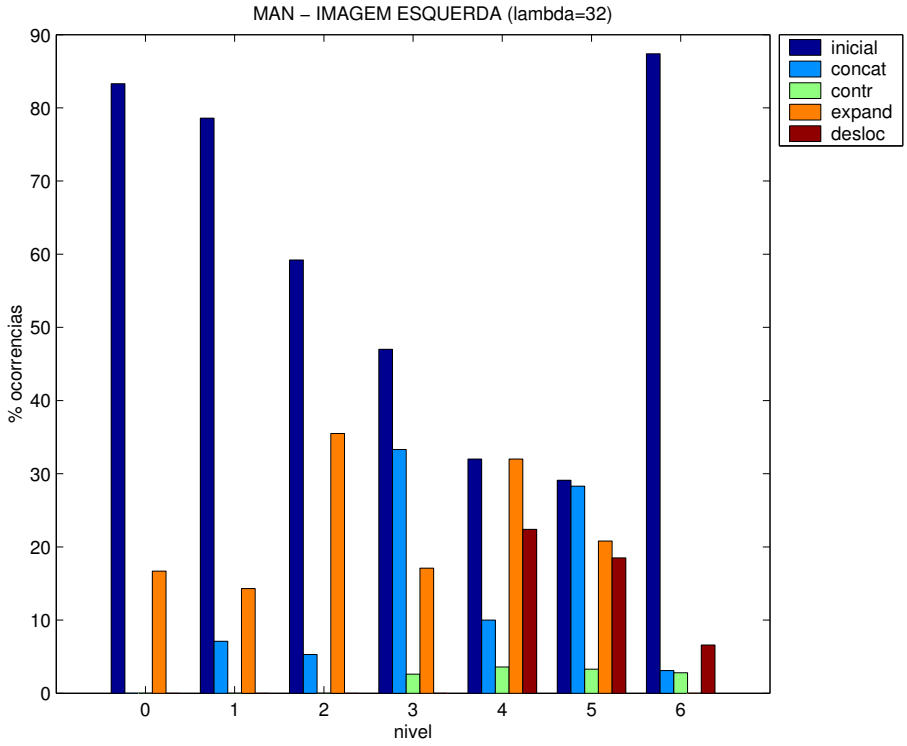


(a)

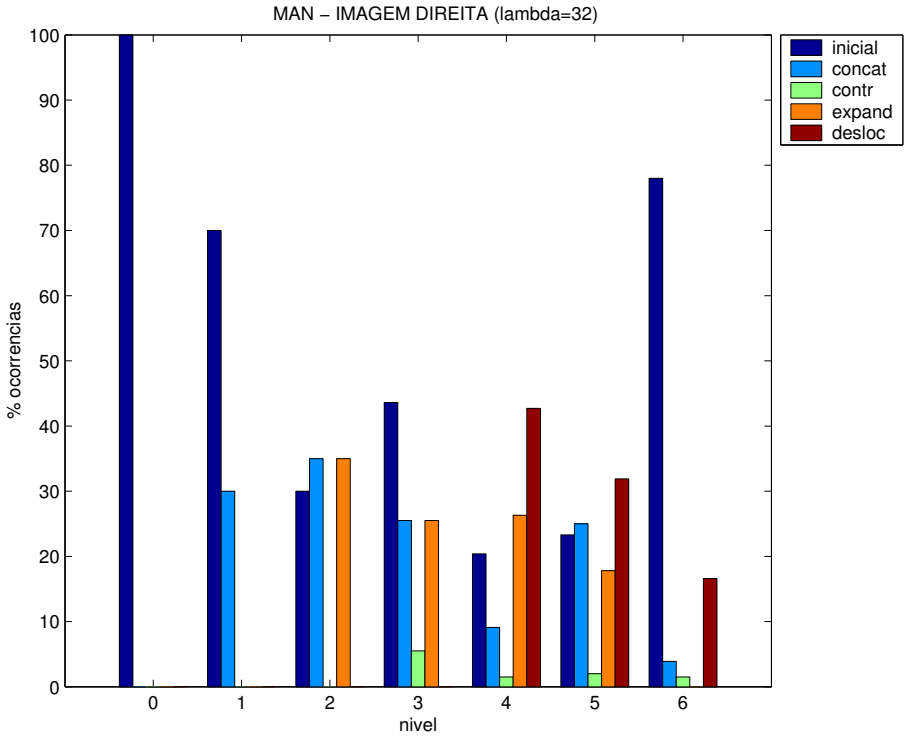


(b)

Figura 4.5: Percentual de ocorrências dos elementos do dicionário por tipo, para cada nível, (a) para a imagem esquerda, e (b) para a imagem direita do par estéreo CORRIDOR.



(a)



(b)

Figura 4.6: Percentual de ocorrências dos elementos do dicionário por tipo, para cada nível, (a) para a imagem esquerda, e (b) para a imagem direita do par estéreo MAN.

devido a similaridades de padrões dentro da imagem. Isto nos levou a conjecturar que elementos deslocados de ROBs anteriores também pudessem ser úteis na codificação de outras ROBs que não fossem necessariamente correspondentes.

Isto foi implementado na versão 2 do MMP estéreo, onde os elementos deslocados gerados são acumulados no dicionário de ROB para ROB. Pudemos observar que os resultados melhoraram para a imagem esquerda, porque o dicionário passou a conter mais elementos que representavam similaridades na mesma imagem, e pioraram para a imagem direita, porque aumentou-se o número de elementos não úteis para a codificação desta imagem, ou seja, só aumentamos o número de elementos do modelo, com elementos não significativos. Isto comprova que para a codificação da ROB da imagem direita é mais significativo considerar apenas os deslocados provenientes da ROB esquerda correspondente. Os deslocados de outras ROBs não são úteis para esta imagem.

A Fig. 4.7 mostra o resultado da comparação entre a versão 1 do MMP estéreo, que utilizava os deslocados apenas da ROB esquerda correspondente, e a versão 2, que acumula os deslocados de ROBs anteriores, para o par estéreo CORRIDOR.

Os resultados observados neste teste nos levaram a seguinte conclusão:

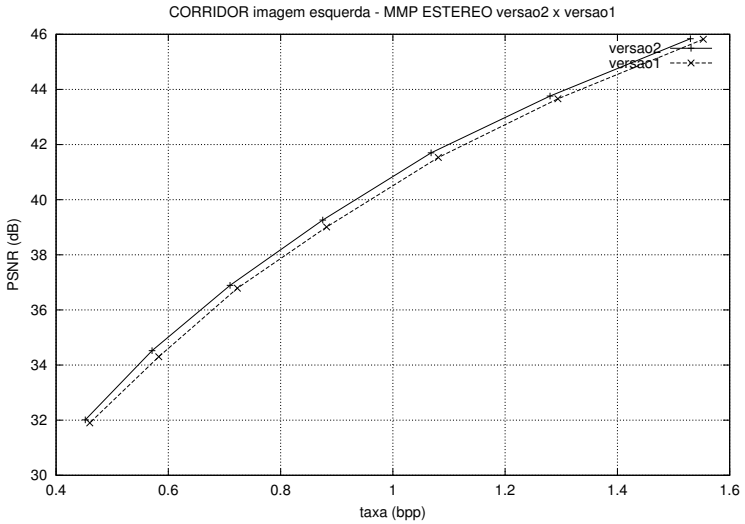
"**Inserir no dicionário elementos deslocados acumulados de ROBs anteriores é bom para a codificação da imagem esquerda, mas piora a codificação da imagem direita.**"

4.5 Versão 3 do MMP estéreo

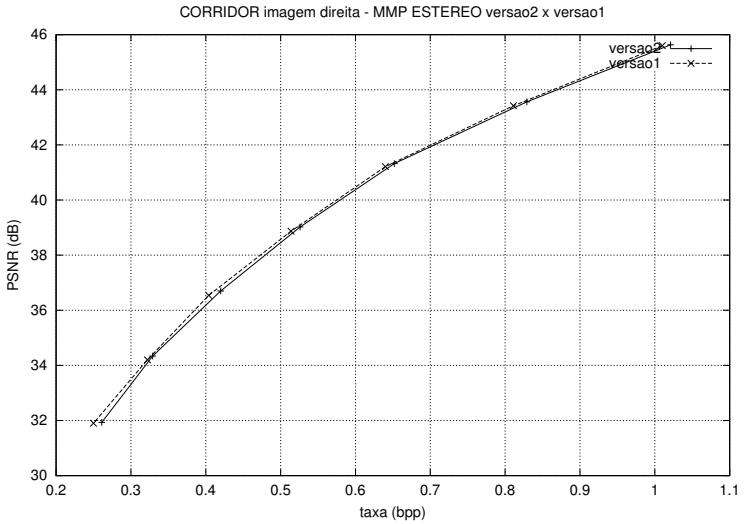
Levando-se em consideração a conclusão da seção anterior, baseada nos resultados das versões 1 e 2 do MMP estéreo, o ideal para a codificação da imagem estéreo com relação aos elementos deslocados seria então **"utilizar elementos deslocados acumulados para codificar a imagem esquerda, e utilizar elementos deslocados somente da ROB esquerda correspondente quando fosse codificar uma ROB da imagem direita"**.

Isto nos leva à seguinte solução:

Utilizar um dicionário para codificar a imagem esquerda, e utilizar um outro



(a)



(b)

Figura 4.7: Comparação entre as versões 2 e 1 do MMP estéreo, para a imagem CORRIDOR. (a) imagem esquerda, (b) imagem direita.

dicionário diferente para codificar a imagem direita.

Porém surge uma questão:

Mas, e os elementos que não são deslocados?

Eles são utilizados pelas duas imagens indiscriminadamente. A ocorrência de qualquer um destes elementos em uma das imagens deve ser contabilizada quando formos codificar a outra imagem. Portanto deve haver um **dicionário comum** para as duas imagens.

A solução proposta, implementada na versão 3 do MMP estéreo, considera então os seguintes dicionários:

1. **Dicionário Comum:** Possui os elementos iniciais, concatenados, contraídos e expandidos (elementos de comuns).

2. **Dicionário de Deslocados da ROB:** Possui os elementos deslocados gerados pela ROB esquerda correspondente à ROB direita atual que está sendo processada.

3. **Dicionário de Deslocados Acumulados:** Possui os elementos deslocados acumulados de ROBs anteriores.

Para codificar a imagem esquerda utilizamos um dicionário formado pela concatenação dos dicionários **Dicionário Comum** e **Dicionário de Deslocados Acumulados**. Chamamos este dicionário de **Dicionário 1**. Ele cresce à medida que codificamos cada par de ROBs da imagem. A tendência é que este dicionário fique muito grande porque além dos elementos comuns, os deslocados de cada ROB são acumulados ao longo do processamento.

Para codificar a imagem direita usaremos o **Dicionário Comum OU** o **Dicionário de Deslocados da ROB**. Neste caso, para sabermos se o índice do elemento escolhido pertence a um dicionário ou a outro, faremos uso de um *flag*. Este *flag* implica em um gasto extra na taxa de codificação da imagem direita. Este gasto, porém, é compensado pela alta ocorrência de elementos no dicionário de deslocados da ROB. Este fato não ocorre na codificação da imagem esquerda, por isto não se justifica separar os dicionários neste caso.

Para demonstrarmos isto, façamos as seguintes considerações:

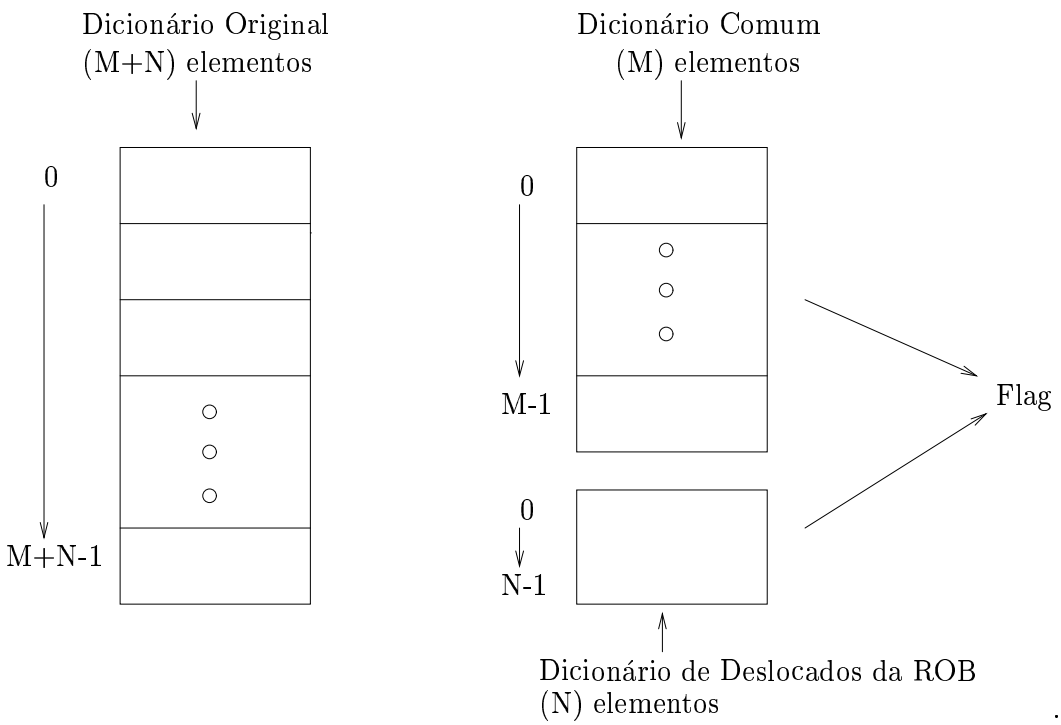


Figura 4.8: Dicionário original, com elementos comuns e deslocados da ROB. Dicionário original separado em dois: necessidade de um *flag*.

Considere um dicionário formado pelos elementos comuns e pelos deslocados da ROB. Este dicionário é denominado "Dicionário Original" na Fig. 4.8, e possui $(M+N)$ elementos, onde M é o total de elementos comuns e N é o total de elementos deslocados. Considere que M é muito maior do que N (**ATENÇÃO: esta é uma consideração importante!**). Se separarmos este dicionário em duas partes, teremos um dicionário que chamamos de "**Dicionário Comum**", com M elementos, e um outro dicionário que chamamos de "**Dicionário de Elementos Deslocados**", com N elementos, ambos mostrados na mesma figura à direita do Dicionário Original. Para utilizarmos os dois dicionários da direita separadamente precisamos de um *flag* para indicar qual dos dicionários foi utilizado. O *flag* é um bit ao qual se atribui o valor 0 quando o dicionário escolhido foi o Dicionário Comum, e 1 quando o dicionário escolhido foi o de Deslocados da ROB.

Sejam p_{f0} a probabilidade de ocorrência do *flag*=0, e p_{f1} a probabilidade de ocorrência do *flag*=1. Considere que $p_{f1} \gg p_{f0}$. (**ATENÇÃO: esta também é uma consideração importante!**). Isto significa que a ocorrência de elementos deslocados é muito maior do que a ocorrência de elementos comuns.

O número de bits gastos para se codificar os índices dos elementos do dicionário comum (usando um método de codificação com tamanho fixo) é dado por $\log_2 M$, e para o dicionário de deslocados da ROB é $\log_2 N$.

Sejam $\log_2 M = Q$ e $\log_2 N = W$. Como $M \gg N$ (lembre-se da consideração importante que chamamos a atenção anteriormente), podemos considerar que $\log_2(M+N) \simeq Q$. Este é o número de bits gastos (aproximadamente) para codificar o dicionário original, que possui elementos comuns e elementos deslocados.

O número de bits gastos quando se utiliza os dois dicionários separadamente, considerando-se os bits gastos com os flags e os bits gastos com os índices, é dado por:

$Nbits = (p_{f0} \log_2 M + p_{f1} \log_2 N) + (p_{f0} \log_2 \frac{1}{p_{f0}} + p_{f1} \log_2 \frac{1}{p_{f1}})$

Vamos chamar $(p_{f0} \log_2 \frac{1}{p_{f0}} + p_{f1} \log_2 \frac{1}{p_{f1}})$ de $NbitsFlags$.

Então, $Nbits = (Q \cdot p_{f0} + W \cdot p_{f1}) + NbitsFlags$

Seja $Q = W + Z$

Então, $Nbits = (W \cdot p_{f0} + Z \cdot p_{f0} + W \cdot p_{f1}) + NbitsFlags$

$Nbits = W(p_{f0} + p_{f1}) + Z \cdot p_{f0} + NbitsFlags$

Seja $p_{f0} + p_{f1} = 1$

Então, $Nbits = W + Z \cdot p_{f0} + NbitsFlags$

$Nbits$ tem que ser menor do que Q senão não vale a pena separar os dicionários.

Então, $Nbits = W + Z \cdot p_{f0} + NbitsFlags < Q$

$W + Z \cdot p_{f0} + (p_{f0} \log_2 \frac{1}{p_{f0}} + p_{f1} \log_2 \frac{1}{p_{f1}}) < Q$

$W + Z \cdot p_{f0} - p_{f0} \log_2 p_{f0} - (1 - p_{f0}) \log_2(1 - p_{f0}) < Q$

No limite, fazendo p_{f0} ir para 0, temos:

$\lim_{p_{f0} \to 0}(W) = W$

$\lim_{p_{f0} \to 0}(Z \cdot p_{f0}) = 0$

$\lim_{p_{f0}\to 0}(p_{f0}\log_2 p_{f0}) =$

$\lim_{p_{f0}\to 0}(\log_2 p_{f0} + p_{f0}\frac{1}{p_{f0}}) =$

$\lim_{p_{f0}\to 0}(\frac{-\log p_{f0}}{\frac{1}{p_{f0}}}) =$

$\lim_{p_{f0}\to 0}(\frac{\frac{-1}{p_{f0}}}{\frac{-1}{{p_{f0}}^2}}) =$

$\lim_{p_{f0}\to 0}(\frac{{p_{f0}}^2}{p_{f0}}) =$

$\lim_{p_{f0}\to 0}(p_{f0}) = 0$

$\lim_{p_{f0}\to 0}(1 - p_{f0})\log_2(1 - p_{f0}) = 0$

Então,

$Nbits = W < Q$

Portanto vale a pena usar dois dicionários, porque mesmo considerando os bits gastos com os flags, gastaremos menos bits do que usando um único dicionário. Vale ressaltar aqui que isto só é válido levando-se em conta duas considerações:

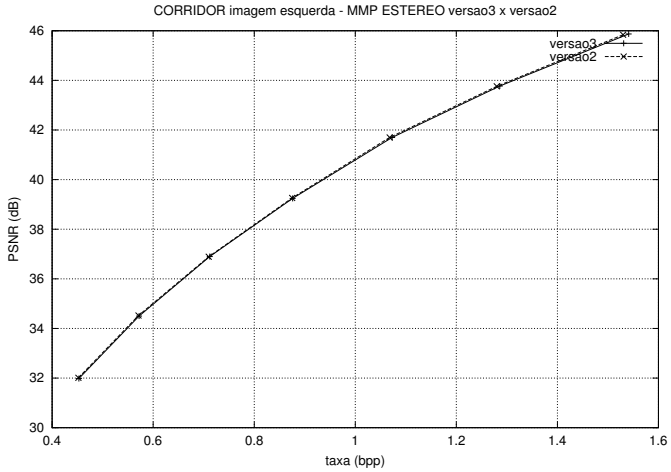
1. $M \gg N$
2. $p_{f1} \gg p_{f0}$

Os resultados comparativos entre as versões 3 e 2 do MMP estéreo para a imagem CORRIDOR são mostrados na Fig. 4.9.

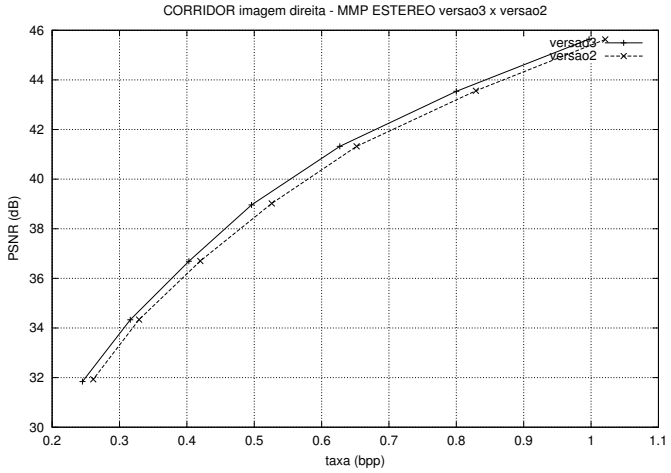
Verificamos, de fato, que a codificação da imagem direita melhorou em relação a versão anterior (versão 2) que utilizava apenas um dicionário e acumulava os elementos deslocados.

Comparando esta versão 3 que usa dicionários separados com a versão 1 do MMP estéreo, que usa o mesmo dicionário para as duas imagens, mas não acumula os elementos deslocados, verificamos que a versão 3 ganha na codificação da imagem esquerda. Os resultados comparativos das versões 3 e 1 para o par estéreo CORRIDOR são mostrados na Fig. 4.10.

Esta versão foi um marco importante no desenvolvimento desta tese. Conseguimos melhorar a codificação da imagem esquerda, introduzindo elementos deslocados acumulados no dicionário, e conseguimos melhorar a codificação da imagem direita, introduzindo no dicionário somente os elementos deslocados da ROB esquerda correspondente. A partir de agora utilizaremos esta versão como base para introduzir outras idéias que possam surgir para melhorar a compressão de imagens estéreo.

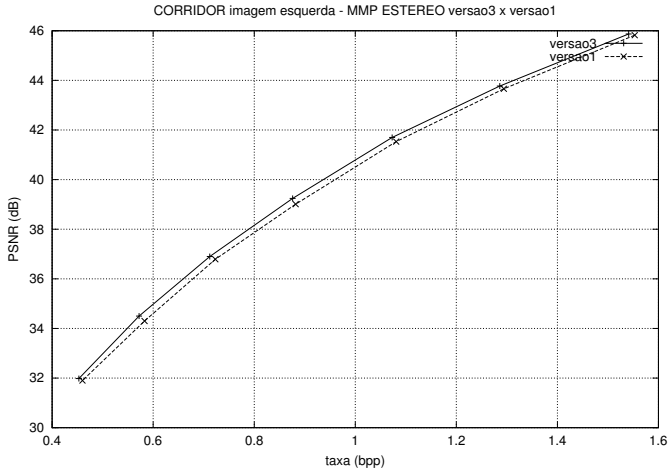


(a)

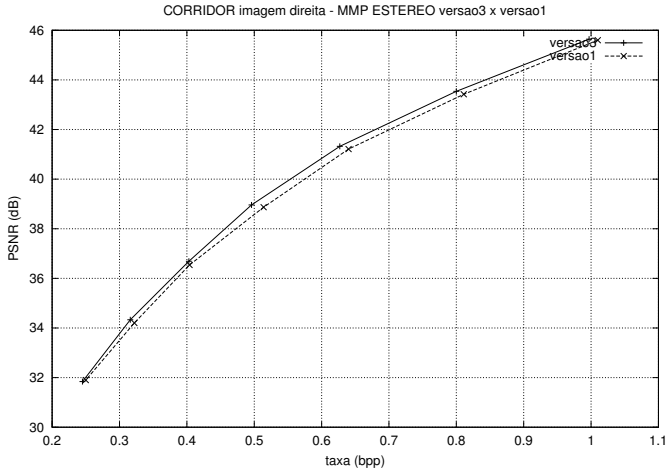


(b)

Figura 4.9: Comparação entre as versões 3 e 2 do MMP estéreo, para a imagem CORRIDOR. (a) imagem esquerda, (b) imagem direita.



(a)



(b)

Figura 4.10: Comparação entre as versões 3 e 1 do MMP estéreo, para a imagem CORRIDOR. (a) imagem esquerda, (b) imagem direita.

4.6 Versão 4 do MMP estéreo

Dado um determinado bloco da imagem direita, procuramos na imagem esquerda um bloco correspondente. A região da imagem esquerda onde realizamos esta procura é a ROB correspondente, de acordo com a restrição da linha epipolar. O mapa de disparidades obtido entre duas imagens de um par estéreo contém valores dentro de uma faixa denominada "janela de busca". Esta janela de busca limita a região da imagem esquerda na qual podemos encontrar um bloco correspondente.

Na versão anterior, os elementos deslocados foram gerados em toda a extensão da ROB esquerda. Isto faz com que o número de elementos deslocados inseridos no dicionário para codificar a imagem direita seja muito grande, e contenha elementos fora da janela de busca, que não seriam blocos correspondentes ao bloco da imagem direita que está sendo processado.

Nesta nova versão (4), geramos os elementos deslocados apenas dentro da janela de busca equivalente ao bloco da imagem direita que está sendo processado. Com isso melhoramos a codificação da imagem direita porque inserimos no dicionário apenas elementos deslocados que poderão corresponder ao bloco que está sendo processado, ou seja, melhoramos a estimativa do modelo que irá representar a imagem direita.

Esta melhoria pode ser observada no gráfico da Fig. 4.11.

4.7 Versão 5 do MMP estéreo

Na versão 3 os elementos deslocados foram formados por um deslocamento de 1 pixel. Em codificação de vídeo monocular foi comprovado que utilizar estimação de movimento de 1/2 pixel melhora o resultado da codificação, porque diminui a diferença entre o bloco original e o bloco estimado. Resolvemos então, nesta versão 5, gerar elementos deslocados de 1/2 pixel, para testar se isto melhoraria os resultados de compressão da imagem direita. Implementamos esta versão tomando como base a versão 3, para sabermos isoladamente o benefício de usar deslocados de 1/2 pixel, sem usar janela de busca.

O resultado comparando as versões 5 e 3 para a imagem CORRIDOR são mostrados na Fig. 4.12.

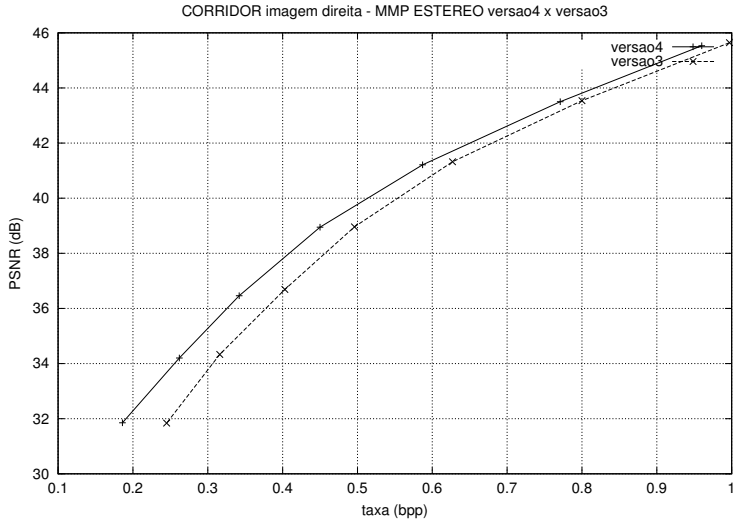


Figura 4.11: Comparação entre as versões 4 e 3 do MMP estéreo, para a imagem direita do par estéreo CORRIDOR.

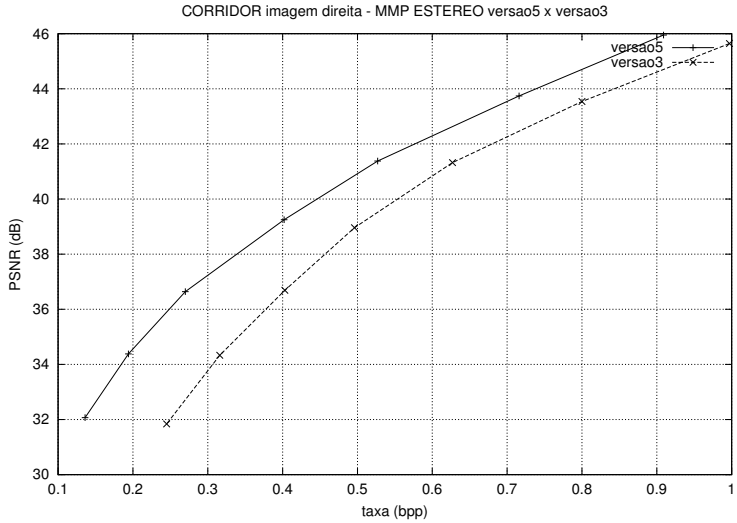


Figura 4.12: Comparação entre as versões 5 e 3 do MMP estéreo, para a imagem direita do par estéreo CORRIDOR.

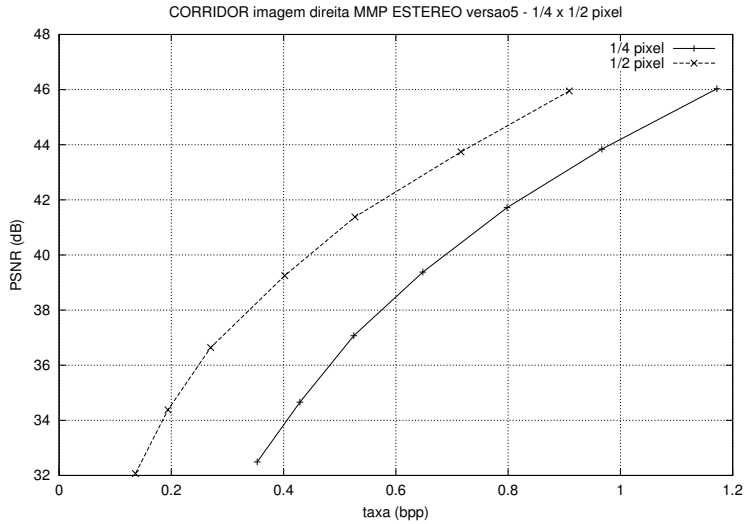


Figura 4.13: Comparação entre usar deslocamentos de 1/4 e 1/2 de pixel, para a imagem direita do par CORRIDOR.

Testamos também gerar deslocamentos de 1/4 de pixel, mas isto piorou o resultado porque aumentamos muito o número de elementos do dicionário com elementos que não agregavam nenhum ganho significativo em relação à precisão na representação da imagem, comparado com os elementos de 1/2 pixel. Por isto adotamos o deslocamento de 1/2 pixel como a melhor opção nas versões de agora em diante.

O resultado comparando o deslocamento de 1/4 de pixel com o deslocamento de 1/2 pixel é mostrada na Fig. 4.13, para a imagem CORRIDOR.

4.8 Versão 6 do MMP estéreo

Resolvemos incorporar todas as melhorias implementadas e mostradas até agora neste capítulo, que são enumeradas a seguir, e criamos a versão 6 do MMP estéreo [37, 38].

1. Gerar e incluir elementos deslocados no dicionário (versão 1). Isto melhorou os resultados de compressão da imagem direita, porque introduzimos no dicionário elementos relacionados com a disparidade existente entre as duas imagens estéreo.

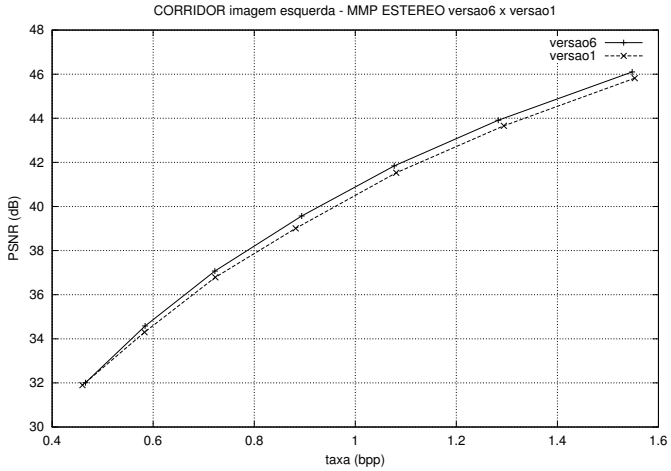
2. Utilizar um dicionário com elementos comuns e elementos deslocados acumulados de ROBs anteriores para codificar a imagem esquerda (versão 2). Isto melhorou a codificação da imagem esquerda, porque introduzimos no dicionário elementos que representavam similaridades existentes entre regiões da mesma imagem.

3. Escolher a melhor opção, do ponto de vista do compromisso entre taxa e distorção, entre utilizar um dicionário com elementos comuns ou um dicionário com elementos deslocados da ROB esquerda correspondente, para codificar um bloco da imagem direita (versão 3). Isto melhorou consideravelmente a codificação da imagem direita, porque utilizamos um dicionário com poucos elementos muito significativos para a imagem direita.

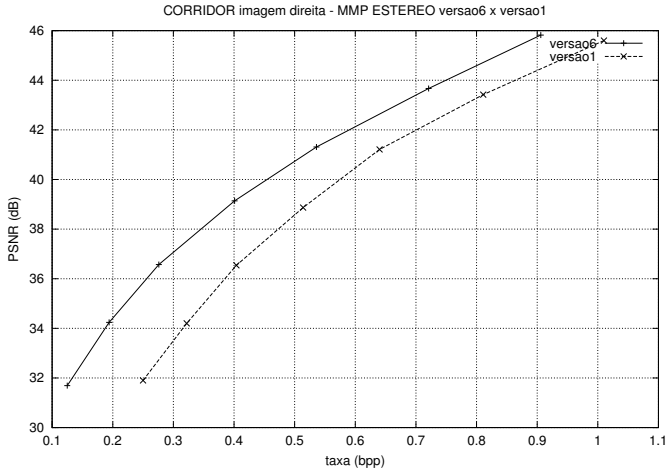
4. Limitar a geração de elementos deslocados dentro de uma janela de busca baseada na faixa de valores de disparidade da imagem estéreo que está sendo codificada (versão 4). Isto melhorou a codificação da imagem direita porque reduzimos ainda mais o número de elementos do dicionário de deslocados, deixando somente aqueles elementos que realmente são importantes para a codificação do bloco da imagem direita.

5. Gerar elementos deslocados de 1/2 pixel (versão 5). Isto melhorou muito a codificação da imagem direita, porque aumentamos a precisão dos elementos deslocados, reduzindo assim a distorção da imagem direita reconstruída.

Os resultados comparativos entre as versões 6 e 1 são mostrados na Fig. 4.14, para a imagem estéreo CORRIDOR, na Fig. 4.15 para a imagem AQUA, na Fig. 4.16 para a imagem MAN, na Fig. 4.17 para a imagem SAXO, e na Fig. 4.18 para a imagem TRAIN.

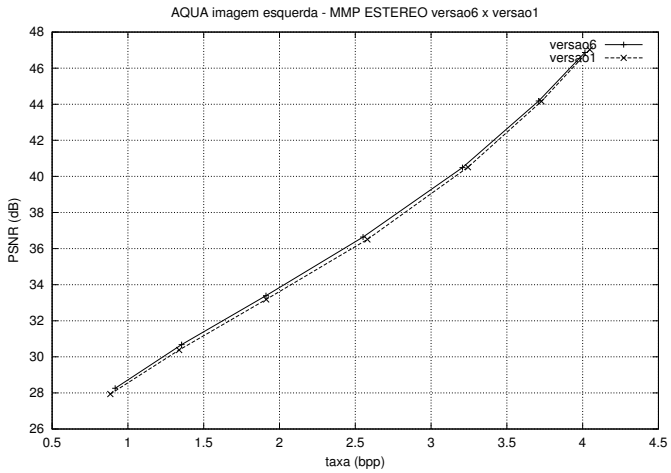


(a)

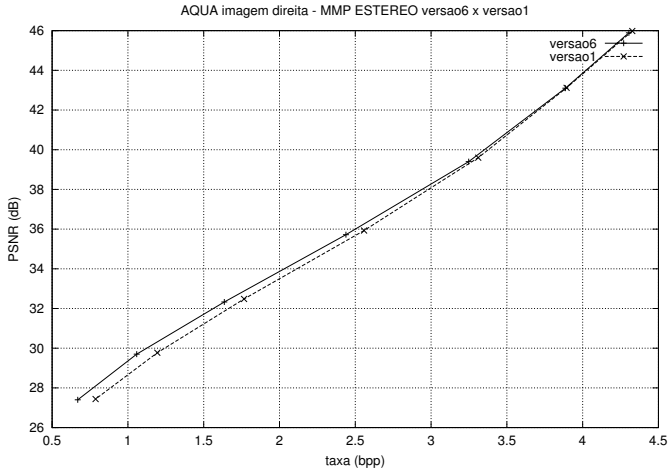


(b)

Figura 4.14: Comparação entre as versões 6 e 1 do MMP estéreo. (a) imagem esquerda, e (b) imagem direita do par estéreo CORRIDOR.

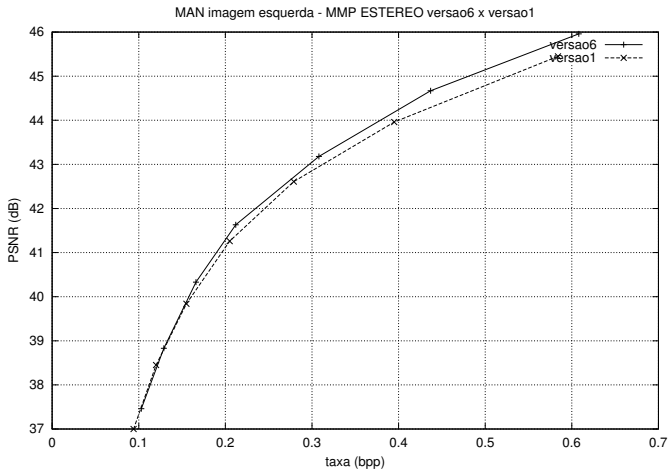


(a)

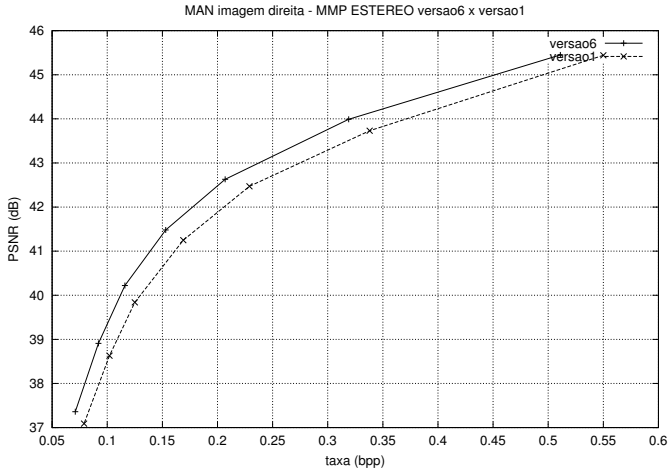


(b)

Figura 4.15: Comparação entre as versões 6 e 1 do MMP estéreo. (a) imagem esquerda, e (b) imagem direita do par estéreo AQUA.

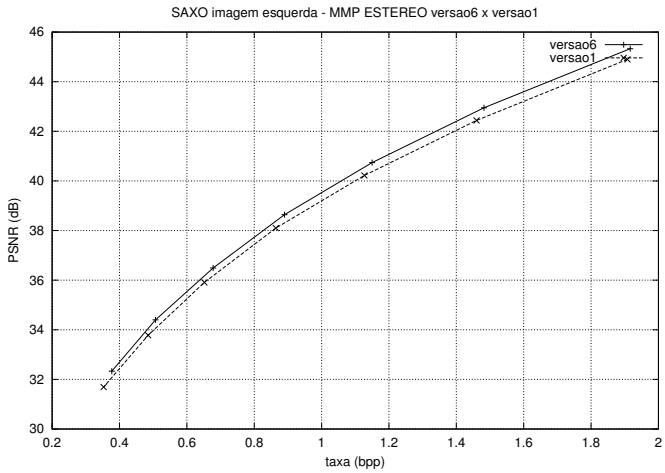


(a)

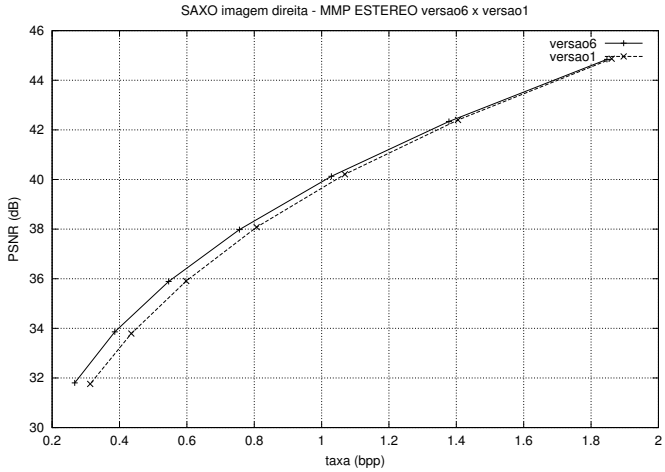


(b)

Figura 4.16: Comparação entre as versões 6 e 1 do MMP estéreo. (a) imagem esquerda, e (b) imagem direita do par estéreo MAN.

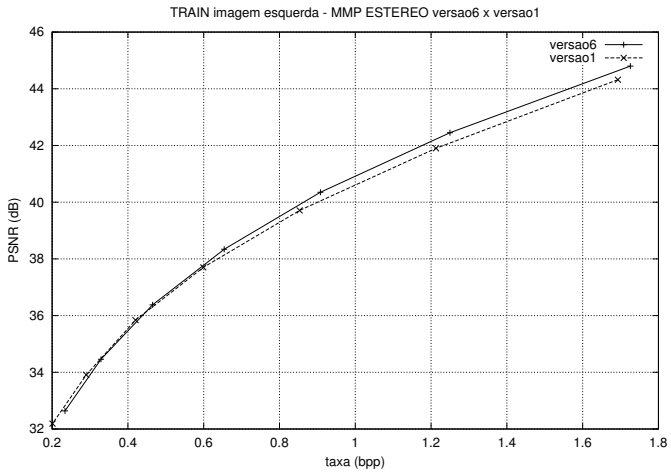


(a)

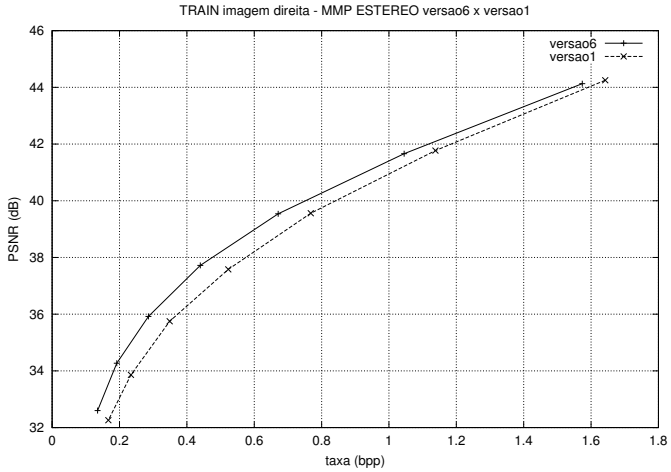


(b)

Figura 4.17: Comparação entre as versões 6 e 1 do MMP estéreo. (a) imagem esquerda, e (b) imagem direita do par estéreo SAXO.



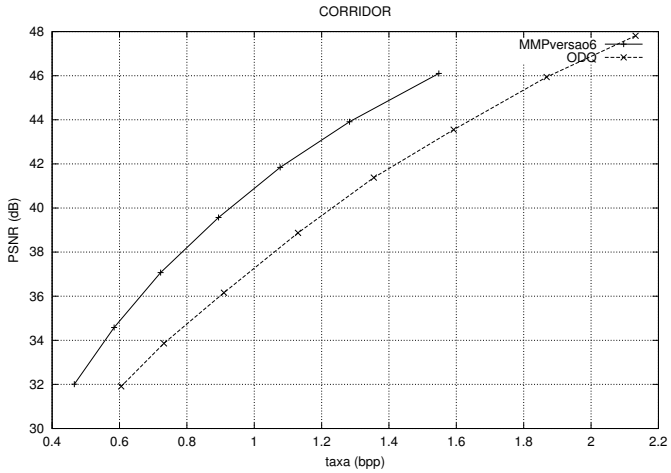
(a)



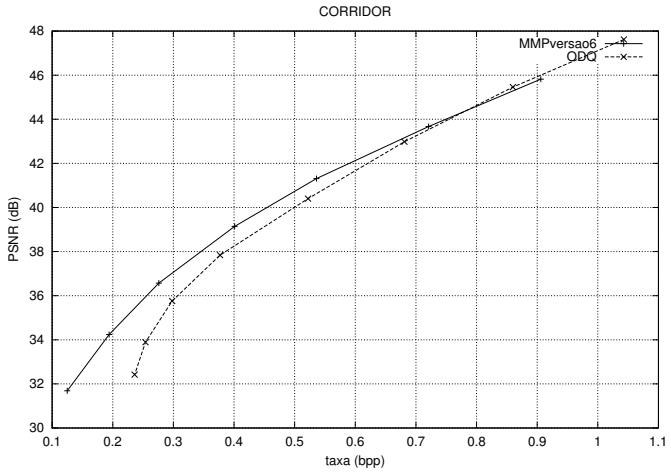
(b)

Figura 4.18: Comparação entre as versões 6 e 1 do MMP estéreo. (a) imagem esquerda, e (b) imagem direita do par estéreo TRAIN.

4.9 Comparação entre MMP versão 6 e ODQ

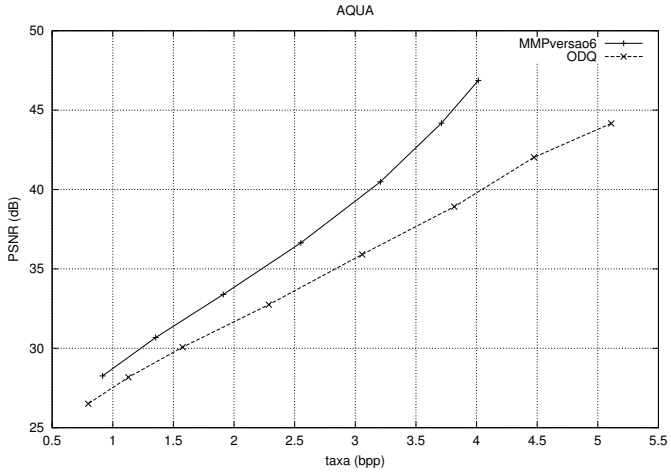


(a)

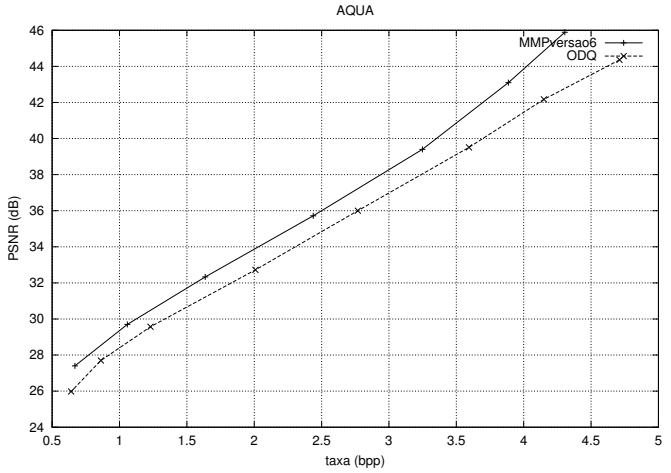


(b)

Figura 4.19: Comparação entre MMP estéreo versão 6 e ODQ, para o par estéreo CORRIDOR. (a) imagem esquerda, (b) imagem direita.



(a)

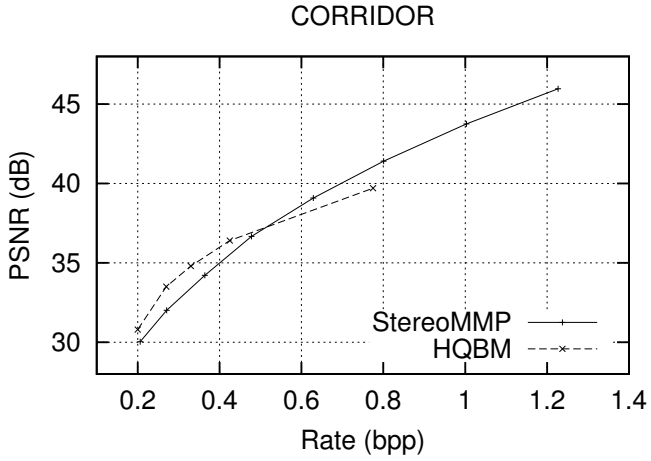


(b)

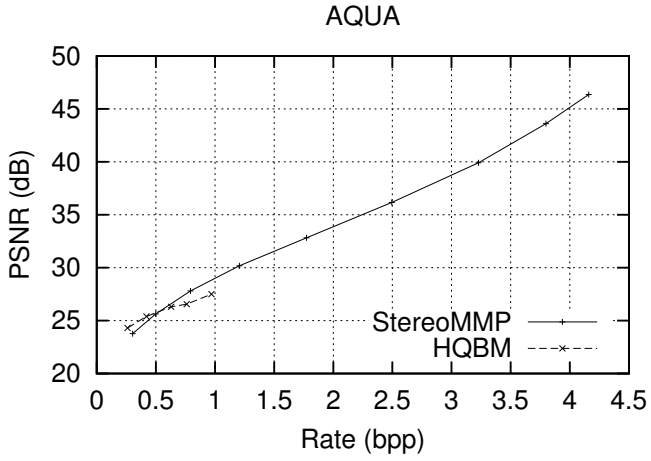
Figura 4.20: Comparação entre MMP estéreo versão 6 e ODQ, para o par estéreo AQUA. (a) imagem esquerda, (b) imagem direita.

4.10 Comparação entre MMP versão 6 e ODQ melhorado

Em [39] Woontack Woo e Ortega apresentam uma nova versão do ODQ utilizando modelo de Markov e casamento de blocos com divisão hierárquica (*hierarquical quadtree*). Chamamos esta nova versão do ODQ de HQBM (*Hierarquical quadtree block matching*). A comparação com esta nova versão é mostrada na Fig. 4.21. Só dispunhamhos dos valores de PSNR para estas duas imagens, e em baixas taxas, conforme os resultados apresentados em [39], por isto não pudemos comparar os resultados para taxas maiores e para outras imagens. Os gráficos indicam que o MMP estéreo ganha do HQBM para valores de taxa acima de 0,5 bpp. Taxas abaixo deste valor não são interessantes para a maioria das aplicações porque implicam em uma imagem reconstruída com qualidade ruim (por exemplo, abaixo de 26 dB para a imagem AQUA, que é uma imagem real).



(a)



(b)

Figura 4.21: Comparação entre MMP estéreo versão 6 e HQBM, para o par estéreo (a) CORRIDOR e (b) AQUA.

4.11 Conclusão

Neste capítulo descrevemos a primeira parte do desenvolvimento desta tese, que corresponde às primeiras implementações do MMP explorando a similaridade existente entre duas imagens estéreo. A novidade desta parte do trabalho é que não calculamos o mapa de disparidades. Com isto evitamos os problemas envolvidos com o cálculo e a transmissão destes vetores. Os vetores de disparidade representam as correspondências existentes entre as duas imagens, e quanto mais preciso for o cálculo dos vetores de disparidade, melhor será a estimação de uma imagem baseada na outra. Porém, quanto maior a precisão do mapa, mais bits serão necessários para o armazenamento e a transmissão desta informação. Conseguimos substituir o uso do mapa de disparidades através da inclusão no dicionário de elementos relacionados com o deslocamento de um bloco da imagem direita em relação ao bloco correspondente da imagem esquerda. Com isto, atingimos um resultado melhor na compressão da imagem direita, comparado com a versão do MMP que não usava estes elementos, sem no entanto ter que calcular nem transmitir o mapa de disparidades. Os elementos deslocados inseridos no dicionário representam indiretamente os vetores de disparidade, porém considera-se apenas o caso particular de o objeto 3D estar à uma distância muito grande das duas câmeras, o que não corresponde à realidade na maioria dos casos. Pudemos notar que a inserção dos elementos deslocados melhorou bastante a compressão da imagem direita, porém precisamos descobrir uma maneira de representar melhor as disparidades, e inserir no dicionário elementos mais significativos, que representem por exemplo, as distorções de perspectiva, que são efeitos bastante comuns causados principalmente pela distância entre as câmeras estéreo. Os deslocamentos simples não representam efeitos como estes, e por isto, para imagens como a MAN estes elementos não foram suficientes para melhorar a codificação da imagem direita. No próximo capítulo iremos descrever as implementações neste sentido que caracterizaram a segunda parte do desenvolvimento desta tese.

Capítulo 5

MMP estéreo com cálculo do mapa de disparidades

5.1 Introdução

No capítulo 4 nós mostramos que a introdução de elementos deslocados no dicionário melhorou a codificação das imagens estéreo testadas. Porém estes deslocamentos consideravam apenas um caso particular no qual todos os objetos da cena se encontravam aproximadamente a uma mesma distância das duas câmeras, ou seja, a uma distância tendendo ao infinito, o que geralmente não ocorre na maioria das imagens estéreo reais.

Nas maior parte das imagens estéreo, a distância de um determinado ponto de um objeto da cena 3D até a câmera esquerda difere da distância do mesmo ponto até a câmera direita. Esta diferença de distâncias pode ser estimada através dos vetores de disparidade. Para fazer uma estimação precisa, que resulta em uma melhor qualidade da imagem reconstruída, muitas restrições devem ser atendidas, o que resulta em um processo razoavelmente complexo. No nosso caso estamos interessados em uma aproximação do mapa de disparidades, portanto para isto apenas fazemos uma estimação pixel a pixel, o que nos fornece um mapa denso e uma aproximação bem mais precisa e consistente do que a estimação bloco a bloco.

Uma vez calculado o mapa de disparidades entre duas imagens estéreo, pode-se estimar uma das imagens (geralmente a imagem direita) a partir da outra imagem (no caso, a esquerda) e do mapa de disparidades.

A principal desvantagem das técnicas que utilizam o mapa de disparidades para estimar a imagem direita a partir da esquerda é a necessidade de enviar o mapa de disparidades. Quanto mais preciso o mapa, maior o *overhead* para transmiti-lo.

A nossa idéia é calcular um mapa de disparidades denso, e utilizá-lo, juntamente com o bloco equivalente da imagem esquerda, para gerar elementos no dicionário que possam representar estimativas bastante aproximadas de blocos da imagem direita. Esperamos que estes elementos gerados a partir das informações precisas dos vetores de disparidades possam representar efeitos como a distorção de perspectiva causada pela distância entre as duas câmeras estéreo. Inserindo estes elementos no dicionário, esperamos melhorar a compressão da imagem direita, uma vez que teremos elementos representando aproximações mais precisas dos blocos da imagem direita.

5.2 Como utilizar (sem transmitir) a informação dos vetores de disparidade

O primeiro teste realizado foi calcular um mapa de disparidades denso entre duas imagens estéreo. A obtenção do mapa de disparidades foi feita pixel a pixel, como mostra a Fig. 5.1. Para um determinado pixel da imagem direita, procura-se numa região da imagem esquerda o pixel correspondente, aquele que resultou no menor erro médio quadrático entre as duas regiões em torno do pixel considerado.

Pegamos a imagem esquerda e aplicamos o mapa de disparidades resultante, para obter uma estimativa da imagem direita (Veja Fig. 5.2). Para o primeiro pixel a ser estimado para a imagem direita, verificamos qual o valor da disparidade no mapa de disparidades, na mesma posição do pixel que queremos estimar na imagem direita. Encontramos neste exemplo o valor 3. Isto indica que, partindo da mesma posição do pixel a ser estimado, nos deslocamos 3 posições à direita na imagem esquerda. O valor do pixel encontrado nesta imagem será o valor do pixel estimado na imagem direita. Para o segundo pixel da imagem direita, encontramos o valor -1 no mapa de disparidades. Isto indica que devemos nos deslocar 1 posição à esquerda da posição do pixel que está sendo estimado, na imagem esquerda. Encontramos então o valor estimado do pixel da imagem direita. O mesmo processo se repete até

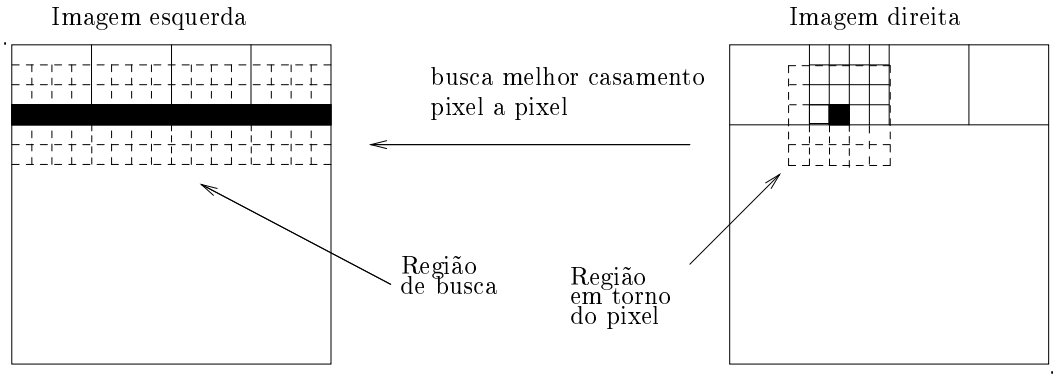


Figura 5.1: Para um determinado pixel da imagem direita, procura-se numa região da imagem esquerda o pixel correspondente.

estimarmos a imagem direita completa. Neste exemplo utilizamos valores inteiros para as disparidades, porém estes valores podem ser decimais, pois a busca foi feita com precisão de até 1/4 de pixel. Podem ocorrer "buracos" no mapa de disparidades por não encontrarmos um casamento pefeito dentro desta precisão utilizada. Quando isto ocorre, fazemos uma interpolação aproximando o valor de disparidade encontrado para o valor mais próximo. Por exemplo, se a disparidade encontrada foi 1/27, este valor não existe na precisão que nós utilizamos (1/25), então aproximamos para 1/25 e achamos o bloco correspondente. O tamanho do bloco utilizado para fazer a busca foi 17x3.

Obtivemos uma imagem reconstruída similar à imagem direita original. Isto significa que, se calculássemos e enviássemos um mapa de disparidades preciso para o decodificador, seria possível reconstruir a imagem direita com uma qualidade muito boa. Porém, o alto custo para enviar este mapa de disparidades não compensa a boa qualidade da imagem reconstruída.

A nossa idéia inicial foi então calcular o mapa de disparidades apenas entre o primeiro par de ROBs correspondentes do par estéreo, e assim transmitir ao decodificador somente esta informação (ver Fig. 5.3). Utilizaríamos este mapa de disparidades para gerar os elementos no dicionário que seriam utilizados para codificar a próxima ROB direita.

Realizamos então o seguinte teste: Calculamos um mapa de disparidades en-

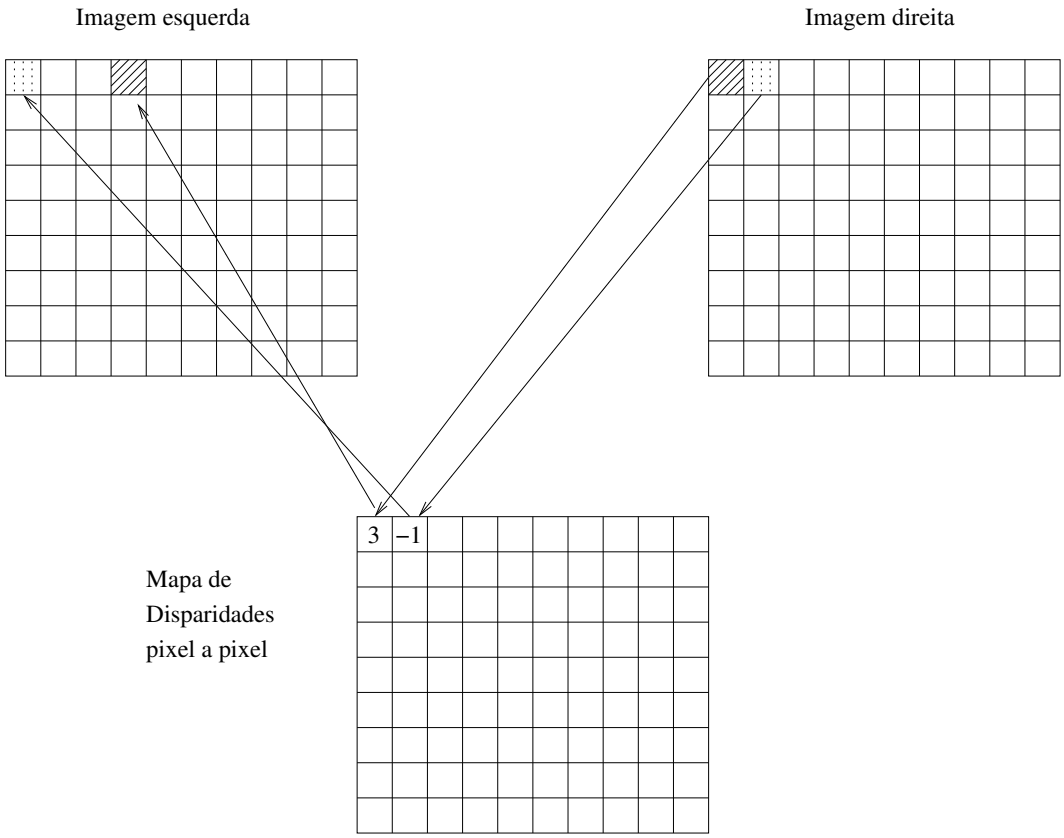


Figura 5.2: Reconstrução da imagem direita a partir da imagem esquerda e do mapa de disparidades.

tre o primeiro par de ROBs. Enviamos este mapa ao decodificador. Aplicamos este mapa de disparidades à próxima ROB esquerda e obtemos uma estimativa aproximada da próxima ROB direita. Copiamos os blocos desta ROB estimada para o dicionário que será utilizado para codificar a próxima ROB direita (veja a Fig. 5.4). Nós chamamos estes elementos do dicionário de "elementos deformados". Estes elementos na verdade são uma estimativa dos blocos da ROB direita, estimativa esta baseada na ROB esquerda correspondente e no mapa de disparidades obtido entre as ROBs anteriores. Eles podem ser bastante úteis para a codificação da ROB direita seguinte.

Assim, ao longo da imagem, utilizamos o mapa de disparidades calculado entre um par de ROBs processadas para codificar a próxima ROB direita (Veja Fig. 5.5). Este teste apresentou bons resultados no PSNR final, com a vantagem de não onerar muito a taxa, pois só enviamos o mapa de disparidades de um par de

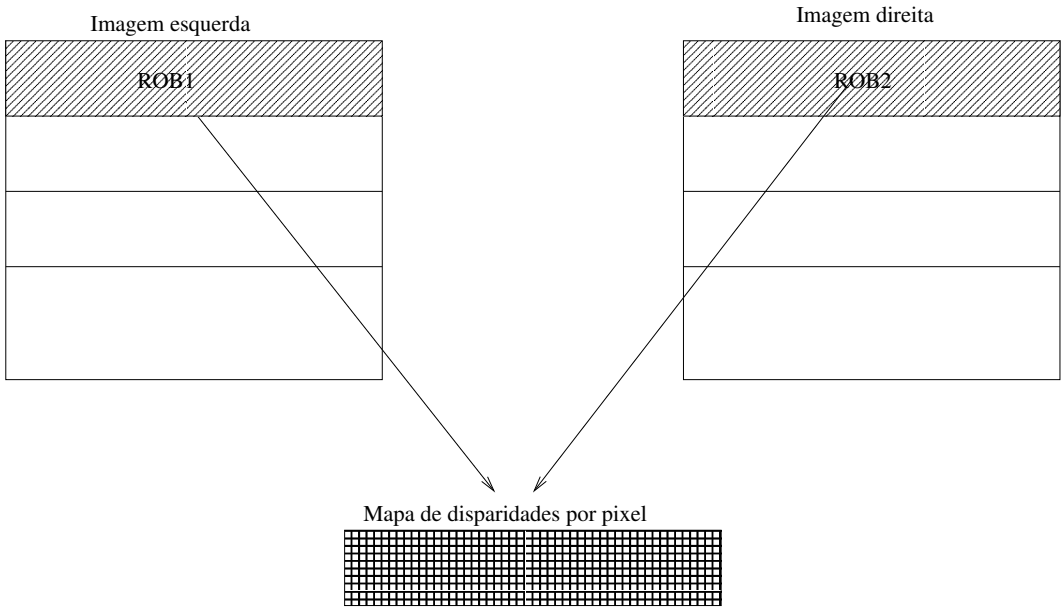


Figura 5.3: Cálculo e transmissão do mapa de disparidades apenas para o primeiro par de ROBs correspondentes.

ROBs (o primeiro par).

No entanto, prosseguimos com a pesquisa porque queríamos continuar na mesma linha de não enviar nenhuma informação relacionada com o mapa de disparidades para o decodificador. A solução então foi não utilizar o mapa de disparidades para codificar o primeiro par de ROBs. Codificamos este primeiro par de ROBs utilizando um dicionário apenas com elementos comuns. Uma vez codificado o primeiro par de ROBs correspondentes, decodificamos estas ROBs e calculamos o mapa de disparidades entre o par de ROBs reconstruído. Este mapa de disparidades pode ser calculado também pelo decodificador, pois este pode reconstruir o primeiro par de ROBs a partir do dicionário de elementos comuns, isto é, não precisa receber nenhuma informação sobre o mapa de disparidades. Utilizaremos este mapa de disparidades calculado entre as duas ROBs reconstruídas, para gerar elementos que serão utilizados para codificar a próxima ROB direita. Veja a Fig. 5.6.

Esta foi a idéia que prevaleceu, e deu origem à versão 7 do MMP estéreo, descrita na próxima seção deste capítulo.

Após processarmos um par de ROBs correspondentes, calculamos o mapa de

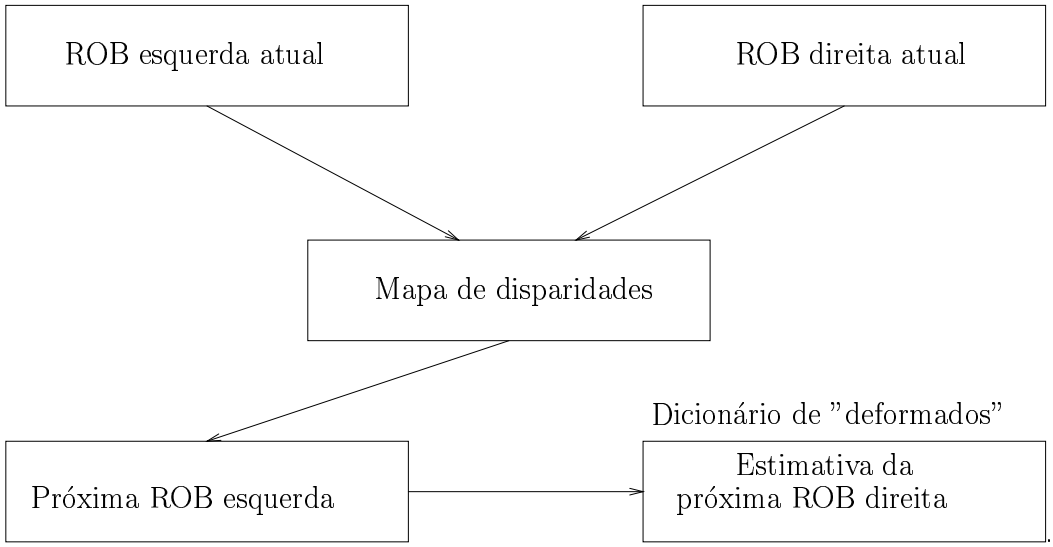


Figura 5.4: Como são gerados os elementos "deformados".

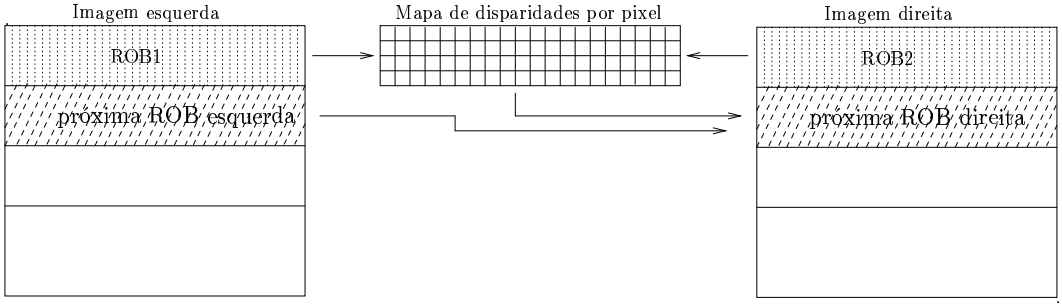


Figura 5.5: Cálculo do mapa de disparidades entre duas ROBs correspondentes, que será utilizado para estimar a próxima ROB direita.

disparidades por pixel entre estas duas ROBs. Para processarmos o próximo par de ROBs, utilizamos o mapa de disparidades obtido entre as ROBs anteriores e geramos elementos para o dicionário, que serão utilizados para codificar a ROB direita. Com estes elementos no dicionário esperamos melhorar a codificação da imagem direita.

Para introduzir estes novos elementos que chamamos de "deformados", pegamos como base a versão 6 do MMP estéreo. Removemos os elementos deslocados do dicionário usado para codificar a imagem direita, e removemos também os deslocados acumulados do dicionário usado para codificar a imagem esquerda. Isto nos

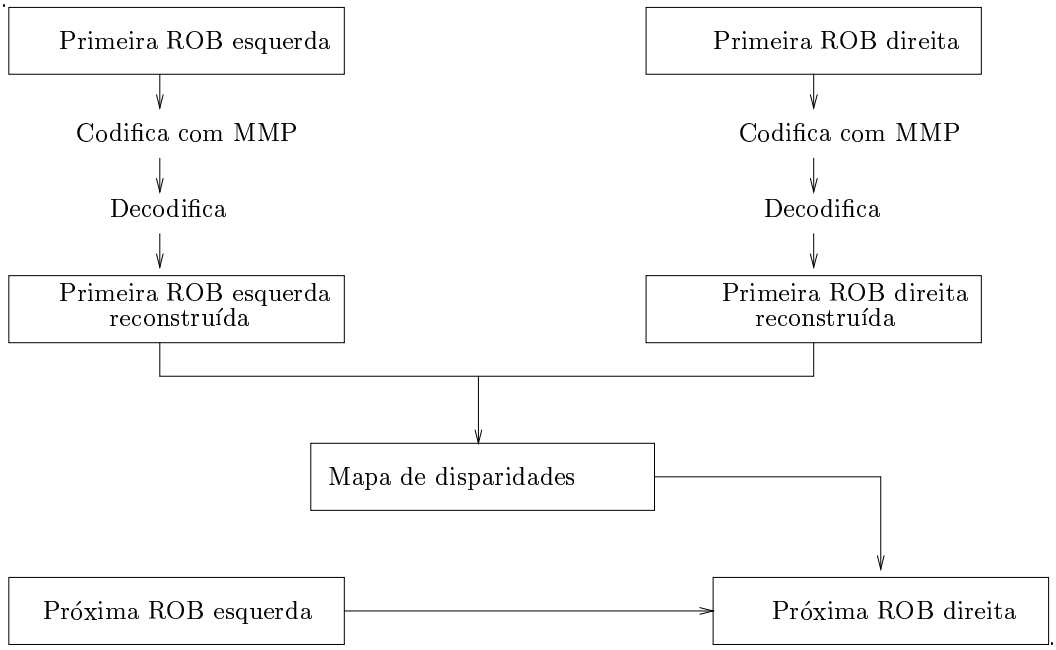


Figura 5.6: Codificamos o primeiro par de ROBs usando um dicionário comum. Reconstruímos este par de ROBs e calculamos o mapa de disparidades, que será utilizado para codificar a próxima ROB direita.

permite isolar os elementos deformados dos deslocados e fazer um estudo detalhado do efeito dos elementos deformados na codificação da imagem direita do par estéreo. Esta nova versão é a versão 7, descrita a seguir.

Os pares de imagens estéreo utilizadas para os testes deste capítulo foram CORRIDOR, AQUA, SAXO, MAN e TRAIN (veja o Apêndice B).

5.3 Versão 7 do MMP estéreo

A versão 7 do MMP estéreo tem como base a versão 6 do MMP estéreo, descrita no Cap. 4, seção 4.8. Na versão 6 tínhamos um dicionário utilizado para codificar a imagem esquerda, composto por elementos comuns e elementos deslocados acumulados de ROBs anteriores. Para codificar a imagem direita tínhamos a opção de usar um dicionário com elementos comuns ou um dicionário com elementos deslocados da ROB esquerda correspondente. Na versão 7 removemos os elementos deslocados acumulados do dicionário usado para a imagem esquerda, e substituí-

mos os elementos deslocados da ROB esquerda pelos elementos "deformados" no dicionário usado para a imagem direita. Fizemos isto para podermos observar isoladamente os efeitos de utilizar os elementos "deformados" para codificar a imagem direita do par estéreo.

Após processarmos um par de ROBs correspondentes, reconstruímos este par de ROBs e calculamos o mapa de disparidades por pixel entre estas duas ROBs. Para processarmos o próximo par de ROBs, utilizamos o mapa de disparidades obtido entre as ROBs anteriores e geramos elementos para o dicionário, que serão utilizados para codificar a ROB direita. Com estes elementos no dicionário esperamos melhorar a codificação da imagem direita.

Os vetores de disparidade são calculados a partir de um par de ROBs correspondentes reconstruídas. Fazemos isto para tornar possível o mesmo procedimento por parte do decodificador. O próprio mapa de disparidade obtido entre um par de ROBs correspondentes pode ser utilizado diretamente para gerar os elementos a serem usados na codificação da ROB direita seguinte (Veja a Fig. 5.7), mas esta é uma estimativa grosseira, porque as disparidades entre um par de ROBs correspondentes podem ser muito diferentes das disparidades do par de ROBs seguinte. Como uma tentativa de melhorar esta estimativa, consideramos as seguintes opções:

1. Repetir a última linha de disparidades obtidas das ROBs anteriores (Veja a Fig. 5.8). Esta ainda é uma estimativa grosseira, mas já ameniza a mudança abrupta dos valores de disparidade ocorrida principalmente na borda da linha de blocos, quando se utilizava diretamente o mapa calculado entre as ROBs anteriores.

2. Obter um média simples das colunas das disparidades obtidas das ROBs anteriores (Veja a Fig. 5.9). Esta é uma medida que suaviza a estimativa do mapa a ser usado.

3. Obter uma média gradual das colunas, dando um peso maior aos valores mais próximos da vizinhança das ROBs e um peso menor aos valores mais distantes (Veja a Fig. 5.10).

4. Obter uma média móvel (de dois ou três pixels) das colunas das disparidades obtidas das ROBs anteriores (Veja a Fig. 5.11).

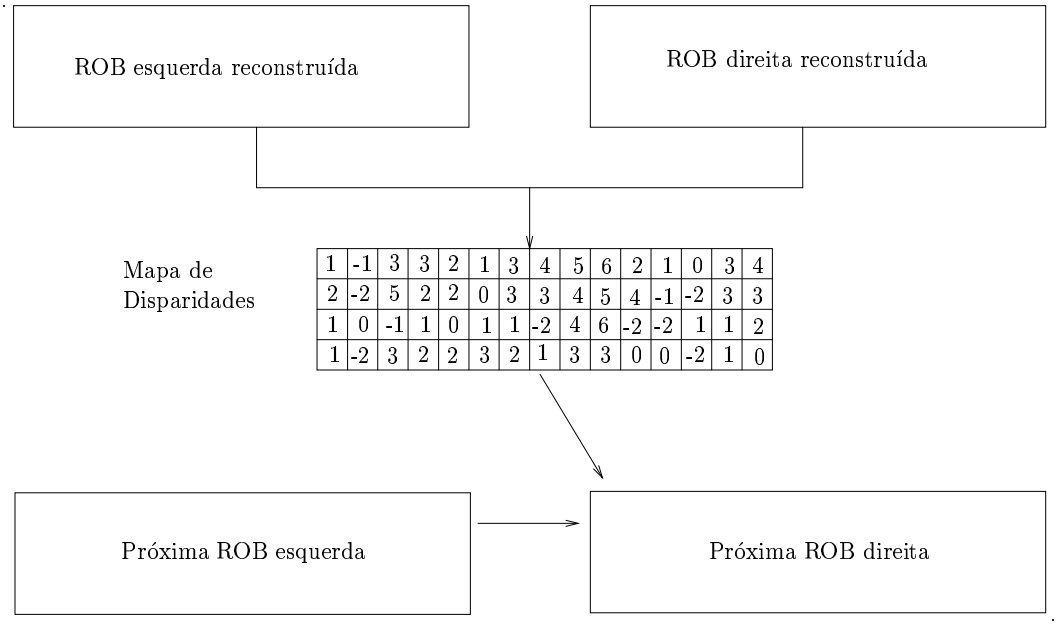


Figura 5.7: Uso do próprio mapa de disparidades calculado entre duas ROBs correspondentes, para gerar elementos "deformados" para codificar a próxima ROB direita.

Nos testes realizados a estimativa que deu melhor resultado de taxa×distorção foi a que utiliza a **média simples** das colunas das disparidades obtidas das ROBs anteriores, e portanto esta será a alternativa utilizada daqui por diante.

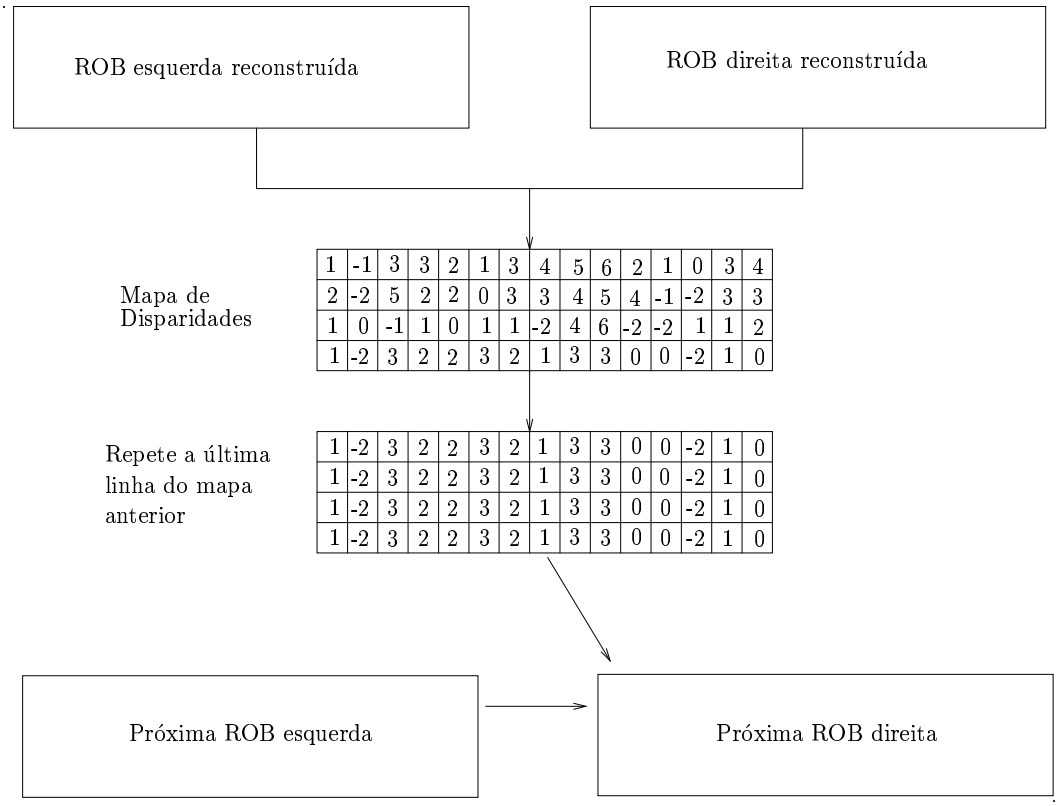


Figura 5.8: Repetição da última linha do mapa de disparidades calculado entre as ROBs anteriores.

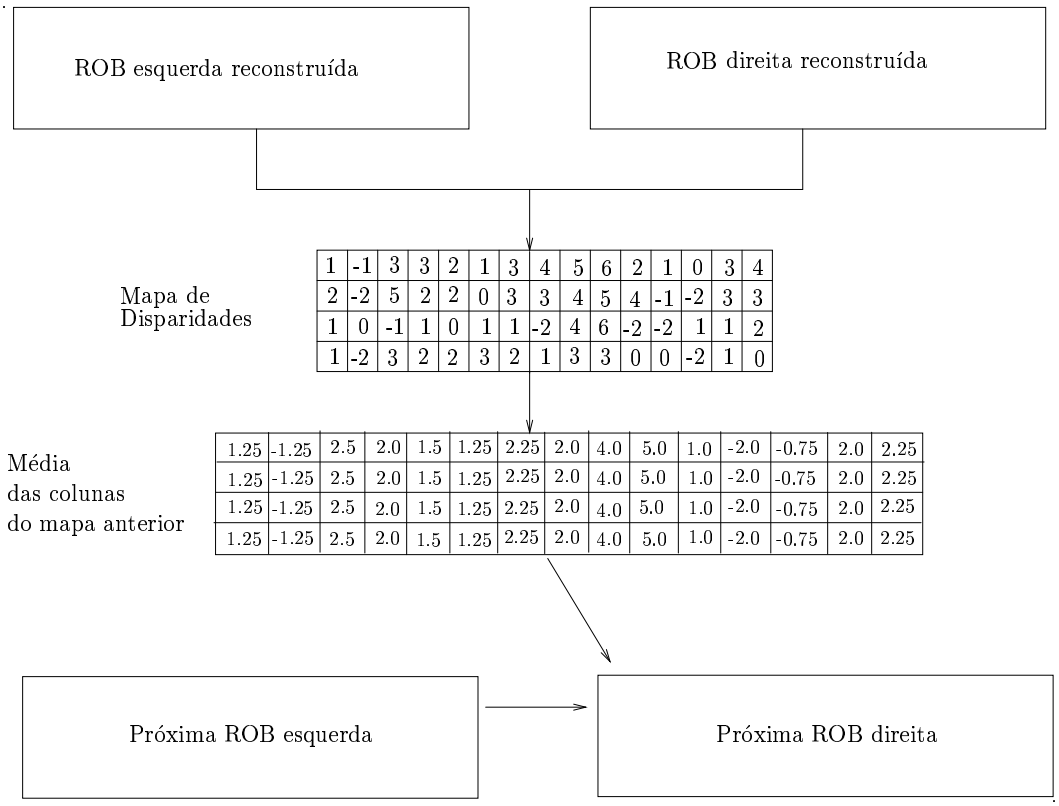


Figura 5.9: Média simples das colunas do mapa calculado entre as ROBs anteriores.

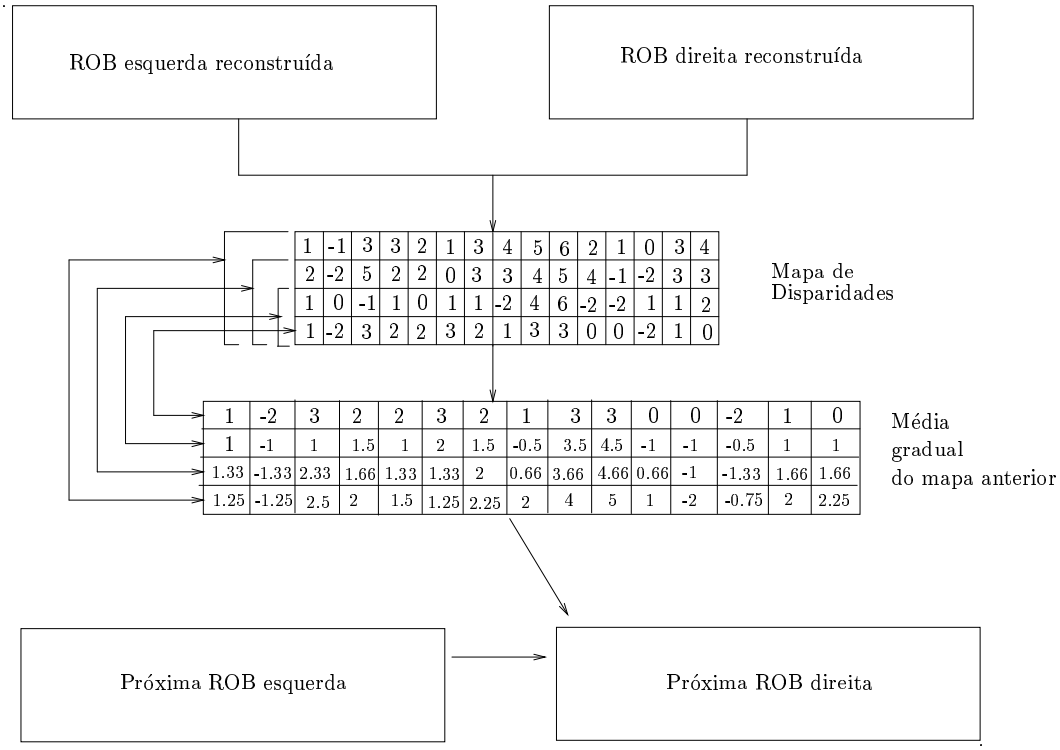


Figura 5.10: Média gradual, dando um peso maior aos valores mais próximos da vizinhança das ROBs e um peso menor aos valores mais distantes.

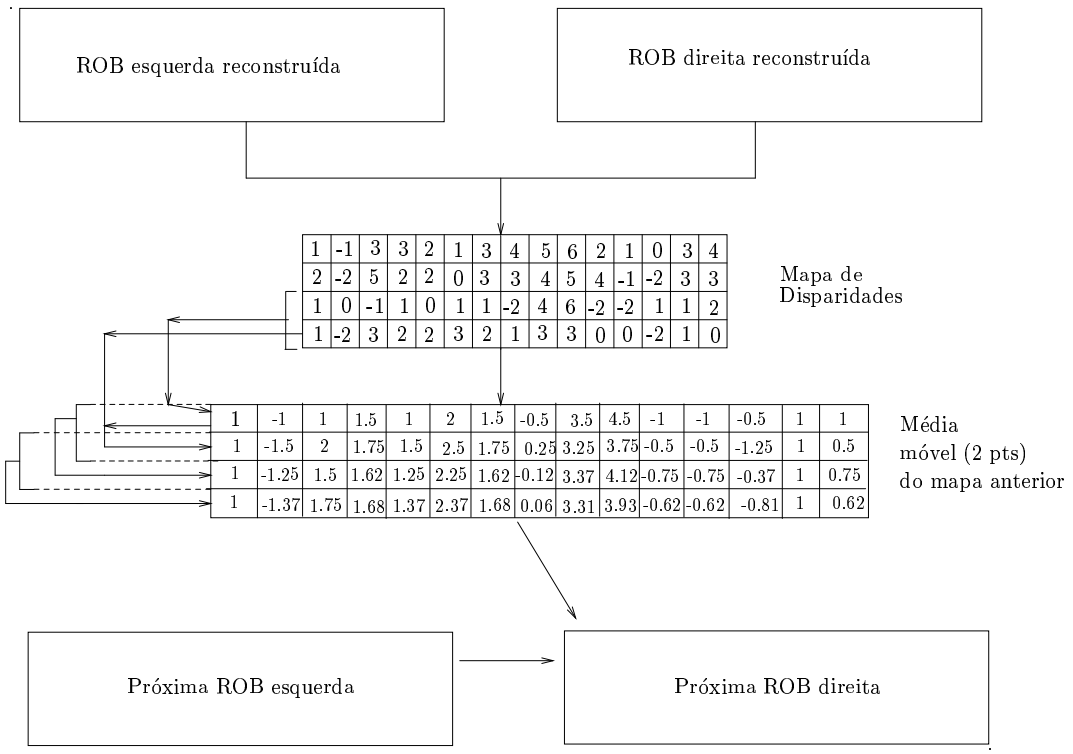


Figura 5.11: Média móvel (de dois pontos) das colunas das disparidades obtidas das ROBs anteriores.

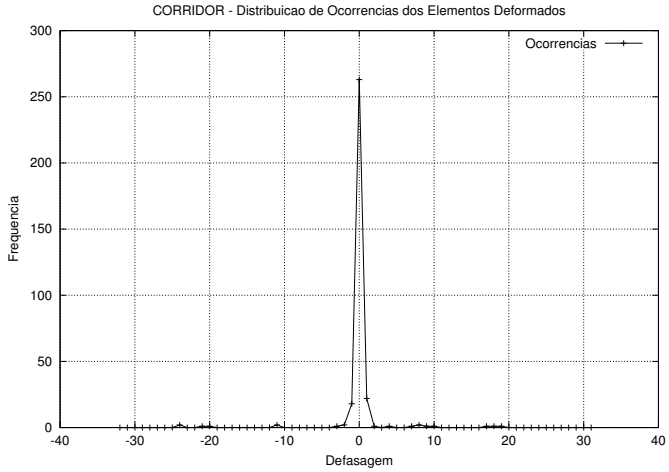
5.4 Versão 8 do MMP estéreo

Fizemos um estudo estatístico baseado nos resultados da versão 7, a respeito do número de ocorrências dos elementos deformados. Para um determinado bloco da imagem direita, verificamos que o elemento do dicionário que mais ocorria na codificação deste bloco era exatamente o elemento deformado que representava o bloco correspondente na ROB esquerda. Veja a Fig. 5.12. No eixo horizontal temos a defasagem do bloco da imagem direita que estamos querendo codificar, em relação ao bloco correspondente da imagem esquerda representado pelo elemento deformado. No eixo vertical temos o número de ocorrências dos elementos deformados. Verificamos um pico de ocorrências exatamente no elemento com defasagem 0, significando que o elemento deformado foi uma aproximação muito boa para o bloco correspondente. Algumas vezes este não foi o elemento escolhido. Atribuimos isto ao fato de utilizarmos o mapa de disparidades do par de ROBs anteriores para gerar os elementos deformados, ou seja, utilizamos uma estimativa "grosseira" para codificar as ROBs seguintes. No entanto, mesmo com esta estimativa considerada "grosseira", verificamos pelos gráficos de distribuição mostrados, que elas foram aproximações muito boas para codificar a imagem direita.

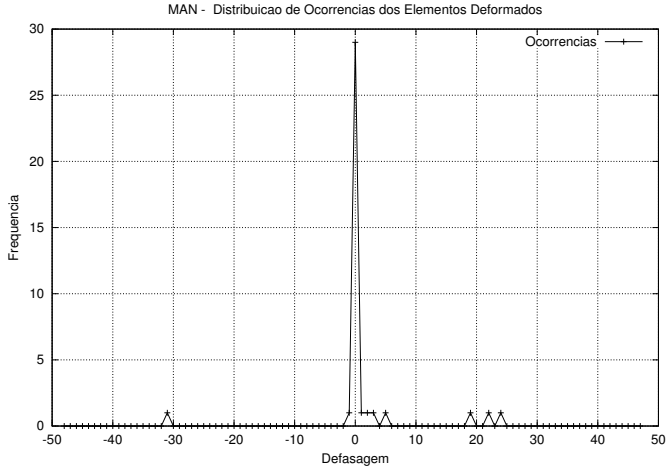
Utilizamos $\lambda = 32$ que é um valor intermediário da faixa que utilizamos desde o início desta tese (2 a 256), e o nível do dicionário para os gráficos mostrados é o nível 6 (nível mais alto para blocos 8x8).

Baseado nos resultados destes gráficos, fizemos a primeira melhoria na versão 7 do MMP estéreo (que introduz os elementos deformados), que foi reduzir o número de elementos deformados a um único elemento, exatamente o elemento deformado correspondente ao bloco que está sendo processado (veja a Fig.5.13). Antes utilizávamos todos os elementos deformados de uma ROB para codificar cada bloco da ROB direita.

Este procedimento equivale a um refinamento na estimativa do modelo, onde reduzimos o número de elementos e utilizamos um elemento mais significativo para representar o bloco da imagem direita. Podemos ver a melhoria do desempenho da codificação na versão 8, comparada com a versão 7, como mostram os gráficos da Fig. 5.14.



(a)



(b)

Figura 5.12: Distribuição de ocorrências dos elementos deformados para as imagens (a) CORRIDOR e (b) MAN.

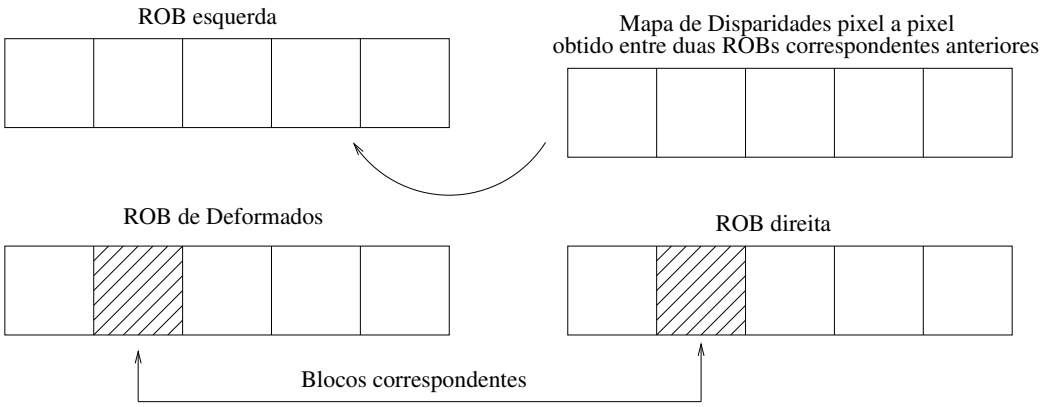
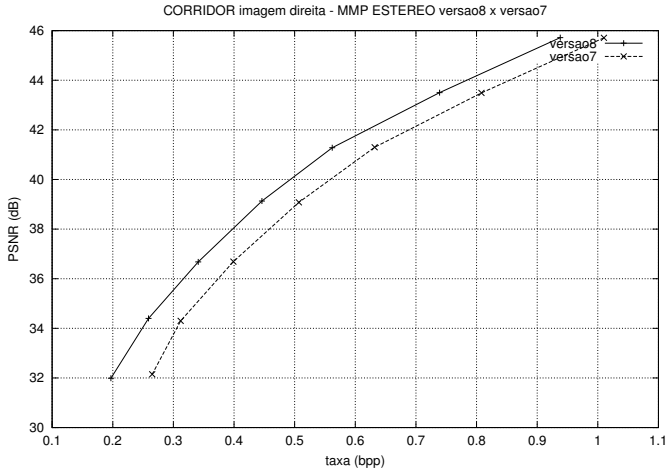
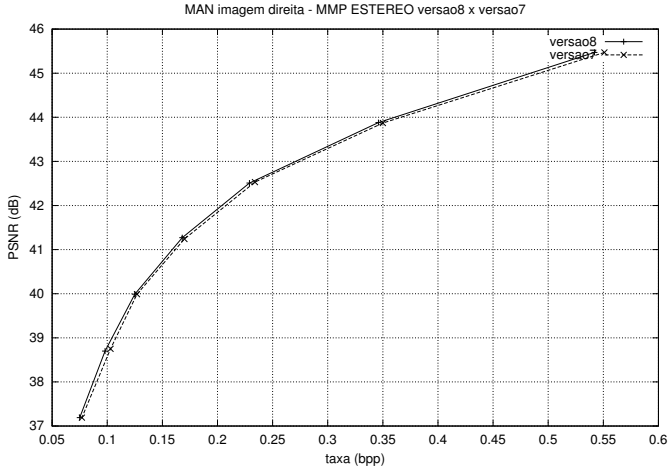


Figura 5.13: Número de elementos deformados do dicionário reduzido a um único elemento, aquele correspondente ao bloco da imagem direita.



(a)



(b)

Figura 5.14: Comparação entre as versões 8 e 7 do MMP estéreo. Resultados para a imagem direita dos pares estéreo (a) CORRIDOR e (b) MAN.

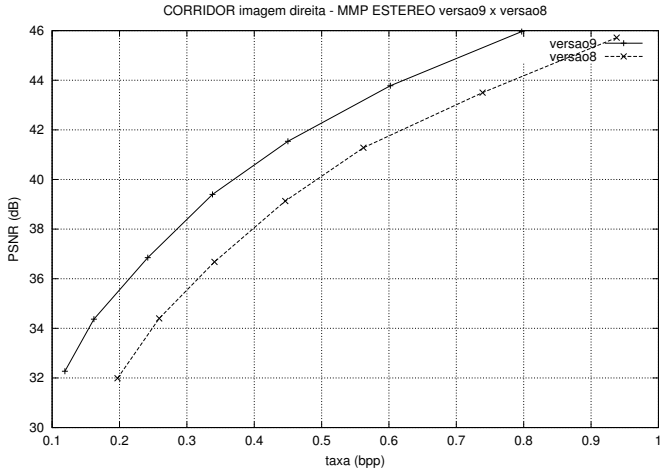
5.5 Versão 9 do MMP estéreo

Uma vez comprovada a eficácia de usar elementos deformados para codificar a imagem direita, trouxemos de volta os elementos deslocados da ROB esquerda correspondente (versão 6) para o mesmo dicionário que contém os deformados. Este dicionário será utilizado para codificar a imagem direita.

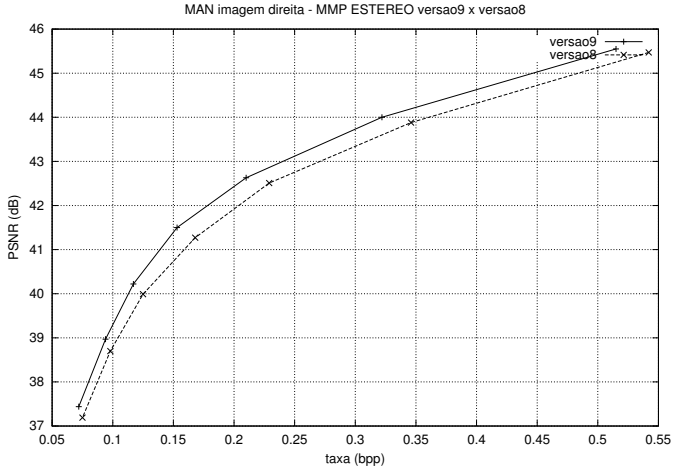
A Fig. 5.15 mostra os resultados comparativos entre a versão 9 do MMP estéreo, que traz de volta os elementos deslocados e a versão 8, que só utilizava os elementos deformados para codificar a imagem direita. Verificamos, pelos resultados, que o uso dos elementos deslocados é bastante relevante para a codificação da imagem direita.

Os elementos deslocados da versão 6, bem como os elementos deslocados desta versão 9, são limitados dentro de uma faixa dada pela janela de busca. A informação da janela de busca para as imagens utilizadas aqui foram obtidas estimando-se o mapa de disparidades através do casamento aproximado bloco a bloco (*block matching*) e tomando-se os valores limites (mínimo e máximo) desta estimativa. Estas informações são diretamente relacionadas às condições de obtenção das imagens, como distância entre as câmeras, distância focal, etc. A janela de busca é uma janela que limita a busca do bloco correspondente na imagem esquerda à uma faixa onde a probabilidade de se encontrar a correspondência seja maior. Como é uma faixa de valores, geramos elementos deslocados que não necessariamente irão corresponder ao bloco da imagem direita que estamos procurando. Fizemos um histograma mostrando a distribuição dos elementos deslocados que foram gerados limitados pela janela de busca. Este resultado é mostrado na Fig. 5.16 para a imagem MAN. Verificamos que esta distribuição é bastante espalhada. No eixo horizontal temos os índices dos elementos deslocados gerados. A janela de busca da imagem MAN é de 28 a 125. Como geramos deslocamentos de 1/2 pixel também, esta faixa passa a ser o dobro. No eixo vertical temos o número de ocorrências de cada elemento deslocado. Este número de ocorrências é bastante distribuído.

Na versão 8 fizemos um estudo estatística sobre as ocorrências dos elementos deformados, que foram obtidos baseados no mapa de disparidade calculado entre um par de ROBs anteriores, e os resultados mostraram que estes elementos são uma boa aproximação para os blocos da imagem direita. O número de ocorrências do



(a)



(b)

Figura 5.15: Comparação entre as versões 9 e 8 do MMP estéreo. Resultados para a imagem direita dos pares estéreo (a) CORRIDOR e (b) MAN.

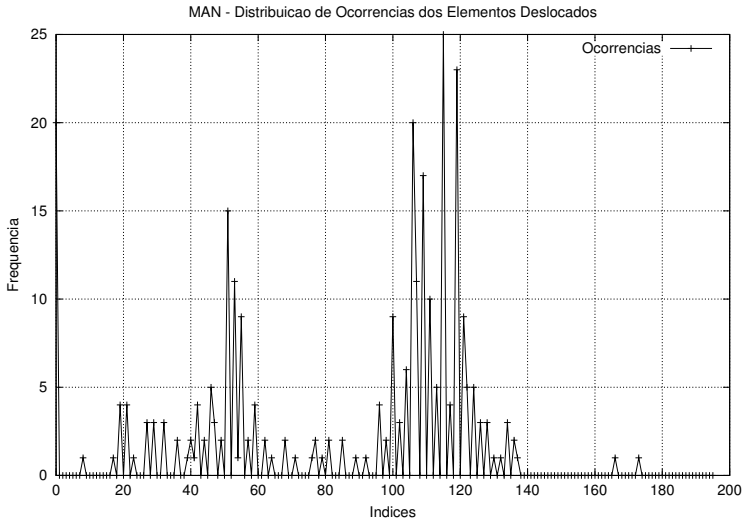


Figura 5.16: Distribuição das ocorrências dos elementos deslocados, limitados pela janela de busca (25 a 128) da imagem MAN.

elemento deformado correspondente ao bloco processado é muito grande, formando um pico nos histogramas mostrados para as imagens CORRIDOR, AQUA e MAN (Fig. 5.12). Ou seja, verificamos que utilizar as informações de disparidade de um par de ROBs correspondentes anteriores é uma boa aproximação para codificarmos a ROB direita atual.

Considerando isto, pensamos em utilizar esta informação de disparidade para nos informar qual o bloco da imagem esquerda é o correspondente ao bloco da imagem direita que estamos querendo codificar. Para fazer isto, calcularmos uma disparidade média do bloco, já que a disparidade calculada era por pixel. Uma vez encontrado este bloco correspondente, geramos os elementos deslocados numa faixa do mesmo tamanho da janela de busca original, porém agora em torno de um elemento central que é exatamente o bloco encontrado como sendo o correspondente, de acordo com a disparidade média (veja a Fig.5.17).

Fizemos um estudo estatístico de como ficaria a distribuição das ocorrências se utilizássemos esta nova maneira de gerar os elementos deslocados. O resultado é mostrado no gráfico da Fig. 5.18 para a imagem MAN. Verificamos que a distribuição agora ficou bem mais concentrada em relação ao gráfico anterior (Fig. 5.16) que gerava os elementos deslocados sem contar com a informação de disparidade.

No eixo horizontal temos a defasagem entre o bloco que está sendo codifi-

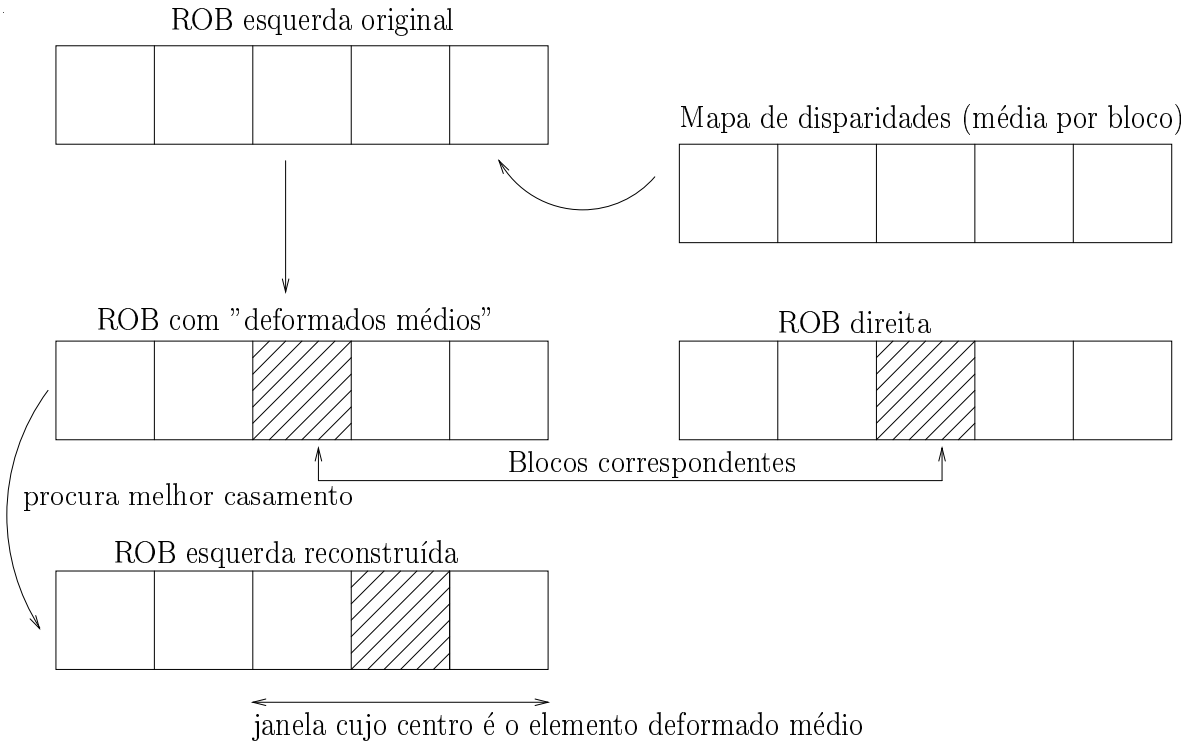


Figura 5.17: Uso da informação de disparidades para encontrar janela de deslocados.

cado na imagem direita, e o bloco encontrado como sendo o bloco correspondente na imagem esquerda, baseado na informação de disparidade obtida entre o par de ROBs correspondentes anteriores. Verificamos que o pico de ocorrências não está na defasagem 0, como poderíamos esperar. Isto pode ser devido a dois problemas de aproximação:

1. O mapa de disparidades utilizado como base foi o mapa encontrado entre um par de ROBs anteriores, e não o mapa do par de ROBs atual.

2. O elemento deslocado encontrado foi baseado numa média do bloco de disparidades por pixel correspondente ao bloco da imagem direita que está sendo processao. Se utilizássemos os valores de disparidade por pixel encontraríamos um bloco deformado, e não um possível bloco deslocado.

No eixo vertical temos o número de ocorrências destes elementos. Verificamos um pico de ocorrências próximo ao elemento deslocado que corresponde ao centro da janela. Este elemento foi encontrado utilizando a informação de disparidade obtida entre o par de ROBs correspondentes anteriores à ROB que está sendo processada.

A distribuição é bastante concentrada. Além de ficar mais concentrada, o número de ocorrências dos elementos deslocados aumentou um pouco (de 25, no gráfico anterior, para aproximadamente 35 neste). Isto nos indica que podemos melhorar o resultado da compressão se utilizarmos esta forma de gerar os elementos deslocados.

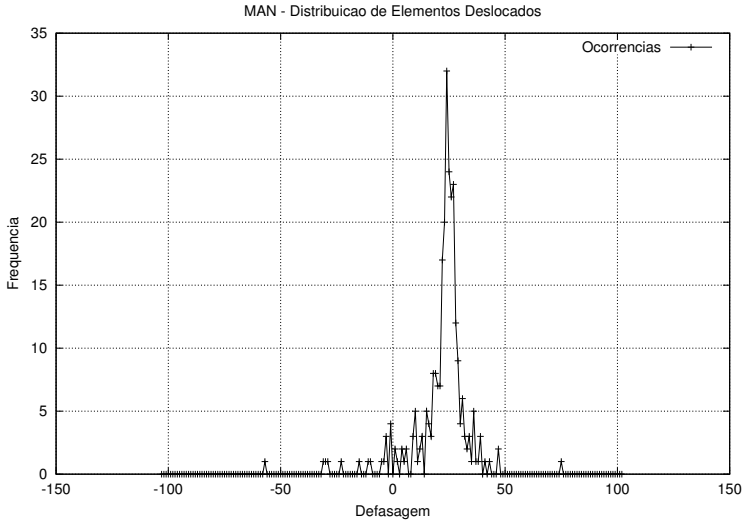


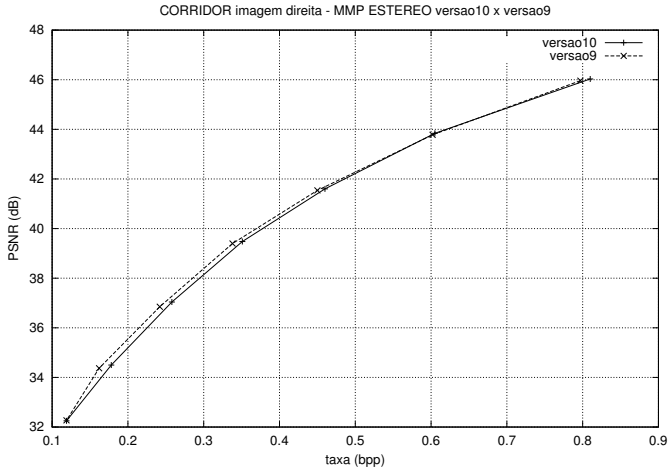
Figura 5.18: Distribuição das ocorrências dos elementos deslocados, que foram gerados baseados na informação de disparidade. Resultados para a imagem MAN.

Os resultados deste estudo estatístico nos levaram à próxima versão do MMP estéreo. Nela incorporamos esta nova maneira de gerar os elementos deslocados, baseados na informação de disparidade obtida entre um par de ROBs anteriores.

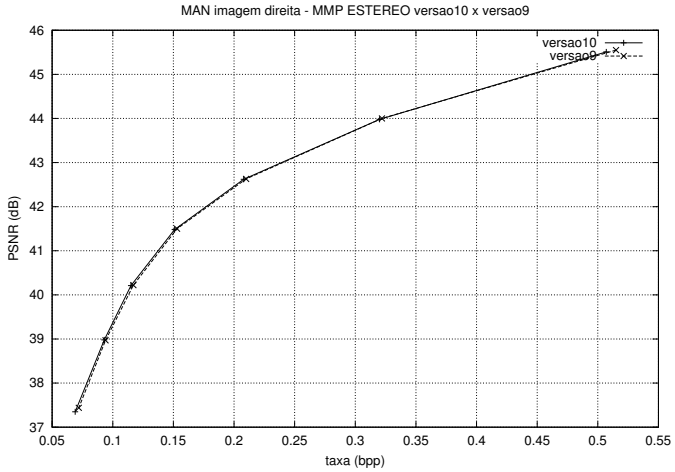
5.6 Versão 10 do MMP estéreo

Quando nós restringimos os elementos deformados a uma janela (no caso, com um único elemento) na versão 8, verificamos que o resultado de compactação melhorou para a imagem direita, porque melhoramos a estimativa do modelo que representaria os blocos da imagem direita. Fizemos o mesmo para os elementos deslocados. Conhecendo a disparidade média de um dado bloco da imagem direita, podemos pegar uma janela de deslocados em torno do elemento que corresponderia a este bloco na outra imagem. Com isso tende-se a escolher para esta janela os elementos deslocados que possuem maior probabilidade de ocorrer para o bloco que está sendo processado na imagem direita.

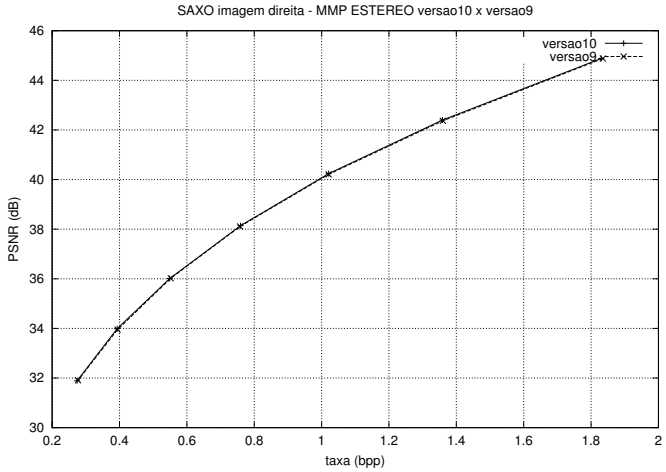
Os resultados comparativos entre a versão 10 do MMP estéreo, que incorpora os elementos deslocados gerados com base na informação de disparidade de ROBs anteriores, e a versão 9 do MMP estéreo que utiliza os elementos deslocados limitados pela janela de busca podem ser vistos nos gráficos da Fig. 5.19. Apesar do estudo realizado mostrar uma concentração das ocorrências quando utilizamos uma janela de deslocados cujo elemento central é dado pelas informações da disparidade, os resultados de taxa×PSNR demonstraram o contrário para a maioria das imagens. Só conseguimos melhores resultados para a imagem SAXO. Este é um problema que precisa ser mais investigado.



(a)



(b)



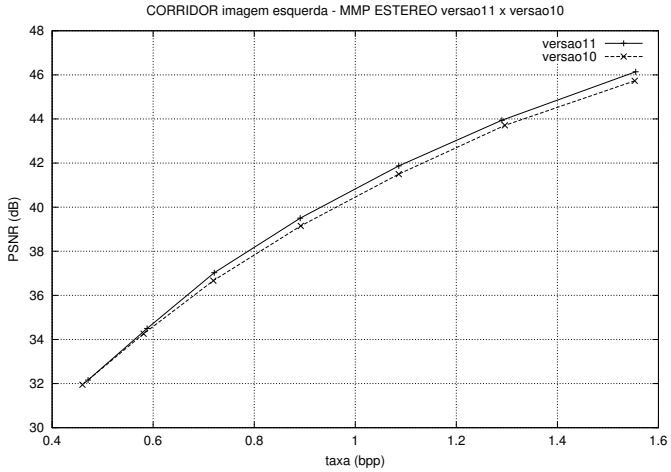
(c)

Figura 5.19: Comparação entre as versões 10 e 9 do MMP estéreo. Resultados para a imagem direita dos pares estéreo (a) CORRIDOR, (b) MAN e (c) SAXO.

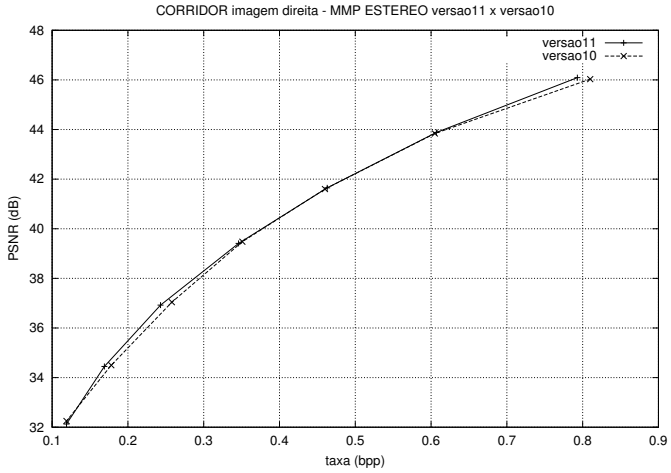
5.7 Versão 11 do MMP estéreo

Até agora só mostramos as melhorias ocorridas na codificação da imagem direita. Uma vez que tínhamos removido os elementos deslocados acumulados do dicionário comum, os resultados da codificação da imagem esquerda haviam piorado comparado com os últimos resultados do capítulo 4 (versão 6 do MMP estéreo).

Agora trazemos de volta estes elementos, melhorando a codificação da imagem esquerda e mantendo, no entanto, todas as melhorias obtidas na codificação da imagem direita (às vezes até melhorando, uma vez que melhorou a estimativa da imagem esquerda). Os resultados são mostrados na Fig. 5.20 para a imagem estéreo CORRIDOR, na Fig. 5.21 para a imagem MAN e na Fig. 5.22 para a imagem SAXO.

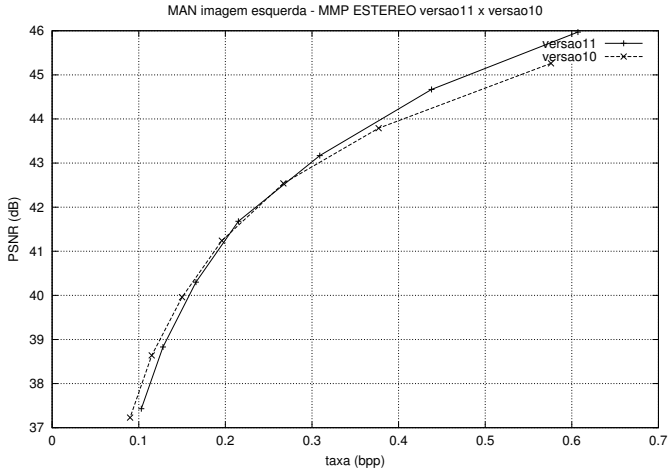


(a)

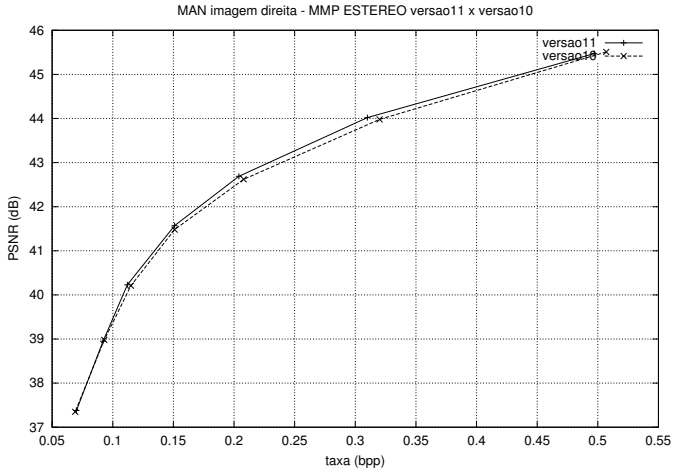


(b)

Figura 5.20: Comparação entre as versões 11 e 10 do MMP estéreo. (a) imagem esquerda, e (b) imagem direita do par estéreo CORRIDOR

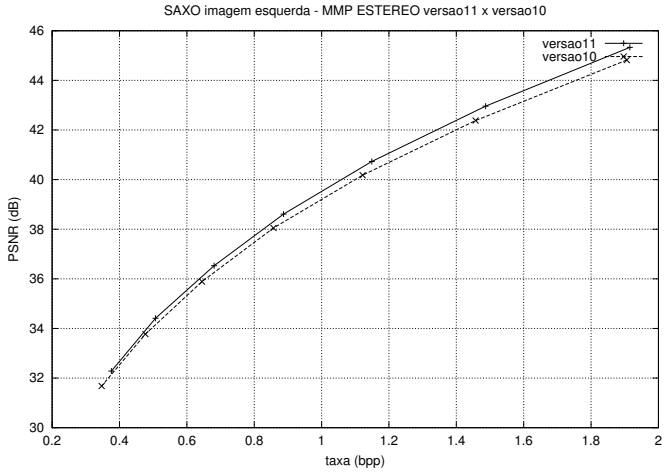


(a)

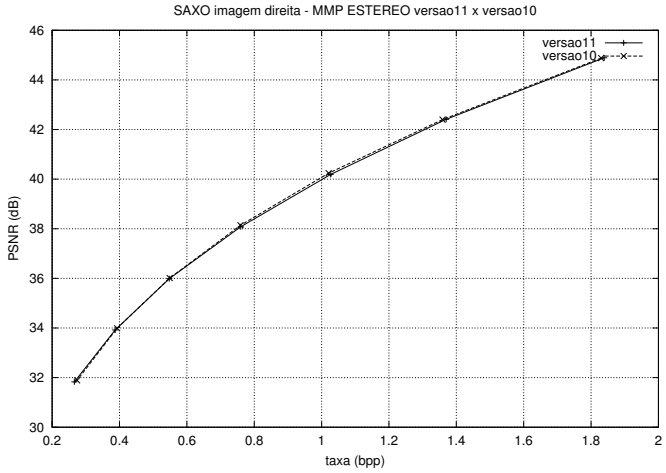


(b)

Figura 5.21: Comparação entre as versões 11 e 10 do MMP estéreo. (a) imagem esquerda, e (b) imagem direita do par estéreo MAN.



(a)



(b)

Figura 5.22: Comparação entre as versões 11 e 10 do MMP estéreo. (a) imagem esquerda, e (b) imagem direita do par estéreo SAXO.

5.8 Versão 12 do MMP estéreo

A partir dos resultados da versão 2 do MMP estéreo, percebemos que aumentar o dicionário utilizado para codificar a imagem esquerda parecia sempre melhorar os resultados de compressão desta imagem. Tendo em mente esta premissa, decidimos introduzir neste dicionário elementos que chamamos de "Rotacionados". Estes elementos são obtidos de rotações dos blocos que melhor representaram os blocos da imagem esquerda no dicionário comum.

Estas rotações foram de 90, 180, 270 (Veja Fig. 5.23), simetrias horizontal, vertical, em torno da diagonal principal, e em torno da diagonal secundária (Veja Fig. 5.24).

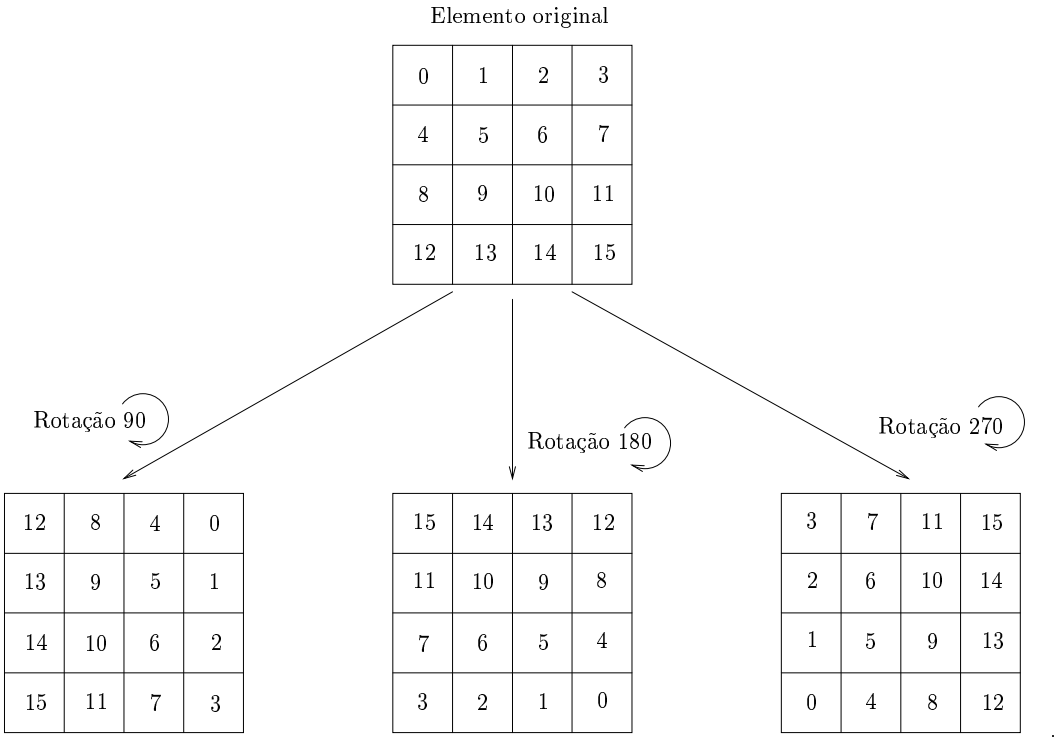


Figura 5.23: Como são formados os elementos rotacionados.

Fizemos uma análise do percentual de ocorrências destes elementos, em relação ao número de ocorrências total de todos os tipos de elementos do dicionário utilizado na codificação da imagem esquerda das imagens CORRIDOR e AQUA. Os resultados são mostrados na Tabela 5.1.

Apesar de notarmos uma predominância dos elementos iniciais e concatenados nesta tabela, percebemos também que o uso de elementos rotacionados e

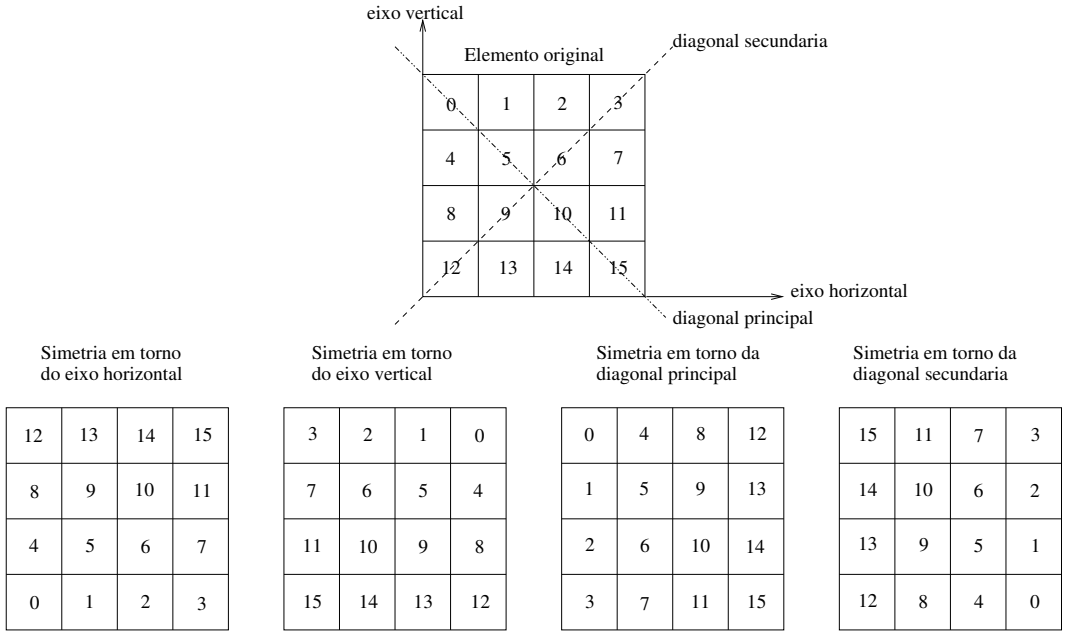
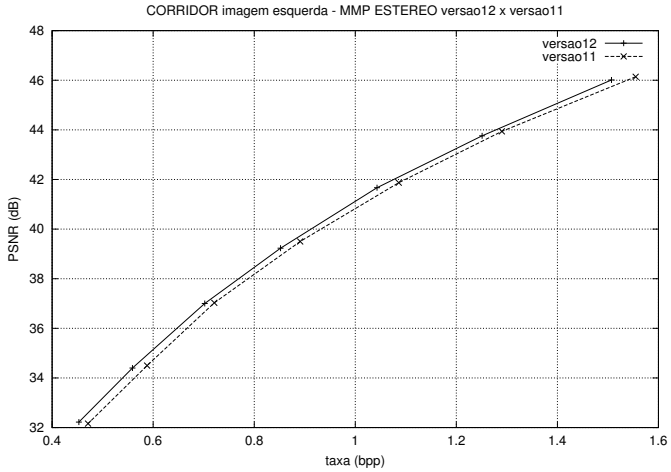


Figura 5.24: Como são formados os elementos simétricos.

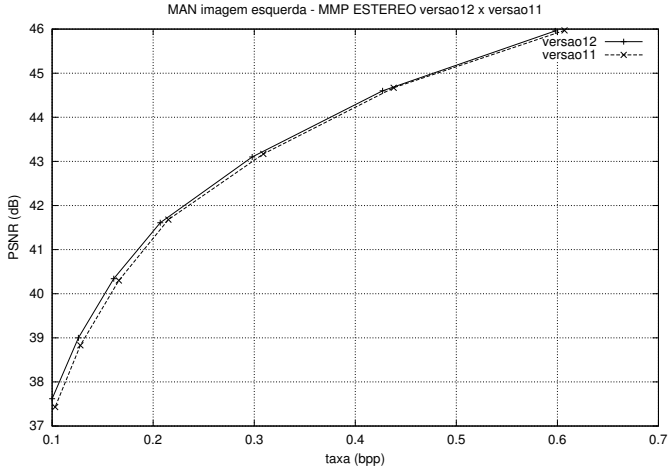
simétricos foi, em alguns casos, muito importante, chegando a percentuais maiores do que elementos expandidos e contraídos.

A utilização dos elementos rotacionados e simétricos para codificar a imagem direita não deu bons resultados.

Os resultados comparativos da versão 12, que utiliza elementos rotacionados e simétricos para codificar a imagem esquerda, e da versão 11, que não utiliza estes elementos, podem ser verificados nos gráficos da Fig. 5.25.



(a)



(b)

Figura 5.25: Comparação entre as versões 12 e 11 do MMP estéreo. Resultados para a imagem direita dos pares estéreo (a) CORRIDOR e (b) MAN.

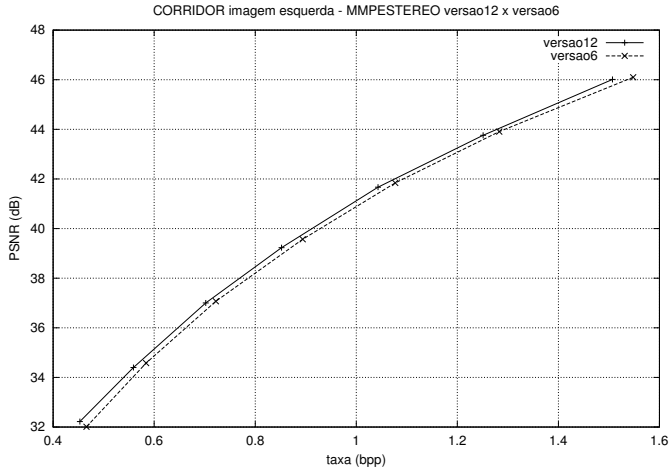
Tabela 5.1: Valores percentuais de ocorrência dos elementos do dicionário utilizado pelo MMP estéreo (versão 12) para codificar a imagem esquerda (a) CORRIDOR e (b) AQUA.

Tipo do elemento	Nível 0	Nível 1	Nível 2	Nível 3	Nível 4	Nível 5	Nível 6
Concatenado	0	20,61	7,78	1,37	1,80	0,53	0,55
Contraído	0	5,50	2,21	0,87	0,58	0,14	0
Deslocado	0	0	0	0,67	0,26	0,12	0,18
Expandido	0	0	1,87	0,76	0,45	0,66	0,33
Inicial	5,59	8,24	31,03	11,32	10,29	10,74	23,38
Rot.180	0	0	2,68	0,39	0,10	0,11	0,06
Rot.270	0	0	0,53	0	0,02	0	0
Rot.90	0	0	0,82	0	0,05	0	0,02
SimetriaDP	0	0	0,50	0	0,08	0	0,03
SimetriaDS	0	0	6,03	0	0,03	0	0,01
SimetriaH	0	0	4,21	0,65	0,10	0,29	0,10
SimetriaV	0	0	0,79	0,48	0,12	0,16	0,26

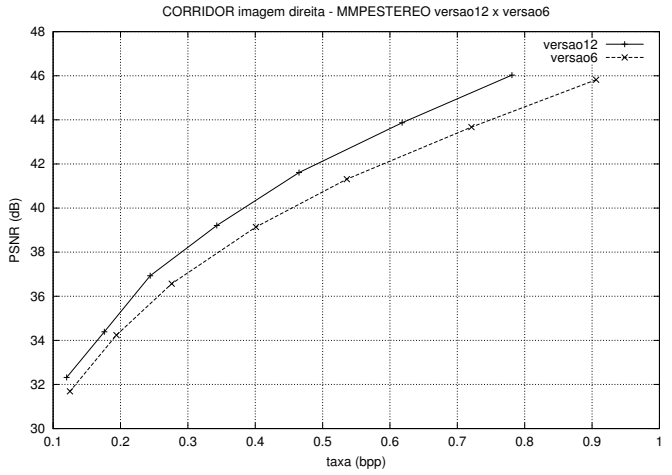
(a)

Tipo do elemento	Nível 0	Nível 1	Nível 2	Nível 3	Nível 4	Nível 5	Nível 6
Concatenado	0	37,50	13,51	1,36	0,63	0,20	0,09
Contraído	0	9,19	7,76	2,57	1,70	0,44	0
Deslocado	0	0	0	0,53	0,94	0,34	0,19
Expandido	0	0	12,50	2,60	0,14	0,14	0,05
Inicial	46,30	27,47	50,46	12,65	12,50	13,12	43,36
Rot.180	0	0	9,66	1,29	0,37	0,21	0,07
Rot.270	0	0	7,36	0	0,39	0	0,04
Rot.90	0	0	8,43	0	0,34	0	0,1
SimetriaDP	0	0	12,31	0	0,41	0	0,11
SimetriaDS	0	0	9,58	0	0,42	0	0,07
SimetriaH	0	0	13,19	1,35	0,39	0,24	0,08
SimetriaV	0	0	9,59	1,39	0,53	0,26	0,04

(b)



(a)



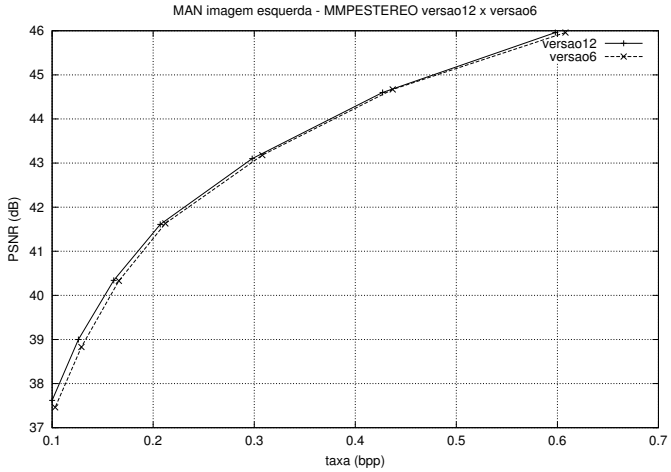
(b)

Figura 5.26: Comparação entre as versões 12 e 6 do MMP estéreo, para o par estéreo CORRIDOR. (a) imagem esquerda, (b) imagem direita.

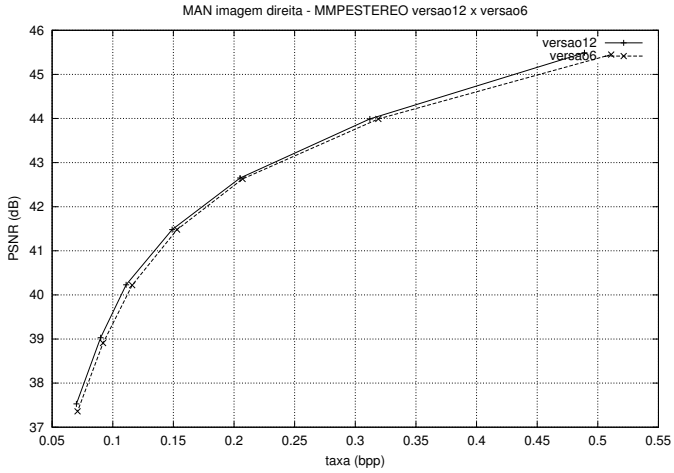
5.9 Comparação entre as versões 6 e 12 do MMP estéreo

Para verificarmos isoladamente as melhorias implementadas neste capítulo, comparamos a versão 12, que corresponde à última versão desta segunda fase de desenvolvimento da tese, com a última versão do capítulo anterior (versão 6 - Cap. 4).

Na Fig. 5.26 temos o resultado comparativo entres as versões 12 e 1 do MMP estéreo para a imagem CORRIDOR, e na Fig. 5.27 para a imagem MAN.



(a)



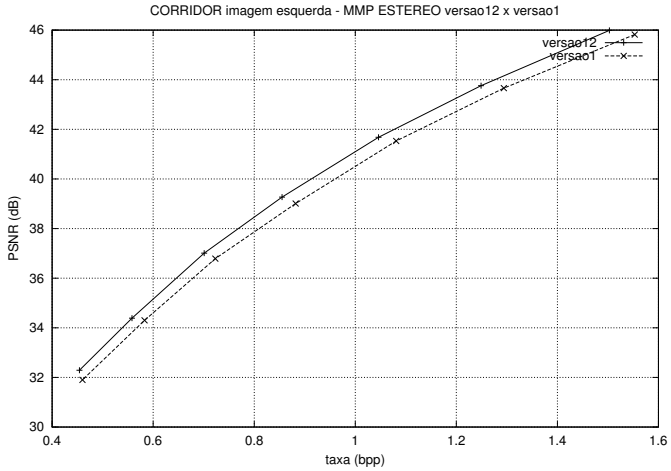
(b)

Figura 5.27: Comparação entre as versões 12 e 6 do MMP estéreo, para o par estéreo MAN. (a) imagem esquerda, (b) imagem direita.

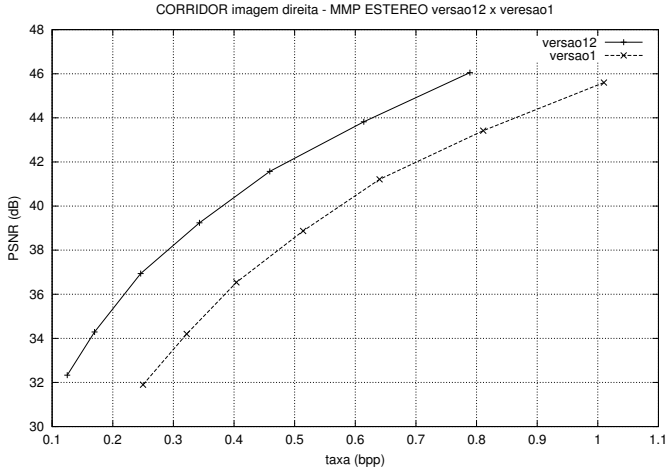
5.10 Comparação entre as versões 1 e 12 do MMP estéreo

Para encerrar este capítulo, que corresponde à segunda fase de desenvolvimento desta tese, mostramos nesta seção uma comparação entre os resultados da versão 1 do MMP estéreo, descrita no Cap. 4, e da versão 12 do MMP estéreo (descrita na seção anterior deste capítulo). Podemos verificar pelos gráficos que conseguimos explorar a similaridade estéreo introduzindo elementos deslocados e deformados no dicionário. A versão 12 apresenta maiores valores de PSNR para a mesma taxa comparado com a versão 1 para todas as imagens mostradas.

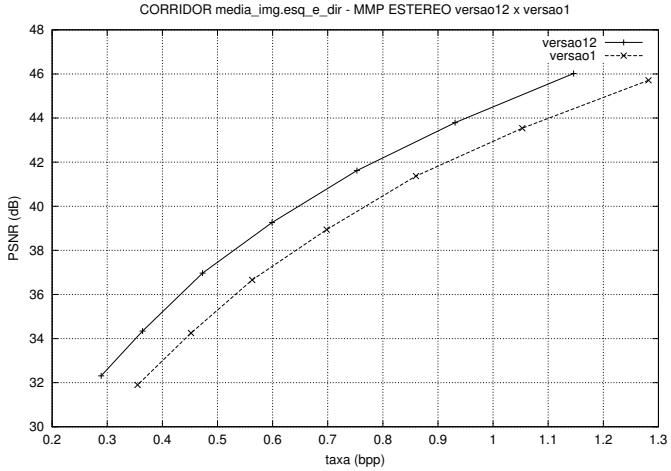
Na Fig. 5.28 temos o resultado comparativo entres as versões 12 e 1 do MMP estéreo para a imagem CORRIDOR, na Fig. 5.29 para a imagem AQUA, na Fig. 5.30 para a imagem MAN, na Fig. 5.31 para a imagem SAXO, e na Fig. 5.32 para a imagem TRAIN.



(a)

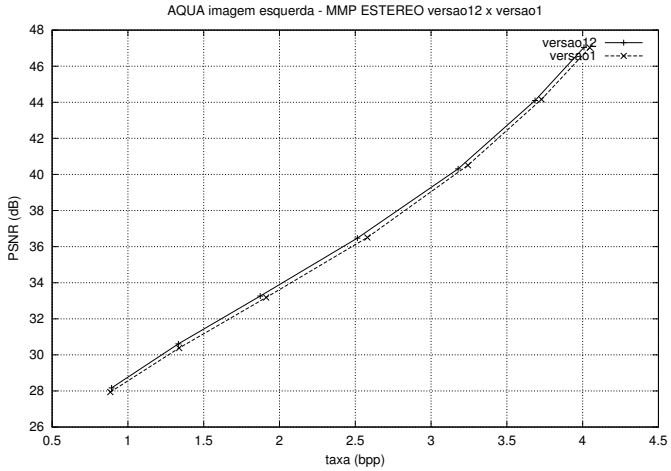


(b)

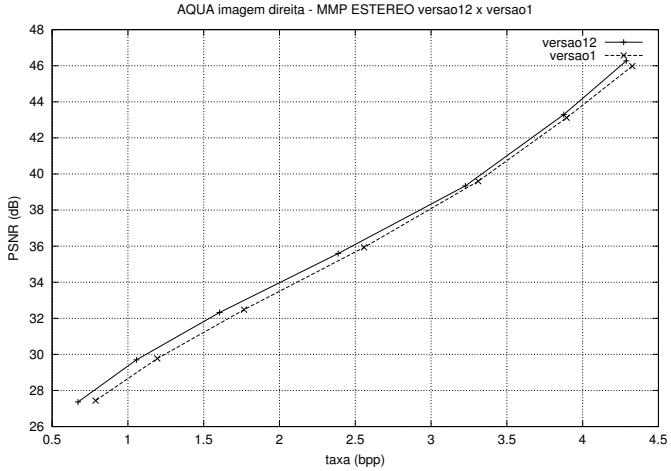


(c)

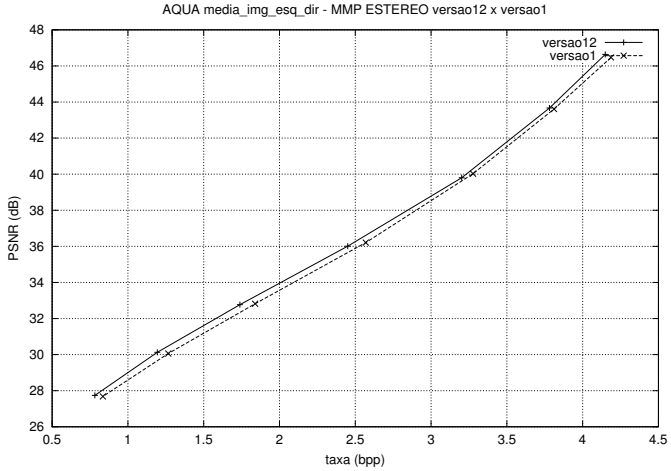
Figura 5.28: Comparação entre as versões 12 e 1 do MMP estéreo, para o par estéreo CORRIDOR. (a) imagem esquerda, (b) imagem direita, (c) média entre as imagens esquerda e direita.



(a)

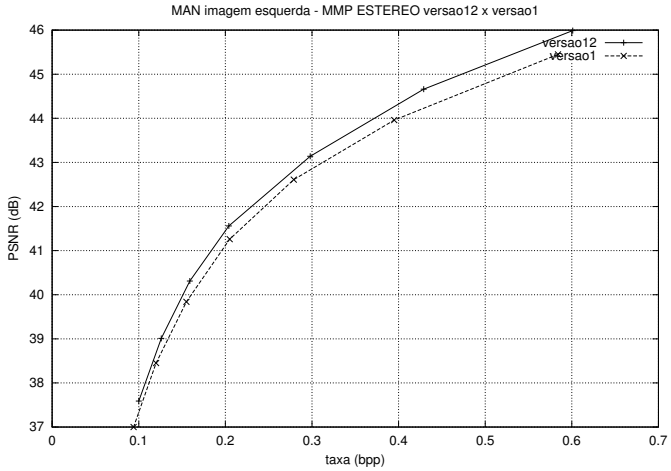


(b)

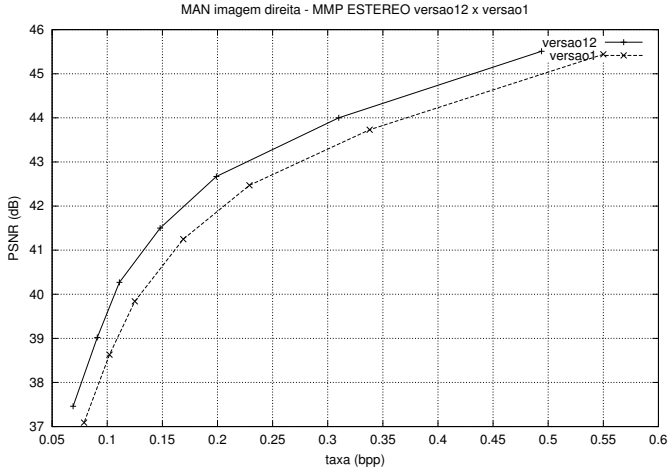


(c)

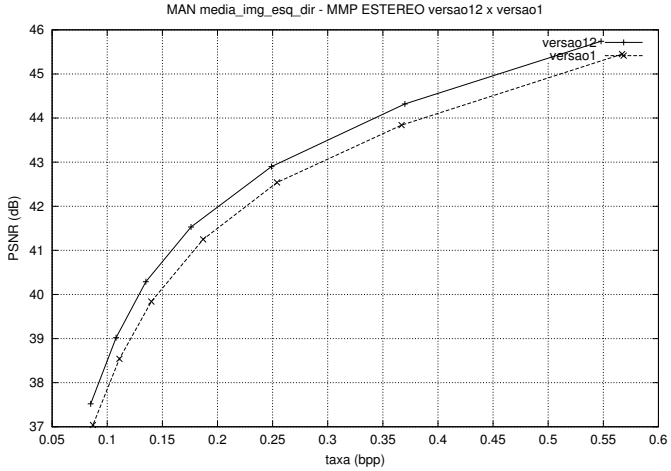
Figura 5.29: Comparação entre as versões 12 e 1 do MMP estéreo, para o par estéreo AQUA. (a) imagem esquerda, (b) imagem direita, (c) média entre as imagens esquerda e direita.



(a)

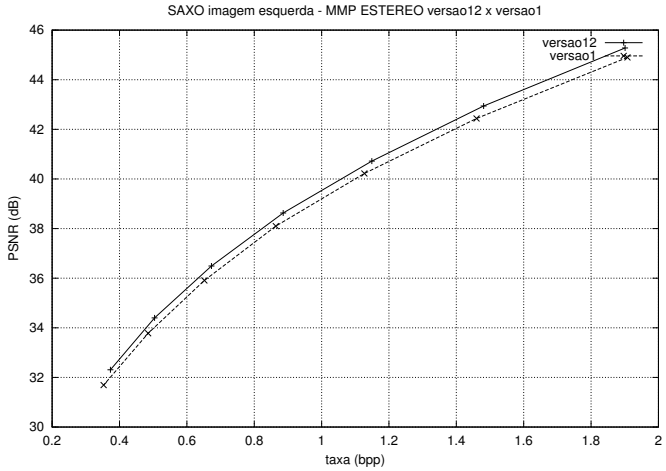


(b)

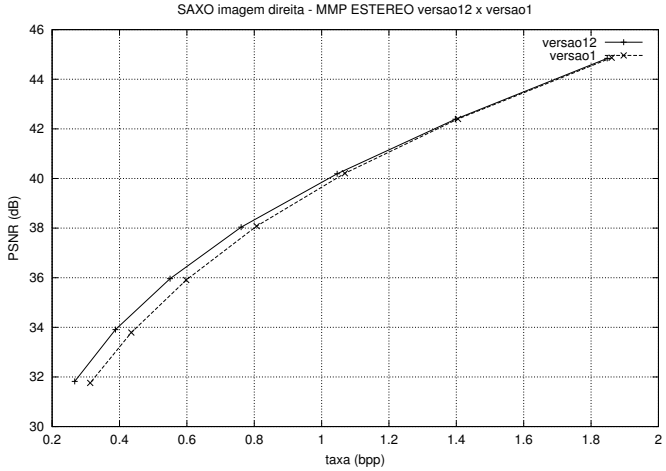


(c)

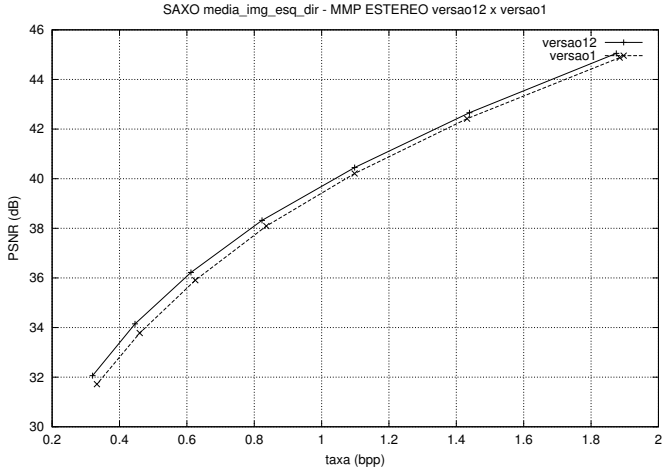
Figura 5.30: Comparação entre as versões 12 e 1 do MMP estéreo, para o par estéreo MAN. (a) imagem esquerda, (b) imagem direita, (c) média entre as imagens esquerda e direita.



(a)

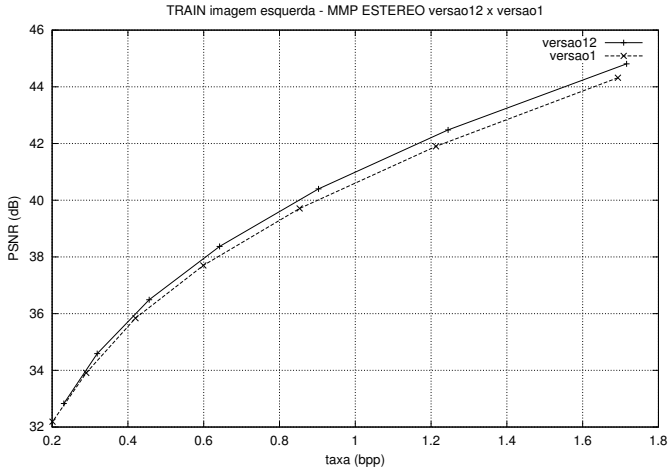


(b)

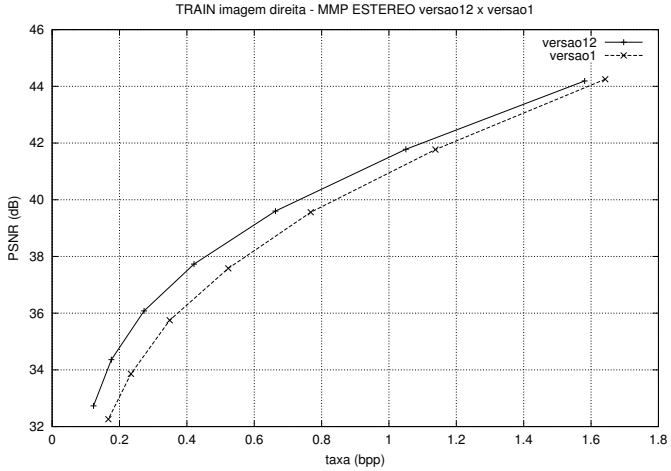


(c)

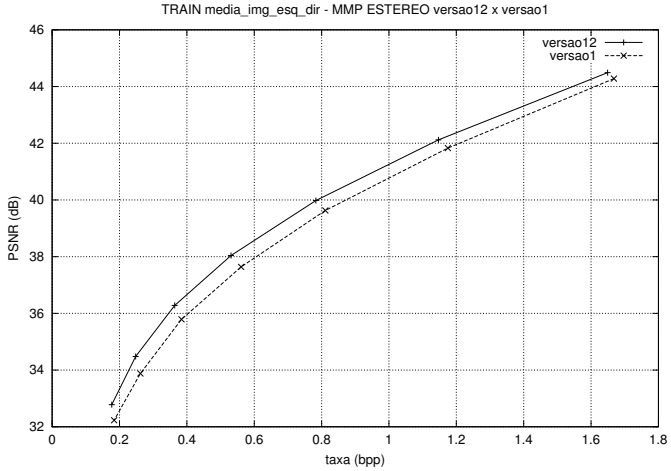
Figura 5.31: Comparação entre as versões 12 e 1 do MMP estéreo, para o par estéreo SAXO. (a) imagem esquerda, (b) imagem direita, (c) média entre as imagens esquerda e direita.



(a)



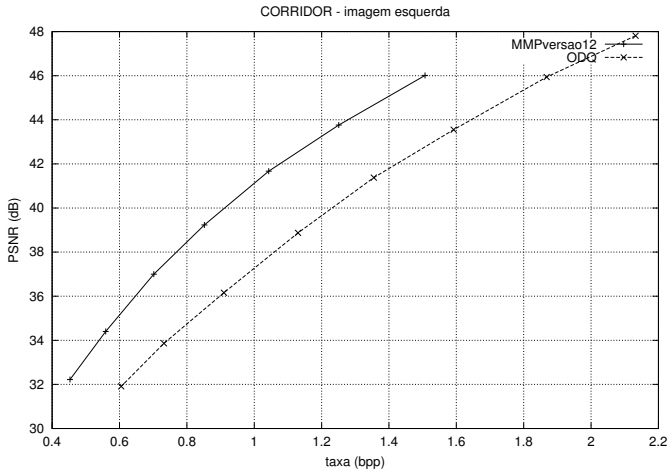
(b)



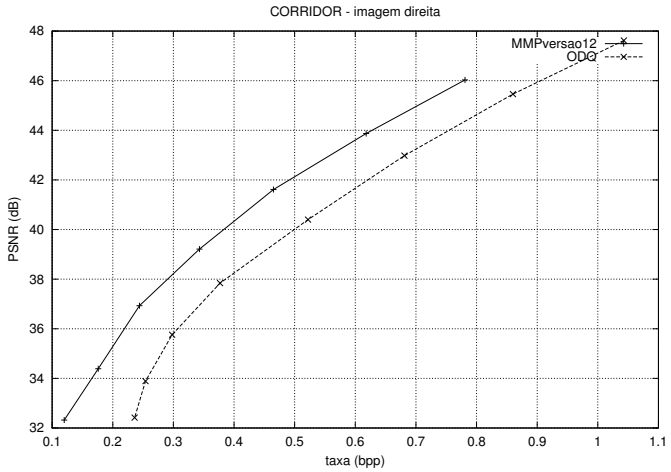
(c)

Figura 5.32: Comparação entre as versões 12 e 1 do MMP estéreo, para o par estéreo TRAIN. (a) imagem esquerda, (b) imagem direita, (c) média entre as imagens esquerda e direita.

5.11 Comparação entre MMP versão 12 e ODQ

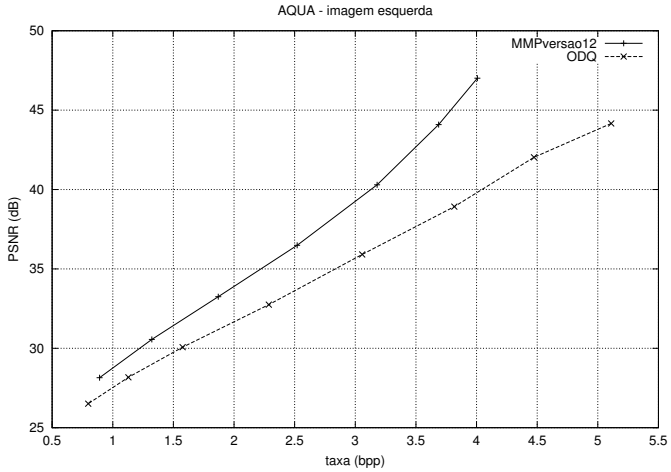


(a)

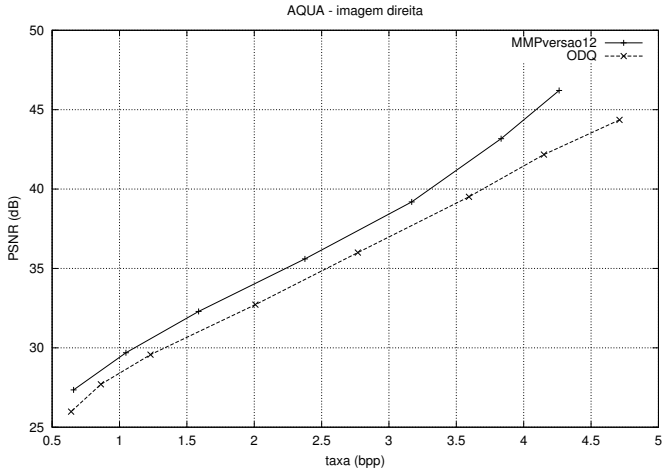


(b)

Figura 5.33: Comparação entre MMP estéreo versão 12 e ODQ, para o par estéreo CORRIDOR. (a) imagem esquerda, (b) imagem direita.



(a)

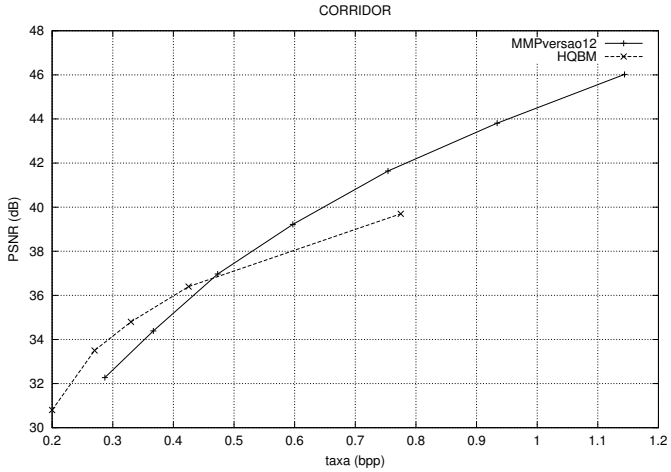


(b)

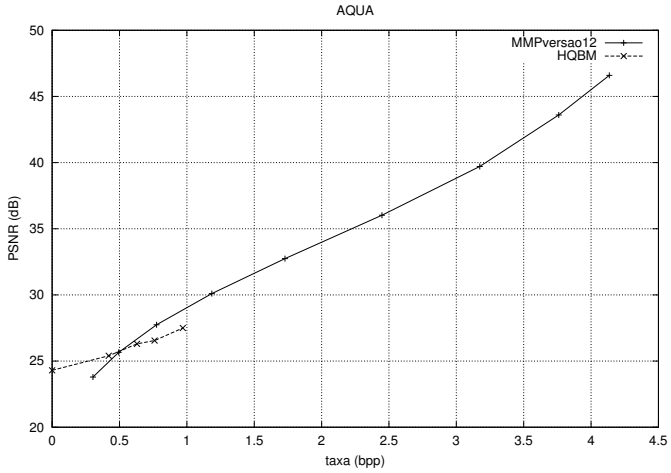
Figura 5.34: Comparação entre MMP estéreo versão 12 e ODQ, para o par estéreo AQUA. (a) imagem esquerda, (b) imagem direita.

5.12 Comparação entre MMP versão 12 e ODQ melhorado (HQBM)

Em [39] Woontack Woo e Ortega apresentam uma nova versão do ODQ utilizando modelo de Markov e casamento de blocos com divisão hierárquica (*hierarquical quadtree*). Chamamos esta nova versão do ODQ de HQBM (*Hierarquical quadtree block matching*). A comparação com esta nova versão é mostrada na Fig. 5.35. Só dispunhamhos dos valores de PSNR para estas duas imagens, e em baixas taxas, conforme os resultados apresentados em [39], por isto não pudemos comparar os resultados para taxas maiores e para outras imagens. Os gráficos indicam que o MMP estéreo ganha do HQBM para valores de taxa acima de 0,5 bpp. Taxas abaixo deste valor não são interessantes para a maioria das aplicações porque implicam em uma imagem reconstruída com qualidade ruim (por exemplo, abaixo de 26 dB para a imagem AQUA, que é uma imagem real).



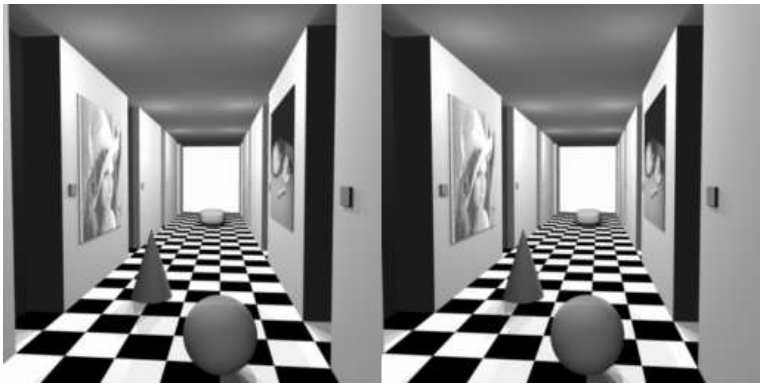
(a)



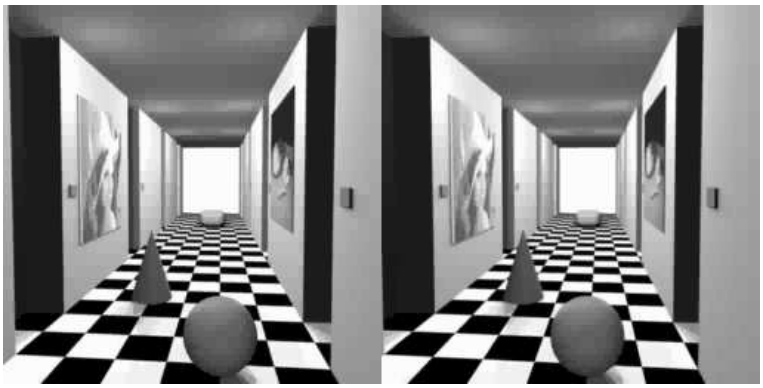
(b)

Figura 5.35: Comparação entre MMP estéreo versão 12 e HQBM, para o par estéreo (a) CORRIDOR e (b) AQUA.

5.13 Imagens reconstruídas



(a)



(b)

Figura 5.36: (a) CORRIDOR original (256x256), 131110 bytes (inclui cabeçalho) (b) CORRIDOR reconstruído ($\lambda = 32$), img esq.: taxa=0,701 PSNR=37,01. img dir.: taxa=0,246 PSNR=36,94. Número bytes imagem compactada: 7765.

(a)

(b)

(c)

Figura 5.37: (a) MAN original (384x384), 294950 bytes (inclui cabeçalho) (b) MAN reconstruído ($\lambda = 32$), img esq.: taxa=0,159 PSNR=40,31. img dir.: taxa=0,111 PSNR=40,27. Número bytes imagem compactada: 4991. (c) MAN reconstruído ($\lambda = 8$), img esq.: taxa=0,298 PSNR= 43,14. img dir.: taxa=0,199 PSNR=42,67. Número bytes imagem compactada: 9182.

5.14 MMP estéreo versão 12: Tabelas com valores de taxa e PSNR

Tabela 5.2: Valores de taxa e PSNR resultantes do MMP estéreo versão 12 aplicado às imagens estéreo (a) CORRIDOR e (b) AQUA.

Imagem esquerda

Lambda	Taxa	PSNR
2	1,507	46,01
4	1,251	43,76
8	1,043	41,67
16	0,852	39,23
32	0,702	37,00
64	0,559	34,40
128	0,453	32,22

Imagem direita

Lambda	Taxa	PSNR
2	0,781	46,03
4	0,618	43,87
8	0,465	41,61
16	0,343	39,21
32	0,244	36,93
64	0,176	34,39
128	0,120	32,32

(a)

Imagem esquerda

Lambda	Taxa	PSNR
2	4,007	47,01
4	3,687	44,09
8	3,180	40,30
16	2,521	36,49
32	1,870	33,25
64	1,322	30,56
128	0,891	28,15

Imagem direita

Lambda	Taxa	PSNR
2	4,263	46,21
4	3,832	43,17
8	3,169	39,19
16	2,376	35,60
32	1,587	32,29
64	1,048	29,69
128	0,659	27,35

(b)

5.15 Versão 12 do MMP estéreo: itens implementados

A versão 12 contém os seguintes itens implementados ao longo do desenvolvimento da tese:

1. A codificação é feita da esquerda para a direita, ROB a ROB. Codifica-se uma ROB da imagem esquerda e depois a ROB correspondente da imagem direita.
2. Utiliza-se um dicionário contendo elementos comuns e elementos deslocados acumulados de ROBs anteriores para codificar uma ROB da imagem esquerda. Para codificar a ROB correspondente da imagem direita, utiliza-se o dicionário contendo elementos comuns ou um outro dicionário contendo somente os elementos deslocados gerados pela ROB esquerda correspondente.
3. Os elementos deslocados são limitados por uma janela de busca.
4. São gerados deslocamentos de 1/2 pixel.
5. Utiliza-se elementos deformados para codificar a imagem direita.
6. Utiliza-se a informação de disparidade de ROBs anteriores para escolher uma janela mais adequada para os elementos deslocados.
7. Utiliza-se elementos rotacionados e simétricos para codificar a imagem esquerda.

5.16 Conclusão

Neste capítulo introduzimos o cálculo do mapa de disparidades no processo de codificação das imagens estéreo utilizando o MMP estéreo. Desde o início tínhamos em mente não transmitir esta informação completa ao decodificador. A princípio chamamos a atenção para a importância de calcular um mapa de disparidades preciso para uma boa qualidade da imagem reconstruída final, porém sabíamos da conseqüência de usar um mapa preciso, que é aumentar consideravelmente a taxa. Além disso, calcular um mapa de disparidades preciso implica em um processo bastante complexo. Por isto optamos por estimar um mapa de disparidades denso pixel a pixel.

A princípio pensamos em calcular e transmitir apenas o mapa de disparidades entre o primeiro par de ROBs correspondentes. O decodificador utilizaria este mapa como um estimativa para codificar o par de ROBs seguinte. Porém, mesmo assim isto ia de encontro a nossa linha inicial de não transmitir nenhuma informação relacionada com o mapa de disparidades. Foi então quando tivemos a idéia de codificar o primeiro par de ROBs com o MMP estéreo, sem utilizar no entanto o mapa de disparidades para tal. Uma vez codificado o primeiro par de ROBs, calculamos o mapa de disparidades entre este par de ROBs reconstruído, e utilizamos este mapa para codificar o par de ROBs seguinte. Fizemos os testes e um estudo estatístico que nos assegurou ser uma boa aproximação utilizar o mapa de disparidades obtido entre um par de ROBs anteriores para codificar um par de ROBs seguinte, e assim conseguimos obter nossos primeiros resultados utilizando as informações do mapa de disparidade, sem no entanto ter que transmiti-lo ao decodificador. Baseado na informação de disparidade do par de ROBs correspondentes anteriores, geramos os elementos que chamamos de "deformados", que são a base deste capítulo. Em seguida, baseado também na informação do mapa de disparidades obtido entre o par de ROBs anteriores, geramos elementos deslocados limitados por uma janela que possui como elemento central o elemento encontrado como sendo o mais próximo do bloco correspondente, baseado na informação de disparidade obtida entre um par de ROBs correspondentes anteriores. Esta janela apresentou uma distribuição mais concentrada do que a janela de busca utilizada em versões anteriores (versão 6 do Cap. 4).

Introduzimos também os elementos rotacionados e simétricos, que mostraram ser importantes para a codificação da imagem esquerda, principalmente para imagens que possuiam padrões simétricos, como a imagem MAN.

Com este capítulo encerramos as implementações desta tese. Conseguimos atingir nosso objetivo principal de explorar as características das imagens estéreo para conseguir melhores resultados de compressão utilizando o MMP.

Capítulo 6

Conclusões

Esta tese teve como objetivo principal explorar eficientemente as características das imagens estéreo, para, utilizando o MMP, alcançar bons resultados na compressão de imagens estéreo.

Dividimos o problema em duas fases. Na primeira fase exploramos as características da imagens estéreo sem utilizar o cálculo do mapa de disparidades. Na segunda fase calculamos e utilizamos as informações do mapa de disparidade para obter elementos mais representativos para a codificação da imagem direita. No entanto, nos mantivemos na mesma linha de não transmitir o mapa de disparidades ao decodificador. Conseguimos bons resultados de compressão mesmo sem transmitir o mapa de disparidades, e com isso evitamos um aumento na taxa final.

A primeira fase, descrita no Cap. 4, teve ao todo seis versões do MMP estéreo implementadas. Na primeira versão introduzimos os elementos deslocados, que se mostraram bastante importantes para a codificação da imagem direita. Depois, feito um estudo das ocorrências de todos os elementos do dicionário, verificamos também que os elementos deslocados eram utilizados na codificação da imagem esquerda. Mais adiante, chegamos à conclusão de que, quanto mais elementos no dicionário que codifica a imagem esquerda melhor para esta imagem, e quanto menor (e com elementos mais significativos) o dicionário para codificar a imagem direita, melhor a codificação desta imagem. Sendo assim, fizemos uma das principais versões desta tese, separando os dicionários para codificar as imagens. Para codificar a imagem esquerda usamos um dicionário com elementos comuns e elementos deslocados acumulados de ROBs anteriores, e para codificar a imagem direita, o MMP estéreo

escolhia entre dois dicionários o que gerava o menor custo. Estes dois dicionários eram o dicionário comum, com os mesmos elementos comuns do dicionário utilizado para a imagem esquerda, e o dicionário com elementos deslocados da ROB esquerda correspondente. Utilizamos um *flag* para indicar qual dicionário foi usado na codificação da imagem direita. Mostramos que este *flag* não onerou a taxa porque a ocorrência de deslocados na imagem direita foi muito grande, compensando o uso deste *flag*.

Depois fizemos algumas melhorias no dicionário de deslocados, que foram limitar os elementos deslocados à uma janela de busca, o que aumentou a probabilidade de encontrar a correspondência entre imagem esquerda e direita; e gerar elementos deslocados de 1/2 pixel, que aumentou a precisão dos elementos deslocados.

Com isto encerramos a primeira fase desta tese. Passamos então a considerar o cálculo do mapa de disparidades para conseguir uma maior eficácia na codificação da imagem direita. Nesta segunda fase implementamos algumas versões do MMP estéreo, descritas no Cap. 5 desta tese.

A primeira versão do MMP estéreo que introduz o cálculo do mapa de disparidades é a versão 7. Codificamos o primeiro par de ROBs correspondentes, utilizando apenas o dicionário com elementos comuns. Depois, calculamos o mapa de disparidades por pixel entre este par de ROBs reconstruídas. Fazemos isto porque o decodificador só tem disponível estas informações, ele não tem as ROBs originais. Depois utilizamos este mapa de disparidade para gerar elementos que chamamos de deformados. Estes elementos deformados são gerados aplicando o mapa de disparidades na próxima ROB esquerda. Estes elementos deformados serão utilizados para codificar a próxima ROB direita. Mostramos que mesmo utilizando um mapa obtido de ROBs reconstruídas, e mesmo utilizando um mapa de um par de ROBs anteriores para codificar a próxima ROB, obtivemos resultados bastante satisfatórios.

Utilizamos a informação de disparidade para encontrar uma janela para os elementos deslocados. O elemento central desta janela era dado por um elemento que representava o bloco correspondente na imagem esquerda. Fizemos um estudo da distribuição das ocorrências dos elementos deslocados com esta nova janela, e vimos que havia uma concentração maior das ocorrências em torno do elemento central. Achamos que isto poderia melhorar os resultados de compressão da imagem

direita, mas pudemos verificar uma melhora apenas em uma das imagens testadas. Este é um problema que precisa ser mais investigado.

Introduzimos também elementos rotacionados e simétricos no dicionário utilizado para codificar a imagem esquerda. Mostramos que conseguimos melhorar a codificação desta imagem, principalmente quando os padrões da imagem possuiam simetrias.

Com isto encerramos a segunda fase desta tese. Conseguimos atingir nosso objetivo principal. Alcançamos resultados satisfatórios, tanto na codificação da imagem esquerda, quanto na codificação da imagem direita, explorando eficientemente as características das imagens estéreo utilizando o MMP, sem no entanto precisar transmitir o mapa de disparidades, o que todos os métodos propostos até hoje para codificar imagens estéreo faziam. Evitamos assim o aumento na taxa devido à transmissão desta informação, sem penalizar a distorção, conseguindo um bom resultado final na compressão.

Algumas sugestões para continuidade deste trabalho são:

- Melhorar o algoritmo MMP : torná-lo melhor para imagens suaves, melhorando a continuidade entre os blocos. Talvez usar um dicionário inicial mais suave possa reduzir os efeitos de blocos.

- Tentar utilizar um modelo geométrico para estimar os elementos deformados com mais precisão. Isto melhoraria a compressão da imagem direita.

- Tentar usar um método mais preciso [10] para obtenção do mapa de disparidades que vai ser utilizado para estimar as próximas ROBs.

- A aplicação do MMP estéreo na codificação de vídeo estéreo. Uma das diferenças essenciais é que a exploração da similaridade temporal entre quadros vizinhos deverá ser feita considerando o movimento nas duas direções (vertical e horizontal) e não só horizontal, como foi considerado na exploração da similaridade espacial entre as imagens esquerda e direita do par estéreo.

Referências Bibliográficas

[1] DHOND, U. R., AGGARWAL, J. K., "Structure from Stereo - A Review", *IEEE Transactions on Systems, Man, and Cybernetics*, v. 19, n. 6, pp. 1489–1510, December 1989.

[2] MARR, D., *Vision: A Computational Investigation into the Human Representation and Processing of Visual Information*. W. H. Freeman and Company, 1982.

[3] FAUGERAS, O., *Three-Dimensional Computer Vision: A Geometric Viewpoint*. Cambridge, Massachusetts, MIT Press, 1993.

[4] HORN, B. K. P., *Robot Vision*. The MIT Press, 1986.

[5] BARNARD, S., FISCHLER, M., "Computational Stereo", *ACM Comput. Surveys*, v. 14, n. 4, pp. 553–572, December 1982.

[6] NALWA, V. S., BINFORD, T. O., "On Detecting Edges", *IEEE Transactions on Pattern Anal. Machine Intell.*, v. PAMI-8, n. 6, pp. 699–714, November 1986.

[7] P. SAINT-MARC, J.-S. C., MEDIONI, G., "Adaptive Smoothing: A General Tool for Early Vision", *IEEE Transactions on Pattern Anal. Machine Intell.*, v. 13, n. 6, pp. 514–529, June 1991.

[8] GOMILA, C., MEYER, F., "Levelings in Vector Spaces". In: *ICIP*, 1999.

[9] BOSWORTH, J., ACTON, S. T., "The Morphological Lomo Filter for Multiscale Image Processing". In: *ICIP*, 1999.

[10] KANADE, T., OKUTOMI, M., "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, September 1994.

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

[12] WOO, W., ORTEGA, A., "Optimal Blockwise Dependent Quantization for Stereo Image Coding", *IEEE Transactions on Circuits and Systems for Video Technology*, v. 9, n. 6, pp. 861–867, September 1999.

[13] MOELLENHOFF, M., MAIER, M. W., "Characteristics of Disparity-Compensated Stereo Image Pair Residuals", *Signal Processing: Image Communication*, v. 14, pp. 55–69, 1998.

[14] PAGLIARI, C. L., *Perspective-View Image Matching in the DCT Domain*. Phd. thesis, University of Essex, Department of Electronic Systems Engineering, June 1999.

[15] BALMELLI, L., KOVACEVIC, J., "Quadtrees for Embedded Surface Visualization: Constraints and Efficient Data Structures". In: *ICIP*, 1999.

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

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

[18] PATRICE WILLEMIN, T. R. R., KUNT, M., "Image Sequence Coding by Split and Merge", *IEEE Transactions on Communications*, v. 39, n. 12, pp. 1845–1855, December 1991.

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

[20] CARVALHO, M. B., *Multidimensional Multiscale Parser - MMP*. Phd. thesis, COPPE/UFRJ, Programa de Engenharia Elétrica, Abril 2001.

[21] SAID, A., PEARLMAN, W. A., "A New, Fast and Efficient Image Codec Based on Set Partitioning in Hierarchical Trees", *IEEE Transactions on Circuits and Systems for Video Technology*, v. 6, n. 3, pp. 243–250, June 1996.

[22] ZIV, J., LEMPEL, A., "Compression of Individual Sequences Via Variable-Rate Coding", *IEEE Transactions on Information Theory*, v. it-24, n. 5, pp. 530–536, September 1978.

[23] ATALLAH, M., GENIN, Y., SZPANKOWSKI, W., "Pattern Matching Image Compression: Algorithmic and Empirical Results", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, v. 21, pp. 618–627, 1999.

[24] ALZINA, M., SZPANKOWSKI, W., GRAMA, A., "2D-Pattern Matching Image and Video Compression", *IEEE Transactions on Image Processing*, , 2001.

[25] CARVALHO, M. B., SILVA, E. A. B. D., FINAMORE, W. A., "Rate Distortion Optimized Adaptive Multiscale Vector Quantization". In: *ICIP*, v. II, pp. 439–442, Thessaloniki, October 2001.

[26] CARVALHO, M. B., SILVA, E. A. B. D., FINAMORE, W. A., "Multidimensional Signal Compression using Multiscale Recurrent Patterns", *Signal Processing - Special Issue on Image and Video Coding Beyond Standards*, , 2002. To appear.

[27] CARVALHO, M. B. D., SILVA, E. A. B. D., FINAMORE, W. A., "Universal Multiscale Matching Pursuits", *Revista da Sociedade Brasileira de Telecomunicações*, v. 15, n. 1, pp. 25–33, June 2000.

[28] SHANNON, C. E., "Coding Theorems for a Discrete Source with a Fidelity Criterion". In: *IRE National Convention Record*, 1959.

[29] RAMCHANDRAN, K., ORTEGA, A., "Rate-Distortion Methods for Image and Video Compression", *IEEE Signal Processing*, v. 15, n. 6, pp. 23–50, November 1998.

[30] TSENG, B., ANASTASSIOU, D., "A Theoretical Study on an Accurate Reconstruction of Multview Images Based on the Viterbi Algorithm". In: *ICIP*, v. II, pp. 378–381, 1995.

[31] HASKELL, B. G., PURI, A., NETRAVALI, A., *Digital Video: An Introduction to MPEG-2*. New York, Chapman & Hall, 1997.

[32] TZOVARAS, D., GRAMMALIDIS, N., STRINTZIS, M. G., "Object-Based Coding of Stereo Image Sequences using Joint 3-D Motion/Disparity Compensation", *IEEE Transactions on Circuits and Systems for Video Technology*, v. 7, n. 2, 1997.

[33] TZOVARAS, D., BOULGOURIS, N. V., ARGYRAKI, K., *et al.*, "Rate-Distortion Optimal Fast Thresholding for MPEG-2 Image Sequence", *European Transactions on Telecommunications*, v. 12, n. 3, May 2001.

[34] XU, J., XIONG, Z., LI, S., "High Performancd Wavelet-Based Stereo Image". In: *ISCAS*, May 2002.

[35] ARDUINO, A., *Programação Dinâmica*. COPPE-UFRJ, Rio de Janeiro, 1972.

[36] FORNEY, G., "The Viterbi Algorithm", *Proceedings of The IEEE (Invited paper)*, v. 61, n. 3, pp. 268–278, March 1973.

[37] DUARTE, H., CARVALHO, M., SILVA, E., *et al.*, "Stereo Image Coding Using Multiscale Recurrent Patterns". In: *ICIP*, September 2002.

[38] DUARTE, H., CARVALHO, M., SILVA, E., *et al.*, "A New Method to Compress Stereo Image Using Multiscale Recurrent Patterns, Without the Use of Disparity Map". In: *ITS*, September 2002.

[39] WOO, W., ORTEGA, A., IWADATE, Y., "Stereo Image Coding Using Hierarquical MRF Model and Selective Overlapped Block Disparity Compensation". In: *ICIP*, Kobe, October 1999.

Apêndice A

Descrição do algoritmo *Optimal Dependent Quantisation* – ODQ

A.1 Introdução

Neste capítulo iremos descrever o funcionamento do algoritmo *Optimal Dependent Quantisation* – ODQ, utilizado no Cap. 3 e no Cap. 4 desta tese para efeito de comparação com o MMP.

O algoritmo ODQ, proposto por Woontack Woo [11], é um método para codificação de imagens estéreo, baseado em JPEG, que tem como principal característica a quantização ótima de cada bloco da imagem. Esta quantização é chamada de "dependente" porque a otimização dos quantizadores ótimos da imagem direita é atrelada aos quantizadores ótimos encontrados para a imagem esquerda do par estéreo. A otimização dos quantizadores é feita utilizando algoritmo de Viterbi [35, 36].

A.2 Sistemas de codificação preditivos

Geralmente um sistema de codificação preditivo inclui as seguintes etapas:

1. Estimação e compensação de deslocamento (disparidade ou movimento);
2. Transformação/quantização;
3. Codificação por entropia.

O desempenho da codificação pode ser controlado por vários fatores. Para imagens estéreo em particular, uma predição eficiente deve reduzir a redundância binocular que existe entre as duas imagens do par. Uma quantização ótima que leve em conta a dependência binocular pode melhorar ainda mais o desempenho da codificação como um todo. Woontack Woo aborda exatamente estes dois fatores: estimação / compensação de disparidade eficiente e alocação de bits ótima.

A.3 Formulação do problema: codificação dependente

Na estrutura de codificação preditiva, uma imagem é selecionada como referência (F_1) e a imagem dependente ou destino (F_2) é estimada a partir da imagem de referência. Similar a outros cenários de codificação preditiva, a estimação / compensação de deslocamento reduz a redundância entre as duas imagens do par estéreo. Ao invés de codificar a imagem destino original (F_2), o mapa de disparidades resultante (DV) e o quadro de disparidade compensada (DCD) é que são codificados. O quadro DCD é calculado pela diferença pixel a pixel entre a imagem destino original (F_2) e a imagem destino estimada ($\hat{F}2$).

Sejam R e D a taxa e a distorção respectivamente. Para otimizar a eficiência da codificação, os bits dados têm que ser distribuidos entre as duas imagens de um par estéreo, minimizando a distorção total. Distribuir os bits considerando as duas imagens conjuntas, e não cada uma independente, é chamada alocação de bits dependente. Na codificação de disparidade compensada, a imagem destino no par estéreo é substituida pelo mapa de disparidades (DV) e pelo quadro DCD, e assim os bits dados são distribuidos entre a imagem de referência (F_1), o mapa DV e o quadro DCD.

Dada uma taxa de bits R_{budget} o problema de alocação de bits ótima para imagens estéreo pode ser formulado como a seguir.

Dado F_1, F_2, R_{budget}

Encontrar $(V, Q_1, Q_2)^*$

Tal que $(V, Q_1, Q_2)^* = minD_1(Q_1) + \alpha D_2(V, Q_2|Q_1(V))$

Sujeito a $R_1(Q_1) + R_2(V, Q_2|Q_1(V)) \leq R_{budget}$

onde V e Q se referem a um mapa DV e um conjunto de quantizadores, respectivamente.

A importância relativa de D_1 e D_2 pode ser controlada pela constante α, que nos permite admitir duas diferentes visões do processo de percepção de profundidade:

1. **Teoria da fusão:** ambas as imagens do par estéreo contribuem igualmente na percepção 3D;

2. **Teoria da supressão:** a imagem de melhor qualidade domina a percepção. De acordo com esta teoria, a imagem destino em um par estéreo pode ser altamente comprimida sem perda na percepção final, contanto que a imagem de referência contenha os detalhes da cena.

   Nos experimentos da tese $\alpha = 1$, que obedece a teoria da fusão.

A.3.1 Estrutura loop fechado

Os vetores de disparidade V são obtidos a partir de F_2 e de $F_1(Q_1)$. Portanto V depende de Q_1. O DCD é obtido entre F_2 e $F_1(Q_1)$, assim o DCD depende de Q_1. O DCD é quantizado com Q_2. O DCD também depende de V, porque dependendo de como foi feita a estimação (com mais ou menos precisão), o DCD resultante será diferente. Deseja-se achar os melhores quantizadores Q_1 e Q_2 que minimizem o custo RD entre o par estéreo, mas para se estimar V já é preciso uma quantização inicial Q_1. Esta dependência entre os pares estéreo parece muito complicada de explorar, porque a estimação de disparidade e a quantização estão acopladas. Esta estrutura, chamada de "Loop fechado", é um problema complicado de otimização.

Na tese este problema é dividido em dois problemas de otimização usando uma estrutura de "Loop aberto":

1. Uma estimação de disparidade eficiente é realizada nos dados originais (não quantizados);

2. Uma quantização dependente ótima é realizada após o mapa DV ter sido determinado.

A.3.2 Estrutura loop aberto

V é obtido a partir de F_1 original (sem ser quantizado) e de F_2. Uma parte dos bits é alocada para V, e o restante é que vai ser alocado conforme a otimização da distorção $(D_1 + D_2)$. V não depende mais de Q_1. Assim, pode-se fazer a predição da melhor maneira possível independente da quantização. Com uma predição eficiente, pode-se achar a quantização ótima, de forma a tornar a codificação o melhor possível segundo o critério RD. Note que a disparidade estimada no codificador "Loop aberto" é sub-ótima, porque a estimação é realizada entre F_1 e F_2, enquanto a compensação é realizada entre $F_1(V, Q_1)$, que é o disponível no decodificador. Porém, o codificador "loop aberto" tende geralmente a gerar um mapa DV consistente e relativamente preciso, comparado com o codificador "loop fechado".

No decodificador, primeiro a imagem referência é decodificada e depois a imagem destino é reconstruída, a partir do frame de referência, do mapa de disparidades e do quadro DCD.

A.4 Quantização dependente ótima por blocos

As pesquisas em codificação de imagens estéreo têm focalizado a estimação / compensação de disparidade, que objetiva explorar a redundância entre as duas imagens estéreo. Menos atenção tem sido dedicada ao problema igualmente importante de alocar bits entre duas imagens. O desempenho da codificação também depende de uma escolha apropriada dos quantizadores (Q_1, Q_2), e não só da escolha dos vetores de disparidade (V). Em [11] é abordado o problema de alocação de bits bloco a bloco, para a codificação de imagens estéreo. O algoritmo de Viterbi é usado para encontrar o conjunto de quantizadores ótimos.

Sejam:

- F_1 : imagem de referência, segmentada em N blocos quadrados.
- $F_1 = B_m, 0 \leq m \leq N-1$, onde B_m representa o m-ésimo bloco da imagem.
- F_2 : imagem destino.
- $F_2 = B_{m\prime}, 0 \leq m\prime \leq N-1$.
- $Q_1 = q_m, 0 \leq m \leq N-1$: quantizador para cada bloco da imagem de referência F_1.
- $Q_2 = p_m, 0 \leq m \leq N-1$: quantizador para cada bloco do quadro DCD.
- $V = v_m, 0 \leq m \leq N-1$, disparidade para o m-ésimo bloco em F_2.

Cada bloco vai ter uma taxa e uma distorção. A taxa e a distorção total são a soma das taxas e distorções de cada bloco.

Medidas de distorção:

- $D_1 = (F_1 - F_1(Q_1))^2$
- $D_2 = (F_2 - \hat{F}_2(Q_1, Q_2, V))^2$, onde $F(Q)$ é a imagem decodificada quando o quantizador Q é usado.

A imagem destino decodificada, $\hat{F}_2$, pode ser reconstruida adicionando a imagem referência compensada com o mapa DV, e o DCD decodificado, isto é:

$$\hat{F}_2(Q_1, Q_2, V) = F_1(Q_1, V) + E(Q_2)$$

, onde $E = F_2 - F_1(Q_1, V)$.

O PSNR médio é definido por:

$PSNR_{medio} = 10 \log 255^2/[(D_1 + D_2)/2]$,

onde D_1 e D_2 são o erro médio quadrático das imagens reconstruídas, F_1 e F_2, respectivamente.

Seja R_{budget} o restante de bits disponível após a alocação para o mapa DV. Para um dado mapa V o problema de alocação de bits ótima pode ser formulado como a seguir:

Dado F_1, F_2, V, R_{budget}

Encontrar $X = (Q_1, Q_2)$

Tal que $X = \arg\min D_1(Q_1) + D_2(Q_2|Q_1)$

Sujeito a $R_1(Q_1) + R_2(Q_2|Q_1) \leq R_{budget}$.

Este problema de otimização com restrição pode ser transformado em um problema sem restrição usando o método do multiplicador de Lagrange:

$J(\lambda) = J_1(Q_1) + J_2(Q_2|Q_1) = D_1(Q_1) + \lambda R_1(Q_1) + D_2(Q_2|Q_1) + \lambda R_2(Q_2|Q_1)$,

onde o multiplicador de Lagrange (λ) é uma constante não negativa.

A.5 Solução usando o algoritmo de Viterbi

Devido a predominância da dependência 1D, uma linha de blocos (ROB) na imagem destino depende somente da ROB na mesma posição na imagem de referência (linha epipolar). Assim, só é necessário considerar a alocação de bits para pares de ROBs, já que outras ROBs não afetam o resultado. Mesmo que exista alguma disparidade vertical, esta é uma aproximação suficientemente boa. Sejam ROB_1 e $ROB2$ ROBs na mesma posição na imagem de referência e na imagem DCD, respectivamente. Representa-se todas as alocações possíveis para cada par de ROBs construindo uma treliça. Os custos dos nós e ramos da treliça correspondem, respectivamente, aos blocos na ROB_1 e $ROB2$.

A.5.1 Definições

1. Seja k_o o índice do estágio.

2. **Estágio:** o k-ésimo estágio na treliça corresponde ao k-ésimo bloco na ROB_1. Assim, o número de estágios K é igual ao número de blocos na ROB_1.

3. **Nó:** Cada nó no k-ésimo estágio corresponde a uma possível escolha de quantizador para o késimo bloco da ROB_1. As escolhas são ordenadas do topo para a base em ordem do mais fino para o mais grosseiro. Assim, o número de nós por estágio é $L = |q|$, isto é, o número de quantizadores disponíveis para a imagem referência.

4. Cada nó tem um custo Lagrangeano correspondente, $J_1(i,k)$, que depende somente da taxa e da distorção do k-ésimo bloco da ROB_1 quando o quantizador i é usado.
   $J_1(i,k) = d(q_k^i) + \lambda r(q_k^i)$

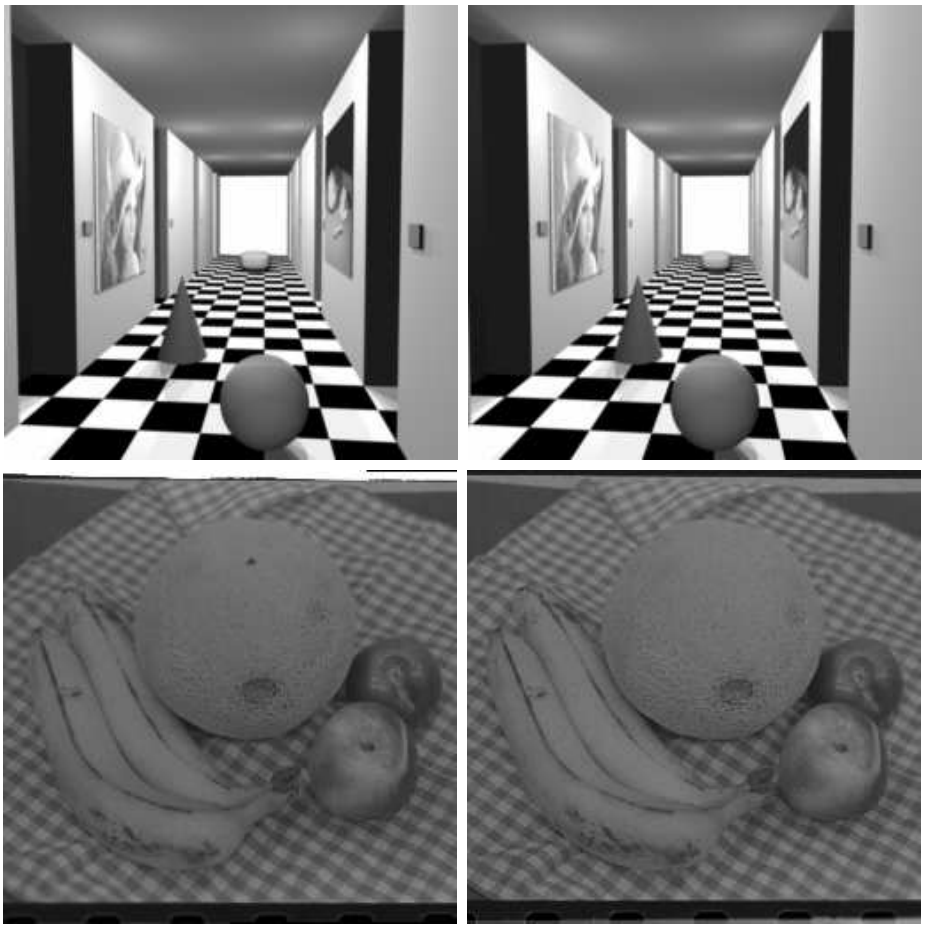
5. **Ramo:** Um ramo, ligando dois nós, corresponde ao vetor ótimo de quantizadores, para os blocos na ROB_2 que dependem dos blocos k e $k+1$ na ROB_1. Cada ramo tem um custo Lagrangeano total:
   $J_2(i,j,k) = \sum_{\eta \in \eta_2(k,k+1)} = d(p_n^i j | q_k^i, q_{k+1}^j) + \lambda r(p_n^i j | q_k^i, q_{k+1}^j)$,
   que adiciona os custos Lagrangeanos correspondendo a cada um dos blocos n.

6. **Caminho:** Um caminho é a concatenação de ramos desde o primeiro estágio até o estágio final da treliça. Cada caminho corresponde a um conjunto de escolhas de quantização para ambas as ROBs: ROB_1 (nós) e $ROB2$ (ramos). O custo de um caminho é o custo acumulado de ramos e nós ao longo do caminho.

Uma vez que a treliça tenha sido construída, o problema de quantização dependente ótima é equivalente a encontrar o caminho de menor custo a partir de um nó no primeiro estágio até um nó terminal no último estágio da treliça.

Apêndice B

Imagens estéreo originais

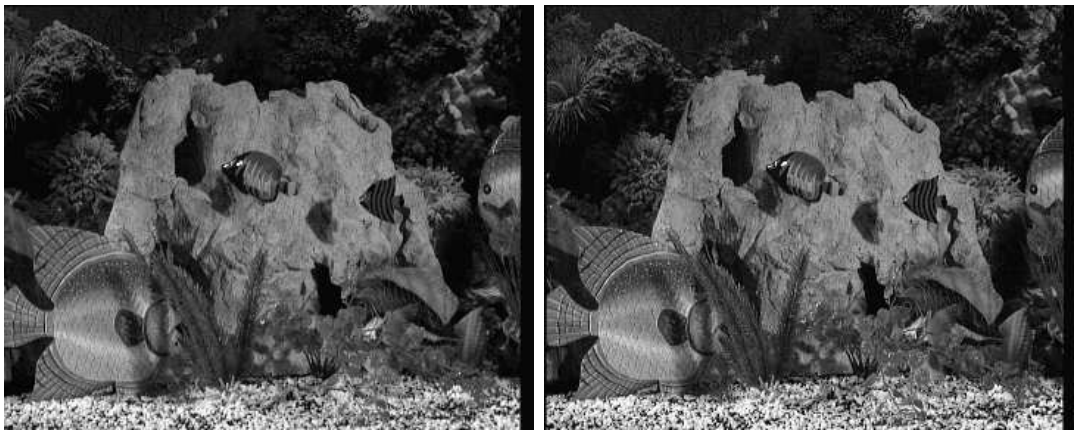


(a)

(b)

Figura B.1: (a) CORRIDOR (256x256), (b) FRUIT (256x256).

(a)



(b)

Figura B.2: (a) MAN (384x384), (b) AQUA (360x288).

(a)

(b)

Figura B.3: (a) SAXO (700x576), (b) TRAIN (700x576).

Origem destas imagens:

CORRIDOR: Computer Vision and Pattern Recognition Group, University of Bonn.

FRUIT: http://vasc.ri.cmu.edu//idb/html/stereo/fruit/index.html.

MAN: HHI: Heinrich-Hertz-Institut, Berlin, Germany.

AQUA, SAXO e TRAIN: Centre Commun d'Etudes de Télédiffusion et Télécommunications (CCETT), France. (test sequences shot and distributed under the RACE DISTIMA European Project).

Apêndice C

Descrição das versões do MMP estéreo

As versões do MMP estéreo desenvolvidas ao longo da tese representam a evolução do algoritmo MMP ao explorar as características das imagens estéreo. A cada versão verificamos melhorias consideráveis, como pudemos comprovar nos diversos resultados apresentados nos capítulos 4 e 5.

As versões, cada uma com sua respectiva descrição, são enumeradas a seguir:

1. **Versão 1**

   A primeira tentativa no sentido de explorar as características estéreo das imagens foi introduzir no dicionário elementos formados à partir de deslocamentos verticais de um pixel entre dois blocos vizinhos de uma ROB da imagem esquerda. Estes deslocamentos são gerados a cada dois blocos de uma ROB esquerda, e os elementos resultantes são inseridos imediatamente no dicionário, de forma que no processamento do próximo bloco estes elementos já podem ser utilizados na codificação. A versão 1 do MMP utiliza um mesmo dicionário para codificar as ROBs das imagens esquerda e direita. Nesta versão os elementos deslocados inseridos no dicionário durante a codificação de uma ROB esquerda são sobrepostos pelos elementos deslocados da próxima ROB esquerda. Isto está de acordo com a restrição da linha epipolar que estabelece a correspondência somente entre ROBs da mesma linha epipolar.

2. **Versão 2**

   A versão 2 do MMP acumula os elementos deslocados de ROB para ROB. O dicionario só cresce. Isto melhora a compressao da imagem esquerda, pois aumenta a freqüência dos elementos no modelo sem aumentar significativamente o número de elementos do modelo, pois o número total de elementos já é muito grande. O resultado piora para a imagem direita, porque o modelo desta imagem possui poucos elementos. Quando aumentamos o numero de elementos deste modelo com elementos não significativos (com freqüência de ocorrência baixa), aumentamos a taxa porque o codificador aritmético gasta mais bits para codificar estes elementos que não ocorrem muito.

3. **Versão 3**

   Baseado nos resultados apresentados pelas duas versões anteriores chegamos a seguinte conclusão: Inserir no dicionário elementos deslocados acumulados de ROBs anteriores é bom para a codificação da imagem esquerda, mas piora a codificação da imagem direita. Esta versão apresenta uma solução para este problema. Usamos dois dicionários separados, com modelos separados, para codificar cada imagem. No dicionário utilizado para codificar a imagem esquerda usaremos os elementos iniciais, concatenados, contraídos e expandidos (chamados a partir de agora de elementos comuns), e os elementos deslocados acumulados de ROBs anteriores. Isto irá fornecer um bom resultado para a codificação da imagem esquerda. Para codificar a imagem direita usaremos os elementos comuns e os elementos deslocados somente da ROB esquerda correspondente. Os elementos deslocados ocorrem muito na imagem direita, isto reduz bastante a taxa de compressão, o que compensa o *overhead* de usar um *flaf* para indicar qual dos dois dicionários foi utilizado. Desta forma conseguimos melhorar a compressão da imagem direita, da mesma forma que na versão 2, mantendo no entanto o mesmo resultado da versão 1 para a imagem esquerda.

4. **Versão 4**

   Na versão anterior (versão 3), os elementos deslocados foram gerados em toda a extensão da ROB esquerda. Isto faz com que o número de elementos deslocados inseridos no dicionário para codificar a imagem direita seja muito grande, e contenha elementos fora da janela de busca, que não seriam blocos correspondentes ao bloco da imagem direita que está sendo processado. Nesta nova versão (versão 4), geramos os elementos deslocados apenas dentro da janela de busca equivalente ao bloco da imagem direita que está sendo processado. Com isso melhoramos a codificação da imagem direita porque inserimos no dicionário apenas elementos deslocados que poderão corresponder ao bloco que está sendo processado, ou seja, melhoramos a estimativa do modelo que irá representar a imagem direita.

5. **Versão 5**

   Na versão 3 os elementos deslocados foram formados por um deslocamento de 1 pixel. Em codificação de vídeo monocular foi comprovado que utilizar estimação de movimento de 1/2 pixel melhora o resultado da codificação, porque diminui a diferença entre o bloco original e o bloco estimado. Resolvemos então gerar elementos deslocados de 1/2 pixel para testar se isto melhoraria os resultados de compressão da imagem direita. Implementamos isto nesta versão 5, tendo como base a versão 3, para sabermos isoladamente o benefício de usar deslocados de 1/2 pixel, sem usar janela de busca. Os resultados mostraram uma melhoria considerável na codificação da imagem direita. Testamos também gerar deslocamentos de 1/4 de pixel, mas isto piorou o resultado. Por isto adotamos o deslocamento de 1/2 pixel como a melhor opção nas versões de agora em diante.

6. **Versão 6** Resolvemos incorporar todas as melhorias obtidas com as versões 3, 4 e 5, e criamos esta nova versão (que encerra o Cap. 4 desta tese).

7. **Versão 7**

   Introduzimos os elementos "deformados" no dicionário. Estes elementos são obtidos da seguinte forma: Após processarmos uma ROB da imagem esquerda e uma ROB da imagem direita, calculamos o mapa de disparidades por pixel entre estas duas ROBs. Logo depois processamos a ROB esquerda seguinte, utilizamos o mapa de disparidades obtido entre as ROBs anteriores e geramos elementos que são estimativas da ROB direita seguinte (estes elementos nós chamamos de "deformados" porque eles podem representar deformações de perspectiva devido à distância entre as duas câmeras . Com estes elementos no dicionário esperamos melhorar a codificação da imagem direita.

8. **Versão 8**

   A primeira melhoria realizada na versão 7, que introduz os elementos deformados, foi reduzir o número de elementos deformados a uma janela que possui como centro o elemento deformado correspondente ao bloco que está sendo processado. Este elemento deformado correspondente é dado pelos vetores de disparidades calculados para o par de ROBs anteriores. Esta janela é um parâmetro dado pelo usuário. Fizemos testes para janelas de vários tamanhos, mas a que deu melhores resultados foi uma janela com um único elemento deformado, exatamente aquele bloco equivalente ao bloco da imagem direita que está sendo processado. Este procedimento equivale a um refinamento na estimativa do modelo, onde reduzimos o número de elementos e utilizamos um elemento mais significativo para representar o bloco da imagem direita.

9. **Versão 9**

   Uma vez comprovada a eficácia de usar elementos deformados para codificar a imagem direita, trouxemos de volta os elementos deslocados para o mesmo dicionário que contém os deformados, ou seja, o dicionário utilizado para codificar a imagem direita. Estes elementos deslocados são gerados para todos os blocos da ROB, e não só para a janela de busca correspondente ao bloco da imagem direita que está sendo processado.

10. **Versão 10**

    Quando nós restringimos os elementos deformados a uma janela (no caso, com um único elemento), verificamos que o resultado de compactação melhorou para a imagem direita, porque melhoramos a estimativa do modelo que representaria os blocos da imagem direita. Podemos fazer o mesmo para os elementos deslocados. Conhecendo a disparidade média de um dado bloco, podemos pegar uma janela de deslocados em torno do elemento que corresponderia a este bloco na outra imagem. Com isso tende-se a escolher para esta janela os elementos deslocados que possuem maior probabilidade de ocorrer para o bloco que está sendo processado na imagem direita. Nesta versão incluimos também os deslocados dos deformados no dicionário que é usado para codificar a imagem direita.

11. **Versão 11**

    Até agora só mostramos as melhorias ocorridas na codificação da imagem direita. Uma vez que tinhamos removido os elementos deslocados acumulados do dicionário comum, os resultados da codificação da imagem esquerda haviam piorado comparado com os últimos resultados do capítulo 4. Agora trazemos de volta estes elementos, melhorando a codificação da imagem esquerda e mantendo, no entanto, todas as melhorias obtidas na codificação da imagem direita (às vezes até melhorando, uma vez que melhorou a estimativa da imagem esquerda).

12. **Versão 12**

    E, finalmente, introduzimos elementos obtidos de rotações dos blocos que melhor representaram os blocos da imagem esquerda no dicionário comum. Ao testarmos, porém, a inserção destes elementos rotacionados apenas no dicionário que vai codificar a imagem esquerda, o resultado foi melhor do que usando rotações para codificar também a imagem direita. Isto indica que para codficar a imagem direita os elementos mais relevantes são os deslocados e os deformados obtidos usando os vetores de disparidade. Para algumas imagens as rotações não foram relevantes. Estas rotações foram de 90, 180, 270, simetria horizontal, simetria vertical, simetria em torno da diagonal principal, simetria em

torno da diagonal secundária. Para imagens como MAN, na qual as simetrias podem ser muito relevantes, observamos que realmente houve uma melhoria da introdução dos elementos rotacionados.

Apêndice D

Descrição dos testes realizados

D.1 Testes realizados no Cap. 3

1. MMP comparado com ODQ aplicados à imagem esquerda e ao quadro de diferenças DCD. Imagens utilizadas: CORRIDOR, FRUIT e MAN.

2. MMP comparado com ODQ, aplicados diretamente às imagens esquerda e direita, sem cálculo do quadro de diferenças DCD. Imagens utilizadas: CORRIDOR, FRUIT e MAN.

3. MMP aplicado às imagens esquerda e direita, comparado com ODQ aplicado à imagem esquerda e ao quadro de diferenças DCD. Imagens utilizadas: CORRIDOR, FRUIT e MAN.

D.2 Testes realizados no Cap. 4

1. MMP aplicado às linhas de blocos esquerda e direita (MMP ROB a ROB), comparado com MMP aplicado à imagem esquerda completa e depois à imagem direita completa. Resultados para a imagem estéreo CORRIDOR.

2. Comparação entre o MMP estéreo versão 1, que introduz elementos deslocados, e MMP ROB a ROB (sem elementos deslocados). Resultados para a imagem estéreo CORRIDOR.

3. Comparação entre a versão 2 do MMP estéreo, que introduz os elementos

deslocados acumulados no dicionário, e a versão 1 do MMP estéreo, que não acumula os elementos deslocados. Resultados para a imagem CORRIDOR.

4. Comparação entre a versão 3 do MMP estéreo, que usa dois dicionários, um para codificar a imagem esquerda (que inclui elementos comuns e elementos deslocados acumulados) e outro para codificar a imagem direita (escolhe entre dicionário com elementos comuns ou dicionário com elementos deslocados da ROB esquerda correspondente), e a versão 2 do MMP estéreo, que utiliza apenas um dicionário para codificar as duas imagens, contendo elementos comuns, deslocados da ROB esquerda e deslocados acumulados de ROBs anteriores. Resultados para a imagem CORRIDOR.

5. Comparação entre a versão 3 do MMP estéreo, que utiliza dois dicionários separados para codificar as imagens esquerda e direita, e a versão 1 do MMP estéreo, que utiliza apenas um dicionário e não acumula os elementos deslocados de ROBs anteriores. Resultados para a imagem CORRIDOR.

6. Comparação entre a versão 4 do MMP estéreo, que utiliza janela de busca para limitar os elementos deslocados, e a versão 3 do MMP estéreo, que não utiliza janela de busca. Resultados para a imagem direita do par estéreo CORRIDOR.

7. Comparação entre a versão 5 do MMP estéreo, que utiliza deslocamentos de 1/2 pixel, e a versão 3 do MMP estéreo, que não utiliza deslocamentos de 1/2 pixel. Resultados para a imagem direita do par estéreo CORRIDOR.

8. Comparação entre usar deslocamentos de 1/4 e 1/2 de pixel. Resultados para a imagem direita do par CORRIDOR.

9. Comparação entre a versão 6 do MMP estéreo, que inclui as melhorias das versões 3, 4 e 5, e a versão 1 do MMP estéreo, que possuía apenas os elementos comuns e os deslocados em um só dicionário. Resultados para as imagens estéreo CORRIDOR, AQUA, MAN, SAXO, e TRAIN.

10. Comparação entre a versão 6 do MMP estéreo versão 6 e o ODQ. Resultados para as imagens estéreo CORRIDOR e AQUA.

11. Comparação entre a versão 6 do MMP estéreo e HQBM (ODQ melhorado). Resultados para as imagens estéreo CORRIDOR e AQUA.

D.3 Testes realizados no Cap. 5

1. Comparação entre a versão 7 do MMP estéreo, que introduz os elementos deformados, e a versão 8 do MMP estéreo, que reduzia o número de elementos deformados no dicionário para codificar o bloco da imagem direita a um único elemento, exatamente o correspondente ao bloco da imagem direita que está sendo processado. Resultados para a imagem direita dos pares estéreo CORRIDOR e MAN.

2. Comparação entre a versão 8 do MMP estéreo, que utiliza um elemento deformado no dicionário para codificar um bloco da imagem direita, e a versão 9 do MMP estéreo, que traz de volta os elementos deslocados para o dicionário usado para codificar a imagem direita. Resultados para a imagem direita dos pares estéreo CORRIDOR e MAN.

3. Comparação entre a versão 9 do MMP estéreo, que usa um elemento deformado e elementos deslocados da ROB esquerda correspondente (limitados pela janela de busca) para codificar um bloco da ROB direita, e a versão 10 do MMP estéreo, que limita os deslocados a uma janela obtida pela informação de disparidades. Resultados para a imagem direita dos pares estéreo CORRIDOR, MAN e SAXO.

4. Comparação entre a versão 10 do MMP estéreo, que usa um deformado e deslocados limitados pela janela obtida através das informações de disparidade para codificar um bloco da imagem direita, e a versão 11 do MMP estéreo, que traz de volta os elementos deslocados acumulados para codificar a imagem esquerda. Resultados para as imagens estéreo CORRIDOR, MAN e SAXO.

5. Comparação entre a versão 11 do MMP estéreo, que usa um deformado e deslocados limitados pela janela obtida através das informações de disparidade para codificar um bloco da imagem direita e usa um dicionário com elementos comuns e elementos deslocados acumulados para codificar a imagem esquerda, e

a versão 12 do MMP estéreo, que tem todas estas implementações da versão 11 mais os elementos rotacionados e simétricos para codificar a imagem esquerda. Resultados para a imagem direita dos pares estéreo CORRIDOR e MAN.

6. Comparação entre a versão 12 do MMP estéreo, que utiliza as informações do mapa de disparidades obtido entre duas ROBs correspondentes anteriores para estimar a ROB direita atual, e a versão 6 do MMP estéreo, que não calculava mapa de disparidades (final do Cap. 4). Resultados para as imagens estéreo CORRIDOR e MAN.

7. Comparação entre a versão 12 do MMP estéreo, que contém as melhorias implementadas no Cap. 4 e no Cap. 5, e a versão 1 do MMP estéreo, que só possuía os elementos comuns e deslocados da ROB esquerda correspondente em um único dicionário. Resultados para as imagens estéreo CORRIDOR, AQUA, MAN, SAXO, e TRAIN.

8. Comparação entre a versão 12 do MMP estéreo e o ODQ. Resultados para as imagens estéreo CORRIDOR e AQUA.

9. Comparação entre a versão 12 do MMP estéreo e o HQBM (ODQ melhorado). Resultados para as imagens estéreo CORRIDOR e AQUA.

Apêndice E

Exemplo de programação dinâmica

O exemplo abaixo foi retirado de uma apostila da COPPE/UFRJ [35], por ser bastante didático na explicação do algoritmo de programação dinâmica.

E.1 O Problema da viagem

Para iniciar a apresentação da Programação Dinâmica, vamos considerar o seguinte exemplo:

O diretor de uma empresa deseja fazer uma viagem de negócios da cidade **A**, para a cidade **J**. Consulta o seu agente de viagens e este lhe informa que não existe uma linha direta entre as duas cidades, e lhe entrega o gráfico da Fig. E.1 que indica as diversas maneiras de atingir **J**, bem como os preços de cada uma das passagens intermediárias.

O seu problema é então determinar a trajetória mais barata da cidade **A** para a cidade **J**. A primeira idéia é a de, a cada cidade, escolher sempre a próxima viagem que seja a mais barata. Desta forma a trajetória escolhida seria **ADFHJ** com um custo total de $3+9+10+7 = 29$. Verifica contudo que ir de **A** a **F** passando por **B**, o custo seria $4 + 6 = 10$, inferior portanto ao de sua trajetória escolhida (passando por **D**), que seria $3 + 9 = 12$. Conclui então que a sua idéia inicial para resolver o problema não era satisfatória.

O nosso diretor não consegue descobrir nenhuma técnica que lhe permita escolher rapidamente a trajetória ótima, e resolve o problema calculando os custos totais associados a cada possível trajetória (são 12 ao todo), decidindo-se pela de

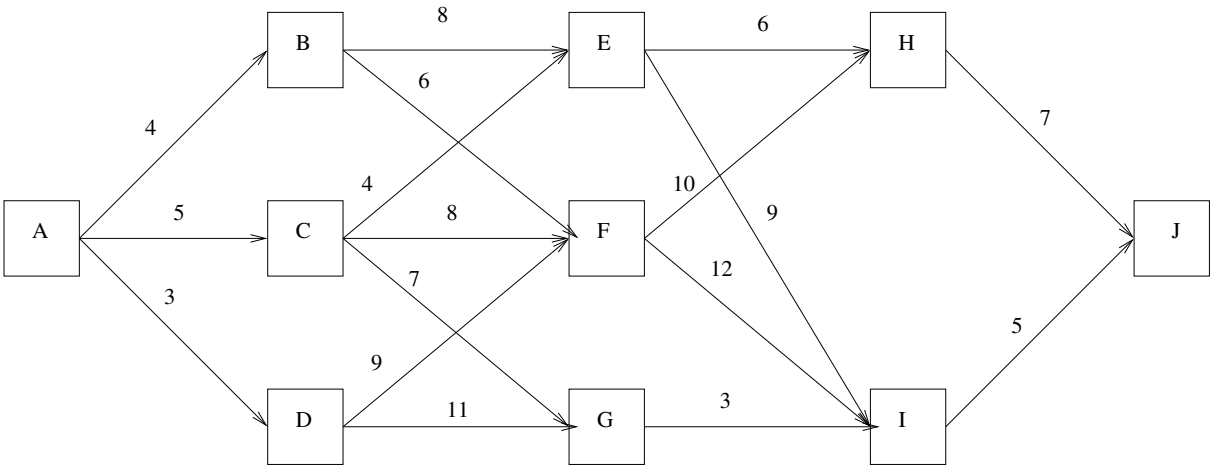


Figura E.1: Gráfico com as trajetórias possíveis para ir da cidade **A** para a cidade **J**.

menor custo (**ACGIJ**, com custo 20).

Evidentemente, inspecionar todas as possíveis trajetórias não é uma técnica muito satisfatória. Se imaginarmos um problema semelhante, com cinco cidades em cada um dos três estágios intermediários, teríamos da ordem de $5 \times 5 \times 5 = 125$ possíveis trajetórias. Seria um trabalho exaustivo calcular os custos totais para cada uma dessas trajetórias.

O problema da viagem, conforme acima enunciado, é um dos problemas clássicos da Programação Dinâmica. Vamos procurar resolvê-lo de uma maneira informal, e em seguida, desenvolveremos um algoritmo baseado no raciocínio empregado na solução. A idéia para a determinação da trajetória ótima é a seguinte:

1. Nós não sabemos, a priori, se ela passará por **H** ou por **I**. Entretanto, sabemos que se ela passar por **H**, então irá para **J** e o seu custo será adicionado de 7; da mesma forma, se ela passar por **I**, então irá para **J** e o seu custo será adicionado de 5.

   Sublinhamos então, no gráfico as trajetórias **H J** e **I J**, e escrevemos num círculo acima das cidades **H** e **I** os custos adicionais.

2. Nós não sabemos se a trajetória ótima passa por **E**, **F** ou **G**. Entretanto:

   (a) Se ela passa por **E**, chegando a **E**, duas decisões poderiam ser tomadas: ir para **H** ou ir para **I**. Se ela for para **H**, então o seu custo até **E** seria

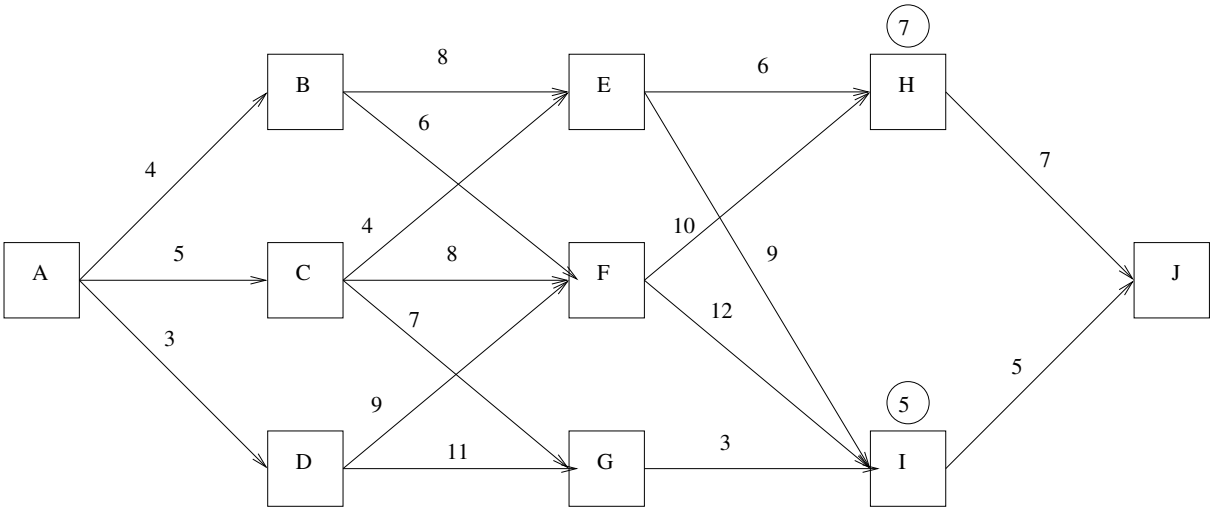


Figura E.2: Círculos acima das cidades **H** e **I** indicam os custos para ir destas cidades para a cidade **J**.

adicionado de $6 + 7 = 13$, para atingir **J**; Se ela for para **I**, o custo até **E** seria adicionado de $9 + 5 = 14$ para atingir **J**. É claro, portanto, que se a trajetória ótima passar por **E**, a próxima cidade será **H**, porque esta é a decisão que proporcionará o menor custo de **E** para **J**.

(b) Se a trajetória passa por **E**, será indiferente dirigir-se para **H** ou para **I**. Ambas as decisões proporcionarão o mesmo custo adicional: $10 + 7 = 12 + 5 = 17$.

(c) Se a trajetória ótima passa por **G**, ela terá necessariamente de dirigir-se para **I** (não há opções) e o custo adicional será $3 + 5 = 8$.

Sublinhamos então as retas **E H**, **F H** (podia ser **F I**), mas escolhemos arbitrariamente **F H**, e **G I**, que correspondem às melhores decisões se a trajetória ótima atingir as cidades **E**, **F** ou **G**, respectivamente, e , num círculo acima de cada cidade escrevemos o menor custo adicional a partir dela.

3. Aplicamos o mesmo raciocínio a B, C e D. Veja a Fig. E.4.

4. Chegamos ao ponto de partida, ou seja, a cidade **A**. É claro que a trajetória ótima necessariamente passa por **A**. De **A** podemos nos decidir ir para **B**, **C** ou **D**. Se nos decidirmos por **B**, o menor custo total que poderemos obter será

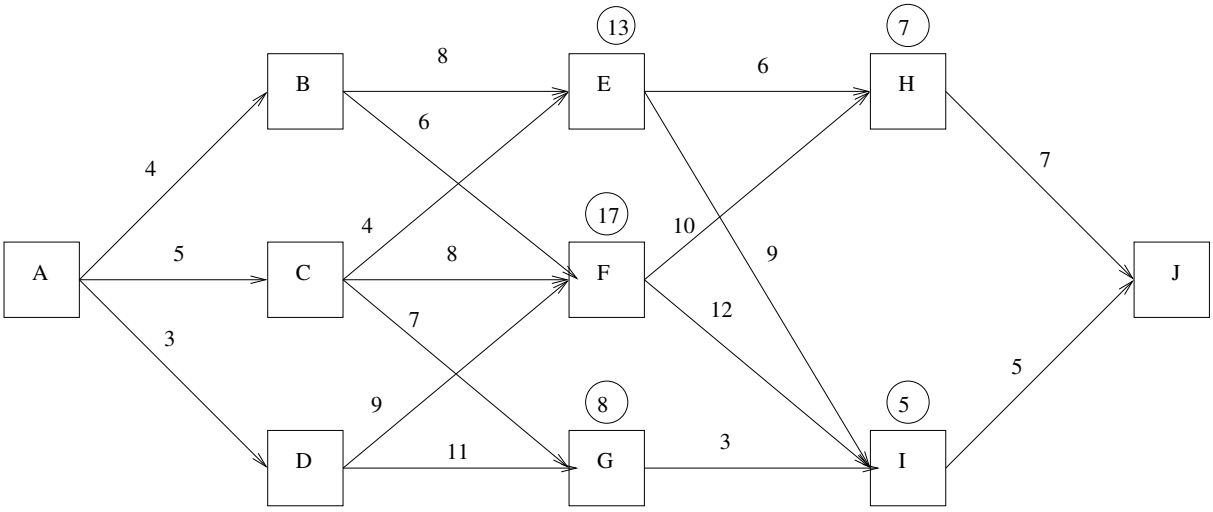


Figura E.3: Círculos acima das cidades **E**, **F** e **G** indicam os custos para ir destas cidades para a cidade **J**.

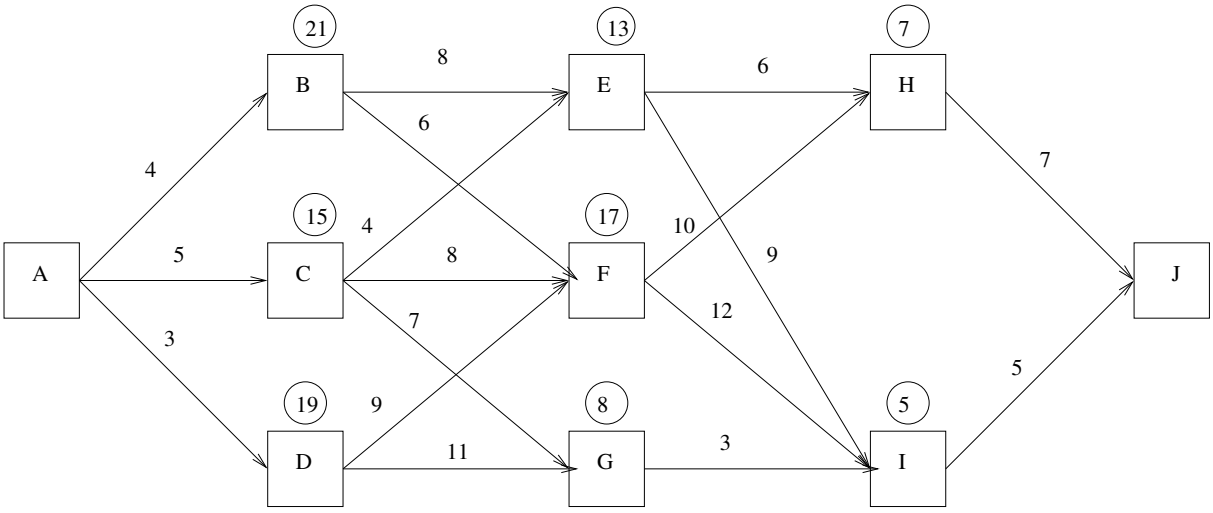


Figura E.4: Círculos acima das cidades **B**, **C** e **D** indicam os custos para ir destas cidades para a cidade **J**.

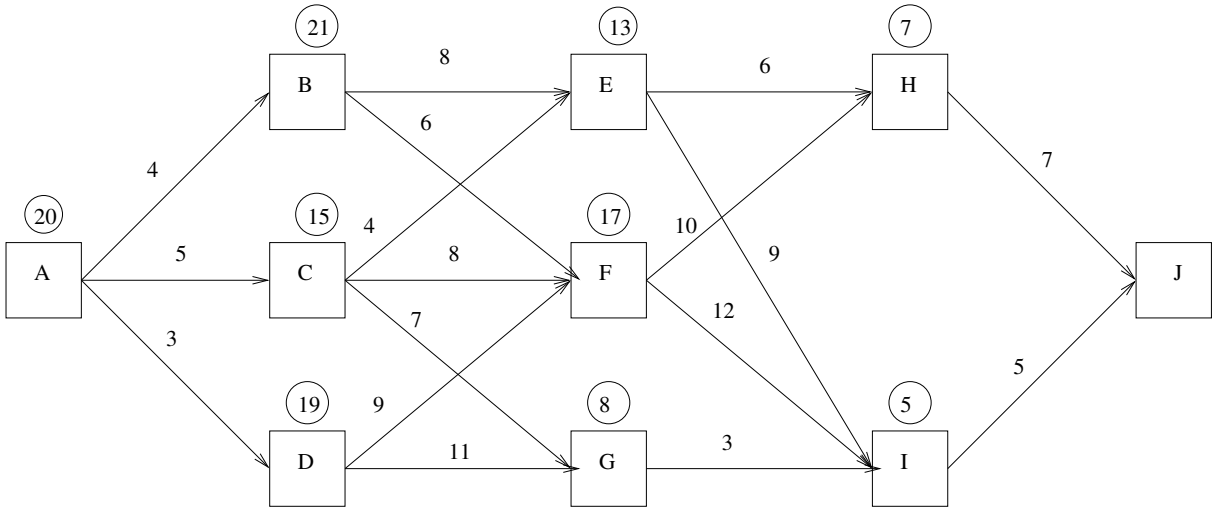


Figura E.5: Círculos acima das cidades **B**, **C** e **D** indicam os custos para ir destas cidades para a cidade **J**.

$4 + 21 = 25$. Se nos decidirmos por **C**, esse valor será $5 + 15 = 20$. Se nos decidirmos por **D**, teremos $3 + 19 = 22$.

Então, em **A**, a melhor decisão será ir para **C**. De **C**, ela prosseguiria indo para **G**. De **G** ela prosseguiria indo para **I**, e finalmente, de **I** para **J**. Determinamos assim a trajetória ótima (**A C G I J**) e seu custo mínimo (20). Veja a Fig. E.5.

E.2 Os elementos de programação dinâmica. O algoritmo

É nesse processo de reconstituir a trajetória ótima do fim para o princípio que se baseiam as técnicas de programação dinâmica. Elas se fundamentam no princípio de que:

"Uma trajetória ótima tem a seguinte propriedade: qualquer que tenha sido o primeiro passo, a trajetória remanescente deverá ser uma trajetória ótima com respeito ao estado resultante dessa primeira decisão".

Vamos agora identificar no exemplo anterior os seguintes elementos: Veja Fig. E.6.

1. Aos instantes em que se inicia ou termina uma viagem denominamos estágios. Temos neste caso, 5 estágios que identificamos pela letra k, $k = 0, 1, 2, 3, 4$.

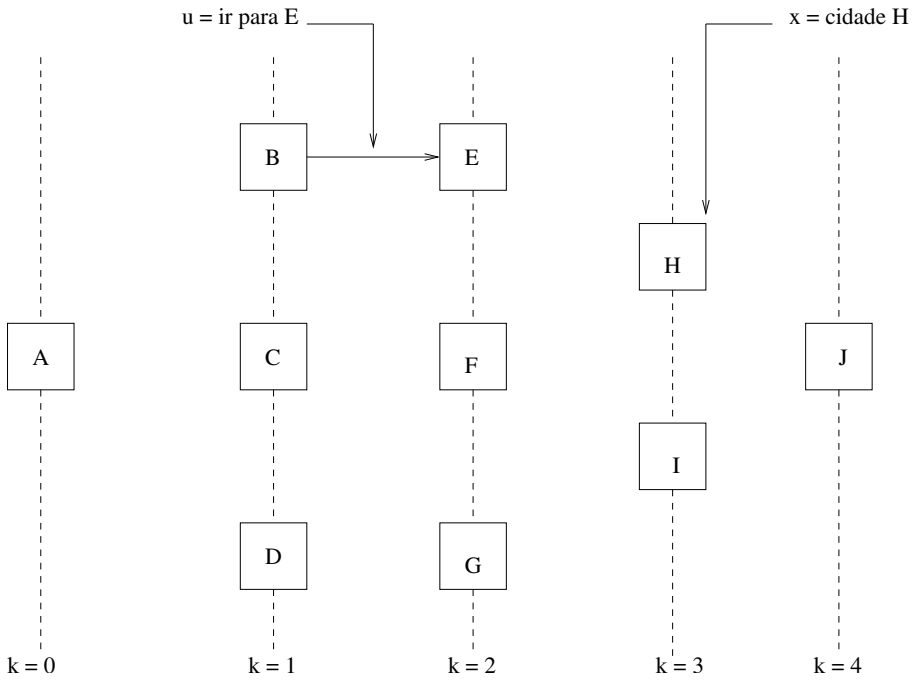


Figura E.6: Elementos de programação dinâmica.

2. Em cada estágio, o viajante encontra-se em uma cidade; A cidade ocupada num dado estágio representa, naquele instante, o estado do viajante (x). Para cada estágio há um número restrito de estados viáveis. Por exemplo: No estágio 3, os únicos estados viáveis são a cidade **H** e a cidade **I**.

3. Para ocupar um novo estado no estágio seguinte, o viajante tem de tomar uma decisão (u). Por exemplo: se o viajante se encontra no segundo estágio, no estado **B**, e deseja no estágio seguinte passar para o estado **E**, deverá tomar a decisão "ir para **E**". Note-se que, apesar do estado **E** ser viável no terceiro estágio, não podemos passar de **D** para **E**, pois no segundo estágio e no estado **D**, a decisão "ir para E" não é uma decisão admissível (é o caso, de, por exemplo, não haver nenhum vôo da cidade **D** para a cidade **E**).

4. Aos diversos custos correspondentes a cada uma das viagens intermediárias denominamos custos elementares (l). Por exemplo: o custo elementar de, no estado **C** tomar a decisão "ir para **F**" é 8.

5. O critério (J) para comparar as diversas possíveis trajetórias e indicar a trajetória ótima foi o de somar os custos elementares.

Vamos agora resolver novamente o problema da viagem com o emprego de um algoritmo que se baseará exatamente nos mesmos princípios que nortearam a nossa solução anteriormente vista. Este algoritmo parecerá, à primeira vista, ser mais complicado ou mais detalhado do que deveria ser. Isto realmente ocorre para este exemplo em particular, que é bastante simples. Entretanto esse mesmo algoritmo se adapta perfeitamente aos problemas mais complicados e dessa forma achamos conveniente não simplifica-lo nesse exemplo para que ele seja apresentado em toda a sua generalidade.

E.2.1 Convenção

- k_o: estágio sob consideração;
- x : estado viável no estágio k_o;
- u : decisão no estágio k_o e no estado x;
- $x\prime$: estado atingido pela aplicação da decisão u ao estado x;
- l : custo elementar da decisão u aplicada ao estado x no estágio k_o;
- $\bar{J}\prime$: custo adicional mínimo a partir de $x\prime$;
- $\bar{J}$: custo adicional mínimo a partir de x;
- $\bar{u}$: melhor decisão dentre as possíveis de serem aplicadas a x;

E.2.2 Observações

1. A política ótima é obtida após o término do último quadro ($k_o = 0$) e pode ser acompanhada (do último para o primeiro quadro), por estar assinalada (*).
2. O custo mínimo a ela associado é o último $\bar{J}$ encontrado (20**) e resulta imediatamente da construção do quadro.
3. No estado F ($k_o = 2$), duas decisões eram igualmente boas. Para evitar problemas não mantemos as duas decisões, escolhemos arbitrariamente uma delas.

x	u	$x\prime$	l	$\bar{J}$	$J = l + \bar{J}'$	$\bar{J}$	$\bar{u}$
$k_o = 4$ - $1 = 3$							
H	ir para J	J	7	0	7	7	ir para J
I*	ir para J	J	5	0	5	5	ir para J*
$k_o = 2$							
E	ir para H ir para I	H I	6 9	7 5	13 14	13	ir para H
F	ir para H ir para I	H I	10 12	7 5	17 17	17	ir para H
G*	ir para I	I	3	5	8	8	ir para I*

x	u	$x\prime$	l	$\bar{J}$	$J = l + \bar{J}'$	$\bar{J}$	$\bar{u}$
$k_o = 1$							
B	ir para E	E	8	13	21	21	ir para E
	ir para F	F	6	17	23		
C*	ir para E	E	4	13	17	15	ir para G*
	ir para F	F	8	17	25		
	ir para G	G	7	8	15		
D	ir para F	F	9	17	26	19	ir para G
	ir para G	G	11	8	19		
$k_o = 0$							
A*	ir para B	B	4	21	25	20**	ir para C*
	ir para C	C	5	15	20		
	ir para D	D	3	19	22		

Figura E.7: Algoritmo de programação dinâmica.

4. A cada etapa (k_o) do algoritmo, só nos utilizamos da primeira, sétima e oitava colunas do quadro da etapa anterior ($k_o + 1$). Após completar-se uma etapa, podemos apagar da segunda até a sexta coluna do quadro.

5. A grande vantagem da utilização da programação dinâmica é, entretanto, a economia de tempo: No processo de examinar todas as possibilidades teríamos de realizar 12 somas de 4 parcelas cada. Empregando-se o algoritmo acima, fizemos 15 somas de duas parcelas cada.

Estes valores podem não mostrar a enorme vantagem de programação dinâmica em relação ao processo de examinar todas as possibilidades, devido à extrema simplicidade do exemplo. Porém este exemplo foi utilizado aqui devido exatamente à sua simplicidade, assim facilitando o entendimento do algoritmo. A vantagem da programação dinâmica é mais notável quanto maior a complexidade do problema.