

**UNIVERSIDADE FEDERAL DE SANTA CATARINA
PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA
COMPUTAÇÃO**

Ismael Seidel

**ANÁLISE DO IMPACTO DE *PEL DECIMATION* NA CODIFICAÇÃO
DE VÍDEOS DE ALTA RESOLUÇÃO**

Florianópolis

2014

Ismael Seidel

**ANÁLISE DO IMPACTO DE *PEL DECIMATION* NA CODIFICAÇÃO
DE VÍDEOS DE ALTA RESOLUÇÃO**

Dissertação submetida ao Programa de Pós-
Graduação em Ciência da Computação para
a obtenção do Grau de Mestre em Ciência
da Computação.

Orientador: José Luís A. Güntzel, Prof. Dr.

Florianópolis

2014

Ficha de identificação da obra elaborada pelo autor,
através do Programa de Geração Automática da Biblioteca Universitária da UFSC.

Seidel, Ismael

Análise do Impacto de Pel Decimation na Codificação de Vídeos de Alta Resolução / Ismael Seidel ; orientador, José Luís Almada Güntzel - Florianópolis, SC, 2014.
153 p.

Dissertação (mestrado) - Universidade Federal de Santa Catarina, Centro Tecnológico. Programa de Pós-Graduação em Ciência da Computação.

Inclui referências

1. Ciência da Computação. 2. Codificação de Vídeo Digital. 3. Soma das Diferenças Absolutas. 4. Subamostragem. 5. Projeto VLSI. I. Almada Güntzel, José Luís. II. Universidade Federal de Santa Catarina. Programa de Pós-Graduação em Ciência da Computação. III. Título.

Ismael Seidel

**ANÁLISE DO IMPACTO DE *PEL DECIMATION* NA CODIFICAÇÃO
DE VÍDEOS DE ALTA RESOLUÇÃO**

Esta Dissertação foi julgada aprovada para a obtenção do Título de “Mestre em Ciência da Computação”, e aprovada em sua forma final pelo Programa de Pós-Graduação em Ciência da Computação.

Florianópolis, 17 de fevereiro 2014.

Ronaldo dos Santos Mello, Prof. Dr.
Coordenador

José Luís A. Güntzel, Prof. Dr.
Orientador

Banca Examinadora:

José Luís A. Güntzel, Prof. Dr.
Presidente

Djones Vinicius Lettnin, Prof. Dr.

Luciano Volcan Agostini, Prof. Dr.

Luiz C. V. dos Santos, Prof. Dr.

Aos meus pais, e à Rafaela.

AGRADECIMENTOS

Agradeço primeiramente a meus pais, Leonice e Edgar, que tanto fizeram por mim, mas ao mesmo tempo me deixaram aprender a caminhar. Volto a agradecer à Rafaela, minha namorada, pelo amor e compreensão nas horas que passavam depressa demais e aquelas que precisei dedicar ao trabalho de mestrado.

Agradeço ao prof. Güntzel pela excelente orientação, pelas horas de revisão em finais de semanas e feriados. Pela motivação para fazer este trabalho de mestrado e as publicações dele derivadas. Pelas ajudas com as viagens, que foram muitas nestes dois anos! Por proporcionar a participação no curso da Synopsys na UFRGS, e buscar recursos para conseguir apresentar alguns trabalhos.

Agradeço ao Bruno Moraes e ao Emílio Wuerges, pelas discussões sobre codificação de vídeo e a colaboração neste trabalho, principalmente no que se refere ao artigo do ICME! Agradeço também aos demais membros do Laboratório de Computação Embarcada (ECL), pelas conversas, bolinhos, pelo ambiente de trabalho. Dentre estes agradeço em especial aqueles que fazem parte do grupo de pesquisa de codificação de vídeo: André Beims Bräscher, Marcio Monteiro e Luiz Henrique Cancellier, pelas ajudas na revisão e também com as diversas figuras presentes nesta dissertação.

Agradeço também aos membros da banca pela participação e pelas observações, as quais enriqueceram este trabalho. Finalmente, agradeço à CAPES (Coordenação de Aperfeiçoamento de Pessoal de Nível Superior) pela bolsa de mestrado e também ao CNPq (Conselho Nacional de Desenvolvimento Científico e Tecnológico) pelos recursos fornecidos através do programa Brazil-IP e do INCT-Namitec.

Obrigado.

The idea of going to the movies made Hugo remember something Father had once told him about going to the movies when he was just a boy, when the movies were new. Hugo's father had stepped into a dark room, and on a white screen he had seen a rocket fly right into the eye of the man in the moon. Father said he had never experienced anything like it. It had been like seeing his dreams in the middle of the day.

Brian Selznick, *The Invention of Hugo Cabret*

Resumo

Ao mesmo tempo em que o número de pixels por quadro tende a aumentar pela iminente adoção de resoluções ultra altas, a subamostragem de pixels, também conhecida por *pel decimation*, surge como uma opção viável para aumentar a eficiência energética da codificação de vídeo. Este trabalho investiga os impactos em energia e qualidade, quando *pel decimation* é aplicado ao cálculo da Soma das Diferenças Absolutas (SAD), a qual é a métrica de similaridade mais utilizada durante a etapa de estimação de movimento. Primeiramente, apresenta-se uma análise de qualidade de 15 padrões de subamostragem. Os 10.860 pontos experimentais usados proporcionam evidência estatística de que a razão de amostragem 4:3 proposta apresenta velocidade de codificação duas vezes maior do que a amostragem completa, perdendo apenas 5% em DSSIM e 1% em PSNR. A razão 4:3 apresenta o melhor custo-benefício entre aceleração e redução de qualidade, comparando-se com razões de menor amostragem. Para obter estimativas de área em silício e energia por bloco, cinco arquiteturas para cálculo da SAD foram projetadas e sintetizadas para uma biblioteca *standard cell* industrial. Dentre elas, uma pode ser configurada para operar com razões de amostragem 1:1, 4:3, 2:1 ou 4:1, ao passo que as demais foram projetadas para operar exclusivamente com cada uma destas razões de amostragem. A arquitetura configurável, operando em amostragem completa, consome 3,54pJ/bloco (60% menos que a versão não-configurável), podendo ser reduzida até 1,34pJ/bloco utilizando-se a razão de amostragem 4:1, com redução de 2,8% em PSNR e 14,1% em DSSIM. Finalmente, demonstra-se que a aceleração de codificação de um determinado padrão de subamostragem deve-se à redução conjunta do número de pixels amostradas e do número total de cálculos de SAD. Assim, modelando-se as componentes de energia da codificação de vídeos, demonstra-se que a eficiência energética do processo de codificação como um todo pode ser melhorada além da razão de subamostragem. Utilizando-se uma arquitetura de SAD configurável, a economia de energia pode ser de até 95,11%.

Palavras-chave: codificação de vídeo, estimação de movimento, SAD, subamostragem, projeto VLSI

Abstract

As the number of pixels per frame tends to increase by the upcoming adoption of ultra high resolutions, pixel subsampling, also known as pel decimation, appears as a viable means to improve the energy efficiency of video coding. This work investigates the impacts on energy and quality when pel decimation is applied to the Sum of Absolute Differences (SAD) calculation, which is the most used similarity metric in motion estimation step of video coding. Firstly, a quality assessment of 15 pel decimation patterns is presented. The 10,680 experimental points used provide statistical evidence that the proposed 4:3 ratio leads to an encoding speedup of more than two times in comparison to full sampling, losing only 5% in DSSIM and 1% in PSNR. Compared with lower sampling ratios, it presents a better trade-off between speedup and quality loss. To obtain estimates for silicon area and energy per block, five SAD architectures were designed and synthesized for an industrial standard cell library. Among those, one can be configured to operate with 1:1, 4:3, 2:1 or 4:1 sampling ratios, whereas the rest are tailored to operate exclusively with each one of these ratios. The configurable architecture consumes 3.54pJ/block operating in full sampling (60% lower than the non-configurable). The energy can be further reduced until 1.34pJ/block by using 4:1 ratio, with losses of 2.8% in PSNR and 14.1% in DSSIM. Finally, it is shown that the speedup of a given subsampling pattern is due the reduction of both the number of sampled pixels and the total number of SAD calculations. Therefore, by modeling the video coding energy components, it is shown that the whole video compression energy efficiency can be increased beyond the sampling ratio. By using a configurable SAD architecture operating in 4:1 ratio the energy savings are up to 95.11%.

Keywords: video coding, motion estimation, SAD, pel decimation, VLSI design

Lista de Figuras

Figura 1	Velocidade de Transmissão de Banda-Larga em Alguns Países.	31
Figura 2	Número total de pixels por quadro, de acordo com a resolução.	33
Figura 3	Número de chamadas à SAD na JM.	34
Figura 4	Exemplos de padrões de subamostragem para blocos 4x4.	35
Figura 5	Exemplo de captura de imagem para o formato RGB.	41
Figura 6	Sensibilidade dos bastonetes e cones.	42
Figura 7	Exemplos de subamostragem croma.	44
Figura 8	Diagrama do H.264/AVC (simplificado).	45
Figura 9	Exemplo de estimação de movimento.	46
Figura 10	Vetores de movimento que apontam para fora do quadro.	47
Figura 11	ME com precisão de 1/4 de pixel.	47
Figura 12	Múltiplas Referências no H.264/AVC.	48
Figura 13	Predição intra de acordo com o H.264/AVC.	52
Figura 14	Distribuição do ruído com mesmo PSNR.	54
Figura 15	Padrões de amostragem avaliados.	60
Figura 16	Formação de padrão para blocos 8x8.	61
Figura 17	Resultados de PSNR para diferentes resoluções.	65
Figura 18	PSNR e velocidade de codificação para amostras 1080p.	66
Figura 19	Organização dos resultados em grafos.	70
Figura 20	Padrões de amostragem adotados por Huang et al. (2009).	77
Figura 21	<i>Datapath</i> projetado para SAD.	83
Figura 22	FSM da arquitetura configurável proposta.	84
Figura 23	Bloco de controle e bloco operativo.	85
Figura 24	FSM para as arquiteturas configurável e fixas.	87
Figura 25	<i>Datapath</i> para a arquitetura fixa com amostragem completa.	88
Figura 26	Fluxo de síntese e simulação.	90
Figura 27	Resultados relativos de área.	92
Figura 28	Resultados de energia/bloco e qualidade <i>versus</i> razão de sub-amostragem.	93
Figura 29	Representação da área de busca.	98
Figura 30	Histograma de valores de SAD.	102
Figura 31	Diagrama de Pareto das ocorrências de tamanho de buffer de	

candidatos.....	104
Figura 32 Compromisso entre energia e qualidade.....	111
Figura 33 Ganho energético total relativo.....	114
Figura 34 Resultados do espaço de síntese explorado da Soma das Diferenças Absolutas - <i>Sum of Absolute Differences</i> (SAD) configurável.....	141
Figura 35 Resultados de potência total (dinâmica e estática).....	142
Figura 36 Resultados de potência dinâmica e estática relativas à potência total (%).	142
Figura 37 Resultados de energia/bloco para frequência alvo.....	144
Figura 38 Resultados de energia/bloco para frequências máximas.....	145

Lista de Tabelas

Tabela 1	Valores de taxa de bits e espaço ocupado por vídeos conforme padrões usados em <i>Blu-rays</i> . A última linha foi incluída para exemplificar o uso de novas resoluções, como o Ultra HD (8K). Adaptado de: (BENNETT, 2011)	30
Tabela 2	Resoluções das amostras de vídeo e suas respectivas taxas de bit <i>lossless</i> (Mbps).	62
Tabela 3	Comparação de Índice de Dissimilaridade Estrutural - <i>Structural Dissimilarity Index</i> (DSSIM) e velocidade codificação para otimização em Índice de Similaridade Estrutural - <i>Structural Similarity Index</i> (SSIM)	68
Tabela 4	Comparação de Relação Sinal-Ruído de Pico - <i>Peak signal-to-noise ratio</i> (PSNR) e velocidade codificação para otimização em PSNR .	68
Tabela 5	Resumo dos trabalhos correlatos, organizados por data de publicação.	81
Tabela 6	Sinais da FSM por estado.	86
Tabela 7	Frequências máximas (GHz) e seus respectivos <i>throughputs</i> (<i>Mblocs/s</i>)	91
Tabela 8	Comparação dos resultados de síntese para frequência máxima	136
Tabela 9	Comparação dos resultados de síntese para frequência alvo ..	136
Tabela 10	Frequências (MHz), <i>throughput</i> (<i>Mbloco/s</i>) e degradação (%)	139
Tabela 11	Comparação com trabalhos correlatos para mesmo <i>throughput</i> , onde A apresenta os resultados de Walter, Diniz e Bampi (2011), B são os resultados aqui apresentados da síntese 130nm, C são os resultados de Walter e Bampi (2011) e D são os resultados aqui apresentados da síntese para 65nm.	146

Lista de Algoritmos

1	FBMA.....	99
2	Algoritmo de Eliminações Sucessivas.....	102
3	x264_ESA	103

Lista de Siglas e Abreviaturas

3 CCD <i>Three-CCD</i>	40
ADS <i>Absolute Difference of Sums</i>	100
ANOVA <i>Análise de Variância - Analysis of Variance</i>	57
APS <i>Alternating pel-subsampling block matching algorithm</i>	57
AVCHD <i>Advanced Video Codec High Definition</i>	30
B <i>Bipredito</i>	48
BDBR <i>Bjontegaard Delta Bitrate</i>	76
BDPSNR <i>Bjontegaard Delta PSNR</i>	76
BM <i>Block Matching</i>	98
CCD <i>Dispositivo de Carga-Acoplada - Charge-Coupled Device</i>	39
CLA <i>Carry-Lookahead Adder</i>	80
CMOS <i>Semicondutor Metal-Óxido Complementar - Complementary Metal-Oxide Semiconductor</i>	77
CRA <i>Carry-Ripple Adder</i>	79
CSA <i>Carry-Select Adder</i>	80
DAG <i>Grafo Acíclico Direcionado - Direct Acyclic Graph</i>	69
DC[®] <i>Synopsys[®] Design Compiler</i>	133
DCT <i>Transformada Discreta dos Cossenos - Discrete Cosine Transform</i> .	51
DS <i>Diamond Search</i>	116
DSP <i>Digital Signal Processing</i>	79
DSSIM <i>Índice de Dissimilaridade Estrutural - Structural Dissimilarity Index</i> 110	
DST <i>Transformada Discreta dos Senos - Discrete Sine Transform</i>	53
DVD <i>Disco de Vídeo Digital - Digital Video Disc</i> -	29
ESA <i>Algoritmo de Busca por Eliminações Sucessivas - Successive Elimination Exhaustive Search Algorithm</i>	63
FBMA <i>Fullsearch Block Matching Algorithm</i>	95
FBSME <i>Estimação de Movimento para Blocos de Tamanho Fixo - Fixed Block-Size Motion Estimation</i>	99
FFT <i>Transformada Rápida de Fourier - Fast Fourier Transform</i>	52
FME <i>Fractional Motion Estimation</i>	77

FPGA <i>Field-Programmable Gate Array</i>	77
fps Quadros por Segundo - <i>Frames per Second</i>	75
FR Taxa de Quadros - <i>Frame Rate</i>	29
FSM Máquina de Estados Finitos - <i>Finite State Machine</i>	107
GA Algoritmo Genético - <i>Genetic Algorithm</i>	58
GEA Algoritmo de Eliminação Global - <i>Global Elimination Algorithm</i> ..	49
HD-DVD <i>High Definition Digital Video Disc</i>	30
HD Alta Definição - <i>High Definition</i>	77
HEVC Codificação de Vídeo de Alta Eficiência - <i>High Efficiency Video Co- ding</i>	95
HVS Sistema Visual Humano - <i>Human Visual System</i>	58
IEC <i>International Electrotechnical Commission</i>	30
IFA <i>Internationale Funkausstellung Berlin</i>	33
IRS Busca Aleatória Iterativa - <i>Iterative Random Search</i>	49
ITU-T <i>International Telecommunication Union Telecommunication Standar- dization Sector</i>	41
JM <i>Joint Model</i>	74
JVT <i>Joint Video Team</i>	45
LH <i>Low-Vdd/High-Vt</i>	134
MAD <i>Mean Absolute Difference</i>	78
MB Macrobloco - <i>Macroblock</i>	58
MC Compensação de Movimento - <i>Motion Compensation</i>	34
ME Estimacão de Movimento - <i>Motion Estimation</i>	95
MPEG-2 <i>Moving Pictures Expert Group - 2</i>	76
MPEG <i>Moving Picture Experts Group</i>	78
MSE <i>Mean Squared Error</i>	78
MV Vetor de Movimento - <i>Motion Vector</i>	99
NN Nominal	139
NTSC <i>National Television System Committee</i>	29
P Predito	48
P2P Ponto-a-Ponto - <i>Peer-to-Peer</i>	32
PMD Dispositivo Móvel Portátil - <i>Portable Mobile Device</i>	93

PMV Vetor de Movimento Predito - <i>Predicted Motion Vector</i>	48
PSNR Relação Sinal-Ruído de Pico - <i>Peak signal-to-noise ratio</i>	96
QME <i>Quartet-pel motion estimation</i>	57
QSDS-DIC <i>Quarter Sub-sampled Diamond Search algorithm with Dynamic Iteration Control</i>	78
RBSAD <i>Reduced Bit Sum of Absolute Differences</i>	77
RD Taxa-Distorção - <i>Rate-Distortion</i>	64
RGB Vermelho, Verde e Azul - <i>Red, Green and Blue</i>	40
SAIF <i>Switching Activity Interchange Format</i>	89
SAD Soma das Diferenças Absolutas - <i>Sum of Absolute Differences</i>	95
SATD Soma das Diferenças Transformadas Absolutas - <i>Sum of Absolute Transformed Differences</i>	50
SBTVD Sistema Brasileiro de Televisão Digital	30
SEA <i>Successive Elimination Algorithm</i>	95
SL Camada Simples - <i>Single Layer</i>	30
SSD Soma das Diferenças Quadráticas - <i>Sum of Squared Differences</i>	50
SSIM Índice de Similaridade Estrutural - <i>Structural Similarity Index</i>	63
TSMC <i>Taiwan Semiconductor Manufacturing Company Limited</i>	133
TSS <i>Three-Step Search</i>	75
TU <i>Transform Unit</i>	53
UMH <i>Uneven MultiHexagon</i>	62
V1 Córtex Visual Primário	43
V2 Área Visual V2	43
VCS[®] <i>Synopsys[®] VCS</i>	89
VECG <i>Visual Coding Experts Group</i>	45
VBSME Estimação de Movimento para Blocos de Tamanho Variável - <i>Variable Block Size Motion Estimation</i>	99
VLSI <i>Very-large-scale Integration</i>	34
VoD Vídeo sob Demanda - <i>Video on Demand</i>	32
VPI <i>Verilog Procedural Interface</i>	89
Y'CbCr Y' (luma) Cb e Cr (" <i>blue-difference</i> " e " <i>red-difference</i> ") dos componentes de cor	41

Sumário

1 INTRODUÇÃO	29
1.1 JUSTIFICATIVA	32
1.2 OBJETIVOS	35
1.2.1 Objetivos Específicos	36
1.3 PRINCIPAIS CONTRIBUIÇÕES	36
1.4 ORGANIZAÇÃO DA DISSERTAÇÃO	37
2 VÍDEO DIGITAL	39
2.1 CAPTURA DA IMAGEM.....	39
2.1.1 Representação Digital da Imagem	40
2.2 SISTEMA VISUAL HUMANO	42
2.2.1 Interpretação da Imagem	43
2.3 SUBAMOSTRAGEM CROMA	44
2.4 REDUNDÂNCIAS INTERPIXEL	45
2.4.1 Estimação De Movimento	46
2.4.2 Algoritmos de Busca para Estimação de Movimento	49
2.4.3 Métricas de Similaridade	50
2.4.3.1 Métricas Objetivas	50
2.4.4 Predição Intra	52
2.4.5 Transformada	52
2.4.6 Quantização	53
2.5 MÉTRICAS DE QUALIDADE	54
2.5.1 Métricas Subjetivas	55
3 AVALIAÇÃO DA QUALIDADE	57
3.1 TRABALHOS CORRELATOS	57
3.2 PADRÕES DE SUBAMOSTRAGEM AVALIADOS	59
3.3 CONFIGURAÇÃO EXPERIMENTAL	61
3.4 RESULTADOS	63
3.5 CONCLUSÕES PARCIAIS	71
4 ARQUITETURAS DEDICADAS AO CÁLCULO DA SAD	73
4.1 TRABALHOS CORRELATOS	74
4.1.1 Arquiteturas para Estimação de Movimento - <i>Motion Estimation</i> (ME)	74
4.1.2 Arquiteturas Dedicadas ao Cálculo da SAD	78
4.1.3 Resumo dos Trabalhos Correlatos Apresentados	80
4.2 APRESENTAÇÃO DA ARQUITETURA PROPOSTA	82
4.3 SÍNTESE E SIMULAÇÃO COM VETORES REALISTAS	85
4.3.1 Arquiteturas Fixas para Comparação	86

4.3.2	Configuração Experimental e Fluxo de Síntese	89
4.3.3	Resultados e Análise	91
4.4	CONCLUSÕES	94
5	IMPACTO DE <i>PEL DECIMATION</i> NA EFICIÊNCIA ENERGÉTICA	95
5.1	CORRELAÇÃO ENTRE O NÚMERO DE SADS E A VELOCIDADE DE CODIFICAÇÃO	96
5.2	CORRELAÇÃO ENTRE O ALGORITMO DE BUSCA E O NÚMERO DE SADS	97
5.2.1	<i>Successive Elimination Algorithm (SEA)</i> no x264	103
5.3	MODELAGEM INICIAL DOS COMPONENTES DE ENERGIA	104
5.4	MODELAGEM DAS COMPONENTES DE ENERGIA CONSIDERANDO VBSME	105
5.5	RELAÇÃO DE ENERGIA ENTRE DUAS RAZÕES DE AMOSTRAGEM	106
5.6	RELAÇÃO DE ENERGIA DA ARQUITETURA CONFIGURÁVEL PROPOSTA DE ACORDO COM PADRÕES DE AMOSTRAGEM	108
5.7	RELAÇÃO DE COMPROMISSO ENTRE QUALIDADE E ENERGIA	110
5.8	MODELAGEM DAS COMPONENTES DE ENERGIA DE UM CODIFICADOR DE VÍDEO	110
5.8.1	Impacto da Arquitetura Configurável no Fluxo de Codificação	112
6	CONCLUSÕES	115
6.1	TRABALHOS FUTUROS	116
	Referências Bibliográficas	119
	Apêndice A – COMPARAÇÃO COM ARQUITETURA NÃO CONFIGURÁVEL DE MESMO <i>DATAPATH</i>	133
	Apêndice B – EXPLORAÇÃO DO ESPAÇO DE SÍNTESE	139
	Apêndice C – Lista de Publicações	151

1 INTRODUÇÃO

Desde o princípio do cinema como forma de arte e como tecnologia, nos anos 1890s¹, a ideia de imagem dinâmica atrai a atenção de empreendedores, artistas, cientistas e políticos (Nowell-Smith, 1999). Não só o cinema exerce essa atração, mas também o vídeo nos mais diversos tipos de mídia e formas de transmissão (TV, Disco de Vídeo Digital - *Digital Video Disc* - (DVD), *Blu-Ray*, Celulares, Internet, etc). Primeiramente, as câmeras eram analógicas com vários padrões definindo seus formatos de gravação. Posteriormente, surge o vídeo digital, facilitando a manipulação dos dados (por exemplo, na edição e transmissão) (MANZATO, 2004).

Um vídeo nada mais é do que uma sequência de imagens rapidamente apresentadas, gerando assim a ilusão de movimento². Segundo Richardson (2002), representar um vídeo em forma digital requer uma grande quantidade de dados (*bits*). Tal fato pode ser facilmente verificado como segue: A chamada Taxa de Quadros - *Frame Rate* (FR), também conhecida como cadência, é o número de quadros (*frames*) exibidos por segundo de vídeo. Levando-se a FR em consideração, admitindo que um pixel em uma imagem colorida ocupe ‘N’ bits, e tomando por base uma imagem com ‘W’ pixels horizontais e ‘H’ pixels verticais, a taxa de bit (*bitrate*), medida em Quadros por Segundo - *Frames per Second* (fps), é dada por:

$$bitrate = N \times W \times H \times FR \quad (1.1)$$

Através da Equação 1.1 é possível calcular o espaço necessário para armazenar um vídeo de duração ‘t’ em sua forma bruta (do inglês “raw”), através da seguinte Equação:

$$espaço_{raw} = bitrate \times t \quad (1.2)$$

Conforme a Equação 1.1, para um filme em DVD com $W \times H = 720 \times 480$ ³,

¹Nesta época muitas pessoas acreditavam que as imagens que estavam assistindo fossem reais. Em 1896, de acordo com um comentarista, o filme *Rough Sea at Dover* : “presented the breaking waves on the seashore. Wave after wave came tumbling on the sand, and as they struck, broke into tiny floods just like the real thing. Some of the people in the front row seemed to be afraid they were going to get wet, and looked about to see where they could run to, in case the waves came too close”. (MATHEWS; MUSSER; ART, 2005)

²Deve-se à fisiologia humana, a capacidade de interpretação dessa sequência de imagens estáticas como movimento. Mais detalhes serão fornecidos no Capítulo 2.

³Considerando linhas visíveis para o padrão *National Television System Committee* (NTSC), embora este padrão tenha no total 525 linhas. Há uma certa variação para os tamanhos. (TAYLOR, 2011)

$N = 24$ bits/pixel, $FR = 29,97^4$ fps, são necessários $24 \times 720 \times 480 \times 29,97 = 248.583.168$ bits/s $= 29,63$ MiB⁵/s. Mesmo não parecendo um valor muito alto, ao se considerar um filme de 90 minutos como exemplo e aplicando-se a Equação 1.2, temos $29,63$ MiB/s $\times 90min \times 60s = 160.002MiB = 156,25GiB$. Sendo a capacidade de um DVD Camada Simples - *Single Layer* (SL) de 4,7 Gb (TAYLOR; JOHNSON; CRAWFORD, 2006), percebe-se logo que há algum bom algoritmo de compressão que torna possível o uso de um DVD com essa capacidade. Além disso, deve-se considerar que também serão armazenados no DVD as faixas de áudio para cada idioma, legendas, menus, extras, etc. Ao se analisar, por exemplo, um vídeo em Alta Definição - *High Definition* (HD) e *Full HD*⁶, como mostrado na Tabela 1, nota-se claramente a necessidade de uma grande compressão. Afinal, poucos usuários poderiam armazenar um filme com, por exemplo, 750 GiB.

Tabela 1: Valores de taxa de bits e espaço ocupado por vídeos conforme padrões usados em *Blu-rays*. A última linha foi incluída para exemplificar o uso de novas resoluções, como o Ultra HD (8K). Adaptado de: (BENNETT, 2011)

Tipo	Tamanho do Quadro (pixels)	Taxa de Quadros	de <i>Bitrate</i>	Espaço _{raw} (90 min)
HD 720 24p	1280x720	24/s	0,53 Gbps	333,71 GiB
HD 720 50p	1280x720	50/s	1,11 Gbps	695,23 GiB
HD 1080 24p	1920x1080	24/s	1,19 Gbps	750,85 GiB
8K UHD 24p	7680x4320	24/s	17,8 Gbps	11,73 TiB

No caso do DVD, o padrão *Moving Pictures Expert Group - 2* (MPEG-2) é usado para a codificação do vídeo. Já para as necessidades atuais, como o *Blu-ray*, *High Definition Digital Video Disc* (HD-DVD), *Advanced Video Codec High Definition* (AVCHD), Sistema Brasileiro de Televisão Digital (SBTVD), foi adotado o padrão H.264⁷/AVC (ITU-T, 2009), o qual melhora

⁴Tal FR está correlacionado ao padrão de codificação analógico NTSC, o qual, ao introduzir a codificação de crominância, precisou baixar o FR de 30 fps para 29,97 fps para que não houvesse interferência na transmissão de áudio.

⁵Lê-se *mebibyte*. A o prefixo binário mebi corresponde à 2²⁰ (IEC, 2008). Tal unidade foi estabelecida pela *International Electrotechnical Commission* (IEC) para evitar a confusão com a base decimal “mega” (10⁶), definida no SI.

⁶A chamada resolução HD corresponde à resolução de 1280x720 pixels. Já *Full HD* corresponde à resolução de 1920x1080 pixels.

⁷O padrão H.264 é uma recomendação da *International Telecommunication Union Telecommunication Standardization Sector* (ITU-T). Esta adota, para padrões de codificação de vídeo, a letra H seguida da numeração da recomendação. O mesmo vale para o H.265 (HEVC).

a compressão em até 50% em relação ao MPEG-2.

Mesmo com o advento da computação em nuvem, a codificação de vídeo não pode migrar dos dispositivos portáteis para a nuvem, uma vez que não haveria banda suficiente disponível para enviar o vídeo sem nenhuma compressão. Além disso, segundo Wien (2013), os dados para representação de vídeo crescem mais do que as capacidades de transmissão de rede. Conforme mostrado na Tabela 1, para um vídeo *Full HD* e 24 fps a banda necessária seria de 1,19 Gbps. Tal taxa de transmissão excede a grande maioria das redes do Brasil. Ao observar a Figura 1 é possível ter uma ideia do estado atual de velocidade de transmissão das redes de Internet. Tomando o Brasil como exemplo, pode-se notar que apenas um pouco mais de 20% de todas as assinaturas de banda larga do país ultrapassam 10 Mbps, sendo que mais de 50% das restantes estão entre 256 Kbps e 2 Mbps (ITU/ICT, 2013b apud ITU/ICT, 2013a).

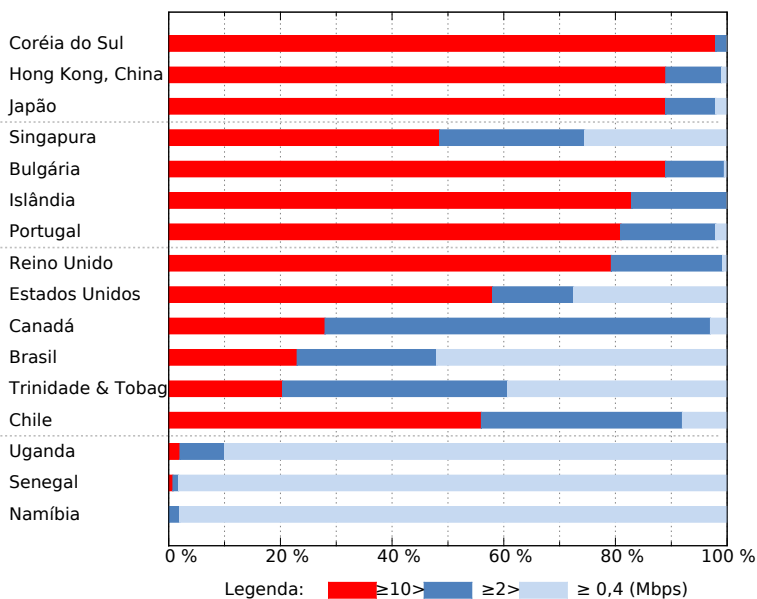


Figura 1: Dados publicados pela *ITU World Telecommunication* no primeiro trimestre de 2013 sobre velocidade de transmissão das redes de Internet Banda-Larga em alguns países. Adaptado de (ITU/ICT, 2013b apud ITU/ICT, 2013a, p. 5)

A empresa Cisco (2013) publicou sua previsão de que, em 2017, a transmissão de todo o tipo de vídeo⁸ represente aproximadamente 80 a 90% do tráfego global de dados. Considerando apenas o tráfego de dados relacionados aos Dispositivos Móveis Portáteis - *Portable Mobile Devices* (PMDs), o mesmo estudo prediz que haverá um crescimento de 13 vezes o atual até 2017, devendo alcançar a marca de 11,2 Exabytes⁹ de dados por mês.

Até o início desta dissertação, o padrão estado da arte era o H.264/AVC. Entretanto, no início de 2013 foi publicado oficialmente o padrão de Codificação de Vídeo de Alta Eficiência - *High Efficiency Video Coding* (HEVC) (UGUR et al., 2010; SULLIVAN et al., 2012; ITU-T, 2013), o qual visa melhorar a compressão em até 50% em relação ao H.264/AVC. Para atingir sua meta, o HEVC, aumenta ainda mais a complexidade do processamento exigido em relação ao H.264/AVC. Ainda assim, é importante perceber que o próprio H.264/AVC, com todo o seu conjunto de ferramentas propostas, é inviável para aplicações com restrições no consumo de energia, devido, principalmente ao alto custo de processamento.

1.1 JUSTIFICATIVA

A codificação de vídeo digital é uma tarefa computacionalmente intensiva que requer um alto desempenho. Particularmente, para aplicações com restrições de tempo real como por exemplo transmissões ao vivo, tal requisito é ainda mais severo. O baixo consumo energético aparece como outro importante requisito, uma vez que o uso de compressão de vídeo tem crescido em aparelhos alimentados por baterias, como câmeras, *smartphones*, etc.

Os dados do site de compartilhamento de vídeos “YouTube” corroboram a larga produção de vídeo: são mais de 100 horas de vídeos carregados em seus servidores a cada minuto (YOUTUBE, 2013). As possíveis fontes destes vídeos são: *webcams*, computação gráfica, captura da área de trabalho e PMDs. Ainda de acordo com dados do “YouTube”, mais de 25% de seu tempo global de visualização advém de dispositivos móveis (o que equivale a mais de um bilhão de visualizações diárias) (YOUTUBE, 2013).

Além do processamento de banda base (ex: 3G, Wi-Fi), o qual é responsável pelo maior consumo energético em modelos de *smartphones* recentes (PATHAK; HU; ZHANG, 2012), a navegação na internet e aplicações multimídia correspondem a outra parte significativa do consumo energético (FALAKI et al., 2010; SHEPARD et al., 2011), uma vez que realizam diver-

⁸Consideram-se aqui TV, Vídeo sob Demanda - *Video on Demand* (VoD), Internet, e Ponto-a-Ponto - *Peer-to-Peer* (P2P).

⁹Um Exabyte corresponde a 10^{18} bytes.

sas operações de codificação/decodificação de imagens e vídeos. No caso de aparelhos específicos para captura de imagem (câmeras e filmadoras), o impacto de codificação de vídeo é obviamente maior. Ainda assim, em ambos os tipos de dispositivos, genéricos ou específicos, qualquer Joule economizado contribui para prolongar o tempo de vida da bateria.

Outro aspecto a ser observado é o crescimento das resoluções. Atualmente, a grande maioria dos dispositivos móveis são capazes de codificar vídeos Full HD, mas já estão sendo demonstrados produtos capazes de codificar vídeos em Ultra HD (4K), que é o caso do *smartphone* Acer Liquid S2 (ACER, 2013), apresentado durante a *Internationale Funkausstellung Berlin* (IFA) em setembro de 2013. Deve-se esperar para os próximos anos dispositivos móveis com capacidade de codificar/decodificar 2K, 4K e até mesmo 8K. A Figura 2 demonstra o crescimento do número total de pixels por quadro, de acordo com a resolução, dando uma ideia da grande massa de dados que devem ser processados de forma eficiente durante a codificação.

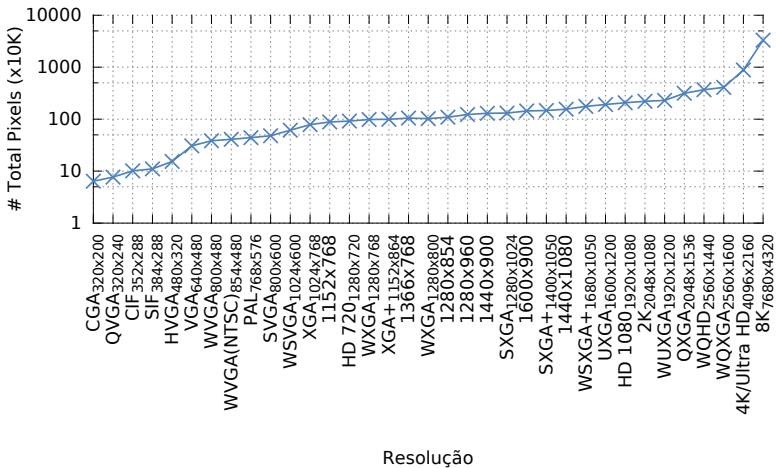


Figura 2: Número total de pixels por quadro, de acordo com a resolução.

Para simplificar a compressão, cada quadro de uma seqüência de vídeo é dividido em blocos menores. Os blocos constituem as matrizes básicas de pixels que são processadas para redução de redundâncias. Durante o processo de codificação de vídeo a Estimaco de Movimento - *Motion Estimation* (ME) corresponde à tarefa computacionalmente mais intensiva: para cada bloco do quadro original, a ME busca um bloco candidato com a maior similaridade para ser usado como referncia quando o bloco original é reconstruído du-

rante a etapa de Compensação de Movimento - *Motion Compensation* (MC). Para a reconstrução do bloco original, é usado o Vetor de Movimento - *Motion Vector* (MV), o qual aponta para essa referência, e o resíduo entre o bloco referência e o bloco original, i.e., as diferenças entre os valores de pixel de tais blocos. Para encontrar o candidato com maior similaridade, utiliza-se uma métrica de similaridade.

Dentre elas, a chamada Soma das Diferenças Absolutas - *Sum of Absolute Differences* (SAD) (RICHARDSON, 2003) é a mais utilizada porque depende apenas de operações básicas (adição, subtração e módulo). Esta simplicidade torna-a muito apropriada para implementações de codificadores em *hardware* (*Very-large-scale Integration* (VLSI)). A fim de se quantificar o número de cálculos de SAD necessários para codificar um vídeo, avaliou-se o *software* de referência do H.264/AVC, o *Joint Model* (JM) (JVT, 2011) codificando 50 quadros da sequência “Foreman” (XIPH.ORG, 2011), para quatro diferentes algoritmos de busca (*SUMHexagon*, *Fast Full*, *Hexagon* e *EPZS*), usando um dentre 29 tamanhos de janelas de busca (de 16 até 128, com passo de 4 unidades). Os resultados obtidos são apresentados na Figura 3.

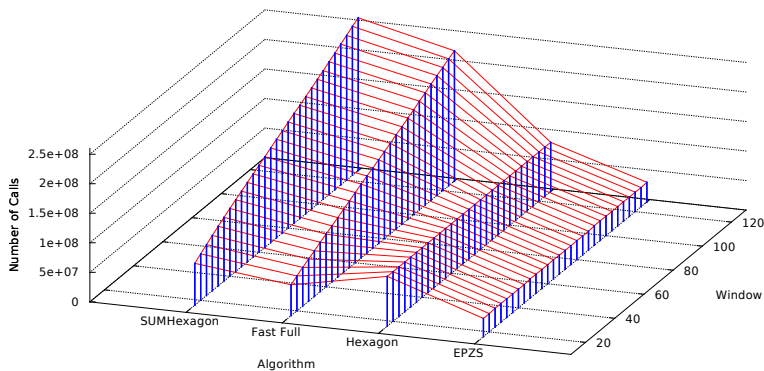


Figura 3: Número total de chamadas ao cálculo de SAD no *software* de referência do padrão H.264/AVC, o JM (JVT, 2011). Resultados obtidos através de análise dinâmica (*profiling*) para diferentes algoritmos e tamanhos da janela de busca, codificando 50 quadros da sequência “Foreman”(XIPH.ORG, 2011), 176x144, 30fps.

O total de chamadas de SAD para codificar a sequência escolhida varia de 35 milhões até 250 milhões, correspondendo respectivamente a 33% e 65% do tempo total de codificação. Porém, o número de operações com cálculo de SAD necessárias para compressão de vídeo aumentará drasticamente

assim que formatos de alta e ultra alta resolução se tornarem amplamente utilizados. Por um lado, o fato dos PMDs serem alimentados por baterias impõe-lhes grandes limitações quanto ao consumo de energia. Por outro lado, a adoção de formatos com maior resolução resulta também em aumento nas redundâncias entre pixels. A exploração apropriada de tais redundâncias extras requer não apenas a adoção de algoritmos de busca altamente efetivos na ME, mas também demanda o uso estratégico da subamostragem de pixels.

À subamostragem regular de pixels em uma janela de busca dá-se o nome de *pel decimation* (KUHNS, 1999). Ao aplicá-la ao cálculo da métrica de similaridade melhora-se a velocidade do codificador. Por outro lado, a correlação entre os candidatos diminui, resultando possivelmente em decréscimo de qualidade na predição. Ainda, a regularidade dos padrões de subamostragem, como pode ser observado na Figura 4, permite implementações VLSI de baixo custo e alta velocidade, o que pode refletir em significativa economia de energia. Os resultados apresentados na Figura 3 corroboram com a importância de aplicar métricas de similaridade simples e, mais importante, eles concedem uma estimativa da velocidade máxima e otimização de energia que podem ser alcançados quando os pixels são subamostrados para cálculo da métrica de similaridade.

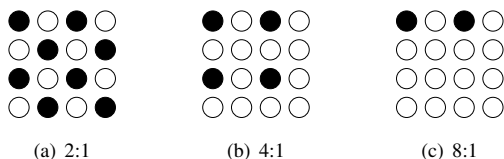


Figura 4: Exemplos de padrões de subamostragem de pixels em blocos de 4x4 pixels. Os círculos pretos indicam os pixels amostrados e que serão considerados no cálculo da métrica de similaridade adotada. As razões de amostragem, 2:1, 4:1 e 8:1, são as mais comumente usadas. Adaptado de (PORTO et al., 2007).

1.2 OBJETIVOS

O consumo energético de uma arquitetura de *hardware* para efetuar o cálculo da SAD depende de diversas decisões de projeto, podendo se beneficiar localmente do uso de *pel decimation*. Também pode existir algum impacto no consumo energético de acordo com o algoritmo de busca da ME com *pel decimation*. Assim, pode ser possível relacionar o uso de *pel deci-*

mation tanto no contexto local da métrica, quanto no contexto global do fluxo de dados do codificador (ME).

1.2.1 Objetivos Específicos

- O-1. **Propor e avaliar padrões alternativos de subamostragem (i.e., não encontrados na literatura) para *pel decimation***, visando minimizar as perdas de qualidade;
- O-2. **Projetar e sintetizar arquiteturas VLSI configuráveis de baixo consumo para cálculo da SAD**, com a finalidade de fornecer estimativas precisas de desempenho e energia: a configurabilidade visa permitir o uso de *pel decimation* com subamostragens 4:3, 2:1 e 4:1, bem como operar sem subamostragem;
- O-3. **Obter estimativas de quanto o uso de *pel decimation* pode minimizar o consumo de energia e aumentar a velocidade de codificação:** i.e., aumento da eficiência energética na codificação de vídeos de alta resolução no contexto do fluxo completo de codificação.

1.3 PRINCIPAIS CONTRIBUIÇÕES

- C-1. Proposta de novos padrões de subamostragem para redução das perdas de qualidade:
 - (a) Análise estatística de qualidade de codificação;
 - (b) Avaliação do aumento de velocidade de codificação;
 - (c) Demonstração do comportamento de *pel decimation* para altas resoluções.
- C-2. Projeto e síntese de arquitetura para cálculo da SAD utilizando *pel decimation*:
 - (a) Arquitetura configurável de alta eficiência energética para subamostragem (*pel decimation*) 1:1, 4:3, 2:1 e 4:1;
 - (b) Análise de atraso, potência e eficiência energética por razão de subamostragem, através de simulação com vetores reais, usando a arquitetura configurável com subamostragens 1:1, 4:3, 2:1 e 4:1;
 - (c) Comparação entre a arquitetura configurável proposta e arquiteturas não configuráveis de alta eficiência energética, em conformidade com trabalhos correlatos;

- (d) Análise do uso de técnicas de projeto *low-power* e frequência alvo, através da arquitetura configurável e considerando subamostragens 1:1, 2:1 e 4:1;
 - (e) Análise comparativa de impacto de *pel decimation versus* mudança de nodo tecnológico, através da arquitetura configurável e considerando subamostragens 1:1, 2:1 e 4:1.
- C-3. Identificação do impacto de *pel decimation* no fluxo de processamento do *x264* (MERRITT et al., 2004);
- C-4. Modelo do impacto de *pel decimation* (C-3) no fluxo de codificadores estado da arte (H.264/AVC (ITU-T, 2009), H.265/HEVC (ITU-T, 2013));
- C-5. Estimativa da redução (média) de energia e da aceleração (média) para cada padrão de *pel decimation* (de C-1), quando se utiliza alguma das versões de arquitetura VLSI configurável (C-2).

1.4 ORGANIZAÇÃO DA DISSERTAÇÃO

Esta dissertação está organizada como segue. No Capítulo 2 são explicados diversos conceitos sobre vídeo digital. Já o Capítulo 3 traz primeiramente a revisão do estado da arte sobre os aspectos de avaliação de qualidade para padrões de *pel decimation*. Também mostra os padrões de subamostragem avaliados, os experimentos conduzidos para suas avaliações e os resultados. Neste capítulo são apresentadas as contribuições do item C-1.

O Capítulo 4 inicia com a revisão do estado da arte sobre arquiteturas de ME e arquiteturas dedicadas para o cálculo da SAD, passando então para as arquiteturas implementadas: a arquitetura configurável para *pel decimation* (1:1, 4:3, 2:1 e 4:1), e alternativas não configuráveis. Os resultados das sínteses são apresentados, e então é descrita uma primeira análise sobre os ganhos em termos de eficiência energética e as perdas de qualidade (já demonstradas no Capítulo 3). Neste Capítulo apresenta as Contribuições 2a, 2b e 2c.

No Capítulo 5, os resultados do Capítulo 3 e Capítulo 4 são analisados em conjunto, mostrando a correlação entre o aumento de velocidade proporcionado pelos padrões e os respectivos ganhos energéticos. Vê-se que, diferentemente do esperado, os padrões demonstram melhor eficiência energética do que apenas o proveniente da razão de subamostragem. Com os dados analisados em conjunto, é feita a análise dos reais ganhos em eficiência energética *versus* perdas de qualidade. Este capítulo apresenta as Contribuições 3, 4 e 5, sendo o ápice da dissertação.

O Capítulo 6 apresenta as conclusões desta dissertação e também lista os possíveis trabalhos futuros. Finalmente, os Apêndices A e B apresentam, respectivamente as Contribuições 2d e 2e.

2 VÍDEO DIGITAL

Um vídeo nada mais é do que uma sequência de amostras espaciais e temporais capturadas eletricamente por uma câmera. Estas amostras são organizadas e armazenadas em matrizes de tamanho definido, sendo que cada elemento de uma matriz corresponde a uma informação específica da amostra. Estas matrizes, arranjadas em sequência, *a priori* na mesma ordem em que foram capturadas, formam o vídeo. Cada uma dessas matrizes pode ser entendida como uma imagem. Assim, um vídeo é uma sequência de imagens amostradas rapidamente¹, o que cria a ilusão de movimento.

A compressão acontece ao se eliminar informações redundantes dessa sequência de matrizes. De acordo com Shi e Sun (1999), tais redundâncias podem ser classificadas em:

1. Psicovisuais: originadas em características do Sistema Visual Humano - *Human Visual System* (HVS).
2. Estatísticas: relativas às repetições de símbolos no vídeo, podendo ainda ser dividida como
 - (a) Interpixel:
 - i. Intra quadro: correlação entre pixels de um mesmo quadro;
 - ii. Inter quadro: correlação entre pixels de diferentes quadros;
 - (b) Codificação: ou seja, a entropia na representação dos pixels.

Neste capítulo serão apresentados diversos conceitos relacionados à codificação de vídeo digital. Após uma introdução sobre o processo de captura da imagem, algumas características do HVS serão apresentadas em conjunto com ferramentas para redução de redundâncias Psicovisuais. Já as ferramentas para redução das redundâncias interpixel presentes em um vídeo serão apresentadas seguindo o fluxo de um codificador de vídeo digital. As redundâncias de codificação (entrópicas) não serão revistas nesta dissertação, uma vez que não há nenhuma exploração direta deste tipo de redundância.

2.1 CAPTURA DA IMAGEM

A imagem é capturada pela câmera usando-se, por exemplo, um Dispositivo de Carga-Acoplada - *Charge-Coupled Device* (CCD), o qual armazena cargas dos fótons capturados durante um período de tempo (exposição).

¹Segundo Vijayakumar (2008), pelo menos 24 quadros por segundo.

O CCD é um componente eletrônico feito de material semicondutor, como o Silício, e é manufaturado para ser sensível à incidência da luz sobre sua superfície (JUNIOR; KANAAN; GOMES, 2002). Ele é composto por uma matriz, onde cada elemento é responsável por capturar a informação de um pixel da imagem final. Quanto maior o tamanho da matriz, maior é o tamanho da imagem, e quanto menor o tamanho de cada elemento sensor (cada um dos quais correspondente à um pixel), maior a resolução da imagem. Em suma, o CCD tem o seguinte funcionamento: quando um fóton incide sobre um dos elementos da matriz este elemento libera um elétron. Cada elemento tem um capacitor que acumula a carga gerada pela incidência de luz. Ao final do tempo de exposição, a carga acumulada em cada elemento é medida, fornecendo assim a informação referente à luz que incidiu sobre cada elemento.

Costuma-se usar sobre o CCD um filtro de Bayer, comumente GRGB, permitindo a passagem de duas componentes verdes para uma componente vermelha e uma componente azul. Segundo (BROWN, 2004), apesar de refletir a maior sensibilidade do olho humano ao verde, o uso do filtro de Bayer causa diminuição da resolução, uma vez que cada pixel do filtro deixa passar luz de apenas um dos canais Vermelho, Verde e Azul - *Red, Green and Blue* (RGB). Uma alternativa ao uso do filtro de Bayer é o uso de *Three-CCDs* (3 CCDs), um CCD por canal (WOOTTON, 2005). Diversos modelos de câmeras profissionais possuem um CCD para cada componente. Outros tipos de sensores existem, como o que utiliza a tecnologia Semicondutor Metal-Óxido Complementar - *Complementary Metal-Oxide Semiconductor* (CMOS) para captura da imagem (SOUZA; CARDOZA, 2012).

2.1.1 Representação Digital da Imagem

Cada um dos canais é separado em uma matriz própria como resumido na Figura 5, sendo que cada um dos pixels de cada um dentre os três canais pode, por exemplo, ser representado digitalmente por 1 *byte*, totalizando assim 24 bits por ponto RGB. Existem outras formas de representação digital de imagens coloridas, com menos ou mais bits por canal, entretanto o mais conhecido e utilizado é o RGB com 24 bits.

Após a captura RGB, outro passo no processo de codificação de imagens/vídeos digitais é a conversão do espaço de cor. O padrão RGB combina as 3 cores para formar outras cores, semelhantemente ao modo como o HVS percebe cor. Adotando RGB com 24 bits, há 16.777.216 diferentes cores que podem ser representadas, muitas não levando em consideração as redundâncias na percepção humana. Outros espaços de cor podem ser mais adequados para a representação da imagem, permitindo alterações *a posteriori* nos dados

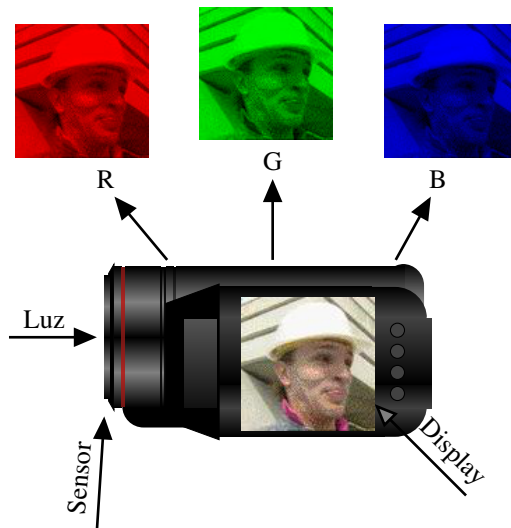


Figura 5: Exemplo de captura de imagem para o formato RGB.

que diminuem as perdas de qualidade percebidas pelo HVS.

Atualmente, o espaço de cor mais utilizado durante o processo de codificação é o $Y'CbCr$ (“*blue-difference*” e “*red-difference*”) dos componentes de cor ($Y'CbCr$). Mas há diversas formas de codificar $Y'CbCr$, de acordo com recomendações da *International Telecommunication Union Telecommunication Standardization Sector* (ITU-T). Considerando imagem em resolução inferior à Alta Definição - *High Definition* (HD), segue-se a recomendação BT.601 (ITU-T, 2011). No caso de representação de alta definição, adota-se a recomendação BT.709 (ITU-T, 2008). Atualmente, para os formatos de ultra alta definição, a nova recomendação a ser adotada é a BT.2020 (ITU-T, 2012b).

No formato $Y'CbCr$, o canal Y' representa a luminosidade, ou seja, a intensidade de luz no ponto. O canal Cb representa a diferença da luminosidade com o canal azul do RGB e o canal Cr representa a diferença da luminosidade com o canal vermelho. Este espaço de cor é adotado, por exemplo, no padrão H.264/AVC, por conter menos correlação entre suas componentes e possibilitar a operação denominada subamostragem cromática. Tal operação trabalha na remoção das redundâncias psicovisuais, mencionadas no início deste capítulo. Assim, a próxima seção apresenta conceitos sobre o HVS, antes abordar o processo de codificação de vídeo digital.

2.2 SISTEMA VISUAL HUMANO

Segundo VanWinkle et al. (2002), a visão é responsável por cerca de 83% do aprendizado. No olho humano, a retina é uma região coberta por um vasto conjunto de fotorreceptores: os cones (responsáveis pela percepção de cor) e os bastonetes. Segundo Kandel et al. (2012), enquanto os bastonetes são mais usados na visão sob baixa luminosidade, chamada escotópica, os cones são usados na visão sob condições de média/alta luminosidade, chamada visão fotópica. Isso ocorre pois os bastonetes possuem mais pigmento que os cones, embora possuam apenas um tipo de pigmento fotossensível, enquanto que nos cones há três tipos de pigmentos. Assim, há 3 tipos de cones: os que respondem mais ao amarelo (cor com comprimento de onda longo (L)), os mais sensíveis ao verde (comprimento de onda médio (M)) e aqueles que respondem mais ao violeta (comprimento de onda curto, S, do inglês “short”). Estes três tem seu pico de resposta (sensibilidade) para os comprimentos de onda de aproximadamente 570 nm, 540 nm e 440 nm, respectivamente (HUNT; POINTER, 2011). A Figura 6 demonstra a sensibilidade dos cones e bastonetes de acordo com o comprimento de onda da luz incidente na retina.

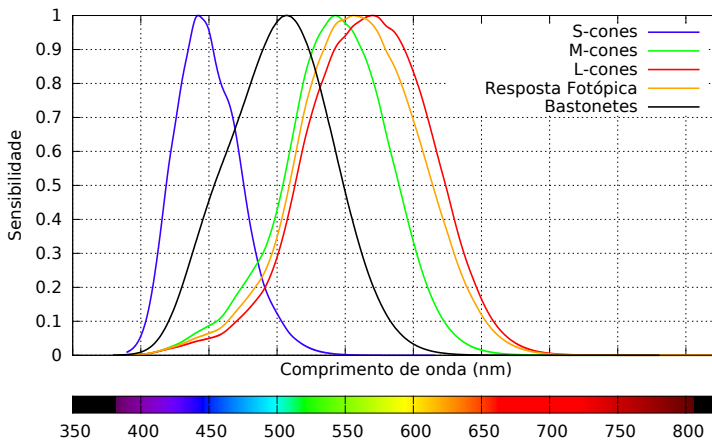


Figura 6: Sensibilidade dos bastonetes e dos três tipos de cones presentes no olho humano, em função do comprimento de onda da luz incidente na retina. Os dados para os gráficos foram obtidos como segue. Cones: (STOCKMAN; SHARPE, 2000 apud CVRL, 2013a). Bastonetes: (WYSZECKI; STILES, 1982 apud CVRL, 2013b). Resposta Fotópica: (SHARPE et al., 2005 apud CVRL, 2013b).

Perceba que a resposta fotópica situa-se na região de comprimentos de onda correspondente ao verde. Tal fato demonstra a grande sensibilidade do olho humano à cor verde, e por isso tal cor é relacionada ao componente de luminância da cena. Já a maior sensibilidade do HVS à luminosidade da cena deve-se à maior quantidade de pigmento dos bastonetes e também seu grande número: de acordo com Hecht (1982), há cerca de 120 milhões de bastonetes, enquanto há apenas cerca de 6 milhões de cones na retina humana.

2.2.1 Interpretação da Imagem

O nervo óptico transmite ao lobo occipital do cérebro as informações de luminosidade captadas através dos fotorreceptores. De acordo com Jonas et al. (1992), cada nervo óptico possui entre 770 mil e 1,7 milhões de fibras nervosas, o que, segundo Kandel et al. (2012, p. 585), corresponde a apenas 1% do total de fotorreceptores. O Córtex Visual Primário (V1), situado no lobo occipital, utiliza as informações recebidas através do nervo óptico para fazer o chamado processamento visual de baixo nível, o qual corresponde à identificação das formas dos objetos (KANDEL et al., 2012, p. 587), primeiramente identificando as bordas² dos objetos e suas orientações. Segundo Kandel et al. (2012), o V1 é também sensível à informação de disparidade e providencia as bases para a percepção de profundidade, embora tal percepção ocorra em uma etapa posterior.

Já os neurônios da Área Visual V2 (V2), também conhecida por córtex prestriate (GAZZANIGA; IVRY; MANGUN, 2002), demonstram realmente sensibilidade a informações de disparidade, tanto binocular como monocular (por exemplo: tamanho, perspectiva, oclusão, brilho e movimento). Tanto V1 como V2 sinalizam relações de plano de fundo e primeiro plano. Tais tarefas, quais sejam, a interpretação de objetos através de contornos e superfícies que lhes pertencem e a separação dos mesmos do plano de fundo, são, conforme Kandel et al. (2012, p. 619), o trabalho mais desafiante que o HVS deve executar. Este processamento das informações visuais recebe o nome de processamento visual de nível intermediário.

O chamado processamento visual de alto nível é o que trata da identificação de objetos. Este integra informações de diversas fontes e é o último

²De acordo com Kandel et al. (2012, p. 587), as células ganglionares presentes na retina já são responsáveis por uma pré-composição da imagem, uma vez que devem selecionar informações obtidas pelos fotorreceptores (que, como visto, estão presentes em número muito superior do que as conexões do nervo óptico). Tipicamente, a saída produzida por um grupo de células ganglionares da retina melhora as regiões da imagem com grande contraste e dá menor ênfase a regiões de luminosidade homogênea. Dessa forma, o processamento visual de baixo nível já nos torna mais sensíveis à percepção de bordas.

estágio no fluxo de informações visuais, o qual leva a experiência visual consciente. A codificação de informações visuais no córtex temporal inferior vem sendo estudada desde o trabalho de Gross, Bender e Rocha-Miranda (1969) e mesmo que muito já tenha sido descoberto sobre o processamento visual de alto nível, segundo Kandel et al. (2012, p. 636), ainda pouco se sabe sobre os mecanismos que causam as representações neurais dos objetos. Ainda menos se conhece sobre como tais mecanismos se modificam através da experiência visual.

2.3 SUBAMOSTRAGEM CROMA

Como visto na seção anterior, o HVS é mais sensível à luminosidade do que à cor. Assim, ao se adotar a codificação Y'CbCr é possível utilizar resoluções menores nas matrizes de crominância (Cb e Cr), com pouco prejuízo para a qualidade subjetiva da imagem. De acordo com a subamostragem efetuada, linhas horizontais ou verticais são eliminadas das matrizes de crominância e, quando decodificadas, as amostras mantidas serão interpoladas para a geração de matrizes com a mesma resolução. Alguns exemplos de subamostragem croma comumente utilizadas em codificação de vídeo são 4:4:4, na qual não há subamostragem; 4:2:2, na qual amostra-se apenas uma dentre duas linhas ou dentre duas colunas dos canais de crominância; e 4:2:0, na qual amostra-se apenas uma linha a cada duas e apenas uma coluna a cada duas dos canais de crominância. A Figura 7 ilustra as subamostragens 4:4:4, 4:2:2 e 4:2:0.

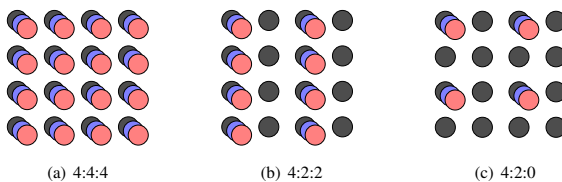


Figura 7: Exemplos de subamostragem croma. Cada círculo representa um pixel, enquanto a “profundidade” representa cada canal de cor: o conjunto de círculos mais escuros (mais ao fundo) representa o canal Y, enquanto os demais representam os canais Cb e Cr.

Embora relacionada a aspectos do HVS, a subamostragem croma é uma etapa da codificação que introduz perdas, pois a reconstrução dos pixels eliminados não será correta. Por outro lado, esta etapa pode gerar grandes ganhos de compressão, afinal, dependendo da subamostragem aplicada,

elimina-se grande parte das matrizes de pixels. Por exemplo, adotando subamostragem 4:2:2, há uma redução de $1/3$ no total de pixels.

2.4 REDUNDÂNCIAS INTERPIXEL

De forma a abranger o estado da arte em compressão de vídeo, foi estudado o padrão de compressão de vídeo H.264/AVC (*Advanced Video Coding*), o qual foi desenvolvido pela ITU-T *Visual Coding Experts Group* (VECG) em conjunto com a ISO/IEC *Moving Picture Experts Group* (MPEG), em uma parceria conhecida por *Joint Video Team* (JVT).

O H.264/AVC é usado em HD-DVD e *Blu-ray*, além de ser o formato de vídeo adotado no Brasil para TV digital. É um dos padrões estado da arte em compressão de vídeo, devido a seu aumento em taxa de compressão, em comparação com padrões anteriores³. O H.264/AVC é o antecessor do padrão de Codificação de Vídeo de Alta Eficiência - *High Efficiency Video Coding* (HEVC). Mesmo com o lançamento do HEVC, o H.264/AVC ainda pode ser considerado estado da arte, principalmente no que diz respeito ao consumo energético em plataformas embarcadas, mantendo qualidade e baixo *bitrate*. A Figura 8 apresenta um diagrama do fluxo de codificação do padrão H.264/AVC.

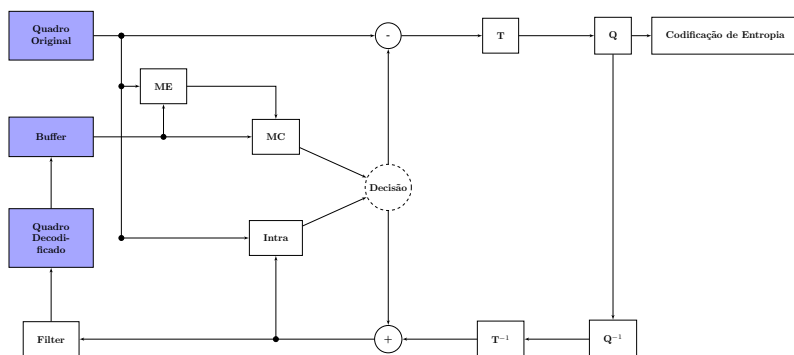


Figura 8: Diagrama simplificado do fluxo de codificação de acordo com o padrão H.264/AVC (ITU-T, 2009). A ME será descrita na Subseção 2.4.1, enquanto a predição intra quadro será descrita na Subseção 2.4.4. Ainda, serão abordadas de forma introdutória as etapas de Transformada (T) e Quantização (Q), nas Subseções 2.4.5 e 2.4.6, respectivamente.

³De acordo com Kamaci e Altunbasak (2003), o padrão H.264/AVC teve ganhos de até 50% em eficiência sobre seu predecessor, o MPEG-2.

Em geral, o fluxo de codificação é muito similar ao dos demais padrões, exceto pela adição de algumas novas ferramentas. Por exemplo, uma novidade do H.264/AVC em relação ao MPEG-2 é a chamada predição intra quadros, a qual será abordada na Subseção 2.4.4. No decorrer desta seção serão apresentadas as ferramentas para redução das redundâncias interpixel, seguindo o diagrama da Figura 8. A Subseção 2.4.1 apresenta a ME, alguns algoritmos de busca para ME e também métricas de similaridade.

2.4.1 Estimação De Movimento

A estimação de movimento (ME) é um dos grandes responsáveis pela alta taxa de compressão em vídeos digitais, sendo por isso um dos passos mais importantes na codificação. Dados um bloco do quadro a ser codificado e uma janela de busca em um quadro de referência, a ME procura o bloco da janela de busca que seja mais similar ao bloco do quadro a ser codificado. Os algoritmos de busca e as métricas de similaridade mais utilizados na ME são abordados nas Subseções 2.4.2 e 2.4.3, respectivamente. O bloco mais similar ao (ou de melhor casamento com o) bloco do quadro a ser codificado é então apontado por um Vetor de Movimento - *Motion Vector* (MV), que é um vetor bidimensional. Como é possível observar na Figura 9, um quadro referência, com a nuvem mais ao alto, é usado como referência para o quadro atual (mais à direita). A entre os quadros representa o MV, cujo valor armazenado é a coordenada, no caso “(2,C)”. O resíduo⁴ é mostrado, na Figura 9, sob a coordenada do MV.

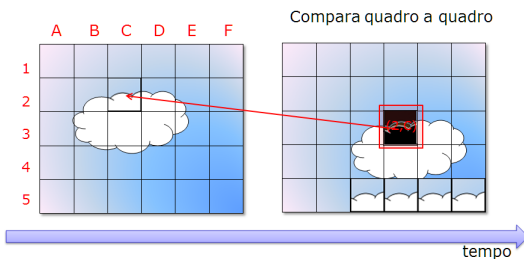


Figura 9: Exemplo de ME.

A ME visa eliminar a redundância temporal, ou seja, entre quadros (inter quadro, conforme visto no início deste capítulo). Por este motivo, a

⁴Resíduo: diferença entre a referência e o bloco a ser codificado. Ao reconstruir o bloco, esse resíduo é somado ao bloco de referência, obtendo assim uma boa aproximação do bloco que foi codificado, exceto pelo erro causado pela quantização desse resíduo.

etapa na qual ela ocorre é chamada predição inter quadros. Após a ME, apenas o MV e o resíduo do bloco atual são enviados para etapas posteriores da codificação, ao invés de toda a informação de cada bloco. Durante o processo de codificação, a estimação de movimento ocorre antes do passo da transformada e quantização, de forma que apenas o resíduo é transformado e quantizado.

No padrão H.264/AVC existe a possibilidade de um MV apontar para fora do quadro. Essa modificação consiste em gerar novos candidatos para a ME, estendendo as bordas de um bloco (localizado na borda do quadro) assim como mostrado na Figura 10. Dessa forma, ao se estender a borda, um novo bloco é criado a partir da borda, similar ao processo da Predição Intra (Subseção 2.4.4). Com este novo bloco como candidato, um MV pode apontar para este candidato, ou seja, apontar para fora do quadro.

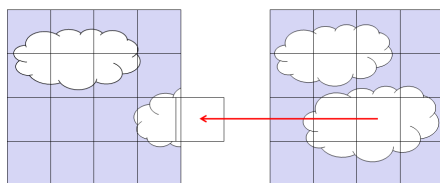


Figura 10: Predição de movimento com uso de vetores que apontam para fora do quadro.

Outra proposta do padrão é a precisão de $1/4$ de pixel na ME. Isso significa mais candidatos na busca, mas, em compensação, movimentos mais sutis podem ser representados na ME, reduzindo o resíduo. Assim, para os blocos candidatos são gerados novos blocos através de interpolação, primeiramente com precisão de $1/2$ pixel e estes são novamente interpolados para precisão de $1/4$ de pixel. A Figura 11 mostra, de maneira simplificada, um MV apontando para uma posição de $1/4$ de pixel.

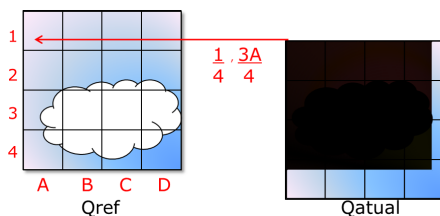


Figura 11: Simplificação da ME com precisão de $1/4$ de pixel.

Ainda, outra novidade do H.264/AVC em relação ao antecessor é que, ao invés de manter apenas uma referência fixa para blocos do tipo Predito (P) e duas referências para os do tipo Bipredito (B) (como em padrões anteriores), o H.264/AVC utiliza uma lista de referências passadas (lista 0) e uma lista com referências futuras (lista 1). Dessa forma, um bloco pode apontar para qualquer uma das referências de qualquer das duas listas. As listas “0” e “1”, bem como um quadro apontando para múltiplas referências, são ilustradas na Figura 12. O tamanho das listas vai requerer um maior ou menor tamanho no *buffer* de codificação/decodificação.

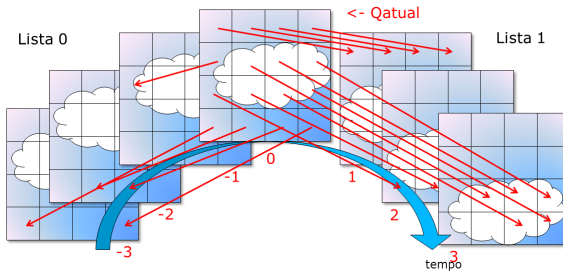


Figura 12: Representação de múltiplas referências de um quadro no padrão H.264/AVC.

A grande maioria dos filmes faz transições suaves entre suas cenas. Normalmente, essa suavidade é atingida utilizando-se de técnicas de “*fade in*”, “*fade out*” e “*cross-fade*”⁵. O H.264/AVC otimiza a compressão deste tipo de técnica, pois implementa a chamada predição com peso. Neste tipo de predição, após detectar que existe uma mudança gradativa na cena, utiliza os blocos já codificados aplicando a eles um “peso”, ou seja, multiplica-os por um fator, dessa forma diminuindo o resíduo para estes blocos.

Por fim, além da predição de movimento, há também a predição de vetores de movimento. Esta modalidade de predição utiliza os MVs dos blocos vizinhos como forma de comprimir o MV do bloco atual. Assim, apenas o resíduo entre o MV do bloco atual e a mediana dos MVs dos blocos vizinhos, chamado de Vetor de Movimento Predito - *Predicted Motion Vector* (PMV), são enviados para etapas posteriores da codificação.

⁵ **Fade in/Fade out:** Recursos de edição de vídeo que permitem transições suaves entre as cenas. *Fade in* normalmente refere-se a ir do escuro para a luz, enquanto *fade out* implica escurecer a cena ou transicionar para o preto. Adaptado de (JACK; TSATSULIN, 2002).

Cross-fade: Diminuir a intensidade de um sinal de vídeo e aumentar de outro, em um movimento simultâneo. Adaptado de (JACK; TSATSULIN, 2002).

2.4.2 Algoritmos de Busca para Estimação de Movimento

O uso do *Fullsearch Block Matching Algorithm* (FBMA) (BARNEA; SILVERMAN, 1972) na ME é proibitivo em aplicações com restrições de tempo real e/ou consumo de energia (SEIDEL et al., 2013b). Assim, diversos outros algoritmos (chamados de algoritmos de busca rápidos) foram propostos para diminuir a área de busca e o número de candidatos a serem comparados, acelerando a ME com perdas mínimas em relação ao FBMA. Dentre os algoritmos propostos, destacam-se os seguintes:

- *2D Logarithmic Search* (JAIN; JAIN, 1981): busca no centro da janela de busca e no centro das quatro bordas, continua a busca no melhor dos resultados, dividindo a largura da janela pela metade, parando de acordo com a distância do último melhor candidato;
- *Alternating pel-subsampling block matching algorithm* (APS) (JUNG et al., 1995): usa *pel decimation*, alternando o padrão de subamostragem pela área de busca;
- Algoritmo de Eliminação Global - *Global Elimination Algorithm* (GEA) (HUANG et al., 2002): derivado do *Successive Elimination Algorithm* (SEA) (LI; SALARI, 1995), o qual elimina alguns dos candidatos através do uso de limiares (*thresholds*);
- *Quartet-pel motion estimation* (QME) (LEE et al., 2004), que é executado em duas etapas principais:
 1. Subamostragem 4:1;
 2. Cálculo da Soma das Diferenças Absolutas - *Sum of Absolute Differences* (SAD) completa com os N principais candidatos resultantes da Etapa 1.
- Busca Aleatória Iterativa - *Iterative Random Search* (IRS) (PORTO et al., 2013): escolhe candidatos aleatórios na área de busca e itera sobre os melhores resultados, buscando ser menos suscetível a mínimos locais.

Os algoritmos de busca tem uma característica comum, que é o uso de métricas de similaridade para o casamento de blocos. Algumas destas métricas serão apresentadas na próxima subseção.

2.4.3 Métricas de Similaridade

As métricas de similaridade, também chamadas de métricas de distorção, podem ser, a grosso modo, classificadas em dois tipos: Objetivas e Subjetivas. As primeiras podem pertencer a uma entre duas abordagens (MORAES, 2010):

- Métricas puramente baseadas em modelos matemáticos, usados em processamento de sinais;
- Métricas psicovisuais, que usam características do HVS.

As primeiras são modelos puramente matemáticos derivados da teoria de sinais e não consideram características do HVS. Entretanto, por questões de desempenho, são as mais usadas. Já as métricas psicovisuais exploram modelos do HVS. As poucas existentes sofrem de baixo desempenho e exploram apenas um pequeno conjunto de características. Um exemplo é o Índice de Similaridade Estrutural - *Structural Similarity Index* (SSIM), que explora a capacidade do olho humano de identificar estruturas. Por sofrerem de baixo desempenho (alta complexidade computacional para seu cálculo), além de não explorar por completo o HVS, não são muito usadas nem mesmo muito aceitas (WU; RAO, 2005). Exemplos de métricas de similaridade comumente utilizadas são a SAD, a Soma das Diferenças Quadráticas - *Sum of Squared Differences* (SSD) e a Soma das Diferenças Transformadas Absolutas - *Sum of Absolute Transformed Differences* (SATD).

As métricas de similaridade também podem ser classificadas em métricas objetivas e métricas subjetivas. As chamadas métricas objetivas são baseadas em critérios e métricas que podem ser medidas objetivamente e avaliadas de forma automática por um programa de computador. São modelos matemáticos, incluem métricas da teoria de sinais e também métricas psicovisuais. Há, dessa forma, certa sobreposição e confusão entre estas classificações. Já as métricas subjetivas tem por objetivo aproximar a opinião de um usuário, e pode incluir o esforço de avaliadores.

2.4.3.1 Métricas Objetivas

As métricas de similaridade objetivas mais usadas são apresentadas a seguir.

- *SAD* – Soma das Diferenças Absolutas: métrica de similaridade constituída pela soma do valor absoluto das diferenças entre cada amostra

original (*Ori*) e a amostra de um bloco candidato a ser usado como referência (*Ref*), após o processo de reconstrução. Devido à sua baixa complexidade computacional é adequada para implementação direta em *hardware* (RICHARDSON, 2003).

$$SAD = \sum_{i=0}^M \sum_{j=0}^N |Ori_{i,j} - Ref_{i,j}| \quad (2.1)$$

- *SSD* – Soma das Diferenças Quadráticas: corresponde à soma do quadrado das diferenças entre cada amostra original e a amostra após o processo de reconstrução. Sua característica principal é evidenciar os erros significativos (MANOEL, 2007).

$$SSD = \sum_{i=0}^M \sum_{j=0}^N (Ori_{i,j} - Dec_{i,j})^2 \quad (2.2)$$

- *SATD* – Soma das Diferenças Transformadas Absolutas: baseada na *SAD*, mas usa a transformada *Hadamard* (*H*) em uma partição 4x4 de diferenças antes de efetuar a soma absoluta dos coeficientes. Apresenta maior custo que a *SAD*, com melhor qualidade por operar no domínio das frequências⁶ (MANOEL, 2007).

$$Hadamard\ H = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \\ 1 & -1 & 1 & -1 \end{bmatrix}$$

$$SATD = \frac{1}{2} \sum_{i=0}^3 \sum_{j=0}^3 |H(Ori_{i,j} - Dec_{i,j})H^T| \quad (2.3)$$

- *MSE* – Erro Quadrático Médio: corresponde à média do quadrado das diferenças entre cada amostra original e a amostra correspondente após o processo de codificação/decodificação (MANOEL, 2007), calculada em um bloco com tamanho *M* por *N*.

$$MSE = \frac{SSD}{M \times N} \quad (2.4)$$

⁶Obtém-se assim uma aproximação da Transformada Discreta dos Cossenos - *Discrete Cosine Transform* (DCT), através da transformada *Hadamard* (ZHU; XIONG, 2009).

2.4.4 Predição Intra

Surgida no H.264/AVC, a chamada predição intra explora as redundâncias intra quadros (espaciais (MANOEL, 2007)). Na predição intra, são gerados blocos candidatos através de amostras das bordas externas do bloco atual. Estas bordas pertencem a blocos previamente codificados e reconstruídos. Busca-se então, dentre os blocos gerados, o candidato cuja similaridade seja a maior com o bloco atual, seguindo o mesmo princípio da ME, mas com menos candidatos. Em muitos casos, métricas de similaridade com mais operações são adotadas na predição intra para melhorar a predição, sem causar grande impacto no fluxo geral de codificação. Na Figura 13 são exemplificados, para blocos 4x4, os 9 candidatos gerados (8 direcionais e um planar), de acordo com o H.264/AVC. No padrão de HEVC são gerados 34 candidatos intra 4x4 direcionais em adição ao planar (WIEN, 2013).

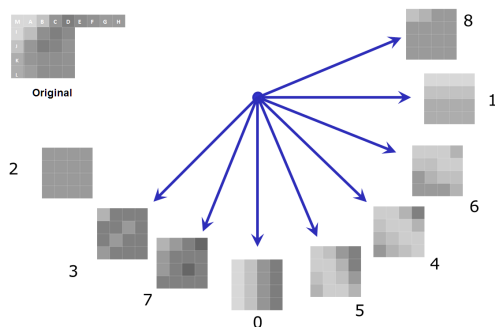


Figura 13: Bloco original e os nove possíveis candidatos intra 4x4, de acordo com o H.264/AVC. Adaptado de:(MANOEL, 2007)

2.4.5 Transformada

Os codificadores de vídeo baseados em blocos, tais como o H.263, *Moving Pictures Expert Group - 2* (MPEG-2), H.264/AVC e HEVC, são fundamentados na codificação de frequências. Uma das transformadas utilizadas para tal propósito é a DCT. Pode-se dizer que a DCT (AHMED; NATARAJAN; RAO, 1974) é uma simplificação da Transformada Rápida de Fourier - *Fast Fourier Transform* (FFT), sendo computacionalmente mais simples e mais efetiva para compressão multimídia. Posteriormente, através da etapa de quantização, a DCT ajuda na eliminação das redundâncias espaciais (intra

quadro), ao remover as frequências às quais o HVS é menos sensível (altas frequências). A transformada ocorre após a ME ou predição intra, sobre o resíduo a ser codificado.

Existem várias otimizações comumente aplicadas para melhorar o desempenho no cálculo da DCT como, por exemplo, a redução dos quadros em blocos (como é o caso da codificação de vídeo), pré-cálculo de cossenos, etc. A DCT do H.264/AVC é simplificada para uma operação de multiplicação de matrizes (matriz da transformada - \mathbf{T}), com inversa exata⁷. A matriz de coeficientes DCT \mathbf{C} para a matriz de amostras \mathbf{A} é descrita como:

$$\mathbf{C} = \mathbf{T}\mathbf{A}\mathbf{T}^T \quad (2.5)$$

Sendo a matriz de transformada (\mathbf{T}) 4x4:

$$\mathbf{T} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 2 & 1 & -1 & -2 \\ 1 & -1 & -1 & 1 \\ 1 & -2 & 2 & -1 \end{bmatrix} \quad (2.6)$$

No HEVC a DCT é definida para mais tamanhos de bloco, neste chamados de Unidades de Transformada - *Transform Units* (TUs): 4x4, 8x8, 16x16, e 32x32 (SULLIVAN et al., 2012). Além disso, os blocos 4x4 que adotam a predição intra são codificados com uma aproximação inteira da Transformada Discreta dos Senos - *Discrete Sine Transform* (DST). Segundo Sullivan et al. (2012), tal medida demonstra 1% de redução no *bitrate*.

2.4.6 Quantização

Durante o fluxo de compressão de vídeo, a quantização é a grande responsável pela perda na qualidade, mas também no ganho de compressão. Resumidamente, o passo de quantização consiste em dividir a matriz de pixels por uma matriz chamada matriz de quantização. A quantização ocorre sobre os coeficientes transformados pela DCT, gerando então uma matriz de coeficientes quantizados. Quanto maiores os coeficientes da matriz de quantização, maior a compressão e também maior a perda. No caso da adoção de aritmética inteira, as perdas são introduzidos ao se abandonar o resto da divisão inteira, o qual não será recuperado no processo inverso.

⁷ Sendo a inversa exata, este passo da codificação é sem perdas. Por outro lado, o reflexo da DCT virá no passo de Quantização, onde os coeficientes, além de ajustados para manter uma melhor aproximação com a DCT real, também sofrerão divisão inteira, a qual, por sua vez, não é inversível e introduz perdas.

2.5 MÉTRICAS DE QUALIDADE

Para finalizar este capítulo de contextualização, é importante mencionar duas métricas de qualidade, as quais serão utilizadas no próximo capítulo, para análise dos experimentos propostos. São elas:

- *PSNR* - Razão Sinal-Ruído de Pico: métrica de qualidade objetiva de vídeo mais amplamente usada. Seu valor é dado em decibéis para amostras de n bits (POYNTON, 2012; MANOEL, 2007; RICHARDSON, 2003)

$$PSNR = 10 \log_{10} \frac{(2^n - 1)^2}{MSE} \quad (2.7)$$

A análise apenas de Relação Sinal-Ruído de Pico - *Peak signal-to-noise ratio* (PSNR) é limitante, uma vez que é uma métrica objetiva baseada em modelos matemáticos de processamento de sinais que pouco se correlaciona com a percepção do HVS (WU; RAO, 2005). Tal métrica é utilizada comumente para medir a diferença entre os quadros fonte e os quadros codificados em uma sequência de vídeo. É bastante conhecido o problema da PSNR quando se trata de correlacionar tal erro (diferença) com o HVS: tal fato pode ser observado nas duas imagens da Figura 14, as quais tem o mesmo PSNR, mas a imagem à esquerda tem melhor qualidade visual que a outra (WINKLER, 2005).



Figura 14: As duas imagens possuem o mesmo PSNR, mas o ruído está distribuído em regiões diferentes. Fonte: (WINKLER, 2005).

Segundo Larbier (2011), ao se comparar duas sequências de vídeo diferentes, o PSNR tem pouco valor. Por outro lado, duas medidas de PSNR para uma mesma sequência demonstra muito bem o potencial de dois codificadores. Por exemplo, assumir o valor de PSNR como medida para determinação da qualidade de um padrão de *pel decimation* pode não refletir seu comportamento mais geral e subjetivo. Há outras métricas, mesmo objetivas, que se aproximam mais do HVS, como por exemplo, o SSIM:

- *SSIM* - Índice de Similaridade Estrutural: tem valor máximo 1, o que corresponde à duas imagens X e Y, exatamente iguais, estruturalmente (WANG et al., 2004b).

$$SSIM(x, y) = \frac{(2\mu_x\mu_y + c_1)(2\sigma_{xy} + c_2)}{(\mu_x^2 + \mu_y^2 + c_1)(\sigma_x^2 + \sigma_y^2 + c_2)} \quad (2.8)$$

Sendo:

- μ_x a média dos valores de pixel de x;
- μ_y a média dos valores de pixel de y;
- σ_x^2 a variância dos valores de pixel de x;
- σ_y^2 a variância dos valores de pixel de y;
- σ_{xy} a covariância dos valores de pixel de x e y;
- $c_1 = (k_1L)^2$, $c_2 = (k_2L)^2$ variáveis para estabilizar a divisão por denominador fraco;
- L a gama dinâmica do pixel, $2^{\#bits \text{ pixel}} - 1$;
- $k_1 = 0.01$ e $k_2 = 0.03$ por padrão.

Esta última métrica tem sido alvo de estudos recentes, tais como os de Wang et al. (2012) e Rehman e Wang (2012), por ter maior correlação com o HVS. Tal correlação vem do fato de que a SSIM, como o próprio nome já diz, mede a similaridade estrutural entre as imagens. Portanto, ela tem maior sensibilidade às bordas (estruturas) dos objetos, da mesma forma que o olho humano, conforme apresentado na Subseção 2.2.1.

2.5.1 Métricas Subjetivas

As métricas de qualidade subjetivas estão relacionadas à percepção do observador do vídeo para uma sequência de um vídeo em particular. Testes de qualidade subjetivos são bastante complexos para serem executados em

termos de tempo e recursos humanos. A recomendação BT.500-13 (ITU-T, 2012a) descreve uma metodologia para a avaliação de qualidade subjetiva, estabelecendo diversos parâmetros para tal avaliação.

3 AVALIAÇÃO DA QUALIDADE

Este capítulo trata da avaliação de qualidade de 14 padrões de subamostragem, sendo sete deles propostos no contexto do presente trabalho e os demais sete, comumente utilizados. Primeiramente, a literatura a respeito de análise de qualidade e padrões de subamostragem é devidamente revisada. Depois, são descritos os padrões de subamostragem, os experimentos feitos para a análise de qualidade e os respectivos resultados, em termos de qualidade e velocidade de codificação. Os resultados deste capítulo foram publicados em forma de artigo completo nos anais do ICME¹ 2013 (SEIDEL et al., 2013), e suas contribuições ao estado da arte são:

- Proposta de dois padrões de subamostragem com razão 4:3, sendo esta razão de subamostragem incomum;
- Ampla avaliação estatística, usando Análise de Variância - *Analysis of Variance* (ANOVA), da qualidade de quinze padrões fixos de amostragem. Esta avaliação foi feita usando duas medidas objetivas de qualidade, Relação Sinal-Ruído de Pico - *Peak signal-to-noise ratio* (PSNR) e Índice de Dissimilaridade Estrutural - *Structural Dissimilarity Index* (DSSIM) (LOZA et al., 2006), com um significativo grupo de amostras de vídeos em muitas resoluções e bitrates, com ênfase em vídeos de alta definição;
- Avaliação da velocidade de codificação em *software* dos padrões de subamostragem;
- Avaliação do impacto de *pel decimation* conforme a resolução do vídeo aumenta.

3.1 TRABALHOS CORRELATOS

Dentre os algoritmos de busca rápidos listados na Seção 2.4.2, alguns usam o já mencionado *pel decimation* para acelerar a Estimção de Movimento - *Motion Estimation* (ME) (*Alternating pel-subsampling block matching algorithm* (APS) e *Quartet-pel motion estimation* (QME)). Este ainda pode ser aplicado em conjunto com as demais buscas mencionadas, mas seu custo, em termos de qualidade, ainda não foi avaliado em tal contexto. A Figura 4 mostrou alguns dos padrões de subamostragem mais utilizados para

¹ *IEEE International Conference on Multimedia and Expo.*

o algoritmo de *pel decimation*, os quais normalmente adotam as razões 2:1, 4:1 e 8:1. A razão de subamostragem influencia o aumento esperado de velocidade através determinação da quantidade de dados desprezados. Porém, como já mencionado, o algoritmo de *pel decimation* não define nenhum padrão de subamostragem. Tal questão permanece em aberto e, neste sentido, a busca por novos padrões tem como objetivo reduzir as perdas de qualidade enquanto se reduz o esforço computacional da ME.

O trabalho de Wang et al. (2004) estuda este problema da escolha do padrão de subamostragem e demonstra que o uso de um padrão referenciado por *N-Queen lattice* no software de referência do MPEG-4 resulta em melhor qualidade de compressão, comparado a padrões anteriormente propostos. Wang et al. (2004) também apresentam um estudo de caso de uma arquitetura de hardware de estimação de movimentos usando o padrão *4-Queen*². Saha (2007) usou um Algoritmo Genético - *Genetic Algorithm* (GA) para procurar por padrões ótimos de *pel decimation*, seguindo os critérios propostos por Wang et al. (2004), quais sejam:

- homogeneidade espacial;
- cobertura direcional.

Porém, ao contrário do esperado, os resultados de qualidade dos padrões “ótimos” encontrados foram piores do que os do *N-Queen*, indicando que tais critérios não são suficientes. O uso do *pel decimation* baseado em bordas é apresentado por Saha, Mukherjee e Sural (2008), como uma forma de detectar um novo objeto entrando em um Macrobloco - *Macrobloc* (MB). Seguindo o trabalho de Saha (2007), GA e os critérios de Wang et al. (2004) são usados na busca por padrões *N-Queen* ótimos combinados com *pel decimation* baseada nas bordas. As principais limitações desses três trabalhos, que prejudicam a aplicabilidade dos seus resultados, são:

- Análise de apenas uma métrica objetiva, a PSNR (RICHARDSON, 2003);
- Uso de poucos quantizadores;
- Testes usando apenas amostras de baixa resolução.

Conforme já mencionado na Seção 2.5, o uso exclusivo da PSNR como métrica de qualidade é um fator limitante da análise, uma vez que esta é incapaz de capturar certas nuances do Sistema Visual Humano - *Human Visual System* (HVS).

²O padrão *4-Queen* está representado na Figura 15(k).

O uso de poucos quantizadores (usados em etapa posterior à ME) define uma limitação da qualidade final dos quadros com a taxa de bits por pixel na compressão. Não foi encontrada demonstração de que o comportamento do *pel decimation*, em termos de qualidade, seja o mesmo para todo quantizador. Dessa forma, são pouco confiáveis os resultados de apenas dois ou três quantizadores. Além disso, o trabalho de Wang et al. (2004) analisa apenas as resoluções CIF e CCIR-601, enquanto os trabalhos de Saha (2007) e Saha, Mukherjee e Sural (2008) usam apenas amostras de vídeos de baixa resolução (QCIF e CIF). Assim, é esperado que o *pel decimation* tenha maior contribuição para o decréscimo na qualidade, uma vez que cada pixel representa maior percentual de informação por quadro. Supõe-se então que tenha menor impacto na qualidade para vídeos de maiores resoluções, mas nenhuma demonstração havia sido encontrada na literatura até o momento em que esta dissertação foi escrita.

3.2 PADRÕES DE SUBAMOSTRAGEM AVALIADOS

Os padrões de subamostragem considerados neste trabalho são definidos em termos de blocos de 4x4 pixels, mas são extensíveis para blocos maiores, como explicado no final desta seção. Apenas padrões fixos são utilizados pois, como afirmado por Wang et al. (2004), padrões de amostragem adaptativos não conduzem a implementações de hardware eficientes, embora poderiam alcançar melhores eficiências de codificação. No total, 14 padrões de subamostragem foram considerados na avaliação de qualidade: sete deles podem ser considerados convencionais, ao passo que os sete restantes são propostos no contexto do presente trabalho de mestrado. Além dos padrões de subamostragem propriamente ditos, a Soma das Diferenças Absolutas - *Sum of Absolute Differences* (SAD) completa (i.e., sem subsamostragem) também foi incluída neste estudo. Todos os padrões considerados neste trabalho estão representados na Figura 15. Nela os padrões propostos estão envolvidos por um quadrado tracejado.

Dentre os padrões convencionais, os mais aceitos e utilizados (KUNH, 1999; JUNG et al., 1995; LEE et al., 2004) são: *checker even 2:1* (Figura 4(a)), também conhecido como *Quincunx* (LENGWEHASARIT; ORTEGA, 2001; WANG et al., 2004), e *checker odd 2:1*. Os padrões chamados *checker* são derivados através da alternância de pixels amostrados sobre o bloco, como em um tabuleiro de xadrez.

Os padrões *checker* de razão 4:1 têm quatro simetrias. Neste trabalho foram consideradas apenas duas delas: *checker even 4:1* (Figura 4(b)) e *checker odd 4:1*. O primeiro é uma eliminação das linhas 1 e 3 do padrão *checker*

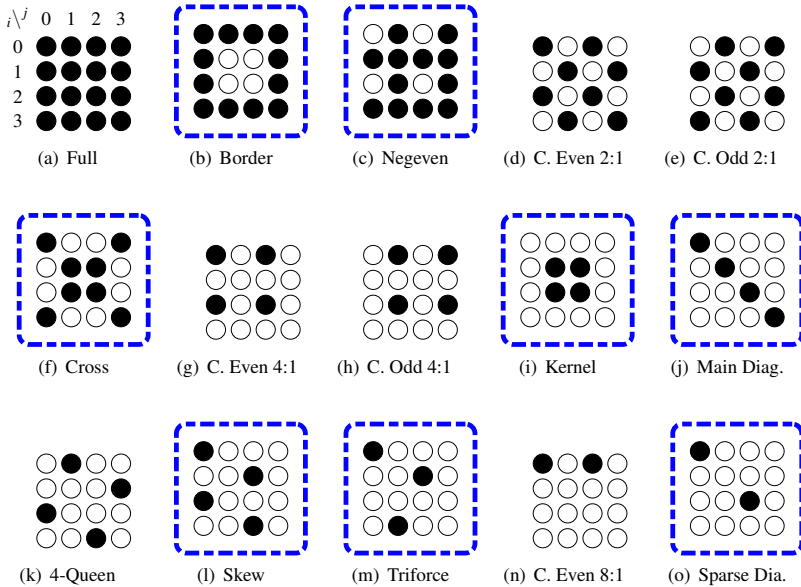


Figura 15: Representação em blocos 4×4 da amostragem completa e os 14 padrões de subamostragem avaliados. Pontos cheios indicam pixels amostrados pela métrica de similaridade. Os padrões estão organizados por razão de subamostragem. Os padrões marcados em blocos tracejados são os 8 padrões propostos.

even 2:1. Já o segundo, é derivado pela eliminação das linhas 0 e 2 do padrão *checker odd 2:1*. Também foi considerado o padrão *checker even 8:1* (Figura 4(c)), derivado do padrão *checker even 4:1*. Além dos cinco padrões *checker*, também foram incluídos os padrões *4-Queen* (Figura 15(k)) (WANG et al., 2004) e *skew 4:1* (Figura 15(l)). O último é similar ao *checker 4:1* mas é mais espalhado. Uma visão alternativa deste é um cisalhamento de um pixel da segunda coluna do padrão *checker*.

Dentre os padrões de subamostragem propostos, o chamado *border* (Figura 15(b)) utiliza apenas pixels das bordas do bloco, resultando em uma razão de subamostragem não convencional, 4:3. O padrão *negeven* (Figura 15(c)) também resulta nesta razão 4:3. É derivado a partir da amostragem dos pixels não amostrados do padrão *checker even 4:1*. Uma vez que os padrões de razão 4:3 incluem mais informação sobre o bloco, espera-se que sejam mais precisos na estimação, embora ainda economizando por volta de 1/4

das operações. Por sua vez, o padrão *cross* (Figura 15(f)) usa as diagonais principal e secundária dos blocos. O padrão *main diagonal* (Figura 15(j)) usa apenas os pixels da diagonal principal de cada bloco.

O padrão *kernel* (Figura 15(i)) é derivado através da amostragem dos pixels apenas do núcleo do bloco (região central), resultando na razão de amostragem 4:1. Pode-se perceber que este padrão é o complemento do padrão *border*. O padrão chamado *triforce* (Figura 15(m)) tem um pixel a menos que padrões de razão 4:1, resultando em outra razão não convencional de subamostragem, de 16:3. Finalmente, o padrão chamado *sparce diamond* (Figura 15(o)) tem razão 8:1.

Os padrões de subamostragem considerados neste trabalho foram definidos em termos de blocos 4x4 pois todas as demais partições de um MB podem ser expressas como composições deste tamanho de bloco, como é ilustrado na Figura 16. Tais composições são amplamente utilizadas em implementações do padrão H.264/AVC, incluindo a *Joint Model* (JM) (JVT, 2011).

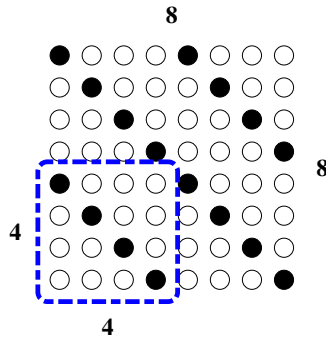


Figura 16: Exemplo da aplicação de um bloco de 4x4 pixels usando o padrão *main diagonal* para a formação de um bloco 8x8.

3.3 CONFIGURAÇÃO EXPERIMENTAL

Para avaliar a qualidade de padrões de subamostragem, tomou-se por base o codificador *x264* (MERRITT et al., 2004), uma vez que este é aproximadamente 50 vezes mais rápido que a JM, enquanto resulta em variações de *bitrate* de apenas 5% (MERRITT; VANAM, 2007)³. Esta performance vi-

³De acordo com Merritt e Vanam (2006), tal aceleração é atribuída ao seu controle de taxa de bits, ME, decisão de modos para MB, quantização e decisão de tipo de quadro. Além disso, o

Tabela 2: Resoluções das amostras de vídeo e suas respectivas taxas de bit *lossless* (Mbps).

Source	CIF	480p	720p	1080p	2160p
Akiyo	4.70				
Big Buck Bunny		20.65			
Crowd Run				642.24	2594.00
Ducks Take Off			324.32	754.15	
Foreman	13.36				
In to Tree			244.77	594.64	
Park Joy			284.71	659.75	
Pedestrian Area				205.82	
Riverbed				253.66	
Sintel trailer		12.83	26.25	54.28	
Soccer	13.38				
Sunflower				218.81	
Tractor				238.05	

abilizou os testes estatísticos apresentados. O código do *x264* (revisão 2106) foi modificado para incorporar cada um dos 14 padrões de subamostragem apresentados. Para avaliar a qualidade, foi escolhida a SAD, como expressa na Equação 2.1. Para cada padrão, a Equação 2.1 foi modificada de acordo. Para que as comparações em termos de velocidade de codificação fossem justas, nos experimentos todas as funções escritas diretamente em linguagem *assembly* no *x264* foram desabilitadas.

Foram selecionados 19 vídeos representativos (com 13 conteúdos diversos) do repositório Xiph.org (XIPH.ORG, 2011). A Tabela 2 mostra suas resoluções e *bitrates lossless*. Para avaliar o impacto da redução de resolução, em comparação com a subamostragem, foram incluídos vídeos com mesmo conteúdo, mas diferentes resoluções (*Crowd Run*, *Ducks Take Off*, *In to Tree*, *Park Joy*⁴ e *Sintel trailer*⁵). Todos foram limitados a 720 quadros (o que cor-

x264 utiliza diversas otimizações de código implementadas diretamente em linguagem *assembly*. Porém, ainda de acordo com Merritt e Vanam (2006), o ganho da ME no *x264* advém da inclusão de término prematuro no caso do algoritmo *Uneven MultiHexagon* (UMH), similar ao da JM.

⁴Todo o processo de obtenção destas três amostras de vídeo foi documentado por Haglund (2006). Utilizou-se uma câmera *ARRI ArriFlex 765 System for 65mm, 5 perf. film*, à 50 Quadros por Segundo - *Frames per Second* (fps). As amostras foram cuidadosamente convertidas para seu formato digital para a resolução máxima de 3840x2160p, os quais foram posteriormente filtrados para que se obtivessem as menores resoluções.

⁵No caso do *Sintel trailer*, as três resoluções foram obtidas através de renderizações para a resolução desejada.

responde, no caso de fontes com 24fps, a 30 segundos). Os seguintes bitrates (em Mbps) foram considerados: 0,25; 0,375; 0,5; 0,75; 1; 1,5; 2; 3; 4; 6; 8; 9; 10; 11; 12; 14; 16; 18; 20; 24; 32; 40; 50; 60; 80; 100; 120; 160; 180 e 200.

Ao comparar a qualidade atingida é importante que os bitrates comparados sejam os mesmos (LARBIER, 2011). Dessa forma, para garantir que os bitrates alvo sejam atingidos, o codificador foi configurado no modo 2 passos (*2-pass*), embora 1 passo seja mais conveniente para implementação em *hardware*. As métricas objetivas de qualidade avaliadas foram DSSIM, a qual é uma métrica de distância derivada do valor de similaridade estrutural Índice de Similaridade Estrutural - *Structural Similarity Index* (SSIM) (WANG et al., 2004a), e a PSNR. Para obter resultados válidos, para cada métrica objetiva uma codificação individual (em 2 passos) foi efetuada com o x264 otimizado para a métrica em questão. Os experimentos foram executados em uma máquina com as seguintes configurações: *2,40GHz 12M L3 Cache HyperThreaded dual Xeon E5620* com 12GB RAM, rodando Linux Debian 64-bit. Todas as 16 *threads* lógicas disponíveis foram utilizadas pelo codificador.

A principal contribuição desta primeira avaliação é obter a qualidade resultante exclusivamente do uso de diferentes padrões de subamostragem. Assim, admitindo que as razões de amostragem dos padrões são o único fator contribuinte para o aumento na velocidade de codificação, foi escolhido o *Successive Elimination Algorithm* (SEA) (LI; SALARI, 1995; GAO; DUANMU; ZOU, 2000; MERRITT; VANAM, 2007) como algoritmo de busca na ME, para melhor garantir o rigor da avaliação de qualidade, evitando ao máximo⁶ a influência da busca nos resultados. No Capítulo 5, veremos que o algoritmo também teve seu impacto nos resultados obtidos. Esta correlação tornou-se, de fato, a principal contribuição deste trabalho de mestrado.

3.4 RESULTADOS

Os 19 vídeos da Tabela 2 foram codificados usando cada um dos 15 padrões (14 padrões de subamostragem mais a amostragem completa), apresentados na Seção 3.2, para os 30 *bitrates* listados na Seção 3.3 e para PSNR e SSIM, traduzindo-se em um total de 17.100 execuções. Ao analisar os relatórios de um determinado vídeo, notou-se que a partir de alguns bitrates a qualidade não aumentava, mas os resultados gerais tornavam-se inconsis-

⁶No x264 o SEA é mais conhecido por Algoritmo de Busca por Eliminações Sucessivas - *Successive Elimination Exhaustive Search Algorithm* (ESA), pois é o mais próximo do algoritmo de busca exaustivo, ou seja, o *Fullsearch Block Matching Algorithm* (FBMA). Com razão, o SEA resulta no mesmo Vetor de Movimento - *Motion Vector* (MV) que o FBMA mas, segundo Merritt (2005) é entre duas a três vezes mais rápido.

tentes. Tais bitrates produzem codificações com perdas, mas eram além dos *bitrates* matematicamente sem perdas⁷. Assim, tais resultados foram descartados, restando ainda 10.680 pontos experimentais. Para cada vídeo, a Tabela 2 mostra os *bitrates* mínimos que resultam em codificações sem perdas.

A partir dos dados, já filtrados, resultantes das codificações foram criadas curvas de Taxa-Distorção - *Rate-Distortion* (RD) (*DSSIM versus bitrate* e *PSNR versus bitrate*) e curvas de velocidade de codificação (*fps versus taxa de bit*) para cada vídeo e para cada padrão de amostragem. Para o caso do trailer Sintel (para o qual havia disponíveis amostras em três resoluções), as curvas de RD relativas ao PSNR são apresentadas na Figura 17.

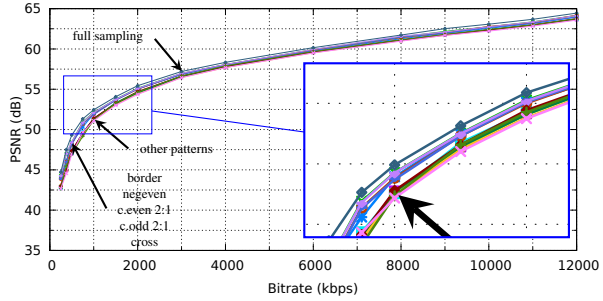
Apesar de ser impossível distinguir individualmente cada uma das curvas RD de todos os 15 padrões de amostragem, o comportamento geral é facilmente observado: enquanto as três figuras tem curvas com formas parecidas, as taxas de bit alcançadas pelo vídeo 1080p são praticamente o dobro daquelas obtidas por vídeos 720p que, por sua vez, são praticamente o dobro daquelas do 480p. Além disso, pode-se observar que quando a resolução e/ou taxa de bit aumenta, a diferença entre os padrões de subamostragem diminui, e todos os padrões de subamostragem se aproximam da amostragem completa. Assim, fica graficamente visível na Figura 17 que, para o mesmo *bitrate* de uma codificação sem *pel decimation*, seu uso tem menor impacto conforme se aumenta a resolução do vídeo.

Já para analisar os padrões de subamostragem usados com resolução de 1080p⁸, foram criadas curvas para o RD médio (*PSNR versus taxa de bit*, limitados a 40Mbps) e a velocidade média de codificação (*fps*) de amostras de vídeo de 1080p (exceto trailer Sintel, que tem um valor PSNR muito maior por ser uma animação) usando dados do x264 ajustado para PSNR. Estas curvas, apresentadas na Figura 18, evidenciam o comportamento dos padrões de subamostragem assim como a correlação entre *bitrate* e velocidade.

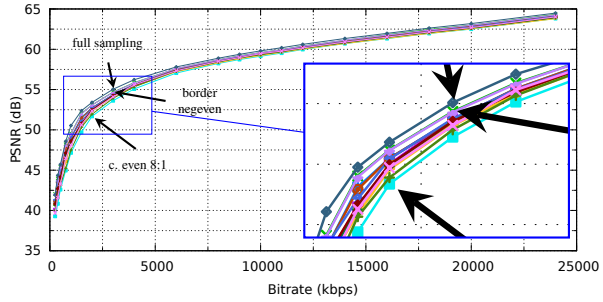
Ao passo que as curvas RD são muito próximas umas das outras, as curvas de velocidade de codificação são mais espaçadas. Além disso, é possível observar na Figura 18(a) que padrões de subamostragem com razões similares apresentam curvas de RD mais próximas. *Border* e *negeven* são os padrões que apresentam PSNR mais alto, seguidos pelos padrões 2:1. Na Figura 18(b) pode-se notar que o uso do padrão *4-Queen* resulta em menor velocidade média de codificação em comparação com outros padrões 4:1. Ainda, pode-se ver a grande diferença entre o uso da amostragem completa (*full sampling*) com a adoção de *pel decimation* 4:3. A maior aceleração apresentada foi do padrão *checker even* 8:1.

⁷Matematicamente sem perdas corresponde ao “predictive lossless profile”, conforme definido no padrão H.264/AVC (ITU-T, 2009), e implementado no x264.

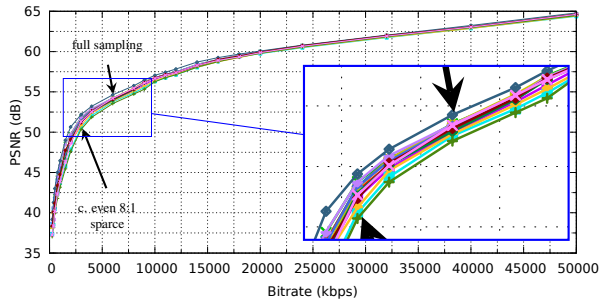
⁸Tal resolução foi selecionada formar o maior conjunto de amostras com a mesma resolução.



(a) 480p

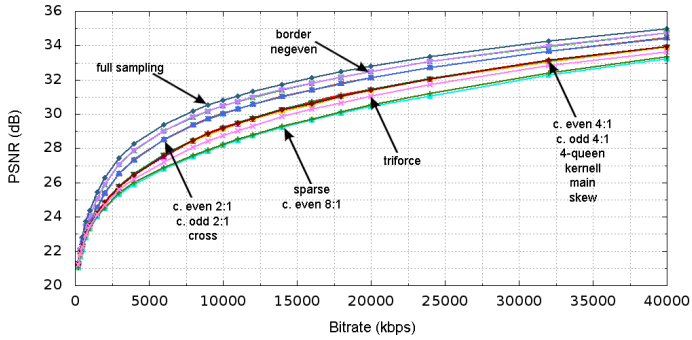


(b) 720p

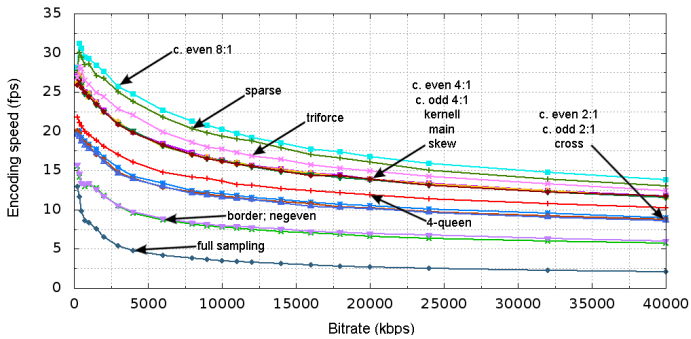


(c) 1080p

Figura 17: Resultados de PSNR para as amostras do trailer “Sintel” em diferentes resoluções.



(a) PSNR médio (dB)



(b) Velocidade média de codificação (fps)

Figura 18: PSNR médio e velocidade de codificação média para amostras de vídeo *full HD* 1080p.

Para permitir uma comparação ampla entre todos os padrões de sub-amostragem, incluindo as cinco resoluções consideradas, para cada padrão de amostragem foram calculadas as médias geométricas de DSSIM, PSNR e suas respectivas velocidades de codificação, sobre todas as 19 amostras de vídeos. Essas médias foram usadas para comparar os padrões de amostragem dois a dois, comparando as medidas de qualidade e velocidade de codificação para DSSIM e PSNR, como mostrado nas Tabelas 3 e 4, respectivamente. Estas tabelas devem ser lidas como segue. A diagonal principal está identificada com hífen. Cada célula acima da diagonal principal contém a medida de qualidade da imagem entre dois padrões. Por exemplo, na Tabela 3 pode-se observar que a razão de qualidade (DSSIM) entre o *border* e a amostragem completa é 0,952. Cada célula abaixo da diagonal principal contém a razão de velocidade de codificação entre dois padrões. Por exemplo, na Tabela 3 pode-se notar que a razão velocidade de codificação entre o *border* e a amostragem completa é de 2,215.

Das médias, foram realizadas análises estatísticas nos dados de DSSIM, PSNR e velocidade de codificação usando Análise de Variância - *Analysis of Variance* (ANOVA). Tal ferramenta estatística permite estudar a resposta experimental para buscar diferenças significantes. Para aumentar a confiança estatística, o *script* usado, *bootstrap.py* (PASCUTTO, 2011)⁹, realiza dez mil permutações aleatórias sobre os resultados, buscando por distribuições estatísticas robustas. Consequentemente, nas Tabelas 3 e 4 as células marcadas com ‘*’ indicam que não há evidência estatística suficiente para afirmar que um padrão é melhor ou igual a outro. Para as células sem ‘*’, a ANOVA resultou em um intervalo de confiança maior do que 95%. Enquanto se compara um grande número de taxas de bit, resoluções e amostras de vídeos com complexidades distintas, a análise de valores absolutos ou mesmo suas diferenças não se mostra prática. Por outro lado, os valores percentuais permitem relativização/normalização dos resultados, simplificando suas interpretações.

Como pode ser observado nas Tabelas 3 e 4, o padrão *border* teve resultados relevantes. Com sua razão de amostragem não convencional de 4:3, um aumento de velocidade de apenas 25% era esperado. Porém, os experimentos mostraram, com confiança, que ele é 2 vezes mais rápido que amostragem completa, perdendo apenas cerca de 4,8% em DSSIM e 0,7% em PSNR. O padrão *negeven*, por sua vez, é 3,3% mais rápido que *border* com o mesmo PSNR, mas com 0,3% menos DSSIM.

⁹De acordo com Hesterberg et al. (2003), “To bootstrap a statistic such as the sample mean, draw hundreds of resamples with replacement from a single original sample, calculate the statistics for each resample, and inspect the bootstrap distribution of the resampled statistic.”

Tabela 3: Comparação de DSSIM e velocidade codificação para otimização em SSIM

		Razão de qualidade DSSIM																
Razão de velocidade	-	-	1:1	4:3			2:1			4:1				16:3	8:1			
	PADRÃO	full	border	negeven	c. even	c. odd	cross	c. even	c.odd	kernel	main	4-queen	skew	triforce	c. even	sparse		
1:1	full	-	0,952	0,949	0,922	0,921	0,918	0,875	0,874	0,859	0,874	0,881	0,880	0,862	0,829	0,837		
4:3	border	2,215	-	0,997	0,969	0,968	0,964	0,919	0,918	0,902	0,918	0,925	0,924	0,906	0,871	0,879		
	negeven	2,288	1,033	-	0,971	0,971	0,966	0,922	0,921	0,905	0,921	0,928	0,927	0,908	0,874	0,882		
2:1	c. even	3,173	1,433	1,387	-	0,999*	0,995	0,949	0,948	0,932	0,948	0,955	0,954	0,935	0,899	0,908		
	c. odd	3,074	1,388	1,344	0,969	-	0,996	0,950	0,949	0,932	0,949	0,956	0,955	0,936	0,900	0,909		
4:1	cross	3,052	1,378	1,334	0,962	0,993	-	0,954	0,953	0,936	0,953	0,960	0,959	0,940	0,904	0,912		
	c. even	3,901	1,761	1,705	1,230	1,269	1,278	-	0,999*	0,982	0,999*	1,007	1,006*	0,985	0,948	0,957		
	c. odd	3,870	1,747	1,692	1,220	1,259	1,268	0,992	-	0,983	1,000*	1,008	1,007*	0,986	0,949	0,958		
	kernel	3,848	1,737	1,682	1,213	1,252	1,261	0,986	0,994*	-	1,017	1,025	1,024	1,004	0,965	0,975		
	main	3,828	1,728	1,673	1,206	1,245	1,254	0,981	0,989	0,995	-	1,008	1,007	0,987	0,949	0,958		
	4-queen	3,514	1,587	1,536	1,108	1,143	1,152	0,901	0,908	0,913	0,918	-	0,999*	0,979	0,941	0,950		
	skew	3,887	1,755	1,699	1,225	1,264	1,274	0,996*	1,004*	1,010	1,015	1,106	-	0,980	0,942	0,951		
16:3	triforce	4,063	1,835	1,776	1,281	1,322	1,331	1,041	1,050	1,056	1,062	1,156	1,045	-	0,962	0,971		
8:1	c. even	4,387	1,981	1,918	1,383	1,427	1,438	1,125	1,134	1,140	1,146	1,248	1,129	1,080	-	1,009		
	sparse	4,342	1,960	1,898	1,368	1,412	1,423	1,113	1,122	1,128	1,134	1,235	1,117	1,069	0,990*	-		

Tabela 4: Comparação de PSNR e velocidade codificação para otimização em PSNR

		Razão de qualidade PSNR																
Razão de velocidade	-	-	1:1	4:3			2:1			4:1				16:3	8:1			
	PADRÃO	full	border	negeven	c. even	c. odd	cross	c. even	c.odd	kernel	main	4-queen	skew	triforce	c. even	sparse		
1:1	full	-	0,993	0,993	0,986	0,986	0,985	0,973	0,974	0,972	0,974	0,975	0,974	0,969	0,961	0,963		
4:3	border	2,122	-	1,000*	0,993	0,992	0,992	0,980	0,981	0,978	0,981	0,981	0,981	0,976	0,968	0,970		
	negeven	2,193	1,033	-	0,992	0,992	0,992	0,980	0,981	0,978	0,981	0,981	0,981	0,976	0,968	0,970		
2:1	c. even	3,061	1,442	1,396	-	1,000*	0,999	0,987	0,988	0,986	0,988	0,989	0,988	0,983	0,975	0,977		
	c. odd	2,972	1,400	1,355	0,971	-	1,000	0,988	0,988	0,986	0,988	0,989	0,989	0,983	0,975	0,977		
4:1	cross	2,951	1,390	1,345	0,964	0,993	-	0,988	0,989	0,986	0,989	0,989	0,989	0,984	0,976	0,978		
	c. even	3,807	1,794	1,736	1,244	1,281	1,290	-	1,001	0,998	1,001	1,001	1,001	0,996	0,988	0,990		
	c. odd	3,772	1,777	1,720	1,232	1,269	1,279	0,991	-	0,998	1,000*	1,001	1,000	0,995	0,987	0,989		
	kernel	3,752	1,768	1,711	1,226	1,263	1,272	0,986	0,995*	-	1,002	1,003	1,003	0,997	0,989	0,991		
	main	3,734	1,760	1,703	1,220	1,257	1,266	0,981	0,990	0,995*	-	1,001	1,000*	0,995	0,987	0,989		
	4-queen	3,418	1,610	1,558	1,116	1,150	1,158	0,898	0,906	0,911	0,915	-	1,000*	0,994	0,986	0,988		
	skew	3,788	1,785	1,727	1,237	1,275	1,284	0,995*	1,004*	1,009	1,014	1,108	-	0,995	0,987	0,988		
16:3	triforce	3,966	1,869	1,808	1,296	1,334	1,344	1,042	1,051	1,057	1,062	1,160	1,047	-	0,992	0,994		
8:1	c. even	4,275	2,014	1,949	1,397	1,439	1,449	1,123	1,133	1,139	1,145	1,251	1,129	1,078	-	1,002		
	sparse	4,234	1,995	1,930	1,383	1,425	1,435	1,112	1,122	1,128	1,134	1,239	1,118	1,068	0,990*	-		

Células marcadas com '*' não atingiram o nível de confiança estatística de 95%.

Dentre todos os padrões 2:1, o *checker even* é o mais rápido enquanto o padrão *cross* é o pior em ambos, velocidade de codificação e qualidade. O ganho na velocidade de codificação do *checker even* 2:1 sobre a amostragem completa é, em média, 211,7%, enquanto a perda de qualidade é de 7,8% em DSSIM e 1,4% em PSNR. O ganho de desempenho do *checker even* sobre o *border* é 43,75% em média, enquanto a perda de qualidade é 3,1% em DSSIM e 0,7% em PSNR. Para todos os padrões com razões de subamostragem menores do que 2:1, as perdas de qualidade aumentam para um dado aumento de velocidade.

Os padrões 4:1 apresentam resultados de qualidade bem similares, variando, no máximo, 2,5% em DSSIM. Com respeito à velocidade de codificação, o *4-Queen* é aproximadamente 10,5% mais lento que qualquer outro padrão 4:1. Tal comportamento também é observado na Figura 18(b). É importante notar que, para padrões 4:1, algumas comparações não demonstram evidência estatística. Além disso, o padrão *kernel* tem os piores resultados de qualidade entre todos os padrões 4:1. Comparado com *triforce* (que amostra um pixel a menos), *kernel* é 0,4% pior em DSSIM, mas surpreendentemente, é 0,3% melhor em PSNR. Por outro lado, *triforce* é 5,7% mais rápido que o *kernel*.

Os padrões 8:1 são os mais rápidos, sendo quase 4,4 vezes mais rápidos que a amostragem completa, com perdas aproximadas de 17% em DSSIM e 4% em PSNR. Os padrões 8:1 são 2 vezes mais rápidos que o *border*, porém exibem perdas de aproximadamente 13% em DSSIM e 3% em PSNR. Em média, o padrão mais rápido foi o *checker even* 8:1 (como pode ser notado na Figura 18(b)), mas a comparação com *sparse diamond* não obteve confiança estatística para a velocidade. A comparação de qualidade mostra que o *sparse diamond* resulta em melhora de 0,9% em DSSIM e 0,2% em PSNR que o *checker even* 8:1. Em uma comparação direta entre os padrões 8:1 e *border*, o último resulta em uma relação melhor entre qualidade e velocidade.

A Figura 19 apresenta dois Grafos Acíclicos Direcionados - *Direct Acyclic Graphs* (DAGs) que resumem as comparações entre todos os padrões de subamostragem, de acordo com os resultados da ANOVA. Cada DAG é um conjunto $G(V, E)$, onde $V = \{p \mid p \text{ é um padrão de subamostragem}\}$ é um vértice e $E = \{(p1, p2) \mid p1 \text{ apresentam melhores resultados}^{10} \text{ que } p2\}$ é uma aresta. Os grupos (*clusters*) representam as razões de subamostragem. A duplicidade das arestas deve-se ao fato da análise de duas métricas de qualidade. Assim, cada DAG na Figura 19 representa, na realidade, a junção de um par de grafos de mesmos vértices, mas com dois conjuntos distintos de arestas: um relativo aos resultados de PSNR (arestas claras), e outro relativo

¹⁰Resultados em termos de qualidade ou velocidade de codificação com $\alpha = 0,05$.

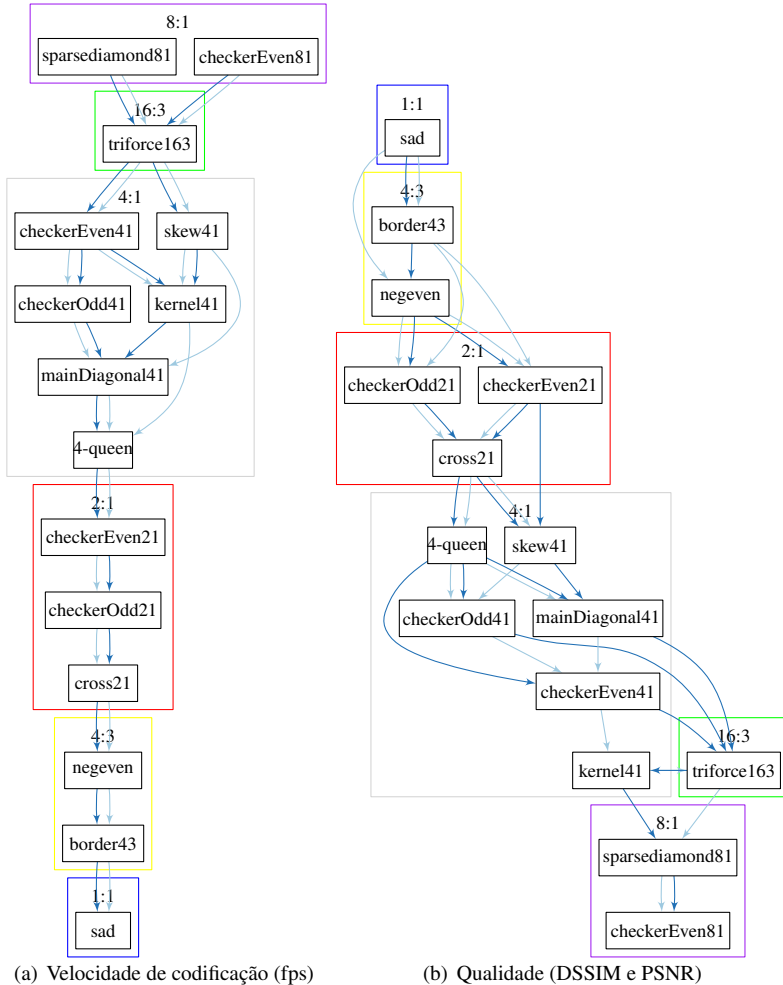


Figura 19: Padrões ordenados por seus resultados com codificações otimizadas em termos de SSIM (arestas escuras) e PSNR (arestas claras).

aos resultados de DSSIM (arestas escuras). Os DAGs das Figuras 19(a) e 19(b) não tem arestas transitivas e reforçam a ideia de ordenamento entre padrões, embora tal ordenamento seja muito mais claro para velocidade do que para qualidade. Este mesmo comportamento pôde ser observado na Figura 17.

3.5 CONCLUSÕES PARCIAIS

Sete padrões de subamostragem para *pel decimation* foram propostos e analisados em conjunto com outros sete padrões comuns de subamostragem e a amostragem completa. Os resultados obtidos mostraram, com confiança estatística, que os padrões de subamostragem mais eficientes são os propostos 4:3, border e negeven. Eles resultam em um aumento de velocidade de cerca de 2 vezes sobre a amostragem completa, com perdas aproximadamente de apenas 4,95% em DSSIM e 0,7% em PSNR.

Com respeito à metodologia para produzir as comparações consistentes, as taxas de bit sem perdas foram filtradas e os resultados foram agrupados por resolução e por complexidade do vídeo, uma vez que não tem sentido realizar uma codificação em vídeo sem redundâncias com uma taxa de bit maior do que sem perdas. Ambas estratégias de agrupamento permitem filtrar distorções indesejadas. Por outro lado, análises estatísticas tornaram possível agrupar todos os resultados e obter nível de confiança sobre eles.

Apesar de algumas comparações não terem obtido confiança, as restantes formaram uma quantidade significativa de dados que permitem estabelecer uma grande comparação entre os padrões de subamostragem, que é resumido didaticamente pela Figura 19. Observou-se que padrões de mesma razão de subamostragem tem pequenas diferenças de qualidade mas devem produzir diferenças significantes na velocidade de codificação. A justificativa para tal diferença na velocidade de codificação é apresentada no Capítulo 5, onde é descrito o efeito de *pel decimation* do algoritmo de ME utilizado nos testes apresentados neste capítulo (SEA).

Finalmente, os resultados apresentados confirmaram a expectativa de que para resoluções maiores (do mesmo vídeo), maiores razões de subamostragem causam perdas negligenciáveis na qualidade do vídeo. Apesar desta conclusão ser intuitiva e parecer óbvia, até onde se tem conhecimento, este é o primeiro trabalho que confirma tal conclusão por meio de extensiva e rigorosa experimentação utilizando amostras de vídeos de alta resolução em um padrão de codificação estado da arte, como o H.264/AVC.

4 ARQUITETURAS DEDICADAS AO CÁLCULO DA SAD

Conforme mencionado nos Capítulos 1 e 2, a Estimação de Movimento - *Motion Estimation* (ME) é a etapa da codificação de vídeo que demanda maior esforço computacional. Logo, é também a principal responsável pelos gastos energéticos. Durante a ME, uma métrica de similaridade é utilizada para encontrar o melhor casamento entre o bloco que está sendo codificado e os blocos candidatos, operação esta conhecida por *Block Matching* (BM). Dentre as métricas de similaridade existentes, a Soma das Diferenças Absolutas - *Sum of Absolute Differences* (SAD), definida na Subseção 2.4.3.1, é a mais utilizada para implementações de codificadores em *hardware*. Desta forma, decidiu-se projetar, sintetizar e avaliar arquiteturas de *hardware* dedicadas ao cálculo da SAD entre dois blocos de 4x4 pixels, com o objetivo de permitir uma análise mais realista do impacto de *pel decimation* no consumo de energia da codificação de vídeos de alta resolução. Em especial, propôs-se uma arquitetura capaz de ser configurada para calcular a SAD 4x4 com as seguintes razões de amostragem: 1:1, 4:3, 2:1 e 4:1. Os detalhes desta arquitetura, incluindo os resultados de sua síntese e simulação, são descritos no presente capítulo. Para fins de comparação com trabalhos correlatos, também foram sintetizadas e avaliadas quatro variações de uma arquitetura paralela não-configurável, dedicadas a cada uma das quatro razões de amostragem citadas anteriormente. Esta arquitetura paralela é equivalente àquela reportada no trabalho de Walter, Diniz e Bampi (2012) como sendo a de maior eficiência energética (*datapath* totalmente combinacional, sem *pipeline*). Os resultados das sínteses e simulações destas quatro variações arquiteturais também são apresentados neste capítulo, juntamente com comparações com a arquitetura configurável proposta. Adicionalmente, para obter-se uma visão mais abrangente dos impactos da subamostragem, são apresentadas nos Apêndices A e B explorações do espaço de síntese para a arquitetura proposta, envolvendo tecnologias (TSMC 130nm, 90nm, 65nm e 45nm), tensões de alimentação e de *threshold* e frequência máxima de operação.

Este capítulo está organizado como segue. A Seção 4.1 apresenta a revisão dos trabalhos correlatos no que se refere às arquiteturas para a ME e às arquiteturas de SAD. A Seção 4.2 apresenta a arquitetura configurável proposta, justificando as decisões arquiteturais de acordo com o apresentado na seção anterior. Na Seção 4.3 são apresentados os resultados da síntese da arquitetura configurável. Faz-se também a comparação entre os resultados da síntese e da simulação da arquitetura configurável proposta com os resultados das sínteses e das simulações das quatro variações da arquitetura não-

configurável. Com o intuito de estabelecer uma comparação justa com um trabalho correlato, aqui representado pela arquitetura não-configurável, todas as arquiteturas foram sintetizadas com o mesmo fluxo, ou seja, mesma ferramenta de síntese, mesma biblioteca *standard cell* e sujeitas às mesmas restrições. Além disso, todas as simulações utilizaram vetores realistas, obtidos a partir da codificação de um vídeo 1080p, o que permitiu obter-se estimativas de potência e de energia bastante precisas, as quais também são apresentados na Seção 4.3. Por fim, a Seção 4.4 apresenta as conclusões parciais deste capítulo.

A proposta da arquitetura configurável, apresentada na Seção 4.2, sua síntese lógica para biblioteca *standard cell* 65nm e a comparação com a síntese de uma versão não-configurável, com mesma Máquina de Estados Finitos - *Finite State Machine* (FSM) e *datapath*, foram publicadas como artigo nos anais do LASCAS¹ 2013 (SEIDEL; MORAES; GÜNTZEL, 2013) e estão descritos no Apêndice A. Uma primeira comparação entre tecnologias foi publicada no SIM² 2013 (SEIDEL et al., 2013a). A análise e comparação com diversas tecnologias, bem como o impacto do uso da técnica *Low-Vdd/High-Vt* e uso de frequência máxima ou alvo, foram publicadas também como artigo nos anais do SBCCI³ 2013 (SEIDEL et al., 2013b). Tais comparações estão descritas com detalhes no Apêndice B. Por fim, a descrição da arquitetura configurável para as razões de amostragem 1:1, 4:3, 2:1 e 4:1, a análise dos resultados de potência usando simulação com vetores realistas, e uma primeira comparação da relação custo-benefício entre qualidade e energia do uso de *pel decimation* foi aceito para publicação nos anais do LASCAS 2014 (SEIDEL et al., 2014).

4.1 TRABALHOS CORRELATOS

4.1.1 Arquiteturas para ME

Huang et al. (2003) propõem uma variação⁴ do algoritmo para Estimção de Movimento para Blocos de Tamanho Variável - *Variable Block Size Motion Estimation* (VBSME), visando implementações em *hardware*. Primeiramente, aplicam o algoritmo proposto no software de referência do H.264/AVC (JM) e então, baseados no algoritmo proposto, apresentam uma

¹IEEE Latin American Symposium on Circuits and Systems.

²Simpósio Sul de Microeletrônica.

³Symposium on Integrated Circuits and Systems Design.

⁴Variação em relação ao disponível no *software* de referência do H.264/AVC (*Joint Model* (JM)).

nova arquitetura de VBSME, sendo o *Fullsearch Block Matching Algorithm* (FBMA) usado para a busca. A proposta utiliza cálculos de SAD de blocos menores para compor os resultados para os blocos com tamanhos maiores. A arquitetura foi implementada para biblioteca *standard cells* em tecnologia 350nm. São reportados os resultados de área, número de portas (*gate count*), frequência e também potência: 737,32mW@66,67MHz.

Saponara e Fanucci (2004) apresentam outra arquitetura para FBMA onde a ideia chave é a alteração do tamanho da área de busca de acordo com estatísticas dos Vetores de Movimento - *Motion Vectors* (MVs) e das SADs que já foram calculados. Tal arquitetura foi sintetizada para biblioteca 250nm. As frequências atingidas são apresentadas de acordo com resoluções de vídeo, sendo as resoluções QCIF, CIF e 4CIF⁵ e as respectivas frequências obtidas foram 18,07MHz, 72,28MHz e 289,12MHz. Também neste trabalho, foi utilizada a técnica *low-power* chamada *clock gating*. Com a adaptação automática da área de busca e *clock gating*, a arquitetura proposta por Saponara e Fanucci (2004) obtém reduções de potência entre 70% e 90%, tomando como referencial a implementação de FBMA sem ambas as técnicas.

Yap e McCanny (2004) propõem uma nova arquitetura para VBSME. Embora usem como base o FBMA, alegam que a arquitetura proposta é facilmente adaptável para outros algoritmos de busca, como por exemplo, o *Three-Step Search* (TSS). Assim como no trabalho de Huang et al. (2003), as SADs já calculadas para blocos menores são posteriormente utilizadas para compor as SADs de blocos maiores. A arquitetura proposta é sintetizada para tecnologia 130nm. A frequência máxima atingida é de 294MHz. Para um vídeo na resolução QCIF à 30 Quadros por Segundo - *Frames per Second* (fps), a potência apresentada é 23,76mW. Mas deve-se considerar que, ao operar com maior frequência, podem ser processados até 724fps e a potência aumenta para aproximadamente 573,4mW.

Miyakoshi et al. (2005) definem quatro requisitos para implementações *low-power* flexíveis para FBMA. São eles:

1. Redução do número de ciclos de carga de dados e cálculo da SAD;
2. Redução do número de ciclos para acesso à memória;
3. Adaptabilidade para diversas ferramentas de codificação, como por exemplo, VBSME;
4. Operação concorrente com *half-pel*.

Assim definidos, tais requisitos são utilizados em sua análise de trabalhos correlatos, aos quais se referem como arquiteturas convencionais. Os

⁵QCIF = 176x144 pixels; CIF = 352x288 pixels e 4CIF = 704x576 pixels.

trabalhos de Jehng, Chen e Chiueh (1993) e Uramoto et al. (1994) apenas cumprem o primeiro e terceiro requisitos. Já o trabalho de Minami et al. (1995) cumpre apenas o segundo. Dessa forma, Miyakoshi et al. (2005) propõem sua arquitetura de forma a cobrir todos os quatro requisitos definidos. A arquitetura proposta é sintetizada, em conjunto com um codificador *Moving Pictures Expert Group - 2* (MPEG-2), para tecnologia 180nm e atinge reduções de até 73% em potência, ao ser comparado com os resultados da arquitetura em árvore proposta por Jehng, Chen e Chiueh (1993).

Chen et al. (2006a) analisam diversas outras arquiteturas da literatura e propõem duas novas arquiteturas, também para FBMA. Além disso, comparam o impacto de VBSME *versus* Estimação de Movimento para Blocos de Tamanho Fixo - *Fixed Block-Size Motion Estimation* (FBSME). Em suas arquiteturas propostas, além de diminuir a precisão de representação dos pixels, fazem também uso de *pel decimation 2:1*, usando o padrão *Quincunx* (ver Figura 4(a)). Tais alterações são apresentadas como forma de redução da área total do circuito. Embora sejam feitas sínteses para tecnologia 180nm, apenas os resultados de número de portas e frequência são demonstrados.

Os autores Chen et al. (2006b) apresentam uma arquitetura de codificador 720p H.264/AVC (ITU-T, 2009) na qual o cálculo da SAD corresponde a 33% do número total de portas, mesmo aplicando *pel decimation 4:1*. Já Liu et al. (2007) descrevem uma arquitetura de ME para VBSME em que a SAD corresponde a 79% do total de portas. Os trabalhos de Vanne et al. (2006) e de Yufei, Xiubo e Qin (2007) apresentam ainda outras arquiteturas para o cálculo de SAD com alta performance.

Larkin, Muresan e O'Connor (2006) fazem uso de limiares (*thresholds*) para o término prematuro (*early-termination*) da SAD. Assim, quando o valor em cálculo da SAD ultrapassa determinado *threshold*, o cálculo atual para e um novo inicia. Além disso, também aplicam *pel decimation 4:1*. A arquitetura apresentada por Kao, Wu e Lin (2010) também usa *thresholds* para *early-termination* do cálculo da SAD. Ambos implementam arquiteturas para FBMA.

Já Fatemi, Ates e Salleh (2009) utilizam a mesma arquitetura proposta por Huang et al. (2003) para VBSME em FBMA, modificando a estrutura dos somadores utilizados para cálculo da SAD. A novidade aqui é o uso de somadores bit-seriais. Com tais alterações, e também outra biblioteca *standard cells* 180nm, conseguem atingir uma redução de 31% no total de portas e aumento na frequência máxima de 66,67MHz para 440MHz.

Huang et al. (2009) propõem o uso de *pel decimation*, adaptativamente. Os padrões de subamostragem aplicados são apresentados na Figura 20. Os autores executam uma breve análise de qualidade do algoritmo proposto, utilizando *Bjontegaard Delta PSNR* (BDPSNR) e *Bjontegaard Delta*

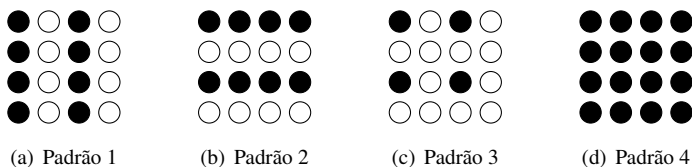


Figura 20: Padrões de amostragem adotados na arquitetura de ME configurável apresentada por Huang et al. (2009).

Bitrate (BDBR) em quatro sequências Alta Definição - *High Definition* (HD) 720p. Os resultados apontam que, ao aplicar diretamente a subamostragem (neste ponto os autores não indicam a razão de subamostragem), os acréscimos em BDBR são maiores do que aplicando subamostragem adaptativa. Ao mesmo tempo, há maior decréscimo em termos de BDPSNR no caso de subsamostragem direta. A partir do algoritmo proposto, Huang et al. (2009) descrevem uma arquitetura de *hardware* configurável para maximizar o reuso de dados ao aplicar subamostragem adaptativa. A arquitetura é sintetizada com tecnologia Semicondutor Metal-Óxido Complementar - *Complementary Metal-Oxide Semiconductor* (CMOS) 180nm para duas frequências, 110,5MHz e 200MHz. Tomando a síntese para 110,5MHz como exemplo, o trabalho de Huang et al. (2009) demonstra aumento da área (número de portas), 53,5%, em relação ao trabalho de Chen et al. (2006a). Em contrapartida, há redução na potência, passando de 573mW para 304mW (aproximadamente 47%).

Olivares (2012) apresenta uma arquitetura para VBSME, reduzindo de forma configurável o número de *bits* por pixel. Ao ser aplicada no cálculo da SAD, esta recebe o nome de *Reduced Bit Sum of Absolute Differences* (RBSAD) e, de acordo com He et al. (2000) e Agha, Dwyer e Chouliaras (2005), apresenta degradação de 0,2% até 1% para redução de 2 *bits* e entre 1% e 2% para redução de 4 *bits*. A configurabilidade dá-se ao calcular a RBSAD no nível de *bit* ao invés de nível de pixel. A arquitetura proposta é sintetizada para *Field-Programmable Gate Array* (FPGA) e também *standard cells*, 130nm. No caso de FPGA, a frequência máxima foi de 380,1MHz. E embora os autores afirmem que a arquitetura proposta é perfeita para dispositivos móveis, os valores de frequência e potência para a síntese *standard cells* são omitidos, restando apenas o número de portas: 54Kgates.

Sanchez et al. (2012) apresentam uma arquitetura para *Fractional Motion Estimation* (FME), adotando a *Diamond Search* (DS) como algoritmo de busca. Foram considerados blocos de 8x8 pixels, sendo que a DS ocorre para as posições inteiras. Além da implementação da DS, também foi apresentado

o *hardware* necessário para gerar as amostras interpoladas para a busca fracionária, tanto para *half-pixel* quanto para *quarter-pixel*. A arquitetura integrada foi sintetizada para FPGA, atingindo a frequência máxima de 260MHz, a qual, de acordo com os autores, ultrapassa o necessário para codificação em tempo real de uma sequência 1080p@30fps.

Embora a arquitetura proposta por Niyas, Guru e Jayakrishnan (2013) não apresente grande inovação, seus resultados de síntese são interessantes, uma vez que usam tecnologia de 45nm. Além disso, apresentam resultados de área, frequência máxima (176,05MHz) e potência (1,03mW). A arquitetura proposta é composta por uma árvore de comparadores, com o propósito de identificar o menor valor de SAD em suas entradas. Tais entradas recebem os valores de SAD calculados par-a-par para cada bloco 4x4, de forma sequencial.

Como visto até aqui, há diversas implementações de *hardware* para a FBMA. De fato, e também de acordo com Porto et al. (2010), a FBMA é o algoritmo de BM mais utilizado para implementações em *hardware* da ME devido sua regularidade, fácil paralelização e boa performance. O fato de a FBMA ser indicada para implementações em *hardware* da ME devido ao fluxo regular também é mencionado por Ghanbari (2003). Outro motivo mencionado por tal autor é que a FBMA sempre obtém o resultado ótimo afinal, esta faz a busca entre todos os candidatos. Ainda assim, outros algoritmos de busca são utilizados para a redução de área, tempo de execução e potência. Este é o caso do trabalho de Porto et al. (2010), onde o algoritmo utilizado é o *Quarter Sub-sampled Diamond Search algorithm with Dynamic Iteration Control* (QSDS-DIC), proposto por Porto et al. (2008). Outros algoritmos de busca rápidos foram apresentados na Subseção 2.4.2.

4.1.2 Arquiteturas Dedicadas ao Cálculo da SAD

O trabalho de Vassiliadis et al. (1998) faz uma análise de três métricas de similaridade, a *Mean Squared Error* (MSE), a *Mean Absolute Difference* (MAD) e a SAD. Terminada a análise, é proposta uma arquitetura para o cálculo da SAD com tamanho de bloco 16x1 (uma linha ou coluna de um bloco 16x16, para o padrão *Moving Picture Experts Group* (MPEG)), dividindo-a em etapas:

1. Determinação do menor de dois operandos;
2. Inversão do menor dos dois operandos e entrega destes para um conjunto de somadores;

3. Adição de um termo de correção⁶;
4. Redução das somas: dos 32 operandos e termo de correção para apenas 2 valores;
5. Redução final da soma, resultando na SAD 16x1.

Estes passos podem ser executados em paralelo para 16 grupos, executando a SAD de um bloco 16x16 apenas com mais uma redução de 32 valores (os 2 resultantes de cada um dos 16 elementos, após a etapa 4) para apenas dois, antes da redução final (etapa 5). Ou seja, antes da etapa 5, repete-se a etapa 4, ignorando o termo de correção. Vassiliadis et al. (1998) dividem o cálculo da SAD em quatro ciclos de relógio: agrupando as etapas 1 e 2 no primeiro ciclo; 3 e 4 no segundo; o terceiro é para a segunda redução da etapa 4 e o quarto ciclo é a redução final, ou seja, a etapa 5.

Outra análise de arquiteturas de SAD é apresentada de forma incremental por Walter, Diniz e Bampi (2011), Walter e Bampi (2011) e Walter, Diniz e Bampi (2012). Estes autores sintetizaram arquiteturas de SAD com diversas configurações de paralelismo e *pipeline*. Com o objetivo de reduzir ainda mais a energia gasta por cálculo de SAD, os autores também sintetizaram as arquiteturas para frequências alvo. Essas frequências alvo foram escolhidas buscando um *throughput* de um milhão de macroblocos/s que, de acordo com os autores, é o necessário para codificar em tempo real um vídeo 1080p@30fps. Eles demonstraram que a arquitetura com 4x4 pixels (16 entradas), com mais alto grau de paralelismo e menos estágios de pipeline, apresenta maior eficiência energética. Também de acordo com os autores, os registradores de pipeline são os maiores contribuintes para maior potência.

O trabalho de Archana e Devi (2013) apresenta algo bem diferente e exótico: o uso de portas reversíveis⁷ para implementação do cálculo da SAD. Utiliza os mesmos somadores compressores 4:2 reversíveis apresentados por Krishnaveni, Priya e Baskaran (2012). A arquitetura de SAD proposta foi simulada e sua área e potência foram obtidas através da ferramenta *Synopsys Design Compiler*⁸. A área reportada é de 7.398 unidades (os autores não deixam claro qual unidade é utilizada). A potência reportada foi de 5,1mW.

Manjunatha e Sainarayanan (2013) apresentam uma análise de implementações de SAD, explorando para isso alguns tipos de somadores:

⁶Tal termo de correção deve ser adicionado pois o menor operador, no passo anterior, foi apenas invertido. Este termo de correção corresponde ao passo de adicionar um, quando usando complemento de dois.

⁷De acordo com Garipelly, Kiran e Kumar (2013), além de formarem os blocos básicos para computação quântica, os circuitos reversíveis são úteis para computação *low-power*, *Digital Signal Processing* (DSP), computação gráfica, entre outros.

⁸À qual os autores chamam de Design Vision, que na realidade é a interface gráfica do Design Compiler.

Carry-Ripple Adder (CRA), *Carry-Lookahead Adder (CLA)* e *Carry-Select Adder (CSA)*. Três implementações de SAD 4x4 e três 8x8 são sintetizadas utilizando tecnologia CMOS de 180nm, cada par usando um dentre os mencionados somadores. São apresentados resultados de atraso, área e potência. Para a implementação de SAD 4x4, a opção que faz uso de CRA foi a que apresentou menor área ($5.492\mu\text{m}^2$) e menor potência ($180,68\mu\text{W}$). Já no caso da SAD 8x8, os resultados demonstraram que a opção que faz uso de CSA foi a que apresentou menor área ($12.999\mu\text{m}^2$) e menor potência ($1.117,54\mu\text{W}$). Um trabalho similar foi feito por Seidel, Moraes e Güntzel (2012), onde avaliou-se o impacto dos somadores na implementação de arquiteturas de SAD utilizando *pel decimation*. Neste último, não foi feita a síntese das arquiteturas: os valores de atraso, área, potência e eficiência energética foram derivados das sínteses dos somadores feitas por Monteiro (2011).

4.1.3 Resumo dos Trabalhos Correlatos Apresentados

A Tabela 5 apresenta, resumidamente, a maioria dos trabalhos apresentados até aqui. Os trabalhos são listados por ordem de publicação. Cada linha da tabela indica um trabalho, o(s) algoritmo(s) implementado(s), se há uso de *pel decimation* e qual sua proporção, o tipo de síntese (FPGA ou *standard cells*), e quais são os resultados apresentados.

É possível perceber que os únicos trabalhos que levam em consideração estimativas de energia são os de Seidel, Moraes e Güntzel (2012) e Walter, Diniz e Bampi (2012). Este último, apresentando análise extensiva de arquiteturas de SAD, será utilizado na Subseção 4.3.3 para comparação com a arquitetura configurável apresentada na Seção 4.2. Dentre os 20 trabalhos que apresentam resultados, seis sequer consideram resultados de potência, mesmo obtendo os demais resultados através de síntese *standard cell*.

Tabela 5: Resumo dos trabalhos correlatos, organizados por data de publicação.

Trabalho	Algoritmo	<i>Pel decimation</i>	FPGA ou <i>standard cell</i>	Resultados
(VASSILIADIS et al., 1998)	SAD	Não	N.A.	N.A.
(HUANG et al., 2003)	FBMA, VBSME	Não	<i>standard cell</i> 350nm	A, F, G e P.
(SAPONARA; FANUCCI, 2004)	FBMA	Não	<i>standard cell</i> 250nm	F, G e P.
(YAP; MCCANNY, 2004)	FBMA, VBSME	Não	<i>standard cell</i> 130nm	F, G e P
(MIYAKOSHI et al., 2005)	FBMA, VBSME	Não	<i>standard cell</i> 180nm	A e P
(CHEN et al., 2006a)	FBMA, VBSME, FBSME	Não	<i>standard cell</i> 180nm	F e G.
(CHEN et al., 2006b)	Codificador H.264/AVC	Sim 4:1	<i>standard cell</i> 180nm	A, F, G e P
(LARKIN; MURESAN; O'CONNOR, 2006)	FBMA	Sim, 4:1	FPGA e <i>standard cell</i> 90nm	A, F e P
(VANNE et al., 2006)	SAD	Não	<i>standard cell</i> 180nm	F e G
(LIU et al., 2007)	VBSME	Sim, 4:1	<i>standard cell</i> 180nm	F, G e P
(YUFEI; XIUBO; QIN, 2007)	SAD	Não	N.A.	A e D
(FATEMI; ATES; SALLEH, 2009)	FBMA, VBSME	Não	<i>standard cell</i> 180nm	G e F.
(HUANG et al., 2009)	<i>Adaptive pel</i>	Sim, 2:1 e 4:1	<i>standard cell</i> 180nm	F, G e P
(PORTO et al., 2010)	QSDS-DIC	Sim, 4:1	FPGA e <i>standard cell</i> 180nm	F e G
(WALTER; DINIZ; BAMPI, 2012)	SAD	Não	<i>standard cell</i> 180nm	A, D, F, P e E
(OLIVARES, 2012)	VBSME e RBSAD	Não	FPGA e <i>standard cell</i> 130nm	F e G.
(SANCHEZ et al., 2012)	DS, FME	Não	FPGA	F.
(SEIDEL; MORAES; GÜNTZEL, 2012)	SAD	Sim, 4:3, 2:1 e 4:1	<i>standard cell</i> 45nm	A, D, P e E.
(ARCHANA; DEVI, 2013)	SAD	Não	<i>standard cell</i>	A e P
(MANJUNATHA; SAINARAYANAN, 2013)	SAD	Não	<i>standard cell</i> 180nm	A, D e P
(NIYAS; GURU; JAYAKRISHNAN, 2013)	SAD	Não	<i>standard cell</i> 45nm	A, F, G e P.

Considerando: **A** = Área; **D** = Atraso; **E** = Energia; **F** = Frequência; **G** = Número de portas; **N.A.** = Não aplicável; **P** = Potência .

4.2 APRESENTAÇÃO DA ARQUITETURA PROPOSTA

Como mostrado por Walter, Diniz e Bampi (2012), as arquiteturas mais eficientes energeticamente usam menos registradores e maiores níveis de paralelismo. Considerando um bloco 4x4 de pixels, a melhor escolha, em termos de eficiência energética, é a arquitetura totalmente combinacional, que processa 16 pixels de cada bloco em paralelo. Porém, no projeto de uma arquitetura configurável para *pel decimation* 2:1 e 4:1 existem duas resoluções menores de blocos a serem levadas em conta, além da amostragem completa. Uma com 8 pixels, para *pel decimation* 2:1, e outra com apenas quatro pixels, para 4:1. No último caso, há apenas quatro entradas paralelas de dados (para cada bloco) para maximizar a eficiência energética (minimizar o consumo de energia por operação). Sem outras exceções, a arquitetura com quatro entradas paralelas se torna a melhor para minimizar o consumo energético. Deve-se adicionar também à árvore combinacional de SAD um registrador de saída de 12 bits, juntamente com um somador extra, como mostrado na Figura 21. Em tal *datapath*, a configurabilidade é atingida através do encerramento antecipado do somatório da SAD. Semelhantemente, uma possível arquitetura não-configurável para SAD 4x4 pode ser obtida com o mesmo *datapath* e pequenas mudanças no controle. A comparação com tal arquitetura não-configurável é apresentada no Apêndice A.

A Figura 22 mostra a FSM da arquitetura de SAD configurável. Cada vez que o estado CALC é executado, quatro diferenças absolutas entre pixels são adicionadas. Assim, quando operando em amostragem completa (i.e., sem subamostragem) a SAD usando todos os 16 pixels de cada bloco requer quatro execuções do estado CALC. O sinal *zero* indica a quarta execução de CALC e, então, a FSM muda para o estado DONE. Dessa forma, os estados LOAD e DONE podem ser considerados dois estados de sincronização. Devido ao alto número de computações (como visto na Seção 1.1), a arquitetura de SAD requer muitos acessos à memória. Em alguns casos, em sínteses para alta frequência, pode ocorrer que a memória acoplada tenha tempo de acesso demasiado longo para que esta responda em apenas um ciclo de relógio da arquitetura. Nesses casos, a FSM continua no estado LOAD até receber um sinal *loaded*, o que a habilita a proceder para o estado CALC. Também no estado DONE é esperado um sinal de confirmação (*ack*) do controle da ME. Considerando uma única execução de cada um dos estados de sincronização, a redução de tempo de 50% e 75%, que seria alcançada usando *pel decimation* 2:1 e 4:1, cai para 40% e 60%, respectivamente. É importante notar que tais melhorias significativas ocorrem somente quando a FSM necessita de apenas um ciclo de relógio para os estados LOAD e DONE. Em resumo, quando se usa *pel decimation* 2:1, o aumento de velocidade em relação à SAD completa

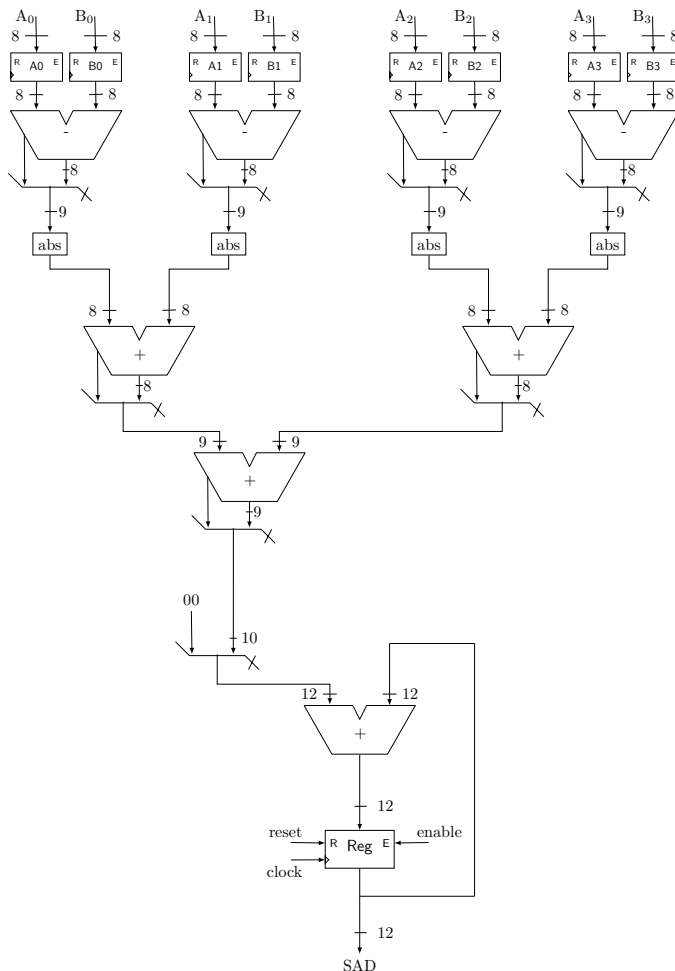


Figura 21: *Datapath* para a arquitetura de SAD projetada. Embora projetada desta forma, durante a síntese a ferramenta escolhida pode mudar a organização dos blocos básicos, enquanto mantém a funcionalidade. É importante notar que as entradas são recebidas quando o sinal *loaded* for válido, e o resultado será armazenado quando o registrador final receber o sinal *enable*. Dessa forma, a cada ciclo um novo valor está disponível na saída **SAD**. Entretanto, este só é válido quando o controle atinge o estado DONE (ver Figura 22). Neste estado a arquitetura envia um sinal externo chamado *done*, indicando para o algoritmo de controle da ME que o resultado da SAD disponível na saída é válido.

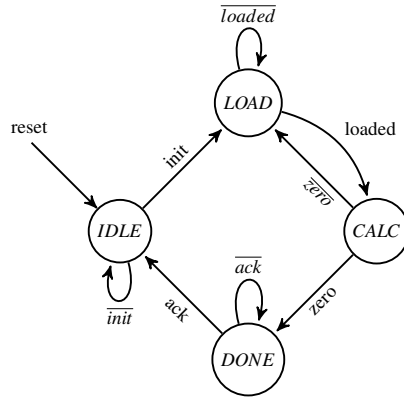


Figura 22: FSM para a arquitetura configurável proposta. A mesma FSM pode ser utilizada para uma versão de arquitetura não-configurável de SAD 4x4, sem subamostragem e com o mesmo *datapath* apresentado na Figura 21. Para tanto, o número de execuções do laço LOAD/CALC deve ser fixo em 4, perfazendo o cálculo de 4 vezes 4 pares de entradas. No caso da arquitetura configurável proposta, o número de vezes que tal laço deve ser executado é recebido através de uma entrada, conforme o nível de amostragem escolhida.

é de 40%. Já a razão 4:1 resulta em 33% de aumento de velocidade em relação à 2:1. Em comparação com amostragem completa, a razão 4:1 resulta em aumento de velocidade de 60%. Essas proporções são mantidas para eficiência energética. A Figura 23 apresenta a relação de sinais dos blocos operativo (*datapath*) e bloco de controle (o qual implementa a FSM apresentada na Figura 22). Sendo que os sinais que controlam o *datapath*, bem como os sinais de saída, são apresentados na Tabela 6.

Como mencionado anteriormente, ao se projetar um sistema com baixo consumo de energia, é imprescindível levar em conta os tempos de acesso à memória utilizada. Os trabalhos de Zhang et al. (2003), Fukano et al. (2008) e Qazi et al. (2011) mostram exemplos de memórias atuais, as quais, mesmo em modo de baixo consumo de energia, têm tempos de acesso entre 0,4ns e 3,4ns. Também os gastos de energia destas variam bastante conforme seus tempos de acesso. Um dispositivo de amostragem deveria funcionar com, no mínimo, o dobro da frequência da árvore de SAD para maximizar a eficiência energética no caso da arquitetura proposta. Tal dispositivo seria encarregado de agregar os dados para o cálculo da SAD, de acordo com o padrão e taxa de subamostragem escolhida. Considerando o tempo de acesso de 3,4ns da memória de baixo consumo de energia (mais lenta dentre as apresentadas),

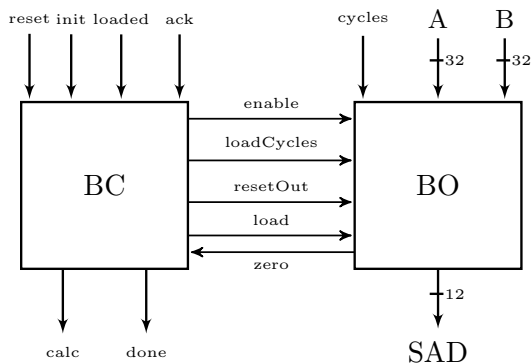


Figura 23: Diagrama do bloco de controle (BC) e bloco operativo (BO) para a arquitetura configurável. Os sinais da FSM apresentada na Figura 22 tem seus valores apresentados na Tabela 6. O sinal zero, saindo do BO, dirigido ao BC é da saída de um contador, o qual recebe o valor de *cycles* quando o sinal *loadCycles* estiver ativo. Através de tal contador obtém-se a configurabilidade, permitindo executar o número adequado de vezes o cálculo da SAD para 4 entradas.

a frequência recomendável para a arquitetura configurável proposta deve ser menor que 147MHz. Para atingir o *throughput* máximo, deve-se usar a memória mais rápida e assim, a frequência máxima da arquitetura configurável deveria ser menor do que 1,25GHz. Para sínteses com frequências superiores, outras memórias devem ser escolhidas.

4.3 SÍNTESE E SIMULAÇÃO COM VETORES REALISTAS

Esta seção apresenta um estudo sobre a relação custo-benefício entre energia e qualidade quando aplicada subamostragem. Foram obtidas estimativas realistas para área e energia por bloco, simulando cinco arquiteturas especialmente projetadas para computar a SAD para blocos de 4x4 pixels. Entre essas arquiteturas, uma pode ser configurada para operar com razões de amostragem de 1:1, 4:3, 2:1 ou 4:1, ao passo que as demais foram projetadas para operar exclusivamente com cada uma daquelas razões. Como mencionado no início deste capítulo, os resultados apresentados nesta seção foram aceitos para publicação no LASCAS 2014 (SEIDEL et al., 2014).

Tabela 6: Sinais da FSM por estado.

Estado	enable	load	done	loadCycles	calc
IDLE	0	0	0	1	1
LOAD	0	1	0	0	0
CALC	1	0	0	0	1
DONE	0	0	1	0	0

4.3.1 Arquiteturas Fixas para Comparação

A fim de obter estimativas realistas para energia por bloco e também para área em silício, foram projetadas quatro variações de uma arquitetura não-configurável para computar a SAD entre dois blocos de 4x4 pixels. A arquitetura configurável, apresentada na Seção 4.2, é capaz de computar a SAD usando todos os 4x4 pixels de entrada dos dois blocos (i.e., amostragem completa) ou aplicando qualquer uma das três taxas de subamostragem mostradas na Figura 4. Ao contrário, cada uma das variações de arquitetura não-configurável, referidas como arquiteturas fixas, foram projetadas para operar com apenas uma dentre as possíveis taxas de amostragem: 1:1, 4:3, 2:1 ou 4:1. Para o projeto das quatro arquiteturas fixas, foram empregados *datapaths* totalmente combinacionais, uma vez que tal topologia foi apontada por Walter, Diniz e Bampi (2012) como a mais eficiente energeticamente. Desta forma, o *datapath* de cada arquitetura fixa foi ajustado para operar com máximo paralelismo, considerando uma dentre as quatro razões de amostragem (1:1, 4:3, 2:1 ou 4:1), conforme apresentado na Figura 25.

Foram projetadas FSMs específicas para controlar os *datapaths* das arquiteturas fixas, conforme apresentado na Figura 24. A respeito da arquitetura configurável (Figura 24a), vale lembrar que a cada vez que o estado CALC é executado, quatro pixels de cada bloco (original e referência) são processados. Então, operando em modo de amostragem completa, a SAD de todos os 16 pixels requer quatro execuções do estado CALC. O sinal *zero* indica a quarta execução de CALC e a FSM muda o seu estado para DONE. Para as arquiteturas fixas (Figura 24b), o estado CALC sempre é executado apenas uma vez, e todos os pixels são processados, de acordo com a razão de subamostragem da arquitetura. Por isso, estados LOAD (arquitetura config) e DONE (ambas as arquiteturas) são estados de sincronização.

Devido ao grande número de computações de SAD a serem efetuadas em uma codificação (como visto na Figura 3 no Capítulo 1), tais arquiteturas

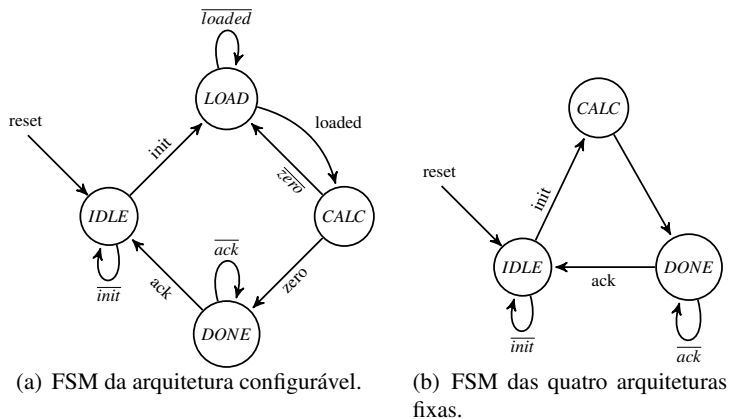


Figura 24: FSM para as arquiteturas configurável e fixas. A primeira apresenta sua configurabilidade através do número de execuções no laço formado pelos estados *LOAD* e *CALC*. As arquiteturas fixas calculam a SAD com uma subamostragem apropriada de uma só vez (paralelamente) durante o estado *CALC*. Dessa forma, a diferença entre as quatro arquiteturas fixas está apenas no *datapath*.

requerem muitos acessos à memória, os quais podem ter atrasos muito grandes para que sejam acomodados em um único ciclo de relógio. Nesses casos, para a arquitetura configurável, a FSM permanece no estado *LOAD* até que receba um sinal *loaded*, o qual a habilitará para proceder para o estado *CALC*. Isso não é necessário para as arquiteturas fixas: embora o estado *IDLE* seja equivalente, uma vez que a próxima SAD deva apenas começar (sinal *init*) após todos os pixels estarem disponíveis nas entradas. Em ambas FSMs, o estado *DONE* espera um sinal de confirmação (*ack*) do controle da ME.

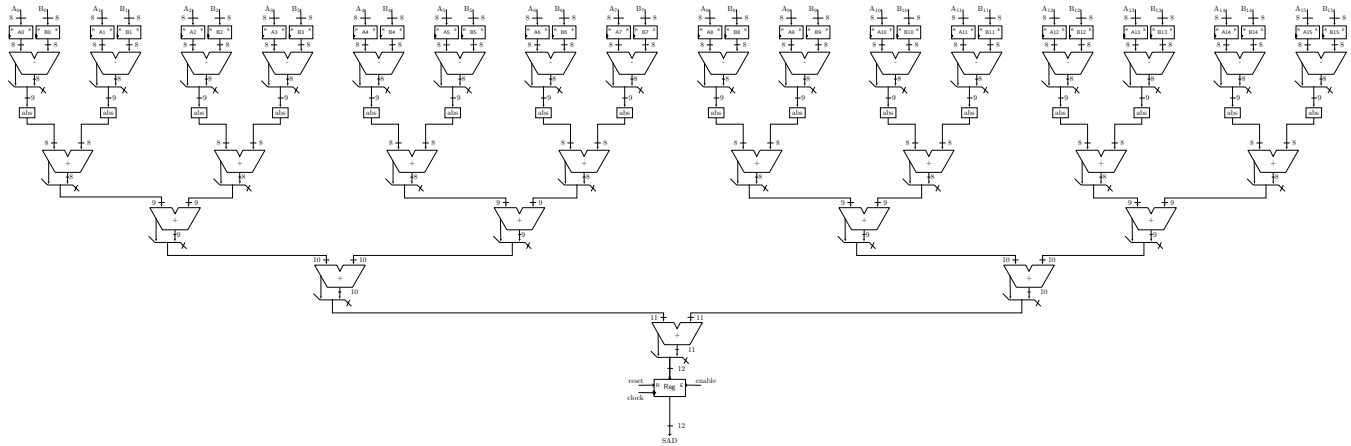


Figura 25: *Datapath* para a arquitetura fixa com amostragem completa. Conforme a FSM apresentada na Figura 24b, em apenas um ciclo (CALC), a SAD 4x4 é totalmente computada.

4.3.2 Configuração Experimental e Fluxo de Síntese

Todas arquiteturas foram descritas em Verilog e sintetizadas com a ferramenta Synopsys[®] Design Compiler (DC[®]) em modo *Topographical* para tensão nominal, biblioteca *standard cell* 45nm TSMC (2011a) para duas frequências operacionais, alvo (“tar”) e máxima (“max”), totalizando 10 sínteses. As frequências alvo visam o mesmo *throughput* assumido por Walter, Diniz e Bampi (2012): 1 milhão de macroblocos/s. A Tabela 7 lista as frequências máximas alcançadas e os respectivos *throughputs*. Para todas as sínteses, foram adotadas as seguintes restrições⁹:

- Os atrasos de entrada e saída foram conservadoramente limitados a 60% do período de relógio;
- A capacitância primária máxima de entrada foi ajustada para 10 vezes uma porta AND de duas entradas;
- A capacitância primária máxima de saída foi ajustada para 30 vezes uma porta AND de duas entradas.

O fluxo de síntese e simulação, ilustrado na Figura 26, foi adotado para obter os resultados apresentados nesta seção. Para cada arquitetura, foi descrito também um *testbench* em Verilog, tendo em mente dois objetivos: validar as arquiteturas e obter o arquivo de atividade de chaveamento (*Switching Activity Interchange Format* (SAIF)), para cada *netlist* sintetizada. Para gerar vetores realistas para simulação com Synopsys[®] VCS (VCS[®]) (SYNOPSYS, 2012), a sequência 1080p chamada “Pedestrian Area” (XIPH.ORG, 2011) foi codificada com o *x264* modificado de forma a criar um arquivo com os valores dos pixels e os resultados da métrica de similaridade correspondente. Foram selecionados deste os cálculos de SAD para blocos 4x4. Apesar de um total de 651.359.478 vetores SAD 4x4 terem sido gerados, apenas 100 mil vetores (pares de blocos 4x4) foram utilizados, para limitar o tempo de execução da simulação. Para cada arquitetura programou-se um tratador (*handler*) em linguagem C com *Verilog Procedural Interface* (VPI) (DAWSON; PATTANAM; ROBERTS, 1996), para fazer a interface entre o arquivo de vetores e o *testbench* em Verilog.

Juntamente com as descrições da arquitetura e as bibliotecas, as sínteses foram restringidas como apresentado na Subseção A.1. Foram obtidas es-

⁹Tais restrições são bastante conservadoras. Baseiam-se no curso de Synopsys[®] Design Compiler durante a Disciplina MIC73 - **Topics on IC Design Flow**, cursada no Programa de Pós-Graduação em Microeletrônica da Universidade Federal do Rio Grande do Sul e ministrada por instrutor da Synopsys[®].

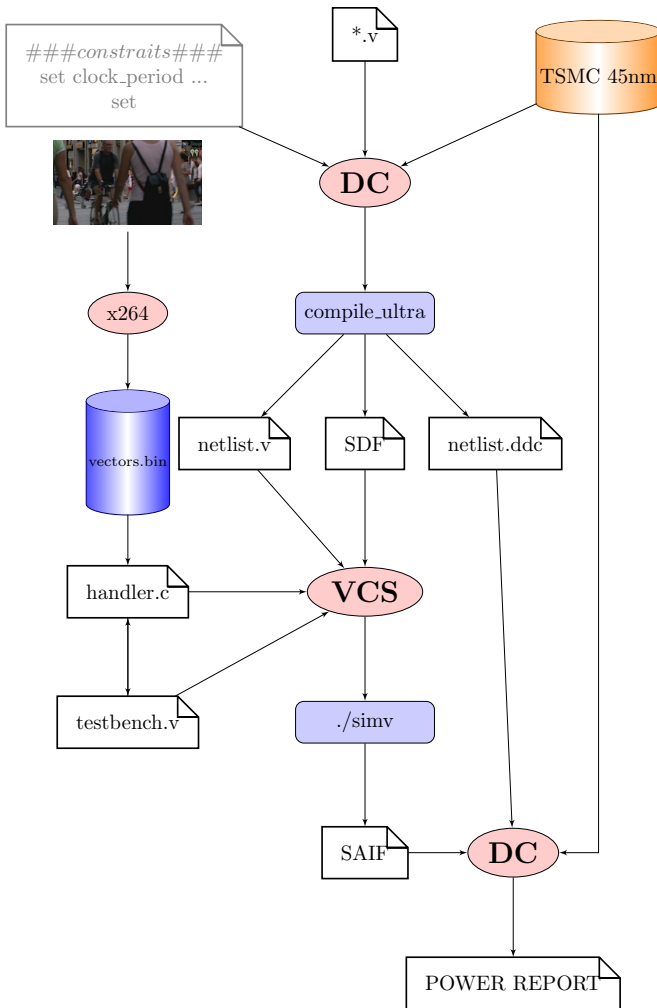


Figura 26: Fluxo de síntese e simulação executado para cada uma das arquiteturas projetadas. As simulações cobrem duas necessidades: validar a arquitetura descrita e sintetizada e ao mesmo tempo prover estatísticas realistas de atividade de chaveamento, permitindo dessa forma estimativas de potência mais precisas.

Tabela 7: Frequências máximas (GHz) e seus respectivos *throughputs* (*Mblocos/s*)

Subamostragem	Configurável	1:1	4:3	2:1	4:1
Frequência (GHz)	1,86	1,43	1,54	1,76	2,05
Throughput(<i>Mblocos/s</i>)	185,5~463,7	476,1	514,8	589,5	685,4

timativas de área a partir da síntese. Então, as *netlists* sintetizadas, os respectivos atrasos e os vetores de simulação são enviados para o VCS[®]. O VCS[®], simulador de Verilog da Synopsys[®], é um simulador compilado, possuindo boa velocidade de simulação (BRUNVAND, 2010). Após a simulação, foi enviado de volta o SAIF para o DC[®] e foram geradas estimativas de potência mais realistas pelo Synopsys[®] Power Compiler.

4.3.3 Resultados e Análise

A Figura 27 apresenta os resultados de área para as quatro arquiteturas fixas usando a arquitetura configurável como base. Também apresenta os valores absolutos de área para as duas versões da arquitetura configurável (freq. alvo e máxima). Basicamente, quanto mais pixels são usados por ciclo de relógio, mais somadores estão presentes no *datapath* e portanto, maior é o circuito. A configurabilidade é outra questão a ser considerada. Por um lado, como a arquitetura configurável precisa de um registrador extra e controle mais complexo, é esperado um ligeiro aumento de área, comparado à arquitetura fixa 4:1. Por outro lado, a arquitetura fixa 2:1 tem o dobro de pixels amostrados por ciclo de relógio e os resultados mostram que os custos de área ao ter *hardware* extra para adicionar mais quatro pares de pixels são maiores do que os custos de um registrador e um estado de controle a mais para a arquitetura configurável. O aumento médio de área das sínteses de frequência máxima em relação às sínteses de frequência alvo é em torno de 92%. Isso se deve ao fato que em sínteses para frequência máxima há menos espaço para otimizações de área, uma vez que as folgas temporais (*slacks*) a serem exploradas são menores.

A energia/bloco é calculada usando a Equação 4.1, onde T é o período, C é o número de ciclos necessários para computar um bloco de 4x4 pixels e P é a potência obtida pela simulação. Para sínteses de frequência alvo, T foi ajustado para 6,5ns para a arquitetura configurável e 20,83ns para todas as ar-

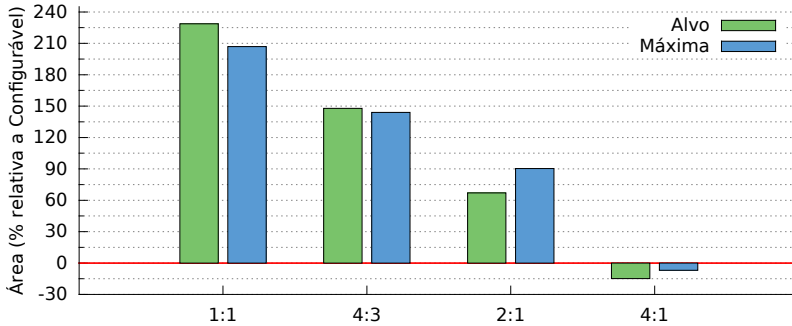


Figura 27: Resultados de área relativos à síntese da arquitetura configurável (%). Pode-se perceber que em ambas as sínteses, para frequência alvo e máxima, a área da arquitetura fixa 4:1 é menor do que a área da arquitetura configurável (o que é indicado por um valor relativo negativo). Os resultados para frequências alvo e máximas não devem ser diretamente comparados, uma vez que são derivados de valores absolutos diferentes.

quitetas fixas, uma vez que todas fixas tem a mesma FSM. Para sínteses de frequência máxima, cada uma das cinco arquiteturas tem um valor específico de T . Em condições ótimas (i.e. usando uma memória suficientemente rápida), os menores números de ciclos para a versão configurável são 10 (1:1), 8 (4:3), 6 (2:1) ou 4 (4:1), assumindo execução única dos estados LOAD e DONE. Assim, para uma determinada frequência (aqui as possíveis são máxima ou alvo), a potência estimada está relacionada diretamente às sínteses, e portanto as reduções de energia alcançadas por subamostragens 4:3, 2:1 e 4:1 com respeito a 1:1 são, respectivamente 20%, 40% e 60%. Mas no caso das arquiteturas configuráveis, C é constante (3 ciclos em condições ótimas, onde IDLE é executado apenas uma vez), e assim as estimativas de potência mudam de acordo com as simulações pós-síntese.

$$E = T \times C \times P \quad (4.1)$$

A Figura 28 demonstra o compromisso entre energia/bloco e qualidade de vídeo, em termos de Relação Sinal-Ruído de Pico - *Peak signal-to-noise ratio* (PSNR) e Índice de Dissimilaridade Estrutural - *Structural Dissimilarity Index* (DSSIM), para as arquiteturas consideradas. Primeiramente, é importante observar que enquanto cada arquitetura fixa opera de acordo com um dado ponto de troca (qualidade *versus* energia) que foi ajustado em tempo de projeto, a arquitetura configurável permite a troca entre energia e qualidade em tempo de operação. Essa capacidade é essencial para prolongar

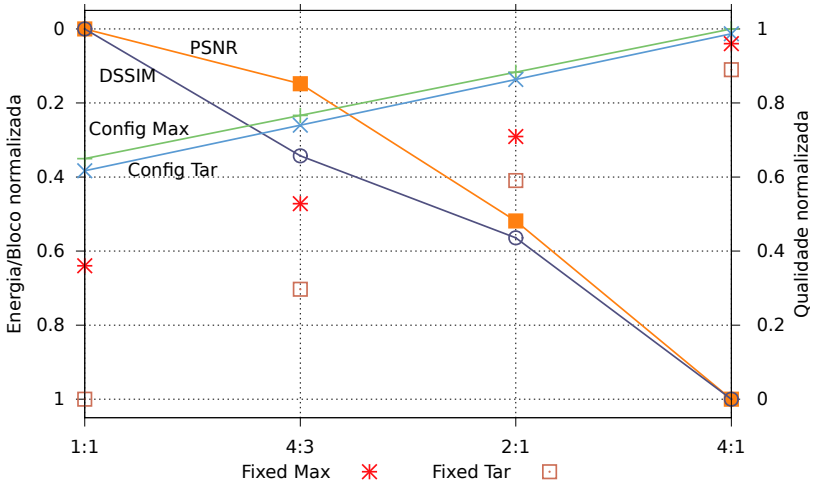


Figura 28: Resultados de energia/bloco e qualidade *versus* razão de subamostragem. Os resultados de energia/bloco são relativos ao máximo (7,08pJ/bloco para a arquitetura fixa, amostragem completa, operando na frequência alvo) e mínimo (1,34pJ/bloco para a arquitetura configurável, subamostragem 4:1, operando em frequência máxima). Já os resultados de qualidade assumem a amostragem completa como a máxima qualidade e a razão 4:1 como a mínima. Os valores intermediários estão ajustados de acordo com os resultados de qualidade, apresentados no Capítulo 3. É importante observar que tanto resultados de energia/bloco quanto os resultados de qualidade estão organizados de forma que os piores resultados estão embaixo.

a vida da bateria de Dispositivos Móveis Portáteis - *Portable Mobile Devices* (PMDs). Porém, as arquiteturas fixas são importantes como referência na avaliação do espaço de soluções, em termos de área e energia. A mais alta energia/bloco foi 7,08pJ/bloco, o que corresponde à arquitetura fixa 1:1 com frequência alvo. A arquitetura mais eficiente energeticamente foi a configurável usando subamostragem 4:1 para frequência máxima. Ela precisou de apenas 1,34pJ/bloco. Considerando a taxa de 1:1 como referência, foi alcançada redução de 20% na energia/bloco com custo de 0,7% em PSNR e 5,1% em DSSIM, em média, para razão 4:3. Para 2:1, a redução de 40% vem com custo de 2,5% PSNR e 8,2% DSSIM. Por fim, para razão 4:1, a redução de 60% vem com custo de 2,8% PSNR e 14,1% DSSIM.

Os resultados apresentados nesta subseção, considerando a amostragem completa, equivalem à comparação com o trabalho correlato de Walter,

Diniz e Bampi (2012). Da forma aqui apresentada, utilizou-se exatamente o mesmo fluxo de síntese e o mesmo conjunto de vetores para a simulação. Obtendo assim, uma comparação mais justa do que ao comparar diretamente os resultados da publicação original. Já nos Apêndices A e B são apresentadas comparações diretas com os trabalhos de Walter, Diniz e Bampi (2011) e Walter e Bampi (2011).

4.4 CONCLUSÕES

O uso de valores de pixel realistas e a simulação da arquitetura, conforme apresentado na Seção 4.3, demonstra uma nova visão sobre a eficiência energética. Ao serem consideradas as arquiteturas, os resultados dão evidência de que as arquiteturas com maior eficiência não são necessariamente as mais paralelas, contrário ao apresentado por Walter, Diniz e Bampi (2012). Afinal, como pode ser visto na Figura 28, quando operando com amostragem completa (i.e., razão 1:1) a arquitetura configurável apresenta melhor eficiência energética que sua equivalente fixa, a qual processa mais pixels em paralelo.

O uso da arquitetura configurável permite trocas entre energia/bloco e qualidade de codificação em tempo de operação e, especialmente para as razões de amostragem 4:3 e 2:1, a penalidade de qualidade é aceitável. Considerando a taxa de 4:1, a degradação da qualidade, medida em DSSIM, torna-se considerável. Ainda assim, a razão 4:1 é muito útil, por exemplo, para prolongar a bateria de um PMD, abrindo mão da qualidade de vídeo. A arquitetura configurável apresentou menor consumo de energia para todas as taxas de amostragem. Também apresentou menor área do que qualquer outra arquitetura fixa, com exceção da arquitetura para razão 4:1.

Apesar de as sínteses para frequência máxima terem apresentado menor energia/bloco que a alvo, suas áreas foram maiores (em alguns casos mais de duas vezes). Além disso, elas devem ser usadas com cuidado: se não for usada realmente a frequência máxima os resultados de energia/bloco podem deteriorar rapidamente, uma vez que a potência pode ser maior pelo número de transistores do circuito, mesmo com o decréscimo da frequência.

Os resultados apresentados neste capítulo serão utilizados no próximo capítulo para avaliar o impacto de *pel decimation* no fluxo de codificação.

5 IMPACTO DE PEL DECIMATION NA EFICIÊNCIA ENERGÉTICA

Neste capítulo serão investigados os resultados de velocidade de codificação apresentados abaixo das diagonais principais nas Tabelas 3 e 4 do Capítulo 3. A Seção 5.1 correlaciona a velocidade de codificação com o número de Somas das Diferenças Absolutas - *Sums of Absolute Differences* (SADs). Na Seção 5.2 analisa-se o *Fullsearch Block Matching Algorithm* (FBMA) (BARNEA; SILVERMAN, 1972) e o *Successive Elimination Algorithm* (SEA) (LI; SALARI, 1995) para averiguar a influência do algoritmo de busca no número de SADs. Através da análise apresentada, pôde-se obter, para efeito de ilustração, uma estimativa da energia/quadro, de acordo com o padrão de amostragem, o que é descrito na Seção 5.3. Já na Seção 5.4 uma modelagem das componentes de energia para as SADs é introduzida, levando-se em consideração tamanhos de bloco variável. Posteriormente, na Seção 5.5, a relação de redução do consumo de energia entre duas razões de amostragem é apresentada. Conforme demonstrado na Seção 5.5, neste tipo de análise os resultados são independentes da síntese das arquiteturas dedicadas para SAD. Através dos resultados de execução de codificação de um vídeo, foi possível utilizar-se o modelo apresentado para a relação de energia entre padrões de amostragem, conforme descrito na Seção 5.6. Assim, na Seção 5.7 foi possível estabelecer a relação de compromisso entre qualidade e energia de padrões de subamostragem no nível de Estimação de Movimento - *Motion Estimation* (ME), ao invés do nível de bloco, como apresentado anteriormente na Seção 4.3.3.

Por fim, apresenta-se na Seção 5.8 um modelo para o consumo de energia de um codificador de vídeo baseado em blocos, categoria da qual fazem parte os padrões de codificação estado da arte: o padrão H.264/AVC e o padrão de Codificação de Vídeo de Alta Eficiência - *High Efficiency Video Coding* (HEVC). Com tal modelo é possível demonstrar o impacto da arquitetura configurável para cada razão de amostragem, utilizando dados derivados de execuções de codificação de vídeo com o codificador *x264*. Também é possível relativizar os ganhos de energia, de acordo com o impacto da SAD na codificação.

5.1 CORRELAÇÃO ENTRE O NÚMERO DE SADS E A VELOCIDADE DE CODIFICAÇÃO

Analisando-se o cálculo da SAD, percebe-se que a redução do número de pixels implica em proporcional redução do número de operações. Por exemplo, o cálculo da SAD para um par de blocos 4x4 (16 pixels/bloco) requer um total de 16 diferenças absolutas e 15 somas. Semelhantemente, ao se aplicar subamostragem com razão 4:3, apenas 75% dos pixels são utilizados (12 pixels/bloco), o que requer 12 diferenças absolutas e 11 somas. Assim, são necessárias aproximadamente 75% das operações, resultando em uma execução 1,3 vezes mais rápida da SAD. Em uma análise preliminar, poderia-se especular que o tempo total de codificação diminua na mesma proporção.

Entretanto, ao se observar os resultados de aceleração de codificação, apresentados abaixo da diagonal principal nas Tabelas 3 e 4, nota-se que o padrão *border*, cuja razão de subamostragem é 4:3, executou 2,12 vezes mais rápido que a amostragem completa ($Speedup_{\text{completa}/\text{border}} = 2,12$).

Como os resultados descritos no Capítulo 3 foram obtidos através da execução de versões modificadas do *x264*, deve-se analisar o impacto das implementações de subamostragem através do código executável gerado pelo compilador. Continuando com a amostragem completa e o padrão *border* como exemplos, verificou-se que o código executável gerado para o padrão *border* é exatamente 75% menor que o gerado para a amostragem completa.

Dessa forma, conclui-se que somente uma parte do aumento de velocidade de codificação advém da diminuição do código. Deixando o cálculo individual da SAD de lado, o restante do ganho muito provavelmente está relacionado ao número de SADs calculadas em cada padrão. Seguindo este caminho, obteve-se o número de SADs executadas para a amostragem completa e para o padrão *border*. Para 50 quadros de determinada sequência¹, foram calculadas 490.145.617 SADs de tamanho de bloco variável. Ao usar o padrão *border* com a mesma sequência e parâmetros, foram calculadas 308.490.643 SADs. Assim, foram calculadas 37% menos SADs com o padrão *border* do que com amostragem completa.

Analisando em conjunto o tamanho de código de uma SAD para um determinado padrão ($TC_SAD_{\text{padrão}}$) e o número total de SADs calculadas para este mesmo padrão ($NSADs_{\text{padrão}}$), pode-se obter a quantidade de código

¹“Park Joy”(XIPH.ORG, 2011), 1080p@50fps, otimização para Relação Sinal-Ruído de Pico - *Peak signal-to-noise ratio* (PSNR), utilizando na ME o SEA conforme os experimentos do Capítulo 3.

executado calculando SADs ($CE_{\text{padrão}}$), conforme a Equação 5.1.

$$CE_{\text{padrão}} = TC_SAD_{\text{padrão}} \times NSADs_{\text{padrão}} \quad (5.1)$$

Tomando-se a amostragem completa como referência, $TC_SAD_{\text{completa}} = 1$, e tem-se que $TC_SAD_{\text{border}} = 0,75$. Para amostragem completa, a Equação 5.1 é reescrita da seguinte forma:

$$CE_{\text{completa}} = TC_SAD_{\text{completa}} \times NSADs_{\text{completa}} = 1 \times 490.145.617 \quad (5.2)$$

De forma semelhante, a Equação 5.1 pode ser reescrita para o padrão *border*:

$$\begin{aligned} CE_{\text{border}} &= TC_SAD_{\text{border}} \times NSADs_{\text{border}} = 0,75 \times 308.490.643 \\ &= 231.367.982,25 \end{aligned} \quad (5.3)$$

Ao calcular a relação $CE_{\text{completa}}/CE_{\text{border}}$, obtém-se 2,1185, valor que equivale à aceleração de codificação apresentada na Tabela 4². Assim, a seguinte equação, que relaciona os tamanhos de código da amostragem completa e de um padrão qualquer, é válida:

$$Speedup_{\text{completa/padrão}} = CE_{\text{completa}}/CE_{\text{padrão}} \quad (5.4)$$

Desta forma, o uso de *pel decimation* teve efeito duplo na aceleração da ME. Uma parte é obtida pela redução de cada cálculo de SAD, de acordo com a razão de subamostragem, e outra parte decorre da redução no número total de cálculos de SAD. A causa da redução na segunda parcela é investigada na próxima seção.

5.2 CORRELAÇÃO ENTRE O ALGORITMO DE BUSCA E O NÚMERO DE SADS

Nesta seção são analisados os algoritmos de busca *Fullsearch Block Matching Algorithm* (FBMA) (BARNEA; SILVERMAN, 1972) e *Successive Elimination Algorithm* (SEA) (LI; SALARI, 1995), bem como um trecho de

²Ou seja, $Speedup_{\text{completa/border}} = 2,12$. Uma das causas para a pequena diferença entre os valores explica-se por conta de que o valor apresentado na Tabela 4 ser a média de todas as amostras e *bitrates* considerados, enquanto o valor obtido através da proporção entre os códigos executados por cada padrão é apenas para uma amostra.

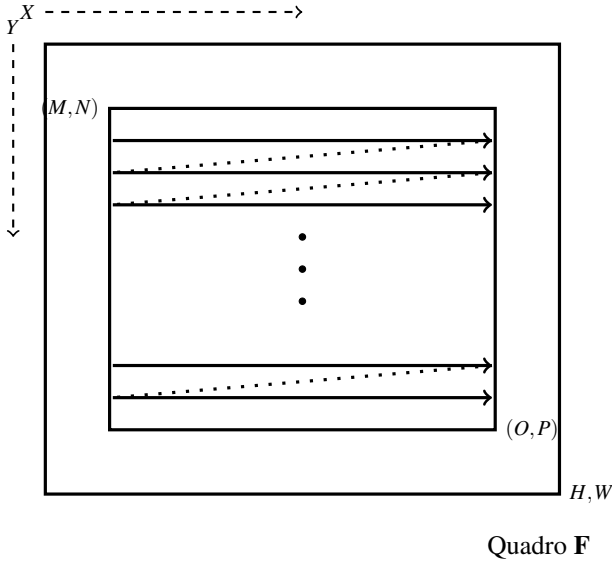


Figura 29: Representação da área de busca em um quadro **F**, delimitada pelas coordenadas (M, N) e (O, P) em um quadro com dimensões H e W . As setas indicam a ordem de percorridamento da área de busca na ordem conhecida por *raster scanning*.

código do *x264*, relativo ao algoritmo SEA. Primeiramente, são necessárias algumas definições. Chama-se de bloco “Original”, ou *Ori*, o bloco que está sendo codificando. Durante a ME, o algoritmo de *Block Matching* (BM) buscará, dentre os blocos candidatos, aquele que minimiza determinada métrica de similaridade. Para tanto, um algoritmo de busca é utilizado. Como apresentado no Capítulo 3, a métrica mais aceita para tal tarefa é a SAD. Considerando a Figura 29, que representa a área de busca em um quadro **F**, delimitada pelas coordenadas (M, N) e (O, P) , a SAD pode ser redefinida para incluir informações mais precisas quanto ao bloco candidato, oriundo do Quadro **F**, conforme a Equação 5.5.

$$SAD_{m,n}(Ori, F, x, y) = \sum_{i=0}^m \sum_{j=0}^n |Ori_{i,j} - F_{x+i,y+j}| \quad (5.5)$$

Por um lado, ao se considerar a FBMA como algoritmo de busca, o número de SADs (NSADs) executadas será o mesmo para cada padrão, contanto que a área de busca seja a mesma. Para comprovar tal afirmação, o

Algoritmo 1: FBMA

Input : Ori, m, n, F, M, N, O, P
Output: MV(x, y)

- 1 $sad_{min} \leftarrow SAD_{m,n}(Ori, F, i, j)$;
- 2 $(x, y) \leftarrow (M, N)$;
- 3 **for** $i=M$ **to** $O-m$ **do**
- 4 **for** $j=N$ **to** $P-n$ **do**
- 5 $sad_{temp} \leftarrow SAD_{m,n}(Ori, F, i, j)$;
- 6 **if** $sad_{temp} < sad_{min}$ **then**
- 7 $sad_{min} \leftarrow sad_{temp}$;
- 8 $(x, y) \leftarrow (i, j)$;
- 9 **end**
- 10 **end**
- 11 **end**

Algoritmo 1 descreve o FBMA. O algoritmo recebe como entradas um bloco *Ori*, o qual se está codificando, e a área de busca delimitada em um quadro **F**, conforme apresentado no início da seção. Para simplificar, usa-se a Estimacão de Movimento para Blocos de Tamanho Fixo - *Fixed Block-Size Motion Estimation* (FBSME) ao invés de Estimacão de Movimento para Blocos de Tamanho Variável - *Variable Block Size Motion Estimation* (VBSME). Na FBMA, o ponto inicial da busca não influencia sua velocidade, afinal todos os candidatos contidos na janela de busca serão analisados. Assim, para todo candidato, será calculada sua SAD com o bloco *Ori*. Ao observar o Algoritmo 1, fica claro que NSADs é definido apenas pelo tamanho da janela de busca.

O FBMA é o algoritmo mais intuitivo e obtém sempre o Vetor de Movimento - *Motion Vector* (MV) ótimo dentro de uma janela de busca. É extremamente lento, pois como pode-se observar no Algoritmo 1, calcula a SAD para todos os candidatos na área de busca. Porém, conforme mencionado no Capítulo 4, é bastante empregado em implementações em *hardware* da ME por conta de sua regularidade e fácil paralelização. Por outro lado, o algoritmo usado nos experimentos que conduziram à análise de qualidade e velocidade de codificação apresentada no Capítulo 3 foi o SEA. Tal algoritmo foi proposto por Li e Salari (1995), sendo um algoritmo de busca mais rápido que o FBMA, enquanto obtém exatamente os mesmos resultados³. Sua maior velocidade é atingida por se deixar de calcular todas SADs, através do uso da

³Desde que seja seguida a mesma ordem de candidatas, obtém o mesmo valor de SAD mínimo e também o menor MV. Caso contrário, obtém apenas o mesmo valor de SAD mínimo.

chamada *Absolute Difference of Sums* (ADS), definida na Equação 5.6.

$$ADS_{m,n}(Ori, F, x, y) = \left| \sum_{i=0}^m \sum_{j=0}^n Ori_{i,j} - \sum_{i=0}^m \sum_{j=0}^n F_{x+i,y+j} \right| \quad (5.6)$$

Para melhor entender para quais candidatos é necessário calcular a SAD, e como o SEA obtém o mesmo MV que o FBMA, é necessário entender a dedução do SEA, conforme apresentado por Li e Salari (1995). Considerando $Ori_{i,j}$ a intensidade do pixel na posição i,j do bloco Original, e $F_{i,j}$ a intensidade do pixel na posição i,j do Quadro \mathbf{F} , deslocados por um MV dado por (x,y) , por uma propriedade de inequações (YOUNG; GREGORY, 1988), tem-se que:

$$\left| |Ori_{i,j}| - |F_{i+x,j+y}| \right| \leq |Ori_{i,j} - F_{i+x,j+y}| \quad (5.7)$$

O que é equivalente às seguintes inequações:

$$|Ori_{i,j}| - |F_{i+x,j+y}| \leq |Ori_{i,j} - F_{i+x,j+y}| \quad (5.8)$$

$$|F_{i+x,j+y}| - |Ori_{i,j}| \leq |Ori_{i,j} - F_{i+x,j+y}| \quad (5.9)$$

Pode-se encontrar o somatório de ambos os lados das Inequações 5.8 e 5.9, conforme as Inequações 5.10 e 5.11:

$$\sum_{i=0}^M \sum_{j=0}^N |F_{i+x,j+y}| - \sum_{i=0}^M \sum_{j=0}^N |Ori_{i,j}| \leq \sum_{i=0}^M \sum_{j=0}^N |Ori_{i,j} - F_{i+x,j+y}| \quad (5.10)$$

$$\sum_{i=0}^M \sum_{j=0}^N |Ori_{i,j}| - \sum_{i=0}^M \sum_{j=0}^N |F_{i+x,j+y}| \leq \sum_{i=0}^M \sum_{j=0}^N |Ori_{i,j} - F_{i+x,j+y}| \quad (5.11)$$

Através da terminologia usada por Li e Salari (1995), o primeiro somatório da Equação 5.11 representa a norma da soma do bloco original, que pode ser denotado por \mathbf{O} . O segundo somatório é a norma de algum candidato deslocado por algum MV (x,y) , podendo ser denotado por $C_{x,y}$. O somatório do lado direito nada mais é do que a SAD. Sendo assim, as desigualdades das Equações 5.10 e 5.11 podem ser reescritas como:

$$\mathbf{O} - C_{x,y} \leq SAD(Ori, F, x, y) \quad (5.12)$$

$$C_{x,y} - O \leq SAD(Ori, F, x, y) \quad (5.13)$$

Suponha que se obteve um MV inicial para o bloco *Ori* no quadro *F*, e que suas coordenadas são *X* e *Y*. Na ME, há o interesse de se obter um MV (x, y) de forma que:

$$SAD(Ori, F, x, y) \leq SAD(Ori, F, X, Y) \quad (5.14)$$

Assim:

$$O - C_{(x,y)} \leq SAD(Ori, F, X, Y) \quad (5.15)$$

$$C_{(x,y)} - O \leq SAD(Ori, F, X, Y) \quad (5.16)$$

O que implica:

$$O - SAD(Ori, F, X, Y) \leq C_{(x,y)} \leq O + SAD(Ori, F, X, Y) \quad (5.17)$$

Dessa forma, apenas para os casos onde $C_{(x,y)}$ satisfaz a Inequação 5.17 uma nova SAD deve ser calculada para esta posição. Se a nova SAD for menor do que a atual mínima, o valor mínimo deve ser atualizado, bem como o melhor MV. O Algoritmo 2 descreve o SEA para FBSME, considerando as mesmas entradas apresentadas para o Algoritmo 1. A condição apresentada na Linha 5 do Algoritmo 2 é equivalente à Inequação 5.17, ajustada de mesma forma que poderia ser aplicado para ir da Inequação 5.8 para a Inequação 5.7.

Analisando o Algoritmo 2 fica claro que *NSADs* irá diminuir ao aplicar *pel decimation*, uma vez que o menor valor de SAD obtido será proporcional à razão de amostragem. Ao mesmo tempo, tal valor é utilizado como *threshold* para os cálculos posteriores. Como o ajuste para subamostragem foi feito apenas para a SAD, e não para ADS, estes últimos permanecem os mesmos. Sabendo disso, parte do decréscimo de qualidade apresentado pelos padrões de subamostragem são também efeito do estreitamento do espaço de busca (ao diminuir o valor de SAD_{min}). A Figura 30 apresenta dois histogramas dos valores de SAD, sendo um para amostragem completa e outro para o padrão *border*. Nota-se que o histograma para o padrão *border* está deslocado em relação ao histograma para amostragem completa. Percebe-se claramente que valores menores de SAD com *border* são mais frequentes do que para a amostragem completa.

Algoritmo 2: Algoritmo de Eliminações Sucessivas

Input : Ori, m, n, F, M, N, O, P
Output: MV(x, y)
 1 $sad_{min} \leftarrow SAD_{m,n}(Ori, F, M, N)$;
 2 $(x, y) \leftarrow (M, N)$;
 3 **for** $i=M$ to $O-m$ **do**
 4 **for** $j=N$ to $P-n$ **do**
 5 **if** $ADS_{m,n}(Ori, F, i, j) < sad_{min}$ **then**
 6 $sad_{temp} \leftarrow SAD_{m,n}(Ori, F, i, j)$;
 7 **if** $sad_{temp} < sad_{min}$ **then**
 8 $sad_{min} \leftarrow sad_{temp}$;
 9 $(x, y) \leftarrow (i, j)$;
 10 **end**
 11 **end**
 12 **end**
 13 **end**

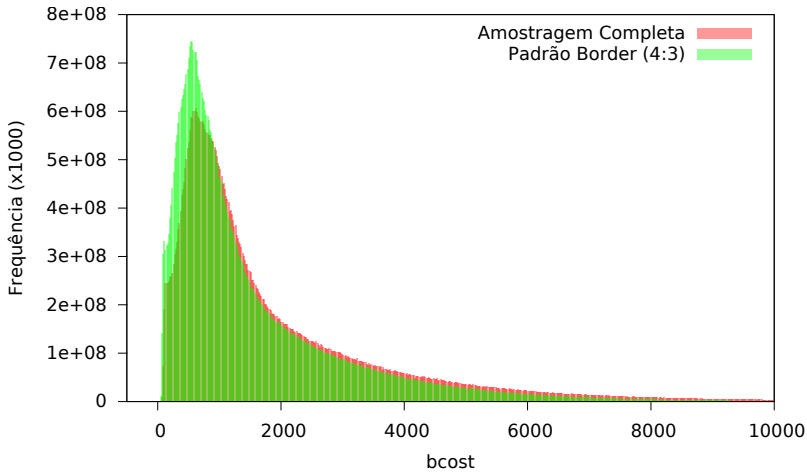


Figura 30: Histogramas dos valores de SAD (bcost) para amostragem completa e para o padrão *border*. Ao diminuir o número de pixels amostrados, a consequência óbvia é a diminuição do valor da SAD. Tal comportamento é ilustrado nesta figura.

5.2.1 SEA no x264

O Algoritmo 3 foi extraído do código do *x264*⁴, e corresponde à implementação de SEA. O laço mais externo é o deslocamento pela área de busca no sentido vertical. A primeira condicional foi adicionada⁵ para *early termination* em casos nos quais o custo do deslocamento (ou seja, do MV) a ser analisado, *ycost*, seja superior ao custo atual (SAD+MV) mínimos (*bcost*).

Algoritmo 3: x264_ESA

```

for( int my = min_y; my <= max_y; my++ )
{
    int i;
    int ycost = p_cost_mv[y<<2];
    if( bcost <= ycost )
        continue;
    bcost -= ycost;
    xn = h->pixf.ads[i_pixel]( enc_dc,
        sums_base + min_x + my * stride, delta,
        cost_fpel_mv+min_x, xs, width, bcost );
    for( i = 0; i < xn-2; i += 3 )
        COST_MV_X3_ABS( min_x+xs[i],my,
            min_x+xs[i+1],my, min_x+xs[i+2],my );
    bcost += ycost;
    for( ; i < xn; i++ )
        COST_MV( min_x+xs[i], my );
}

```

Para uma linha da janela de busca, são calculados todos os ADS ($h \rightarrow \text{pixf.ads}$), e apenas os candidatos que cumpram a Inequação 5.17 são escritos em um *buffer*, *xs*. Para tais candidatos é feito o cálculo de SAD e possíveis atualizações na SAD_{min} e MV, o que é descrito pelas macros *COST_MV_X3_ABS* e *COST_MV*. O número de cálculos de SAD está diretamente relacionado com o valor de *xn*, que é o número total de elementos no *buffer* de candidatos que satisfizeram à Inequação 5.17. A Figura 31 apresenta o diagrama de Pareto com as frequências relativas de valores de *xn* para amostragem completa e para o padrão *border*. Pode-se perceber pelas curvas cumulativas que há maior frequência de valores pequenos de *xn* para *border*, justificando o menor número de cálculos de SADs.

⁴x264 Revisão 2106, arquivo “me.c”, linhas 710 até 724.

⁵Commit 2f75bcc1b74e41ae39134719bcdfc2a6a0265.

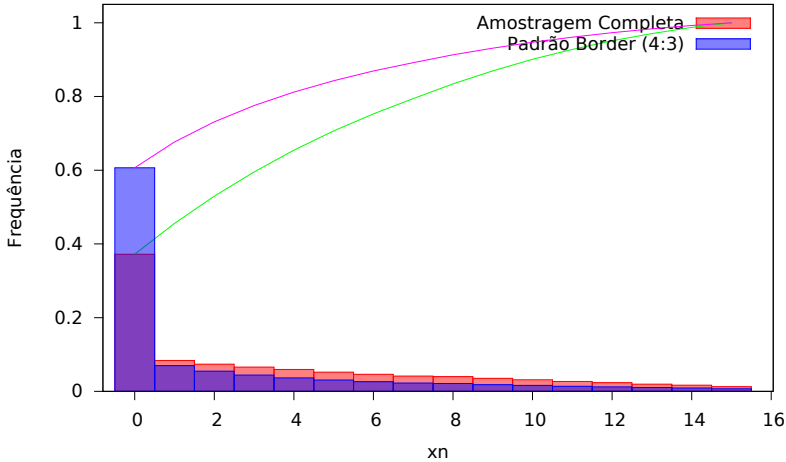


Figura 31: Diagrama de Pareto apresentando ocorrências relativas de tamanho (x_n) do *buffer* de candidatos (x_s) que satisfizeram a Inequação 5.17, de acordo com a execução do código do *x264* apresentado no Algoritmo 3.

5.3 MODELAGEM INICIAL DOS COMPONENTES DE ENERGIA

Uma vez identificado e demonstrado o impacto de *pel decimation* na ME, torna-se viável obter uma estimativa em termos de consumo de energia, que leve em consideração os resultados da seção anterior. Para tal tarefa, é necessária a modelagem da energia de um codificador de vídeos, levando em conta o número total de cálculos de SAD. Sabendo-se o tamanho de código do *x264* gerado para a SAD com determinado padrão ($TC_{\text{padrão}}$), e através do número já conhecido de SADs da amostragem completa (490.145.617), é possível, a partir dos valores de aceleração apresentados nas Tabelas 3 e 4 ($Speedup_{\text{completa/padrão}}$), calcular o número total de SADs para tal padrão, conforme a Equação 5.18, derivada da Equação 5.4.

$$NSADs_{\text{padrão}} = \frac{NSADs_{\text{completa}}}{Speedup_{\text{completa/padrão}} \times TC_{\text{padrão}}} \quad (5.18)$$

Com tais valores, uma pré-estimativa de ganhos energéticos por quadro codificado pode ser traçada, considerando também os valores de energia/bloco apresentados na Seção 4.3. Afinal, conforme propõem-se na Equação 5.19, a energia/quadro de determinado padrão ($EF_{\text{padrão}}$) pode ser obtida através do produto do número de cálculos de SAD pela energia por cálculo

de SAD, dividido pelo número total de quadros codificados ($N_{Quadros}$) para obter o número de SADs.

$$EF_{\text{padrão}} = \frac{NSADs_{\text{padrão}} \times E_{\text{padrão}}}{N_{Quadros}} \quad (5.19)$$

Para efeito de ilustração, considerando a amostragem completa, têm-se que $NSADs_{\text{completa}} = 490.145.617$, $E_{\text{completa}_{90nm@tar}} = 15,58pJ/\text{bloco}$, $N_{Quadros} = 50$, $EF_{\text{completa}} = 152,72\mu J/\text{frame}$. Para *border*, $NSADs_{\text{border}} = 308.490.643$, $E_{\text{border}_{90nm@tar}} = 12,46pJ/\text{bloco}$, logo, têm-se que $EF_{\text{border}} = 76,8\mu J/\text{frame}$. Sendo assim, estima-se que a energia/quadro requerida pelo padrão *border* é aproximadamente metade da requerida pela amostragem completa. Porém, tal estimativa está aquém da realidade, uma vez que o algoritmo de busca do $x264$ considera tamanhos de bloco variáveis. Assim, na próxima seção é apresentada uma modelagem mais precisa para as componentes de energia para um codificador de vídeo, a qual leva em conta o tamanho do bloco.

5.4 MODELAGEM DAS COMPONENTES DE ENERGIA CONSIDERANDO VBSME

Ainda considerando FBSME, a exemplo da seção anterior, a energia dos cálculos de SAD (E_{SADs}), pode ser definida em função do número de SADs ($NSADs$) e da energia/SAD ($ESAD$), conforme apresentado na Equação 5.20.

$$E_{SADs} = NSADs \times ESAD \quad (5.20)$$

Já para VBSME, cada tamanho de bloco, de acordo com o número de pixels, deve ser considerado separadamente. Sejam as tuplas $NSADs_i = (x_0, x_1, \dots, x_{n-1})$ e $ESAD_i = (e_0, e_1, \dots, e_{n-1})$. Cada elemento x_i de $NSADs_i$ corresponde ao número de SADs para o índice i . Já cada elemento de $ESAD_i$ denota a energia por SAD para determinado tamanho de bloco, de acordo com o índice i . Ainda, 2^{i+4} , por sua vez, corresponde ao número total de pixels de um bloco. Assim, n é o número de diferentes tamanhos de bloco, considerando o número de pixels por bloco⁶. Com tais componentes pode-se

⁶Tomando o H.264/AVC como exemplo, há sete possíveis tamanhos de bloco: 4x4, 4x8, 8x4, 8x8, 8x16, 16x8 e 16x16. Ao considerar o número de pixels por bloco, há 5 possibilidades: 16 pixels (bloco 4x4), 32 (4x8 e 8x4), 64 (8x8), 128 (8x16 e 16x8) e 256 (16x16). Dessa forma, para o H.264/AVC, $n = 5$.

calcular E_{SADs} para VBSME a partir da Equação 5.21.

$$E_{SADs} = \sum_{i=0}^{n-1} (NSADs_i \times ESAD_i) \quad (5.21)$$

Para as Equações 5.20 e 5.21, dada uma arquitetura de SAD qualquer, obtendo-se seu período (T), potência (P) e o número de ciclos por SAD (C_i), a seguinte equação é válida:

$$ESAD_i = C_i \times T \times P \quad (5.22)$$

Ainda, é possível adicionar *pel decimation* ao modelo. Considerando j como o índice da razão de amostragem aplicada no cálculo da SAD, as tuplas $NSADs_i$ e $ESAD_i$ passam a matrizes, $NSADs_{i,j}$ ⁷ e $ESAD_{i,j}$. Considerando VBSME, as Equações 5.21 e 5.22 podem ser reescritas como:

$$E_{SADs}^j = \sum_{i=0}^{n-1} (NSADs_{i,j} \times ESAD_{i,j}) \quad (5.23)$$

$$ESAD_{i,j} = C_{i,j} \times T \times P \quad (5.24)$$

Substituindo a Equação 5.24 na Equação 5.23, obtém-se a Equação 5.25. Esta, por sua vez, pode ser reescrita conforme a Equação 5.26.

$$E_{SADs}^j = \sum_{i=0}^{n-1} (NSADs_{i,j} \times C_{i,j} \times T \times P) \quad (5.25)$$

$$E_{SADs}^j = T \times P \times \sum_{i=0}^{n-1} (NSADs_{i,j} \times C_{i,j}) \quad (5.26)$$

5.5 RELAÇÃO DE ENERGIA ENTRE DUAS RAZÕES DE AMOSTRAGEM

A relação de energia para SADs entre dois padrões j e k pode ser escrita como:

$$\text{relação_energia}_{j,k} = \frac{E_{SADs}^j}{E_{SADs}^k} \quad (5.27)$$

⁷ $NSADs$ passa a matriz pois, como visto na Seção 5.1, o número de SADs pode variar de acordo com a razão de amostragem. Pode-se garantir que não haverá mudanças apenas no caso do FBMA, pois este calcula a similaridade para todos os candidatos na área de busca, sendo independente de um controle mais refinado.

A partir da Equação 5.26, a Equação 5.27 pode ser reescrita como:

$$\text{relação_energia}_{j,k} = \frac{T_j \times P_j \times \sum_{i=0}^{n-1} (NSADs_{i,j} \times C_{i,j})}{T_k \times P_k \times \sum_{i=0}^{n-1} (NSADs_{i,k} \times C_{i,k})} \quad (5.28)$$

Considerando uma arquitetura configurável para *pel decimation*, T e P são constantes, independentemente da razão de amostragem, ou seja, $T_j = T_k$ e $P_j = P_k$. Dessa forma, pode-se simplificar a Equação 5.28 como:

$$\text{relação_energia}_{j,k} = \frac{\sum_{i=0}^{n-1} (NSADs_{i,j} \times C_{i,j})}{\sum_{i=0}^{n-1} (NSADs_{i,k} \times C_{i,k})} \quad (5.29)$$

Comparando as equações 5.26 e 5.29, pode-se observar que embora os valores absolutos de energia dependam diretamente de T e P , os valores relativos, dada uma arquitetura configurável para *pel decimation*, são independentes de tais valores. Dessa forma, o ganho, em termos de eficiência energética do uso de determinada razão de subamostragem em relação à outra, depende apenas do número de ciclos executados por SAD ($C_{i,j}$), e o número total de SADs executadas ($NSADs_{i,j}$), para cada tamanho de bloco (2^{i+4}) e razão de subamostragem (indexado por j).

O número de ciclos ($C_{i,j}$) depende da implementação da Máquina de Estados Finitos - *Finite State Machine* (FSM) da arquitetura configurável, na forma de: número de estados (ciclos) de cálculo ($Ciclos_{calc}$), número de estados de sincronização ($Ciclos_{sync}$). Também é dependente da razão de amostragem ($Razão_j$), do número de pixels por bloco (2^{i+4}) e do número de entradas paralelas por bloco ($N_Entradas_Paralelas$). Assim, $C_{i,j}$ pode ser obtido a partir da seguinte equação:

$$C_{i,j} = \frac{2^{i+4} \times Ciclos_{calc} \times Razão_j}{N_Entradas_Paralelas} + Ciclos_{sync} \quad (5.30)$$

Substituindo $C_{i,j}$ em 5.26, obtém-se:

$$E_{SADs}^j = T \times P \times \sum_{i=0}^{n-1} \left(NSADs_{i,j} \times \left(\frac{2^{i+4} \times Ciclos_{calc} \times Razão_j}{N_Entradas_Paralelas} + Ciclos_{sync} \right) \right) \quad (5.31)$$

A Equação 5.31 pode ser reescrita conforme a Equação 5.32. Nesta última, coloca-se em evidência que o impacto dos ciclos de sincronização

está relacionado apenas com o número de SADs.

$$E_{SADs}^j = T \times P \times \left(\frac{CiclosCalc \times Razão_j}{N_Entradas_Paralelas} \left(\sum_{i=0}^{n-1} (NSADs_{i,j} \times 2^{i+4}) \right) + CiclosSinc \left(\sum_{i=0}^{n-1} NSADs_{i,j} \right) \right) \quad (5.32)$$

5.6 RELAÇÃO DE ENERGIA DA ARQUITETURA CONFIGURÁVEL PROPOSTA DE ACORDO COM PADRÕES DE AMOSTRAGEM

Na arquitetura proposta na Seção 4.2 considerou-se apenas blocos fixos de tamanho 4x4 pixel. A mesma arquitetura pode ser utilizada para outros tamanhos de bloco, bastando-se para isso aumentar a largura do último somador e registrador e executar o laço *LOAD-CALC* de acordo com o tamanho de bloco. O número de execuções deste laço pode ser facilmente calculada. Já o impacto do aumento na largura do somador e do registrador mencionados causará impacto de área, potência (P) e atraso (T). Ora, na seção anterior foi demonstrado que a relação de energia entre duas razões é independente destes valores (P e T). Logo, pode-se tomar a arquitetura proposta para relacionar a energia entre duas razões de amostragem. Resta apenas obter-se o número de SADs para cada razão de amostragem e tamanho de bloco ($NSADs_{i,j}$). Considerando que os experimentos apresentados no Capítulo 3 utilizaram *x264*, pode-se obter a partir deste os valores necessários. Ainda, como visto no Capítulo 3, a aceleração obtida depende do padrão de amostragem. Como visto na Seção 5.1, a aceleração decorre em parte pelo número de SADs executadas. Dessa forma, $NSADs_{i,j}$ dependerá do padrão utilizado, com taxa de índice j . Uma vez que o *x264* segue o padrão H.264/AVC, conforme mencionado, $n = 5$. Substituindo n em 5.32 e expandindo o somatório, têm-se:

$$E_{SADs}^j = T \times P \times (8 \times Razão_j \times (NSADs_{0,j} + 2 \times (NSADs_{1,j} + 2 \times NSADs_{2,j} + 4 \times NSADs_{3,j} + 8 \times NSADs_{0,j})) + 2 \times (NSADs_{0,j} + NSADs_{1,j} + NSADs_{2,j} + NSADs_{3,j} + NSADs_{4,j})) \quad (5.33)$$

Considerando a amostragem completa (razão 1:1, índice $j = 0$) e os padrões *border* (razão 4:3, índice $j = 1$), *checker even* (2:1, $j = 2$) e *skew* (4:1,

$j = 3$), para o mesmo teste efetuado na Seção 5.1, obtém-se:

$$NSADs = \begin{bmatrix} 1.068.516 & 1.287.851 & 1.703.111 & 1.143.532 \\ 0 & 0 & 0 & 0 \\ 257.448.179 & 190.445.213 & 149.581.844 & 73.819.978 \\ 70.559.218 & 31.371.581 & 10.642.137 & 1.781.458 \\ 161.069.704 & 85.385.998 & 42.563.397 & 26.637.665 \end{bmatrix} \quad (5.34)$$

Um fato curioso a ser observado é que para $i = 1$, $NSADs = 0$. Isto significa que não foram calculadas SADs com tamanhos de bloco 4x8 e 8x4 durante a ME do $x264$ para o vídeo utilizado⁸. Substituindo os valores de $NSADs$ na Equação 5.33, obtém-se:

$$\begin{aligned} E_{SADs}^0 &= T \times P \times 34.359.893.154 \\ E_{SADs}^1 &= T \times P \times 14.898.285.200 \\ E_{SADs}^2 &= T \times P \times 5.873.708.718 \\ E_{SADs}^3 &= T \times P \times 1.680.520.762 \end{aligned} \quad (5.35)$$

Tomando E_{SADs}^0 como referência e utilizando a Equação 5.27, a proporção de energia de um determinado padrão de índice j em relação à amostragem completa ($GE_{padrão/completa}$) pode ser calculada como segue:

$$GE_{padrão/completa} = 1 - \frac{E_{SADs}^j}{E_{SADs}^0} \quad (5.36)$$

Substituindo os resultados de 5.35 na Equação 5.36, obtém-se:

$$\begin{aligned} GE_{border/completa} &= 0,566404786731253 \\ GE_{checkerEven2:1/completa} &= 0,829053347410767 \\ GE_{skew/completa} &= 0,951090628993869 \end{aligned} \quad (5.37)$$

Tais valores demonstram que a energia consumida por *border* é cerca de 56,64% menor que da SAD com amostragem completa. Da mesma forma, *checker even 2:1* e *skew* consomem, respectivamente, 82,91% e 95,11% menos energia que a SAD com amostragem completa. Tais valores valem para o algoritmo de busca SEA e para o vídeo codificado. Por outro lado, os valores refletem o comportamento médio apresentado na Tabela 4.

⁸Tal fato é justificado através dos parâmetros utilizados na codificação dos vídeos de teste. No $x264$, por *default*, os tamanhos de blocos 4x8 e 8x4 são desabilitados. De acordo com seu manual, o uso desses tamanhos de bloco resulta em alta degradação na velocidade de codificação, para pouco ganho de qualidade. De acordo com o manual de outra ferramenta que utiliza o $x264$, apenas utiliza-se tais tamanhos de bloco em amostras de vídeo com muito movimento. Ainda assim, esperam-se ganhos de apenas cerca de 0,1dB PSNR (TEAM, 2014).

5.7 RELAÇÃO DE COMPROMISSO ENTRE QUALIDADE E ENERGIA

De maneira semelhante à Figura 28, a Figura 32 apresenta a relação de compromisso entre qualidade e energia, porém levando em conta a VBSME. Pode-se observar o comportamento linear ao se considerar apenas a SAD individualmente. Isto se deve ao fato de que tais resultados estão apenas relacionados à FSM da arquitetura proposta. Dessa forma, considerando apenas uma SAD, os ganhos em relação à SAD com amostragem completa foram de 20%, 40% e 60%⁹, para as razões 4:3, 2:1 e 4:1, respectivamente.

Já os resultados obtidos ao se considerar VBSME, como visto na seção anterior, são mais promissores: 56,64%, 82,91% e 95,11%, para os padrões *border*, *checker even 2:1* e *skew*, respectivamente. Com tais dados é possível observar o alto ganho ao se usar o padrão *border*, obtido ao eliminar apenas 1/4 dos pixels de cada bloco. Mesmo com tamanha economia de energia, tal padrão apresentou pouca degradação de qualidade. Pode-se observar nas curvas de qualidade presentes na Figura 32 que há pouquíssima degradação de PSNR, e mesmo o Índice de Dissimilaridade Estrutural - *Structural Dissimilarity Index* (DSSIM) não cruzou a curva de energia¹⁰. Avançando a subamostragem, o ganho de energia, mesmo que elevado no geral, vem ao custo de uma maior degradação de qualidade. Afinal, enquanto que para avançar de 1:1 para *border*, ganhou-se 56,64% de economia de energia, avançando de *border* para *checker even 2:1* ganha-se um acréscimo na economia já alcançada de apenas 26,27%. Avançando de *checker even 2:1* para *skew* o acréscimo na economia é ainda menor: 12,2%.

5.8 MODELAGEM DAS COMPONENTES DE ENERGIA DE UM CODIFICADOR DE VÍDEO

Pode-se dividir a energia total de codificação ($E_{\text{codificação}}$) de uma arquitetura qualquer de codificador de vídeo baseado em blocos, tais como os codificadores estado da arte H.264/AVC e HEVC em duas componentes, quais sejam, energia de processamento (EP) e energia das memórias (EM), conforme a Equação 5.38.

$$E_{\text{codificação}} = EP + EM \quad (5.38)$$

⁹Passo entre as razões, de acordo com o número de ciclos da FSM. Para maiores detalhes sobre estes valores, consultar o a Subseção 4.3.3 e o Apêndice B.

¹⁰Antes da interseção entre as curvas, o ganho de energia era maior do que o custo em qualidade (ambos normalizados).

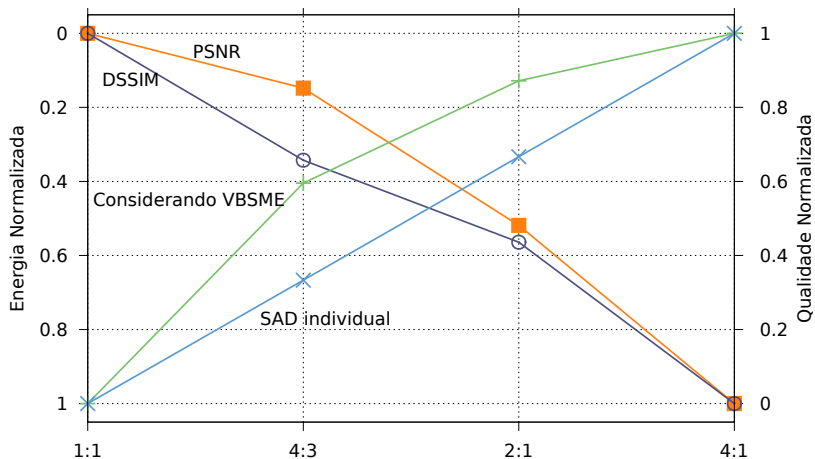


Figura 32: Compromisso entre energia e qualidade. Os valores são relativos aos valores máximos e mínimo, da mesma forma como na Figura 28. A reta “SAD individual” foi obtida dos resultados apresentados no Capítulo 4, e corresponde à síntese da arquitetura configurável para frequência-máxima. Tal arquitetura obteve os melhores resultados no capítulo anterior. Já a outra curva de valor normalizado de energia equivale aos ganhos relativos obtidos, deduzidos neste capítulo. Assim, o valor máximo é a SAD com amostragem completa, e o mínimo é resultante do padrão *skew*. Ao contrário dos resultados da SAD individual, que independem do padrão, os resultados considerando VBSME são dependentes. Assim, os padrões utilizados foram: *border* (4:3), *checker even* (2:1) e *skew* (4:1). Vale relembrar que os resultados de energia e qualidade estão ordenados de forma que os piores resultados estão embaixo.

Qualquer dessas componentes pode ser expandida para cada etapa da codificação de vídeo. Como neste trabalho o impacto se dá na energia de processamento da ME (EP_{ME}), pode-se reescrever EP em termos de EP_{ME} e da energia de processamento das demais etapas ($EP_{\bar{N}ME}$), conforme a Equação 5.39.

$$EP = EP_{ME} + EP_{\bar{N}ME} \quad (5.39)$$

Da mesma forma, pode-se reescrever EP_{ME} como:

$$EP_{ME} = E_{SADs} + E_{ME_Control} \quad (5.40)$$

Considerando as equações apresentadas na Seção 5.4 pode-se reescrever a Equação 5.38 expandida, para VBSME e subamostragem j , conforme a Equação 5.41:

$$E_{\text{codificação}}^j = \left(T \times P \times \sum_{i=0}^{n-1} (NSADs_i \times C_{i,j}) \right) + E_{ME_Control} + EP_{\bar{N}ME} + EM \quad (5.41)$$

5.8.1 Impacto da Arquitetura Configurável no Fluxo de Codificação

Na Equação 5.41 pode-se isolar o primeiro termo como E_{SADs}^j . Todos os termos posteriores são componentes de energia que não são diretamente ligados à SAD¹¹. Assim, pode-se modelar $E_{\text{codificação}}^j$ conforme a Equação 5.42.

$$E_{\text{codificação}}^j = E_{SADs}^j + \overline{E_{SADs}^j} \quad (5.42)$$

Dessa forma, através dos resultados apresentados em 5.37, pode-se obter o impacto da redução, mudando o padrão de amostragem, de E_{SADs}^j sobre $E_{\text{codificação}}^j$. De fato, tais reduções podem ser escritas em função do percentual de E_{SADs}^j em relação à $E_{\text{codificação}}^j$ (x no intervalo]0, 100]) para cada um dos padrões de amostragem considerados. As funções descritas nas Equações

¹¹ Alguns termos podem mudar de forma indireta. Pode-se projetar, por exemplo, uma hierarquia de memória que obtenha ganhos através da subamostragem.

5.43 até 5.45 são apresentadas no gráfico da Figura 33.

$$\begin{aligned} \text{ganho_total} \frac{\text{border}}{\text{completa}}(x) &= GE_{\text{border/completa}} \times x \\ &= 0,561811788785931 \times x \end{aligned} \quad (5.43)$$

$$\begin{aligned} \text{ganho_total} \frac{\text{checkerEven2:1}}{\text{completa}}(x) &= GE_{\text{checkerEven2:1/completa}} \times x \\ &= 0,823018020979194 \times x \end{aligned} \quad (5.44)$$

$$\begin{aligned} \text{ganho_total} \frac{\text{skew}}{\text{completa}} &= GE_{\text{skew/completa}} \times x \\ &= 0,951090628993869 \times x \end{aligned} \quad (5.45)$$

Por fim, considerando $\overline{E_{SADs}^j}$ fixa, obteve-se as funções que demonstram os ganhos relativos ao total da codificação, devidos ao uso dos padrões escolhidos, tomando-se por base a arquitetura configurável proposta no Capítulo 4. Tais funções, à princípio obtidas para SEA, podem ser obtidas de forma semelhante, para qualquer algoritmo de busca, tendo-se disponíveis dados de execução de SADs no algoritmo de busca escolhido. Obtidas as funções de ganho total e, conhecendo-se posteriormente o percentual de energia de cálculos de SADs em relação ao total, pode-se obter o real ganho para qualquer padrão de subamostragem em qualquer algoritmo de busca. Além disso, o modelo apresentado também suporta a alteração da arquitetura, contanto que esta se mantenha configurável.

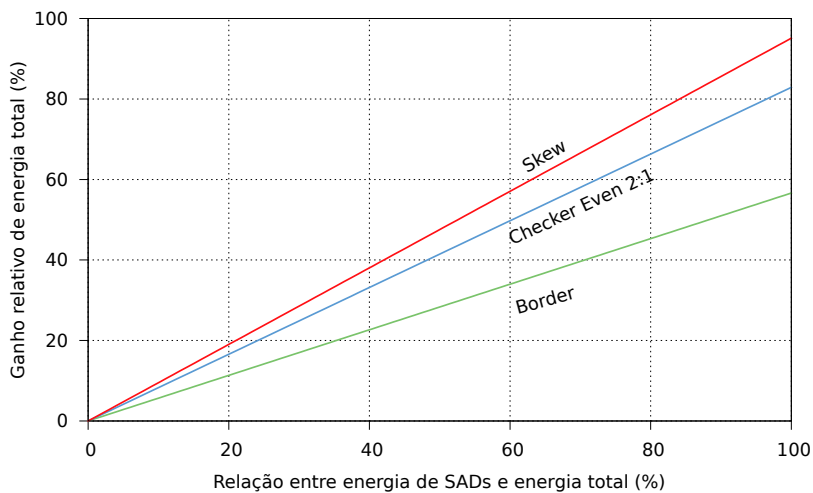


Figura 33: Funções de ganho relativo no total de energia de uma codificação de uso, de acordo com padrões de amostragem e a relação de energia das SADs no total de energia. A reta marcada como *border* representa a função descrita na Equação 5.43. De forma semelhante, as retas marcadas como *Checker Even 2:1* e *Skew* representam as funções presentes nas Equações 5.44 e 5.45, respectivamente.

6 CONCLUSÕES

Nesta dissertação, apresentou-se uma ampla análise de qualidade da codificação de vídeos de alta definição em função dos padrões e das próprias razões de subamostragem usados em *pel decimation*. O padrão com maior destaque é o chamado *border*, o qual apresentou as menores degradações de qualidade em relação à amostragem completa (1% em PSNR e 5% em DSSIM), ao mesmo tempo que proporcionou aumento de 2 vezes na velocidade de codificação, apesar de sua razão de subamostragem ser 4:3. O padrão *checker even*, com razão 8:1, obteve o maior aumento de velocidade, sendo aproximadamente 4,3 vezes mais rápido que a amostragem completa. Por outro lado, também sofreu a maior degradação, 4% PSNR e 17% DSSIM. Os padrões com melhor qualidade foram os com maior taxa de amostragem, enquanto os mais rápidos foram os piores em termos de qualidade. As três razões de subamostragem com menor redução de qualidade foram 4:3, 2:1 e 4:1, nesta ordem.

Para tais razões, em conjunto com a amostragem completa, propôs-se uma arquitetura configurável para o cálculo da SAD. Considerando apenas a energia/bloco, as configurações de subamostragem 4:3, 2:1 e 4:1 demonstraram ganhos de 20%, 40% e 60%, respectivamente, em relação à amostragem completa.

Posteriormente, demonstrou-se que os ganhos de velocidade obtidos na codificação em *software* sofreram efeito direto do algoritmo de busca utilizado, o SEA. Dessa forma, o uso de *pel decimation* teve um efeito duplo na redução do número de operações na ME. Parte da redução foi obtida na redução de cada cálculo de SAD, de acordo com a razão de subamostragem. Outra parte foi decorrente da redução no número total de cálculos de SAD.

Ao modelar o comportamento da arquitetura configurável através de equações, pôde-se concluir que, dada uma arquitetura configurável, os ganhos relativos de eficiência energética entre razões de subamostragem não dependem da síntese da arquitetura. Por outro lado, os resultados de frequência e potência influem no valor absoluto do consumo energético, conforme apresentado no Capítulo 4. Dessa forma, pôde-se estimar a energia consumida considerando VBSME, conforme o apresentado no Capítulo 3, no qual o *software* utilizado para os testes foi o *x264*, usando tamanho de bloco variável.

Modelando-se as componentes de energia das SADs, através de dados de codificação de um vídeo 1080p, obteve-se a redução possível de ser alcançada com os padrões de subamostragem. Considerando os padrões *border*, *checker even 2:1* e *skew*, as reduções de energia em relação à amostra-

gem completa foram de 56,64%, 82,91% e 95,11%, respectivamente. Dessa forma, o padrão com melhor relação custo-benefício continua o *border*: apresenta 56,54% de redução de energia em relação à amostragem completa, com degradação média de apenas 1% em PSNR e 5% em DSSIM. Finalmente, os resultados foram extrapolados para considerar o impacto do número total de SADs na energia de todo o codificador.

A limitação deste trabalho advém do uso do *x264* como software base para os experimentos. Por um lado, ganha-se no número de experimentos que podem ser feitos, em função da velocidade de codificação superior àquela do *software* de referência. Por outro, perde-se parte da precisão dos dados que poderiam ser obtidos utilizando-se o *software* de referência.

6.1 TRABALHOS FUTUROS

Com relação à análise de qualidade, listam-se os seguintes trabalhos futuros:

- Análise de qualidade para os padrões de *pel decimation* para outros algoritmos de busca, tais como *Diamond Search* (DS), *Hexagon*, etc;
- Para o SEA, adaptação do cálculo de ADS, de forma a levar em consideração os pixels amostrados conforme o mesmo padrão adotado na SAD. Espera-se que melhore a qualidade da predição e portanto, a qualidade final do vídeo codificado. Por outro lado, ao diminuir também o valor de ADS na mesma razão de subamostragem, aumentará o número total de SADs calculadas.

Considerando ao projeto e síntese de arquiteturas, pode-se listar os seguintes trabalhos futuros:

- Análise de outras arquiteturas de SAD, para melhor demonstrar o comportamento sob simulação. Afinal, através dos dados apresentados no Capítulo 4, é possível perceber que houve divergências entre os resultados obtidos e aqueles apresentados por Walter, Diniz e Bampi (2012);
- Suporte a outros tamanhos de bloco para a arquitetura configurável, além do 4x4;
- Projeto e síntese de arquitetura configurável para cálculo da ADS;
- Projeto e síntese de arquitetura para ME utilizando o algoritmo SEA. Assim, podem-se obter valores de área, potência e eficiência energética para comparação;

Por fim, outros desdobramentos, em relação à estimativa da energia para codificação de vídeo, podem ser:

- Estimativa do impacto de *pel decimation* na organização de hierarquias de memórias para ME;
- Estimativas de energia de variações arquiteturais, utilizando o modelo apresentado no Capítulo 5;

Referências Bibliográficas

ACER. *Acer Showcases New 10.1-inch Iconia A3 Android Tablet; Liquid S2 Smartphone with 4K Recording; and Touch & Type Devices at IFA 2013*. 2013. <<http://us.acer.com/ac/en/US/press/2013/66580>>. Acessado em 10/12/2013.

AGHA, S.; DWYER, V.; CHOULIARAS, V. Motion estimation with low resolution distortion metric. *Electronics Letters*, IET, v. 41, n. 12, p. 693–694, 2005.

AHMED, N.; NATARAJAN, T.; RAO, K. R. Discrete cosine transform. *IEEE Transactions on Computers*, C-23, n. 1, p. 90–93, jan 1974.

ARCHANA, S.; DEVI, D. R. Area efficient sad architecture for block based video compression standards. *International Journal of Computer Applications in Engineering Sciences*, v. 3, p. 1–9, 2013.

BARNEA, D. I.; SILVERMAN, H. A class of algorithms for fast digital image registration. *IEEE Transactions on Computers*, C-21, n. 2, p. 179–186, Feb 1972. ISSN 0018-9340.

BENNETT, H. *Hugh's News*. 2011. <<http://www.hughsnews.ca/faqs-/authoritative-blu-ray-disc-bd-faq/4-physical-logical-and-application-specifications4.6>>. Acessado em 15/08/2011.

BROWN, M. *Advanced Digital Photography*. [S.l.]: Media Publishing, 2004. ISBN 9780958188852.

BRUNVAND, E. *Digital VLSI Chip Design With Cadence and Synopsys CAD Tools*. [S.l.]: ADDISON WESLEY Publishing Company Incorporated, 2010. ISBN 9780321547996.

CHEN, C.-Y. et al. Analysis and architecture design of variable block-size motion estimation for h. 264/avc. *IEEE Transactions on Circuits and Systems I: Regular Papers*, IEEE, v. 53, n. 3, p. 578–593, 2006.

CHEN, T. et al. Analysis and architecture design of an HDTV720p 30 frames/s H.264/AVC encoder. *IEEE Transactions on Circuits and Systems for Video Technology*, v. 16, n. 6, p. 673–688, jun 2006.

CISCO. *Cisco Visual Networking Index: Forecast and Methodology, 2012-2017*. [S.l.], maio 2013.

CVRL. *Cone Fundamentals*. 2013. <<http://cvrl.ucl.ac.uk/cones.htm>>. Acessado em 31/12/2013.

CVRL. *Luminosity Functions*. 2013. <<http://cvrl.ucl.ac.uk/lumindex.htm>>. Acessado em 31/12/2013.

DAWSON, C.; PATTANAM, S.; ROBERTS, D. The verilog procedural interface for the verilog hardware description language. In: *Proceedings of the 1996 IEEE International Verilog HDL Conference, 1996*. [S.l.: s.n.], 1996. p. 17–23. ISSN 1085-9403.

FALAKI, H. et al. Diversity in smartphone usage. In: . [S.l.]: ACM, 2010. (MobiSys '10), p. 179–194.

FATEMI, M.; ATEH, H.; SALLEH, R. A bit-serial sum of absolute difference accelerator for variable block size motion estimation of h. 264. In: *IEEE. Proceedings of the Innovative Technologies in Intelligent Systems and Industrial Applications, 2009. CITISIA 2009*. [S.l.], 2009. p. 1–4.

FUKANO, G. et al. A 65nm 1Mb SRAM macro with dynamic voltage scaling in dual power supply scheme for low power SoCs. In: *Proceedings of the Joint Non-Volatile Semiconductor Memory Workshop, 2008 and 2008 International Conference on Memory Technology and Design. NVSMW/ICMTD 2008*. [S.l.: s.n.], 2008. p. 97–98.

GAO, X.; DUANMU, C.; ZOU, C. A multilevel successive elimination algorithm for block matching motion estimation. *IEEE Transactions on Image Processing*, v. 9, n. 3, p. 501–504, 2000. ISSN 1057-7149.

GARIPELLY, R.; KIRAN, P. M.; KUMAR, A. S. A review on reversible logic gates and their implementation. *International Journal of Emerging Technology and Advanced Engineering Volume*, v. 3, n. 3, 2013.

GAZZANIGA, M.; IVRY, R.; MANGUN, G. *Cognitive Neuroscience: The Biology of the Mind*. [S.l.]: W. W. Norton, Incorporated, 2002. ISBN 9780393927061.

GHANBARI, M. *Standard Codecs: Image compression to advanced video coding*. [S.l.]: Iet, 2003.

GROSS, C.; BENDER, D.; ROCHA-MIRANDA, C. Visual receptive fields of neurons in inferotemporal cortex of the monkey. *Science*, v. 166, n. 910, p. 1303–1306, 1969.

- HAGLUND, L. *The SVT High Definition Multi Format Test Set*. 2006. <http://media.xiph.org/video/derf/vqeg.its.blrdoc.gov/HDTV-SVT_MultiFormat/SVT_MultiFormat_v10.pdf>. Acessado em 27.02.2014.
- HE, Z.-L. et al. Low-power vlsi design for motion estimation using adaptive pixel truncation. *IEEE Transactions on Circuits and Systems for Video Technology*, IEEE, v. 10, n. 5, p. 669–678, 2000.
- HECHT, E. *Optics*. Second editon. Boston: Addison Wesley, 1982.
- HESTERBERG, T. et al. *Bootstrap Methods and Permutation Tests – Companion Chapter 18 to the Praticce of Business Statistics*. New York: W. H. Freeman and Company, 2003.
- HUANG, Y. et al. Reconfigurable sad tree architecture based on adaptive sub-sampling in hdtv application. In: *Proceedings of the 19th ACM Great Lakes Symposium on VLSI*. New York, NY, USA: ACM, 2009. (GLSVLSI '09), p. 463–468. ISBN 978-1-60558-522-2.
- HUANG, Y.-W. et al. An efficient and low power architecture design for motion estimation using global elimination algorithm. In: *Proceedings of the 2002 IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*. [S.l.]: IEEE, 2002. v. 3, p. III–3120–III–3123.
- HUANG, Y.-W. et al. Hardware architecture design for variable block size motion estimation in mpeg-4 avc/jvt/itu-t h. 264. In: *Proceedings of the 2003 International Symposium on Circuits and Systems. ISCAS'03*. [S.l.]: IEEE, 2003. v. 2, p. II–796.
- HUNT, R. W. G.; POINTER, M. R. *Measuring Colour (The Wiley-IS&T Series in Imaging Science and Technology)*. [S.l.]: Wiley, 2011. 492 p. ISBN 1119975379.
- IEC. *Quantities and units — Part 13: Information science and technology*. Genebra, 2008.
- ITU-T. *Parameter values for the HDTV standards for production and international programme exchange*. Genebra, maio 2008.
- ITU-T. *H.264 Corrigendum 1*. jan 2009.
- ITU-T. *Studio encoding parameters of digital television for standard 4:3 and wide-screen 16:9 aspect ratios*. Genebra, mar. 2011.
- ITU-T. *Methodology for the subjective assessment of the quality of television pictures*. Genebra, 2012.

ITU-T. *Parameter values for ultra-high definition television systems for production and international programme exchange*. Genebra, 2012.

ITU-T. *Recommendation ITU-T H.265: High efficiency video coding*. Genebra, 2013.

ITU/ICT. *ICT Facts and Figures*. 2013.

ITU/ICT. *ITU World Telecommunication /ICT Indicators database*. 2013.

JACK, K.; TSATSULIN, V. *Dictionary of video and television technology*. [S.l.]: Gulf Professional Publishing, 2002. ISBN 9781878707994.

JAIN, J.; JAIN, A. Displacement measurement and its application in interframe image coding. *IEEE Transactions on Communications*, v. 29, n. 12, p. 1799–1808, 1981.

JEHNG, Y.-S.; CHEN, L.-G.; CHIUEH, T.-D. An efficient and simple vlsi tree architecture for motion estimation algorithms. *IEEE Transactions on Signal Processing*, IEEE, v. 41, n. 2, p. 889–900, 1993.

JONAS, J. B. et al. Human optic nerve fiber count and optic disc size. *Investigative ophthalmology & visual science*, ARVO, v. 33, n. 6, p. 2012–2018, 1992.

JUNG, H.-K. et al. A vlsi architecture for the alternative subsampling-based block matching algorithm. *IEEE Transactions on Consumer Electronics*, v. 41, n. 2, p. 239–247, may 1995.

JUNIOR, R. C. F.; KANAAN, A.; GOMES, J. M. S. de M. *As Ferramentas do Astrônomo: O que medimos, como medimos e o que aprendemos*. 2002. <<http://www.telescopiosnaescola.pro.br/ferramentas.pdf>>.

JVT. *JM JOINT VIDEO TEAM Reference Software*. 2011. <<http://iphome.hhi.de/suehring/tml/>>. Acessado em 15/05/2011.

KAMACI, N.; ALTUNBASAK, Y. Performance comparison of the emerging h.264 video coding standard with the existing standards. In: *Proceedings of the 2003 International Conference on Multimedia and Expo, 2003. ICME '03*. [S.l.]: IEEE, 2003. v. 1, p. I–345–8 vol.1.

KANDEL, E. R. et al. *Principles of Neural Science, Fifth Edition*. [S.l.]: McGraw-Hill Professional, 2012. 1760 p.

- KAO, C.-Y.; WU, C.-L.; LIN, Y.-L. A high-performance three-engine architecture for h. 264/avc fractional motion estimation. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, IEEE, v. 18, n. 4, p. 662–666, 2010.
- KRISHNAVENI, D.; PRIYA, M. G.; BASKARAN, K. Design of an efficient reversible 8x8 wallace tree multiplier. *World Applied Sciences Journal*, v. 20, n. 8, p. 1159–1165, 2012.
- KUHN, P. M. Fast mpeg-4 motion estimation: Processor based and flexible vlsi implementations. *The Journal of VLSI Signal Processing*, Springer Netherlands, v. 23, p. 67–92, 1999.
- LARBIER, P. Using 10-bit AVC/H.264 encoding with 4:2:2 for broadcast contribution. In: . Las Vegas Convention Center, Las Vegas, Nevada USA: [s.n.], 2011. <<http://extranet.ateme.com/download.php?file=1114>>. Acessado em 12/06/2011.
- LARKIN, D.; MURESAN, V.; O'CONNOR, N. A low complexity hardware architecture for motion estimation. In: *Proceedings of the 2006 IEEE International Symposium on Circuits and Systems, 2006. ISCAS 2006*. [S.l.]: IEEE, 2006. p. 4–pp.
- LEE, K. et al. QME: an efficient subsampling-based block matching algorithm for motion estimation. In: *Proceedings of the 2004 International Symposium on Circuits and Systems, 2004. ISCAS '04*. [S.l.]: IEEE, 2004. v. 2, p. II– 305–8 Vol.2.
- LENGWEHASARIT, K.; ORTEGA, A. Probabilistic partial-distance fast matching algorithms for motion estimation. *IEEE Transactions on Circuits and Systems for Video Technology*, v. 11, n. 2, p. 139 –152, feb 2001.
- LI, W.; SALARI, E. Successive elimination algorithm for motion estimation. *IEEE Transactions on Image Processing*, v. 4, n. 1, p. 105–107, jan 1995.
- LIU, Z. et al. 32-Parallel SAD tree hardwired engine for variable block size motion estimation in HDTV1080P Real-Time encoding application. In: *Proceedings of the 2007 IEEE Workshop on Signal Processing Systems*. [S.l.]: IEEE, 2007. p. 675–680. ISBN 978-1-4244-1222-8.
- LOZA, A. et al. Structural similarity-based object tracking in video sequences. In: *Proceedings of the Information Fusion*. [S.l.: s.n.], 2006. p. 1 –6.

MANJUNATHA, D.; SAINARAYANAN, G. Power efficient sum of absolute difference algorithms for video compression. *IOSR Journal of VLSI and Signal Processing*, v. 1, n. 6, p. 10–18, 2013.

MANOEL, E. T. M. *Codificação de Vídeo H.264 - Estudo de Codificação Mista de Macroblocos*. Dissertação (Monografia) — UFSC, 2007.

MANZATO, M. G. *Técnicas e Padrões de Codificação de Vídeo*. Dissertação (Trabalho de Conclusão de Curso) — UEL, 2004.

MATHEWS, N. M.; MUSSER, C.; ART, W. C. M. of. *Moving pictures: American art and early film, 1880-1910*. [S.l.]: Hudson Hills, 2005.

MERRITT, L. *x264 commit 25b40141a3d6569bfdc58a94d3004a89211029d6*. 2005.

MERRITT, L. et al. *Projeto x264*. 2004. <<http://developers.videolan.org/x264.html>>. Acessado em 10/05/2012.

MERRITT, L.; VANAM, R. *x264: A high performance h. 264/avc encoder*. 2006.

MERRITT, L.; VANAM, R. Improved rate control and motion estimation for h.264 encoder. In: *Proceedings of the 2007 IEEE International Conference on Image Processing (ICIP)*. [S.l.: s.n.], 2007. v. 5, p. V –309 –V –312. ISSN 1522-4880.

MINAMI, T. et al. A proposal of a one-dimensional array architecture for the full-search block matching algorithm. *IEICE Transactions on Inf. and Syst. (Japanese Edition)*, The Institute of Electronics, Information and Communication Engineers, v. 78, n. 12, p. 913–925, 1995.

MIYAKOSHI, J. et al. A low-power systolic array architecture for block-matching motion estimation. *IEICE Transactions on Electronics*, The Institute of Electronics, Information and Communication Engineers, v. 88, n. 4, p. 559–569, 2005.

MONTEIRO, J. L. *Hierarchical Add-One Carry-Select Adder: Um Somador Select-Adder com Cadeia de Carry Logarítmica*. Dissertação (Trabalho de Conclusão de Curso) — UFSC, 2011.

MORAES, B. G. de. *Uma métrica para Taxa de Distorção voltada à codificação de vídeo perceptiva*. Dissertação (Trabalho de Conclusão de Curso) — UFSC, 2010.

NIYAS, R. M.; GURU, M.; JAYAKRISHNAN, P. Implementation of sad architecture for motion estimation in h. 264/avc. *International Journal of Engineering and Technology*, 2013.

Nowell-Smith, G. *The Oxford History of World Cinema*. [S.l.]: Oxford University Press, USA, 1999.

OLIVARES, J. Reconfigurable vbsme architecture using rbsad. *Journal of Universal Computer Science*, v. 18, n. 2, p. 264–285, 2012.

PASCUTTO, G.-C. *bootstrap.py*. 2011. <<http://sjeng.org/ftp/bootstrap.py>>. Acessado em 12/12/2011.

PATHAK, A.; HU, Y. C.; ZHANG, M. Where is the energy spent inside my app?: fine grained energy accounting on smartphones with eprof. In: . [S.l.]: ACM, 2012. (EuroSys '12), p. 29–42.

PORTO, M. et al. High throughput hardware architecture for motion estimation with 4:1 pel subsampling targeting digital television applications. In: *Proceedings of the 2nd Pacific Rim conference on Advances in Image and Video Technology*. Berlin, Heidelberg: Springer-Verlag, 2007. (PSIVT'07), p. 36–47.

PORTO, M. et al. Motion estimation architecture using efficient adder-compressors for hdtv video coding. *Journal of Integrated Circuits and Systems*, v. 5, p. 78–88, 2010.

PORTO, M. S. et al. Architectural design for the new qsd with dynamic iteration control motion estimation algorithm targeting hdtv. In: ACM. *Proceedings of the 21st Annual Symposium on Integrated Circuits and System Design*. [S.l.], 2008. p. 216–221.

PORTO, M. S. et al. Iterative random search: a new local minima resistant algorithm for motion estimation in high-definition videos. In: . [S.l.: s.n.], 2013. p. 107–127.

POYNTON, C. *Digital Video and HD: Algorithms and Interfaces*. [S.l.]: Morgan Kaufmann, 2012. (Morgan Kaufmann Series in Computer Graphics and Geometric Modeling). ISBN 9780123919267.

QAZI, M. et al. A 512kb 8T SRAM macro operating down to 0.57 V with an AC-Coupled sense amplifier and embedded data-retention-voltage sensor in 45 nm SOI CMOS. *IEEE Journal of Solid-State Circuits*, v. 46, n. 1, p. 85–96, jan. 2011.

REHMAN, A.; WANG, Z. Ssim-inspired perceptual video coding for hevc. In: *Proceedings of the 2012 IEEE International Conference on Multimedia and Expo (ICME)*. [S.l.: s.n.], 2012. p. 497–502. ISSN 1945-7871.

RICHARDSON, I. E. *Video codec design: developing image and video compression systems*. [S.l.]: John Wiley and Sons, 2002.

RICHARDSON, I. E. *H. 264 and MPEG-4 video compression: video coding for next-generation multimedia*. [S.l.]: John Wiley & Sons Inc, 2003.

SAHA, A. Toward optimal pixel decimation patterns for block matching in motion estimation. In: *Proceedings of the ADCOM*. [S.l.: s.n.], 2007. p. 635–640.

SAHA, A.; MUKHERJEE, J.; SURAL, S. New pixel-decimation patterns for block matching in motion estimation. *EURASIP Signal Processing: Image Communication*, v. 23, n. 10, p. 725 – 738, 2008.

SANCHEZ, G. et al. Hardware design focusing in the tradeoff cost versus quality for the h. 264/avc fractional motion estimation targeting high definition videos. *Analog Integrated Circuits and Signal Processing*, Springer, v. 73, n. 3, p. 931–944, 2012.

SAPONARA, S.; FANUCCI, L. Data-adaptive motion estimation algorithm and vlsi architecture design for low-power video systems. *Computers and Digital Techniques, IEE Proceedings -*, v. 151, n. 1, p. 51–59, 2004. ISSN 1350-2387.

SEIDEL, I. et al. Exploring pel decimation to trade off between energy and quality in video coding. In: *Proceedings of the 2014 IEEE Latin American Symposium on Circuits and Systems (LASCAS)*. [S.l.]: IEEE, 2014.

SEIDEL, I. et al. Comparison of 90nm and 65nm logic synthesis of a sad configurable vlsi architecture. In: *Proceedings of the 28th South Symposium on Microelectronics (SIM)*. [S.l.]: SBC, 2013.

SEIDEL, I. et al. On the impacts of pel decimation and high-vt/low-vdd on sad calculation. In: *Proceedings of the 26th Symposium on Integrated Circuits and Systems Design (SBCCI)*. [S.l.]: IEEE, 2013.

SEIDEL, I.; MORAES, B. G.; GÜNTZEL, J. L. A low-power configurable vlsi architecture for sum of absolute differences calculation. In: *Proceedings of the 2013 IEEE Latin American Symposium on Circuits and Systems (LASCAS)*. [S.l.]: IEEE, 2013.

SEIDEL, I. et al. Quality assessment of subsampling patterns for pel decimation targeting high definition video. In: *Proceedings of the 2013 IEEE International Conference on Multimedia and Expo (ICME)*. [S.l.]: IEEE, 2013.

SEIDEL, I.; MORAES, B. G. de; GÜNTZEL, J. L. A. Evaluating the impact of carry-ripple and carry-lookahead adders in pel decimation vlsi implementations. In: *Proceedings of the 27th South Symposium on Microelectronics (SIM)*. [S.l.]: SBC, 2012.

SHARPE, L. T. et al. A luminous efficiency function, $v^*(\lambda)$, for daylight adaptation. *Journal of Vision*, Association for Research in Vision and Ophthalmology, v. 5, n. 11, 2005.

SHEPARD, C. et al. Livelab: measuring wireless networks and smartphone users in the field. *SIGMETRICS Perform. Eval. Rev.*, ACM, v. 38, n. 3, p. 15–20, 2011.

SHI, Y.; SUN, H. *Image and Video Compression for Multimedia Engineering: Fundamentals, Algorithms, and Standards*. [S.l.]: Taylor & Francis, 1999. (Image Processing Series). ISBN 9780849334917.

SOUZA, J. S. de; CARDOZA, J. A. S. Sensores de imagem digitais CCD e CMOS. In: *VII Congresso Norte Nordeste de Pesquisa e Inovação (CONNEPI)*. [S.l.]: Instituto Federal de Educação, Ciência e Tecnologia do Tocantins - IFTO, 2012.

STOCKMAN, A.; SHARPE, L. T. The spectral sensitivities of the middle-and long-wavelength-sensitive cones derived from measurements in observers of known genotype. *Vision research*, Elsevier, v. 40, n. 13, p. 1711–1737, 2000.

SULLIVAN, G. et al. Overview of the high efficiency video coding (HEVC) standard. *IEEE Transactions on Circuits and Systems for Video Technology*, v. 22, n. 12, p. 1649–1668, dez 2012.

SYNOPSISYS. *Synopsys Design Compiler, Version F-2011.09-SP5-2*. 2011.

SYNOPSISYS. *Synopsys VCS, Version G-2012.09*. 2012.

TAYLOR, J. *DVD Frequently Asked Questions (and Answers)*. 2011. <<http://www.dvddemystified.com/dvdfaq.html3.4.1>>. Acessado em 15/08/2011.

TAYLOR, J.; JOHNSON, M. R.; CRAWFORD, C. G. *DVD demystified*. [S.l.]: McGraw-Hill Professional, 2006. ISBN 9780071423960.

- TEAM, M. *Encoding with the x264 codec*. 2014. <<http://www.mplayerhq.hu/DOCS/HTML/en/menc-feat-x264.html>>. Acessado em 25.02.2014.
- TSMC. *TSMC STANDARD CELL Library TCBN45GSBWPTC*. [S.l.], 2011.
- TSMC. *TSMC STANDARD CELL Library TCBN45GSBWPTC*. [S.l.], 2011.
- UGUR, K. et al. Low complexity video coding and the emerging HEVC standard. In: *Proceedings of the 2012 Picture Coding Symposium (PCS)*. [S.l.]: IEEE, 2010. p. 474–477.
- URAMOTO, S.-i. et al. A half-pel precision motion estimation processor for ntsc-resolution video. *IEICE Transactions on Electronics*, The Institute of Electronics, Information and Communication Engineers, v. 77, n. 12, p. 1930–1936, 1994.
- VANNE, J. et al. A high-performance sum of absolute difference implementation for motion estimation. *IEEE Transactions on Circuits and Systems for Video Technology*, v. 16, n. 7, p. 876–883, jul 2006.
- VANWINKLE, R. et al. *Active Teaching – Active Learning. Teaching Techniques and Tools*. [S.l.], 2002.
- VASSILIADIS, S. et al. The sum-absolute-difference motion estimation accelerator. In: IEEE. *Proceedings of 24th the Euromicro Conference, 1998*. [S.l.], 1998. v. 2, p. 559–566.
- VIJAYAKUMAR. *A peek @ H.264*. 2008. Material de curso com Dr. K. R. Rao, em maio 2009.
- WALTER, F.; BAMPI, S. Synthesis and comparison of low-power architectures for sad calculation. *26th South Symposium on Microelectronics*, p. 45 – 48, Abril 2011.
- WALTER, F.; DINIZ, C.; BAMPI, S. Synthesis and comparison of low-power high-throughput architectures for sad calculation. *Analog Integrated Circuits and Signal Processing*, Springer US, v. 73, n. 3, p. 873–884, 2012.
- WALTER, F. L.; DINIZ, C. M.; BAMPI, S. Synthesis and comparison of low-power high-throughput architectures for SAD calculation. In: *Proceedings of the 2011 IEEE Second Latin American Symposium on Circuits and Systems (LASCAS)*. [S.l.]: IEEE, 2011. p. 1–4.

- WANG, C.-N. et al. A hierarchical n-queen decimation lattice and hardware architecture for motion estimation. *IEEE Transactions on Circuits and Systems for Video Technology*, v. 14, n. 4, p. 429 – 440, 2004.
- WANG, S. et al. Ssim-motivated rate-distortion optimization for video coding. *IEEE Transactions on Circuits and Systems for Video Technology*, v. 22, n. 4, p. 516–529, April 2012. ISSN 1051-8215.
- WANG, Z. et al. Image quality assessment: from error visibility to structural similarity. *IEEE Transactions on Image Processing*, v. 13, n. 4, p. 600 –612, 2004.
- WANG, Z. et al. Image quality assessment: From error visibility to structural similarity. *IEEE Transactions on Image Processing vol. 13 no. 4*, p. 13, apr 2004.
- WIEN, M. *High Efficiency Video Coding - Coding Tools and Specification*. 2013. Tutorial.
- WINKLER, S. *Digital video quality: vision models and metrics*. [S.l.]: John Wiley and Sons, 2005. ISBN 9780470024041.
- WOOTTON, C. *A practical guide to video and audio compression: from sprockets and rasters to macroblocks*. [S.l.]: Elsevier, 2005. ISBN 9780240806303.
- WU, H.; RAO, K. *Digital Video Image Quality and Perceptual Coding*. [S.l.]: CRC Press, 2005. ISBN 9780824727772.
- WYSZECKI, G.; STILES, W. S. *Color science: concepts and methods, quantitative data and formulae*. Second editon. New York: Wiley, 1982.
- XIPH.ORG. *Xiph.org Test Media Repository*. 2011. <<http://media.xiph.org/>>. Acessado em 25/01/2012.
- YAP, S. Y.; MCCANNY, J. V. A vlsi architecture for variable block size video motion estimation. *IEEE Transactions on Circuits and Systems II: Express Briefs*, IEEE, v. 51, n. 7, p. 384–389, 2004.
- YOUNG, D.; GREGORY, R. *A Survey of Numerical Mathematics*. [S.l.]: Dover Publications, 1988. (A Survey of Numerical Mathematics, v. 2). ISBN 9780486656922.
- YOUTUBE. *Statistics – YouTube*. 2013. <<http://www.youtube.com/yt/press/statistics.html>>. Acessado em 30/12/2013.

YUFEI, L.; XIUBO, F.; QIN, W. A High-Performance low cost SAD architecture for video coding. *IEEE Transactions on Consumer Electronics*, v. 53, n. 2, p. 535–541, may 2007.

ZHANG, K. et al. A fully synchronized, pipelined, and re-configurable 50 mb sram on 90 nm cmos technology for logic applications. In: *Proceedings of the Symposium on VLSI Circuits, 2003. Digest of Technical Papers*. [S.l.: s.n.], 2003. p. 253 – 254.

ZHU, C.; XIONG, B. Transform-exempted calculation of sum of absolute hadamard transformed differences. *IEEE Transactions on Circuits and Systems for Video Technology*, v. 19, n. 8, p. 1183–1188, 2009. ISSN 1051-8215.

**ApêndiceA – COMPARAÇÃO COM ARQUITETURA NÃO
CONFIGURÁVEL DE MESMO *DATAPATH***

Neste apêndice são apresentados os resultados de síntese da arquitetura configurável apresentada na Seção 4.2 e uma arquitetura não configurável, de mesmo *datapath*. Os resultados demonstraram que a configurabilidade tem impacto desprezível em termos de área e potência, resultando, por outro lado, em ganhos de até 60% no tempo de execução do cálculo da métrica de similaridade e, dessa forma, 60% de ganho em eficiência energética¹. Como mencionado no início do Capítulo 4, os resultados apresentados neste anexo foram publicados nos anais do LASCAS 2013 (SEIDEL; MORAES; GÜNTZEL, 2013).

A.1 Configuração Experimental

A arquitetura configurável e a arquitetura não configurável (sem subamostragem), cujo *datapath* e FSM estão descritos nas Figuras 21 e 22 respectivamente, foram descritas em Verilog e sintetizadas para biblioteca *standard cells* da *Taiwan Semiconductor Manufacturing Company Limited* (TSMC) 90nm (TSMC, 2011b) com a ferramenta *Synopsys® Design Compiler* (DC®) (SYNOPTIS, 2011) em modo *Topographical*. Ambas as arquiteturas foram sintetizadas para *Low-Vdd/High-Vt*², além de para tensão de alimentação e de *threshold* nominais. Outra variação de síntese é relativa à frequência. Adotou-se uma frequência alvo de 66,67MHz (similar ao trabalho de Walter, Diniz e Bampi (2011), para comparações) e também a frequência máxima atingida pela síntese de cada arquitetura, através de tentativa e erro³. As mesmas restrições de síntese apresentadas no Capítulo 4 são utilizadas para todas as sínteses apresentadas.

A.2 Resultados de Síntese

A Tabela 8 mostra estimativas de área e de potência para ambas as arquiteturas (configurável e não configurável) de acordo com os relatórios de síntese para frequência máxima, assim como suas respectivas eficiências energéticas. Também são mostrados os resultados apresentados por Walter, Diniz e Bampi (2011). Como pode ser observado pelos valores nas colunas

¹A eficiência energética é inversamente proporcional à energia/bloco.

²Todas as células utilizam a mesma tensão de alimentação e *threshold*, de modo diferente à técnica de *Multi-Vdd/Multi-Vt*, onde a ferramenta de síntese pode escolher entre *standard cells* com diferentes tensões.

³As sínteses para frequência máxima apresentadas nesta seção foram feitas manualmente, ou seja, os valores de restrição de período foram ajustados manualmente. Assim, a precisão não é tão alta quanto apresentado no Capítulo 4 e no Apêndice B.

A (não configurável), **B** (configurável) e **B/A**, a arquitetura proposta requer 1,8% maior área e demonstra potência similar à arquitetura não configurável, mostrando o baixo impacto da configurabilidade. As colunas **C**, **D** e **D/C** demonstram os resultados da síntese usando *Low-Vdd/High-Vt*. Neste caso, a arquitetura configurável requer 3,8% menos área e 4,4% menos potência que a arquitetura não configurável.

Os resultados apresentados dão indícios de que a arquitetura proposta se beneficiou mais da síntese usando *Low-Vdd/High-Vt* do que a não configurável. É importante notar que as sínteses nominais alcançaram a frequência de 800 MHz, resultando em *throughput* de 1,28GPixel/s, enquanto as sínteses *Low-Vdd/High-Vts* (LHs) chegaram apenas em 270,3 MHz, o que resulta em *throughput* de 432MPixel/s. Isto se traduz em uma degradação de desempenho de 45%. Resultados similares são obtidos no Apêndice B, os quais serão por sua vez justificados.

A Tabela 9 apresenta estimativas de área, potência e energia/bloco para ambas as arquiteturas (configurável e não configurável) a partir da síntese para frequência alvo de 66,67 MHz. Também apresenta os resultados apresentados por Walter, Diniz e Bampi (2011), para biblioteca TSMC 180nm e Walter e Bampi (2011), para biblioteca IBM 65nm. A coluna **B/A** mostra que a arquitetura configurável requer praticamente a mesma área e consome aproximadamente a mesma potência que a arquitetura não configurável, confirmando o pequeno impacto da configurabilidade. A coluna **D/C** mostra que estes cenários não mudam usa-se *Low-Vdd/High-Vt*.

É interessante também observar o impacto de *Low-Vdd/High-Vt* em relação à área. Para frequência máxima (Tabela 8) *Low-Vdd/High-Vt* resulta em 14% maior área para a arquitetura não configurável e 9% maior área para a arquitetura configurável. Para a frequência alvo de 66,67MHz (Tabela 9) *Low-Vdd/High-Vt* resulta em 18,5% maior área para ambas as arquiteturas. Finalmente, é válido observar os resultados de economia em termos de energia pelo uso de subamostragem nas três frequências operacionais consideradas. O que está de acordo com a Seção 4.2.

A.3 Conclusões

Os resultados de síntese apresentados nesta seção demonstram que os impactos da configurabilidade, ao comparar com uma arquitetura com *datapath* equivalente, são desprezíveis. Os resultados também apresentaram a redução de potência quando aplicadas a técnica *Low-Vdd/High-Vt*. Tais valores podem ser comparados de forma justa usando a mesma frequência, que é o caso da frequência alvo (aqui, 66,67 MHz): considerando a arquitetura

configurável, de $105,27\mu W$ reduziu para $51,8\mu W$ com *Low-Vdd/High-Vt*. Quando analisadas as frequências máximas, pôde-se observar de imediato a degradação de frequência causada pelo uso de *Low-Vdd/High-Vt*. Ainda, o uso de subamostragem, com a arquitetura configurável proposta, trouxe ganhos de desempenho e assim, reduções de energia/bloco de 40% (1:1/2:1) e 60% (1:1/4:1).

Estes aspectos abrem caminho para uma exploração mais aprofundada do espaço de síntese, para melhor entender o comportamento da arquitetura configurável. Tal exploração será apresentada na próxima seção.

Tabela 8: Comparação dos resultados de síntese para frequência máxima

		R1	A	B	B/A	C	D	D/C
Tecnologia (nm)		TSMC 180	TSMC 90	TSMC 90	–	TSMC 90	TSMC 90	–
Frequência (MHz)		204	800	800	1.000	270.27	270.27	1.000
Área (μm^2)		n.a.	5011.877	5104.311	1.018	5845.19	5622.927	0.962
Dinâmica		n.a.	1517.8	1524.9	1.005	244.473	233.691	0.956
Estática		n.a.	12.752	13.049	1.023	1.092	1.025	0.939
Total		8065	1530.552	1537.949	1.005	245.565	234.716	0.956
Potência (μW)	Amostragem Completa	158.074	19.132	19.224	1.005	9.086	8.684	0.956
	Pel 2:1	n.a.	n.a.	11.535	–	n.a.	5.211	–
	Pel 4:1	n.a.	n.a.	7.69	–	n.a.	3.474	–
Energia/bloco ($pJ/block$)	Amostragem Completa	158.074	19.132	19.224	1.005	9.086	8.684	0.956
	Pel 2:1	n.a.	n.a.	11.535	–	n.a.	5.211	–
	Pel 4:1	n.a.	n.a.	7.69	–	n.a.	3.474	–

Considerando: **R1** = Resultados de Walter, Diniz e Bampi (2011); **A** = Arquitetura não-configurável nominal; **B** = Arquitetura configurável proposta nominal; **C** = Não-configurável com *Low-Vdd/High-Vt*; **D** = Configurável com *Low-Vdd/High-Vt*.

Tabela 9: Comparação dos resultados de síntese para frequência alvo

		R1	A	B	B/A	R2	C	D	D/C
Tecnologia (nm)		TSMC 180	TSMC 90	TSMC 90	–	IBM 65	TSMC 90	TSMC 90	–
Frequência (MHz)		66	66.67	66.67	1.000	66	66.67	66.67	1.000
Área (μm^2)		n.a.	3025.613	3027.73	1.001	n.a.	3710.045	3717.1	1.002
Dinâmica		n.a.	100.197	99.0639	0.989	n.a.	51.537	51.2166	0.994
Estática		n.a.	6.219	6.2042	0.998	n.a.	0.579	0.5785	0.999
Total		1276	106.416	105.268	0.989	108	52.116	51.795	0.994
Potência (μW)	Amostragem Completa	76.56	15.962	15.790	0.989	6.48	7.817	7.769	0.994
	Pel 2:1	n.a.	n.a.	9.474	–	n.a.	n.a.	4.662	–
	Pel 4:1	n.a.	n.a.	6.316	–	n.a.	n.a.	3.108	–
Energia/bloco ($pJ/block$)	Amostragem Completa	76.56	15.962	15.790	0.989	6.48	7.817	7.769	0.994
	Pel 2:1	n.a.	n.a.	9.474	–	n.a.	n.a.	4.662	–
	Pel 4:1	n.a.	n.a.	6.316	–	n.a.	n.a.	3.108	–

Considerando: **R1** = Resultados de Walter, Diniz e Bampi (2011); **A** = Arquitetura não-configurável nominal; **B** = Arquitetura configurável proposta nominal; **R2** = Resultados de Walter, Diniz e Bampi (2011); **C** = Não-configurável com *Low-Vdd/High-Vt*; **D** = Configurável com *Low-Vdd/High-Vt*.

ApêndiceB – EXPLORAÇÃO DO ESPAÇO DE SÍNTESE

Como o número de pixels por quadro tende a aumentar nos novos padrões de codificação de vídeos de alta definição como o HEVC, a *pel decimation* aparece como uma forma viável de aumentar a eficiência energética no cálculo da SAD. Tal fato foi demonstrado na seção anterior, através de uma proposta de arquitetura configurável para operar sob as razões de amostragem 1:1, 2:1 e 4:1. Esta seção apresenta a síntese para bibliotecas de *standard cells* de 130nm, 90nm, 65nm e 45nm, assumindo tanto a operação com tensão Nominal (NN) quanto LH, para *throughput* máximo e para um determinado alvo (diferente do apresentado na Seção A). Foram analisados os impactos da sub-amostragem e LH no atraso, potência e eficiência energética, totalizando 16 sínteses. Como mencionado no início do Capítulo 4, os resultados apresentados neste apêndice foram publicados nos anais do SBCCI 2013 (SEIDEL et al., 2013b).

B.1 Configuração Experimental

A arquitetura de SAD configurável foi descrita em Verilog e sintetizada logicamente, com DC[®] (SYNOPTIS, 2011), para bibliotecas de *standard cells* TSMC 130nm, 90nm, 65nm e 45nm, para duas tensões de operação, Nominal (NN) e *Low-Vdd/High-Vt* (LH), e para duas frequências, alvo (“tar”) e máxima (“max”). A frequência alvo foi definida como 160Mhz, o que resulta em um *throughput* de 1 milhão de macroblocos/s. Tal *throughput* é o mesmo considerado por Walter, Diniz e Bampi (2011) e, segundo os autores, é o suficiente para codificar um vídeo 1080p@30fps. Dessa forma, a comparação com estes trabalhos deixa de ser para a mesma frequência, e passa a ser para o mesmo *throughput*. A frequência máxima corresponde ao *throughput* máximo que a arquitetura atinge quando sintetizada para uma dada tecnologia e uma dada tensão de operação. Os valores de frequências máximas e seus respectivos *throughputs* são apresentados na Tabela 10.

Tabela 10: Frequências (MHz), *throughput* (Mblocos/s) e degradação (%)

	130nm	90nm	65nm	45nm
NN	545,3/136,3	870,7/217,7	1434,2/358,5	1855,1/463,8
LH	407,3/101,8	640,4/160,1	1201,9/300,5	1404,7/351,2
<i>degr.</i> (%)	25,3	26,5	16,1	24,3

B.2 Resultados das Sínteses e Análise

A Figura 34 mostra os resultados para a arquitetura configurável de SAD. É possível observar que, para qualquer tecnologia, os circuitos sintetizados para a frequência alvo são sempre menores que os sintetizados para frequência máxima. Isso ocorre porque a frequência alvo é menor que qualquer frequência máxima (mostrado na Tabela 10) e portanto, a restrição de atraso para a frequência alvo pode ser atingida usando *standard cells* mais lentas (i.e., *standard cells* com menor capacidade de carga na saída), as quais são menores que suas equivalentes mais rápidas.

Comparando NN e LH para frequência alvo, pode-se observar que as sínteses para LH tiveram aumento de área de 6%, 18%, < 1% e 3% para as tecnologias de 130nm, 90nm, 65nm e 45nm, respectivamente, enquanto que, para frequências máximas, as sínteses para LH resultaram em acréscimo de área de 4.3%, 7.8%, 14% e 5%, respectivamente. Genericamente, tais acréscimos de área devem-se ao fato de *standard cells* LHs serem mais lentas do que suas equivalentes NNs e consequentemente, devem ser empregadas *standard cells* LH mais rápidas (que são maiores), para alcançar as restrições de atraso crítico. Também podem ser empregadas transformações nas sínteses, como por exemplo duplicação de lógica, para reduzir o *fanout* das portas. Vale mencionar que *standard cells* em tecnologias nanométricas, tais quais 65nm e 45nm, apresentam atrasos muito baixos e assim, o aumento do atraso crítico absoluto das *standard cells*, resultante do uso de LH, é pequeno. Dessa forma, as sínteses para frequência alvo com LH para 65nm e 45nm conseguiram atingir os limites de atraso crítico com poucas modificações de *standard cells* e/ou *standard cells* ligeiramente mais rápidas, enquanto que para 130nm e 90nm foram necessárias mais mudanças nas *standard cells* e/ou *standard cells* muito mais rápidas precisaram ser empregadas.

A respeito das sínteses para frequência máxima, o uso de LH também resultou em aumento de área para todas as tecnologias analisadas, com 65nm exibindo o maior aumento. O fato de cada versão do circuito ter a sua frequência máxima específica dificulta muito uma comparação direta entre diferentes tecnologias. Porém, analisando cuidadosamente a Tabela 10 pode-se observar que 65nm é o caso com o maior aumento na frequência máxima em relação à tecnologia imediatamente mais antiga (90nm, neste caso). Isto indica que a ferramenta de síntese foi capaz de explorar melhor o espaço de soluções devido a características específicas da biblioteca de 65nm e então, as sínteses para frequência máxima com LH para 65nm resultaram na menor degradação de frequência, calculada usando a Equação B.1, mas com custo

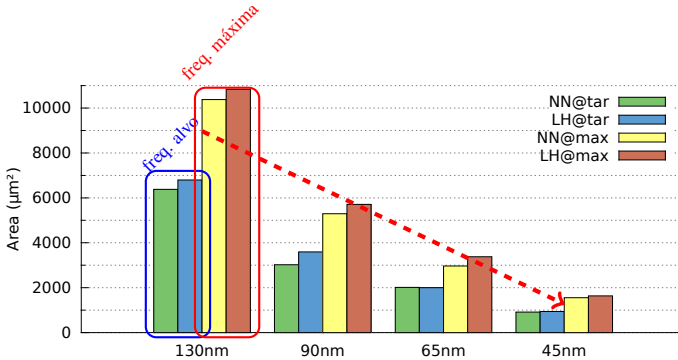


Figura 34: Resultados para exploração do espaço de síntese da arquitetura de SAD configurável. A linha pontilhada indica a tendência de decréscimo das áreas de acordo com o nodo tecnológico. Para a síntese de 130nm está indicado a organização em relação à frequência no gráfico. Assim, cada par corresponde à tensão nominal ou *low-Vdd/high-Vt* para determinada frequência, alvo ou máxima.

de 14% maior área, o maior acréscimo dentre todas sínteses LH.

$$\text{degradação} = 1 - \frac{\text{maxFreq@LowVdd_HighVt}}{\text{maxFreq@Nominal}} \quad (\text{B.1})$$

A Figura 35 apresenta os valores absolutos de potência total para a arquitetura configurável de SAD, enquanto a Figura 36 apresenta as potências dinâmica e estática, como percentual da potência total. Da Figura 36 pode-se observar que a potência dinâmica foi dominante em todos os casos e portanto, a potência total é determinada principalmente pelo atraso crítico do circuito. Uma vez que a frequência alvo é no máximo três vezes menor que a frequência máxima, já era esperado que os circuitos para frequência alvo consumissem significativamente menos potência que circuitos para frequência máxima.

Considerando as sínteses para frequência alvo, foram observadas pequenas reduções na potência ao aplicar LH, para 130nm, 65nm e 45nm. Para 90nm, por sua vez, ocorreu inclusive um pequeno aumento na potência. Isto porque as folgas de tempo (*slacks*) consideradas dos circuitos para baixa frequência (alvo) foram exploradas mais efetivamente nas sínteses NN para atingir baixo consumo de energia e de alguma forma as sínteses para LH não compensaram pelo aumento do atraso das *standard cells*. É interessante notar que, enquanto a potência dinâmica é dominante, a potência estática não

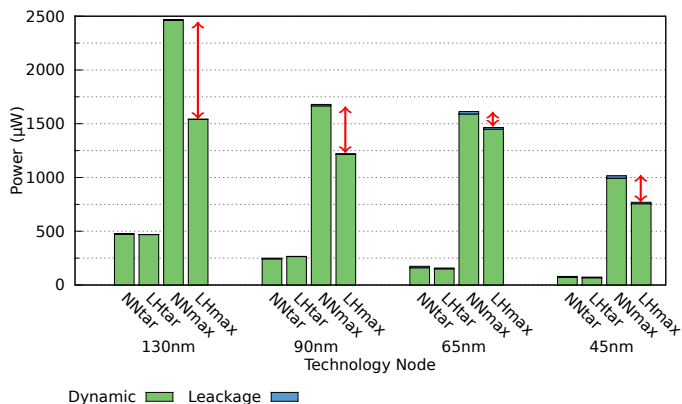


Figura 35: Resultados de potência total (dinâmica e estática). As setas demonstram o decréscimo da potência total ao aplicar LH nas sínteses para frequência máxima. É importante lembrar que, ao aplicar LH, ocorre degradação de frequência, conforme Tabela 10, sendo tal degradação grande responsável pela diminuição da potência dinâmica e por consequência também potência total.

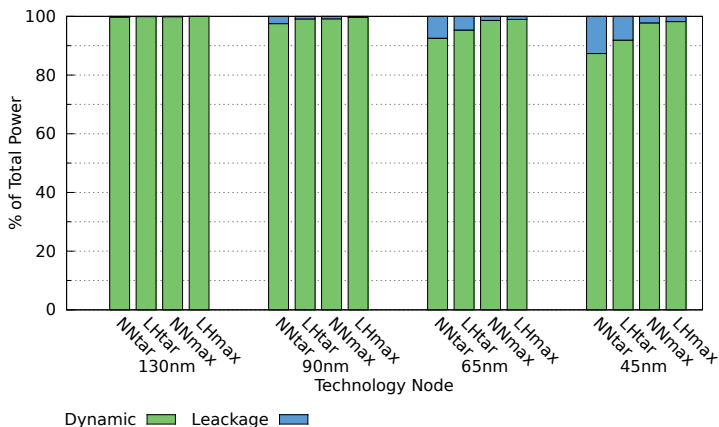


Figura 36: Resultados de potência dinâmica e estática relativas à potência total (%). É possível notar que para os nodos com menor tamanho de *gate*, o impacto da potência estática se torna maior. Também é visível que ao aplicar LH tal impacto é reduzido.

é desprezível para 65nm e 45nm na frequência alvo. Nesses casos específicos, o aumento em V_t reduziu com sucesso a potência estática, contribuindo consequentemente para a redução na potência total.

Da Figura 35 fica claro que os maiores benefícios de LH foram obtidos nas sínteses para frequência máxima, sendo 38%, 27%, 9% e 24% de redução de potência total para as tecnologias de 130nm, 90nm, 65nm e 45nm, respectivamente. Porém, deve-se lembrar que o uso de LH gerou degradação na frequência máxima, como mostrado na Tabela 10 e portanto, os atrasos críticos do circuito são menores, contribuindo para a redução de potência dinâmica.

Para avaliar as vantagens alcançadas pelo uso da arquitetura configurável SAD, é conveniente analisar-se a eficiência energética quando se opera com cada um dos três tipos de amostragens (1:1, 2:1 e 4:1). As Figuras 37 e 38 mostram a energia necessária para processar um bloco 4x4 com frequência alvo e máxima, respectivamente. Tais valores de energia foram calculados através da Equação B.2 (reapresentada para facilitar a leitura), onde T é o período de relógio (obtido ou da frequência alvo ou máxima), C é o número de ciclos para processar um bloco 4x4 e P é a potência total reportada pelo DC[®].

$$E = T * C * P \quad (\text{B.2})$$

Como detalhado na Seção 4.2, o menor número de ciclos necessários para processar um bloco 4x4 com amostragem 1:1, 2:1 ou 4:1 é de 10, 6 ou 4, respectivamente, assumindo apenas uma execução para os estados LOAD e DONE. Portanto, demonstra-se aqui que para uma versão qualquer de síntese (combinação de frequência, tecnologia e tensão) a redução de energia alcançada por subamostragens 2:1 e 4:1 em relação à 1:1 também é de 40% e 60%, respectivamente. Essas reduções são facilmente observadas nos gráficos apresentados nas Figuras 37 e 38. Os impactos da tecnologia e tensão na energia da arquitetura configurável de SAD são analisados separadamente para frequências alvo e máxima nos dois parágrafos a seguir.

Confrontando a Figura 37 com a Figura 38 fica claro que as versões para frequência alvo são mais eficientes energeticamente do que suas equivalentes para frequência máxima. A principal razão para isso é a baixa frequência alvo (ao menos três vezes menor que qualquer outra frequência máxima). Além disso, uma vez que o período é o mesmo para todas as versões para frequência alvo ($T = 6,25ns$) e o número de ciclos é $C = \{10,6,4\}$ para os padrões de amostragem 1:1, 2:1 e 4:1, respectivamente, qualquer redução/aumento da energia é consequência direta de uma redução/aumento na potência total (P). Dessa forma, a Figura 37 meramente reapresenta as informações para frequências alvo apresentadas na Fig 35. Isto explica a redução

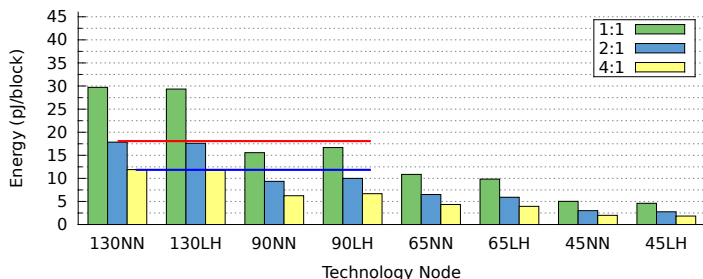


Figura 37: Resultados de energia/bloco para frequência alvo.

imperceptível de energia alcançada por LH com 130nm, 65nm e 45nm e o aumento de energia para 90nm.

Como mostrado na Figura 38, as versões para frequência máxima pagam um custo significativo em energia e conseqüentemente, podem se beneficiar do uso de LH. Considerando 130nm, que é a versão menos energeticamente, aplicar LH resultou em uma redução de energia de 15,5% por operação. Porém, para 90nm e 45nm as reduções foram muito pequenas, enquanto que para 65nm, observou-se aumento de energia. No caso de 130nm, a redução total de potência foi de aproximadamente 38%, enquanto a degradação de frequência foi de 25,3% e então, a redução de potência se sobressaiu sobre a degradação de frequência. Para 90nm, observou-se uma pequena redução de energia (0,0012%), uma vez que potência e frequência reduziram praticamente pelo mesmo fator. Por outro lado, a redução de potência para 65nm foi menor que a respectiva degradação de frequência e portanto, a energia aumentou. Finalmente, para 45nm, a degradação de frequência e a potência foram praticamente da mesma ordem, resultando na mesma energia por bloco para ambas as versões, NN e LH (5,48pJ/bloco para amostragem 1:1).

Pode-se observar o impacto de *pel decimation* com a arquitetura configurável de SAD, analisando cuidadosamente os gráficos das Figuras 37 e 38. Para qualquer tecnologia, a subamostragem trouxe reduções de energia muito mais significativas do que as obtidas sintetizando com LH. Essa tendência é naturalmente mais proeminente nas versões para frequência máxima. Apesar de LH poder ser usado em conjunto com *pel decimation*, o aumento de área não deve ser negligenciado. Por outro lado, para as atuais e futuras altas resoluções, a redução de qualidade proveniente do uso de *pel decimation* tende a ser menos perceptível, como apresentado no Capítulo 3. Além disso, o usuário/aplicação pode escolher entre obter uma codificação com maior qualidade

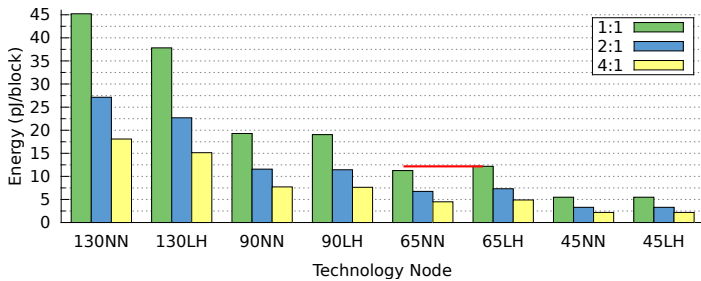


Figura 38: Resultados de energia/bloco para frequências máximas.

ou maior economia de energia.

B.3 Comparação com Trabalhos Correlatos

Dentre os trabalhos correlatos existentes, aqueles de Walter, Diniz e Bampi (2011) e Walter e Bampi (2011) apresentam resultados para arquiteturas de SAD sintetizadas para as bibliotecas TSMC 180nm/NN e IBM 65nm/LH. As características específicas desses trabalhos foram tratadas na Seção 4.1. Para estabelecer uma comparação justa, foram selecionadas as arquiteturas de 4 entradas sem pipeline, projetadas para atingir um *throughput* de 1 milhão de macroblocos/s, que é o mesmo *throughput* das sínteses com frequência alvo apresentadas.

A Tabela 11 mostra as informações de tecnologia e resultados das sínteses (frequência, potência e energia) para todas as arquiteturas comparadas. Uma vez que a FSM da arquitetura apresentada tem dois estados a mais (LOAD e DONE) que a arquitetura correlata, a frequência alvo deve ser maior para atingir o mesmo *throughput*.

Comparando a arquitetura de Walter, Diniz e Bampi (2011) (TSMC 180/NN) com a arquitetura configurável, com TSMC 130/NN, é possível observar redução de energia de 2,6 vezes, o que é maior do que o fator de *scaling* entre essas duas tecnologias. Apesar de que tal comparação apenas dá uma ideia de possíveis melhorias, muito provavelmente esconde o impacto da tecnologia, tornando-a não confiável. Por outro lado, pode-se estabelecer uma comparação mais justa entre as sínteses de 65nm/LH. Em tais casos, a arquitetura configurável, operando sem subamostragem, apresenta potência total maior do que Walter e Bampi (2011). Porém, deve-se notar que a arquitetura configurável é capaz de reduzir a energia por bloco através da subamostra-

Tabela 11: Comparação com trabalhos correlatos para mesmo *throughput*, onde **A** apresenta os resultados de Walter, Diniz e Bampi (2011), **B** são os resultados aqui apresentados da síntese 130nm, **C** são os resultados de Walter e Bampi (2011) e **D** são os resultados aqui apresentados da síntese para 65nm.

	A	B	C	D
Tech. (nm)	TSMC 180	TSMC 130	IBM 65	TSMC 65
NN/LH	NN	NN	LH	LH
Freq. (MHz)	66	160	66	160
Potência (μW)	1276	475.64	108	157.65
1:1 (pJ/block)	76,56	29,73	6,48	9,85
2:1 (pJ/block)	–	17,84	–	5,91
4:1 (pJ/block)	–	11,89	–	3,94

gem, o que não é possível na arquitetura de Walter e Bampi (2011). Mais especificamente, enquanto a arquitetura de Walter e Bampi (2011) requer $6.48 pJ/block$ para 1:1, a arquitetura configurável precisa de $5.91 pJ/block$ para 2:1 e apenas $3.94 pJ/block$ para subamostragem 4:1.

B.4 Conclusões

Mostrou-se novamente que, ao configurar a arquitetura para operar com razões de subamostragem 2:1 e 4:1, a energia gasta para cada bloco pode ser reduzida em 40% e 60%, respectivamente, em relação a amostragem completa. O maior consumo de energia foi de $45,22 pJ/block$ e ocorreu na síntese de 130nm para frequência máxima operando com tensão nominal (NN) e amostragem completa. O menor consumo de energia foi de $2,19 pJ/block$ e corresponde à síntese de 45nm para frequência alvo, tanto para NN quanto LH, operando com subamostragem 4:1. Porém, no caso das sínteses com LH, houve aumento de área de aproximadamente 3% em relação as sínteses NNs. A síntese para 130nm/NN e frequência máxima gasta aproximadamente 20,64 vezes mais energia por bloco do que a síntese para 45nm com frequência alvo.

Os resultados das sínteses também mostraram que, em alguns casos, o impacto de LH para essas arquiteturas simples pode resultar em pior eficiência energética, além do aumento na área. Também mostrou-se que 130nm obteve maior vantagem de LH do que as tecnologias mais recentes.

Finalmente, em comparação com Walter e Bampi (2011), a arquitetura apresentada alcançou resultados piores de eficiência energética para amostra-

gem completa, devido sua FSM possuir estados de sincronização. Por outro lado, a arquitetura apresentada pode ser configurada para executar subamostragem 2:1 ou 4:1, o que é responsável por economias de energia relevantes.

ApêndiceC – Lista de Publicações

1. LASCAS 2013 – 4th IEEE Latin American Symposium on Circuits and Systems;

- Qualis CC:** B5;
- Data:** Fevereiro-Março, 2013;
- Título:** A Low-Power Configurable VLSI Architecture for Sum of Absolute Differences Calculation;
- Abstract:** *This paper presents a new configurable VLSI architecture for Sum of Absolutes Differences (SAD) calculation with pel decimation capabilities. It was also described a non-configurable architecture for comparison. The proposed SAD architecture as well as a non-configurable architecture were synthesized to both nominal and Low-Vdd/High-Vt versions of a commercial 90nm technology. Synthesis results demonstrated that the configurability comes at the cost of negligible area and power overhead. In addition, pel decimation improvements of up to 60% in latency and energy efficiency were observed. It was also observed that the configurable architecture benefit more from the Low-Vdd/High-Vt synthesis than the non-configurable architecture.*

2. SIM 2013 – 28th South Symposium on Microelectronics;

- Data:** Abril-Maio, 2013;
- Título:** Comparison of 90nm and 65nm Logic Synthesis of a SAD Configurable VLSI Architecture;
- Abstract:** *This paper evaluates the impact of the technology node on the area, performance and power consumption of a configurable VLSI architecture for Sum of Absolutes Differences (SAD). Such architecture may be configured to take benefit from pel decimation, trading off quality for energy. The proposed architecture was synthesized for 90nm and 65nm technologies assuming both nominal and Low-Vdd/High-Vt (LH) cases. Results showed that pel decimation itself is responsible for up to 60% of energy efficiency with a negligible area and power overhead with respect to a non-configurable reference SAD architecture. Our best result, using pel decimation 4:1 in 65nm/LH spends only 2.6 pJ for each 4x4 block, which corresponds to about 7.4 times less energy when compared with our proposed architecture in 90nm/nominal operating in full SAD mode.*

3. ICME 2013 – IEEE International Conference on Multimedia and Expo;

- Qualis CC:** A1;
- Data:** Julho, 2013;
- Título:** Quality Assessment of Subsampling Patterns for Pel Decimation Targeting High Definition Video;
- Abstract:** *Although pel decimation has been widely used to reduce the computational effort in video coding, there is no consensus about the optimal subsampling pattern. This paper presents an extensive analytical and statistical comparison of several different subsampling patterns using analysis of variance. The investigation includes commonly used patterns as well as seven proposed ones. The experiments were conducted on 19 video samples in a range of five resolutions (being nine videos at 1080p) and 30 bitrates by using the state-of-the-art x264 encoder running successive elimination exhaustive search. Two objective quality metrics were reported, PSNR and DSSIM, resulting in 10680 experimental points. The analysis of such amount of data allowed us to conclude that the proposed 4:3 ratio shows less than 5% in DSSIM and 1% in PSNR losses, being more than two times faster than full sampling. Compared with higher decimation, it presents a better trade-off between speedup and quality loss.*

4. SBCCI 2013 – 26th Symposium on Integrated Circuits and Systems Design;

- Qualis CC:** B1;
- Data:** Setembro, 2013;
- Título:** On the Impacts of Pel Decimation and High-Vt/Low-Vdd on SAD Calculation;
- Abstract:** *As the number of pixels per frame tends to increase in new high definition video coding standards such as HEVC, pel decimation appears as a viable means of increasing the energy efficiency of Sum of Absolute Differences (SAD) calculation. This paper presents a VLSI architecture that can be configured to compute the SAD of 4x4 pixel blocks with no subsampling or with 2:1 or 4:1 subsampling (pel decimation). The proposed architecture was synthesized for 130nm, 90nm, 65nm and 45nm standard cell libraries assuming both nominal and Low-Vdd/High-Vt (LH) cases for maximum and a given target throughput. The impacts of*

subsampling and Low-Vdd/High-Vt on delay, power and energy efficiency are analyzed. In a total of 16 syntheses, the 45nm/LH configurable SAD architecture achieved the highest energy efficiency for target frequency when operating in pel decimation 4:1, spending only 2.19pJ for each 4x4 block, which corresponds to about 20.64 times less energy than the 130nm/nominal configurable architecture operating in full SAD mode. Aside the improvements achieved by using LH, pel decimation solely was responsible for energy reductions of 40% and 60% when 2:1 and 4:1 subsamplings are chosen, respectively, in the configurable architecture.

5. LASCAS 2014 – 5th IEEE Latin American Symposium on Circuits and Systems;

•**Qualis CC:** B5;

•**Data:** Fevereiro, 2014;

•**Título:** Exploring Pel Decimation to Trade off Between Energy and Quality in Video Coding;

•**Abstract:** *This work investigates the trade-offs between energy and quality in video coding when pel decimation is applied. Realistic estimates for area and energy per block were obtained by simulating five different architectures specially designed to compute the Sum of Absolute Differences (SAD) for 4x4 pixel blocks. Among these architectures, one can be configured to operate with 1:1, 4:3, 2:1 or 4:1 sample ratios, whereas the rest are tailored to operate exclusively with each one of those ratios. The five VLSI architectures were logically synthesized for a 45nm industrial standard cell library for a target frequency and also for the maximum achievable frequency. They were also simulated with 100k input vectors obtained by using an H.264/AVC encoder running on one full HD (1080p) video sample. The obtained results show that by using the configurable architecture with full sampling, the best energy/block result was 3.54pJ/block (60% better than the non-configurable with 7.08pJ/block). The energy/block value can be further reduced until 1.34pJ/block at the cost of 2.8% in PSNR, on average, and 14.1% in SSIM, on average.*