

UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL  
INSTITUTO DE INFORMÁTICA  
PROGRAMA DE PÓS-GRADUAÇÃO EM MICROELETRÔNICA

DIEISON SOARES SILVEIRA

**Gestão e Aproximação de Dados em  
Sistemas de Memórias com NV-RAMs  
Dedicados a Codificadores de Vídeo  
Eficientes**

Tese apresentada como requisito parcial para a  
obtenção do grau de Doutor em Microeletrônica

Orientador: Prof. Dr. Sergio Bampi

Co-orientador: Prof. Dr. Marcelo Schiavon Porto

Porto Alegre  
2021

## CIP — CATALOGAÇÃO NA PUBLICAÇÃO

Silveira, Dieison Soares

Gestão e Aproximação de Dados em Sistemas de Memórias com NV-RAMs Dedicados a Codificadores de Vídeo Eficientes / Dieison Soares Silveira. – Porto Alegre: PGMICRO da UFRGS, 2021.

203 f.: il.

Tese (doutorado) – Universidade Federal do Rio Grande do Sul. Programa de Pós-Graduação em Microeletrônica, Porto Alegre, BR-RS, 2021. Orientador: Sergio Bampi; Co-orientador: Marcelo Schiavon Porto.

1. Codificação de vídeo, computação aproximada, gestão de memória, tecnologias de memória NV-RAMs, redução do consumo de energia. I. Bampi, Sergio. II. Porto, Marcelo Schiavon. III. Título.

UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL

Reitor: Prof. Carlos André Bulhões Mendes

Vice-Reitora: Prof<sup>a</sup>. Patricia Helena Lucas Pranke

Pró-Reitor de Pós-Graduação: Prof. Júlio Otávio Jardim Barcellos

Diretora do Instituto de Informática: Prof<sup>a</sup>. Carla Maria Dal Sasso Freitas

Coordenador do PGMICRO: Prof. Tiago Roberto Balen

Bibliotecária-chefe do Instituto de Informática: Beatriz Regina Bastos Haro

*Dedico esta tese à minha filha,  
Cecília.*

## AGRADECIMENTOS

Agradeço a minha família, minha esposa Cristiane e minha filha Cecília. Minha esposa, grande amiga e companheira inseparável nessa longa jornada, a qual teve início em 2008 quando entramos juntos na graduação na UFPel. Minha filha Cecília que chegou em nossas vidas no início desse trabalho, trazendo um mundo de novidades para os pais de primeira viagem. Sou muito feliz por ter vocês ao meu lado me dando incentivo e motivação diariamente, me aguentando nos momentos mais difíceis. Obrigado por estarem sempre presentes e por fazerem parte da minha vida. Amo vocês.

Aos meus familiares, meus pais e meu irmão, que mesmo a quilômetros de distância acompanharam minha trajetória, comemorando as minhas conquistas. Obrigado pelo incentivo constante para que eu continuasse nos momentos de adversidades, e pela compreensão nos meus momentos de ausência.

Aos meus orientadores, Sergio Bampi e Marcelo Porto, que participaram ativamente na construção deste trabalho, estando sempre à disposição para conselhos e ensinamentos. Obrigado pela confiança, amizade, ajuda e paciência durante a orientação.

Aos colegas do Lab 215 pelos momentos de descontração e de trabalho, em especial aos meus queridos Mateus Grellert, Leonardo Soares, Eduarda Monteiro, Ana Mativi, Brunno Abreu, Leandro Mateus e Guilherme Paim. Além dos meus parceiros de pesquisa, Lívia e Guilherme, que mesmo distantes conseguimos manter nosso time ativo. Tenham certeza que vocês foram fundamentais no desenvolvimento deste trabalho. Meu muito obrigado e saibam que nossas parcerias vão além desse trabalho.

Ao Instituto Federal de Ciência e Tecnologia do Rio Grande do Sul em especial ao Campus Canoas, onde sou professor desde 2017, pelo seu Programa de Capacitação de Servidores, o qual me oportunizou tempo para que eu pudesse concluir essa tese.

Agradeço muito a todos/as os/as professores/as que fizeram parte de toda minha formação, desde o ensino fundamental na Escola Municipal Osvaldo Aranha, o ensino médio no Colégio Estadual Sete de Setembro, o curso técnico no Instituto Estadual Cónego Luiz Walter Hanquet, a graduação e o mestrado na Universidade Federal de Pelotas e o doutorado na Universidade Federal do Rio Grande do Sul. Agradeço muito a todos/as esses/as servidores/as públicos/as que se desdobram no seu dia a dia para levar um ensino público, gratuito e de qualidade a todos seus alunos, principalmente nesse período de precarização e desvalorização da educação e da pesquisa.

Muito obrigado a todos!

*“Se a educação sozinha não transforma a sociedade,  
sem ela tampouco a sociedade muda.”*

— PAULO FREIRE

## RESUMO

A codificação e a decodificação de vídeos digitais estão presentes na maioria das aplicações multimídia, principalmente nos sistemas embarcados atuais, atingindo bilhões de usuários principalmente com a popularização dos dispositivos móveis, como *smartphones* e *tablets*. Além disso, as melhorias nos serviços de *streaming* motivam os usuários a gerarem, transmitirem e visualizarem conteúdos em vídeos digitais de altíssima resolução pela Internet como 4K e 8K. Nesse sentido, sistemas de codificação de vídeo atuais vêm exigindo largura de banda com a memória cada vez maiores para codificar um único quadro do vídeo. Entre os principais módulos dos codificadores de vídeo atuais, um dos módulos que mais acessa a memória é a Estimção de Movimento (ME). A ME exige uma grande largura de banda de memória, a qual é utilizada para ler e escrever os quadros de referência na memória. Esse processo acaba gerando um elevado consumo de energia, uma vez que os acessos à memória externa são as operações que mais dissipam potência nos sistemas digitais atuais. Desta forma, essa tese avalia o impacto em consumo de energia e eficiência de codificação da utilização conjunta de estratégias de reuso de dados, compressão de quadros de referência, computação aproximada e memórias não-voláteis (NV-RAMs). Para estimar os resultados dessas estratégias foi desenvolvido em software e hardware um sistema eficiente de redução de energia relacionada à memória para arquiteturas de codificadores de vídeo digitais. Para isso, esse sistema utiliza uma hierarquia de memória composta por tecnologia NV-RAMs, a qual utiliza o reuso de dados Level C e *power gating* em bancos das memórias internas. Ainda, o sistema possui quatro níveis de configuração, permitindo o uso de computação aproximada através da redução do tamanho da área de busca e *bit dropping* nas amostras de referência. O *bit dropping* é utilizado juntamente com um eficiente compressor de quadros de referência. Os resultados gerados para quatro configurações mostram que o sistema atinge altas taxas de redução de energia, os quais variam de 88% até 95%, quando comparados a um sistema com reuso de dados Level C, sendo essas as maiores taxas de redução em consumo de energia entre todos os trabalhos estado da arte encontrados na literatura. Para atingir essas taxas de redução de energia, o sistema apresenta uma variação na eficiência de codificação de 0,03% até 3,3%.

**Palavras-chave:** Codificação de vídeo, computação aproximada, gestão de memória, tecnologias de memória NV-RAMs, redução do consumo de energia.

# Data Management and Approximation in Memory Systems with NV-RAMs Dedicated to Efficient Video Encoders

## ABSTRACT

The process of coding and decoding digital videos are presents in most multimedia applications, especially in today's embedded systems, reaching billions of users mainly due to the popularization of mobile devices such as smartphones and tablets. In addition, the improvements in the streaming services motivate users to generate, stream and view ultra-high-resolution digital videos over the Internet as 4K and 8K. In this way, current video coding systems require a growing external memory bandwidth to encode a single video frame. Among the main modules of the current video encoders, the module that performs intensive memory access is the Motion Estimation (ME). The ME requires a large memory bandwidth, which is mostly used for reading and write the reference frames in the memory. This operation generates high energy consumption since memory accesses are one of the main power-demanding elements in current digital systems. Therefore, this thesis evaluates the impact on energy consumption and coding efficiency of the common use of data reuse strategies, reference frame compression, approximate computation, and non-volatile memories (NV-RAMs). To estimate the results of these strategies an efficient memory energy reduction system for video encoder architectures was developed in software and hardware. Therefore, this system uses a memory hierarchy composed of NV-RAMs technology, employing the Level C data reuse scheme and *power gating* in the internal memory banks. Furthermore, the system has four configuration levels, allowing approximate computation by reducing the search area size and *bit dropping* in the reference samples. The *bit dropping* is used with an efficient reference frame compressor. The results achieved by these configurations show that the system reaches high energy reduction rates varying from 88% up to 95% when compared to a baseline system that employs Level C data reuse. Besides, the energy reduction results surpasses all state-of-the-art works found in the literature. To achieve these energy savings rates, the system has a coding efficiency loss of 0.03% up to 3.3%.

**Keywords:** video encoding, approximate computing, memory management, NV-RAMs technologies, energy consumption reduction.

## LISTA DE ABREVIATURAS E SIGLAS

ACF	Autocorrelation Function
AFLR	Autocorrelation-Based Frame Lossless Recompression
AOM	Alliance for Open Media
APBT	Adaptive Prefix Bit Truncation
ASIC	Application Specific Integrated Circuits
AVC	Advanced Video Coding
BD	Bit Dropping
BD-rate	Bjøntegaard Delta Bitrate
BD-PSNR	Bjøntegaard Delta Peak Signal-to-Noise Ratio
CABAC	Context Adaptive Binary Arithmetic Coding
CCT	Condições Comuns de Teste
CD	Codificação Diferencial
CIF	Common Intermediate Format
CTU	Coding Tree Units
CU	Coding Unit
DCT	Discrete Cosine Transform
DDR	Double Data Rate
DDRFC-BD	Double Differential Reference Frame Coder with Bit Dropping
DDRFVLC	Double Differential Reference Frame Variable-Length Coder
DMMA	Dual-Mode Memory Addressing
DPB	Decoded Image Buffer
DPCM	Differential Pulse Code Modulation
DRAM	Dynamic Random-Access Memory
DWT	Discrete Wavelet Transform

EDA	Electronic design automation
FeFET	Ferroelectric Field-Effect Transistor
FeRAM	Ferroelectric Random Access memory
HACP	Hierarchical Average and Copy Prediction
HCC	Head Code Compression
HD	High-definition
HEVC	High Efficiency Video Coding
HMD	Hierarchical Minimum and Difference
HM	HEVC Model
ITU-T	International Telecommunication Union
JEM	Joint Exploration Model
JVET	Joint Video Experts Team
LCU	Largest Coding Unit
MC	Motion Compensation
MDLR	Multi-Directional Lossless Recompression
MDP	Multi-Direction Prediction
ME	Motion Estimation
MERSy	Memory Energy Reduction System
MMSQ	Min-Max-Scalar-Quantization
MPEG	Moving Picture Experts Group
MRAM	Magnetoresistive Random Access Memory
ms	Milissegundos
MSE	Mean Squared Error
MTJ	Magnetic Tunnel Junction
nm	Nanômetro
NV-RAM	Non-Volatile Random Access Memory

PCRAM	Phase Change Random Access Memory
PD	Pattern Decision
PLE	Physically-Aware Layout Estimation
PSNR	Peak Signal-to-Noise Rate
Pt	Platina
PU	Prediction unit
QP	Quantization Parameter
RAM	Random Access Memory
ReRAM	Resistive Random Access Memory
RGB	Red Green Blue
SAD	Sum of Absolute Differences
SBT	Significant Bit Truncation
SBTVD	Sistema Brasileiro de TV Digital
SFL	Semi-Fixed Length Coding
SPIHT	Set Partitioning in Hierarchical Trees
SPM	Scratchpad Memory
SRAM	Static Random Access Memory
STT-MRAM	Spin-Transfer Torque Magnetic Random Access Memory
TCF	Toggle Count Format
Ti	Titânio
TiN	Nitreto de Titânio
TZS	Test Zone Search
UHD	Ultra High Definition
VCEG	Video Coding Experts Group
VGA	Video Graphics Array
VHDL	VHSIC Hardware Description Language

VLC	Variable Length Coding
VTM	Versatile Test Model
VVC	Versatile Video Coding
YCbCr	Luminance, Chrominance Blue, Chrominance Red

## LISTA DE FIGURAS

Figura 1.1 Aumento na resolução dos vídeos: Em 2023, 66% das TVs de tela plana serão UHD 2160p.....	19
Figura 1.2 Resultados de simulação para todos os <i>presets</i> do codificador x265, sendo em (a) os resultados de performance e em (b) resultados de <i>bitrate</i> vs consumo de energia. ....	21
Figura 1.3 Distribuição de acessos à Cache do codificador HEVC. ....	23
Figura 2.1 Diagrama de blocos simplificado do Codificador HEVC.....	32
Figura 2.2 (a) Quatro CTUs com suas respectivas <i>quadtree</i> e (b) divisões das PUs. ....	34
Figura 2.3 Modos de predição intra-quadro no HEVC. ....	35
Figura 2.4 Custo computacional do codificador HEVC por bloco. ....	39
Figura 2.5 Definição da área de busca em torno do bloco atual. ....	40
Figura 2.6 Largura de banda de memória utilizando Level C.....	42
Figura 2.7 Processo de expansão do algoritmo de busca TZS.....	45
Figura 3.1 Organização interna de uma memória com a árvore-H. ....	49
Figura 3.2 Estrutura do <i>array</i> de dados da memória.....	51
Figura 3.3 Frequência de operação dos processadores e do barramento das memórias DRAM ao longo das últimas décadas. ....	52
Figura 3.4 Níveis de uma hierarquia de memória.....	53
Figura 3.5 Comparativo de consumo de energia por acesso entre diferentes configurações de memórias Cache e memórias <i>scratchpad</i> .....	55
Figura 3.6 Célula de memória DRAM.....	59
Figura 3.7 Célula de memória SRAM com 6 transistores e a indicação dos caminhos de <i>leakage</i> .....	61
Figura 3.8 (a) Diagrama de uma célula PCRAM. (b) Perfil de temperatura do material calcogeneto nas operações de SET e RESET. (c) Mudanças de fase entre os estados amorfo e cristalino. ....	64
Figura 3.9 (a) MTJ nos estados paralelo e anti-paralelo (b) Célula STT-MRAM. ....	65
Figura 3.10 (a) Estrutura 1T1R de uma típica célula ReRAM (b) Imagem microscópica de alta resolução da célula ReRAM. ....	66
Figura 3.11 Ilustração da funcionalidade de um FeFET tipo n.....	69
Figura 3.12 Ilustração de memórias não-voláteis de acesso aleatório usando óxido de háfnio ferroelétrico e antiferroelétrico. ....	70
Figura 4.1 Comunicação utilizando a técnica de reuso de dados.....	73
Figura 4.2 Estratégias de reuso de dados propostas em Tuan, Chang e Jen (2002): (a) Level A, (b) Level B, (c) Level C e (d) Level D.....	74
Figura 4.3 Estratégia de reuso de dados com Level C+.....	75
Figura 4.4 Comunicação utilizando a compressão de quadros de referência. ....	77
Figura 4.5 Computação aproximada aplicada na codificação de vídeo.....	86
Figura 5.1 Histogramas com acessos à área de busca $192 \times 192$ para vídeos 1080p. ....	95
Figura 5.2 Histogramas com acessos à área de busca $192 \times 192$ para vídeos 2160p. ....	96
Figura 5.3 Relação entre quantidade de acessos e área acessada.....	97
Figura 5.4 Histograma com a média dos acessos à área de busca $192 \times 192$ para os vídeos HD 1080p e UHD 2160p. ....	99
Figura 5.5 Histogramas de acessos na área de busca com a média para os 4 QPs para o vídeo HD 1080p BasketballDrive. ....	103

Figura 5.6	Resultados de BD-rate para as configurações utilizando <i>bit dropping</i> .....	106
Figura 5.7	Comparativo de resultados BD-rate entre as três abordagens utilizando computação aproximada. ....	110
Figura 5.8	Comparativo de resultados de redução de área entre as três abordagens utilizando computação aproximada. ....	111
Figura 5.9	Resultados comparativos de redução de acessos entre as três abordagens utilizando computação aproximada. ....	113
Figura 5.10	Frente de Pareto com resultados de BD-rate e redução de acessos para as três abordagens utilizando computação aproximada. ....	114
Figura 6.1	MERSy: sistema de redução de energia da memória em codificadores de vídeo.....	120
Figura 6.2	Exemplo de <i>bit dropping</i> com deslocamento de 1 bit e do processo da dupla codificação diferencial. ....	125
Figura 6.3	Exemplo do processo da dupla codificação diferencial inversa.....	129
Figura 6.4	Histogramas das amostras dos vídeos, sendo (a) vídeo HD 1080p <i>BasketballDrive</i> , (b) vídeo UHD 2160p <i>CatRobot</i> e (c) acumulado com as 12 seqüências de vídeo HD e UHD. ....	130
Figura 6.5	Histogramas dos resíduos após o <i>bit dropping</i> e a dupla codificação diferencial para as quatro configurações do sistema, sendo (a) 48-7bpp, (b) 32-7bpp, (c) 16-7bpp e (d) 16-4bpp.....	132
Figura 6.6	Arquitetura de hardware do módulo codificador do DDRFC-BD.....	136
Figura 6.7	Arquitetura de hardware do módulo decodificador do DDRFC-BD.....	138
Figura 6.8	Processo de reuso de dados Level C entre duas áreas de busca.....	144
Figura 6.9	Arquitetura da memória interna composta por 44 bancos com 512B de capacidade cada. ....	147
Figura 6.10	Organização da memória externa .....	148
Figura 6.11	Histograma das ocorrências de palavras de 32 bits nos blocos $8 \times 8$ codificado.....	149
Figura 6.12	Resultados de consumo de energia total (dinâmica e estática) por quadro (em <i>mJ</i> ) para todas as combinações de tecnologias de memória para o reuso de dados Level C. ....	156
Figura 6.13	Comparativo entre o reuso de dados Level C tradicional (DRAM e SRAM) e o Level C NV-RAM excluindo resultados de configurações com PCRAM. ....	157
Figura 7.1	Resultados de redução no consumo de energia entre as quatro configurações. ....	168
Figura 7.2	Comparativo entre os três principais resultados para as quatro configurações. ....	172
Figura A.1	Histogramas com acessos à área de busca para quatro vídeos 1080p. ....	189
Figura A.2	Histogramas com acessos à área de busca para quatro vídeos 2160p. ....	190

## LISTA DE TABELAS

Tabela 5.1	Resultados médios de <i>bitrate</i> e PSNR para os 3 intervalos de busca.....	93
Tabela 5.2	Informações sobre os padrões de corte da área de busca. ....	100
Tabela 5.3	Resultados médios da variação de BD-rate (em %) para as 9 configurações de corte na área de busca. ....	101
Tabela 5.4	Resultados médios da variação de BD-rate para as 8 configurações de <i>bit dropping</i> . ....	105
Tabela 5.5	Resultados médios da variação de BD-rate para as 12 configurações da abordagem utilizando cortes na área de busca e <i>bit dropping</i> .....	108
Tabela 6.1	Resultados médios de taxa de compressão para as 4 configurações com 6 tamanhos de tabela estática. ....	134
Tabela 6.2	Tabela com códigos de <i>Huffman</i> para o intervalo $[-8, +8]$ .....	135
Tabela 6.3	Resultados de síntese lógica para a tecnologia ST 65nm com 1.0V de tensão das arquiteturas de hardware do DDRFC-BD .....	141
Tabela 6.4	Comparativo entre os dados das memórias internas utilizando somente o Level C e Level C mais as quatro configurações com cortes na área de busca e <i>bit dropping</i> . ....	146
Tabela 6.5	Resultados de consumo de energia das memórias externas.....	152
Tabela 6.6	Resultados de consumo de energia das memórias internas. ....	153
Tabela 7.1	Resultados de taxas de compressão obtidas pelo módulo DDRFC-BD para as 4 configurações do MERSy, utilizando 24 sequências de vídeo.....	161
Tabela 7.2	Resultados de redução de acessos às memórias externa e interna para as 4 configurações do MERSy.....	163
Tabela 7.3	Resultados de redução de consumo de energia da memória para as 4 configurações do MERSy.....	167
Tabela 7.4	Resultados médios da variação de BD-rate do MERSy com as 4 configurações para as 24 sequências de vídeo. ....	170
Tabela 7.5	Comparativos com trabalhos relacionados .....	174
Tabela B.1	Resultados de redução de acessos para as configurações com cortes na área de busca. ....	191
Tabela B.2	Resultados de redução de acessos para as configurações com <i>bit dropping</i> . ....	192
Tabela B.3	Resultados de redução de acessos para as configurações com <i>bit dropping</i> e cortes na área de busca de tamanho 48. ....	192
Tabela B.4	Resultados de redução de acessos para as configurações com <i>bit dropping</i> e cortes na área de busca de tamanho 32. ....	193
Tabela B.5	Resultados de redução de acessos para as configurações com <i>bit dropping</i> e cortes na área de busca de tamanho 16. ....	193
Tabela C.1	Resultados de taxa de compressão em relação ao <i>bit dropping</i> , considerando as 4 configurações com 6 tamanhos de tabela estática.....	195
Tabela C.2	Resultados de taxa de compressão em relação as amostras originais, considerando as 4 configurações com 6 tamanhos de tabela estática. ....	196
Tabela D.1	Sequências HD 1080p utilizadas para os experimentos e para os resultados. ....	198
Tabela D.2	Sequências HD 1080p utilizadas para os resultados.....	199

Tabela D.3 Sequências HD 2160 <i>p</i> utilizadas para os experimentos e para os resultados. ....	200
Tabela D.4 Sequências HD 2160 <i>p</i> utilizadas para os resultados.....	201

## SUMÁRIO

<b>1 INTRODUÇÃO</b> .....	<b>18</b>
<b>1.1 Formulação do problema e discussões</b> .....	<b>20</b>
<b>1.2 Hipótese de tese</b> .....	<b>24</b>
<b>1.3 Organização da tese</b> .....	<b>25</b>
<b>2 FUNDAMENTOS DA CODIFICAÇÃO DE VÍDEO</b> .....	<b>27</b>
<b>2.1 Características dos vídeos digitais</b> .....	<b>28</b>
<b>2.2 Padrões de codificação de vídeo</b> .....	<b>29</b>
<b>2.3 Padrão HEVC</b> .....	<b>32</b>
2.3.1 Fluxo de codificação .....	32
2.3.2 Etapa das predições .....	35
2.3.3 Transformadas, quantização e entropia .....	37
2.3.4 Reconstrução dos quadros de referência .....	38
<b>2.4 Estimação de movimento e o acesso à memória externa</b> .....	<b>38</b>
2.4.1 Algoritmo de busca <i>Test Zone Search</i> – TZS .....	43
<b>2.5 Métricas de qualidade e eficiência de codificação</b> .....	<b>46</b>
<b>2.6 Considerações finais do capítulo</b> .....	<b>47</b>
<b>3 FUNDAMENTOS DO SISTEMA DE MEMÓRIA E SUAS TECNOLOGIAS</b> .....	<b>49</b>
<b>3.1 Sistema de memória</b> .....	<b>49</b>
<b>3.2 Hierarquia de memória</b> .....	<b>51</b>
3.2.1 Memória Cache .....	53
3.2.2 Memória scratchpad .....	56
<b>3.3 Tecnologias de memória</b> .....	<b>57</b>
3.3.1 Memórias voláteis .....	58
3.3.1.1 <i>Dynamic Random Access Memory</i> – DRAM .....	59
3.3.1.2 <i>Static Random Access Memory</i> – SRAM .....	60
3.3.2 Memórias não voláteis .....	62
3.3.2.1 <i>Phase-Change Random-Access Memory</i> – PCRAM .....	63
3.3.2.2 <i>Spin-Transfer Torque Magnetic Random Access Memory</i> – STT-MRAM .....	64
3.3.2.3 <i>Resistive Random Access Memory</i> – ReRAM .....	66
3.3.2.4 <i>Ferroelectric Random Access memory</i> – FeRAM .....	68
<b>3.4 Desafios relacionados às tecnologias de memórias</b> .....	<b>70</b>
<b>4 TRABALHOS RELACIONADOS</b> .....	<b>72</b>
<b>4.1 Reuso de dados</b> .....	<b>72</b>
<b>4.2 Compressão de quadros de referência</b> .....	<b>77</b>
4.2.1 Compressão com perdas de dados .....	78
4.2.2 Compressão sem perdas de dados .....	80
<b>4.3 Intervalo de busca dinâmico</b> .....	<b>82</b>
<b>4.4 Memórias não voláteis</b> .....	<b>83</b>
<b>4.5 Computação aproximada</b> .....	<b>85</b>
<b>4.6 Síntese dos trabalhos relacionados</b> .....	<b>88</b>
<b>5 ANÁLISE DE EFICIÊNCIA DE CODIFICAÇÃO DO TZS</b> .....	<b>90</b>
<b>5.1 Metodologia</b> .....	<b>90</b>
<b>5.2 Definição do intervalo de busca</b> .....	<b>92</b>
<b>5.3 Análise dos acessos à área de busca</b> .....	<b>93</b>
<b>5.4 Computação aproximada no TZS</b> .....	<b>97</b>
5.4.1 Cortes na área de busca .....	98
5.4.2 <i>Bit dropping</i> .....	104
5.4.3 Corte na área de busca e <i>bit dropping</i> .....	106

5.4.4 Comparativo entre as técnicas de computação aproximada.....	109
<b>5.5 Considerações finais do capítulo.....</b>	<b>116</b>
<b>6 SISTEMA DE REDUÇÃO DE ENERGIA DA MEMÓRIA .....</b>	<b>119</b>
<b>6.1 Modelagem do consumo de energia.....</b>	<b>120</b>
<b>6.2 Compressor de quadros de referência com <i>bit dropping</i> .....</b>	<b>124</b>
6.2.1 Algoritmo DDRFC-BD.....	124
6.2.2 Análise da distribuição das amostras e definição da tabela de <i>Huffman</i> .....	130
6.2.3 Arquitetura de hardware do DDRFC-BD .....	135
6.2.4 Síntese lógica das arquiteturas .....	139
<b>6.3 Organização das memórias e reuso de dados .....</b>	<b>143</b>
6.3.1 Organização da memória interna .....	143
6.3.2 Organização da memória externa.....	147
<b>6.4 Caracterização das memórias para diferentes tecnologias .....</b>	<b>150</b>
6.4.1 Aplicação na codificação de vídeo.....	155
<b>7 RESULTADOS E DISCUSSÕES.....</b>	<b>159</b>
<b>7.1 Reduções de acessos às memórias externa e internas .....</b>	<b>160</b>
<b>7.2 Redução do consumo de energia.....</b>	<b>165</b>
<b>7.3 Eficiência de codificação .....</b>	<b>169</b>
<b>7.4 Comparativo com trabalhos relacionados .....</b>	<b>173</b>
<b>8 CONCLUSÕES .....</b>	<b>177</b>
<b>8.1 Principais contribuições .....</b>	<b>177</b>
<b>8.2 Oportunidades de trabalhos futuros .....</b>	<b>179</b>
<b>REFERÊNCIAS .....</b>	<b>180</b>
<b>APÊNDICE A — HISTOGRAMAS DOS ACESSOS À ÁREA DE BUSCA.....</b>	<b>189</b>
<b>APÊNDICE B — RESULTADOS DE ACESSOS DAS TÉCNICAS DE COM- PUTAÇÃO APROXIMADA.....</b>	<b>191</b>
<b>APÊNDICE C — RESULTADOS DE TAXA DE COMPRESSÃO PARA OS DIFERENTES TAMANHOS DE TABELA ESTÁTICA DE <i>HUFFMAN</i> .....</b>	<b>194</b>
<b>APÊNDICE D — SEQUÊNCIAS DE VÍDEO .....</b>	<b>197</b>
<b>APÊNDICE E — PUBLICAÇÕES .....</b>	<b>202</b>

## 1 INTRODUÇÃO

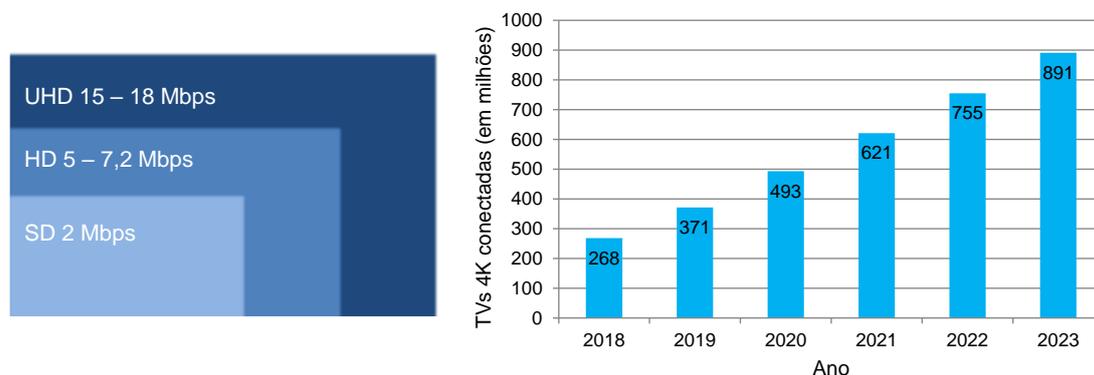
Os dispositivos digitais estão sempre em uma evolução constante, onde *smartphones* e *tablets* são produtos extremamente complexos e têm um tempo de vida no mercado muito curto, sendo constantemente atualizados. Naturalmente, esta corrida tecnológica em ritmo acelerado afeta o mercado de bens de eletrônica de consumo, estabelecendo uma crescente demanda por dispositivos móveis mais baratos e mais poderosos, melhor qualidade em serviços de mídia – em especial os serviços de áudio/vídeo cuja qualidade é um diferencial importante para o consumidor final – entre outras características dos dispositivos de computação móvel pessoal.

As aplicações de vídeo digital são um exemplo típico desse cenário. O hábito de compartilhar informações através de texto em uma página da web está constantemente sendo substituída por conteúdo de vídeo em sites como Youtube, ou por vídeos ao vivo em redes sociais. Além disso, devido à pandemia da COVID-19, as videoconferências e videoaulas tornaram-se essenciais para o desenvolvimento mais seguro das atividades econômicas e atividades de ensino. Aliados a isso, as crescentes melhorias nas tecnologias de comunicação, tornando-as mais acessíveis e de melhor qualidade, oferecendo maior largura de banda, além de vídeos estéreos ou até mesmo vídeos 3-D, tornam as aplicações baseadas em vídeo essenciais nos dias atuais.

Para quantificar esta tendência, um artigo publicado pela Cisco mostra que o tráfego de vídeo na Internet crescerá quatro vezes de 2017 a 2022, representando 82% do tráfego total da Internet até o final do período de previsão (CISCO, 2019). Ainda, os *smartphones* representarão 44% do tráfego IP total, acima dos 18% em 2017 (CISCO, 2019), e a quantidade de dados armazenados em dispositivos móveis será 4,5 vezes maior do que os dados armazenados em data centers (CISCO, 2019). Fato que é relevante, dada a memória limitada e a disponibilidade de energia em dispositivos móveis.

O efeito de aplicações de vídeo no tráfego de Internet é mais acentuado devido a introdução de ultra-alta definição (*Ultra High Definition (UHD)* –  $3840 \times 2160$  pixels), ou UHD 2160p, em *streaming* de vídeo. Essa tecnologia tem esse efeito porque a taxa de bits para vídeos codificados em UHD 2160p é em torno de 15 a 18 Mbps, isso é mais do que o dobro da taxa de bits de vídeos HD ( $1920 \times 1080$  pixels) e nove vezes mais do que a taxa de bits de vídeos de definição padrão (SD –  $720 \times 480$  pixels). Estimativas apontam que em 2023, dois terços (66%) dos aparelhos de TV de tela plana instalados serão UHD 2160p, contra 33% em 2018 (CISCO, 2020). A Figura 1.1 apresenta as estimativas de

Figura 1.1: Aumento na resolução dos vídeos: Em 2023, 66% das TVs de tela plana serão UHD 2160p.



Fonte: Cisco (2020).

crescimento na utilização da resolução 4K.

As principais desvantagens que advirão desta tendência de uso de vídeos de forma crescente é que, com o grande aumento no uso de vídeos digitais de resoluções cada vez maiores, aliado com a necessidade de *displays* de resoluções e taxas de quadros maiores, haverá um incremento muito grande na largura de banda necessária para transmissão de vídeos, bem como uma grande demanda de capacidade de memória para armazená-los. Além disso, com o desenvolvimento de dispositivos móveis e de Internet das Coisas (*Internet-of-Things* – IoT), os serviços de vídeo atualmente utilizam cada vez mais dispositivos portáteis alimentados por bateria, como câmeras, *smartphones*, veículos aéreos não tripulados, atuadores, dispositivos vestíveis e vários sensores, gerando desafios de consumo de energia e processamento de grande porte. Deste cenário decorre a urgente necessidade de codificar de forma eficiente os vídeos digitais antes de armazená-los e/ou transmiti-los. Desta forma, a codificação de vídeo representa um processo fundamental para tornar o suporte às aplicações multimídia uma tarefa possível.

A codificação de vídeo pode ser descrita, de forma simplificada, como um processo que explora blocos e regiões dentro de outros blocos em busca de informações redundantes que podem, de alguma forma, serem compactadas pela exploração de tais redundâncias. A saída desse processo é um fluxo de bits que deve ser decodificado sempre que o vídeo for exibido. Existem várias técnicas de codificação de vídeo disponíveis, cada uma realizando a mesma tarefa de uma maneira diferente. Desta forma, padrões de codificação de vídeo foram criados para permitir uma linguagem comum entre codificadores e decodificadores em diferentes plataformas.

Com o advento de resoluções UHD 2160p e UHD 4320p (7680 × 4320 pixels)

(ITU-R, 2020), e a crescente demanda por vídeos digitais na Internet motivaram o desenvolvimento de um padrão de codificação de vídeo mais eficiente que o padrão H.264/AVC (JCT-VC, 2011). Neste sentido, foi desenvolvido, o *High Efficiency Video Coding* (HEVC) (ITU-T, 2013). O padrão HEVC atinge uma redução de mais de 50% na taxa de bits, mantendo a mesma qualidade visual, quando comparado com o padrão H.264/AVC (SULLIVAN et al., 2012).

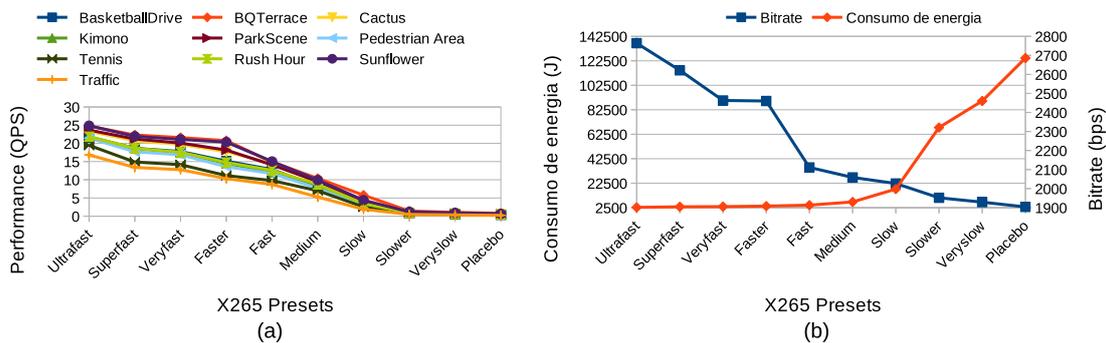
Para atingir esta taxa de redução, o HEVC introduziu uma grande quantidade de inovações no processo de codificação do vídeo, por exemplo, novas estruturas de codificação, transformadas de tamanho variável, unidades de predição de maior tamanho (até  $64 \times 64$  amostras), entre outros (SULLIVAN et al., 2012). No entanto, estas novas ferramentas incorreram em um aumento no esforço computacional do codificador de 9% a 502%, dependendo das configurações utilizadas durante a codificação do vídeo (CORREA et al., 2015). Além disso, duplicou o volume de dados que são acessados na memória, quando comparado com o codificador H.264/AVC (SHAFIQUE; HENKEL, 2014), aumentando a complexidade dos projetos de hardware. Esse aumento na largura de banda de memória deve-se ao armazenamento na memória externa dos quadros de referência, que são utilizados na codificação do vídeo. Esta grande quantidade de acessos à memórias internas e externas do chip, além do aumento do tamanho das memórias internas, levam a um elevado consumo de energia no codificador HEVC (SHAFIQUE; HENKEL, 2014).

## 1.1 Formulação do problema e discussões

O padrão HEVC foi desenvolvido para atender aos requisitos de alta largura de banda para transmissão e armazenamento de conteúdo de vídeo de alta e ultra-alta definição (SULLIVAN et al., 2012). Como resultado, atinge quase o dobro da eficiência de codificação em comparação aos padrões antecessores (PAKDAMAN; HASHEMI; GHANBARI, 2020). Para isso, o HEVC introduziu novas ferramentas de codificação, além de aumentar a precisão das ferramentas existentes. Como resultado, seu processo de codificação é muito mais complexo do que os padrões de codificação de vídeo anteriores, como o H.264/AVC (PAKDAMAN; HASHEMI; GHANBARI, 2020).

Nesse sentido, o trabalho de Monteiro (2017) avaliou e comparou os software de referência HM e JM dos padrões HEVC e H.264/AVC, respectivamente. Para a mesma qualidade de vídeo, a implementação do HEVC atinge uma taxa de compressão de até

Figura 1.2: Resultados de simulação para todos os *presets* do codificador x265, sendo em (a) os resultados de performance e em (b) resultados de *bitrate* vs consumo de energia.



Fonte: Próprio autor.

44,15% (25% em média) maior do que o H.264/AVC. Além disso, estes ganhos alcançados pelo HEVC impactam em um maior consumo de energia, o qual pode variar de 0,11% até 41,6%, sendo 17,4% em média para todos os vídeos testados. Essa ampla variação no custo computacional e no consumo de energia se deve ao alto número de ferramentas de codificação que podem ser usadas dentro dos limites do padrão.

Entre as novas ferramentas apresentadas pelo HEVC, a ME com a nova estrutura de blocos recursiva chamada *Coding Tree Unit* (CTU), tamanhos variáveis de unidades de predição (*Prediction Unit* – PU) e a predição avançada de vetor de movimento (*Advanced Motion Vector Prediction* – AMVP), constitui a maior parte desse aumento na complexidade (PAKDAMAN; HASHEMI; GHANBARI, 2020). Além disso, os esforços mais recentes para o padrão de codificação de vídeo de próxima geração (JVET, 2019a) mostram que ferramentas semelhantes de ME foram estendidas ainda mais, o que leva a uma complexidade da ME ainda maior em um futuro próximo (PAKDAMAN; HASHEMI; GHANBARI, 2020) (CERVEIRA et al., 2020).

Outro exemplo é o x265 (X265, 2019), que é um codificador de código aberto que visa fornecer o codificador HEVC mais rápido e eficiente em termos de computação. Este codificador foi desenvolvido de forma eficiente com muitas técnicas de otimização, e ainda assim não é possível codificar vídeos HD 1080p em tempo real, mesmo em sua configuração mais rápida (SILVEIRA; PORTO; BAMPI, 2017). Esse fato pode ser observado na Figura 1.2(a) onde são apresentados resultados médios de performance para 10 simulações com 10 sequências de vídeo HD 1080p, utilizando os 10 *presets* do codificador x265. Desta forma, soluções arquiteturais de hardware dedicadas são imprescindíveis para codificação de vídeo em tempo real.

Além disso, na Figura 1.2(b) são apresentados os resultados de *bitrate*, em bps, vs resultados de consumo de energia, para o mesmo conjunto de testes. Nota-se nessa figura,

que o *preset* de configuração mais rápido do x265, o *ultrafast*, apresenta um *bitrate* 45% mais alto que o *bitrate* do *preset placebo*, sendo esse o mais lento de todos. Todos esses resultados foram obtidos usando um processador Xeon E3, com quatro núcleos rodando a 3,60 GHz.

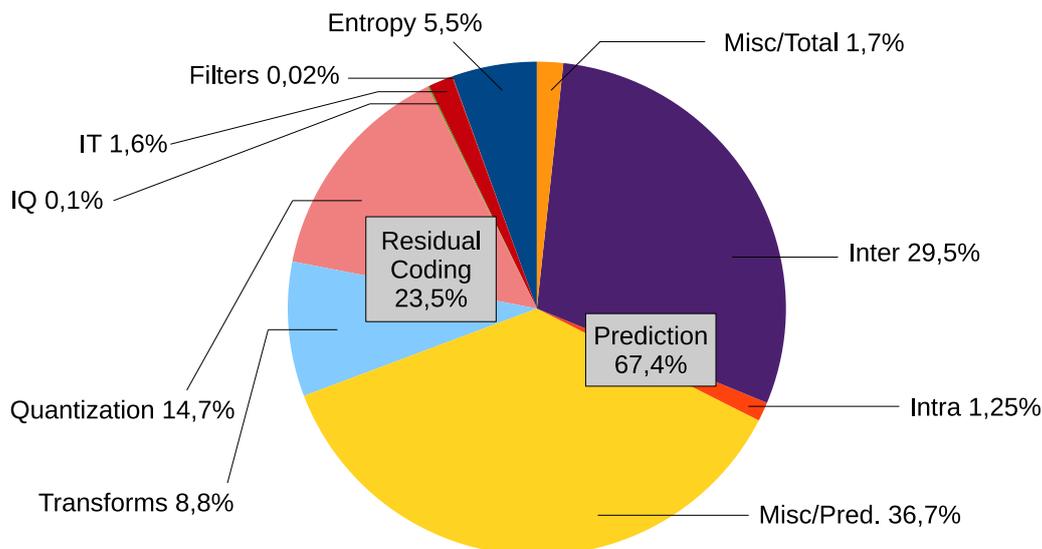
Esses resultados também mostram que o consumo de energia aumenta 45×, do *preset ultrafast* para o *preset placebo*. Isto significa que para cada 1% de redução em *bitrate* há um aumento de 145% em consumo de energia no codificador do x265. Esse aumento ocorre principalmente por alterações nas ferramentas utilizadas pela ME, tais como: algoritmo de busca mais eficiente, maior quantidade de quadros de referência, avaliação de amostras de crominância na ME, aumento de quadros na bi-predição, além do aumento na quantidade de quadros de pré processamento do *rate control* (X265, 2019).

Segundo alguns artigos, durante o processo de codificação do vídeo a etapa mais custosa no codificador HEVC é a ME, a qual, dependendo da configuração utilizada, corresponde em mais de 80% do tempo de processamento e de consumo de energia nos codificadores (SHAFIQUE; HENKEL, 2014), (LUO et al., 2019) e (CEBRIÁN-MÁRQUEZ; MARTÍNEZ; CUENCA, 2019). A ME procura a melhor correspondência de um bloco do quadro atual em um conjunto de blocos presentes nos quadros de referência. A busca é executada em uma área de pesquisa restrita. Os quadros de referência são armazenados tipicamente na memória externa, i.e. uma memória externa ao chip de processamento, geralmente são usadas memórias de tecnologia DRAM. Enquanto que as áreas de busca são armazenadas em memórias internas, i.e. internas ao chip de processamento, onde a tecnologia mais empregada é a SRAM (ZHENG et al., 2019) (SINANGIL et al., 2012).

Devido ao gerenciamento de memória para buscar as amostras da área de busca, a qual geralmente está alocada na memória externa, e ao aumento no consumo de *leakage* das memórias internas, a ME torna-se o bloco de processamento que mais consome energia relacionada à memória no codificador (ZATT et al., 2011) (SHAFIQUE et al., 2012). A Figura 1.3 apresenta resultados de acessos à memória Cache utilizando o software de referência do HEVC, o HM 16.6. Esses são resultados médios de acessos para quatro vídeos de diferentes resoluções, nota-se que as etapas de predição, incluindo a ME, são as etapas que mais realizam acessos aos dados na memória. Além disso, segundo os trabalhos de Zatt et al. (2011) e Shafique et al. (2012), dependendo da configuração, em torno de 70% a 90% do consumo energético da ME é gasto no acesso à memórias internas e externas (*leakage* e dinâmico).

Todos estes cenários demandam soluções que reduzam o número de acessos às

Figura 1.3: Distribuição de acessos à Cache do codificador HEVC.



Fonte: Mativi, Monteiro e Bampi (2016).

memórias e/ou reduzam a largura de banda de memória. Quanto ao acesso à memória externa, existem soluções baseadas em reuso de dados, onde são utilizadas memórias Caches ou memórias *scratchpad* entre o codificador e a memória externa. Desta forma, há uma redução das taxas de leitura da memória externa bastante alta. Entretanto, os dados informados anteriormente já utilizam este tipo de abordagem. Desta forma, o aumento no uso de memórias internas no codificador pode representar aumento no consumo de energia, mesmo reduzindo a comunicação com a memória externa. Isto ocorre, pois as memórias internas consomem energia para manter o dado íntegro em suas células, e quanto maior for a quantidade de bits a serem alimentados maior será o consumo de energia (MUTLU; SUBRAMANIAN, 2015).

Algumas soluções que visam minimizar os problemas ocasionados pela comunicação entre a memória externa e o núcleo de processamento também são apresentadas na literatura. Estes trabalhos se baseiam em algumas técnicas, que são: estratégias de reutilização de dados utilizando *memórias Caches ou memórias scratchpad*; e *compressão de quadros de referência*, antes de serem armazenadas na memória externa. Além disso, soluções específicas para o sistema de memória podem ser utilizadas, tais como: *computação aproximada* nos dados armazenados nas células de memória (MUTLU; SUBRAMANIAN, 2015); e uso de tecnologias de *memórias não voláteis* – NV-RAMs, tais

como: STT-MRAM <sup>1</sup>, PCM <sup>2</sup> e ReRAM <sup>3</sup>. Essas tecnologias de memória apresentam vantagens e desvantagens em relação as tradicionais memórias, SRAM e DRAM. No Capítulo 4 essas estratégias serão apresentadas e discutidas juntamente com trabalhos da literatura que aplicam uma ou mais dessas técnicas em conjunto.

## 1.2 Hipótese de tese

Diante do exposto na seção anterior, se percebe o grande desafio relacionado à utilização de memórias de forma eficiente nos codificadores de vídeo, principalmente com foco na codificação de vídeos de alta definição para dispositivos móveis. Seja pela quantidade de aplicações que demandam esta utilização ou pelas diversas soluções que podem ser aplicadas, sendo que estas soluções podem ser utilizadas de forma individual ou em conjunto. Essas condições, aliadas à falta de trabalhos na literatura que explorem este conjunto de técnicas, indicam um grande potencial de inovações para a área.

A hipótese de tese defendida nesse trabalho é que o consumo de energia relacionado à hierarquia de memória de codificadores de vídeo pode ser drasticamente reduzido com a utilização de computação aproximada e tecnologias de memórias NV-RAMs, gerando baixo impacto na eficiência de codificação. Para confirmar essa hipótese, essa tese explora a gestão e aproximação de dados em sistemas de memórias com NV-RAMs dedicados a codificadores de vídeo eficientes.

Nesse sentido, essa tese avalia o impacto em consumo de energia e eficiência de codificação da utilização conjunta de estratégias de reuso de dados, compressão de quadros de referência, computação aproximada e NV-RAMs. Para avaliar os impactos da computação aproximada na eficiência de codificação serão realizados experimentos com o software de referência do HEVC, com intuito de medir os impactos na eficiência de codificação causados pela redução da área de busca e a utilização do *bit dropping*. Assim, as configurações mais eficientes serão selecionadas e utilizadas em conjunto com as demais técnicas.

Além disso, serão avaliadas hierarquias de memórias compostas por memórias DRAM, SRAM e NV-RAMs. Após esses experimentos, as configurações mais eficientes serão avaliadas em conjunto com as outras técnicas. Para estimar os resultados dessas

---

<sup>1</sup> *Spin-Transfer Torque Magnetic RAM* (STT-MRAM), as vezes apresentada como *Spin-Transfer Torque RAM* (STT-RAM).

<sup>2</sup> *Phase Change RAM* (PCM, PCRAM ou PRAM).

<sup>3</sup> *Resistive RAM* ou *memristor* (ReRAM ou RRAM).

estratégias foi desenvolvido em software e hardware um sistema de redução de energia relacionada à memória para arquiteturas de codificadores de vídeo digitais. Para isso, esse sistema contará com uma hierarquia de memória composta por tecnologia NV-RAMs, empregando a técnica de reuso de dados Level C, a qual permite a reutilização de amostras de blocos vizinhos. Além disso, para aumentar a eficiência energética dessa hierarquia de memória, será utilizado o *power gating* em bancos de memória. Ainda, o sistema utilizará níveis de configuração, permitindo a utilização de computação aproximada através da redução do tamanho da área de busca e *bit dropping* nas amostras utilizadas, sendo que o *bit dropping* é utilizado juntamente com um compressor de quadros de referência.

Assim, o sistema de redução de energia desenvolvido a partir dos estudos e avaliações das hipóteses levantadas, empregará seis diferentes técnicas em conjunto, sendo elas: memórias de tecnologias NV-RAMs, reuso de dados, *power gating*, computação aproximada com *bit dropping* e redução do tamanho da área de busca e compressão de quadros de referência. Desta forma, atingindo altas taxas de redução no consumo de energia relacionadas à hierarquia de memória com o mínimo impacto em eficiência de codificação.

### 1.3 Organização da tese

O Capítulo 2 deste texto apresenta as principais ferramentas de codificação de vídeo, bem como o fluxo de codificação de vídeo nos codificadores de vídeo atuais. O algoritmo de busca estado da arte e utilizado nessa tese, o *Test Zone Search* (TZS), é explicado em detalhes. Além disso, as questões relacionadas aos acessos à memória externa e os desafios que envolvem a comunicação entre memória externa e núcleo de processamento são abordadas.

No Capítulo 3 são apresentadas e discutidas as principais características do sistema de memória. Ainda, é discutido o funcionamento de uma hierarquia de memória e quais os tipos de memória que geralmente são utilizadas para compor esse sistema. Além disso, são apresentadas as tecnologias de memória que serão exploradas nesse trabalho, as quais envolvem as tecnologias voláteis tradicionais como: SRAM e DRAM, e as não voláteis, NV-RAMs, tais como: STT-MRAM, PCRAM, ReRAM e FeRAM.

O Capítulo 4 traz os principais trabalhos encontrados na literatura que abordam as técnicas de redução da largura de banda de memória em codificadores de vídeo, como: reuso de dados, compressão de quadros de referência, uso de tecnologias NV-RAMs e computação aproximada na codificação de vídeo.

No Capítulo 5 são apresentadas as análises sobre a distribuição dos acessos às amostras da área de busca realizadas pelo algoritmo de busca TZS. Estas análises possibilitam melhor entendimento sobre o comportamento de acessos realizados pelo TZS dentro de uma área de busca e o conhecimento das regiões mais e menos acessadas, sendo de grande importância para utilização da computação aproximada.

No Capítulo 6 o sistema completo é apresentado e todas suas etapas são discutidas, incluindo as técnicas de computação aproximada, as tecnologias de memória, o reuso de dados e o armazenamento das informações na memória interna, o compressor de quadros de referência e a comunicação com a memória externa. Além disso, a comunicação entre os módulos e a organização da memória para acesso aleatório são também apresentados e discutidos.

O Capítulo 7 apresenta os resultados de redução de largura de banda de memória, de redução de consumo de energia e o impacto em eficiência de codificação, medido em *Bjontegaard Delta Bitrate* (BD-rate). Ainda, esses resultados são comparados aos trabalhos estado da arte.

O Capítulo 8 finaliza este texto com as conclusões acerca do trabalho desenvolvido e dos resultados alcançados, além apresentar algumas perspectivas de trabalhos futuros.

## 2 FUNDAMENTOS DA CODIFICAÇÃO DE VÍDEO

Aplicações que manipulam vídeos digitais estão presentes hoje em diversos dispositivos, que vão desde aparelhos celulares a televisores digitais de alta definição, com os mais diversos propósitos tais como: entretenimento, educação, comunicação, segurança, entre outros. Esses vídeos digitais são compostos por imagens estáticas dispostas sequencialmente a uma determinada frequência de exibição. As imagens estáticas que compõem um vídeo digital são chamadas de quadros, os quais são compostos por pixels.

Com a tecnologia atual, a manipulação adequada dos vídeos digitais se torna um desafio. Este empenho em atingir altas taxas de compressão fez com que a pesquisa no meio acadêmico e na indústria aumentasse consideravelmente nos últimos anos. A expectativa é que a pesquisa nesse tema cresça cada vez mais, dado o aumento considerável nas resoluções dos televisores digitais, que hoje já chega a UHD 8K com  $7680 \times 4320$  pixels, sendo o pixel a menor informação que forma uma imagem digital.

Além disso, estes dispositivos recomendam taxas de processamento de 60 a 120 quadros por segundo, para que se obtenha uma boa qualidade visual (SZE; BUDAGAVI; SULLIVAN, 2014). Como os vídeos digitais são utilizados para os mais diversos propósitos, eles necessitam ser armazenados e eventualmente transmitidos, sendo que isso deve ser feito de forma eficiente. A Equação 2.1 mostra a largura de banda ( $LB$ ) necessária para transmitir uma sequência de vídeo não comprimido, considerando subamostragem de pixel 4:2:0. Os resultados de largura de banda são medidos em bits por segundo ( $bits/s$ ).

$$LB = L \times A \times QPS \times B \times tx_{sub}[bits/s] \quad (2.1)$$

Na Equação 2.1,  $L \times A$  é o número de amostras em um quadro, onde  $L$  representa a direção horizontal e  $A$  representa a direção vertical;  $QPS$  representa a quantidade de quadros por segundo da sequência de vídeo;  $B$  representa a quantidade de bits por amostra de luminância ou crominância (tipicamente 8 bits);  $tx_{sub}$  representa o fator de multiplicação para a subamostragem de pixel, sendo  $tx_{sub} = 1, 5$  para subamostragem 4:2:0, onde para a cada 4 amostras de luminância existem 2 amostras de crominância, uma para cada camada (RICHARDSON, 2011), caso não fosse utilizada a subamostragem (formato 4:4:4) esse fator seria 3.

Para ilustrar essa largura de banda, pode-se utilizar um vídeo digital com resolução HD 1080p, comercialmente conhecido como *Full HD*, capturado a 30 quadros por segundo e subamostragem de pixel 4:2:0. Esse vídeo necessitaria de uma taxa de 712 Mb/s

para ser transmitido em tempo real, e uma hora de captura deste vídeo ocuparia aproximadamente 312 GB de espaço de armazenamento sem compressão. Agora, se fosse um vídeo HD 4K capturado a 60 quadros por segundo e subamostragem de pixel 4:4:4 esses valores aumentariam drasticamente e iriam requerer uma taxa de 11 Gb/s para transmissão em tempo real e ocuparia 5 TB de espaço em disco em apenas uma hora.

Desta forma, pode-se perceber que a manipulação de vídeos digitais sem compressão é uma tarefa inviável, especialmente para aplicações em tempo real e para dispositivos móveis com limitação de processamento e consumo de energia, devido ao poder computacional necessário para as ferramentas de exibição/transmissão e também devido ao espaço para o armazenamento. Para que esta manipulação ocorra de forma adequada a codificação dos vídeos digitais seguem normas de padronização. Dentre os padrões atuais destaca-se o HEVC, um dos padrões mais utilizado no mercado, o qual foi lançado em 2013 (ITU-T, 2013). Nas próximas seções serão apresentadas as principais características dos vídeos digitais e um breve histórico dos padrões de codificação, além de uma ideia geral sobre o fluxo de codificação do padrão HEVC, em especial o processo de estimação de movimento, que será importante para a compreensão do escopo deste trabalho.

## **2.1 Características dos vídeos digitais**

Nessa seção serão apresentadas as principais características dos vídeos digitais e como elas impactam a taxa de compressão e a qualidade. Como citado anteriormente, o pixel é a menor informação que forma uma imagem digital, sendo que cada pixel possui três canais de informação, e o tipo de informação apresentada em cada canal irá depender do espaço de cores utilizado (RICHARDSON, 2004). O espaço de cores mais utilizado para vídeos digitais é o YCbCr, o qual é composto por três elementos, sendo eles, luminância (Y), croma azul (Cb) e croma vermelho (Cr) (RICHARDSON, 2011). Estes três elementos, quando combinados, podem formar milhares de cores distintas (RICHARDSON, 2004). Este formato facilita a subamostragem de cores, pois as informações de luminância (Y) e croma (Cb e Cr) são completamente independentes. E o processamento individual dessas informações, é importante devido ao fato de que o sistema visual humano é mais sensível a informações de luminância do que croma (RICHARDSON, 2004).

A principal vantagem do espaço de cor YCbCr em relação ao RGB é que os componentes Cb e Cr podem ser representados com uma menor resolução quando comparados

ao componente Y, já que o sistema visual humano é menos sensível a cor do que a luminosidade, como dito anteriormente. Essa operação, que é utilizada nos codificadores de vídeo, é chamada de subamostragem de cor. Com o uso dessa operação, há uma redução significativa na quantidade de dados necessários para representar os componentes de crominância, sem haver um efeito perceptível na qualidade visual (RICHARDSON, 2004). Os três principais padrões de subamostragem para a crominância são: i) 4:4:4 significa que os três componentes (Y:Cb:Cr) tem a mesma resolução, portanto, há uma amostra de cada componente para cada pixel; ii) a 4:2:2, onde os componentes de crominância tem a mesma resolução na vertical, mas metade da resolução na horizontal; e iii) 4:2:0, a qual utiliza metade da resolução dos componentes de crominância tanto na vertical como na horizontal. Desta forma, na subamostragem 4:2:2, é possível reduzir em 50% o número de amostras de crominância. E na subamostragem 4:2:0 há uma redução de 75% no número de amostras de crominância, o que representa uma redução de 50% no tamanho do vídeo (RICHARDSON, 2004). Assim, nota-se que a subamostragem representa uma técnica de computação aproximada eficiente na codificação de vídeo.

Outros fatores que influenciam a qualidade do vídeo são a taxa de amostragem e a resolução do vídeo. A resolução de um quadro é definida pela quantidade de pixels distribuídos na horizontal e na vertical. Existem muitos tamanhos de resolução, tais como CIF (*Common Intermediate Format*), com  $352 \times 288$  pixels e o VGA (*Video Graphics Array*) com  $640 \times 480$  pixels, sendo estes, formatos antigos e com poucas aplicações atualmente. As resoluções de maior interesse atualmente são o HD 720p ( $1280 \times 720$  pixels), o HD 1080p ( $1920 \times 1080$  pixels) e o UHD 4K com  $3840 \times 2160$  pixels. A taxa de amostragem é o número de quadros exibidos durante um período de tempo: quanto maior a taxa de amostragem mais suave será a transição dos quadros, fazendo com que a percepção de movimento seja a mais real possível. A taxa de amostragem geralmente é de 24 a 30 quadros por segundo (RICHARDSON, 2004). Entretanto, esta taxa pode ser maior se o vídeo tiver uma resolução muito alta. Se a resolução e a taxa de amostragem aumentarem, maior será a demanda de processamento computacional do codificador de vídeo.

## 2.2 Padrões de codificação de vídeo

A padronização de técnicas de codificação de vídeo digital é essencial para permitir que equipamentos de diferentes fabricantes se comuniquem. Deste modo, padrões de

codificação de vídeo foram desenvolvidos. Os primeiros esforços para a padronização da codificação de vídeo digital datam do final da década de 80 e início da década de 90, com a criação do padrão H.261 pelo ITU-T (ISO/IEC, 1993). Este padrão foi o responsável por introduzir ferramentas como, estimação de movimento na direção temporal, transformada discreta do cosseno e quantização linear seguida de codificação de entropia aplicada aos resíduos de predição (RICHARDSON, 2004). Estas técnicas continuam sendo utilizadas pelos padrões atuais.

Após o padrão H.261, surgiu o padrão MPEG-1 da ISO/IEC (ISO/IEC, 1993), seguido do padrão MPEG-2 da ISO/IEC, que também foi padronizado pela ITU-T como H.262 (ITU-T, 2000). O MPEG-2, ou H.262, ainda é amplamente utilizado pelos sistemas de televisão digital. Apesar do sucesso do padrão MPEG-2, o desenvolvimento de novos padrões continuou. O padrão H.263 (ITU-T, 2005) foi lançado e incorporou alguns avanços obtidos pelos padrões MPEG-1 e MPEG-2. O bom resultado alcançado com o padrão MPEG-2/H.262 motivou os grupos MPEG (*Moving Picture Experts Group*) e VCEG (*Video Coding Experts Group*) a trabalharem em conjunto novamente. Em 1997, estes grupos uniram-se sob o nome JVT (*Joint Video Team*), para desenvolver um novo padrão capaz de atingir taxas de compressão superiores às do H.263.

Este novo projeto foi intitulado de H.26L. Em 2001, o grupo MPEG concluiu o desenvolvimento do padrão MPEG-4 Parte 2 (ISO/IEC, 2004). A partir deste padrão e do H.26L, o grupo JVT, iniciou um novo projeto. Este novo padrão tinha como objetivo atingir 50% a mais de compressão para a mesma qualidade, quando comparado a padrões anteriores. A versão final deste padrão foi concluída em 2003, sendo intitulado de H.264 pelo grupo VCEG, e MPEG-4 Parte 10 - AVC (*Advanced Video Coding*) pelo grupo MPEG. Em um artigo de 2004, descrevendo o padrão, o coordenador do JVT, Gary Sullivan, decidiu utilizar o nome H.264/AVC para se referir ao padrão de forma equilibrada entre os nomes dados ao padrão pela ITU-T e pela ISO (SULLIVAN; TOPIWALA; LUTHRA, 2004) (ITU-T, 2012). Desde então, este padrão tem sido foco de incessante pesquisa e desenvolvimento por grupos espalhados ao redor do mundo.

Nos últimos anos os vídeos digitais tiveram um aumento considerável nas suas resoluções e taxa de amostragens. Com isso, um novo padrão começou a ser desenvolvido a partir de 2010 pelo JCT-VC (*Joint Collaborative Team on Video Coding*), sendo este grupo formado por especialistas do VCEG e do MPEG. O *High Efficiency Video Coding* (HEVC) (ITU-T, 2013) supriu as necessidades de codificação citadas anteriormente com o máximo de eficiência e, para isso, reuniu ferramentas de compressão especializa-

das para vídeos de altas resoluções atingindo altas taxas de processamento e o dobro da taxa de compressão, para a mesma qualidade do vídeo, quando comparado com o padrão H.264/AVC (OHM et al., 2012).

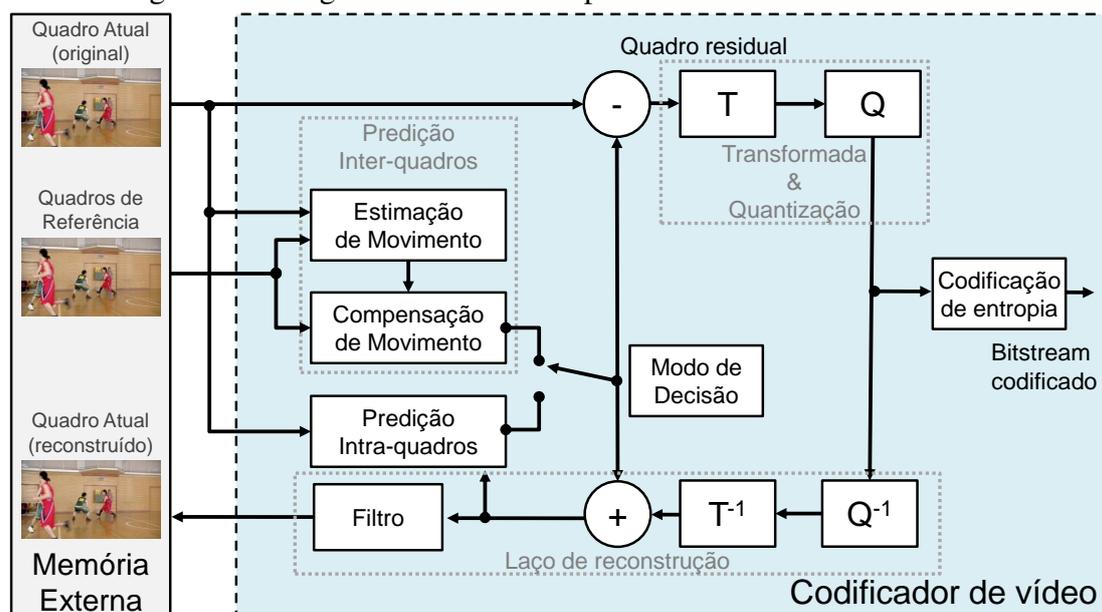
Atualmente, está em desenvolvimento pela comunidade de codificação de vídeo dois padrões para uma futura tecnologia de compressão de vídeo além do HEVC: AV1 (AV1, 2019), desenvolvido pela *Alliance for Open Media* (AOM) e o *Versatile Video Coding* (VVC) (JVET, 2019a), desenvolvido pela *Joint Video Experts Team* (JVET) no grupo *Future Video Coding* do ITU-T VCEG e ISO/IEC MPEG.

A AOM anunciou em março de 2018 o lançamento da especificação para a versão 1 do AV1 (AOM, 2019) (AV1, 2019), um codec de vídeo de código aberto e livre de *royalties*. O AV1 foi construído sobre os codecs de software livre, VP9 e VP10 do Google, Thor da Cisco e Daala do Mozilla, ele inclui novas ferramentas de codificação e demonstra ganhos significativos de compressão em comparação com seu predecessor VP9 ou a implementação do padrão HEVC de código aberto, o x265 (AV1, 2019). Este codec emergente ambiciona competir com o padrão HEVC propondo um esquema eficiente de compressão de vídeo sem quaisquer limitações relativas a alto custo de complexidade ou termos de uso legal.

A JVET anunciou em abril de 2018 o início oficial do trabalho em um novo padrão de codificação de vídeo, conhecido como *Versatile Video Coding*, o qual foi lançado oficialmente em Julho de 2020 como norma ITU-T H.266 e ISO/IEC 23090-3 (JVET, 2019a). Desde 2016, a JVET já vinha estudando e desenvolvendo a próxima geração de codec de vídeo sob o software *Joint Exploration Model* (JEM) (JVET, 2019b). O VVC aproveita os avanços do JEM e já demonstra uma forte redução na taxa de bits para a mesma qualidade de vídeo quando comparado ao HEVC/HM. A JVET já publicou o novo software de referência para o padrão VVC, o *Versatile Test Model* (VTM), o qual possui a implementação dessa primeira versão do padrão (JVET, 2019c).

Nesta tese foi utilizado o software de referência do padrão HEVC, o HM. Desta forma, na próxima seção será apresentado em detalhes o fluxo de codificação desse padrão. Entretanto, esse fluxo de codificação apresentado para o HEVC também pode ser usado para os demais padrões como H.264/AVC, AV1 e VVC. No entanto, mesmo que esses outros padrões apresentem peculiaridades distintas do HEVC, como tamanhos de blocos, particionamento, tipos de transformadas, entre outras ferramentas, o fluxo de codificação residual, utilizando quadros de referência, ainda é utilizado em todos eles.

Figura 2.1: Diagrama de blocos simplificado do Codificador HEVC.



Fontes: Zatt et al. (2013) e Sze, Budagavi e Sullivan (2014).

## 2.3 Padrão HEVC

Nesse sentido, esta seção apresenta uma breve introdução ao padrão de codificação de vídeo HEVC, o qual é um dos objetos de estudo nessa tese, descrevendo seu fluxo de codificação e inovações, destacando os acessos à memória externa durante o processo da ME, discutindo o impacto disso no consumo energético do codificador.

### 2.3.1 Fluxo de codificação

O padrão HEVC adota o esquema híbrido de codificação em blocos já consagrado em padrões anteriores, baseado na previsão com compensação de movimento, transformadas e codificação de entropia de alta eficiência (SULLIVAN et al., 2012). A Figura 2.1 apresenta o diagrama de blocos simplificado do codificador HEVC.

Como apresentado na Figura 2.1, a previsão inter-quadros é composta pelos módulos de estimativa e compensação de movimento, sendo responsável por explorar as redundâncias temporais entre os quadros vizinhos, enquanto que a previsão intra-quadro explora as redundâncias espaciais no quadro que está sendo codificado. A saída das etapas de previsões é um grande conjunto de previsões candidatas. Entre todos os diferentes tamanhos de blocos para as previsões intra e inter-quadros, o melhor modo de previsão

deve ser selecionado pelo Modo de Decisão.

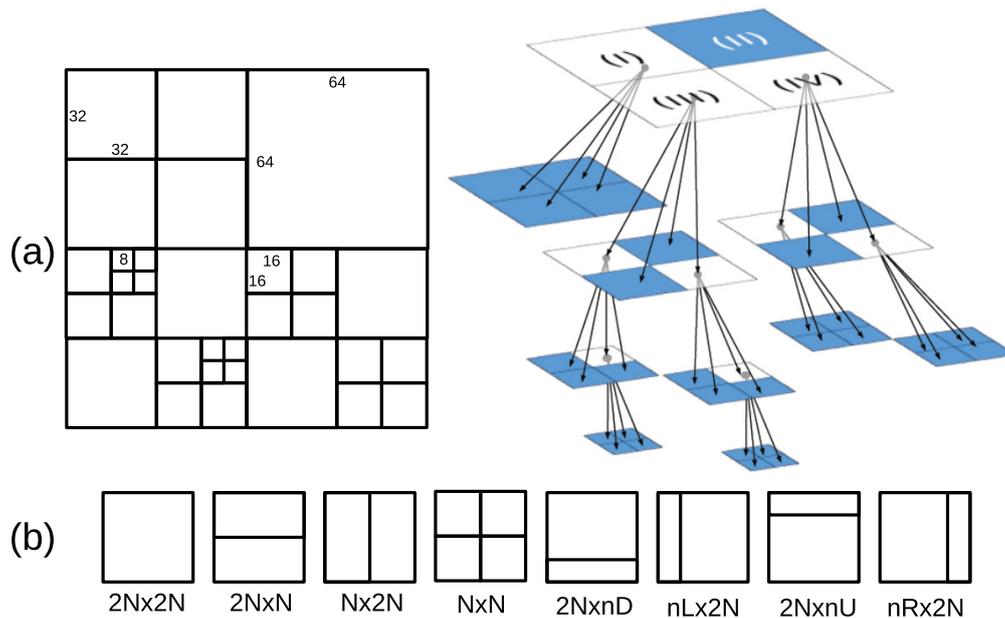
Passadas as etapas das predições e a seleção pelo Modo de Decisão, o bloco predito e o bloco atual são subtraídos e os resíduos gerados são enviados para as transformadas. Nessa etapa, as informações são transformadas do domínio espacial para o domínio das frequências espaciais e, em seguida, esses coeficientes resultantes das transformadas são quantizados, descartando informações menos relevantes para o sistema visual humano. No final da codificação, a quantização de coeficientes de transformação, vetores de movimento, direções de predição, modos de bloco e outros tipos de informação são codificados sem perdas pela Codificação de Entropia.

Após a Codificação de Entropia, o *bitstream* é montado e a codificação completa. No entanto, cada bloco deve ser reconstruído para funcionar como referência para outros blocos. Esse processo é apresentado na Figura 2.1 na etapa Laço de Reconstrução, para isso, a quantização inversa e as transformadas inversas são aplicadas aos coeficientes quantizados (os mesmos dados enviados anteriormente para a codificação de entropia), uma vez que os resíduos são quantizados inversamente, eles são adicionados ao bloco previsto para reconstruir o bloco decodificado. Esse laço de reconstrução garante a consistência das informações do codificador e do decodificador, evitando diferenças de informação entre o codificador e o decodificador.

Esse esquema híbrido de codificação é utilizado por todos codificadores de vídeos atuais. Entretanto, as ferramentas usadas em cada uma dessas etapas podem ser diferentes, sendo esse diferencial que faz um codificador ser mais eficiente que outro. Nesse sentido, o processo de codificação do HEVC é mais complexo do que o utilizado no H.264/AVC. Isto se deve ao fato de que o HEVC utiliza uma estrutura flexível de árvore quadrática (*quadtree*), o que permite o uso eficiente de blocos com grandes dimensões e formatos variados nas etapas de predição e transformadas (SULLIVAN et al., 2012). O fluxo de codificação no HEVC ocorre da seguinte forma: o quadro do vídeo é primeiramente dividido em um conjunto de blocos de mesmo tamanho, chamados *Coding Tree Units* (CTUs). Uma CTU consiste em um bloco de  $M \times M$  amostras de luminância e dos dois blocos de crominância correspondentes, sendo  $M = 16, 32$  ou  $64$  amostras. Os tamanhos maiores tipicamente atingem melhores taxas de compressão (SULLIVAN et al., 2012). As CTUs eram também chamadas de *treeblocks* em versões anteriores do padrão HEVC. Na versão atual, o tamanho máximo do bloco de luminância de uma CTU é de  $64 \times 64$  amostras.

A CTU é composta por uma ou mais unidades básicas de codificação, chamada

Figura 2.2: (a) Quatro CTUs com suas respectivas *quadtree* e (b) divisões das PUs.

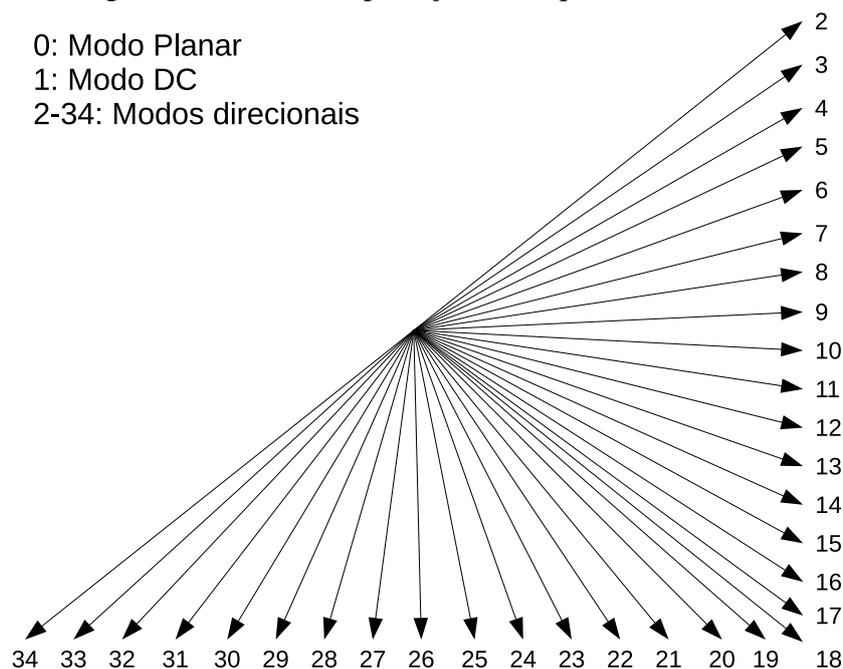


Fonte: Correa et al. (2015).

*Coding Units* (CUs). O conceito de CU permite a sua divisão recursiva em quatro blocos de tamanhos iguais a partir da CTU. Este processo de divisões recursivas acaba por formar uma estrutura de codificação em forma de árvore quadrática composta por blocos de CU, que podem ter dimensões que variam desde  $8 \times 8$  amostras até o tamanho da própria CTU. A Figura 2.2 (a) apresenta quatro CTU com as suas respectivas *quadtree*. Nesta figura cada CTU (I, II, III, IV) tem um tamanho de  $64 \times 64$ . As folhas da *quadtree* representam o tamanho escolhido para a CU. No exemplo em questão, o menor nível ( $8 \times 8$ ) ocorre na profundidade 4.

O HEVC provê a utilização de *Prediction Units* (PUs), que são as unidades básicas que transportam as informações dos processos de predição (SULLIVAN et al., 2012). Cada CU pode conter uma ou mais PUs dependendo do modo de partição. Existem oito configurações de partições de PUs diferentes, quatro com formatos simétricos  $2N \times 2N$ ,  $2N \times N$ ,  $N \times 2N$ ,  $N \times N$  e quatro de formato assimétrico  $2N \times nU$ ,  $2N \times nD$ ,  $nL \times 2N$  e  $nR \times 2N$ . Isso significa que uma CU com partição  $2N \times 2N$  contém uma PU. Se a CU tiver partição  $N \times N$ , ela terá 4 PUs. Nas demais configurações de partição a CU terá 2 PUs (SULLIVAN et al., 2012). Todos estes modos podem ser utilizados na predição interquadros. Para a predição intra-quadro só podem ser utilizadas PUs de tamanho  $2N \times 2N$  e  $N \times N$ . A Figura 2.2 (b) apresenta todas as possíveis representações das PUs, simétricas e assimétricas. Mais detalhes do fluxo e das etapas de codificação do HEVC são informadas nas próximas seções.

Figura 2.3: Modos de predição intra-quadro no HEVC.



Fonte: Lainema et al. (2012).

### 2.3.2 Etapa das predições

O processo de predição é o primeiro processo a ser executado por um codificador HEVC, sendo que a predição pode ser de três tipos: intra-quadro (I) e inter-quadros (P) ou (B). Quadros do tipo I são codificados utilizando apenas a predição intra, em quadros do tipo P e B podem ser aplicados tanto o processo de predição intra-quadro quanto no processo de predição inter-quadros. Nos quadros de tipo B cada CTU pode utilizar mais de um quadro como referência para a predição inter-quadros, enquanto nos quadros do tipo P é utilizado apenas um quadro como referência (RICHARDSON, 2011).

O módulo da predição intra-quadro do HEVC é o responsável por reduzir a redundância espacial, assim como é feito no H.264/AVC. No HEVC este módulo sofreu um grande aumento na sua complexidade, uma vez que apresenta 33 modos direcionais de predição intra-quadro, apresentados na Figura 2.3, mais os modos *DC* e *planar*, para os componentes de luminância de cada PU (SULLIVAN et al., 2012). No H.264/AVC são utilizados apenas nove modos direcionais para cada macrobloco (SULLIVAN et al., 2012).

A predição intra-quadros do HEVC usa como blocos de referência os blocos adjacentes previamente codificados que estão acima e à esquerda do bloco a ser codificado. A predição angular utiliza 33 direções e usa amostras de referência para interpolar valo-

res de pixels em um bloco predito. Embora seja essa predição seja mais adequada para blocos com várias arestas, ela pode resultar em algum contorno visível, também chamado de *blockiness*, em áreas da imagem (LAINEMA et al., 2012). A predição planar supera esses problemas criando uma área de predição sem descontinuidades nos limites do bloco, enquanto que a predição DC calcula a média dos valores das amostras para preencher o bloco predito com valores constantes.

Para escolher o melhor modo de predição, o HEVC avalia recursivamente todos os níveis dentro de uma *quadtree*, onde o tamanho do bloco varia de  $64 \times 64$  a  $4 \times 4$ . Todos os 35 modos de predição são avaliados para cada tamanho de bloco utilizando o algoritmo *Rough Mode Decision* (RMD) (LAINEMA et al., 2012), o qual usa a Soma das Diferenças Transformadas Absolutas (*Sum of Absolute Transformed Differences – SATD*) para prever sem a necessidade do custo do *Rate-Distortion* (RD) completo. Os modos com o menor custo são inseridos em uma lista, para serem avaliados posteriormente, juntamente com os modos mais prováveis escolhidos de acordo com os blocos vizinhos. Desta forma, se percebe que a predição intra-quadro do HEVC tem alto custo computacional, pois todos os modos são testados recursivamente para encontrar o menor custo.

A predição inter-quadros contém, assim como no H.264/AVC, o módulo mais complexo do codificador de vídeo, a ME. A ME também é um dos módulos responsáveis por gerar os maiores ganhos de compressão no codificador, reduzindo as redundâncias temporais entre quadros vizinhos de uma cena. No HEVC a predição inter-quadros é utilizada em todas as avaliações de CU durante o processamento das *quadtree*. Para cada CU na *quadtree*, a predição inter-quadros precisa executar a ME várias vezes, sendo uma vez para o tamanho da partição  $2N \times 2N$ , duas vezes para a partição  $2N \times N$ , duas vezes para a  $N \times 2N$ , quatro vezes para avaliar a partição  $N \times N$  e mais quatro vezes para os tamanhos assimétricos, em um total de treze operações de ME por CU (SULLIVAN et al., 2012). Todas estas avaliações para serem realizadas exigem um elevado número de cálculos, o que torna a ME o processo mais demorado na codificação de vídeo. A ME será discutida de forma mais detalhada na Seção 2.4.

Ao final das etapas de predições intra-quadro e inter-quadros, um grande conjunto de predições candidatas é gerado. Esse conjunto é formado pelos modos de predição mais prováveis encontrados pelos diferentes tamanhos de blocos avaliados tanto para as predições intra-quadro quanto para as predições inter-quadros. O melhor modo de predição deve ser selecionado pelo modo de decisão, avaliando o custo do *Rate-Distortion*. Após escolhido o modo de predição, o bloco predito e o bloco atual são subtraídos e os resíduos

gerados são enviados para as transformadas.

### 2.3.3 Transformadas, quantização e entropia

O HEVC utiliza uma unidade básica para o processo de transformada e quantização, chamada de *Transform Unit* (TU) (SULLIVAN et al., 2012). Cada CU pode conter uma ou mais TUs. O módulo de transformadas é responsável por converter as informações do domínio espacial para o domínio das frequências, para que assim, a quantização possa ser aplicada, reduzindo a redundância espacial presente nos resíduos de predição (RICHARDSON, 2011). A transformada principal (*core transform*) utilizada no HEVC é uma aproximação inteira da DCT (*Discrete Cosine Transform*), já consagrada em padrões anteriores de codificação de vídeo. A DCT opera apenas sobre blocos quadrados de resíduos, sendo que os tamanhos de bloco permitidos são de  $4 \times 4$ ,  $8 \times 8$ ,  $16 \times 16$  e  $32 \times 32$  amostras (SULLIVAN et al., 2012).

O processo de quantização é aplicado logo após as transformadas, aplicando uma operação de corte aos coeficientes obtidos pelo processo de transformadas. A etapa de quantização recebe os resíduos transformados pela DCT e aplica operações irreversíveis, eliminando ou atenuando dados de alta frequência, menos significativos ao olho humano. A quantização está diretamente ligada a um parâmetro de quantização (QP) e quanto maior o QP, maiores serão os cortes nos resíduos, o que tende a maiores perdas na qualidade do vídeo codificado. Por outro lado, valores altos de QP contribuem diretamente para a obtenção de altas taxas de compressão.

A última etapa da codificação do vídeo é a codificação de entropia. Nessa etapa são recebidas as informações da quantização de coeficientes de transformação, vetores de movimento, direções de predição, modos de bloco e outros tipos de informação. Com todas essas informações são aplicados algoritmos de codificação sem perdas para manter a qualidade de vídeo e obter uma alta taxa de compressão. As técnicas de codificação utilizadas nesta etapa são fortemente baseadas em análises estatísticas e conseguem representar valores com uma maior densidade de probabilidade com um menor número de bits. A saída desta etapa é o *bitstream* do vídeo codificado. No HEVC, o algoritmo utilizado para a codificação de entropia utiliza o método de Codificação Aritmética Binária Adaptativa ao Contexto (CABAC) (SULLIVAN et al., 2012).

### 2.3.4 Reconstrução dos quadros de referência

Durante o processo de codificação do vídeo, um laço de reconstrução do quadro codificado se faz necessário. Nesse processo, são gerados os quadros de referência que serão utilizados na codificação dos próximos quadros. A reconstrução do quadro é realizada após a etapa de quantização, onde são aplicadas quantizações inversas e transformadas inversas sobre os resíduos, que depois são adicionados aos quadros reconstruídos de acordo com o processo de predição utilizado. Aqui também são aplicados filtros para remover artefatos inseridos pela codificação por blocos. Após a aplicação destes filtros, o quadro reconstruído é armazenado e poderá ser utilizado como referência para a codificação dos próximos quadros do vídeo. No HEVC, os filtros inseridos nesta etapa são o *Sample Adaptive Offset* (SAO), que é aplicado em áreas cujas amostras possuem intensidade homogêneas (SULLIVAN et al., 2012) e o *Deblocking Filter* (DF) que diminui o efeito de bloco gerado pela codificação por CUs (SULLIVAN et al., 2012). Este processo de reconstrução do quadro é fundamental para que os quadros de referência sejam os mesmos tanto no codificador quanto no decodificador, evitando inconsistências no processo de decodificação (*drifting*).

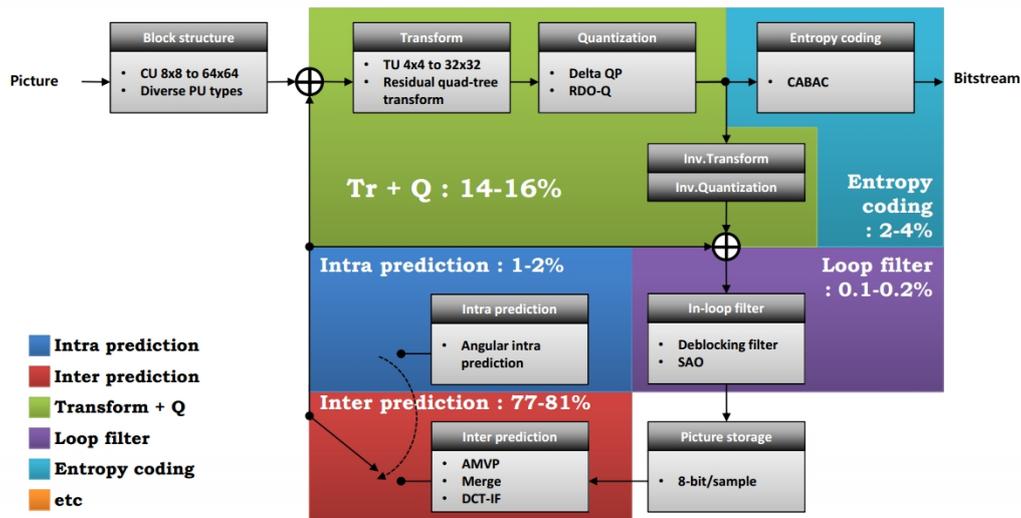
### 2.4 Estimação de movimento e o acesso à memória externa

O padrão HEVC atinge altas taxas de compressão. Entretanto, tanto a complexidade quanto o consumo de energia desse codificador aumentaram bastante em relação ao padrão H.264/AVC. Segundo Correa et al. (2012), o custo computacional do HEVC, introduzida através de novas ferramentas, varia de 9% a 502% dependendo da configuração utilizada pelo software de testes do HEVC, o HM. Em Vanne et al. (2012), a complexidade computacional do HEVC é 40% maior que a do H.264/AVC HP quando apenas as ferramentas essenciais de codificação são utilizadas.

Muitas das ferramentas responsáveis pelos ganhos em compressão e também pelo aumento em complexidade estão na ME. A ME está contida na predição inter-quadros, sendo a etapa mais complexa do processo de codificação de vídeo, representando entre 60% e 80% do tempo total de computação, tanto no H.264/AVC (KUHN, 1999) quanto no HEVC (BOSSSEN et al., 2012).

A Figura 2.4 apresenta o diagrama de blocos do codificador HEVC com os respectivos custos associados a cada módulo do codificador, onde pode ser visto que o módulo

Figura 2.4: Custo computacional do codificador HEVC por bloco.



Fonte: IPSL (2012).

mais complexo do codificador é a previsão inter-quadros, representando uma taxa entre 77% e 81%.

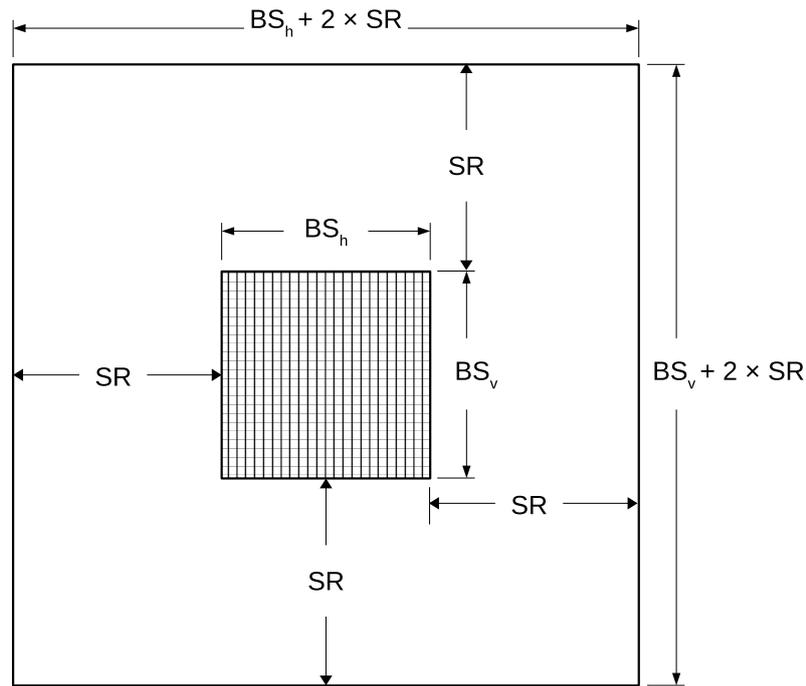
A ME é a responsável por explorar a redundância temporal presente em uma sequência de quadros, onde para cada bloco do quadro atual, a ME procura um bloco em quadros de referência previamente codificados que apresenta a maior similaridade com o bloco do quadro atual (SULLIVAN et al., 2012). Dentre os critérios de similaridades mais utilizados está a soma das diferenças absolutas (*Sum of Absolute Differences* – SAD) (KUHN, 1999), definido pela Equação 2.2.

$$SAD = \sum_{i=0}^{m-1} \sum_{j=0}^{n-1} |R_{ij} - O_{ij}| \quad (2.2)$$

O SAD é um dos critérios de similaridade mais utilizados devido à simplicidade do cálculo e facilidade de implementação em hardware. O cálculo do SAD consiste na soma das diferenças absolutas entre as amostras do bloco atual e dos blocos do quadro de referência. A Equação 2.2 apresenta o cálculo do SAD, onde  $R$  são as amostras do bloco de referência e  $O$  as amostras do bloco do quadro original.

O processo de busca da ME é realizada dentro de uma área de busca (AB), a fim de reduzir o custo de computação de ME. A Figura 2.5 apresenta a formação da área de busca em torno do bloco atual. O tamanho da área de busca considera o tamanho do bloco e o tamanho do intervalo de busca (*Search Range* - SR). Esse intervalo de busca define as distâncias horizontais e verticais máximas (em relação à posição do bloco atual) em que a busca é permitida. A Equação 2.3 define o tamanho da área de busca, onde  $BS_h$  e  $BS_v$

Figura 2.5: Definição da área de busca em torno do bloco atual.



Fonte: Próprio autor.

são os tamanhos horizontal e vertical do bloco e  $SR$  indica o intervalo de busca horizontal e vertical.

$$AB = (2 \times SR + BS_h) \times (2 \times SR + BS_v) \quad (2.3)$$

Para realizar a busca pelos blocos candidatos é utilizado algum algoritmo de busca. Diversos algoritmos são apresentados na literatura, os quais podem ser divididos em dois grandes grupos: algoritmos ótimos e algoritmos rápidos. Para comparar esses algoritmos, métricas como PSNR, *bitrate* e BD-rate ou BD-PSNR, são utilizadas.

Algoritmos ótimos, como o *Full Search* (FS) (HUANG et al., 2006), apresentam a melhor qualidade e *bitrate* que pode ser obtida para uma determinada área de busca. Porém, são algoritmos de força bruta, que comparam todos os possíveis blocos candidatos, isso faz com que esse algoritmo seja lento, além de realizar muitos acessos à memória. Já os algoritmos rápidos, como o *Test Zone Search* (TZS) (TANG; DAI; CAI, 2010) e *Diamond Search* (DS) (KUHN, 1999), seguem uma heurística para encontrar o melhor bloco candidato, reduzindo assim, a quantidade de comparações e acessos à memória, deixando a qualidade e o *bitrate* próximos dos resultados obtidos por algoritmos ótimos.

Assim que o bloco é encontrado, a ME deve gerar um vetor indicando o deslocamento deste bloco no quadro de referência, em relação à posição do bloco do quadro atual. Este vetor é chamado de vetor de movimento e deve ser enviado, junto com os

resíduos resultantes das diferenças entre o bloco original e o bloco candidato escolhido, para as demais etapas do codificador.

Os vetores de movimento servem para compensar o movimento dentro de um quadro e são, juntamente com o resíduo gerado entre o bloco escolhido e o bloco do quadro atual, codificados e transmitidos. Portanto, os vetores de movimento devem ser escolhidos de forma a minimizar este resíduo e assim, reduzir o número de bits para a codificação deste erro nas etapas posteriores.

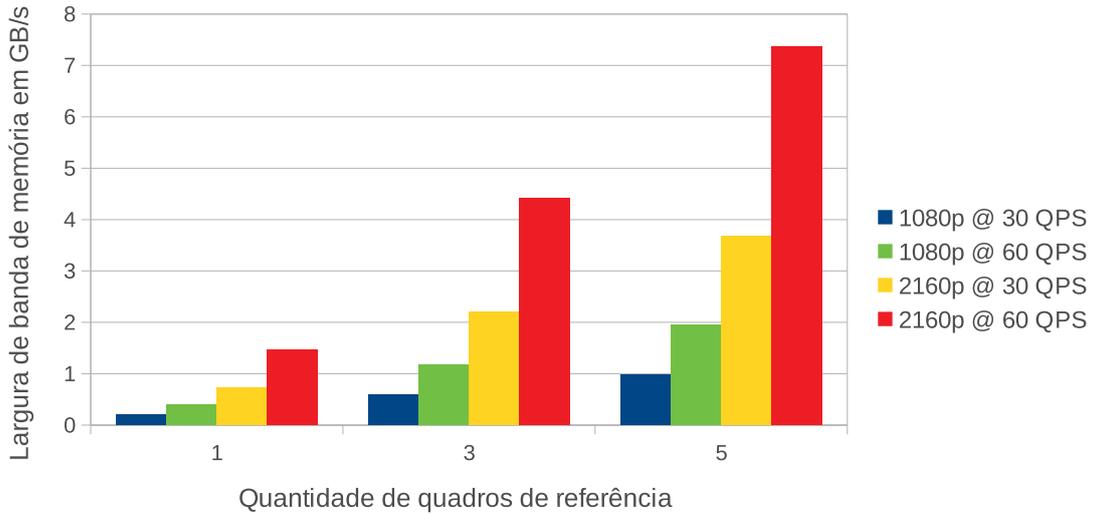
Nos codificadores de vídeo atuais, apenas o componente de luminância do bloco costuma ser considerado para a realização da estimação de movimento. Os vetores que são definidos para a luminância são reutilizados para a crominância. Esta estratégia é utilizada porque as informações de crominância são menos relevantes para a composição do vídeo (RICHARDSON, 2011), além de reduzir a quantidade de dados a serem processados.

O processo de busca da ME é bastante complexo e envolve um número elevado de acessos à memória. Para cada bloco codificado, a ME deverá buscar na memória uma área de busca, que pode ser um quadro de referência inteiro. Porém, geralmente se restringe a uma parte do quadro referente à região de interesse. Por exemplo, considerando um vídeo HD 1080p, com blocos de tamanho  $64 \times 64$  e área de busca  $256 \times 256$ , seria necessário buscar 506 áreas de busca para codificar um quadro apenas. E se o quadro de referência fosse todo armazenado em memória interna, seriam necessários 1,98MB de memória, considerando apenas amostras de luminância.

Os quadros de referência ficam armazenados em um *buffer* chamado *Decoded Picture Buffer* (DPB) (SULLIVAN et al., 2012). Usualmente, esse *buffer* fica alocado na memória externa devido à grande quantidade de informação, sendo essa tipicamente uma memória DRAM (*Dynamic Random-Access Memory*) (ZHENG et al., 2019) (SINANGIL et al., 2012). Além disso, o HEVC permite o uso de vários quadros de referência na ME durante o processo de codificação (BOSSSEN et al., 2012). Esta ferramenta gera melhores resultados de codificação ao custo de um aumento significativo de acessos à memória e aumento de complexidade (BOSSSEN et al., 2012).

Esse elevado número de acessos à memória externa realizada pela ME exige grande largura de banda de memória nesta comunicação. A largura de banda é a taxa de transferência máxima do canal de comunicação, que existe entre a memória e as unidades de processamento. Essa largura de banda de memória está diretamente relacionada aos parâmetros de codificação utilizados pelo codificador, sendo que alguns parâmetros acabam

Figura 2.6: Largura de banda de memória utilizando Level C.



Fonte: Próprio autor.

gerando mais acessos à memória que outros, tais como: quantidade de quadros de referência ( $QR$ ), taxa de quadros por segundo ( $QPS$ ), tamanho da área de busca ( $AB$ ), entre outros.

Para quantificar esses acessos em um cenário real, a Equação 2.4 pode ser utilizada para calcular a largura de banda de memória ( $LB_{MEM}$ ), em bytes por segundo, utilizando o esquema de reuso de dados Level C (TUAN; CHANG; JEN, 2002), onde  $TQ$  é o tamanho do quadro,  $AB$  se refere ao tamanho da área de busca,  $TB$  ao tamanho do bloco e  $W$  e  $H$  denotam largura e altura, respectivamente.

$$LB_{Mem} = QR \times QPS \times \frac{TQ_H}{TB_H} \times (AB_W \times AB_H + (\frac{TQ_W}{TB_W} - 1) \times TB_W \times AB_H) \quad (2.4)$$

A Figura 2.6 apresenta a largura de banda de memória externa necessária para a ME em diferentes cenários. São apresentados dados para codificação de vídeos HD 1080p e UHD 2160p a 30 e 60 QPS, usando um intervalo de busca de  $128 \times 128$  amostras e um tamanho de bloco de  $64 \times 64$ .

Conforme apresentado na Figura 2.6, dependendo da quantidade de quadros de referência e da taxa de quadros por segundo, uma maior largura de banda de memória é necessária. Esses parâmetros, em especial, tem relação direta com a largura de banda de memória externa, caso algum deles aumente, a largura de banda de memória aumenta na mesma proporção. Esta comunicação é realizada utilizando barramento externo ao chip e

é, de maneira geral, ordens de grandeza mais lenta que uma comunicação interna ao chip. Além disso, há a dissipação de potência dos dados que trafegam ao longo do circuito.

De acordo com alguns trabalhos da literatura, durante o processamento da ME, a energia consumida relacionada às memórias contribui com aproximadamente 90% do total de energia consumida, sendo que, deste total, metade do consumo é dado pelas memórias externas e a outra metade pelas memórias internas (ZATT et al., 2011). Em um decodificador HEVC, em torno de 74% do consumo de energia está relacionada aos acessos à DRAM (TIKEKAR et al., 2014). Desta forma, o projeto do codificador de vídeo deve considerar que um grande gargalo encontra-se na comunicação entre a memória externa e a unidade de processamento. Assim, existe uma forte necessidade de técnicas para reduzir o consumo de energia relacionado a comunicação com a memória externa, especialmente durante o processo de ME.

#### 2.4.1 Algoritmo de busca *Test Zone Search* – TZS

No algoritmo *Full Search* (FS), um bloco de tamanho  $N \times N$  do quadro atual é comparado com todos os blocos candidatos da área de busca no quadro anterior. Nesse processo, a Equação 2.2 é calculada para todas as  $(2 \times SR)^2$  posições de blocos candidatos dentro da área de busca, e o bloco com menor SAD (distorção mínima) é usado para predição, ou seja, a posição desse bloco dentro da janela de busca corresponde ao vetor de movimento. Diante do alto custo computacional do FS, métodos rápidos para estimar o movimento são utilizados.

Nesse sentido, um dos algoritmos de ME mais eficientes propostos na literatura é o algoritmo *Test Zone Search* (TZS) (TANG; DAI; CAI, 2010). Este algoritmo divide o processo de busca em quatro etapas: Predição do Vetor de Movimento (MVP), Busca Inicial, Busca Raster e Refinamento. Com esta estratégia o TZS atinge resultados próximos do ótimo, mas sem o custo proibitivo de uma busca exaustiva como a feita pelo algoritmo FS (HUANG et al., 2006). O TZS é considerado o algoritmo de ME de última geração e compatível com qualquer padrão de codificador de vídeo. Além disso, o TZS é adotado como padrão pelo software de referência HEVC, o HEVC Test Model (HM) (JCT-VC, 2014). Como o TZS é o foco deste trabalho, esta seção detalhará as quatro etapas que compõem o algoritmo.

A primeira etapa do TZS consiste na fase do MVP. Esta etapa é responsável por realocar a área de busca para o local mais promissor no quadro, com base nos blocos pre-

viamente codificados (TANG; DAI; CAI, 2010). Assim, são testados cinco preditores, que são herdados de vetores de movimento dos blocos previamente codificados. Estes preditores são: o bloco colocalizado, o bloco imediatamente da esquerda, o bloco imediatamente superior, o bloco imediatamente superior direito e a mediana dos quatro preditores anteriores (TANG; DAI; CAI, 2010). O MVP favorece bons resultados de codificação, uma vez que permite que o TZS faça menos comparações, diminuindo o tempo de codificação. Entretanto, o fato de que a área de busca nem sempre pode ser formada em torno do bloco colocalizado dificulta as implementações de hardware e o uso de estratégias eficientes de reutilização de dados (PAKDAMAN; HASHEMI; GHANBARI, 2020).

A Busca Inicial é a segunda etapa do algoritmo TZS. Ela começa no centro da área de busca definida pelo MVP e se expande em direção às bordas da área de busca usando um formato quadrado ou losangular (TANG; DAI; CAI, 2010). A Busca Inicial pode chegar até o tamanho máximo da área de busca, expandindo sempre em potência de dois. Essa etapa tem dois critérios de parada. O primeiro é quando atinge o limite da área de pesquisa. O outro ocorre quando nenhum bloco com maior similaridade ao bloco sendo codificado é encontrado após a execução dos três níveis de expansão. A execução de três níveis de expansão é uma estratégia do TZS para evitar os mínimos locais.

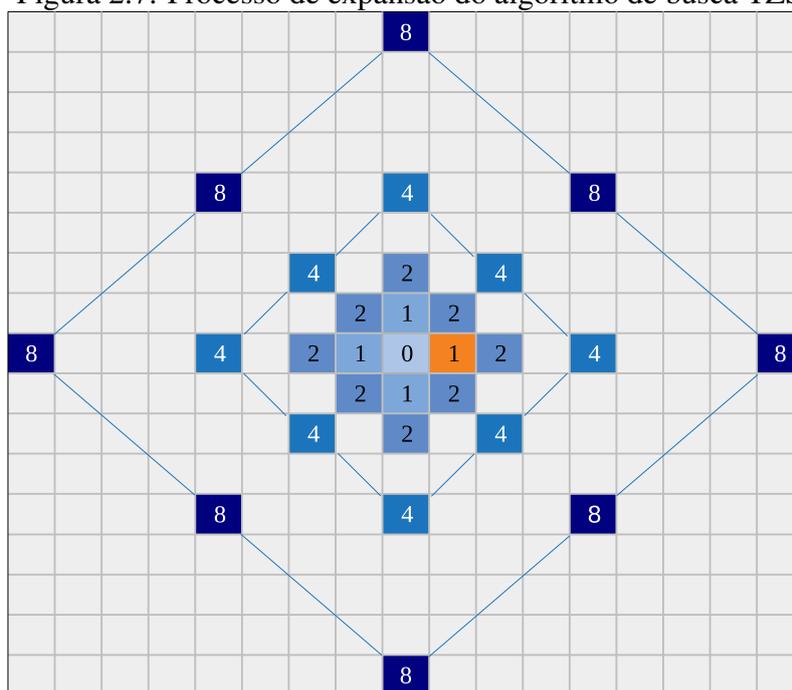
O processo de expansão do TZS é apresentado na Figura 2.7. Nessa figura o bloco zero foi previamente comparado na fase MVP e considerado o melhor preditor. Assim, o TZS expande em potência de dois, comparando blocos com distância de uma amostra do bloco central (primeira expansão), distância de duas amostras do bloco central (segunda expansão), distância de quatro amostras do bloco central (terceira expansão) e assim por diante.

Se nenhum dos pontos comparados nas três primeiras expansões, como mostrado na Figura 2.7, tiver um resultado melhor do que o bloco central, a Busca Inicial para e retorna o bloco 0 como melhor resultado. Neste exemplo, entretanto, o bloco laranja da primeira expansão alcançou um resultado melhor do que o bloco 0. Então, a quarta expansão é feita, seguindo a potência de dois, com uma distância de oito amostras do bloco central.

Neste exemplo, o melhor bloco foi aquele encontrado na primeira expansão, que é destacado em laranja na Figura 2.7. Como as três próximas expansões não retornaram um resultado melhor e a quarta expansão atingiu a borda da área de busca, a Busca Inicial finaliza a busca e retorna o bloco laranja como o melhor.

A terceira etapa do TZS é a Busca Raster. No entanto, a busca Raster só ocorre

Figura 2.7: Processo de expansão do algoritmo de busca TZS.



Fonte: Próprio autor.

quando o vetor de movimento da Busca Inicial é maior que a constante  $iRaster$ . Esta constante tem valor igual a 5, como padrão, no software de referência HM (JCT-VC, 2014). Quando executada, a Busca Raster aplica uma varredura completa com subamostragem de bloco em toda a área de pesquisa, retornando o bloco de melhor resultado encontrado. A subamostragem de blocos horizontal e vertical também é definida pela constante  $iRaster$ , então, por padrão, apenas um em cada cinco blocos horizontais e cinco verticais são comparados. Mesmo com esta subamostragem, esta é a etapa TZS com maior custo computacional (TANG; DAI; CAI, 2010).

Finalmente, a quarta e última etapa do TZS é o refinamento. O Refinamento assume o bloco de melhor resultado, da Busca Inicial ou da Busca Raster (quando realizada) e executa o mesmo padrão da Busca Inicial. Porém, enquanto que a Busca Inicial começa no centro e vai para as bordas da área de pesquisa, o Refinamento sempre atualiza seu centro com o melhor resultado encontrado em sua última iteração. A etapa de Refinamento tem duas condições de parada: i) quando a expansão atinge a borda da área de pesquisa, e ii) quando nenhum bloco mais semelhante ao que está sendo codificado é encontrado após uma expansão de dois níveis (TANG; DAI; CAI, 2010).

## 2.5 Métricas de qualidade e eficiência de codificação

Dada a dificuldade em avaliar a qualidade de um vídeo subjetivamente, métricas para medir objetivamente são utilizadas. As métricas objetivas ainda não substituíram por completo as métricas subjetivas, no entanto, elas são muito utilizadas, uma vez que são de fácil aplicação, e muito úteis para efeito de comparação (RICHARDSON, 2004). Uma das métricas objetivas mais utilizadas é o *Peak Signal to Noise Ratio* (PSNR). O PSNR é medido em uma escala logarítmica e é baseado no erro quadrático médio (*Mean Squared Error* - MSE) entre um quadro original e um quadro candidato, o qual apresenta degradação da qualidade (RICHARDSON, 2004). A Equação 2.5 apresenta o fórmula para cálculo do MSE, enquanto que a Equação 2.6 é utilizada para calcular o PSNR.

$$MSE = \sum_{i=0}^{w-1} \sum_{j=0}^{h-1} (C_{i,j} - O_{i,j})^2 \quad (2.5)$$

$$PSNR_{db} = 10 \log_{10} \frac{(2^n - 1)^2}{MSE} \quad (2.6)$$

Na Equação 2.5,  $C$  é o bloco de candidato e  $O$ , o bloco original. Ambos com  $w$  denotando a largura e  $h$  a altura. Além do  $PSNR$  ser uma métrica muito popular, também é fácil de ser calculada e é largamente utilizada como um método para comparação da qualidade de vídeos e imagens comprimidas e descomprimidas (RICHARDSON, 2004).

Entretanto, ao compararmos resultados de codificação de vários vídeos, outro parâmetro além da qualidade também é importante, o *bitrate* ou taxa de bits. O *bitrate* representa a quantidade de informação que está sendo gerada/transmitida. Desta forma, para avaliar a eficiência de codificação de um vídeo é necessário que esses dois parâmetros, PSNR e *bitrate*, sejam considerados. Entretanto, esses dois parâmetros não são equivalentes entre si e, quando apresentados separadamente, podem não deixar claro qual codificação foi mais eficiente.

Diante disso, o JCT-VC, através de um documento que descreve as Condições Comuns de Teste (CCT) (BOSSON, 2012), incentiva o uso de métricas baseadas na diferença de Bjøntegaard (BD) (BJONTEGAARD, 2001). Essas métricas são chamadas de *BD-bitrate* (BD-rate), medida em porcentagem e *BD-Peak Signal-to-Noise Ratio* (BD-PSNR), medida em dB. Desta forma, com o BD-rate ou o BD-PSNR é possível comparar resultados de eficiência de codificação (PSNR vs. *bitrate*) de forma precisa.

Segundo Bjøntegaard (2001) para calcular o BD-rate, as seguintes etapas devem

ser executadas: i) As sequências são codificadas em duas configurações de destino: uma considerada como referência e a outra objeto de comparação. Para cada alvo, são utilizados quatro valores de QP (*Quantization Parameter*): 22, 27, 32 e 37, conforme estabelecido no CTC, gerando oito pares (bitrate, PSNR), sendo quatro de cada alvo. ii) Um interpolador Hermite cúbico por peça é então usado para gerar duas curvas de taxa de distorção, dados quatro (*bitrate*, PSNR) pontos para cada curva. Essas curvas serão referidas como *REF* e *TEST*, representando o codificador HEVC de referência e a versão modificada, respectivamente. iii) A área diferencial entre as duas curvas é integrada, usando o eixo *X* como referência para BD-rate e o eixo *Y* para BD-PSNR. O valor final é obtido dividindo a integral por seu intervalo de integração.

Para fins de compreensão, a fórmula BR-rate é exibida na equação abaixo:

$$BD - rate = \frac{\int_a^b REF_{PSNR}(x) - TEST_{PSNR}(x)dx}{b - a} \quad (2.7)$$

Na Equação 2.7,  $REF_{PSNR}(x)$  e  $TEST_{PSNR}(x)$  representam, respectivamente, os valores PSNR obtidos com os codificadores referência e de teste, e  $a$  e  $b$  são o segundo valor mínimo e o segundo máximo PSNR de ambas as curvas. A equação para BD-PSNR é análoga, mas os valores de *bitrate* são usados no lugar dos valores de PSNR.

A métrica BD-rate pode ser interpretada como o quão menor/maior é o *bitrate* do sujeito de teste em comparação com sua referência considerando a mesma qualidade. Assim, a maioria dos trabalhos encontrados divulga apenas resultados do BD-rate, visto que esse valor é suficiente para determinar como um alvo se compara ao outro (BJONTEGARD, 2001).

## 2.6 Considerações finais do capítulo

Este capítulo apresentou alguns conceitos básicos utilizados na codificação de vídeo, além das ferramentas utilizadas pelos codificadores. Estas ferramentas são empregadas para explorar e reduzir as redundâncias de dados nos vídeos digitais, porém apresentam um alto custo em complexidade e acessos à memória. Foi apresentado um breve histórico dos padrões de codificação de vídeo e como os padrões anteriores influenciaram na construção do padrão HEVC. O fluxo de codificação do HEVC foi apresentado, destacando a importância de cada um dos módulos do codificador para geração do *bitstream* final.

Entre os módulos apresentados, a ME é destaque, pois é uma das ferramentas presentes em todos os padrões de codificação, sendo responsável pelos maiores ganhos em taxa de compressão. Entretanto, é o módulo do codificador que apresenta a maior complexidade e o que mais realiza acessos à memória, sendo esse um dos maiores gargalos nos codificadores de vídeo atuais. Além disso, foram apresentadas as etapas de codificação do algoritmo TZS, o qual é um dos algoritmos mais eficientes encontrados na literatura, além de ser utilizado nesta tese. Esse algoritmo se divide em quatro etapas: MVP, Busca Inicial, Busca Raster e Refinamento e aplica heurísticas e buscas subamostradas na tentativa encontrar o melhor resultado com um menor número de comparações.

Além dos problemas relacionados aos acessos à memória externa durante o processo da ME, foi também discutida a importância de reduzir o consumo energético relacionado à memória, principalmente para atender as demandas do processamento em tempo real de vídeos de alta definição, além das restrições de desempenho e consumo de energia apresentados por dispositivos móveis alimentados por bateria.

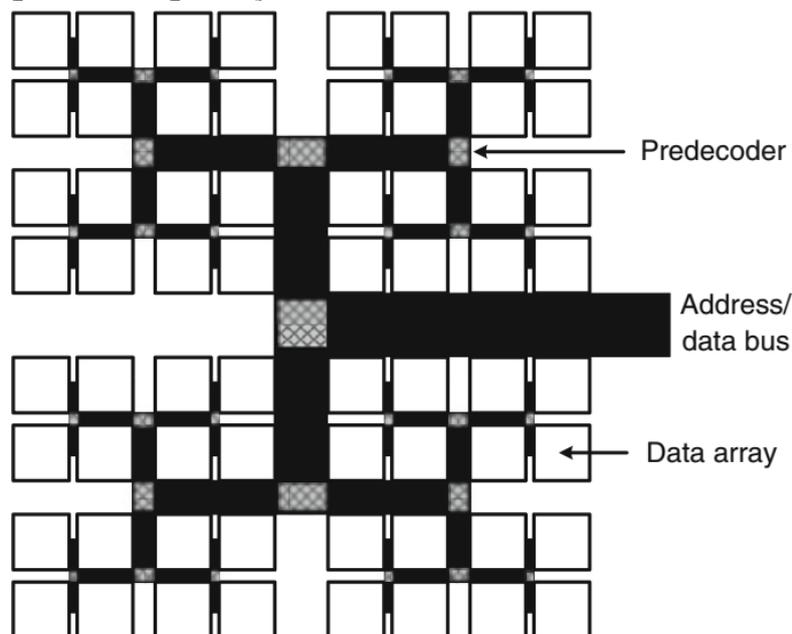
### 3 FUNDAMENTOS DO SISTEMA DE MEMÓRIA E SUAS TECNOLOGIAS

Neste capítulo serão discutidas as principais características de um sistema de memória. Esse sistema pode estar presente tanto em aplicações de propósito geral quanto em aplicações específicas como codificadores de vídeo. Serão apresentados o funcionamento de uma hierarquia de memória e quais os tipos de memória que geralmente são utilizadas para compor esse sistema. Antes de introduzir tecnologias específicas de memória, serão abordados os componentes eletrônicos básicos dos quais as memórias são compostas e seu funcionamento.

#### 3.1 Sistema de memória

Um chip de memória consiste de milhões a bilhões de células de memória e usa endereçamento binário como entrada para encontrar as células correspondentes (YU; WANG, 2014). Desta forma, as operações de leitura e escrita podem ser executadas. Para realizar essas tarefas de forma eficiente, os barramentos são distribuídos em um formato chamado de árvore-H, o qual busca igualar o comprimento dos mesmos a fim de manter o custo de energia e o tempo de atraso próximos para cada célula de memória (YU; WANG, 2014). A organização das células e barramentos pode ser visualizado na Figura 3.1.

Figura 3.1: Organização interna de uma memória com a árvore-H.



Fonte: Yu e Wang (2014).

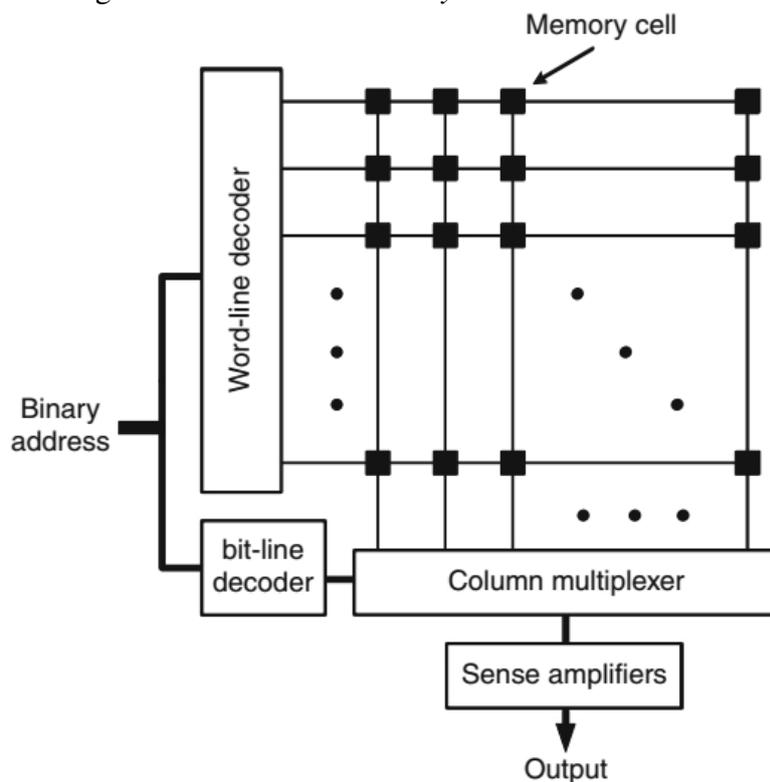
As células são divididas em vários *arrays* de dados, que são conectados pela rede

da árvore-H. O endereço de entrada é logicamente dividido em duas partes. A primeira parte do endereço indica a posição do *array* de dados, na qual as células de destino são mantidas. A segunda parte do endereço revela a posição das células de destino dentro do *array* de dados. O identificador do *array* de dados é usado pelos predecessores ao longo dos caminhos da rede da árvore-H para rotear os sinais elétricos para atingir o *array* de dados (YU; WANG, 2014). A maior vantagem da rede da árvore-H é que ela pode garantir a mesma distância de propagação do sinal para cada *array* de dados. Isso é importante para garantir que o sistema seja determinístico e que a latência de acesso seja fixa.

As unidades de armazenamento na memória são os *arrays* de dados. Todas as células de memória estão nos cruzamentos entre as *wordlines* e as *bitlines*. As *wordlines* e as *bitlines* são fios que propagam sinais, os quais geram certos atrasos devido as resistências e capacitâncias parasitas do fio. Portanto, quanto maior o *array* de dados, maior a latência de acesso que pode ser esperada. Cada célula armazena um bit de informação, representado pela lógica alta ou baixa, e seu valor pode ser lido pelos *sense amplifiers*. Se cada célula estiver conectada diretamente com as entradas e saídas externas, bilhões de entradas e saídas serão necessárias, o que é praticamente impossível de alcançar. Desta forma, decodificadores são usados para obter o endereço binário das células designadas (YU; WANG, 2014). A Figura 3.2 apresenta um esquemático com a estrutura do *array* de dados.

No *array* de memória, as linhas de saída dos decodificadores são conectadas às *wordlines*, as quais, dado um endereço específico, habilitam uma linha inteira do *array* de dados. Como nas *bitlines* os sinais elétricos são bidirecionais, ou seja, as *bitlines* podem direcionar a célula na operação de escrita e a célula pode direcionar a *bitline* na operação de leitura, as linhas de saída do decodificador de *bitlines* são conectadas a um multiplexador, que seletivamente permite que o sinal elétrico de uma coluna específica passe.

O principal objetivo do circuito de leitura da saída das memórias é distinguir os estados biestáveis das suas células. Para tecnologias de memória convencionais como SRAM e DRAM, o estado da memória é representado pela tensão elétrica:  $V_{DD}$  para lógica “1” e  $GND$  para lógica “0”. Portanto, esse circuito pode ser projetado como um comparador de tensão que compara a tensão do estado com a tensão intermediária, ou seja,  $V_{DD}/2$ . No entanto, na prática, devido ao compartilhamento de carga entre a célula de memória e o capacitor parasita da *bitline*, a margem de tensão detectável é reduzida para menos de um décimo do valor anterior, e um circuito de leitura mais sensível é

Figura 3.2: Estrutura do *array* de dados da memória.

Fonte: Yu e Wang (2014).

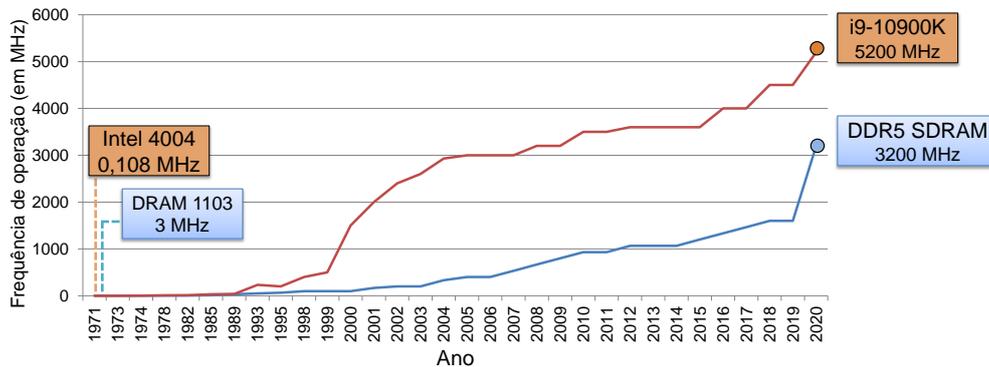
necessário (YU; WANG, 2014).

Nesse sentido, os circuitos dos *sense amplifiers* são projetos como *latches*, os quais são capazes de detectar diferenças de tensão menores que  $100mV$  em escala de nanossegundos. Quando uma leitura é realizada, a tensão da entrada aumenta ou diminui ligeiramente devido a carga compartilhada no *latch*. Essa variação é determinado pela razão entre a capacitância da célula de memória e a capacitância da *bitline*, e quando o *array* é muito grande, a variação se tornará muito pequena para ser detectado (YU; WANG, 2014). Essa é a principal razão pela qual a memória não pode ser transformada em um único *array* e a sua divisão é necessária. Na próxima seção será apresentado e discutido o funcionamento e as vantagens da utilização de hierarquias de memória em sistemas computacionais.

### 3.2 Hierarquia de memória

Nas últimas décadas o desempenho dos processadores aumentou a uma taxa de 50% a 100% por ano, enquanto que a velocidade das memórias DRAM aumentou a uma taxa de 7% ao ano (VERMA; MARWEDEL, 2007). A Figura 3.3 apresenta o crescimento

Figura 3.3: Frequência de operação dos processadores e do barramento das memórias DRAM ao longo das últimas décadas.



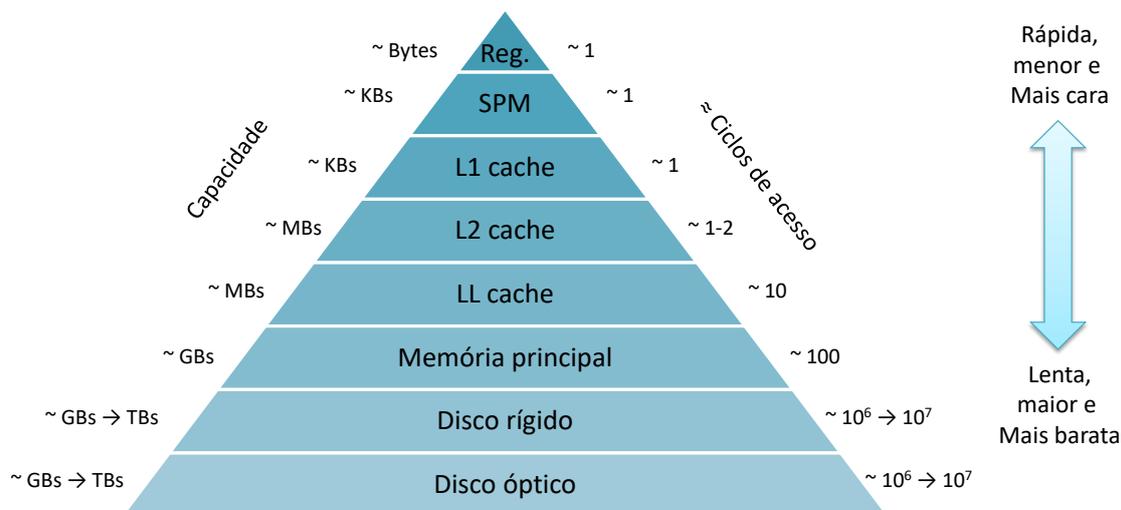
Fontes: Intel (2021) e Micron (2021)

da frequência de processadores e dos barramentos de memórias DRAM ao longo das últimas décadas. Nota-se nessa figura que essa taxa de crescimento teve sua maior diferença no ano de 2004. Após, esse período iniciou-se o uso dos multiprocessadores e poucos anos antes das memórias DDR SDRAM, os quais atingem grandes desempenhos através do uso de diversas técnicas. A partir daí a diferença de frequência entre processador e DRAM estabilizou, ainda que a frequência dos processadores seja em média  $3,5 \times$  maior.

Para amenizar essa diferença, nos projetos de arquiteturas de computadores modernas, o armazenamento de instruções e dados segue um arranjo hierárquico chamado hierarquia de memória, a qual aproveita a localização e o desempenho das tecnologias de memória. O projeto de uma hierarquia de memória é um dos principais componentes dos sistemas de computadores modernos (XIE, 2014). A importância da hierarquia de memória aumenta com os avanços no desempenho dos microprocessadores. O projeto tradicional de uma hierarquia de memória consiste em memórias internas (como SRAM ou eDRAM) utilizadas como Caches internas, e memórias DRAM como memória principal e *drivers* de disco rígido magnéticos (HDD) ou drives de estado sólido (SSD) para o armazenamento permanente em massa. A Figura 3.4 apresenta uma hierarquia de memória com vários níveis e tipos de memória. Quanto mais próxima a memória é colocada do microprocessador, topo da hierarquia, menor será a latência e maior a largura de banda, com a penalidade de ter menor capacidade de armazenamento (XIE, 2014).

Entretanto, memórias mais rápidas possuem custo (que pode ser definido em custo econômico, área e consumo energético) mais alto por byte e, portanto, a saída é a utilização de hierarquias de memória eficiente, em que o sistema de memória é formado por uma combinação de memórias mais caras e rápidas com memórias mais baratas e lentas, onde memórias pequenas e mais rápidas são colocadas mais próximas ao processador.

Figura 3.4: Níveis de uma hierarquia de memória



Fonte: Yu e Wang (2014)

Nesse sentido e até muito recentemente, as memórias Caches foram considerados como sinônimo de hierarquias de memória e, na verdade, elas ainda são a memória padrão a ser usada em processadores de propósito geral. Sua principal vantagem é que elas trabalham de forma autônoma e são altamente eficientes no gerenciamento de seus conteúdos para armazenar o atual contexto de trabalho da aplicação (VERMA; MARWEDDEL, 2007). Além das Caches há também as memórias *scratchpads*, que são memórias mais simples que as memórias Cache, mas o controle dos dados armazenados na memória fica a cargo da aplicação. Nas próximas subseções serão discutidas de forma mais detalhada esses dois tipos de memória.

### 3.2.1 Memória Cache

O princípio básico das memórias Cache é o de manter uma cópia dos dados e instruções mais utilizados recentemente para que os mesmos não precisem ser buscados na memória principal (JACOB; NG; WANG, 2007). Isso é vantajoso devido ao princípio da localidade, temporal e espacial, presente na grande maioria dos algoritmos. Na localidade temporal diz que um dado acessado recentemente tem mais chances de ser usado novamente, do que um dado usado há mais tempo, já a localidade espacial trata sobre informações que tem maior probabilidade de serem acessadas em endereços próximos àqueles acessados recentemente (JACOB; NG; WANG, 2007). Como já discutido anteriormente, a codificação de vídeo é uma aplicação de grande importância atualmente e sua

dinâmica de execução é baseada na exploração das redundâncias temporais e espaciais.

Com esse sistema, todo dado a ser lido ou escrito em memória pelo processador antes passa para a Cache. Se o dado estiver na Cache, a operação é feita nela e não precisa ir até a memória principal. Caso contrário, um bloco inteiro de dados é trazido da memória principal e salvo na Cache. Após isso é que o processador realizará a tarefa com o dado.

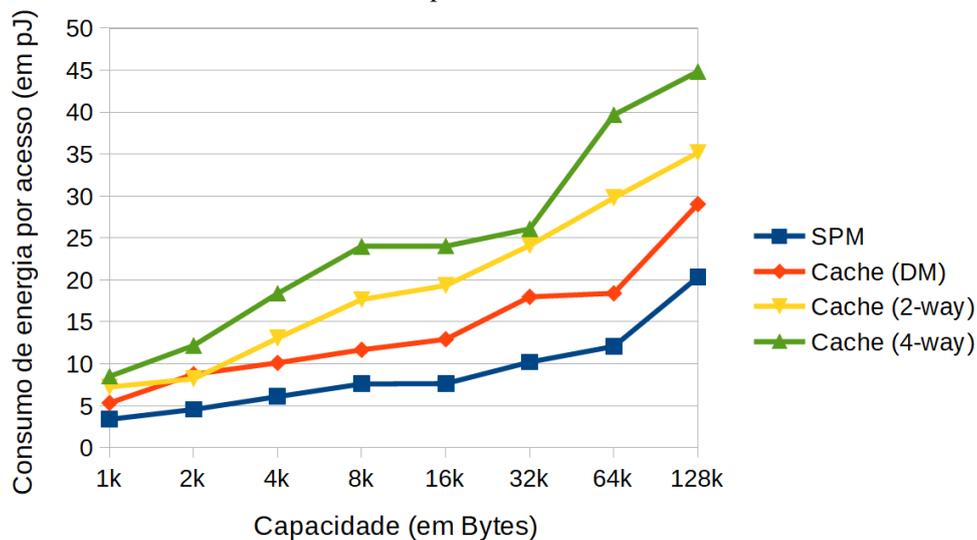
Sendo assim, o desempenho do computador ao acessar um dado de memória é probabilístico. Para cada dado a ser acessado há uma probabilidade dele estar na memória Cache. Se isso ocorrer haverá um *Cache hit* e o sistema ganha muito tempo com isso. Caso contrário, ocorre um *Cache miss* e o desempenho é bastante prejudicado. De acordo com Jacob, Ng e Wang (2007) para aumentar a probabilidade de um determinado dado estar na memória Cache ao invés da memória principal há três estratégias: aumentar o tamanho da memória Cache; mudar a função de mapeamento e mudar a política de substituição de dados.

Como a memória Cache trabalha armazenando cópias de dados da memória principal, quanto maior for a memória Cache, mais dados ela é capaz de armazenar, sendo assim, maior a probabilidade do processador buscar por um dado e ele estar na Cache. Entretanto, é importante observar que quanto maior for a Cache maior será o consumo de energia dessa memória. Esse comportamento pode ser observado na Figura 3.5, onde é apresentado um comparativo de consumo de energia por acesso entre diferentes configurações de memórias Cache e memórias *scratchpad*. Para obter os dados apresentados nessa figura, foi utilizada a ferramenta Cacti da HP (CACTI, 2018) para simular diferentes configurações de memórias internas, Caches e *scratchpad*, para para diferentes tamanhos de memórias.

A função de mapeamento diz respeito a estratégia utilizada para determinar onde cada dado da memória principal estará na Cache (JACOB; NG; WANG, 2007). Ela determina onde cada dado da memória principal será copiado na Cache caso ele seja acessado. Isso é importante porque o processador vai seguir essa mesma estratégia para conseguir localizar se o dado está, ou não na Cache. Há três tipos de mapeamento: direto; totalmente associativo e associativo por conjunto.

O intervalo de Caches entre o mapeamento direto e o totalmente associativo corresponde aos vários níveis de associatividade por conjuntos (JACOB; NG; WANG, 2007). Caches de mapeamento direto têm conjuntos com apenas um bloco em cada conjunto, cada bloco é seu próprio conjunto. Caches totalmente associativas possuem apenas um

Figura 3.5: Comparativo de consumo de energia por acesso entre diferentes configurações de memórias Cache e memórias *scratchpad*.



Fonte: Próprio autor.

conjunto que engloba todos os blocos, assim, todos os blocos na Cache estão no mesmo conjunto. As Caches conjunto associativas atingem um equilíbrio entre essas duas opções. Uma Cache conjunto associativa tem mais de um conjunto (se apenas um, a Cache seria totalmente associativa), e cada conjunto na Cache incorpora mais de um bloco em seu conjunto (se apenas um, a Cache seria mapeamento direto).

Conforme pode ser observado na Figura 3.5, a associatividade utilizada pelas memórias Cache impacta no consumo de energia por acesso à essas memórias, onde uma Cache com associatividade *4-way* consome  $2\times$  mais energia que uma Cache diretamente mapeada.

As políticas de substituição de dados na Cache podem ser realizadas de acordo com a especificação do projetista e de forma que se obtenha os melhores resultados. No entanto, há formas de políticas de substituição clássicas, as quais são muito utilizadas, tais como: randômica; FIFO e LRU. Nos mapeamentos associativo e associativo por conjunto uma outra política deve ser adotada. Quando a memória Cache enche e um novo bloco precisa ser armazenado, o sistema de memória deve escolher que bloco deve ser removido para dar espaço ao novo bloco. No mapeamento direto isso não existe porque cada bloco sempre fica na mesma posição.

Na substituição randômica o sistema simplesmente escolhe aleatoriamente o bloco que deve ser removido. Ele sai da Cache dando lugar ao novo bloco que foi acessado. Este método tem a vantagem de ser muito fácil de implementar e, por consequência, rápido de executar. Porém ele pode não ser muito eficiente. Já o FIFO (*First-In First-Out*) adota o

princípio de fila. Aquele bloco que chegou primeiro, está há mais tempo na Cache, já se beneficiou bastante e deve então dar lugar ao novo bloco. No LRU (*Least-Recently Used*), ou “menos usado recentemente” aplica o princípio da localidade temporal e torna-se por isso mais eficiente na maioria dos casos. Nesta política o sistema escolhe o bloco que menos foi utilizado recentemente e o remove. Isso faz com que fiquem na Cache aqueles blocos que são acessados mais vezes nos últimos instantes.

As configurações como tamanho, associatividade e políticas de substituição impactam no desempenho e consumo energético do sistema de memória. As arquiteturas de processadores atuais incorporam grandes memórias Cache as quais são projetadas e organizadas em diferentes níveis hierárquicos: registradores, primeiro nível da Cache (*Level 1 Cache – L1C*), segundo nível ou intermediários (*Level 2 Cache – L2C*) e o último nível (*Last Level Cache – LLC*). O primeiro nível, localizado mais perto do núcleo de processamento, consiste no armazenamento de dados e instruções. Por sua vez, os níveis localizados mais acima (L3C e/ou LLC) são utilizados para armazenar os dados que são compartilhados pelos núcleos do multiprocessador.

Os acessos aos Caches são acessos a pequenas memórias e, portanto, podem exigir menos energia por acesso do que grandes memórias. No entanto, para as Caches, é necessário que um hardware verifique se a Cache tem ou não uma cópia válida das informações associadas a um determinado endereço. Essa verificação envolve a verificação dos campos de *tag* das Caches, os quais contém um subconjunto relevante dos bits de endereço (MARWEDEL, 2011). Além disso, ler e armazenar essas *tags* requer energia adicional.

Além disso, no domínio de sistemas embarcados, e em especial nos codificadores de vídeos embarcados, os aplicativos que podem ser executados no processador são restritos e a principal vantagem das Caches torna-se um passivo. Nesse contexto, as memórias Caches apresentam alto consumo de energia, baixo desempenho e limites exagerados de tempo de execução no pior caso (VERMA; MARWEDEL, 2007), tornando o projeto da hierarquia de memória um ponto importante para um sistema eficiente.

### 3.2.2 Memória *scratchpad*

As memórias *scratchpad*, ao contrário de uma Cache, consistem de apenas um *array* de memória de dados e uma lógica para decodificação de endereço. A ausência de *tags* e de lógica para comparar endereços tornam a memória *scratchpad* mais eficiente em termos de área e consumo de energia (VERMA; MARWEDEL, 2007). A Figura 3.5

apresenta o comparativo de consumo de energia por acesso entre diferentes tamanhos de memórias *scratchpad* e memórias Caches de tamanho e associatividade variáveis. A partir dessa figura, pode-se observar que a energia por acesso em uma memória *scratchpad* é sempre menor do que em Caches do mesmo tamanho. Em particular, a energia consumida por uma memória *scratchpad* de 4KB é  $3\times$  menor que a energia consumida por uma memória Cache de mesmo tamanho e associatividade *4-way*.

No entanto, as memórias *scratchpad* exigem suporte explícito da aplicação para sua utilização. Uma atribuição cuidadosa de instruções e dados é um pré-requisito para uma utilização eficiente da memória *scratchpad*. Um ponto positivo é que essa atribuição de instruções e dados permite limites de execução em pior caso mais ajustados ao sistema.

Assim, memórias *scratchpad* exigem o conhecimento prévio sobre o acesso aos dados, enquanto que a memória Cache consegue ter um bom desempenho onde o padrão de acessos não é conhecido. Nesse sentido, pensando no funcionamento do codificador de vídeo e dos algoritmos de busca, algoritmos como o TZS (que podem deslocar totalmente a área de busca de forma imprevisível) podem se beneficiar de Caches. Enquanto que as memórias *scratchpad* podem ser mais indicadas se a pesquisa ocorrer sempre dentro de uma janela de busca colocalizada.

Desta forma, diante de tantas possíveis configurações que podem ser empregadas na construção de uma hierarquia de memória eficiente, fica claro o desafio em encontrar essa configuração para o módulo de estimação de movimento de um codificador de vídeo de alta definição.

### 3.3 Tecnologias de memória

O grande progresso tecnológico proporcionado pelo silício nos últimos 50 anos também impulsionou o uso de memórias baseadas em silício como SRAM, DRAM e, mais recentemente, Flash, para as atuais densidades multi-gigabit disponíveis em um único chip a um custo cada vez menor por bit (HONG; AUCIELLO; WOUTERS, 2014). No entanto, estes componentes apresentam alguns problemas como, por exemplo, a escalabilidade das memórias, a qual está chegando a seu limite, conforme é relatado em um artigo da Samsung e da Intel (KANG et al., 2014). Isso é mais crítico em memórias SRAM e DRAM, mas também para memórias Flash, e pode ser atribuído em grande parte ao fato de que essas memórias são baseadas em uma carga para armazenar a informação (HONG; AUCIELLO; WOUTERS, 2014).

Além disso, há ainda questões relacionadas ao consumo de energia, tanto na transferência dos dados quanto no armazenamento em memória. Por exemplo, os dados transferidos entre memória externa e processador consomem  $\approx 100\times$  mais energia que uma operação de ponto flutuante feita pelo processador (KECKLER et al., 2011). Ainda, como a DRAM é uma memória volátil, é essencial atualizar as cargas armazenadas periodicamente. Caso contrário, as informações são perdidas com o tempo (KANG et al., 2014). Esse processo de atualização das informações é chamado de *refresh*. Já nas memórias SRAM sofrem do mesmo problema que as novas tecnologias CMOS em escala nanométrica, um crescimento exponencial na corrente de *leakage* devido à redução do *threshold* do transistor (SHAFIQUE et al., 2014).

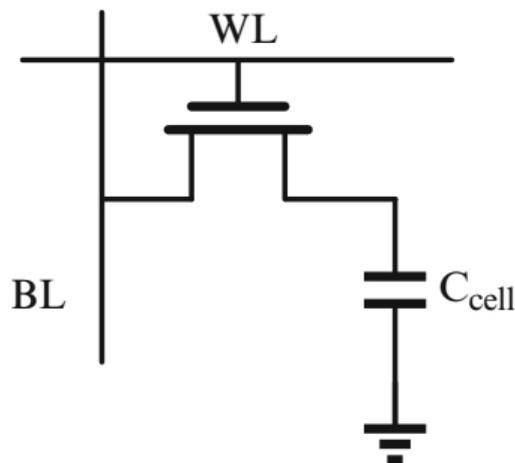
Portanto, há um grande interesse na pesquisa por alternativas de memórias baseadas em outros mecanismos de armazenamento e tecnologias que permitam um novo nível de escalabilidade e menos problemas em consumo de energia. As próximas seções apresentarão o funcionamento das tradicionais tecnologias de memória e das tecnologias emergentes baseadas em memórias não voláteis, as quais reduzem os problemas acima mencionados, mas acabam introduzindo outros problemas.

### 3.3.1 Memórias voláteis

As memórias semicondutoras referem-se aos dispositivos físicos baseados em transistores de silício em sistemas de computador que são usados para armazenar temporariamente ou permanentemente programas ou dados. Em contraste com mídias de armazenamento como discos rígidos e discos ópticos, dos quais os acessos precisam seguir uma ordem predeterminada devido às limitações do drive mecânico, as memórias semicondutoras possuem a propriedade de acesso aleatório, o que significa que o tempo necessário para acessar qualquer dado é idêntico, independentemente da localização dos dados.

De acordo com a volatilidade, as memórias semicondutoras podem ainda ser classificadas em memória volátil e memória não volátil. A memória volátil armazena dados como sinais elétricos e perde seus dados quando o chip é desligado. Atualmente, as memórias voláteis mais utilizadas são a SRAM e a DRAM, cujos dados são indicados pelos níveis de tensão elétrica. Pelo contrário, a memória não volátil é capaz de reter seus dados mesmo quando o chip está desligado, já que seus dados são preservados principalmente por estados não elétricos. Nas próximas seções, os princípios e as operações das

Figura 3.6: Célula de memória DRAM.



Fonte: Yu e Wang (2014).

memórias DRAM e SRAM serão apresentadas e discutidas.

### 3.3.1.1 Dynamic Random Access Memory – DRAM

Nessa seção será apresentada a memória DRAM (*Dynamic Random Access Memory*). A ideia por trás da DRAM está na simplicidade, onde cada célula DRAM consiste em apenas um capacitor e um transistor de acesso. Os dados “0” ou “1” são representados pelo fato de o capacitor estar totalmente carregado ou descarregado. No entanto, a carga elétrica no capacitor irá gradualmente diminuir, e após um período de tempo, a tensão no capacitor é tão baixa que o *sense amplifier* não consegue diferenciar entre “0” e “1”. Portanto, o tempo de retenção para DRAM é finito e todas as células DRAM necessitam serem lidas e escritas periodicamente para garantir a integridade das informações. Normalmente, as células são atualizadas uma vez a cada  $32ms$  ou  $64ms$  (YU; WANG, 2014) (MUTLU; SUBRAMANIAN, 2015). Esse processo é conhecido como *refresh*. A Figura 3.6 apresenta o diagrama da célula DRAM com um transistor de acesso e o capacitor.

Para escrever em uma célula DRAM, primeiramente a *bitline* é definida como alta ou baixa, baseado no valor a ser escrito. Depois, o transistor de acesso é ativado pela *wordline* e o capacitor na célula selecionada é carregado em “1” ou descarregado para “0”. Como o acesso ocorre através de um transistor NMOS, existe uma queda de  $V_{th}$  durante a escrita “1”. Para evitar esta queda e manter um longo período de *refresh*, a tensão na *wordline* é normalmente incrementada para  $V_{PP} = V_{DD} + V_{th}$  (YU; WANG, 2014).

Para a leitura em uma célula DRAM, a *bitline* é pré-carregada para  $V_{DD}/2$  e, em

seguida, a *wordline* é ativada. Devido ao compartilhamento de carga, a tensão da *bitline* diminuirá ou aumentará levemente, dependendo da tensão que o capacitor foi previamente carregado, i.e.,  $V_{DD}$  ou  $0V$ . Se for previamente carregado, o compartilhamento de carga aumentará um pouco a tensão da *bitline*; caso contrário, alguma carga será distribuída da *bitline* para o capacitor da célula. Em ambos os casos, a tensão do capacitor de armazenamento será alterada após a operação de leitura (YU; WANG, 2014). Assim, a operação de leitura é chamada de destrutiva e uma gravação instantânea é necessária.

Segundo Yu e Wang (2014) em memórias DRAM modernas, a capacitância do capacitor de armazenamento é muito menor que a capacitância da *bitline*. Normalmente, a capacitância de um capacitor de armazenamento é um décimo da capacitância da *bitline*, a qual está conectada a centenas ou milhares de outras células. Desta forma, quando a capacitância contida em um capacitor de armazenamento é colocada na *bitline*, a tensão resultante na *bitline* é pequena e difícil de medir em um sentido absoluto. Nas DRAM, esse problema de detecção de tensão é resolvido através do uso de um *sense amplifier* diferencial, o qual compara a tensão da *bitline* a uma tensão de referência.

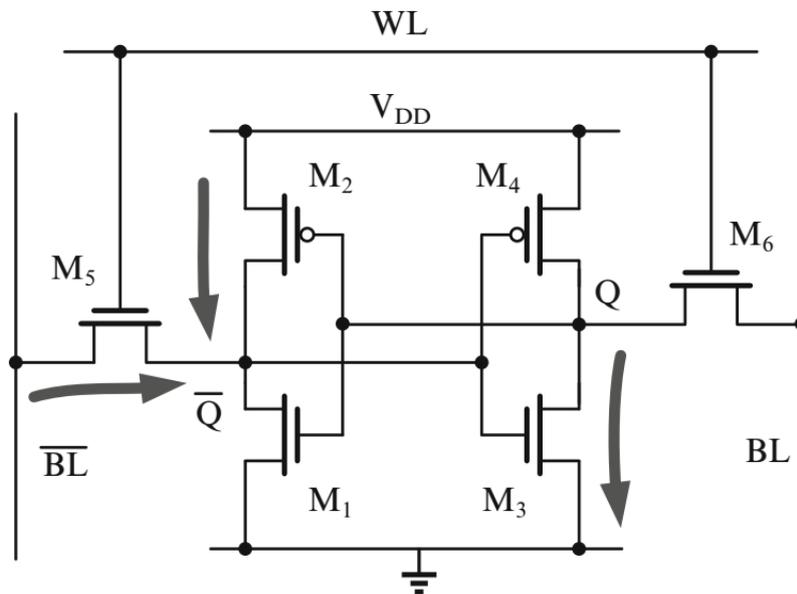
Entretanto, o uso do *sense amplifier* diferencial, introduz alguns requisitos na estrutura do *array* de dados da DRAM. Em vez de uma única *bitline*, um par de *bitlines* precisa ser usado para detectar o valor de tensão contido em qualquer célula DRAM. Além disso, para garantir que os valores de tensão e capacitância no par de *bitlines* sejam muito próximos, as *bitlines* devem ser semelhantes em termos de tamanho e número de células (YU; WANG, 2014).

### 3.3.1.2 Static Random Access Memory – SRAM

A SRAM (*Static Random Access Memory*) é o tipo de memória mais usada como memória interna em processadores, tais como Caches e scratchpad (YU; WANG, 2014). O termo estático se refere do fato de que a célula não precisa ser atualizada como a DRAM e os dados podem ser retidos enquanto tiver energia. Como a SRAM usa o mesmo processo de fabricação dos processadores atuais, a sua integração com os processadores é facilitada. Uma célula CMOS SRAM típica consiste em seis transistores, como apresentado na Figura 3.7. O flip-flop formado pelos transistores  $M1-M4$  armazenam a informação do bit. Os transistores  $M5$  e  $M6$ , conectados com a *wordline* e duas *bitlines*, são usados como transistores de acesso para selecionar as células de destino.

Existem três estados de operação para uma célula SRAM: escrita, leitura e *standby* (YU; WANG, 2014). Para uma operação de escrita, o valor a ser escrito precisa ser

Figura 3.7: Célula de memória SRAM com 6 transistores e a indicação dos caminhos de *leakage*



Fonte: Yu e Wang (2014).

aplicado em ambas as *bitlines*, i.e.,  $BL$  e  $\overline{BL}$ , as quais são complementares. Por exemplo, desejamos escrever “0” na célula, i.e.,  $Q$  em “0” e  $\overline{Q}$  em “1”; o  $BL$  vai para baixo e  $\overline{BL}$  para alto. Depois que  $M_5$  e  $M_6$  forem ativados pela  $WL$  em “1”, os transistores *drivers* da *bitline* substituirão o valor anteriormente armazenado. Para substituir facilmente o estado anterior no flip-flop, é necessário que os transistores da *bitline* sejam mais fortes do que os transistores no flip-flop.

Para uma operação de leitura, ambas as *bitlines* são pré-carregadas antes do início do ciclo de leitura e a ativação da *wordline* significa o início da operação de leitura. Por causa das tensões opostas em  $Q$  e  $\overline{Q}$ , uma das *bitlines* será descarregada pela célula, sendo isso detectado pelo *sense amplifier* da *bitline*. Em uma abordagem de detecção do modo de tensão, o sinal da diferença de tensão da *bitline*  $\Delta V$  ( $V_{BL}$  menos  $V_{\overline{BL}}$ ) determina o valor do bit armazenado. Um  $\Delta V$  em dezenas de milivolts já é significativo o suficiente para distinguir com eficiência qual *bitline* está sendo descarregada. Por exemplo, suponha que o bit armazenado seja “1” em  $Q$  e “0” em  $\overline{Q}$ , e assim que o  $WL$  for declarado, o  $BL$  irá descarregar para “0”; quando um  $\Delta V$  positivo de dezenas de milivolts é gerado, o *sense amplifier* amplificará a pequena diferença de tensão com um retorno positivo e, finalmente, a lógica de saída será “1”.

Quando a  $WL$  está conectada ao  $GND$ , desligando os dois transistores de acesso, a célula está no estado de espera (*standby*). Durante o *standby*, os dois inversores de cross-acoplamento reforçam-se mutuamente devido à realimentação positiva, o valor é preser-

vado enquanto a energia é fornecida. Um problema proeminente relacionado à SRAM no estado de *standby* é o *subthreshold leakage*. O *subthreshold leakage* é a corrente dreno – fonte do transistor quando a tensão gate – fonte é menor que a tensão de *threshold*. A corrente de *subthreshold* depende exponencialmente da tensão de *threshold*, o que resulta em uma corrente de *subthreshold* grande quando tecnologias submicrônicas são empregadas. A Figura 3.7 indica os três caminhos de *leakage* em uma célula SRAM, assumindo que o bit armazenado em  $Q$  seja “1”. Independentemente do valor armazenado na célula, sempre haverá três transistores consumindo corrente de *leakage*. Ainda,  $BL$  e  $\overline{BL}$  são sempre pré-carregados em  $V_{DD}$  para agilizar as futuras operações de leitura.

Em comparação com outras tecnologias de memória, a SRAM é capaz de fornecer velocidade de acesso mais rápida, mas a vantagem surge como uma compensação em relação à densidade e à potência. Como uma célula SRAM requer área para seis transistores, a SRAM possui uma densidade muito limitada sendo, portanto, mais cara que outras tecnologias de memória. Além disso, apresenta elevado consumo de energia devido ao problema *leakage* quando em *standby*. Portanto, a SRAM atende melhor em aplicações onde o alto desempenho é a principal preocupação e a capacidade não é significativa, i.e., em Caches para processadores.

### 3.3.2 Memórias não voláteis

Além das tecnologias acima mencionadas, as quais estão bem estabelecidas no mercado, existem outras tecnologias emergentes de memórias que podem suprir algumas desvantagens apresentadas pelas memórias tradicionais, DRAM e SRAM. Nesse sentido, surgem alguns candidatos promissores em tecnologia emergente de memória, tais como: *Phase-Change RAM* (PCRAM), *Spin Torque Transfer Magnetic RAM* (STT-MRAM), *Resistive RAM* (ReRAM), e *Memristor*. Estas tecnologias vem se destacando e ganhando atenções substanciais de pesquisadores e da indústria, explorando seu uso tanto como memórias externas quanto memórias internas (IRDS, 2017) (XU et al., 2015) (BI et al., 2017). Nesta seção, será apresentado e discutido os fundamentos e funcionamento dessas promissoras tecnologias de memória.

### 3.3.2.1 Phase-Change Random-Access Memory – PCRAM

A *Phase-Change Random-Access Memory* (PCRAM) apresenta um mecanismo de armazenamento não volátil com grande escalabilidade. A PCRAM é baseada em efeitos analógicos e térmicos e não requer controle sobre elétrons discretos (WONG et al., 2010). À medida que as escalas de tecnologia e as áreas de contato de aquecimento diminuem, a corrente de programação é escalonada linearmente. Esse escalonamento da PCRAM já foi demonstrado em um protótipo de dispositivo de  $20nm$  e é projetado para ser dimensionado para  $9nm$  (WONG et al., 2010).

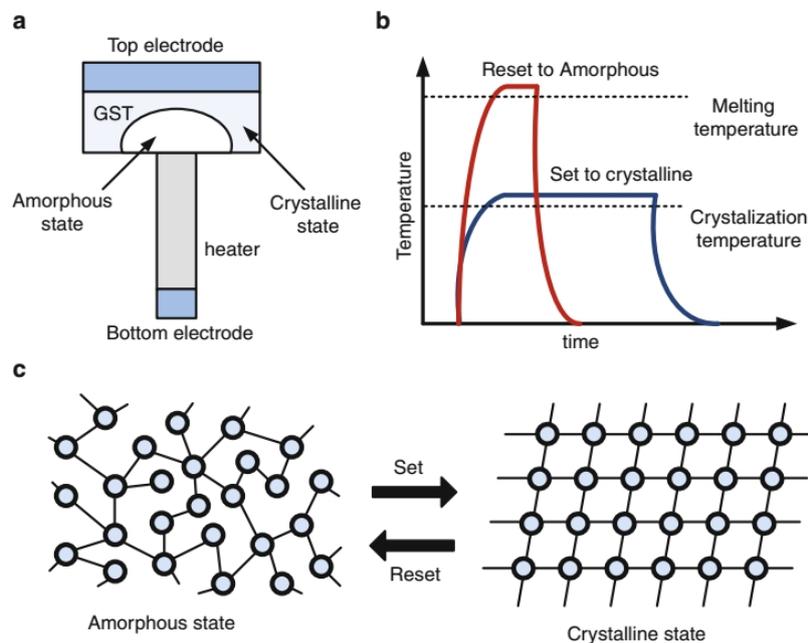
No processo de escrita de dados, um transistor de acesso injeta corrente no material calcogeneto, induzindo termicamente a mudança de fase, a qual é detectada durante as leituras. Entre os materiais calcogenetos mais utilizados destaca-se o  $Ge_2Sb_2Te_5$ , o qual é conhecido como GST. O GST apresenta rápida transição e estabilidade em uma ampla faixa de temperaturas (HONG; AUCIELLO; WOUTERS, 2014).

Os materiais calcogenetos têm dois estados estáveis diferentes: cristalinos e amorfo, como mostrado na Figura 3.8c. A resistência do calcogeneto no estado amorfo é maior, em várias ordens de magnitude, do que no estado cristalino. Além disso, a estrutura cristalina do calcogeneto pode ser alternada termicamente entre os dois estados. Explorando esses comportamentos únicos do calcogeneto, a utilização da PCRAM tornou-se possível nos últimos anos (HONG; AUCIELLO; WOUTERS, 2014).

Uma típica célula PCRAM em forma de cogumelo é ilustrada na Figura 3.8a. Um ponto de aquecimento é colocado logo abaixo da camada de GST com uma pequena área de contato, o qual é cercado por materiais termicamente isolantes. Esta estrutura sob certa corrente irá produzir calor a uma taxa maior do que pode ser dissipada, o que leva ao aumento de temperatura do GST, aplicando pulsos de corrente com formas diferentes ao ponto de aquecimento, o processo de mudança de fase pode ser ativado bidirecionalmente. Conforme é apresentado na Figura 3.8b, quando um estreito pulso com grande corrente é aplicado, a temperatura da GST aumentará acima de seu ponto de fusão, levando à ativação do processo da fase amorfa. Por outro lado, quando um longo pulso com pequena corrente é aplicado, a temperatura da GST aumentará, mas permanecerá abaixo do seu ponto de fusão, resultando na ativação do processo de cristalização. O processo de leitura da informação nas células da PCRAM envolve a detecção da resistência.

A memória PCRAM é uma alternativa à DRAM, a PCRAM fornece funcionalidades que aumentam a densidade e a capacidade da memória principal. No entanto, para por em prática sua utilização, deve-se considerar as desvantagens da PCRAM em relação à

Figura 3.8: (a) Diagrama de uma célula PCRAM. (b) Perfil de temperatura do material calcogeneto nas operações de SET e RESET. (c) Mudanças de fase entre os estados amorfo e cristalino.



Fonte: Yu e Wang (2014).

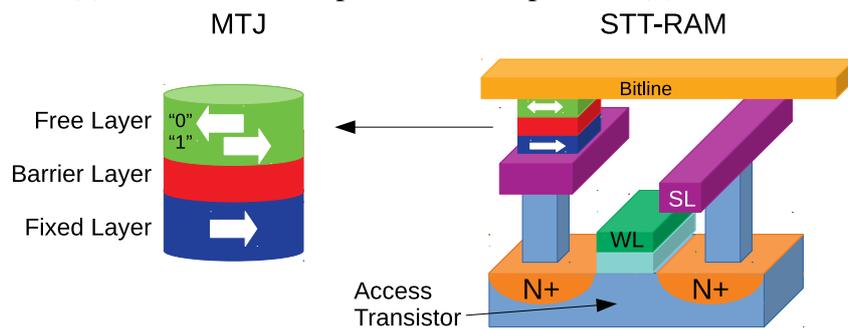
DRAM. A latência de acesso, apesar de dezenas de nanossegundos, são várias vezes mais lentas que as da DRAM. Atualmente, em tecnologias mais novas, as escritas na PCRAM exigem uma pulso de corrente com intensa energia (WONG et al., 2010). Além disso, as escritas induzem expansão e contração térmicas no elemento de armazenamento, degradando os contatos e limitando a resistência a centenas de milhões de escritas ( $10^9$  segundo Jokar, Arjomand e Sarbazi-Azad (2016)) por célula nos processos atuais.

### 3.3.2.2 Spin-Transfer Torque Magnetic Random Access Memory – STT-MRAM

A STT-MRAM (*Spin-Transfer Torque Random Access Memory*) é um tipo de memória magnética (*Magnetoresistive RAM* – MRAM) e essas memórias, assim como a PCRAM, não utilizam carga elétrica para armazenamento da informação. Nas MRAM são utilizados junções de tunel magnéticas (*Magnetic Tunnel Junction* – MTJ), os quais são usados como elementos de armazenamento de dados (YU; WANG, 2014). Um MTJ é composto por três camadas, como mostrado na Figura 3.9: uma placa magnética livre (*free layer*) na parte superior, uma placa magnética fixa (*fixed layer*) na parte inferior e uma placa isolante (*barrier layer*) na parte intermediária, a qual pode ser um óxido como óxido de magnésio (MgO) (BI et al., 2017).

No MTJ a camada fixa é fortemente magnetizada; assim sua magnetização não

Figura 3.9: (a) MTJ nos estados paralelo e anti-paralelo (b) Célula STT-MRAM.



Fonte: MRAM-Info (2018).

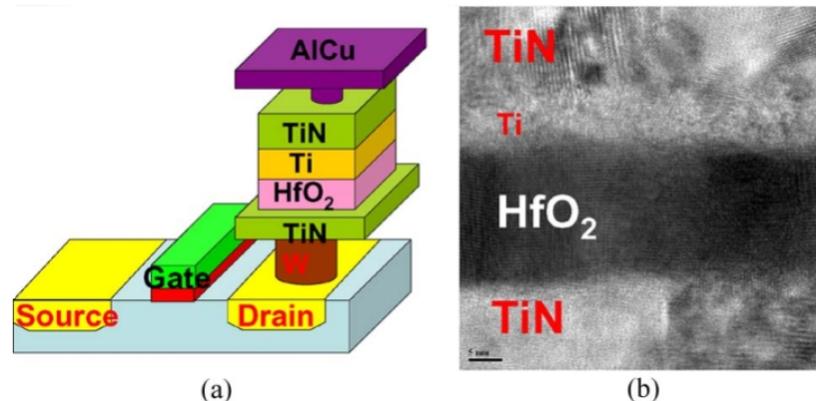
pode ser alterada. O bit armazenado é denotado pela orientação de magnetização na camada livre, a qual é construída com uma fina película de material ferromagnético e sua magnetização pode ser alterada através da aplicação de campo magnético externo, cuja força excede o campo original. Se a orientação da magnetização de ambas as placas estão no mesmo sentido tem-se o estado paralelo, o qual apresenta baixa resistência. Caso a orientação da camada livre esteja em sentido contrário à camada de referência tem-se o estado anti-paralelo, apresentando alta resistência. A baixa resistência é representada pelo valor lógico “1”, enquanto que a alta resistência representa o valor lógico “0” (YU; WANG, 2014).

Esse efeito ficou conhecido como efeito de transferência de spin (HOSOMI et al., 2005) e baseia-se na ideia de que uma corrente elétrica spin polarizada, ao atravessar uma fina camada ferromagnética com orientação de magnetização diferente da orientação de sua polarização, produzirá um efeito de torque clássico que agirá sobre a magnetização dessa camada, resultando em mudança de direção da mesma. Essa mudança pode levar a uma completa reversão da orientação da magnetização e também a estados intermediários. A Figura 3.9 apresenta o esquemático mais popular de uma célula STT-MRAM, a qual contém um transistor de acesso nMOS e um MTJ.

Semelhante à célula DRAM, a célula STT-MRAM também possui um transistor de acesso que conecta o dispositivo de armazenamento e a *bitline*. No entanto, diferente da DRAM, a outra extremidade do dispositivo de armazenamento não está conectada ao *GND*; em vez disso, ele está conectado à *source-line* (SL), conforme pode ser observado na Figura 3.9.

Como visto na seção anterior, na DRAM, os dados são armazenados como tensão através de um capacitor, o qual é detectado usando o circuito de *sense amplifier*. Na STT-MRAM, é a resistência do MTJ que muda com base nos dados armazenados. Portanto, diferentes mecanismos de sensoriamento e escrita devem ser empregados. Para ler os

Figura 3.10: (a) Estrutura 1T1R de uma típica célula ReRAM (b) Imagem microscópica de alta resolução da célula ReRAM.



Fonte: Wong et al. (2012).

dados armazenados em uma célula, uma pequena tensão é aplicada entre a *bitline* e a *source-line*, e a quantidade de fluxo de corrente é detectada. Da mesma forma, escrever em uma célula STT-MRAM não é uma operação em modo de tensão, mas uma operação usando corrente. Especificamente, para escrever dados em um MTJ, uma grande corrente deve ser colocada através do MTJ para alterar a orientação magnética da camada livre (BI et al., 2017).

### 3.3.2.3 Resistive Random Access Memory – ReRAM

As tecnologias de NV-RAM apresentadas anteriormente, STT-MRAM e PCRAM, são baseadas em efeitos de comutação resistiva induzida eletricamente. Entretanto, as memórias RAM resistivas (ReRAM) envolvem efeitos eletroquímicos e termoquímicos na mudança de resistência do sistema metal/óxido/metal (XIE, 2014). A Figura 3.10 apresenta uma estrutura típica de uma célula ReRAM. Uma célula ReRAM consiste de uma camada de óxido de metal (por exemplo, *Ti*, *Ta* e *Hf*) entre dois eletrodos de metal (por exemplo, *Pt* e *TiN*). Para tornar o processo de fabricação compatível com CMOS uma estrutura *TiN* é usada como camada de cobertura metálica. Neste caso evita-se o uso de *Pt*, pois a *Pt* é de difícil gravação. A camada de cobertura de metal é inserida para esvaziar os átomos de *O* da camada de *HfO<sub>x</sub>* e agir como um reservatório de oxigênio (WONG et al., 2012).

O comportamento eletrônico das interfaces metal/óxido depende da concentração de lacunas de oxigênio na camada de óxido de metal. Normalmente, a interface metal/óxido apresenta comportamento ôhmico no caso de dopagem muito alta e retificação no caso de baixa dopagem (XIE, 2014). Na Figura 3.10, a região *HfO<sub>2</sub>* é semi-isolante

indicando menor concentração de lacunas de oxigênio, enquanto que o  $Ti$  é condutivo indicando maior concentração de lacunas de oxigênio.

As lacunas de oxigênio no óxido de metal são dopantes do tipo n, onde o campo elétrico pode causar mudanças nos perfis de dopagem. Assim, a aplicação da corrente elétrica pode modular a curva IV da célula ReRAM e mudar a célula de um estado para outro. Normalmente, para uma ReRAM bipolar, a célula só pode ser ligada (ON, operação SET) aplicando uma polarização negativa e desligada (OFF, operação RESET) apenas aplicando a polarização oposta. Vários protótipos de ReRAM foram desenvolvidos e mostram propriedades promissoras na velocidade de comutação e baixo consumo de energia (XIE, 2014).

A operação de leitura em memórias ReRAM é bastante semelhante as demais NV-RAM. Como a célula de memória tem resistência diferente nos estados ON e OFF, a operação de leitura pode ser realizada aplicando uma pequena tensão na *bitline* e detectando a corrente que passa pela célula de memória ou injetando uma pequena corrente na *bitline* e sentindo a tensão através da célula de memória (XIE, 2014). A resistência no estado ON é milhares de vezes menor que a resistência ao estado OFF. Diferentemente da SRAM que gera sinais de leitura complementares de cada célula, as NV-RAM geralmente tem um grupo de células *dummy* para gerar a corrente de referência ou tensão de referência. A corrente gerada (ou tensão) da célula a ser lida é então comparada com a corrente de referência (ou tensão) usando *sense amplifiers* (XIE, 2014).

A célula ReRAM é capaz de duas operações de substituição: SET (valor lógico “1”) e RESET (valor lógico “0”). A operação de SET altera a célula ReRAM de um estado de alta resistência para um estado de baixa resistência, i.e. OFF para ON, aplicando uma tensão SET à *bitline* e 0V à *source-line*. A operação RESET altera a célula ReRAM de baixa resistência para alta resistência, i.e. ON para OFF, aplicando uma tensão RESET à *source-line* e 0V à *bitline* (CHANG et al., 2013).

A memória ReRAM apresenta grande potencial para ser utilizada em sistemas embarcados, uma vez que a ReRAM dispõe de alta densidade e velocidade de acesso (WONG et al., 2012). Além disso, apresenta alta compatibilidade com tecnologia CMOS, podendo ser fabricada no mesmo *die* dos processadores e memórias SRAM (JOKAR; ARJOMAND; SARBAZI-AZAD, 2016).

### 3.3.2.4 Ferroelectric Random Access memory – FeRAM

No final dos anos 90, o óxido de háfnio ( $HfO_2$ ) tornou-se um dos materiais de alto- $k$  mais proeminentes, onde  $k$  representa a constante dielétrica ou permissividade, para substituir o bem estabelecido dióxido de silício em transistores semicondutores de efeito de campo de óxido metálico. Além disso, o óxido de háfnio tem permissividade e *bandgap* razoavelmente alto e é estável em silício (MIKOLAJICK et al., 2018).

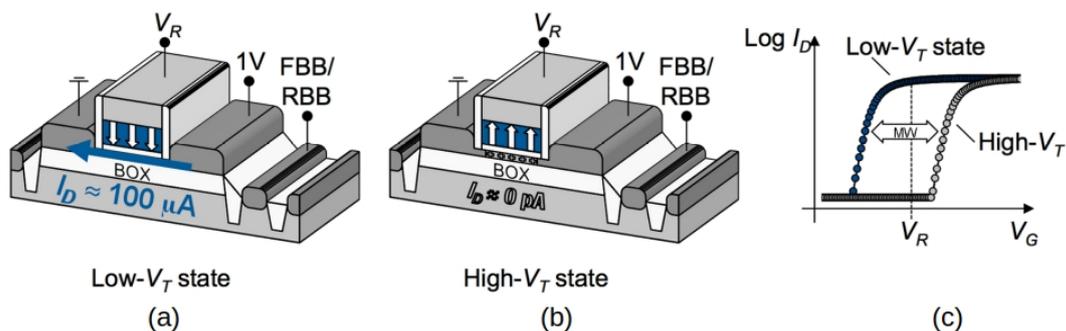
Com o óxido de háfnio ferroelétrico, todas as vantagens da integração 3D podem ser atingidas (MIKOLAJICK et al., 2018). No entanto, ao usar o material em uma configuração de RAM 1T-1C (1 transistor e 1 capacitor), a restrição com o alto campo do óxido de háfnio ferroelétrico é um fator limitante. Além disso, a operação do dispositivo em baixas tensões é um desafio, e os ciclos de campo usam campos próximos ao valor de *breakdown*, o que limita o *endurance* do dispositivo. Ainda, devido à leitura destrutiva realizada no dispositivo, os ciclos de leitura também contribuem para uma redução ainda maior no *endurance*, que atualmente está em  $10^4$  ciclos (MÜLLER et al., 2013) (ALI et al., 2018). Porém, ao integrar o óxido de háfnio ferroelétrico no *gate* do transistor, resultando em um FeFET, ambos os aspectos conferem a esse material uma vantagem importante (MIKOLAJICK et al., 2018).

O FeFET apresenta características não voláteis devido ao fato de que dois estados estáveis de polarização *gate* ferroelétrico modificam a tensão limite mesmo quando a tensão de alimentação é removida. Assim, os estados binários são codificados na tensão limite do transistor, conforme pode ser observado na Figura 3.11. Quando a polarização ferroelétrica está apontando para baixo (3.11 (a)), os elétrons invertem a região do canal permanentemente trazendo o FeFET para o estado “ON”. Se a polarização estiver apontando para cima (3.11 (b)), uma acumulação permanente é criada e o FeFET está no estado “OFF”. Desta forma, os estados binários são codificados na tensão limite do transistor (3.11 (c)).

A mudança da polarização do FeFET pela aplicação de um pulso de tensão positivo ou negativo induz uma mudança de tensão limiar estável, mas reversível, dependendo da orientação dos dipolos (*low-VT* e *high-VT*, respectivamente). Esses dois estados distintos do dispositivo separados pela chamada janela de memória (*MW*), podem ser detectados através de uma leitura da corrente de dreno do transistor (DÜNKEL et al., 2017).

A Figura 3.12 (a) apresenta o diagrama de circuitos de memória de acesso aleatório ferroelétrico convencional (FeRAM), usando (b) óxido de háfnio ferroelétrico para atingir a integração 3D, onde *FE*, indica ferroelétrico; *BL*, *bitline*; *WL*, *wordline*; *PL*,

Figura 3.11: Ilustração da funcionalidade de um FeFET tipo n.

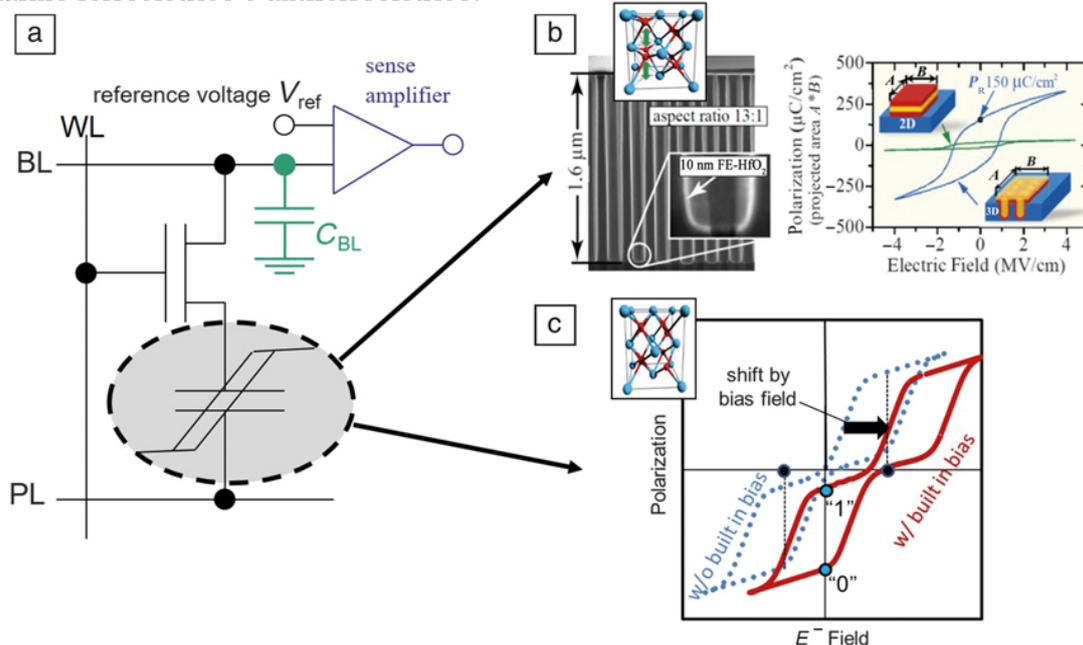


Fonte: Dünkler et al. (2017).

plateline;  $C_{BL}$ , capacidade de bitline; e  $P_R$ , polarização remanescente. Figura 3.12 (b) aparece a imagem microscópica da seção transversal do óxido de háfnio dopado com alumínio a  $10nm$  em uma estrutura 3D; e a histerese para a estrutura 3D (em azul) e uma estrutura planar com a mesma área projetada  $A \times B$  (em verde). Na Figura 3.12 (c) é apresentado a ideia de memória antiferroelétrica para superar as limitações de ciclo de *endurance* e operação em baixa tensão. O laço de histerese pontilhado mostra a histerese antiferroelétrica original. O laço de histerese sólida mostra a histerese depois de ter sido deslocado por uma tensão interna usando eletrodos com diferentes funções de trabalho. Os pontos marcados com “0” e “1” no laço de histerese sólida indicam os dois estados usados para armazenamento de informações. A estrutura do óxido de háfnio tetragonal apresentada indica a fase cristalina requerida, onde esferas vermelhas representam ânions de oxigênio, e esferas azuis representam cátions de háfnio; setas verdes indicam os possíveis movimentos dos íons de oxigênio entre as duas possíveis configurações estáveis.

O progresso do desenvolvimento de FeFETs aumentou significativamente com a adoção do óxido de háfnio ferroelétrico. No entanto, alcançar um ciclo de *endurance* além do nível de memórias não voláteis baseadas em carga convencional permanece um desafio. Diferentes estratégias têm sido propostas para superar essas limitações, conforme pode ser visto no artigo de Ali et al. (2018). O fato de que FeFETs otimizados de forma diferente também são opção para fazer dispositivos *subthreshold*, estabilizando a região de capacitância negativa, inspirou ainda mais a pesquisa e desenvolvimento desses dispositivos (MIKOLAJICK et al., 2018).

Figura 3.12: Ilustração de memórias não-voláteis de acesso aleatório usando óxido de háfnio ferroelétrico e antiferroelétrico.



Fonte: Mikolajick et al. (2018).

### 3.4 Desafios relacionados às tecnologias de memórias

Nas seções anteriores foi apresentado o funcionamento das principais tecnologias de memória, tais como: DRAM, SRAM, PCRAM, STT-MRAM, ReRAM e FeFRAM. Todas essas tecnologias de memória estão bem estabelecidas e tem sido foco de pesquisas por vários grupos de trabalho ao redor do mundo. E mesmo essas tecnologias apresentam certas limitações nos mais diversos aspectos.

Nas memórias DRAM as células de memória são, em essência, formadas por capacitores. O armazenamento ou não da carga define o estado lógico. Os capacitores responsáveis pelo armazenamento da informação tendem a descarregar com o tempo. Assim, é necessário renovar de tempos em tempos a informação armazenada na memória, essa atividade é chamada de *refresh*. E esse *refresh* apresenta um consumo de energia representativo relacionado as células de memória. Além disso, os ciclos de renovação, i.e. *refresh cycles*, competem com os ciclos de acesso à memória. No trabalho de Mutlu e Subramanian (2015) é apresentado um estudo sobre as taxas de *refresh* em memórias DRAM, esse trabalho mostra que em uma memória DRAM de 4Gb são gastos 8% do tempo e 15% de energia apenas com o *refresh*. Além disso, em uma projeção para uma futura memória de 64Gb esses valores chegariam a mais de 45%.

A SRAM é a tecnologia de memória mais rápida disponível atualmente, mas ela

apresenta um consumo de *leakage* significativo para reter os dados armazenados, o que piora quando tecnologias submicrônicas são empregadas (YU; WANG, 2014). Além disso, sua capacidade é limitada a alguns megabytes devido ao alto custo gerado por uma estrutura de 6 transistores, os quais utilizam uma grande área para serem implementadas.

Quanto as memórias não voláteis, quatro problemas são bastante discutidos na literatura: a quantidade de sobrescrita que podem ser feitas nas células de memória, conhecido como *write endurance*, o tempo de retenção dos dados nas células, o consumo de energia relacionado a escrita e o tempo de acesso (YU; WANG, 2014). Entre todas as tecnologias NV-RAM citadas anteriormente, apenas a STT-MRAM não sofre com o problema de *write endurance*. PCRAM, ReRAM e FeFRAM têm *write endurance* limitada. A sobrescrita da PCRAM está atualmente no intervalo entre  $10^5$  e  $10^9$ . A sobrescrita da ReRAM atualmente apresenta quantidades em torno de  $10^{10}$  e na FeFRAM essa taxa está em  $10^5$ . Entretanto, esse cenário deve mudar em seguida, uma projeção sobre as NV-RAMs para 2024 estima que essas memórias devem apresentar *write endurance* na ordem de  $10^{15}$  ou mais (YU; WANG, 2014).

O tempo de retenção diz respeito ao tempo em que os dados ficam retidos nas células NV-RAM quando estas estão sem energia. Normalmente, as tecnologias NV-RAM exigem tempo de retenção maior que 10 anos (YU; WANG, 2014). No entanto, em alguns casos, um tempo de retenção tão alto não é necessário, principalmente quando se trata de memórias internas onde os dados são frequentemente lidos e escritos nas células. Desta forma, dependendo do lugar em que a NV-RAM será utilizada o tempo de retenção pode ser relaxado, fazendo com que o consumo de energia da memória seja reduzido.

Quando as NV-RAMs são empregadas em memórias internas como Cache e SPM, vários problemas de projeto precisam ser resolvidos. O mais importante é a sobrecarga de desempenho e consumo de energia nas operações de escrita. A NV-RAM tem um mecanismo mais estável para manter dados, quando comparado a uma memória volátil como SRAM. Assim, ela leva mais tempo e consome mais energia para sobrescrever os dados existentes (YU; WANG, 2014). Essa é uma característica intrínseca das NV-RAMs. Desta forma, a simples substituição das Caches SRAM por Caches NV-RAMs pode não ser suficiente para gerar benefícios para o sistema de memória. Portanto, é importante estudar técnicas e abordagens que reduzam o impacto desses problemas, aproveitando o melhor dessas memórias. Nesse sentido, no próximo capítulo serão discutidos trabalhos com soluções que podem ser empregadas para contornar esses problemas.

## 4 TRABALHOS RELACIONADOS

Muitas das aplicações que utilizam codificadores de vídeo são sistemas embarcados, onde a minimização na energia consumida é uma restrição extremamente severa. Como visto na seção 2.4 a ME é um dos módulos do codificador de vídeo que mais faz acessos às memórias interna e externa durante a codificação, sendo essa comunicação um dos principais gargalos do sistema em termos de consumo de energia. Assim, o projeto de um codificador de vídeo deve buscar a diminuição dos acessos à memória, melhorando a eficiência na comunicação com a memória.

Alguns trabalhos que visam minimizar os problemas ocasionados pela comunicação entre a memória externa e o núcleo de processamento são sugeridos na literatura, e serão discutidas com mais detalhes nas próximas seções, estes trabalhos se baseiam em cinco técnicas principais, que são: 1) estratégias de reutilização de dados utilizando memórias internas; 2) compressão de quadros de referência, antes de serem armazenadas na memória externa; 3) controle dinâmico do intervalo de busca para o processo de *block matching* da ME. 4) utilização de NV-RAMs na hierarquia de memória; 5) computação aproximada, sendo que esta pode ser utilizada tanto na hierarquia de memória através de reduções no tamanho da área de busca, quanto em algoritmos de compressão com perdas, utilizando quantização, subamostragem, *bit dropping*, entre outras.

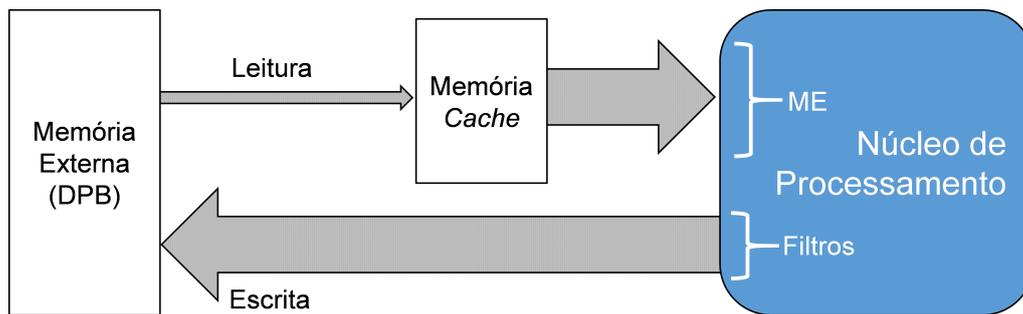
Desta forma, nesse capítulo serão apresentadas as principais técnicas encontradas na literatura para a redução do consumo de energia relacionada às memórias em codificadores de vídeo. Além disso, serão apresentados os trabalhos estado da arte que empregam essas estratégias para reduzir o consumo de energia relacionado ao sistema de memória dos codificadores de vídeo.

### 4.1 Reuso de dados

A reutilização de dados através de uma memória Cache interna integrada ao chip, trabalha de modo a armazenar os dados que potencialmente serão mais acessados pela unidade de processamento. Entretanto, além do emprego de uma memória local, o projetista deve, também, se preocupar em garantir a sua eficiência.

Para tanto, técnicas como a antecipação da busca de dados e previsão de acessos são amplamente utilizadas. A Figura 4.1 exemplifica a utilização da técnica de reuso de dados na comunicação entre a memória externa e as unidades de processamento. Na

Figura 4.1: Comunicação utilizando a técnica de reuso de dados.



Fonte: Próprio autor.

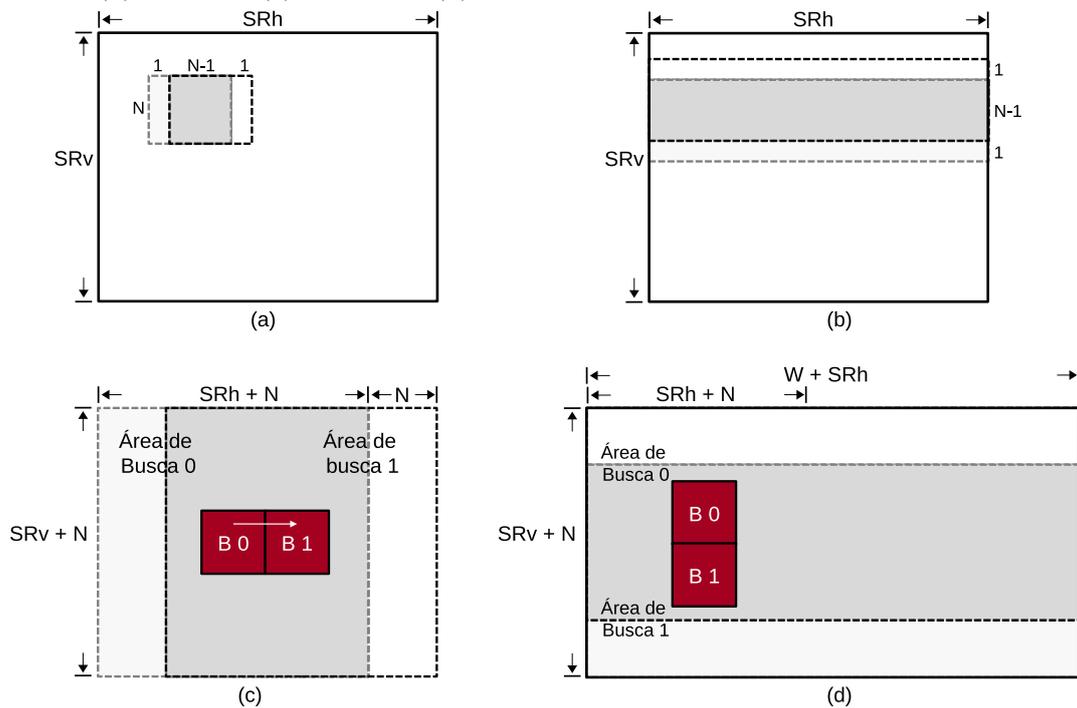
sequência serão discutidos alguns trabalhos que exploram o reuso de dados na codificação de vídeo.

O trabalho de Tuan, Chang e Jen (2002) apresenta várias estratégias de reuso de dados considerando, basicamente, dois diferentes níveis de redundâncias de informações: (1) regiões sobrepostas entre blocos candidatos vizinhos dentro de uma mesma área de busca (AB) (Level A e B) e (2) regiões sobrepostas entre as áreas de busca de blocos atuais vizinhos (Level C e D). A Figura 4.2 apresenta as quatro estratégias, onde as regiões destacadas em cinza escuro representam os dados que são reusados em cada estratégia. O intervalo de busca (*search range*,  $SR$ )  $(-X, +Y]$  apresentado na Figura 4.2, indica que a pesquisa suporta um deslocamento de  $X$  pixels para a esquerda e  $Y$  pixels para a direita do bloco original na direção horizontal, e  $X$  pixels para cima e  $Y$  pixels para baixo na direção vertical do bloco original. Por exemplo: uma  $SR(-96, +96]$  é representado como  $SRv = SRh = 96 + 96 = 192$ . Neste caso a dimensão real de uma área de busca para um bloco  $N \times N$  seria  $(SRh + N) \times (SRv + N)$ .

O reuso de dados Level A, como mostrado na Figura 4.2(a), aplica o reuso de dados entre dois blocos candidatos vizinhos. Neste caso, uma região de  $N \times (N - 1)$  é reusada. No Level B Figura 4.2(b), o reuso de dados é explorado considerando uma faixa de blocos de candidatos vizinhos horizontalmente dentro da área de busca. Nesta estratégia, uma região de  $(N - 1) \times (SRh + N - 1)$  é reusada. Em ambos Level A e Level B, o reuso é explorado dentro de uma mesma área de busca.

Nos esquemas Level C e Level D é explorado um nível mais alto de reuso: as áreas comuns entre áreas de buscas de blocos atuais vizinhos. O Level C reaproveita as informações entre áreas de busca entre dois blocos atuais vizinhos. Isto pode ser observado na Figura 4.2(c). O número de amostras reusadas com o Level C é igual a  $(SRv + N - 1) \times (SRh - 1)$ . A estratégia Level D aplica a mesma ideia, porém, con-

Figura 4.2: Estratégias de reuso de dados propostas em Tuan, Chang e Jen (2002): (a) Level A, (b) Level B, (c) Level C e (d) Level D.



Fonte: Tuan, Chang e Jen (2002).

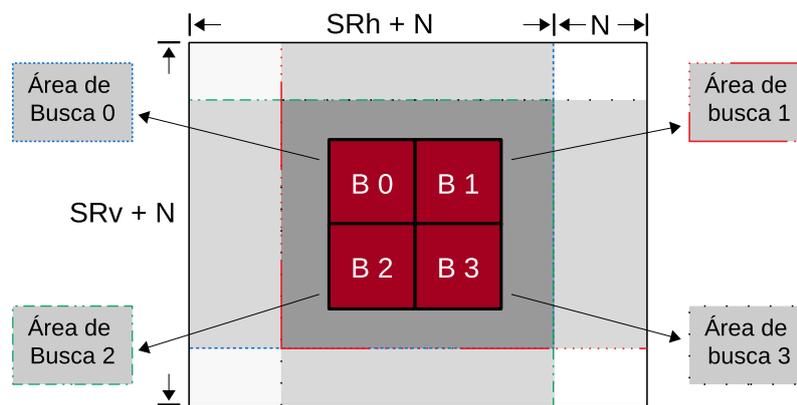
considerando as áreas de pesquisa de todos os blocos vizinhos dentro de todo o quadro (a dimensão  $W$  na Figura 4.2(d) representa a largura do quadro). Neste caso, o número de amostras reusadas é  $(SRv - 1) \times (W + SRh - 1)$ . O trabalho de Li, Zheng e Zhang (2007) apresenta uma solução arquitetural utilizando reuso de dados com Level D.

A estratégia apresentada no trabalho de Chen et al. (2006) é uma expansão do reuso Level C. O Level C+ considera o reuso de regiões sobrepostas das áreas de buscas entre blocos atuais vizinhos nas duas direções: horizontal e vertical.

O Level C+ apresenta melhores resultados de eficiência quando comparado às estratégias Level A-D. A medida de eficiência utilizada pelo autor considera o compromisso entre a largura de banda com a memória principal e o tamanho da memória interna ao chip para guardar localmente os dados reusados. Como a abordagem Level C+ exige um processamento fora da ordem usual, o trabalho também propõe algumas ordens de processamento de blocos atuais de modo a não prejudicar o desempenho dos módulos posteriores a ME. A Figura 4.3 apresenta o reuso de dados obtido coma utilização do Level C+, dados quatro blocos atuais e suas respectivas áreas de busca.

No Level C+, o número de amostras reusadas é  $(SRv + n \times N - 1) \times (SRh - 1)$ , onde  $n$  representa a quantidade de blocos na mesma coluna. Esta solução é capaz de atingir uma taxa de redução de 46% na largura de banda de memória quando comparado

Figura 4.3: Estratégia de reuso de dados com Level C+.



Fonte: Chen et al. (2006).

a estratégia Level C, e uma redução de 82% quando comparado ao método tradicional, sem utilização de reuso de dados, mas para que isso aconteça, há um aumento na memória interna de 18%. Esta redução ocorre na comunicação com a memória externa, porém, com a utilização de uma memória Cache uma nova comunicação é inserida e ela não pode ser descartada.

O trabalho apresentado em Grellert et al. (2011) traz uma estratégia de reuso de dados de vários níveis, e é baseado na sobreposição das áreas dos blocos vizinhos durante o processo de buscas na memória. Este trabalho propõe uma solução com dois níveis de hierarquia de memória: uma Cache local e um *buffer* local. Esta estratégia de reuso de dados com dois níveis faz com que esta solução atinja uma redução de largura de banda de memória de até 90% para acessos de leitura, quando utilizado na codificação de vídeos HD 1080p. No entanto, há uma sobrecarga em memória interna de 185KB para a implementação da Cache e do *buffer* local. Esta solução foi projetada em VHDL, e a sua arquitetura apresentou uma alta taxa de transferência, sendo capaz de processar vídeos em HD 720p a 56 QPS e vídeos HD 1080p a 25 QPS.

No trabalho de Song, Ju e Jia (2015) é proposta uma arquitetura de memória híbrida composta por memórias Caches e *scratchpads* (SPM), ambas SRAM. Nesse trabalho um *power gating* adaptativo é proposto para desligar setores da SPM para reduzir ainda mais o consumo de energia. Os resultados experimentais mostram que a abordagem proposta suporta processamento HEVC paralelo de alta velocidade com tamanho de memória interna muito pequeno, atingindo até 76% de redução no consumo de energia no chip e 33% de economia de energia no subsistema de memória geral.

No trabalho proposto em Xu et al. (2018) uma abordagem de reuso de dados interquadros é apresentada. O novo método inclui dois níveis, inter-D e inter-E. No Level

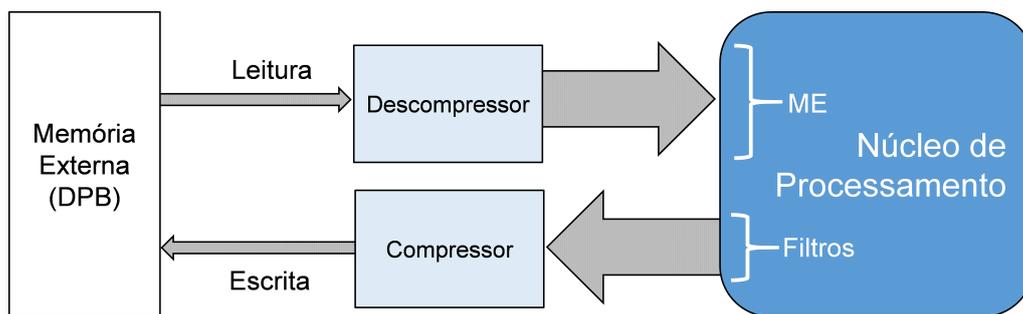
inter-E dois *buffers* locais são utilizados para armazenar dois quadros do vídeo, o quadro atual e o quadro de referência. Após a codificação do quadro atual é gerado o quadro reconstruído, o qual será utilizado como quadro de referência para a codificação do próximo quadro. Com o inter-E o quadro reconstruído não é enviado para a memória externa, ele já fica armazenado no *buffer* de memória interna como quadro de referência. Dessa forma, o quadro é carregado da memória externa apenas uma vez, quando ele está sendo processado como quadro atual. Com esta abordagem uma grande área de memória interna se faz necessário, uma vez que dois quadros inteiros devem ser armazenados.

Com intuito de reduzir essa área, os mesmos autores propuseram o Level inter-D, o qual utiliza múltiplas áreas de buscas ao invés de todo quadro. As duas abordagens utilizam mais memória interna que os métodos apresentados anteriormente, porém, foram empregadas com o algoritmo *Full Search*, apresentando maior redução na comunicação com a memória externa. Desta forma, essas propostas apresentam menor consumo de energia quando comparada as demais estratégias de reuso. Quando comparada ao Level C, o inter-D apresenta uma redução de 67,8% em consumo de energia e 68% em largura de banda de memória. Entretanto, caso fossem utilizadas com algoritmos rápidos muitas dessas informações armazenadas em memórias internas seriam pouco acessadas, levando a um gasto desnecessário em área e energia.

Em Martins et al. (2018) é proposto uma hierarquia de memória utilizando memórias Cache adaptativas ao contexto do vídeo codificado. Assim, dependendo da resolução, textura e demais configurações um tipo de Cache pode ser utilizado e a escolha é feita em tempo de codificação. Nessa abordagem são propostas 8 memórias SRAM de  $2KB$  cada uma, as quais podem ser rearranjadas para até 6 configurações diferentes. Quando comparada ao reuso Level C e a uma Cache estática de  $16KB$  com associatividade *8-way*, a Cache proposta atinge uma redução de energia de 79,3% e 24,8% em média, respectivamente. Entretanto, os autores não apresentaram a arquitetura de hardware para o gerenciamento desse sistema de memória. Desta forma, há um gasto energético importante que não está contabilizado.

Os trabalhos sobre reuso de dados geralmente atingem uma alta taxa de redução de largura de banda de memória, esta redução ocorre entre o núcleo de processamento e a memória externa. Entretanto, cabe salientar que esta abordagem reduz apenas os acessos de leitura da memória e utiliza uma grande quantidade de bits de memória para implementar as memórias caches. Além disso, alguns autores não contabilizam os custos relacionados ao gerenciamento do sistema de memória ou alterações que devem ser feitas

Figura 4.4: Comunicação utilizando a compressão de quadros de referência.



Fonte: Próprio autor.

no algoritmo de busca durante a codificação do vídeo. Nas próximas seções serão apresentadas estratégias que utilizam compressão de quadros de referência, reuso de dados com memórias NV-RAM e computação aproximada.

## 4.2 Compressão de quadros de referência

As soluções que utilizam a compressão de quadros de referência requerem duas etapas a mais no processo de codificação, que são: (1) comprimir o quadro de referência antes de ele ser enviado a memória; e (2) descomprimir o quadro que está na memória externa quando ele for utilizado na codificação. Esta abordagem introduz ganhos de diferentes maneiras, diminuindo o número de acessos a memória, diminuindo o tamanho da memória externa ou reduzindo a largura de banda necessária na comunicação, gerando uma redução no consumo de energia do codificador. O uso desta técnica pode ser observado na Figura 4.4, onde é ilustrada a comunicação entre memória externa e as unidades de processamento através da utilização de um compressor/descompressor de quadros de referência.

Conforme pode ser observado na Figura 4.4, as soluções com compressão dos quadros de referência diminuem os acessos de leitura e escrita na memória externa. Enquanto que as soluções baseadas no reuso de dados com memória Cache diminuem apenas os acessos de leitura a memória externa. Assim, soluções baseadas em compressão podem obter maior redução no consumo de energia do codificador, já que esta solução atua nos dois sentidos da comunicação.

Os trabalhos encontrados na área apresentam dois tipos de compressão: com perdas e sem perdas de qualidade. As soluções com perdas de qualidade geralmente apresentam algoritmos mais simples quando comparado a soluções sem perdas, algoritmos

com quantização, transformadas e codificação diferencial são amplamente utilizados. Por outro lado, as soluções sem perdas de qualidade utilizam algoritmos mais complexos, porém garantem a qualidade do vídeo codificado. As soluções sem perdas de qualidade geralmente utilizam algoritmos com codificação diferencial entre amostras, transformadas, codificação de entropia baseada em algoritmos de Huffman, Golomb, Exp-golomb e codificação de dicionário. A taxa de compressão ( $CR$ ) obtida pelos trabalhos relacionados é calculado através da Equação 4.1. A seguir, esses trabalhos são apresentados em detalhes.

$$CR = \left( 1 - \frac{\text{dados comprimidos}}{\text{dados originais}} \right) \times 100 \quad (4.1)$$

#### 4.2.1 Compressão com perdas de dados

O trabalho de Gupte et al. (2011) utiliza um algoritmo de compressão de comprimento fixo baseado em blocos de quantização escalar (SAYOOD, 2005), chamada *Min-Max-Scalar-Quantization* (MMSQ). Foram desenvolvidos dois algoritmos, o MMSQ 6bpp e o MMSQ 5bpp, os quais utilizam *bit dropping* no processo. Entretanto, as perdas geradas são armazenadas, de modo que estas possam ser retornadas aos blocos correspondentes durante o processo da compensação de movimento. Com esta solução a taxa de redução de largura de banda fica em torno de 23% para vídeos HD 1080p, com uma perda de qualidade de 0,01 dB no quadro de referência codificado.

Esta solução é interessante, mas também gera perdas durante o processo, porque a ME irá utilizar amostras de referência do quadro quantizado, e portanto, o melhor vetor de movimento pode não ser encontrado, simplesmente porque as amostras do quadro de referência não são mais iguais às originais. Além disso, esta solução não é compatível com as arquiteturas que aplicam os processos de ME e MC em conjunto. As arquiteturas que implementam ME e MC combinados podem pular o acesso à memória externa realizada pela MC, armazenando o bloco residual no momento em que a ME encontra o melhor casamento para o bloco atual, reduzindo a largura de banda de memória no processo de codificação.

O trabalho de Ma e Segall (2011) apresenta um algoritmo de compressão híbrido. Primeiro, a imagem é decomposta em amostras de baixa resolução (LR) e amostras de alta resolução (HR). Em seguida, é aplicada uma codificação residual entre as amostras

HR e LR. Após a codificação, uma quantização é aplicada sobre o resíduo. Esta solução apresenta uma taxa de compressão de 31% e uma perda de qualidade de 0,15 dB, para cinco sequências de vídeo HD 1080p avaliadas.

No artigo de Fan, Shang e Zeng (2015) é proposto o algoritmo chamado de *mixed lossy and lossless reference frame recompression*. Esse algoritmo combina duas técnicas para atingir altas taxas de redução, que são o truncamento de pixels e a compressão dos dados. O truncamento de pixels são usados na estimação de movimento inteira (IME) e nas etapas de estimação de movimento de fracionária (FME) e compensação de movimento (MC) os dados são utilizados sem truncagem. Como o acesso aos dados na IME é muito maior do que na FME e no MC, ele reduz cerca de 37,5% de largura de banda no truncamento de 3 bits. Na compressão de dados empregada aos dados truncados em 3 bits a taxa de compressão chega a 84,5%. Para a compressão, dois métodos são propostos: uma predição intra bloco  $8 \times 8$  e um VLC otimizado. Nos experimentos, a largura de banda de memória externa é reduzida em 74,5%, utilizando reuso de dados Level C juntamente com o compressor. A degradação da qualidade de vídeo chega a 0,14db com aumento de 0,2% na taxa de bits em média.

No trabalho de Lian et al. (2018) é proposto um algoritmo de compressão com perdas que utiliza três etapas de codificação. A primeira é uma quantização adaptativa que depende dos valores das amostras do bloco; a segunda é uma codificação diferencial entre amostras vizinhas do bloco, o algoritmo trabalha com blocos  $16 \times 16$  e o subdivide em 64 blocos  $2 \times 2$ . A terceira etapa é uma codificação VLC Exp-Golomb aplicado sobre os resíduos. Nos resultados, a taxa de compressão do algoritmo é de 72,9%, em média, para o modo que atinge a melhor taxa, utilizando vídeos HD 1080p e 2160p. Porém, nesse modo há uma perda na eficiência de codificação de 1,27%, medidos em BD-rate. Os autores também propõem a arquitetura de hardware para o algoritmo, a qual utiliza 110k gates para o codec completo, além de uma organização de memória utilizando o reuso de dados Level D. Com esses resultados, o sistema atinge uma taxa de redução de energia relacionada a memória externa de 54,3%.

Na abordagem proposta por Willème et al. (2018) os autores utilizam o algoritmo de compressão JPEG XS (ISO/IEC, 2019) como algoritmo de compressão para os quadros de referência. Além disso, os autores propõem a utilização do compressor juntamente com um sistema de reuso de dados. Para identificar as soluções eles as nomearam como CFBC (*Level C Frame Buffer Compression*) e DFBC (*Level D Frame Buffer Compression*). O CFBC é utilizado juntamente com Level C e utiliza blocos de tamanho  $64 \times 64$ , tamanho

esse de uma CTU, para realizar a compressão. Enquanto que o DFBC utiliza um bloco com altura de uma CTU, 64 pixels, e largura da área de busca. Com essas abordagens é possível atingir altas taxas de compressão, as quais atingem 66,7% no CFBC e 83,3% no DFBC. Apesar das altas taxas de compressão as abordagens inserem perdas consideráveis em BD-rate, 20% no CFBC e 14% no DFBC, utilizando os QPs 22, 27, 32 e 37 para vídeos 2160p e 1080p. Neste ponto, os próprios autores identificam que as perdas de qualidade inseridas nos quadros de referência fazem com que a eficiência de codificação diminua. Além disso, nenhum tipo de arquitetura de hardware é proposta ou mesmo estimada.

Todas as soluções apresentadas nessa subseção apresentam degradação de qualidade no processo de codificação dos quadros de referência. Isto ocorre, pois os algoritmos desenvolvidos utilizam uma etapa de quantização, descartando, de forma irreversível, algumas informações. Este é um problema importante, uma vez que este erro pode ser cumulativo, e dependendo da configuração que o codificador de vídeo utilizar, um erro nas amostras de referência pode se propagar por diversos quadros codificados a partir destas amostras de referência, causando uma grande perda na qualidade final do vídeo codificado. Outro problema é que o codificador e o decodificador usariam referências diferentes, podendo ocasionar *drifting* no processo de decodificação do vídeo.

#### 4.2.2 Compressão sem perdas de dados

Em Guo, Zhou e Goto (2013) é proposto um algoritmo de compressão que utiliza uma predição no domínio espacial com múltiplos modos de direção. Esta abordagem é adotada com o intuito de se adaptar às várias características de uma imagem. Após a etapa de predição, um reagrupamento residual é realizado a fim de melhorar o desempenho da compressão, utilizando para isso uma codificação de comprimento semifixo (*Semi-Fixed Length* – SFL). Os resultados mostram que esta solução pode atingir uma taxa de compressão média de 57% para os vídeos das Condições Comuns de Testes (CCTs) (BOSSSEN, 2012). No trabalho de Zhou et al. (2014) os autores apresentam uma discussão sobre o acesso aleatório à memória externa, utilizando esse algoritmo. Além disso, nesse artigo novos resultados da implementação em hardware são apresentados. Em Guo, Zhou e Goto (2014) é apresentada uma extensão desses trabalhos, que inclui o algoritmo de compressão e introduz as arquiteturas com acesso aleatório à memória externa.

Em Lee, Chen e You (2014) é utilizado o algoritmo *Autocorrelation-Based Frame Lossless Recompression* (AFLR). Esse algoritmo realiza uma predição de 4 modos dis-

tintos nas amostras. Após a etapa de predição, é realizada uma codificação utilizando o algoritmo Golomb-Rice adaptativo. Esse trabalho também propõe uma memória DMMA (*Dual-Mode Memory Addressing*), para prover acesso aleatório aos dados codificados na memória externa. A taxa de compressão dessa solução atinge em média 51%. Esse resultado foi obtido utilizando 10 vídeos HD 1080p.

No trabalho de Chen, Lee e Chen (2014) apresenta um algoritmo de compressão multidirecional (Multi-Directional Lossless Recompression - MDLR). Esse algoritmo trabalha com blocos de tamanho  $16 \times 16$  e para cada bloco é executada uma predição diferencial multidirecional (*Multi-Directional Prediction* - MDP), a fim de obter todos os resíduos do bloco. Por último, uma codificação Golomb-Rice adaptativa é realizada sobre os resíduos. Esta estratégia atinge uma taxa de compressão média de 42% para vídeos HD 1080p.

No trabalho de Lian et al. (2016) é proposto um algoritmo de compressão que utiliza dois passos de codificação. O primeiro é uma codificação diferencial entre amostras vizinhas aplicada sobre blocos  $16 \times 16$  e o segundo é uma codificação VLC Exp-Golomb aplicado sobre esses resíduos. Nos resultados apresentados pelo autor, a taxa de compressão do algoritmo é de 68%, em média, para vídeos HD 1080p e 2160p. Os autores também propõem a arquitetura de hardware para o algoritmo que utiliza 70k gates para o codec completo, além de uma organização de memória. Com esses resultados, o sistema atinge uma taxa de redução de energia relacionada a memória externa de 56%.

O algoritmo de compressão proposto em Li et al. (2017) utiliza codificação diferencial vertical ou horizontal em amostras vizinhas em blocos  $16 \times 16$  e, ao resíduos gerados, é aplicado uma codificação VLC fixa. Além disso, para reduzir os elementos de sintaxe gerado pelos modos de predição, um VLC baseado em códigos de *Huffman* é empregado. Os autores também propuseram uma arquitetura de hardware para o codec, a qual utiliza 65k gates em tecnologia CMOS 90nm e 3,7KB de memória SRAM. A taxa de redução de largura de banda reportada pelos autores é de  $\approx 55\%$ . Entretanto, resultados de consumo de energia não são apresentados pelos autores.

Em Gu, Zeng e Fan (2018) um algoritmo de compressão sem perdas e que atinja altas taxas de processamento e baixo custo de hardware é proposto. A solução proposta, como em outros trabalhos, utiliza uma codificação DPCM com alguns modos de codificação aplicada a blocos  $8 \times 4$ . Após os resíduos gerados uma codificação VLC fixa é aplicada sobre eles. Além disso, em blocos adjacentes que utilizaram o mesmo modo de codificação são reagrupados para otimizar a compressão. A taxa de compressão obtida

com essa solução é de 51%, mas não há informações detalhadas a respeito dos vídeos utilizados para gerar esses resultados. A arquitetura proposta foi sintetizada para uma biblioteca de células CMOS de 65nm, a qual utilizou 44K gates no codec e é capaz de processar 60 QPS na resolução UHD 2160p. Entretanto, os autores não informaram os resultados de consumo de energia da solução proposta.

No artigo de Lee et al. (2020) é proposto um algoritmo de compressão sem perdas que incorpora três técnicas: partição em árvore, predição de meio pixel e uma codificação binária baseada em grupo. A partição em árvore classifica um bloco de  $16 \times 8$  em Tronco (*Trunk*), Ramificação (*Branch*) e Folha (*Leaf*). Com a partição em árvore, a predição de meio pixel produz resíduos individuais para Tronco, Ramificação e a Folha. Uma codificação binária baseada em grupo converte esses resíduos em códigos mais eficientes. A compressão sem perdas gera uma economia em largura de banda de memória de 55,4%. A arquitetura foi implementada usando a tecnologia CMOS de 180nm, utilizando 76K gates com 19mW de dissipação de potência. A taxa de processamento máxima pode chegar a 6,4 Gpixels/s, atingindo UHD 2160 a 60 QPS.

### 4.3 Intervalo de busca dinâmico

Estratégias que utilizam intervalo de busca dinâmico (*Dynamic Search Range – DSR*) propõem reduzir a largura de banda da memória usando algoritmos para ajustar a área de busca dinamicamente. Esse ajuste é geralmente executado com base nos tamanhos dos vetores de movimento encontrados para o quadro anterior.

A relação entre a predição do vetor de movimento, a diferença entre as PUs e o tamanho da área de busca são considerados no trabalho de Li et al. (2015). Nesse trabalho, a codificação é dividida em duas partes: i) a etapa de aprendizado, que coleta alguns dados relacionados ao tamanho dos vetores de movimento, os quais são utilizados no futuro para formar uma árvore no processo de predição; e ii) a etapa de predição que calcula o melhor tamanho da área de pesquisa. Essa estratégia foi implementada no algoritmo *Full Search*, apresentando uma redução em complexidade de 91% com um custo em BD-rate de 0,34%. Nenhuma arquitetura de hardware ou resultado em energia foram apresentados.

Em Pakdaman et al. (2018) é proposto um novo algoritmo rápido para ME visando o HEVC, o qual estima um único ponto de busca inicial adequado com intervalo de busca reduzido, para isso utiliza informações disponíveis a partir dos blocos vizinhos codifica-

dos. Com o método apresentado no artigo, o tempo de codificação do TZS foi reduzido em aproximadamente 70%, enquanto que a largura de banda com a memória externa foi reduzida em 78% em média. As perdas em BD-rate atingiram 1,1% e os autores não apresentaram arquitetura de hardware nem resultados de energia.

Singh e Ahamed (2018) propõe um algoritmo e uma arquitetura de hardware para ME no HEVC. Esta ME usa um algoritmo chamado *Modified Hexagonal Grid Search* (MHGS), o qual utiliza um menor número de pontos iniciais para o processo de busca da ME. Essa ME apresentada por Singh e Ahamed (2018) utiliza reuso de dados entre áreas de busca vizinhas, e técnicas de computação aproximada como sub-amostragem, *bit dropping* e um intervalo de busca adaptativo para reduzir ainda mais a dissipação de potência. Essa ME apresentou uma redução de 54% em complexidade quando comparado ao TZS, com um incremento de 1,8% em BD-rate. A ME proposta atinge uma taxa de processamento de 30 QPS na resolução UHD 2160p com 151,7mW de dissipação de potência.

#### 4.4 Memórias não voláteis

Nesta seção serão discutidos alguns trabalhos que exploram o uso de memórias não voláteis, PCRAM, STT-MRAM e ReRAM, em codificadores de vídeo para reduzir o consumo de energia. A utilização de memória voláteis com não voláteis tem sido encontrado na literatura como hierarquia de memória híbrida, i.e. NV-RAM + DRAM e SRAM.

O artigo apresentado por Bi et al. (2017) propõe a substituição das memórias SRAM por memórias magnéticas STT-MRAM. Desta forma, é possível armazenar dois ou mais bits em uma célula de memória. A arquitetura proposta pelos autores divide a memória em duas partes: uma região utilizando memórias multiníveis (MLC STT-MRAM), as quais são mais densas e lentas; e outra em memórias com um nível (SLC STT-MRAM), que são mais rápidas e menores. Com essa arquitetura é possível melhorar a performance do sistema em  $\approx 10\%$  e o consumo de energia em  $\approx 9\%$ .

No artigo proposto por Kultursay et al. (2013) os autores exploram a possibilidade de usar a tecnologia de memória STT-MRAM para substituir completamente uma DRAM na memória principal. Os autores utilizaram um abordagem para melhorar o desempenho de escrita na memória STT-MRAM, para isso, foram utilizadas técnicas intituladas (i) escrita parcial (*partial write*) e (ii) desvio de escrita no *buffer* linha (*row buffer write by-*

*pass*). Na primeira, quando ocorrer um conflito no *buffer* de linha apenas onde ocorreu o conflito deverá ser feita a reescrita ao invés de todo *array* de dados. Na segunda, o comportamento padrão de escrita na STT-MRAM é alterado, quando essa técnica é utilizada a escrita não utiliza o *buffer* de linha, acessando diretamente o *array* de dados. Com isso, as escritas não impactam no consumo energético do *buffer* de linha, havendo somente um aumento no consumo energético necessário para a troca da orientação magnética do MTJ da STT-MRAM. Com essas técnicas foi possível melhorar o desempenho e o consumo de energia da memória principal STT-MRAM. Os resultados obtidos indicam que uma memória principal STT-MRAM de mesma capacidade pode fornecer desempenho comparável à memória principal DRAM, com uma redução média de 60% no consumo de energia.

No artigo de Khan, Shafique e Henkel (2013) é proposta uma arquitetura de memória chamado AMBER. Essa arquitetura é composta por aceleradores de hardware e uma memória híbrida, SRAM e MRAM, interna de alta capacidade de armazenamento. Um esquema de gerenciamento de energia adaptativo é proposto, ele controla as regiões e setores específicos de memória híbrida para economizar energia. Com esse gerenciamento é possível realizar *power gating* nos módulos das memórias MRAM sem perder dados. Essa abordagem foi simulada juntamente com o software de referência do HEVC (HM-9.2), os resultados mostraram que foi possível reduzir o consumo de energia das memórias internas, em média, em 43%.

Em Sampaio et al. (2019) é proposta uma arquitetura de memória de vídeo distribuída (*hybrid scratchpad video memory – Hy-SVM*) para o HEVC paralelo. Essa arquitetura explora o processamento paralelo de vídeo baseado em Tiles para processadores com múltiplos núcleos. A arquitetura emprega memórias *scratchpad* distribuídas, as quais integram vários SRAM SPMs privados de nível L1 e STT-MRAM privados e compartilhados de nível L2, essas memórias suportam a reutilização de dados intra-tiles e inter-tiles entre vários núcleos. Além disso, é proposto um gerenciamento adaptativo de energia para o Hy-SVM, o qual leva em consideração o tamanho e a forma da área de sobreposição prevista para selecionar *power gating* apropriados para diferentes regiões das SPMs privadas e compartilhadas. Os resultados de redução no consumo de energia de memória externa dessa abordagem quando comparados ao Level C atingem 81%, em média, para janelas de busca grandes,  $256 \times 256$ , onde há maior reaproveitamento de dados.

No trabalho de Penny et al. (2019) são avaliadas hierarquias de memória composta por tecnologia NV-RAMs para codificadores de vídeo. Nesse trabalho os autores,

exploram a utilização de memória STT-MRAM e ReRAM com alternativas à tecnologia DRAM, mantendo a memória interna com SRAM. Os resultados gerados pelos autores mostram que uma hierarquia de memória formada com STT-MRAM na memória externa reduz o consumo de energia em 33,4%, em média, enquanto que utilizando a ReRAM na memória externa a redução atinge 57,6%. Ambas hierarquias utilizam SRAM como memória Cache.

#### **4.5 Computação aproximada**

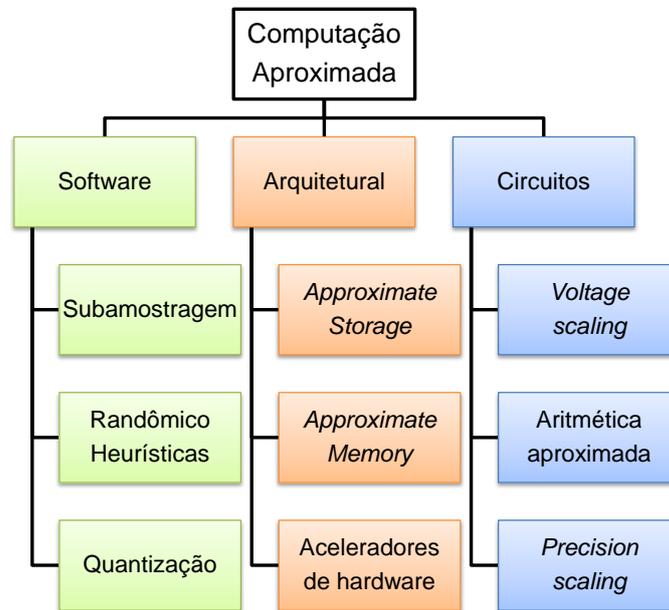
Os sistemas computacionais estão cada vez mais complexos e exigem algoritmos sofisticados para fornecer respostas suficientemente boas de forma rápida, que apresente escalabilidade e eficiência energética (XU; MYTKOWICZ; KIM, 2016). Na codificação de vídeo isso não é diferente, uma vez que apresenta resoluções e taxas de amostragem cada vez maiores, algoritmos e particionamento cada vez mais complexos além do uso crescente em dispositivos móveis, onde a capacidade energética é sempre uma restrição. Nesse sentido, a computação aproximada é muitas vezes a única maneira de atender a esses objetivos de forma aceitável.

A computação aproximada introduz de forma deliberada erros que são aceitáveis no processo de computação, gerando ganhos significativos em eficiência energética (XU; MYTKOWICZ; KIM, 2016). Nesse sentido, os codificadores de vídeo apresentam uma propriedade intrínseca de resiliência de erro, pois como visto no capítulo 2, os codificadores de vídeo utilizam técnicas de codificação com dados aproximados, como subamostragem e quantização, as quais diminuem de forma considerável o tempo de codificação, mantendo a qualidade do vídeo e a taxa de bits em valores aceitáveis. Isso é possível devido à limitada capacidade perceptiva dos seres humanos (RICHARDSON, 2004), e erros ocasionais como a perda de um quadro específico ou uma pequena perda de qualidade de imagem, raramente afetam a satisfação do usuário. Desta forma, técnicas de computação aproximada são bastante exploradas na codificação de vídeo, principalmente para uma melhor eficiência energética.

Segundo Xu, Mytkowicz e Kim (2016), a computação aproximada pode estar presente em três níveis do sistema: i) no nível de software, onde algo randômico pode ser utilizado para se chegar ao resultado de forma mais rápida ou quando nem todas as alternativas são testadas, como ocorre com a utilização de algoritmos rápidos utilizados na ME; ii) no nível arquitetural, por exemplo utilizando memórias aproximadas ou armaze-

namento de dados aproximados; iii) no nível de circuitos, onde pode ser utilizada lógica imprecisa, *voltage scaling*, *precision scaling*, entre outras. A Figura 4.5 ilustra esses níveis com alguns exemplos aplicados na codificação de vídeo.

Figura 4.5: Computação aproximada aplicada na codificação de vídeo



Fonte: Próprio autor.

Assim, nessa seção serão apresentados alguns trabalhos que utilizam computação aproximada na codificação de vídeo, principalmente no nível arquitetural. Esses trabalhos exploram principalmente algum fenômeno que causa erros nos dados armazenados em memória, e desta forma, se valem da resiliência do codificador de vídeo para economizar energia.

O artigo de Gong et al. (2017) apresenta o *sizing-priority-based application-driven memory* (SPIDER), uma metodologia de projeto de memória orientada a aplicação baseada em prioridades. As duas prioridades avaliadas são o tamanho da memória e a qualidade do vídeo codificado. Nesse trabalho foram avaliados o comportamento de memórias SRAM com baixa tensão de alimentação. Os resultados para aplicações de vídeo demonstraram que para memórias SRAM a  $500mV$  é possível reduzir em até 70% o consumo de energia quando comparado a uma memória SRAM com tensão de  $1V$ . O trabalho apresenta apenas os resultados de qualidade comparados a uma memória SRAM sem nenhuma estratégia que diminua os erros gerados pelo processo de *voltage downscaling*. Esse abordagem dificulta uma análise da perda de qualidade real gerada pela memória. Além disso, os resultados de taxa de bits não foram apresentados, o que dificulta ainda mais uma avaliação mais profunda dessa metodologia.

O trabalho de Sampaio et al. (2015) explora o uso de MLC STT-MRAM como substituto das SRAM para memórias cache. As células MLC exigem um consumo de energia considerável para manter os dados armazenados protegidos e livres de erro. Nesse trabalho os autores diminuem essa proteção e dividem a memória em duas partes, uma como um conjunto de Cache “confiável” onde as informações precisam ser armazenadas com confiabilidade, e outra como “não confiável”, onde as informações não precisam necessariamente confiabilidade e podem tolerar erros. Os autores utilizaram como estudo de caso para os testes do sistema de memória dois codificadores vídeo do HEVC, o HM e o X265. Os resultados de consumo de energia mostram uma redução que varia de 7% a 19%, enquanto que a qualidade do vídeo codificado sofre uma pequena degradação 0.5db.

Além desses trabalhos, a computação aproximada também pode ser utilizada nas memórias DRAM. Os parâmetros característicos das DRAM, como os tempos e as correntes, listados nos *datasheets* são muito pessimistas devido às altas margens de segurança adicionadas pelos fornecedores para garantir o funcionamento correto e suficiente em condições extremas (JUNG et al., 2016). Semelhante a estes tempos e correntes, a taxa de *refresh* das DRAM recomendada pelos fornecedores e pelo JEDEC (64ms) adiciona uma grande faixa de proteção (JUNG et al., 2016). Segundo Bhati et al. (2015) as taxas de *refresh* nas memórias DRAM são responsáveis por  $\approx 50\%$  do consumo de energia dessa memória.

No trabalho de Jung et al. (2016) é realizado um levantamento estatístico sobre o comportamento de memórias DRAM em relação ao tempo de retenção dos dados, variação do processo e temperatura quando o tempo de *refresh* aumenta. Com base nos dados desse trabalho é possível verificar que aplicações que possuem uma resiliência aos erros, como codificação de vídeo, podem tolerar esses erros e, portanto, o consumo de energia geralmente pode ser reduzido com uma perda mínima da qualidade final (JUNG et al., 2016).

Além dos trabalhos citados anteriormente, todos os trabalhos apresentados na Seção 4.2.1, que trata sobre os algoritmos de compressão com perdas de informação, são também soluções que utilizam computação aproximada para armazenar nas células de memória as informações de forma aproximadas.

#### 4.6 Síntese dos trabalhos relacionados

Alguns apontamentos podem ser feitos sobre os trabalhos estado da arte. Primeiro, os trabalhos sobre reuso de dados geralmente atingem uma alta taxa de redução de largura de banda entre o núcleo de processamento e a memória externa. Entretanto, cabe salientar que esta abordagem reduz apenas os acessos de leitura da memória, sendo que o número de acessos para a escrita na memória externa permanecem iguais. Além disso, os autores não contabilizam os custos adicionais da inserção da memória cache, necessária para o reuso. A inserção desta Cache implica em novos custos para manter e acessar os dados armazenados, tanto em área quanto em consumo de energia.

Os trabalhos com soluções de compressão sem perdas de qualidade utilizam diversas técnicas de codificação em conjunto para atingirem altas taxas de compressão. Isto implicará em alta dissipação de potência e elevada latência quando estas soluções forem implementadas em hardware. Além disso, poucos trabalhos apresentam uma solução arquitetural em hardware, o que dificulta a avaliação do consumo energético dessas soluções.

Os trabalhos que utilizam computação aproximada podem gerar *drifting* entre codificador e decodificador, e isto resulta em perdas de eficiência de codificação, uma vez que a predição inter-quadros usará as amostras do quadro de referência (que agora apresenta perdas em relação ao original), degradando ainda mais a qualidade do vídeo codificado. Nenhum dos trabalhos citados apresenta dados sobre a perda de qualidade final do vídeo codificado a partir destes quadros de referência com perdas de qualidade. Estas perdas podem ser cumulativas, pois os quadros codificados a partir destas amostras de referência com perdas podem ser usados como referência para a codificação de quadros futuros.

Além disso, nos trabalhos que utilizam NV-RAM sempre devem ser observados o *write endurance*, o tempo de retenção dos dados nas células, o consumo de energia relacionado a escrita, o tempo de acesso e a largura de banda dessas memórias, os quais mudam de tecnologia para tecnologia.

Como pode ser visto nas seções anteriores, há uma clara oportunidade de pesquisa no tema de redução de consumo de energia de memória em codificadores de vídeo, uma vez que vários grupos de trabalho tem atuado em diferentes frentes neste tema ao redor do mundo. Ainda, os trabalhos relacionados não são definitivos em alguns aspectos como taxa de compressão, custo computacional, quantidade de memória utilizada, tecnologia de

memória, técnica de aproximação, entre outros. Além disso, poucos trabalhos propõem soluções para o HEVC ou padrões além do HEVC, a maioria são propostas para o padrão H.264/AVC. E a crescente demanda por vídeos de alta resolução em dispositivos móveis faz com que este tema tenha novos desafios ao longo do tempo. Desta forma, esta tese busca preencher estas lacunas, com uma aplicação de alta eficiência energética explorando as diversas técnicas que podem ser empregadas para a redução do consumo de energia.

## 5 ANÁLISE DE EFICIÊNCIA DE CODIFICAÇÃO DO TZS

Nesse capítulo serão apresentadas análises sobre a distribuição dos acessos à área de busca realizadas pelo algoritmo de busca *Test Zone Search* (TZS). Estas análises possibilitam melhor entendimento sobre o comportamento de acessos realizados pelo TZS dentro de uma área de busca e o conhecimento das regiões mais e menos acessadas. Estas análises e o domínio dessas informações são de fundamental importância para o desenvolvimento dessa tese, uma vez que as técnicas de computação aproximada exploradas nessa tese, cortes na área de busca e *bit dropping*, utilizam essas informações para atingirem o máximo de eficiência, gerando alta redução no consumo de energia com o mínimo de perda na eficiência de codificação.

Esse capítulo está dividido em cinco seções: a Seção 5.1 apresenta a metodologia empregada nesse trabalho para a obtenção das informações referentes aos acessos à área de busca e os resultados de eficiência de codificação. A Seção 5.2 o primeiro estudo realizado no TZS, que foi a definição do intervalo de busca que é utilizado pelo sistema *baseline*. A análise dos acessos às amostras da área de busca é apresentada na Seção 5.3. A Seção 5.4 apresenta as técnicas de computação aproximada empregadas nessa tese, *bit dropping* e cortes na área de busca. Essas técnicas são individualmente exploradas e em conjunto uma com a outra. Análises comparando essas técnicas também são apresentadas nessa seção. A Seção 5.5 apresenta as considerações finais do capítulo.

### 5.1 Metodologia

Esta seção apresenta a metodologia utilizada nesse trabalho para quantificar tanto os acessos às amostras dentro da área de busca quanto a eficiência de codificação do TZS.

O algoritmo TZS foi utilizado como algoritmo de busca da ME, uma vez que esse algoritmo é um dos mais eficientes encontrados na literatura, conforme discutido e apresentado no capítulo 2. A etapa de Pré-Busca do TZS foi desabilitada, já que o fato de que a área de busca pode não ser sempre formada ao redor do bloco colocalizado dificulta implementações em hardware e impede o uso eficiente de estratégias de reuso de dados.

A etapa de Pré-Busca do TZS possui cinco modos de predição para selecionar a posição na área de busca do quadro de referência: um baseado no bloco colocalizado e quatro preditores baseados em blocos vizinhos. Ao aplicar preditores baseados em blocos vizinhos, o TZS pode mover o área de busca para uma região distante da posição do bloco

colocalizado, impedindo um comportamento preditivo dos acessos à memória externa e gerando acesso irregular à memória externa. Esses dois fatores limitam a utilização de abordagens eficientes para o gerenciamento da memória, aumentando o consumo de energia e limitando a taxa de processamento da ME.

Dessa forma, para conseguir um melhor custo benefício entre desempenho e custo de hardware, apenas a predição baseada no bloco colocalizado foi mantida no TZS para todas as avaliações. Esta abordagem também é adotada por outras soluções de hardware propostas na literatura, tais como Singh e Ahamed (2018), Porto et al. (2020) e Afonso et al. (2019). Conforme apresentado em Singh e Ahamed (2018), a consequência de não usar todos os cinco preditores do TZS leva a uma perda de 0,25% em eficiência de codificação. Assim, o TZS foi modificado para utilizar apenas a predição em torno do bloco colocalizado.

Além disso, como este trabalho foca na estimação de movimento, este módulo foi isolado em todo o trabalho. Com isso, técnicas como o *early skip* e a ocorrência de blocos intra em quadros inter também foram desabilitadas. Ainda, a configuração *Low Delay P* foi adotada para todo o trabalho. Essas configurações foram realizadas no software de referência do HEVC, o HM 16.6 (JCT-VC, 2014).

Em todos testes realizados foram utilizados quatro valores de QPs: 22, 27, 32 e 37. Além disso, foi utilizada uma CTU de tamanho  $64 \times 64$  amostras, tamanho máximo de CTU. Ainda, dado que os blocos quadráticos são muito mais representativos em termos de BD-rate e complexidade do que os blocos não-quadráticos (AFONSO et al., 2016), os blocos não-quadráticos foram desabilitados no processo de ME para esses experimentos. O tamanho do intervalo de busca é o mesmo em todos os experimentos, para definição desse tamanho será apresentado na próxima seção um estudo sobre o impacto na eficiência de codificação do tamanho do intervalo de busca.

Por fim, o HM 16.6 foi modificado para obter as informações necessárias para estas análises por meio de um arquivo de *trace*. O *trace* implementado no HM retorna para cada PU processada cinco informações, sendo elas: as posições,  $X$  e  $Y$ , referentes a posição da CU em relação ao quadro original, que está sendo codificado; o deslocamento,  $X$  e  $Y$ , da PU em relação à CU em processamento, e o tamanho do bloco. Assim, após a codificação do vídeo, é possível extrair do *trace* gerado as informações referentes as amostras acessadas, contabilizando quantas vezes cada amostra foi acessada dentro da área de busca. Além disso, com essas informações a quantidade de PUs processadas também pode ser obtida.

O conjunto de testes é composto por doze sequências de vídeo. As sequências de testes foram divididas em dois grupos, sendo  $\alpha$  o grupo com vídeos HD 1080p composto pelos vídeos *BasketballDrive* ( $\alpha1$ ), *BQTerrace* ( $\alpha2$ ), *Cactus* ( $\alpha3$ ), *Kimono* ( $\alpha4$ ), *ParkScene* ( $\alpha5$ ) e *Tennis* ( $\alpha6$ ). E o grupo  $\beta$  composto pelos vídeos UHD 2160p *CampFire* ( $\beta1$ ), *CatRobot* ( $\beta2$ ), *DaylightRoad* ( $\beta3$ ), *FoodMarket* ( $\beta4$ ), *ParkRunning* ( $\beta5$ ) e *Tango* ( $\beta6$ ). Em todas as simulações sempre foram considerados os 100 primeiros quadros de cada vídeo.

## 5.2 Definição do intervalo de busca

O intervalo de busca, como explicado no capítulo 2, é a área que fica ao redor da CTU, as quais juntas formam a área de busca. Esse intervalo de busca pode ser especificado de diversos tamanhos e acaba sendo definido de acordo com o projeto do codificador. Porém, o tamanho do intervalo de busca acaba interferindo na eficiência de codificação, quanto maior o intervalo de busca maior será a área disponível para o algoritmo de busca encontrar o melhor casamento, e conseqüentemente, aumentam as chances de atingir uma melhor eficiência na codificação. Entretanto, quanto maior for essa área maior será o custo energético de armazenamento e processamento dessas informações. Desta forma, esta seção irá explorar os principais tamanhos de intervalo de busca a fim de encontrar o melhor custo benefício entre eficiência de codificação, qualidade e área utilizada da área de busca.

A Tabela 5.1 apresenta os resultados de *bitrate* e PSNR para os três tamanhos de intervalo de busca testados:  $[-128, +128]$  (SR 128),  $[-96, +96]$  (SR 96) e  $[-64, +64]$  (SR 64). Intervalos de busca menores não foram testados, pois uma das técnicas de computação aproximada utilizada nessa tese é realização de cortes na área de busca. Desta forma, ao restringir demais essa área será imposta uma restrição ao uso da técnica, limitando sua contribuição. Esses resultados representam a média para os quatro QPs avaliados. Dado que o PSNR médio é igual entre os tamanhos de intervalo de busca, será avaliado o resultado de *bitrate* desses intervalos. Nota-se que para vídeos HD 1080p o intervalo de busca que apresentou menor *bitrate* foi o SR 64. Para os vídeos UHD 2160p o menor *bitrate* é encontrado no SR 96. Na média para os doze vídeos o menor *bitrate* é encontrado no SR 128. Entretanto, a diferença entre os resultados de *bitrate* dos três tamanhos de intervalo é muito pequena, 0,008%.

Além disso, outro fator que deve ser considerado na escolha do intervalo de busca

Tabela 5.1: Resultados médios de *bitrate* e PSNR para os 3 intervalos de busca.

vídeo	[-128, +128]		[-96, +96]		[-64, +64]	
	Bitrate	PSNR	Bitrate	PSNR	Bitrate	PSNR
$\alpha_1$	8290,4	36,7	8287,5	36,7	8292,0	36,7
$\alpha_2$	19663,5	34,6	19687,6	34,6	19653,8	34,6
$\alpha_3$	9084,4	35,3	9095,0	35,3	9088,6	35,3
$\alpha_4$	3104,8	38,7	3104,2	38,7	3102,4	38,7
$\alpha_5$	3938,4	35,5	3936,5	35,5	3936,2	35,5
$\alpha_6$	4056,8	37,4	4054,1	37,4	4057,6	37,4
1080p	8023,1	36,4	8027,5	36,4	8021,8	36,4
$\beta_1$	39201,4	36,8	39201,5	36,8	39191,0	36,8
$\beta_2$	17841,6	37,8	17838,3	37,8	17842,9	37,8
$\beta_3$	30583,2	36,0	30587,6	36,0	30621,5	36,0
$\beta_4$	6694,7	43,3	6694,1	43,3	6725,9	43,3
$\beta_5$	71241,1	37,4	71228,9	37,4	71214,3	37,4
$\beta_6$	15448,2	38,7	15437,7	38,7	15441,6	38,7
2160p	30168,4	38,3	30164,7	38,3	30172,9	38,3
<b>Total</b>	19095,7	37,3	19096,1	37,3	19097,3	37,3
<b>AB (bytes)</b>	102.400		65.536		36.864	

Fonte: Próprio autor.

é o seu tamanho, uma vez que quanto maior essa área, maior será o custo para armazenar e processar essas informações. Nesse sentido, a última linha da Tabela 5.1 apresenta os tamanhos das áreas de busca, em Bytes, considerando os respectivos intervalos de busca para uma CTU  $64 \times 64$ . O SR 128 é o que possui o maior tamanho, requerendo uma área de  $100KB$  para armazenar toda área de busca. No SR 96 há uma redução de 36% nessa área quando comparado ao SR 128, totalizando  $64KB$ . Para o SR 64, o tamanho de memória necessária para armazenar essa área de busca é de  $36KB$ , isso representa uma redução de 64% em relação ao SR 128 e de 43,8% em relação ao SR 96. Devido as altas taxas de redução de área em relação aos tamanhos maiores e a pequena diferença em *bitrate* em relação a esses mesmos tamanhos, a utilização do SR 64 se torna propícia para ser empregado como intervalo de busca base para os próximos experimentos.

### 5.3 Análise dos acessos à área de busca

A partir das configurações apresentadas na seção anterior foi possível gerar um trace para cada sequencia de vídeo. Com as informações obtidas pelo trace foi contabilizado o número de acessos feitos a cada amostra na área de busca durante o processamento de 100 quadros de cada uma das doze sequencias de testes, a fim de verificar o comporta-

mento dos acessos que ocorre dentro da área de busca.

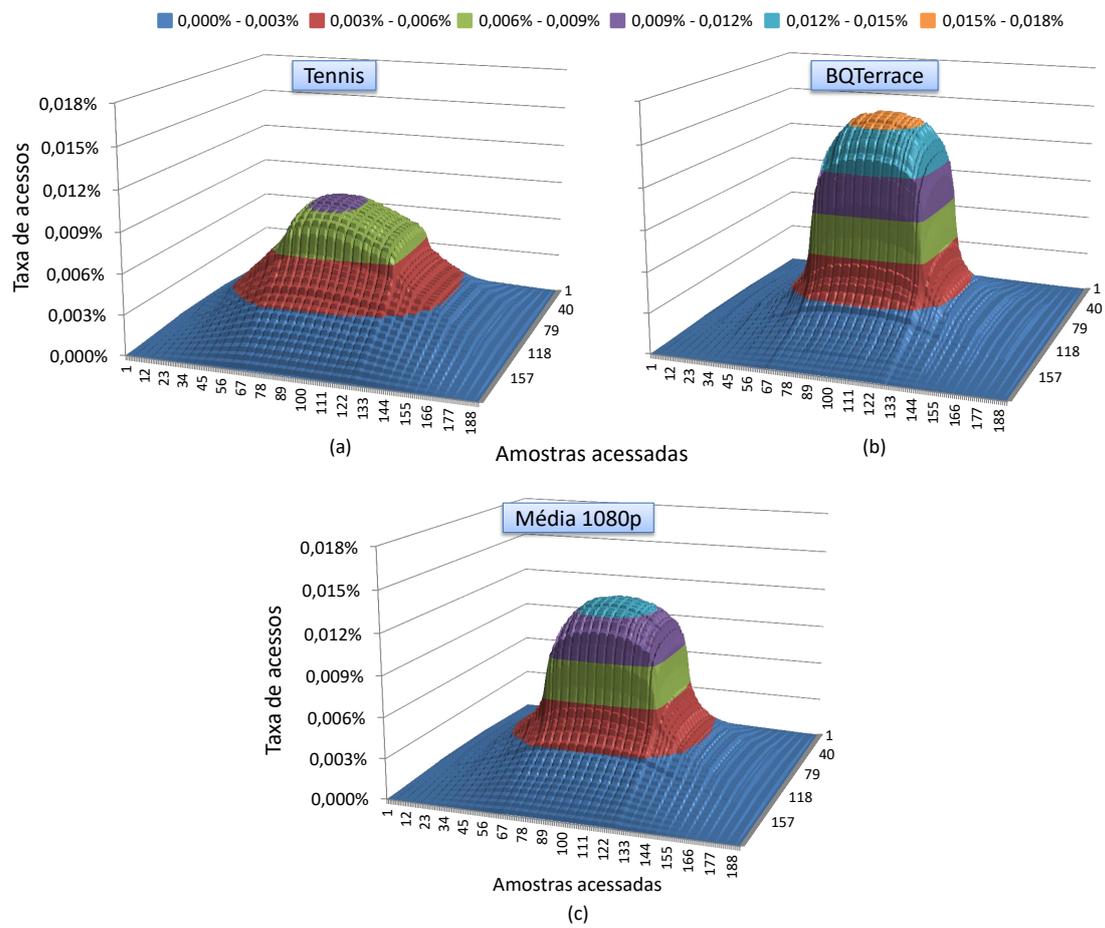
As Figuras 5.1 e 5.2 apresentam os histogramas com o percentual de acesso de cada amostra dentro da área de busca. Somando o total de ocorrências de acessos às amostras, chega-se em 100%. Cada histograma apresentado nessas figuras representam a média para os 4 QPs utilizados. A Figura 5.1 mostra os histogramas para vídeos HD 1080p, considerando dois casos extremos, *Tennis* (Figura 5.1(a) - acessos menos centralizados na área de busca) e *BQTerrace* (Figura 5.1(b) - o mais centralizado). Além disso, a Figura 5.1(c) apresenta o histograma da média de acessos para os seis vídeos HD 1080p. A Figura 5.2, apresenta os histogramas para os vídeos UHD 2160p, onde o vídeo *CampFire* (Figura 5.2(a)) representa o vídeo com os acessos menos centralizados e o vídeo *CatRobot* (Figura 5.2(b)) representa o vídeo UHD 2160p onde os acessos foram mais centralizados na área de busca. A Figura 5.2(c) apresenta o histograma para a média dos 6 vídeos UHD 2160p. Os histogramas para o restante dos vídeos do conjunto de testes estão disponíveis no Apêndice A.

Como se pode notar nas Figuras 5.1 e 5.2, a região mais acessada na área de busca é a central, de onde começa a expansão do TZS durante a primeira busca. Isso aconteceu para todas as sequências de vídeo tanto nos vídeos HD 1080p quanto nos UHD 2160p. Por exemplo, se considerarmos uma região central de  $128 \times 128$  amostras na área de busca que possui  $192 \times 192$  amostras, i.e. 44,4% da área de busca original, essa área central representa 85,1% dos acessos à área de busca para os vídeos HD 1080p. Já para os vídeos UHD 2160p essa mesma área é responsável por 82,6% dos acessos. Na média para os doze vídeos, essa área central corresponde a 83,8% dos acessos às amostras da área de busca. Ou seja, ocorrem 5,2 vezes mais acessos a essa área central do que nas áreas de borda da área de busca.

Entretanto, vídeos com características de alta movimentação ou textura, como os exemplos *Tennis* (Figura 5.1(a)) e *CampFire* (Figura 5.2(a)), apresentaram mais acessos próximos da borda da área de busca do que os vídeos que possuem baixa movimentação ou textura, onde os acessos estão mais concentrados na região central. Tomando como exemplo as sequências HD 1080p, para o vídeo *Tennis*, essa área central corresponde por 80,4% dos acessos. Já para a sequência *BQTerrace*, essa mesma área central corresponde por 89,2% dos acessos à área de busca.

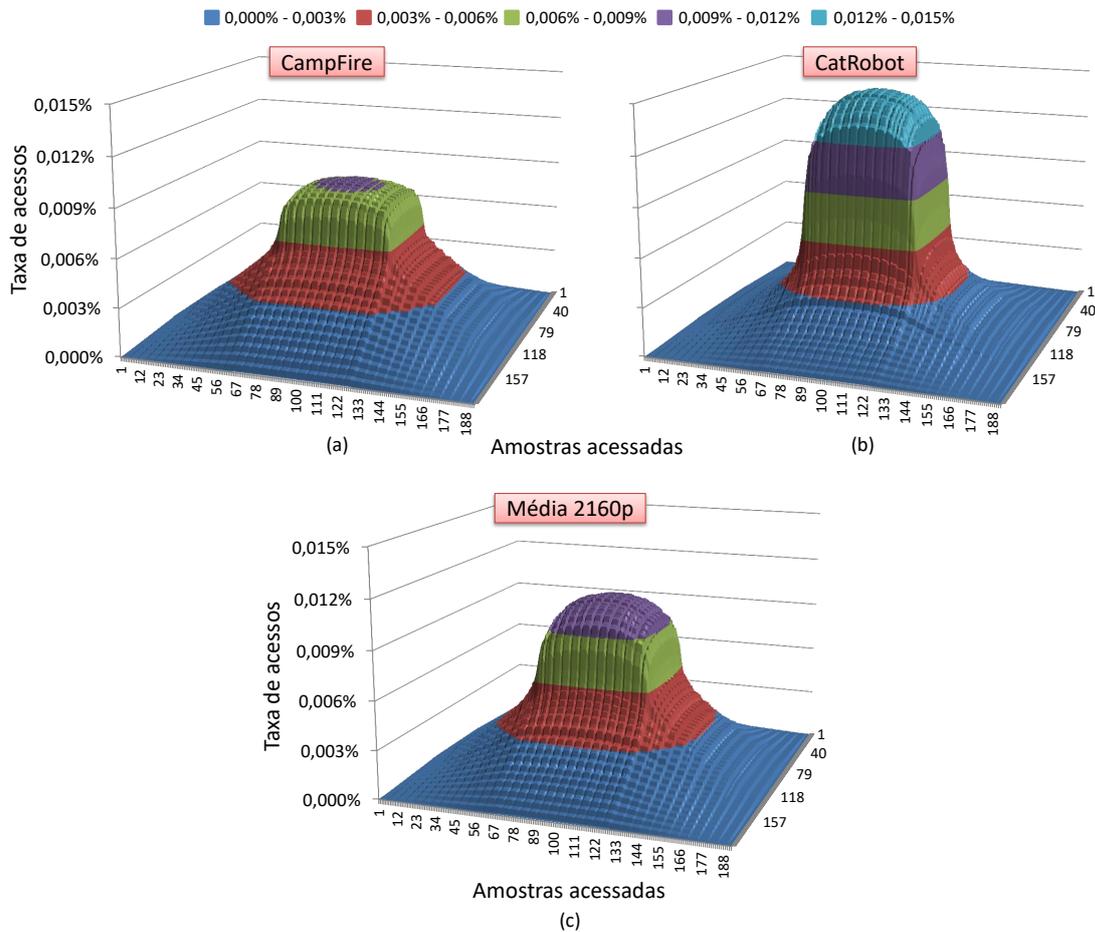
Esses histogramas reafirmam a ocorrência da maioria dos acessos na região central da área de busca, ou seja, alguns trechos da área de busca recebem poucos acessos enquanto a pequena região central concentra a maior parte dos acessos. Nesse sentido foi

Figura 5.1: Histogramas com acessos à área de busca  $192 \times 192$  para vídeos 1080p.



Fonte: Próprio autor.

Figura 5.2: Histogramas com acessos à área de busca  $192 \times 192$  para vídeos 2160p.



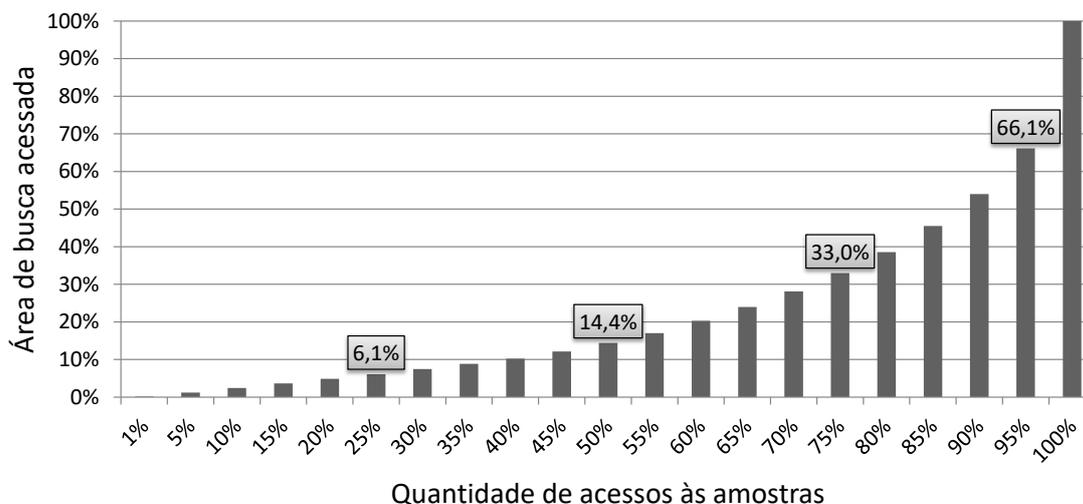
Fonte: Próprio autor.

feito um estudo a partir da média de acessos gerados para cada vídeo, onde foi possível obter uma relação entre o número de acessos e a proporção da área utilizada dentro da área de busca. Essa relação é apresentada na Figura 5.3.

O gráfico da Figura 5.3 foi gerado a partir da média dos doze vídeos do conjunto de testes, 6 HD  $1080p$  e 6 UHD  $2160p$ , para os quatro QPs. Assim, esse gráfico é composto pelos resultados de traces de 48 simulações com 100 quadros cada uma.

Nota-se no gráfico da Figura 5.3 que 25% dos acessos às amostras da área de busca (eixo horizontal) estão em uma região que corresponde a apenas 6,1% do tamanho total da área de busca. Ao percorrermos o gráfico se percebe que 50% dos acessos às amostras estão em apenas 14,4% da área de busca. Além disso, 95% dos acessos são realizados em uma região correspondente a 66,1% da área de busca. A relação apresentada neste gráfico é importante, pois demonstra que existem oportunidades para reduzir os acessos a essas amostras pouco acessadas da área de busca, gerando baixo impacto na eficiência de

Figura 5.3: Relação entre quantidade de acessos e área acessada.



Fonte: Próprio autor.

codificação.

Nesse sentido, essa tese explora duas técnicas de computação aproximada, sendo que a primeira delas são cortes nas regiões da área de busca onde se tem os menores acessos, e a segunda é um *bit dropping* das amostras da área de busca. Essas técnicas e os resultados dos experimentos são discutidos nas próximas seções.

#### 5.4 Computação aproximada no TZS

Os codificadores de vídeo apresentam propriedade intrínseca de resiliência de erro, uma vez que já utilizam técnicas de codificação com dados aproximados, como subamostragem e quantização, as quais diminuem de forma considerável o tempo de codificação, mantendo a qualidade do vídeo e a taxa de bits em valores aceitáveis.

Como discutido na seção anterior, o algoritmo de busca TZS realiza muito mais acessos às amostras na região central da área de busca do que nas bordas, em média 5,2 vezes mais acessos para uma área central de tamanho  $128 \times 128$ . Nesse sentido, há uma clara oportunidade de explorar técnicas de computação aproximada que se beneficiem desse comportamento observado no TZS, principalmente para melhorar eficiência energética do codificador.

Assim, nessa seção serão apresentadas três técnicas de computação aproximada para reduzir os acessos às amostras da área de busca com o mínimo de impacto na efi-

ciência de codificação. Uma das técnicas é restringir os acessos as regiões da área de busca pouco acessadas, i.e. as bordas da área de busca. A segunda é a utilização de *bit dropping* nas amostras da área de busca. Desta forma, os bits menos significativos das amostras são deliberadamente cortados. Além disso, após análises dos resultados, as melhores configurações dessas técnicas foram unidas, formando uma abordagem que utiliza ao mesmo tempo cortes de regiões da área de busca e *bit dropping* na área remanescente. As próximas seções apresentam em detalhes essas abordagens, discutindo seus resultados.

#### 5.4.1 Cortes na área de busca

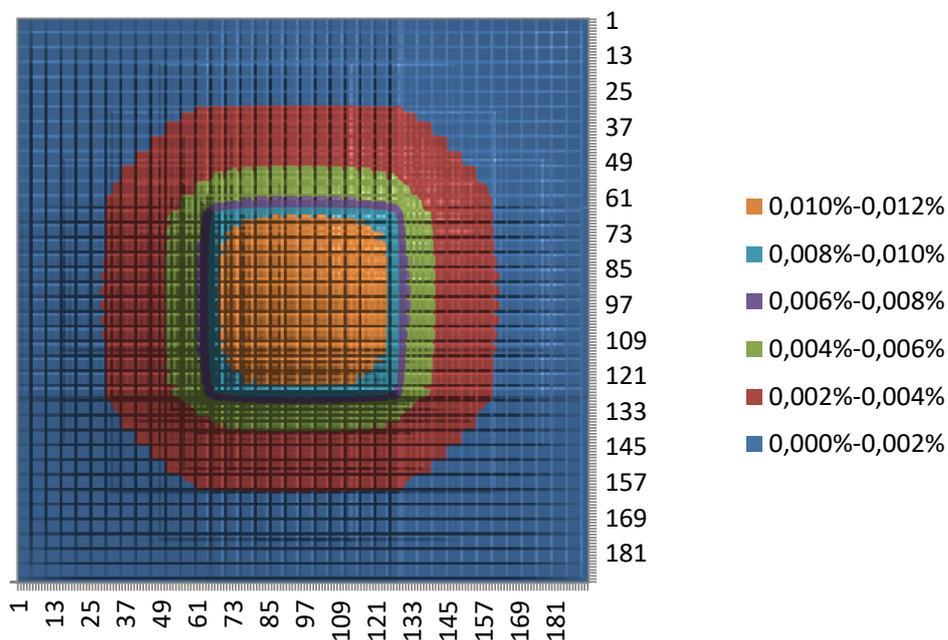
A técnica de cortes da área de busca restringe os acessos as regiões da área de busca pouco acessadas. Conforme visto anteriormente, essas regiões se referem as bordas da área de busca. Entretanto, para definir os tamanhos das áreas a serem cortadas foram realizadas análises referentes ao seu tamanho e qual impacto desse corte na eficiência de codificação. Para medir a eficiência de codificação dos vídeos foram utilizados resultados de BD-rate (BJONTEGAARD, 2001) para todas configurações e vídeos simulados.

A Figura 5.4 apresenta um histograma com a média de acessos para os doze vídeos do conjunto de testes e os 4 QPs, i.e. essa figura apresenta resultados de trace de 48 simulações. Nessa figura, a região em azul, mais próxima da borda da área de busca, representa as amostras que receberam menos acessos, enquanto as regiões em tons de vermelho, verde e roxo, mais centrais e mais próximas do topo do histograma, representam as amostras que receberam o maior número de acessos. Esse histograma representa a média dos histogramas apresentados nas Figuras 5.1(c) e 5.2(c) em uma vista superior.

Na Figura 5.4 podem ser observados dois comportamentos: i) a alta concentração de acessos na área central da área de busca, conforme discutido anteriormente; e ii) o formato ocasionado pelos acessos às amostras na área de busca, gerando um padrão de acessos losangular em torno da região central da área de busca. Esse formato losangular é devido ao processo de expansão do TZS, o qual pode seguir um padrão retangular ou losangular. Conforme discutido no capítulo 2.4.1, nesse trabalho foi utilizado o padrão losangular, mais rápido, que acabou gerando regiões de maiores acessos nesse formato.

Devido ao formato observado na Figura 5.4 dois padrões de cortes foram especificados, sendo um retangular e outro losangular. Além disso, para determinar as regiões a serem cortadas considerou-se 3 tamanhos simétricos de área de busca. Estes tamanhos foram escolhidos com base na quantidade de acessos que cada área recebeu. A

Figura 5.4: Histograma com a média dos acessos à área de busca  $192 \times 192$  para os vídeos HD 1080p e UHD 2160p.



Fonte: Próprio autor.

Tabela 5.2 relaciona os padrões de corte da área de busca e os tamanhos utilizados nos experimentos. Na primeira coluna dessa tabela estão relacionadas as configurações utilizadas, são três tamanhos de intervalo de busca com cortes:  $[-16, +16]$  (16),  $[-32, +32]$  (32) e  $[-48, +48]$  (48) para os dois padrões de cortes, retangular ( $R$ ) e losangular ( $L$ ). O tamanho  $[-64, +64]$  (64) é utilizado como referência para os demais tamanhos, uma vez que o intervalo de busca  $[-64, +64]$  não apresenta cortes. Como pode ser visto na segunda coluna da Tabela 5.2, as áreas de busca correspondentes são do mesmo tamanho tanto para o padrão retangular quanto para o losangular. Entretanto, os acessos às amostras nessas áreas de busca, terceira coluna, tem uma pequena mudança entre os dois padrões, o padrão losangular tem 0,5% a mais de acessos, em média, em relação ao padrão retangular para o mesmo tamanho de intervalo de busca.

Como essa diferença de quantidades de acessos entre os padrões pode apresentar melhores resultados de eficiência de codificação, ambos padrões e tamanhos foram avaliados nos experimentos de cortes da área de busca. Para realizar estes cortes na área de busca uma modificação foi realizada no HM 16.6, a fim de garantir que qualquer bloco que possuísse amostras fora da área restrita pelo corte não fosse acessado pelo algoritmo de estimação de movimento TZS. Para simplificar a implementação e a validação deste corte no software de referência, um bitmap foi utilizado.

Tabela 5.2: Informações sobre os padrões de corte da área de busca.

Intervalo de busca	Área de busca	Acessos correspondentes	Redução de área
16-R	96 × 96	66, 23%	75, 00%
32-R	128 × 128	83, 89%	55, 56%
48-R	160 × 160	95, 57%	30, 56%
16-L	96 × 96	66, 36%	75, 00%
32-L	128 × 128	84, 24%	55, 56%
48-L	160 × 160	95, 87%	30, 56%
64	192 × 192	100, 00%	0, 00%

Fonte: Próprio autor.

Nesse bitmap, cada posição representa um bloco de  $8 \times 8$  amostras, menor tamanho de bloco definido. Desta forma, como a área de busca original possui um tamanho total de  $192 \times 192$  amostras, o bitmap é uma matriz quadrada de tamanho  $24 \times 24$ , onde cada uma das 576 posições do bitmap armazenam o valor '1' ou '0'. Onde, o valor '1' indica que o corte está habilitado e, assim, a posição com este valor se torna inacessível para o TZS. O valor '0' indica que o corte está desabilitado e que a posição está acessível para o TZS. Deste modo, a posição do bloco a ser buscado é comparada com o bitmap e caso essa posição esteja fora dos limites disponíveis é ignorado e o TZS continua sua busca pelo melhor casamento.

A Tabela 5.3 apresenta os resultados de BD-rate para as 6 configurações de cortes na área de busca. Esses resultados de BD-rate são em relação a área de busca completa com intervalo de busca  $[-64, +64]$ . Além das configurações com cortes na área de busca, também são apresentados resultados de BD-rate para configurações onde o intervalo de busca foi especificado no HM. Esses intervalos especificados no HM são do mesmo tamanho das configurações de corte. Os resultados são apresentados para os 12 vídeos, sendo 6 vídeos 1080p e 6 vídeos 2160p. Além da média para os 12 vídeos testados, também são apresentados os resultados de BD-rate médio para os vídeos 1080p e 2160p. Para cada grupo de vídeo é apresentado o desvio padrão ( $\sigma$ ), bem como o desvio padrão para os doze vídeos. Ainda, a última linha da tabela apresenta a redução da área de busca quando um determinado tamanho de corte é utilizado.

Várias análises podem ser feitas com os resultados da Tabela 5.3, a primeira a ser abordada refere-se aos resultados de BD-rate para os dois padrões de corte da área de busca, retangular e losangular. Nos resultados apresentados na Tabela 5.2 o padrão losangular abrange uma área com 0,5% mais acessos que o padrão retangular. Entretanto, ao observar os resultados apresentados na Tabela 5.3 nota-se que os resultados médios de

Tabela 5.3: Resultados médios da variação de BD-rate (em %) para as 9 configurações de corte na área de busca.

SR	48-R	48-L	48*	32-R	32-L	32*	16-R	16-L	16*
$\alpha_1$	0,17	0,07	0,09	0,03	0,18	0,42	0,61	0,53	5,61
$\alpha_2$	0,05	0,02	0,25	-0,01	0,01	0,14	0,11	0,15	0,27
$\alpha_3$	0,02	0,06	-0,02	0,07	0,00	0,23	0,18	0,21	0,69
$\alpha_4$	0,06	0,06	0,07	0,00	0,13	0,02	0,05	-0,03	0,07
$\alpha_5$	0,01	-0,04	-0,02	-0,06	-0,01	0,15	-0,03	0,05	0,94
$\alpha_6$	-0,01	-0,09	-0,01	0,14	0,10	0,45	0,95	0,95	6,12
1080p	0,05	0,01	0,06	0,03	0,07	0,23	0,31	0,31	2,28
$\sigma$	0,03	0,06	0,11	0,07	0,06	0,14	0,35	0,35	2,27
$\beta_1$	-0,01	0,06	0,01	0,01	0,00	-0,02	0,00	0,04	-0,13
$\beta_2$	-0,01	-0,03	0,12	-0,02	0,00	-0,02	0,01	0,16	0,21
$\beta_3$	0,29	0,25	1,27	0,67	0,71	4,58	2,75	3,13	19,73
$\beta_4$	0,18	0,27	0,42	0,60	0,78	1,67	1,81	1,75	5,40
$\beta_5$	-0,04	0,01	-0,09	-0,06	0,06	-0,01	0,91	0,89	9,47
$\beta_6$	0,17	0,11	0,31	0,46	0,49	4,64	1,68	1,82	18,21
2160p	0,10	0,11	0,34	0,28	0,34	1,81	1,19	1,30	8,82
$\sigma$	0,12	0,11	0,45	0,31	0,33	2,07	1,00	1,07	7,89
<b>Total</b>	0,07	0,06	0,20	0,15	0,21	1,02	0,75	0,80	5,55
$\sigma$	0,10	0,10	0,35	0,25	0,28	1,67	0,87	0,93	6,71
Área	30,5%	30,5%	30,5%	55,5%	55,5%	55,5%	75,0%	75,0%	75,0%

\*Redução da área de busca via configuração no HM.

Fonte: Próprio autor.

BD-rate para os dois padrões são muito semelhantes. Em alguns vídeos dependendo do tamanho do corte na área de busca o padrão losangular apresenta melhores resultados, mas mudando o tamanho do corte para o mesmo vídeo o padrão retangular tem melhores resultados. Ao observar os resultados médios para vídeos 1080p para o tamanho de SR 48 o padrão retangular (0,05%) apresenta uma variação no BD-rate  $5\times$  maior que no padrão losangular (0,01%). Entretanto, para SR 32, a variação é menor,  $2,3\times$  com o padrão retangular apresentando melhores resultados, 0,03% de BD-rate no retangular e 0,07% no losangular. Para os vídeos 1080p com SR 16 os resultados médios de BD-rate foram os mesmos, inclusive do desvio padrão.

Para os vídeos 2160p o padrão retangular apresenta melhores resultados médios de BD-rate quando comparado ao padrão losangular. Esse fato se deve ao comportamento do TZS nos vídeos 2160p, conforme pode ser observado nos histogramas de acessos das Figuras 5.1 e 5.2, anteriormente discutidas. Nos vídeos 2160p, devido a alta similaridade das amostras dentro da área de busca, as bordas são mais exploradas do que nos vídeos 1080p, levando o TZS a melhores escolhas de blocos no processo de busca. Esse resultado também pode ser observado na média para o conjunto total de vídeos, onde o padrão retangular apresentou melhores resultados inclusive de desvio padrão, mostrando que para esse conjunto de vídeos, o padrão retangular apresenta resultados com menor dispersão.

A variabilidade apresentada nos resultados de BD-rate, principalmente nos vídeos 1080p, onde dependendo do tamanho do corte na área de busca um padrão de corte apresenta melhores resultados, ocorre porque durante o processo de codificação cada CTU faz otimização local sem considerar o impacto nas próximas CTUs. Nesse cenário, uma decisão sub-ótima pode beneficiar as CTUs seguintes. Assim, gerando ganhos residuais.

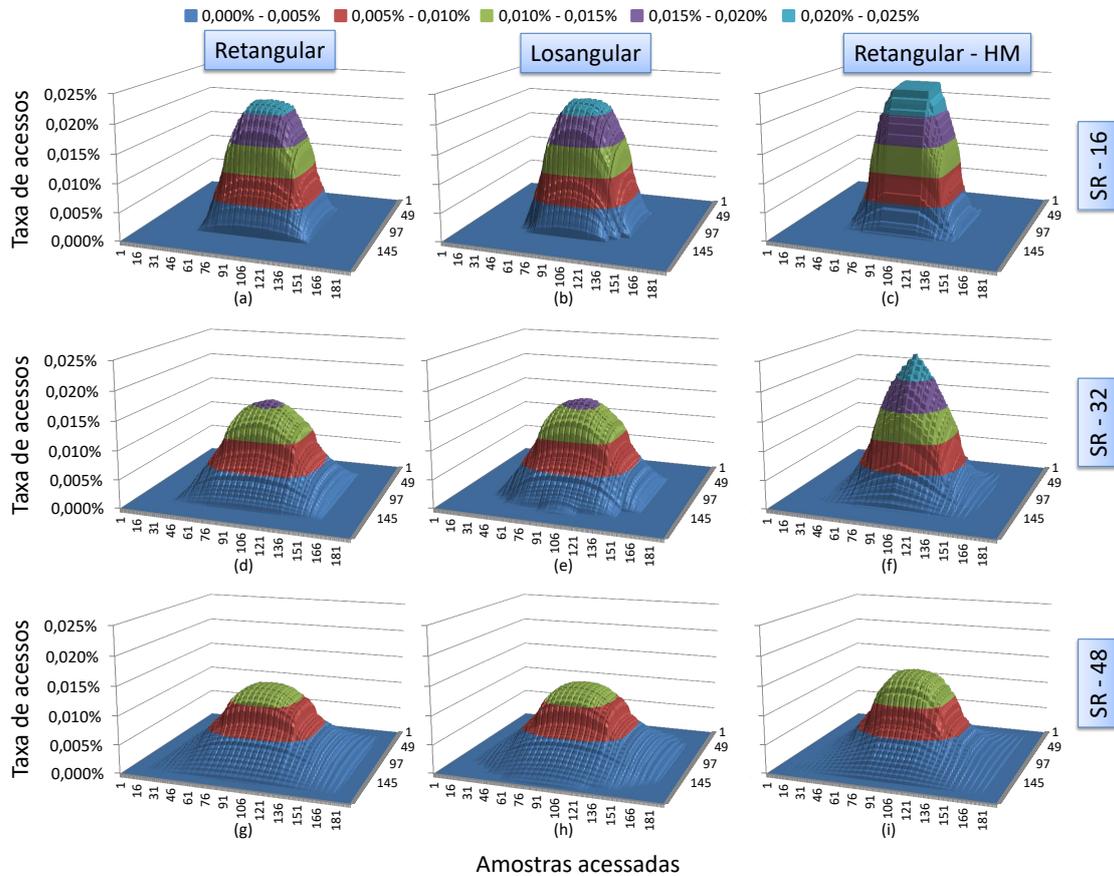
Além disso, em alguns casos, a redução da área de busca resultou em pequenos ganhos na eficiência de codificação (valores negativos de BD-rate), isso ocorre principalmente porque os maiores vetores de movimento não são gerados, uma vez que o tamanho da área de busca está reduzido. Com vetores pequenos, o impacto na taxa de bits é reduzida, mesmo com um resultado de predição um pouco pior.

Outro aspecto que pode ser observado na Tabela 5.3, refere-se aos resultados para os diferentes tamanhos de SR. As áreas com menores cortes (SR 48) possuem as menores perdas de BD-rate, em média. Isso já era esperado, uma vez que em áreas maiores o TZS tem mais possibilidades de encontrar o bloco candidato com menor distorção. Por exemplo, a configuração 48-L apresentou na média uma variação de 0,06% de BD-rate enquanto que na configuração 16-L essa variação foi de 0,8%, i.e. um valor  $13\times$  maior. Ainda, a configuração 16-L apresenta uma redução na área de busca de 75%, enquanto que na configuração 48-L esse redução é de 30,5%.

Na média para os doze vídeos, todas as áreas com cortes, losangulares ou retangulares, tiveram perdas de BD-rate inferior a 1%. Quando comparado as reduções via configuração no HM, observa-se que quando o intervalo de busca 32 é utilizado a perda em BD-rate já é superior a 1%, aumentado de forma considerável quando utiliza intervalos de busca menores. As configurações com cortes na área de busca sempre apresentam melhores resultados de BD-rate quando comparadas com a mesma redução, mas feita através da configuração do HM. Para as reduções de tamanho 48 os cortes na área de busca apresentam resultados de BD-rate  $3,3\times$  menor e para os demais tamanhos esse valor chega a  $6,8\times$  menor. Isso se deve ao fato de que quando são realizados os cortes na área de busca o TZS não sabe que a região é cortada, uma vez que o algoritmo apenas não recebe os dados fora da área delimitada e desta forma acaba explorando mais informações de bordas dentro da área delimitada.

Essa informação pode ser observada na Figura 5.5, onde são apresentados nove histogramas de acessos para o vídeo *BasketballDrive*. Esses histogramas representam a média para os 4 QPs em cada uma das configurações. Nota-se na Figura 5.5 uma grande diferença no padrão de acessos entre as configurações com cortes, losangular e

Figura 5.5: Histogramas de acessos na área de busca com a média para os 4 QPs para o vídeo HD 1080p BasketballDrive.



Fonte: Próprio autor.

retangular, e a configuração com a redução da área de busca através de configuração do HM, chamada retangular - HM. Ao observar os intervalos de busca 16 e 32, nota-se um comportamento de acesso mais centralizado quando se utiliza o retangular - HM, sem cortes, justamente pelo TZS conhecer o tamanho da área de busca. Isso acaba afetando a eficiência de codificação e o BD-rate médio para SR 32 atinge 1,02% enquanto que para SR 16 a perda é ainda maior, ficando em 5,55%, conforme dados da Tabela 5.3. Quando são utilizados os padrões de cortes losangular ou retangular essas perdas são minimizadas e os acessos são mais distribuídos dentro da área de busca, sendo que os dois padrões de cortes possuem histogramas de acessos bastante semelhantes, com pequena diferença no contorno dos acessos na base dos histogramas, onde é possível perceber a diferença entre os padrões de corte.

Com base nos resultados apresentados, se percebe que a técnica de corte da área de busca apresenta melhores resultados de eficiência de codificação quando comparada a redução da área de busca por configuração do HM. Os resultados mostram que a técnica

de cortes da área de busca pode apresentar resultados até  $6,8\times$  melhor, em média, para as configurações com SR 32 e 16, sendo que para alguns vídeos a técnica apresentou resultados  $10\times$  melhor para o mesmo SR. Além disso, comparando os resultados dos padrões de corte, losangular e retangular, nota-se uma pequena melhora na eficiência de codificação do padrão retangular em relação ao padrão losangular, principalmente nos vídeos UHD 2160p.

#### 5.4.2 *Bit dropping*

Nessa seção serão apresentados e discutidos os resultados de eficiência de codificação utilizando a técnica de *bit dropping*. A técnica de *bit dropping* refere-se à eliminação de um ou mais bits de cada amostra da área de busca. A abordagem usual é descartar os bits menos significativos (*Least Significant Bits* – LSBs), uma vez que, para imagens e vídeo, os bits mais significativos (*Most Significant Bits* – MSBs) influenciam mais a qualidade da imagem (FRUSTACI et al., 2016). Desta forma, nessa tese foram empregados cortes dos LSBs e diferentemente da técnica anterior, no *bit dropping* todas as amostras da área de busca sofrem cortes, entretanto, a quantidade de amostras da área de busca continua o mesmo, i.e. o intervalo de busca possui  $[-64, +64]$  em todas as configurações.

A Tabela 5.4 apresenta os resultados de BD-rate para as configurações de *bit dropping*. Foram testadas oito configurações de *bit dropping*, onde foram realizados cortes de bits variando de 1 bit (*7bpp*) até 8 bits (*0bpp*). Na configuração *0bpp* foram cortados todos os 8 bits das amostras, i.e. a ME não realiza a busca e o bloco colocalizado é utilizado. Estes cortes mais agressivos, acima de 4 bits de corte, foram realizados com intuito de observar o comportamento do codificador em relação a esta alta perda de informações, esperando uma perda de eficiência de codificação também bastante alta. Os experimentos foram realizados para as 12 sequências de vídeos, sendo 6 HD 1080p e 6 UHD 2160p, utilizando os quatro QPs. Os resultados de BD-rate apresentados são em relação a um intervalo de busca  $[-64, +64]$  com 8 bits por amostra.

Nota-se na Tabela 5.4 que a perda em BD-rate para os vídeos HD 1080p são sempre menores quando comparadas as perdas nos vídeos UHD 2160p para a mesma configuração. Isso ocorre devido a alta similaridade entre amostras vizinhas na área de busca dos vídeos UHD, tornando assim, vídeos com essa resolução mais sensíveis a técnicas de computação aproximada. Esse comportamento também pode ser observado na seção anterior, onde foi apresentada a técnica de cortes na área de busca, os resultados de BD-rate

Tabela 5.4: Resultados médios da variação de BD-rate para as 8 configurações de *bit dropping*.

bits por amostra	7bpp	6bpp	5bpp	4bpp	3bpp	2bpp	1bpp	0bpp
$\alpha_1$	0,062	0,190	0,856	2,765	6,932	15,242	28,409	65,392
$\alpha_2$	-0,035	0,080	0,198	0,568	1,485	4,632	19,619	49,934
$\alpha_3$	-0,008	0,000	0,124	0,912	3,308	8,225	13,812	27,469
$\alpha_4$	0,038	0,138	0,511	2,290	7,386	15,602	20,851	51,944
$\alpha_5$	0,063	0,095	0,348	1,024	2,865	8,366	9,571	32,511
$\alpha_6$	0,008	0,365	1,678	4,815	10,414	18,034	25,571	33,636
1080p	0,021	0,145	0,619	2,062	5,398	11,683	19,639	43,481
$\sigma$	0,04	0,11	0,53	1,46	3,10	4,85	6,44	13,34
$\beta_1$	0,034	0,109	0,059	0,049	0,388	0,454	0,497	0,436
$\beta_2$	1,314	1,467	2,101	4,374	10,120	21,037	31,972	58,874
$\beta_3$	0,912	1,126	1,946	5,975	18,974	55,507	86,387	157,969
$\beta_4$	0,214	0,941	3,577	9,455	19,844	30,406	36,308	57,843
$\beta_5$	0,739	0,797	1,379	4,421	12,061	23,711	36,203	70,412
$\beta_6$	0,751	1,529	3,876	9,741	22,998	42,173	56,691	98,861
2160p	0,661	0,995	2,156	5,669	14,064	28,881	41,343	74,066
$\sigma$	0,43	0,47	1,29	3,31	7,57	17,24	26,05	47,58
<b>Total</b>	0,341	0,570	1,388	3,866	9,731	20,282	30,491	58,773
$\sigma$	0,44	0,55	1,25	3,13	7,23	15,31	21,86	38,14
Red. Área	12,5%	25,0%	37,5%	50,0%	62,5%	75,0%	87,5%	100%

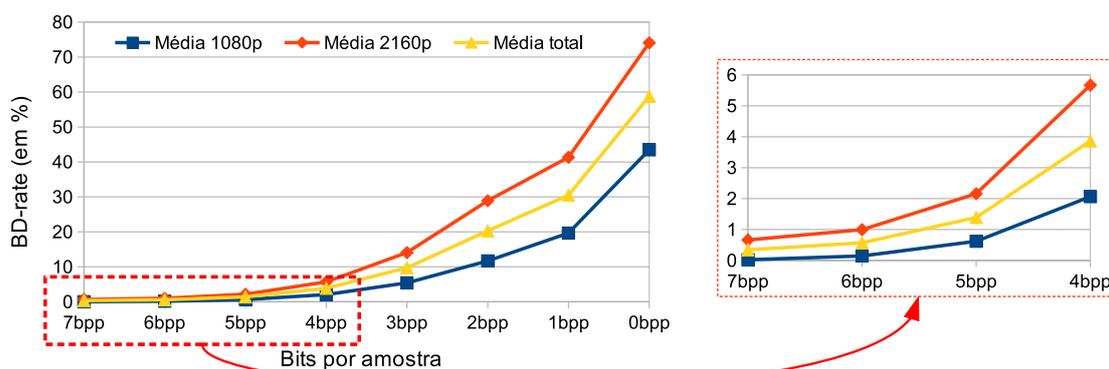
Fonte: Próprio autor.

para vídeos UHD apresentam resultados piores quando comparados aos dos vídeos HD.

A perda em BD-rate nos vídeos UHD 2160p na configuração 7bpp, onde apenas um LSB é descartado, é 30× maior, em média, quando comparado aos vídeos HD 1080p. Entretanto, essa taxa diminui quando os cortes nos LSBs aumentam. Na configuração 6bpp essa diferença cai para 6,8×, na 5bpp é de 3,4× e na 4bpp é de 2,7×. A partir desse ponto a perda em BD-rate em ambas resoluções é alta e a diferença entre elas ficam em  $\approx 2,5\times$ .

Conforme os resultados apresentados na Tabela 5.4, o BD-rate médio para os doze vídeos é inferior a 1% para as configurações 7bpp e 6bpp. Todavia, a cada bit cortado o BD-rate mais que dobra o seu valor. Esta tendência pode ser observada na Figura 5.6. Nessa figura, o gráfico da direita apresenta os resultados de BD-rate médio para os vídeos HD 1080p (curva azul), UHD 2160p (curva laranja) e para a média dos 12 vídeos (curva amarela) para todas as oito configurações testadas. No gráfico da esquerda, para uma melhor visualização, são ampliados os resultados de 4 configurações: 7bpp, 6bpp, 5bpp e 4bpp. Nota-se nesses dois gráficos, que o aumento do BD-rate de uma configuração para a outra segue a mesma tendência, independentemente da resolução, a qual afeta de forma significativa os resultados de BD-rate. Isso se deve ao fato de que com o *bit dropping* todas

Figura 5.6: Resultados de BD-rate para as configurações utilizando *bit dropping*.



Fonte: Próprio autor.

amostras da área de busca perdem informação, tanto as áreas de maior acesso quanto as de menor acesso.

Além dos resultados de BD-rate, a Tabela 5.4 apresenta os resultados de redução de área proporcionados pelo *bit dropping*, esses resultados podem ser vistos na última linha da tabela. A cada bit que é reduzido da amostra gera, de forma constante, uma redução na área de busca de 12,5%. Nesse sentido, é possível fazer uma relação BD-rate por área reduzida, onde dividindo o valor do BD-rate pela redução de área se tem a variação de BD-rate para cada 1% de redução de área para todas as configurações. Desta forma, gerando os valores para todas as configurações temos para cada 1% de área reduzida uma variação de BD-rate nos seguintes valores: *7bpp*: 0,027, *6bpp*: 0,022, *5bpp*: 0,037, *4bpp*: 0,077, *3bpp*: 0,156, *2bpp*: 0,270, *1bpp*: 0,348 e *0bpp*: 0,587. Nota-se nesses valores que as três primeiras configurações, *7bpp*, *6bpp* e *5bpp*, são as que apresentam proporcionalmente os menores valores de variação de BD-rate, sendo as mais eficientes nesse ponto. A partir da configuração *4bpp* essa taxa dobra em relação a configuração imediatamente anterior, sendo assim, as configurações menos eficientes na relação BD-rate por área reduzida.

### 5.4.3 Corte na área de busca e *bit dropping*

Nas seções anteriores foram abordadas as técnicas de cortes na área e *bit dropping*, onde foram realizados cortes na área de busca em dois padrões distintos para três tamanhos e o *bit dropping* foi realizado cortando os LSBs de 1 até 8 bits. Ambas as técnicas discutidas anteriormente apresentam altas taxas de redução de área de busca com diferen-

tes taxas de perdas em BD-rate. Desta forma, essa seção apresenta uma abordagem onde são utilizadas as duas técnicas discutidas anteriormente, com cortes na área de busca e na área restante dessa área de busca é utilizado o *bit dropping*.

Os resultados apresentados anteriormente foram considerados para definir as configurações utilizando as duas técnicas. Os cortes na área de busca utilizam o padrão retangular, uma vez que esse padrão apresentou melhores resultados de BD-rate quando comparado ao padrão losangular. Além disso, esse formato segue um padrão mais benéfico para uma implementação em hardware, pois os dados podem ser acessados sequencialmente. Os tamanhos dos intervalos de busca dos cortes foram mantidos os mesmos, sendo eles:  $[-16, +16]$  (16),  $[-32, +32]$  (32) e  $[-48, +48]$  (48). No *bit dropping*, por sua vez, foram utilizados cortes de no máximo 4 bits. Assim, foram utilizadas as configurações *7bpp*, *6bpp*, *5bpp* e *4bpp*. Com isso, a utilização conjunta das duas técnicas totaliza 12 configurações distintas, onde cada um dos três tamanhos de intervalo de busca foi testado nos 4 níveis de *bit dropping*.

A Tabela 5.5 apresenta os resultados de BD-rate para as 12 configurações. Os resultados de variação do BD-rate de cada vídeo são geradas através dos resultados de *bitrate* e PSNR para os 4 QPs e relacionados com os resultados do vídeo sem cortes na área de busca ou *bit dropping*, i.e. SR  $[-64, +64]$  com 8 bits por amostra. Nesse sentido, para obter os resultados apresentados nessa tabela foram necessárias 576 novas simulações com 100 quadros cada uma.

Como pode ser observado na Tabela 5.5 os vídeos HD 1080p responderam melhor as técnicas de computação aproximada quando comparados aos vídeos UHD 2160p, e novamente apresentaram menores taxas de BD-rate. O BD-rate médio dos vídeos HD 1080p para a configuração com o menor corte, *48-7bpp*, foi de 0,04% enquanto que para a configuração com os maiores cortes, *16-4bpp*, foi de 1,92%. Para os vídeos UHD 2160p essa diferença entre as configurações extremas foi percentualmente maior, na configuração *48-7bpp* a variação em BD-rate foi de 0,12% enquanto que na configuração *16-4bpp* foi de 5,57%. Entretanto, a diferença entre essas duas configurações foi de  $\approx 47\times$  para ambas as resoluções. Na média para os doze vídeos, a configuração *48-7bpp* apresentou um BD-rate de 0,08% enquanto que a configuração *16-4bpp* apresentou um BD-rate de 3,75%, mantendo a mesma proporção citada anteriormente.

Ao analisar os resultados entre as três áreas de corte, 48, 32 e 16, nota-se que o *bit dropping* é a técnica que mais influencia na variação do BD-rate, uma vez que as configurações com maiores cortes de bits apresentam as maiores taxas de BD-rate. Isso

Tabela 5.5: Resultados médios da variação de BD-rate para as 12 configurações da abordagem utilizando cortes na área de busca e *bit dropping*.

	48- 7bpp	48- 6bpp	48- 5bpp	48- 4bpp	32- 7bpp	32- 6bpp	32- 5bpp	32- 4bpp	16- 7bpp	16- 6bpp	16- 5bpp	16- 4bpp
$\alpha_1$	0,01	0,32	1,05	3,22	0,12	0,34	1,04	3,12	0,68	0,82	1,51	3,26
$\alpha_2$	0,18	0,19	0,45	0,84	0,09	0,13	0,56	0,94	0,31	0,12	0,51	0,84
$\alpha_3$	0,03	0,05	0,18	0,75	0,09	0,10	0,17	0,76	0,21	0,24	0,41	0,87
$\alpha_4$	0,09	0,09	0,58	2,55	0,06	0,17	0,65	2,45	0,01	0,16	0,60	2,45
$\alpha_5$	-0,10	0,02	0,24	1,06	-0,05	0,01	0,31	0,88	0,00	0,00	0,27	0,86
$\alpha_6$	0,01	0,20	0,71	2,73	0,13	0,29	0,76	2,57	1,04	1,24	1,82	3,22
1080p	0,04	0,14	0,53	1,86	0,07	0,17	0,58	1,79	0,37	0,43	0,85	1,92
$\sigma$	0,08	0,10	0,29	1,00	0,06	0,11	0,28	0,95	0,38	0,45	0,59	1,09
$\beta_1$	0,02	0,11	0,03	-0,07	-0,04	0,03	-0,03	0,01	-0,01	0,04	-0,02	-0,04
$\beta_2$	0,03	0,19	0,88	3,00	0,00	0,21	0,82	2,96	-0,07	0,18	0,83	2,73
$\beta_3$	0,31	0,42	1,33	5,13	0,57	0,84	1,72	5,38	2,72	2,86	3,73	7,30
$\beta_4$	0,24	1,10	3,41	9,35	0,68	1,43	3,95	9,65	1,64	2,35	4,54	9,90
$\beta_5$	-0,05	0,03	0,58	3,29	-0,01	0,06	0,50	2,86	0,95	1,04	1,54	4,05
$\beta_6$	0,17	0,96	3,34	8,99	0,56	1,18	3,54	9,09	1,77	2,26	4,33	9,50
2160p	0,12	0,47	1,60	4,95	0,29	0,62	1,75	4,99	1,17	1,46	2,49	5,57
$\sigma$	0,13	0,42	1,31	3,35	0,31	0,55	1,51	3,47	1,00	1,10	1,78	3,63
<b>Total</b>	0,08	0,31	1,06	3,40	0,18	0,40	1,17	3,39	0,77	0,94	1,67	3,75
$\sigma$	0,12	0,34	1,09	2,92	0,25	0,46	1,23	3,01	0,85	0,98	1,56	3,24
Área	-39%	-48%	-57%	-65%	-61%	-67%	-72%	-78%	-78%	-81%	-84%	-88%

Fonte: Próprio autor.

fica mais evidente ao observarmos as configurações *48-5bpp* e *48-4bpp* as quais possuem taxas de BD-rate mais alta que as configurações *32-7bpp*, *32-6bpp*, *16-7bpp* e *16-6bpp* as quais apresentam áreas de busca menores que as configurações com SR 48, mas possuem amostras com mais informações, 7 e 6 bits por amostra.

Esse efeito causado pelo *bit dropping* também pode ser observado nas taxas de reduções da área de busca, última linha da Tabela 5.5, onde configurações com maiores reduções na área de busca apresentam menores taxas de BD-rate. Isso pode ser visto nas configurações citadas anteriormente e também na configuração *48-6bpp* quando comparada a configuração *32-7bpp*. Nesse último caso, a configuração *32-7bpp* é melhor tanto em BD-rate quanto em redução da área de busca, sendo 0,13% melhor em BD-rate e 13% melhor em redução de área quando comparada a configuração *48-6bpp*.

No geral, nota-se nos resultados que todas configurações que utilizam cortes de 1 ou 2 bits, 7bpp e 6bpp, apresentam taxas de BD-rate menores que 1%, independentemente do tamanho do corte na área de busca. Além disso, essas configurações apresentam altas taxas de redução no tamanho da área de busca, as quais variam de 39% para a configuração *48-7bpp* até 81% na configuração *16-6bpp*. Para configurações com 3 bits de corte, *5bpp*, a taxa de BD-rate apresentou variação entre 1,06% até 1,67% para uma redução na área de busca de 57% até 84%. Já as configurações com *4bpp* as taxas de redução de área de busca são ainda maiores, variando entre 65% até 88% com variação em BD-rate entre

3,4% e 3,75%. Assim, para uma redução de 88% da área de busca, configuração *16-4bpp*, haverá um custo em BD-rate de 3,75%. Na próxima seção, será apresentado um comparativo entre esses resultados e os principais resultados de *bit dropping* e cortes na área de busca, apresentados nas seções anteriores. Desta forma, os resultados das três abordagens serão discutidos com base nos resultados de BD-rate, redução de área e quantidades de acessos às amostras da área de busca.

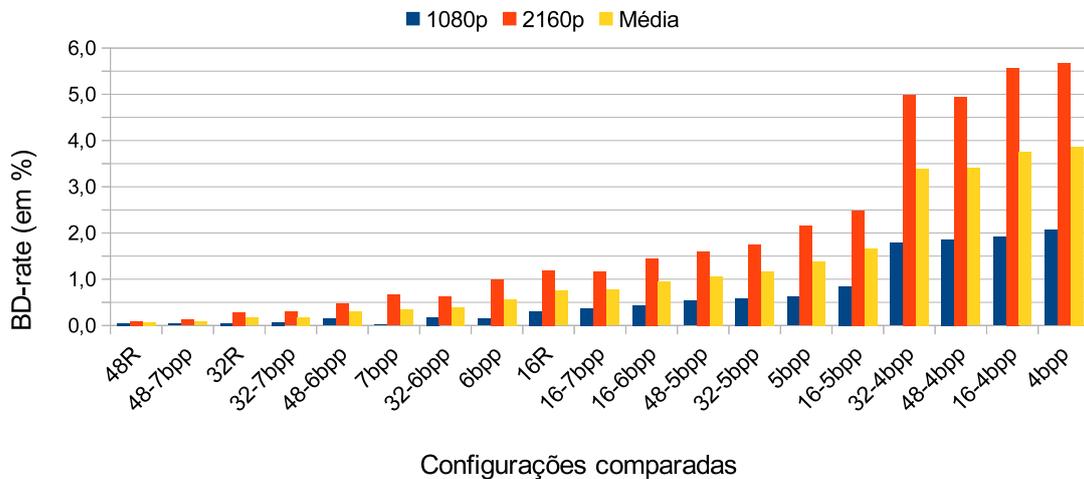
#### 5.4.4 Comparativo entre as técnicas de computação aproximada

Nessa tese, três técnicas de computação aproximada foram utilizadas, sendo elas: i) *bit dropping*, a qual utiliza cortes controlados dos LSBs; ii) cortes na área de busca, onde regiões com menos acessos são excluídas das buscas do TZS; e iii) uma abordagem onde são utilizadas as duas técnicas anteriores em conjunto, cortes na área de busca e *bit dropping* na área restante.

Nas seções anteriores estas técnicas foram apresentadas e seus resultados discutidos individualmente. Nessa seção, os resultados de BD-rate, redução da área de busca e redução de acesso às amostras da área de busca dessas três abordagens serão comparadas e discutidas. Para isso, foram selecionadas as configurações que apresentaram os melhores resultados de BD-rate e redução de área de busca. Desta forma, três configurações de cortes na área de busca foram escolhidas: intervalos de busca 48, 32 e 16 com padrão de corte retangular, quatro configurações de *bit dropping*: *7bpp*, *6bpp*, *5bpp* e *4bpp*, e as 12 configurações da abordagem que utiliza *bit dropping* e cortes na área de busca.

O gráfico da Figura 5.7 reúne os resultados de BD-rate médio (barras azuis) para os doze vídeos dessas três abordagens, totalizando 19 configurações distintas. Além dos resultados da média para os doze vídeos, são também apresentados os resultados médios para os 6 vídeos HD 1080p (barras laranjas) e para os 6 vídeos UHD 2160p (barras amarelas). Os resultados estão na ordem de apresentação das técnicas citadas anteriormente. A menor taxa de variação do BD-rate, 0,07%, observada na Figura 5.7 é encontrada na configuração *48-R*, a qual não utiliza *bit dropping*, e a maior taxa, 3,86% é encontrada na configuração *4bpp*, a qual utiliza somente *bit dropping*. Nota-se nesses dois resultados, os quais são os extremos, que utilizam apenas uma técnica de computação aproximada. Enquanto isso, as abordagens que utilizam as duas técnicas em conjunto não apresentaram nem o melhor nem o pior resultado em BD-rate, mas se distribuíram ao longo do gráfico. Os resultados do gráfico da Figura 5.7 foram apresentados nas Tabelas 5.3, 5.4 e

Figura 5.7: Comparativo de resultados BD-rate entre as três abordagens utilizando computação aproximada.



Fonte: Próprio autor.

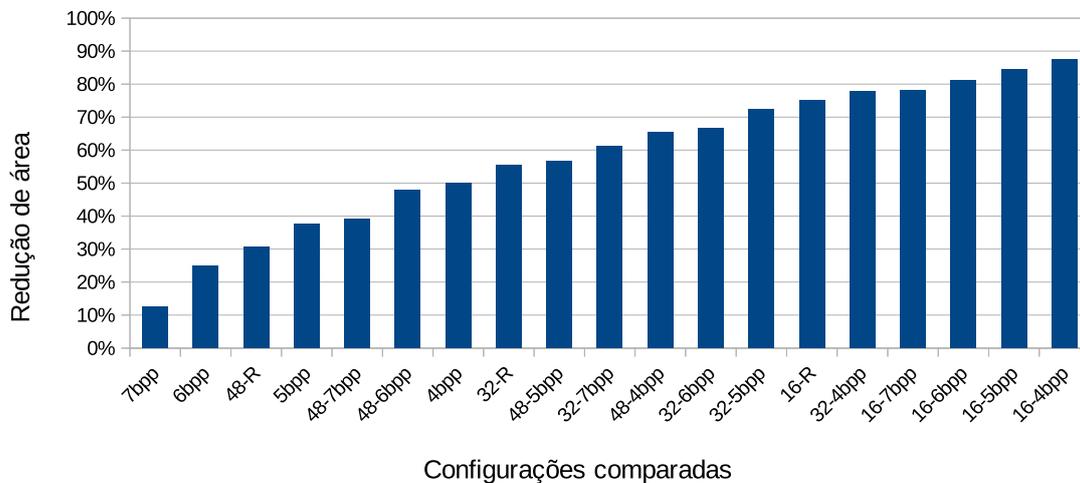
5.5 discutidas nas seções anteriores.

A menor taxa de BD-rate para a abordagem que utiliza as duas técnicas é encontrada na configuração *48-7bpp*, 0,08%, uma diferença de apenas 0,01% para a configuração *48-R*. A configuração *16-4bpp* apresentou a maior variação de BD-rate na abordagem conjunta, 3,75%. Esse BD-rate é 0,11% menor que a configuração *4bpp*, a qual também possui 4 bits por amostra, mas utiliza um intervalo de busca  $[-64, +64]$ , enquanto que a configuração *16-4bpp* utiliza um intervalo de busca  $[-16, +16]$ .

Esse efeito de melhora no resultado de BD-rate da abordagem com as duas técnicas em comparação ao *bit dropping*, pode ser observado em todas as configurações de *bit dropping*. Onde sempre haverá no mínimo uma configuração com o mesma quantidade de bits por amostra, mas com área menor que apresenta melhores taxas de BD-rate. Por exemplo, as quatro maiores taxas de BD-rate são aquelas que utilizam 4 bits por amostra, mas a maior taxa entre elas é da configuração que utiliza apenas o *bit dropping*. Desta forma, nota-se que a utilização da abordagem em conjunto consegue minimizar as perdas ocasionadas pelo *bit dropping*, uma vez que com essa técnica todas as amostras da área de busca são cortadas, diminuindo as diferenças entre as amostras vizinhas, o que acaba impactando no resultado da predição. Além disso, áreas de busca de menor tamanho geram vetores de movimento menores, reduzindo o impacto na taxa de bits, mesmo que o resultado da predição seja um pouco pior.

Conforme pode ser visto na Figura 5.7, as soluções com maiores áreas e meno-

Figura 5.8: Comparativo de resultados de redução de área entre as três abordagens utilizando computação aproximada.



Fonte: Próprio autor.

res cortes de bits apresentam as menores taxas de BD-rate, isso já era esperado uma vez que essas configurações preservam mais as informações da área de busca que as demais. Entretanto, as soluções que apresentam as maiores taxas de BD-rate são aquelas que introduzem maiores cortes nos bits das amostras, mas não necessariamente essa seja a menor área. Além disso, nota-se que 11 das 19 configurações apresentam resultados de BD-rate inferiores a 1% de variação, sendo elas: as três configurações de cortes na área de busca, duas configurações de *bit dropping* e seis configurações de corte na área de busca com *bit dropping*, todas essas configurações tem no mínimo seis bits por amostras. As configurações que utilizam cinco bits por amostra, com cortes ou não na área de busca, apresentam resultados de BD-rate entre 1% e 2%. As últimas quatro configurações apresentam resultados de BD-rate entre 3% e 4% e todas elas utilizam 4 bits por amostra. Isso reafirma o impacto do *bit dropping* nos resultados de BD-rate das configurações. Entretanto, o *bit dropping* contribui com altas taxas de redução de área.

O impacto das reduções de área nas configurações podem ser observadas na Figura 5.8. Nessa figura são apresentados os resultados de redução de área de busca para as 19 configurações testadas. A menor taxa de redução de área é encontrada na configuração 7bpps, 12,5%, onde o *bit dropping* mínimo é utilizado sendo apenas 1 bit cortado. A configuração com maior redução na área de busca utiliza corte na área de busca com intervalo de busca  $[-16, +16]$  e apenas 4 bits por amostra na área restante, 16-4bpps. Nessa configuração a redução da área de busca atinge 88%. Os resultados do gráfico da Figura 5.8 foram apresentados nas Tabelas 5.3, 5.4 e 5.5 discutidas nas seções anteriores.

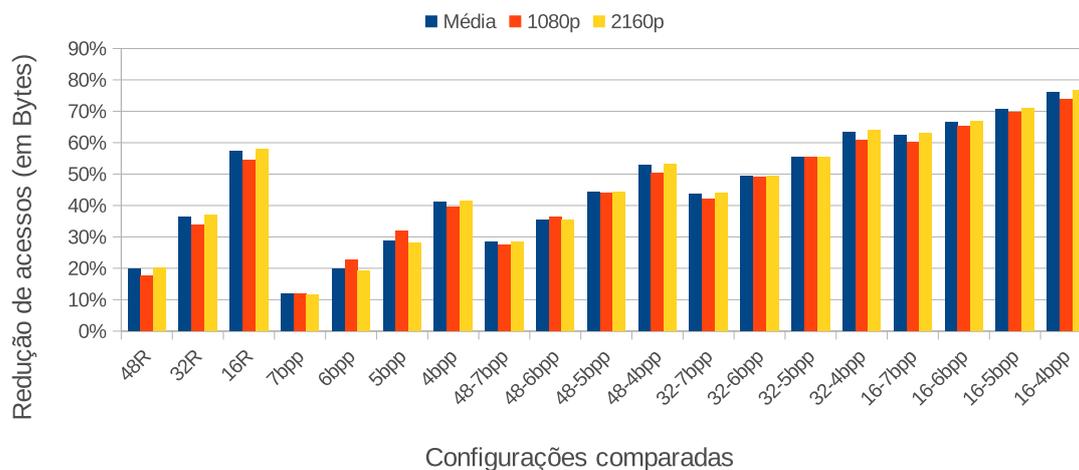
Como pode ser visto na Figura 5.8, os resultados de redução de área nas configurações que utilizam apenas *bit dropping* estão entre as menores, apresentando no máximo 50% de redução na configuração *4bpp*. Nas configurações que utilizam apenas cortes na área de busca a maior redução é encontrada na configuração *16-R*, atingindo 75%. Enquanto isso, as configurações que utilizam cortes na área de busca e *bit dropping* são as que atingem maiores taxas de redução da área de busca. Cinco dessas configurações atingem taxas superiores aos 75% da configuração *16-R*, atingindo no máximo 88% de redução. O mínimo de redução de área de busca alcançado pelas configurações que utilizam as duas técnicas em conjunto é de 39% na configuração *48-7bpp*. Desta forma, as abordagens que utilizam as duas técnicas em conjunto possibilitam maiores taxas de redução da área da busca, atingindo melhores resultados de BD-rate quando comparadas a técnica de *bit dropping* somente.

Entretanto, os resultados de redução de área e BD-rate podem não ser suficientes para definir as melhores configurações a serem utilizadas no sistema de redução de energia de memória, principalmente para configurações com resultados bastante semelhantes. Visto que essas alterações na área de busca, pode levar a uma maior quantidade de acessos por parte do TZS, mesmo tendo seu tamanho reduzido. Isso ocorre devido a utilização de técnicas de computação aproximada, as quais alteram o processo de busca e também a quantidade de acessos realizadas pelo TZS, influenciando na eficiência de codificação dos vídeos. Desta forma, outra métrica que deve ser considerada nesse comparativo é a quantidade de acessos realizados pelo TZS durante o processo codificação. Nesse sentido, a Figura 5.9 apresenta os resultados de acessos para todas as configurações discutidas nessa seção.

A Figura 5.9 apresenta os resultados comparativos de redução de acessos, nas respectivas áreas de busca para as 19 configurações. Essa redução é em relação a área de busca  $64 \times 64$  com 8 bits por amostra. Os resultados de redução de acessos foram obtidos através do trace de memória implementado no software HM e representam a média dos 4 QPs de cada conjunto de vídeo, sendo as barras em laranja para os vídeos HD 1080p, as barras amarelas para os vídeos UHD 2160p e as barras azuis representam a média para esses dois conjuntos de vídeos. Esses resultados são apresentados em bytes, uma vez que se tem amostras de diferentes tamanhos, variando de 8 bits até 4 bits. Os resultados detalhados de cada uma das configurações apresentadas no gráfico da Figura 5.9 são relacionados nas Tabelas B.1 – B.5 do Apêndice B.

Nota-se na Figura 5.9 que as configurações que apresentam os maiores cortes,

Figura 5.9: Resultados comparativos de redução de acessos entre as três abordagens utilizando computação aproximada.



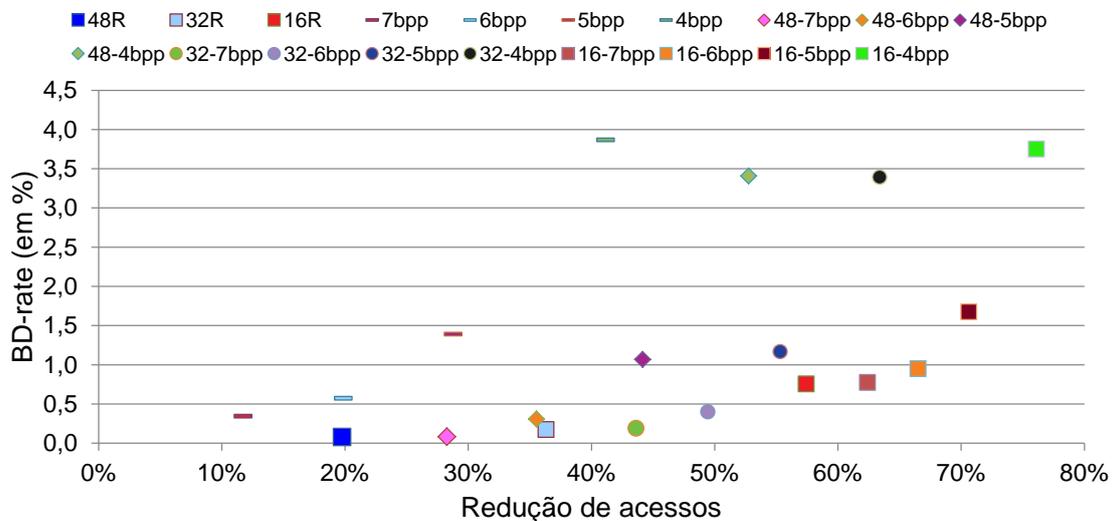
Fonte: Próprio autor.

menor área de busca e maior corte em bits, também são aquelas que apresentam as maiores reduções nos acessos. Esse era um resultado esperado, uma vez que quanto menor o corte mais próximo estará da configuração *baseline*, com intervalo de busca  $64 \times 64$  e 8 bits por amostra. A menor redução em acessos foi obtida na configuração *7bpp*, 11,7%, enquanto que a maior redução encontra-se na configuração *16-4bpp* com 76,6%, sendo estas configurações os extremos também na redução de área.

Entretanto, ao relacionar os resultados de acessos (Figura 5.9) com os resultados de redução de área (Figura 5.8), percebe-se que algumas técnicas apresentam resultados de redução de acessos próximos aos resultados de redução de área. Esse é o caso do *bit dropping*, onde para a configuração *7bpp* os acessos são reduzidos em 11,7%, enquanto que a área para essa mesma configuração é reduzida em 12,5%, apenas 0,8% de diferença. Quando analisa-se a configuração com cortes mais agressivos, *4bpp*, a diferença aumenta, gerando uma redução de acessos de 41,2% em área que teve uma redução de 50%, mas mesmo assim continua uma diferença pequena, 8,8%.

Quando avaliados os resultados da abordagem de corte na área de busca, percebe-se que nestas configurações a redução em acessos é menos agressiva quando comparada com as reduções em área. Na configuração *48R* a redução de acessos média foi de 19,8%, enquanto que a redução de área foi de 30,6%. Para a configuração *16R* a redução de acessos média foi de 57,5%, enquanto que a redução de área foi de 75%, resultando em uma diferença de 17,5%. Essa grande diferença entre os resultados de redução de área e acessos, ocorre porque as regiões que foram cortadas foram aquelas em que se tinha as

Figura 5.10: Frente de Pareto com resultados de BD-rate e redução de acessos para as três abordagens utilizando computação aproximada.



Fonte: Próprio autor.

menores taxas de acesso.

Para as configurações que utilizam cortes na área de busca e *bit dropping* a menor taxa de redução de acessos é encontrada na configuração *48-7bpp*, 28,3%, e a maior taxa na configuração *16-4bpp*, 76,1%. Estas configurações também são as que apresentam menor e maior redução de área, respectivamente. Além disso, algumas configurações apresentaram maior redução de acessos mesmo com maior área de busca. Por exemplo, a configuração *48-5bpp* apresenta maior redução de acessos que a *32-7bpp* mesmo com maior área de busca. Isso também acontece em outros dois casos, são eles: entre as configurações *48-4bpp* e *32-6bpp*, e entre as configurações *32-4bpp* e *16-7bpp*. O fato comum entre esses pares de configurações é que com maior *bit dropping* ocorre maior redução de acessos.

Os resultados de acessos também devem ser relacionados com os resultados de BD-rate, uma vez que essa métrica refere-se diretamente a eficiência de codificação. Além disso, para a escolha das melhores configurações, estes dois resultados são importantes, sendo que as melhores configurações devem apresentar as menores variações em BD-rate com maior redução de acessos. Desta forma, para combinar esses dois resultados foi gerado uma frente de Pareto, a qual pode ser observada na Figura 5.10.

A Figura 5.10 apresenta uma frente de Pareto formada com resultados da variação de BD-rate e com os resultados de redução de acessos às amostras da área de busca. Segundo Ghosh e Dehuri (2004), uma frente de Pareto é composta por pontos que sejam

eficientes à Pareto, isto é, pontos ótimos de operação em um problema multiobjetivo. O conceito consiste no fato de que a solução de um problema de otimização multiobjetivo não é normalmente um único valor, mas um conjunto de valores denominado *frente de Pareto*. Nesse contexto, os objetivos seriam reduzir os acessos às amostras da área de busca com o mínimo de impacto em BD-rate.

A partir das informações no gráfico da Figura 5.10, é possível combinar essas duas informações, de forma que seja possível encontrar uma frente de Pareto composta por algumas configurações. De acordo com a Figura 5.10, as configurações que compõem a frente de Pareto são aquelas que estão mais à direita (maior redução de acessos) e mais abaixo (menor BD-rate) no gráfico. Desta forma, fazem parte da frente de Pareto as seguintes configurações: *48R*, *48-7bpp*, *32R*, *32-7bpp*, *32-6bpp*, *16-7bpp*, *16-6bpp*, *16-5bpp* e *16-4bpp*. Dessas nove configurações que compõem a frente de Pareto, a configuração *48R* é a que apresenta o menor BD-rate, enquanto que a configuração *16-4bpp* é a que apresenta a maior redução de acessos às amostras da área de busca. Considerando-se ambos objetivos, nenhuma das configurações é ótima, neste caso, não se pode afirmar que uma é melhor que a outra. Além disso, ainda existem as configurações que representam uma situação intermediária. Dessa forma, existem configurações que fazem parte da frente de Pareto, na qual um ponto ou outro será melhor ou pior, dependendo da métrica adotada.

Além disso, conforme pode ser observado na Figura 5.10 existem dez configurações que não pertencem à frente de Pareto. A razão pela qual estes pontos não fazem parte da frente de Pareto é porque, existe no mínimo uma configuração com melhores resultados de BD-rate e redução de acessos às amostras da área de busca. Desta forma, essas configurações são dominadas pelas configurações que fazem parte da frente de Pareto.

O conjunto de configurações gerado pela frente de Pareto pode ser reduzido ainda mais, uma vez que algumas configurações apresentam um ganho muito pequeno em um dos resultados, mas com uma perda grande no outro resultado, quando comparado a outra configuração. Nesse sentido, a configuração *48R* quando comparada a configuração *48-7bpp* apresenta uma taxa de BD-rate 0,01% melhor, enquanto que a taxa de redução de acessos é 8,5% pior. Outro caso, é a configuração *32R*, a qual apresenta resultado de BD-rate 0,02% melhor que a configuração *32-7bpp*, mas com taxa de redução de acessos 7,3% pior.

Além dessas configurações, outro subconjunto que pode ser reduzido é aquele formado pelas configurações *32-6bpp*, *16-7bpp* e *16-6bpp*. A configuração *32-6bpp* apre-

senda uma variação de BD-rate de 0,4% com 49,5% de redução de acessos, na configuração *16-7bpp* a variação de BD-rate é de 0,77% e 62,4% de redução de acessos. Essas duas configurações quando comparadas à *16-6bpp* apresentam melhores resultados de BD-rate. Porém, a configuração *16-6bpp* apresenta resultados de reduções de acessos 17% melhor que a configuração *32-6bpp*, e 4,1% melhor que a configuração *16-7bpp* para uma variação em BD-rate de apenas 0,94%.

Quando comparadas as configurações *16-6bpp* e *16-5bpp*, nota-se que a configuração *16-5bpp* aumenta a redução de acessos em 4,1%. Entretanto, o BD-rate passa de 0,94% para 1,67%, sendo esse um aumento considerável para uma melhoria na redução de acessos de apenas 4,1%. A configuração *16-4bpp* é aquela que apresenta a maior variação em BD-rate, 3,75%, porém, é a configuração que apresenta a maior redução de acessos às amostras da área de busca, 76,1%.

Considerando esse refinamento no conjunto de configurações gerado pela frente de Pareto, chegou-se em conjunto de 4 configurações: *48-7bpp*, *32-7bpp*, *16-7bpp* e *16-4bpp*, as quais apresentam diferentes taxas de variação em BD-rate e redução de acessos às amostras da área de busca. Desta forma, o sistema proposto nessa tese poderá atender diferentes níveis de eficiência de codificação e reduções de acessos, os quais impactam diretamente no consumo de energia.

## 5.5 Considerações finais do capítulo

Nesse capítulo foram apresentadas análises sobre a distribuição dos acessos à área de busca realizadas pelo algoritmo de busca TZS. Estas análises possibilitaram um melhor entendimento sobre o comportamento dos acessos realizados pelo TZS dentro de uma área de busca e o conhecimento das regiões mais e menos acessadas. Para gerar os dados de acessos às amostras da área de busca o software de referência do HEVC, o HM, foi modificado, possibilitando a extração de informações referentes as PUs processadas. Desta forma, um arquivo de *trace* para cada simulação foi gerado e, através desses *traces* foi possível contabilizar todos os acessos feitos às amostras durante o processo de codificação.

O conjunto de testes utilizado nos experimentos é composto por doze sequências de vídeo, sendo seis vídeos HD 1080p e seis vídeos UHD 2160p. Além disso, todas as simulações foram realizadas para os quatro QPs 22, 27, 32 e 37, utilizando 100 quadros de cada vídeo. Porém, antes da apresentação dos dados de acessos, foi definido o intervalo

de busca utilizado na configuração *baseline* dos experimentos. Nesse sentido, o intervalo de busca  $64 \times 64$  apresentou o melhor custo benefício entre *bitrate*, PSNR e tamanho da área de busca em relação aos outros tamanhos. Assim, esse intervalo foi definido para a configuração *baseline*, a qual foi utilizada na extração dos *traces* e como referência para os resultados de computação aproximada.

A partir dessas análises, foram exploradas técnicas de computação aproximada para a área de busca do algoritmo TZS. As técnicas implementadas se beneficiam do comportamento de busca do algoritmo, o qual realiza muito mais acessos às amostras na região central da área de busca do que nas bordas. Nesse sentido, três técnicas de computação aproximada foram exploradas nessa tese até o momento, sendo elas: i) cortes na área de busca, a qual restringe os acessos as regiões da área de busca pouco acessadas, i.e. as bordas da área de busca. ii) A segunda é a utilização de *bit dropping* nas amostras da área de busca. Desta forma, os bits menos significativos das amostras são deliberadamente cortados. iii) Além disso, após análises dos resultados, as melhores configurações dessas técnicas foram unidas, formando uma terceira abordagem, na qual são utilizadas ao mesmo tempo cortes de regiões da área de busca e *bit dropping* na área remanescente. Com a utilização das técnicas de computação aproximada foi possível atingir altas taxas de redução no acesso às amostras da área de busca com o mínimo de impacto na eficiência de codificação.

Essas três técnicas permitiram a utilização de 26 configurações distintas envolvendo computação aproximada, sendo: 6 configurações para cortes na área de busca, 8 configurações de *bit dropping* e 12 configurações envolvendo as configurações que utilizam cortes na área de busca e *bit dropping*. Todas essas configurações foram simuladas para os 12 vídeos e cada vídeo para os 4 QPs, totalizando 1248 simulações com 100 quadros cada uma.

Para definir as melhores configurações, as quais serão integrados ao sistema de redução de energia da memória, foram avaliadas as 19 configurações mais promissoras entre as 3 técnicas exploradas. Para essas configurações foram gerados os *traces* de acessos às amostras para todos os vídeos e QPs. Neste processo de avaliação foi utilizada uma frente de Pareto, uma vez que dois resultados deveriam ser avaliados: o BD-rate e a redução de acessos às amostras da área de busca.

Após um refinamento no conjunto de configurações gerado pela frente de Pareto, chegou-se em conjunto composto por 4 configurações: *48-7bpp*, *32-7bpp*, *16-7bpp* e *16-4bpp*, as quais apresentam taxas de BD-rate que variam de 0,08% até 3,75% e taxas de

redução de acessos às amostras da área de busca que variam de 28,3% até 76%. Desta forma, o sistema proposto nessa tese poderá atender diferentes níveis de eficiência de codificação e reduções de acessos, os quais impactam diretamente no consumo de energia. No próximo capítulo o sistema de redução de energia será apresentado em detalhes.

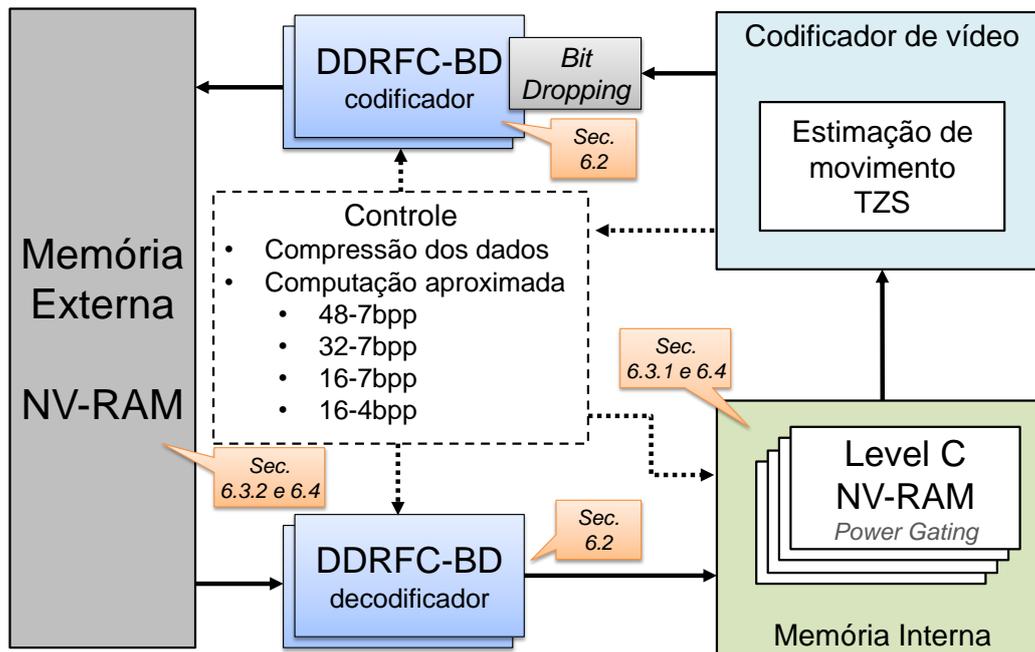
## 6 SISTEMA DE REDUÇÃO DE ENERGIA DA MEMÓRIA

Nessa tese, foi desenvolvido um sistema intitulado *Memory Energy Reduction System* (MERSy). Esse sistema possibilita a utilização conjunta de diferentes técnicas para atingir alta redução de consumo de energia em codificadores de vídeo. Para o desenvolvimento do MERSy foram exploradas, desenvolvidas e avaliadas técnicas de reuso de dados com memórias *scratchpad*, utilizando memórias de diferentes tamanhos e tecnologias (voláteis e não voláteis), compressão de quadros de referência e computação aproximada através de *bit dropping*, *power gating* e redução da área de busca. Além das várias técnicas que são utilizadas em conjunto, o sistema utiliza quatro níveis de configuração, *48-7bpp*, *32-7bpp*, *16-7bpp* e *16-4bpp*, as quais foram definidas no capítulo anterior, atingindo alta redução em consumo energético com baixo custo computacional e baixo impacto em eficiência de codificação.

A Figura 6.1 apresenta o diagrama de blocos em alto nível com a solução desenvolvida. Esse sistema é composto por eficientes memórias internas e externas, as quais utilizam diferentes tecnologias. Além disso, nas memórias internas é utilizada a técnica de reuso de dados Level C (TUAN; CHANG; JEN, 2002), a qual permite a reutilização de amostras de blocos vizinhos. Ainda, conforme discutido no capítulo anterior, a área de busca teve seu tamanho reduzido conforme a configuração utilizada. Com isso os bancos de memória interna podem ser desligados através da técnica de *power gating*, reduzindo ainda mais o consumo de energia relacionado às memórias internas. O controle do MERSy é o responsável por informar aos módulos qual das configurações será utilizada. Além disso, ele mantém a informação sobre o início da área de busca na memória interna. A discussão sobre a organização das memórias internas e externas é apresentada na Seção 6.3. A Seção 6.4 apresenta as tecnologias de memória avaliadas e utilizadas pelo MERSy.

Além da utilização de memórias com tecnologia NV-RAM, reuso de dados e *power gating*, o MERSy também utiliza um codec (compressor/descompressor) para os quadros de referência. A utilização desse codec tem grande importância nesse sistema, uma vez que as NV-RAMs apresentam restrições na escrita como: alta latência, alto consumo de energia e quantidade de escrita nas células limitadas, *a.k.a write endurance*. Desta forma, a compressão dos dados a serem escritos na memória externa trazem melhorias no consumo de energia, tempo de acesso e tempo de vida do sistema. O artigo publicado em Silveira et al. (2019) apresenta o algoritmo base utilizado nesse processo de

Figura 6.1: MERSy: sistema de redução de energia da memória em codificadores de vídeo



Fonte: Próprio autor.

compressão, o qual foi adaptado e a técnica de bit dropping foi integrada no compressor de quadros de referência. A Seção 6.2 apresenta em detalhes o algoritmo de compressão de quadros de referência, as arquiteturas de hardware desenvolvidas e os resultados de síntese lógica dessas arquiteturas. Além dessas seções, as quais apresentam em detalhes o funcionamento do MERSy, a Seção 6.1 apresenta o modelo de consumo de energia desenvolvido para calcular a redução total de consumo de energia da memória gerada pelo MERSy.

### 6.1 Modelagem do consumo de energia

Esta seção apresenta o modelo de consumo de energia desenvolvido para calcular a redução total de consumo de energia da memória gerada pelo sistema proposto. Este modelo é capaz de calcular o consumo total de energia dos sistemas de codificação de vídeo com diferentes configurações, como estratégias de reutilização de dados, hierarquias de memória e compressores de quadro de referência. A entrada deste modelo é o *trace* do algoritmo TZS, com informações detalhadas de todos os blocos solicitados durante a execução do TZS. A saída do modelo consiste no consumo de energia das soluções avaliadas.

As próximas equações são utilizadas pelo modelo para estimar o consumo de ener-

gia dinâmica do sistema proposto. As Equações 6.1 e 6.2 apresentam as fórmulas utilizadas para estimar o consumo de energia dinâmico das operações de leitura e escrita, respectivamente, da memória externa. Nas equações é estimado o consumo de energia ( $CE$ ) relacionado à memória externa ( $MemE$ ) para operações de leitura ( $Re$ ) e escrita ( $Wr$ ) para uma quantidade de palavras ( $W$ ) sem reuso de dados, e  $E$  é a energia para uma operação de leitura ou escrita em uma memória específica.

$$CE_{MemE-Re} = E(MemE_{Re}) \times W(Algoritmo_{TZS}) \quad (6.1)$$

$$CE_{MemE-Wr} = E(MemE_{Wr}) \times W(Quadro) \quad (6.2)$$

Nas Equações 6.1 e 6.2 o dado  $Algoritmo_{TZS}$  refere-se a quantidade de acessos realizados pelo algoritmo de busca usado na ME, TZS nesse caso.  $Quadro$  é o tamanho do quadro de referência (e.g.,  $1920 \times 1080$  pixels). A Equação 6.3 apresenta o cálculo para o consumo de energia total, quando não é utilizado reuso de dados. Esse resultados é a soma do consumo de energia gasto nas operações de leitura (Equação 6.1) e do consumo de energia gasto nas operações de escrita (Equação 6.2).

$$CE_{MemE} = CE_{MemE-Re} + CE_{MemE-Wr} \quad (6.3)$$

Para estimar o consumo de energia total quando é utilizado reuso de dados, deve-se considerar as operações de leitura e escrita nas memórias externa e internas. Desta forma, a Equação 6.4 determina o consumo de energia das operações de leitura da memória externa, considerando a quantidade de acessos do reuso  $Level_C$ ,  $W(Level_C)$ .

$$CE_{MemE-RD-Re} = E(MemE_{Re}) \times W(Level_C) \quad (6.4)$$

A Equação 6.5 define o consumo de energia relacionado as operações de escrita na memória interna ( $MemI$ ), considerando a quantidade de acessos do reuso de dados  $Level_C$ . Enquanto que a Equação 6.6 apresenta o cálculo para o consumo de energia relacionado as operações de leitura na memória interna, considerando os acessos realizados pelo TZS.

$$CE_{MemI-Wr} = E(MemI_{Wr}) \times W(Level_C) \quad (6.5)$$

$$CE_{MemI-Re} = E(MemI_{Re}) \times W(Algoritmo_{TZS}) \quad (6.6)$$

A Equação 6.7 define o consumo de energia total quando o reuso de dados Level C é considerado. Desta forma, essa equação é a soma dos resultados de consumo de energia para ler da memória externa as informações do quadro de referência (Equação 6.4), escrever essas informações na memória interna (Equação 6.5), ler da memória interna (Equação 6.6) para processar a informação e escrever o quadro reconstruído na memória externa (Equação 6.2).

$$CE_{RD} = CE_{MemE-RD-Re} + CE_{MemI-Wr} + CE_{MemI-Re} + CE_{MemE-Wr} \quad (6.7)$$

A seguir são apresentadas as equações para cálculo do consumo de energia para uma configuração que utilize reuso de dados e compressão de quadros de referência. As Equações 6.8, 6.9 e 6.10 apresentam os cálculos de consumo de energia para enviar o quadro reconstruído para a memória externa. A Equação 6.8 define o consumo de energia gasto para comprimir o quadro reconstruído, onde  $E(RFC_{Enc})$  é a energia gasta para comprimir uma palavra ( $W$ ) do quadro reconstruído ( $Quadro$ ).

$$CE_{RFC-Enc} = E(RFC_{Enc}) \times W(Quadro) \quad (6.8)$$

Na Equação 6.9 é apresentado o cálculo do consumo de energia para escrever os dados comprimidos pelo compressor na memória externa, onde  $E(MemE_{Wr})$  é a energia gasta para escrever uma palavra na memória externa e  $W(QuadroCod)$  é a quantidade de palavras a serem escritas na memória externa.

$$CE_{MemE-RFC-Wr} = E(MemE_{Wr}) \times W(QuadroCod) \quad (6.9)$$

O gasto total de energia para escrever na memória externa, considerando as operações de escrita e compressão do quadro reconstruído é definido pela Equação 6.10. Nessa equação, o consumo de energia gasto para comprimir o quadro reconstruído,  $CE_{RFC-Enc}$ , é apresentado na Equação 6.8, e o consumo de energia gasto para escrever o quadro comprimido na memória externa,  $CE_{MemE-RFC-Wr}$ , é dado pela Equação 6.9.

$$CE_{Sistema-Wr} = CE_{RFC-Enc} + CE_{MemE-RFC-Wr} \quad (6.10)$$

As próximas equações definem os cálculos utilizados para estimar o consumo de energia relacionado à leitura do quadro de referência da memória externa, decodificação

e armazenamento na memória interna. A Equação 6.11 define o consumo de energia para ler os dados referentes ao quadro de referência da memória externa, onde  $E(MemE_{Re})$  é a energia gasta para ler uma palavra da memória externa e  $W(RFC_{Dec})$  é a quantidade de palavras a serem lidas.

$$CE_{MemE-RFC-Re} = E(MemE_{Re}) \times W(RFC_{Dec}) \quad (6.11)$$

A Equação 6.12 define o consumo de energia para decodificar toda a informação vinda da memória externa. Nessa equação,  $E(RFC_{Dec})$  é o consumo de energia gasto para decodificar uma palavra pelo compressor de quadros de referência e  $W(Level_C)$  é a quantidade de palavras necessárias para o reuso de dados. O consumo de energia para escrever os quadros de referência na memória interna e, posteriormente, esses dados serem lidos pelo algoritmo de busca já foram definidos pelas Equações 6.5 e 6.6, respectivamente.

$$CE_{RFC-Dec} = E(RFC_{Dec}) \times W(Level_C) \quad (6.12)$$

Desta forma, a Equação 6.13 define o gasto em consumo de energia para ler o quadro comprimido na memória externa ( $CE_{MemE-RFC-Re}$ ), decodificar a informação pelo compressor de quadros de referência ( $CE_{RFC-Dec}$ ), armazenar na memória interna ( $CE_{MemI-Wr}$ ) e serem lidos ( $CE_{MemI-Re}$ ) pelo algoritmo de busca TZS durante o processamento do vídeo.

$$CE_{Sistema-Re} = CE_{MemE-RFC-Re} + CE_{RFC-Dec} + CE_{MemI-Wr} + CE_{MemI-Re} \quad (6.13)$$

O gasto total de consumo de energia relacionada à memória, utilizando compressão de quadros de referência e reuso de dados é definido na Equação 6.14. Nessa equação é realizada a soma de todas operações envolvendo a leitura e escrita dos dados. Nota-se que esse modelo pode ser empregado para qualquer tecnologia de memória e qualquer tipo de compressor de quadros de referência que o sistema utilize.

$$CE_{Sistema} = CE_{Sistema-Re} + CE_{Sistema-Wr} \quad (6.14)$$

## 6.2 Compressor de quadros de referência com *bit dropping*

Com intuito de reduzir ao máximo os acessos à memória externa do codificador, este trabalho propõe a utilização de um algoritmo para compressão de quadros de referência, o qual utiliza a técnica de *bit dropping* discutida na seção 5.4.2. O algoritmo de compressão utiliza ainda uma dupla codificação diferencial, aliada a uma codificação de entropia eficiente. A codificação de entropia utiliza códigos de tamanho variável (VLC) que ficam armazenados em uma tabela estática. As próximas seções apresentam o algoritmo, com o processo de codificação e decodificação das amostras, e as arquiteturas de hardware propostas para esse algoritmo de compressão.

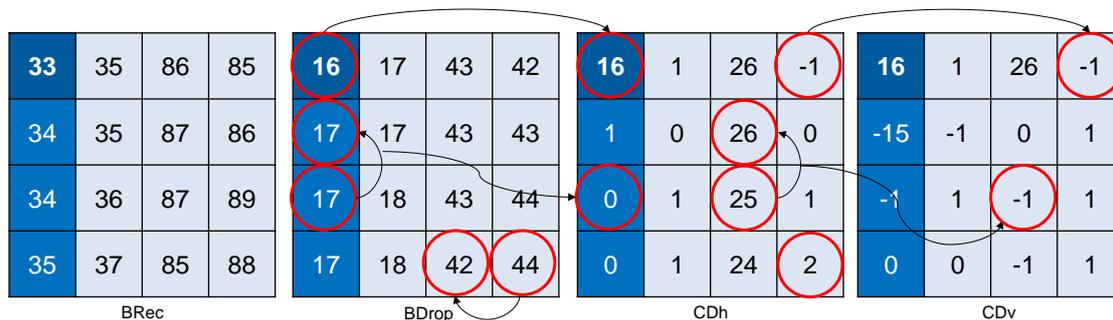
### 6.2.1 Algoritmo DDRFC-BD

O algoritmo proposto foi intitulado *Double Differential Reference Frame Compressor with Bit Dropping* - (DDRFC-BD), o qual é uma evolução de um trabalho anterior publicado pelo autor em Silveira et al. (2019). O DDRFC-BD realiza o corte de LSBs de cada amostra do bloco  $8 \times 8$  a ser comprimido, em seguida é aplicada uma dupla codificação diferencial, horizontal e vertical, e os resíduos gerados são codificados através de uma codificação de *Huffman* que usa códigos VLC em uma tabela estática. O DDRFC-BD garante acesso aleatório em nível de bloco, evitando dependências de dados entre blocos vizinhos.

O *bit dropping* é a primeira técnica a ser aplicada no bloco reconstruído, onde em cada uma das amostras do bloco é cortado 1 bit ou 4 bits LSB dependendo da configuração utilizada. Conforme discutido anteriormente, as 4 configurações utilizadas são: *48R-7bpp*, *32R-7bpp*, *16R-7bpp*, todas essas com corte de 1 bit, e *16R-4bpp* com corte de 4 bits.

A Figura 6.2 apresenta um exemplo de *bit dropping* de 1 bit seguido pelo processo de dupla codificação diferencial. Nesse figura, *BRec* representa o bloco reconstruído pela ME que será enviado para a memória e *BDrop* é o bloco após o processo de *bit dropping* de 1 bit. Nota-se que, desta forma, é realizada uma divisão inteira das amostras, diminuindo pela metade, para esse caso, o intervalo das amostras. Antes desse *bit dropping*, o intervalo das amostras era de 0 até 255, com o *bit dropping* de 1 bit esse intervalo é reduzido pela metade, indo de 0 até 127. Para o caso de *bit dropping* de 4 bits, os valores das amostras variam apenas de 0 até 15, uma redução de 93,75% na representatividade

Figura 6.2: Exemplo de *bit dropping* com deslocamento de 1 bit e do processo da dupla codificação diferencial.



Fonte: Próprio autor.

nos valores das amostras.

O *bit dropping* posto desta forma gera benefícios para o processo de compressão, uma vez que os resíduos gerados também terão intervalos menores, as operações de soma e subtração poderão ser feitas utilizando menos recursos de hardware, as tabelas estáticas de Huffman, usada na codificação de entropia, poderão ser menores e o espaço de armazenamento das amostras será menor também. Entretanto, nota-se que os valores das amostras entre os blocos de exemplo *BRec* e *BDrop* ficam bastante distintos. Assim, os LSBs deverão retornar às amostras quando forem utilizadas pelo TZS no processo de codificação de vídeo, retornando aos 8 bits do tamanho original.

Após o *bit dropping* dos LSBs, inicia-se o processo da dupla codificação diferencial (DCD), o qual é composto por duas codificações diferenciais em sequência, codificação diferencial horizontal (*CDh*) e codificação diferencial vertical (*CDv*). Essa nova abordagem para codificação diferencial acabou gerando melhores resultados quando comparado a outros modos de predição, onde são utilizadas codificações diferenciais simples em algum sentido. Isso ocorre porque a DCD codifica tanto as bordas horizontais quanto as bordas verticais em um mesmo quadro, sendo esses os sentidos de borda predominante nos vídeos. Além disso, essa abordagem não insere informação lateral para identificar o sentido de codificação utilizado, nem gera custo computacional extra para avaliar qual o melhor modo utilizado.

Desta forma, este processo reduz os altos valores de resíduos que podem ter sido gerados por bordas verticais dentro do bloco. A Figura 6.2 apresenta um exemplo do processo de dupla codificação diferencial, onde *CDh* representa o bloco processado após a primeira codificação diferencial, no sentido horizontal, e *CDv* é o bloco após a segunda codificação diferencial, no sentido vertical. Após, o bloco *CDv* é então enviado à

codificação de entropia.

A Equação 6.15 define o processo da  $CDh$  em três casos. Esses três casos também são ilustrados na Figura 6.2. O primeiro caso ocorre quando a primeira amostra do bloco é processada,  $i = 0$  e  $j = 0$ , onde  $i$  e  $j$  denotam as linhas e colunas do  $BDrop$ , respectivamente. Esta amostra é simplesmente copiada para o bloco residual, número 16 na Figura 6.2. O segundo caso é a codificação da primeira coluna do bloco,  $i > 0$  e  $j = 0$ , onde o resíduo é obtido subtraindo a posição  $(i - 1, j)$  de  $(i, j)$  do bloco. O terceiro caso é a codificação das linhas do bloco, onde  $j > 0$ . Nesse caso, o resíduo é obtido subtraindo a posição  $(i, j - 1)$  de  $(i, j)$ .

$$\begin{cases} CDh_{i,j} = BDrop_{i,j}, & \text{se } i, j = 0 \\ CDh_{i,j} = BDrop_{i,j} - BDrop_{i-1,j}, & \text{se } i > 0 \text{ e } j = 0 \\ CDh_{i,j} = BDrop_{i,j} - BDrop_{i,j-1}, & \text{se } j > 0 \end{cases} \quad (6.15)$$

A segunda etapa do processo DCD, denominada  $CDv$ , é semelhante a  $CDh$ . A entrada é o valor residual gerado no processo da  $CDh$ , e a codificação diferencial é aplicada verticalmente. Portanto, dois casos são definidos, sendo esses casos apresentados na Equação 6.16. No primeiro caso para  $i = 0$ , a linha é copiada para o bloco residual. Caso contrário, para  $i > 0$ , o resíduo é obtido subtraindo  $(i - 1, j)$  de  $(i, j)$ .

$$\begin{cases} CDv_{i,j} = CDh_{i,j}, & \text{se } i = 0 \\ CDv_{i,j} = CDh_{i,j} - CDh_{i-1,j}, & \text{se } i > 0 \end{cases} \quad (6.16)$$

É importante notar que no exemplo apresentado na Figura 6.2, o  $BRec$  é composto por duas regiões (separadas por uma borda vertical) com valores de amostras bastante diferentes. Neste tipo de cenário, o bloco  $CDh$  residual pode apresentar valores com altas amplitudes, como pode ser visto na Figura 6.2. Nestes casos, a aplicação de uma codificação diferencial na direção vertical pode atenuar a amplitude de grande parte destes resíduos, sem causar impactos significativos nos demais valores.

Quando o processo DCD termina, os resíduos gerados estão prontos para serem codificados, e o processo de codificação de entropia é iniciado. O processo de codificação por entropia sem perdas utiliza uma tabela estática baseada na codificação de *Huffman*. O processo de codificação de *Huffman* exige uma análise de duas passagens, uma para calcular a probabilidade de ocorrência de cada símbolo, criando o dicionário de códigos, e outra para traduzir os símbolos para os novos códigos.

Devido a este processo de duas passagens, uma alta latência pode ser inserida, tornando esta abordagem ineficiente para aplicações de codificação de vídeo. Para resolver este problema, uma tabela de *Huffman* estática pode ser utilizada, onde a tabela é definida anteriormente usando um conjunto de dados predefinido. O conjunto de dados, no entanto, precisa ser escolhido cuidadosamente para cobrir casos que possuem símbolos com diferentes probabilidades de ocorrência. Nesta tese, o conjunto de 12 vídeos utilizados nos outros experimentos foi utilizado.

Outra questão referente a tabela de códigos de *Huffman* é a sua quantidade de entradas, uma vez que esses códigos devem ser armazenados em uma memória, portanto, tabelas de códigos menores são melhores. No entanto, como os resíduos gerados pelo módulo DCD variam de  $-510$  a  $510$ , seriam necessárias 1021 entradas para armazenar esta tabela, uma para cada resíduo possível.

No entanto, com a utilização do *bit dropping* essa tabela é reduzida. Com o *bit dropping* de 1 bit os resíduos variam de  $-254$  a  $254$ , uma vez que o valor máximo da amostra seria 127. Com o *bit dropping* de 4 bits, essa tabela tem no máximo valores de  $-30$  a  $30$ , sendo que o valor máximo de amostra é 15.

Além disso, apenas os resíduos em torno de zero têm alta probabilidade de ocorrência, conforme discutido anteriormente. Assim, é possível reduzir a tabela de códigos de forma a não armazenar os resíduos que possuem baixa probabilidade de ocorrência. Para fazer isso sem perder nenhuma informação, é necessário um código de exceção para representar os resíduos que estão fora da faixa estabelecida e, conseqüentemente, fora da tabela de códigos. Assim, quando o resíduo está dentro da faixa estabelecida, o código armazenado na tabela é usado; caso contrário, o código de exceção é usado seguido pelo valor do resíduo usando representação de comprimento fixo (10 bits).

Desta forma, o processo de codificação de entropia consiste em traduzir o resíduo gerado na etapa da DCD para o código de *Huffman* armazenado na tabela estática. Após este processo, os códigos são armazenados na memória de referência. A tabela de *Huffman* utilizada nesse trabalho será apresentada na próxima seção, uma vez que foi realizado um estudo com 5 tamanhos de tabelas para as 4 configurações, avaliando a taxa de compressão obtida por cada um desses tamanhos.

No processo de decodificação das informações é realizado o processo inverso da codificação, sendo que os passos são: i) executar o processo inverso da codificação de entropia, i.e. a partir dos códigos de *Huffman* gerar o resíduo correspondente; ii) executar a codificação diferencial inversa, nesta etapa o resíduo gerado será somado à amostra

anterior; iii) e gerar o bloco para compor a área de busca a ser utilizada pela ME.

A fase de decodificação inicia com a leitura do bloco codificado a partir da memória externa, aplicando o processo inverso da fase de codificação. Na decodificação de tamanho variável, não há nenhuma informação explícita para detectar o início ou o fim do código, exigindo assim uma leitura bit a bit da informação codificada.

No processo de decodificação, sabe-se que os primeiros  $N$  bits correspondem à primeira amostra do bloco original, o valor de  $N$  pode ser 7 para as 3 configurações que utilizam  $7bpp$  ou 4 caso seja a configuração  $16R-4bpp$ . Essa primeira amostra foi enviada para a memória apenas com o processo de *bit dropping*. Assim, ela é o ponto inicial para a decodificação das informações seguintes. Após esses  $N$  bits serem recuperados, a próxima etapa é responsável por obter um código válido.

Para obter um código válido, uma palavra com o tamanho máximo dos códigos armazenados na tabela é lida da memória, essa palavra é então comparada com todos os códigos da tabela estática. Apenas o tamanho proporcional ao tamanho do código da tabela é comparado, i.e. caso o código da tabela tenha 3 bits, apenas os 3 primeiros bits de  $W$  são comparados. Essa é uma propriedade interessante dos códigos de Huffman, nenhum código é parte de outro código de maior tamanho. Quando o código é encontrado na tabela, obtém-se o resíduo original. Esta etapa é repetida até que o bloco seja totalmente decodificado.

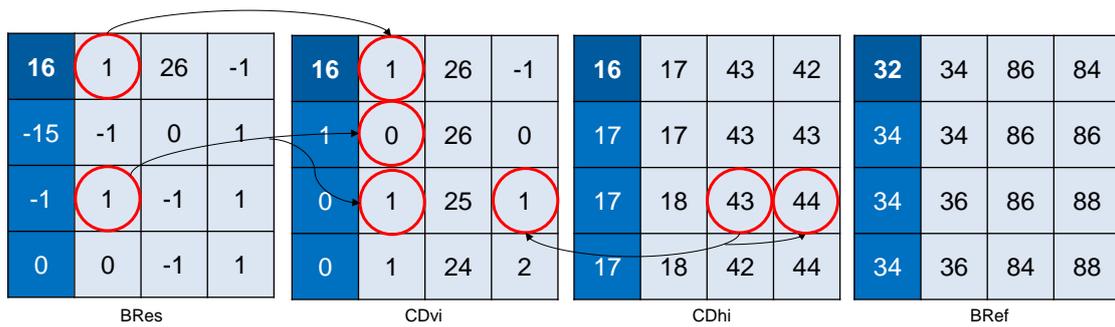
Quando o bloco é totalmente decodificado, o processo da dupla codificação diferencial inversa (DCDI) é aplicado para obter o bloco de referência. O processo DCDI é o processo DCD inverso e é separado em duas etapas. A primeira etapa aplica a  $CDv$  inversa ( $CDvi$ ), tendo como entrada o bloco residual gerado pela etapa de decodificação de entropia. O processo  $CDvi$  é definido na Equação 6.17.

O processo da  $CDvi$  tem dois casos. O primeiro caso, com  $i = 0$ , apenas copia a primeira linha armazenada no bloco residual,  $BRes$ . Esta linha é utilizada para obter os resíduos restantes. Em seguida, no segundo caso,  $i > 0$ , as outras linhas são processadas somando o resíduo  $BRes_{i,j}$  com o resíduo  $CDvi_{i,j}$ . Esses casos são exemplificados na Figura 6.3, onde são apresentados os resultados do processo da dupla codificação inversa para um bloco fictício de tamanho  $4 \times 4$ .

$$\begin{cases} CDvi_{i,j} = BRes_{i,j}, & \text{se } i = 0 \\ CDvi_{i,j} = BRes_{i,j} + CDvi_{i-1,j}, & \text{se } i > 0 \end{cases} \quad (6.17)$$

Após completar o processo de  $CDvi$ , a codificação diferencial horizontal inversa

Figura 6.3: Exemplo do processo da dupla codificação diferencial inversa.



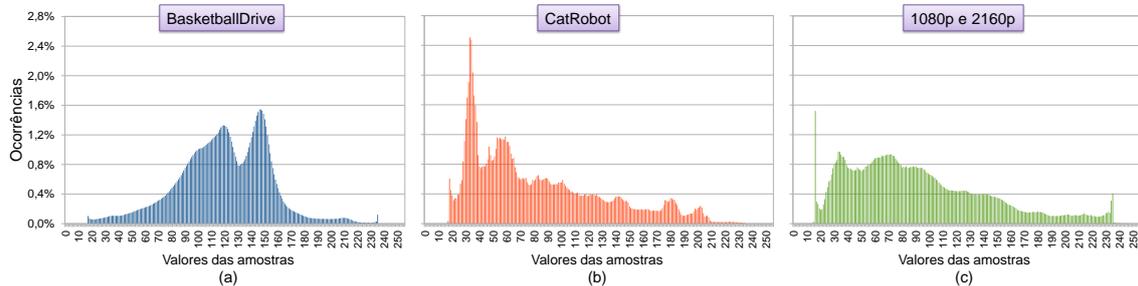
Fonte: Próprio autor.

( $CDhi$ ) é aplicada. O processo  $CDhi$  recebe como entrada o resíduo obtido pela  $CDvi$ . A  $CDhi$  é definida por três casos conforme pode ser visto na Equação 6.18. O primeiro caso, com  $i = 0$  e  $j = 0$ , ocorre quando a primeira amostra do bloco é processada. Essa amostra é simplesmente copiada para o bloco residual  $CDhi$ . O segundo caso,  $i > 0$  e  $j = 0$ , é a decodificação da primeira coluna do bloco, em que o resíduo é obtido somando o valor da posição  $CDvi_{i,j}$  com o valor da posição  $CDhi_{i-1,j}$ . O terceiro caso,  $j > 0$ , é a decodificação das linhas do bloco. Neste caso, o resíduo é obtido somando o valor da posição  $CDvi_{i,j}$  com o valor  $CDhi_{i,j-1}$ .

$$\begin{cases} CDh_{i,j} = CDv_{i,j}, & \text{se } i, j = 0 \\ CDh_{i,j} = CDv_{i,j} + CDh_{i-1,j}, & \text{se } i > 0 \text{ e } j = 0 \\ CDh_{i,j} = CDv_{i,j} + CDh_{i,j-1}, & \text{se } j > 0 \end{cases} \quad (6.18)$$

Na Figura 6.3 pode-se perceber que o bloco  $CDhi$  utilizado nessa figura é o mesmo bloco  $BDrop$  apresentado na Figura 6.2. Desta forma, se percebe que não há perdas de informações além das introduzidas pelo *bit dropping*. Além disso, nota-se na Figura 6.3 o bloco  $BRef$ , esse bloco contém as amostras que serão utilizadas pelo TZS, ele recebeu o processo inverso do *bit dropping*. Nesse processo as amostras foram deslocadas para a esquerda em 1 bit ou em 4 bits, conforme a configuração utilizada na codificação, com isso os bits cortados foram repostos com valores em 0. Conforme pode ser visto no bloco  $BRef$  da Figura 6.3, apenas as amostras com valores ímpares foram decrescidas em uma unidade quando comparado ao bloco reconstruído,  $BRec$  da Figura 6.2. Na próxima seção as arquiteturas desenvolvidas para o CODEC DDRFC-BD serão apresentadas em detalhes.

Figura 6.4: Histogramas das amostras dos vídeos, sendo (a) vídeo HD 1080p *Basketball-Drive*, (b) vídeo UHD 2160p *CatRobot* e (c) acumulado com as 12 sequências de vídeo HD e UHD.



Fonte: Próprio autor.

### 6.2.2 Análise da distribuição das amostras e definição da tabela de *Huffman*

Nessa seção será apresentada uma análise da distribuição das amostras dos vídeos e a definição da tabela de *Huffman*. A análise realizada na distribuição tem impacto direto na construção da tabela de *Huffman*, uma vez que a codificação de *Huffman* utiliza probabilidades de ocorrência dos símbolos, nesse caso amostras, a serem comprimidos para determinar códigos de tamanho variável para cada símbolo.

Na Figura 6.4 são apresentados três histogramas de amostras de luminância para duas sequências de vídeo completas, *BasketballDrive* (Figura 6.4 (a)) e *CatRobot* (Figura 6.4 (b)), em resoluções HD 1080p e UHD 2160p, respectivamente. Além disso, na Figura 6.4 (c) é apresentado o histograma com o acumulado das ocorrências dos valores das amostras para as 12 sequências de vídeos utilizadas, sendo 6 sequências HD e 6 UHD. A partir dessa figura percebe-se que os histogramas apresentam uma distribuição completamente distinta, sendo que essa distribuição varia de acordo com as características dos vídeos. Este tipo de distribuição esparsa, observada nesses três histogramas, é altamente indesejável para algoritmos de codificação de entropia, especialmente para uma codificação de tamanho variável, uma vez que as probabilidades das ocorrências está distribuída ao longo do conjunto de amostras.

Entretanto, as amostras de vídeos naturais apresentam elevada redundância espacial. Isto significa que amostras vizinhas tendem a ter valores semelhantes, exceto quando ocorre alguma borda (amostras vizinhas com grandes diferenças de valores). Assim, técnicas como a codificação diferencial podem ser utilizadas para alcançar taxas de redução de dados mais elevadas. Entretanto, a codificação diferencial precisa seguir algum modo de predição, que pode ser horizontal, vertical ou algum tipo de ordem diagonal. Esses

modos de predição para a codificação diferencial são semelhantes aos modos da predição intra dos padrões H.264/AVC e HEVC (SULLIVAN et al., 2012).

Em artigo publicado anteriormente (SILVEIRA et al., 2019), estes modos de codificação diferencial juntamente com a dupla codificação foram avaliados, sendo que a dupla codificação diferencial gerou melhores resultados quando comparada a codificações diferenciais simples. Nesse trabalho além da utilização da dupla codificação diferencial o *bit dropping* também foi utilizado. Com isso o conjunto de amostras foi modificado, gerando resíduos altamente concentrados em torno do valor 0, conforme pode ser visto na Figura 6.5. Desta forma, a codificação de entropia através da codificação de *Huffman* torna-se mais eficiente, uma vez que a maior probabilidade ficará concentrada em poucos valores, gerando símbolos de pequenos tamanhos e grandes ocorrências.

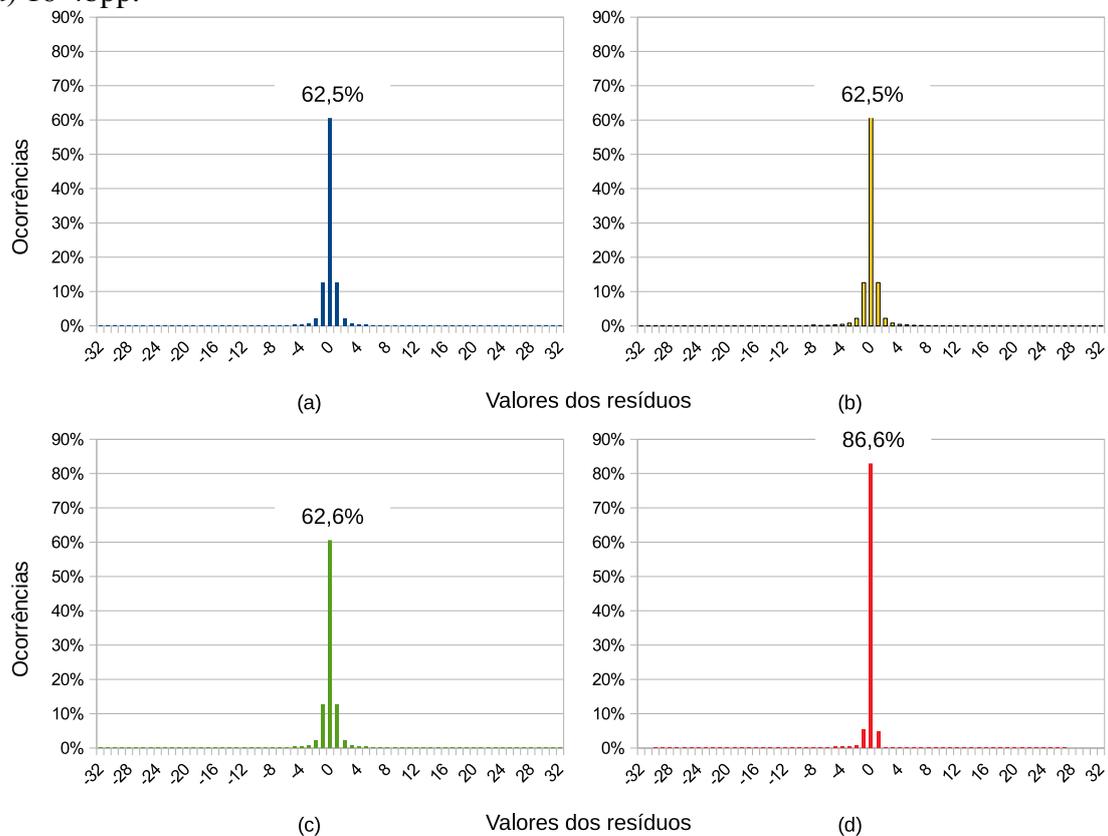
A Figura 6.5 apresenta os histogramas dos resíduos das amostras de luminância gerados após a etapa de *bit dropping* e da dupla codificação diferencial para as 4 configurações utilizadas. Os histogramas nessa figura foram gerados para os 12 vídeos e para os 4 QPs. Nota-se nessa figura que os 4 histogramas apresentam valores de resíduos altamente concentrados em torno do valor 0. Isso se deve a dois fatores: i) alta similaridade entre amostras vizinhas, e ii) a utilização do *bit dropping* nas amostras, o que aumenta a similaridade entre as amostras.

Esse segundo ponto pode ser visto na Figura 6.5, onde as três configurações que utilizam 7 bits por amostra geraram histogramas bastante semelhantes com o resíduo de valor 0 correspondendo a 62,5% das ocorrências entre todos os valores de resíduos. Já na configuração *16-4bpp*, que utiliza 4 bits por amostra, o resíduo de valor 0 teve 86,6% das ocorrências. Além disso, se considerarmos o intervalo com valores de resíduos  $[-2, 2]$  essa taxa apresenta uma aumento considerável, chegando nas configurações com 7bpp a 92,1% e na configuração *16-4bpp* a 97,5%.

Essa alta concentração em torno de um único valor faz com que a taxa de compressão obtida pelo algoritmo de *Huffman* seja mais alta, uma vez que com um valor tendo mais que 50% das ocorrências esse mesmo valor será representado por apenas 1 bit. Nesse sentido, nota-se a contribuição do *bit dropping* para o aumento da taxa de compressão do algoritmo, onde com a maior concentração dos resíduos em torno de um valor maior será a taxa de compressão do algoritmo.

Aliado a esta concentração das amostras ou dos resíduos outras configurações durante a codificação também geram impactos na taxa de compressão. Parâmetros como o tamanho do bloco a ser codificado e o tamanho da tabela estática para armazenar os có-

Figura 6.5: Histogramas dos resíduos após o *bit dropping* e a dupla codificação diferencial para as quatro configurações do sistema, sendo (a) 48-7bpp, (b) 32-7bpp, (c) 16-7bpp e (d) 16-4bpp.



Fonte: Próprio autor.

digos usados no processo de codificação influenciam diretamente os resultados obtidos. Entretanto, nesse trabalho o tamanho do bloco foi definido para  $8 \times 8$  amostras, uma vez que esse é o menor tamanho de PU utilizado no HEVC. Ao utilizar um tamanho de bloco maior que esse exigiria um *buffer* auxiliar para armazenar as amostras que não seriam imediatamente utilizadas.

Com o processo de dupla codificação diferencial, *bit dropping* e o tamanho de bloco definidos, o último parâmetro a ser definido é a tabela estática de *Huffman*. Para realizar a compressão, a tabela estática com os códigos deve ser armazenada em uma memória ROM. Além disso, na etapa de decodificação, o tamanho da tabela e a quantidade de códigos com diferentes tamanhos trazem grandes impactos em custo computacional, além de custos distintos de área e consumo de energia para a implementação em hardware. Isso se deve ao fato de que na decodificação de um código VLC não há nenhuma informação explícita para detectar o limite do início ou do fim do código em um fluxo de entrada, o que leva a uma procura bit a bit, ou a uma versão paralela para testar todos os

tamanhos ao mesmo tempo. Logo, pensando em custo computacional, tabelas de códigos menores são preferíveis.

Além disso, os resíduos gerados pela dupla codificação diferencial com *bit dropping* variam de -255 a 255 e desta forma, para armazenar esta tabela completa seriam necessárias 511 entradas, uma para cada valor de resíduo possível. No entanto, apenas os resíduos mais próximos de zero possuem alta ocorrência, como visto na Figura 6.5. Assim, é possível reduzir a tabela de código de modo que os resíduos que tenham baixa ocorrência não sejam armazenados de forma direta. Para fazer isso sem perda de informações, um código de exceção é necessário para representar os resíduos que estão fora do intervalo estabelecido. Assim, quando o resíduo está dentro do intervalo, o código armazenado na tabela de códigos é usado; caso contrário, o código de exceção é utilizado, seguido pelo valor do resíduo usando representação de comprimento fixo.

Para encontrar o melhor custo-benefício entre taxa de compressão e custo computacional foram avaliados 6 tamanhos de tabela estática, sendo 5 para as áreas de maior concentração de resíduos (próximos ao valor zero), e outra com tabela completa com códigos para todos os resíduos possíveis. Os resultados foram gerados para as 4 configurações. Para este experimento foi utilizado o conjunto de testes com 12 sequências de vídeos, considerando os 4 valores de QP: 22, 27, 32 e 37. Desta forma, cada um dos resultados apresentados na Tabela 6.1, representam o valor da taxa de compressão média para 48 simulações, composta por 12 sequências de vídeo para os 4 valores de QPs. Os resultados completos das taxas de compressão utilizados para gerar estes gráficos podem ser vistos no Apêndice C.

A Tabela 6.1 apresenta os resultados médios de taxa de compressão tanto em relação a técnica de *bit dropping* (linhas com resultados *Bit-configuração*), quanto em relação as amostras originais (linhas com resultados *Orig-configuração*). Os resultados em relação ao *bit dropping* são os ganhos em compressão além do ganho obtido com o *bit dropping*. Nota-se na Tabela 6.1 que as configurações que utilizam *7bpp* apresentam resultados semelhantes, independentemente do tamanho da tabela estática utilizada, com pequeno aumento na taxa de compressão conforme aumenta o tamanho da tabela. Entretanto, utilizando a tabela completa (resultados na coluna *Completa*) a taxa de compressão obtida não é a maior. Isso se deve ao fato de que com a tabela completa muitos códigos tem tamanhos maiores que os códigos utilizados nas tabelas com cortes, onde um código de exceção é utilizado junto com o resíduo. Além disso, a configuração *16-4bpp* apresenta maiores taxas de compressão que as outras configurações. Isso já era esperado,

Tabela 6.1: Resultados médios de taxa de compressão para as 4 configurações com 6 tamanhos de tabela estática.

<b>Configuração</b>	<b>[-2, 2]</b>	<b>[-4, 4]</b>	<b>[-8, 8]</b>	<b>[-16, 16]</b>	<b>[-32, 32]</b>	<b>Completa</b>
Bit-48-7bpp	58, 55	61, 09	62, 44	62, 66	62, 86	58, 48
Orig-48-7bpp	63, 73	65, 95	67, 14	67, 33	67, 50	63, 66
Bit-32-7bpp	58, 55	61, 08	62, 43	62, 65	62, 86	58, 48
Orig-32-7bpp	63, 73	65, 95	67, 13	67, 32	67, 50	63, 66
Bit-16-7bpp	58, 59	61, 11	62, 46	62, 68	62, 89	58, 49
Orig-16-7bpp	63, 76	65, 99	67, 16	67, 35	67, 53	63, 68
Bit-16-4bpp	63, 11	63, 88	64, 57	64, 64	64, 71	48, 72
Orig-16-4bpp	81, 56	81, 94	82, 29	82, 32	82, 35	74, 36
<b>Bit-Média</b>	59, 70	61, 79	62, 98	63, 16	63, 33	56, 04
<b>Orig-Média</b>	68, 20	69, 95	70, 92	71, 08	71, 22	66, 34

Fonte: Próprio autor.

uma vez que nessa configuração são gerados mais resíduos com valor zero ou próximo de zero, conforme pode ser visto na Figura 6.5, os quais são os valores de resíduos com os menores códigos de *Huffman*

Considerando os resultados apresentados na Tabela 6.1, observa-se que a partir do intervalo  $[-8, +8]$  o aumento na taxa de compressão é pequeno. Do intervalo  $[-8, +8]$  para o intervalo  $[-16, +16]$  houve uma melhora na taxa de compressão de apenas 0, 16%, na média dos resultados. Essa taxa de compressão aumenta em 0, 3% quando comparado ao intervalo  $[-32, +32]$ , mesmo esse intervalo possuindo uma tabela 4 vezes maior, sendo que o tamanho da tabela é um fator importante na escolha da mesma, uma vez que essa tabela é implementada em hardware.

Desta forma, com base nos resultados das taxas de compressão do algoritmo, o tamanho da tabela de códigos, definida para ser utilizada nas arquiteturas de hardware é a tabela com intervalo  $[-8, +8]$ , a qual armazena 18 códigos, sendo 17 códigos para o intervalo e mais um código de exceção. Com isso, a taxa de compressão do algoritmo DDRFC-BD foi de 70, 92% na média para as 4 configurações, sendo que a taxa de compressão média para as 3 configurações com 7bpp foi de 67, 1%, e 82, 2% para a configuração 16-4bpp. Os resultados de taxa de compressão serão detalhados no próximo capítulo na Seção 7.1 juntamente com os resultados de redução de largura de banda de memória do MERSy.

Os símbolos da tabela de *Huffman* para o intervalo  $[-8, +8]$  podem ser observados na Tabela 6.2. Como pode ser visto na Tabela 6.2, os menores tamanhos de código são gerados para símbolos que ocorrem com mais frequência, os símbolos próximos de zero

Tabela 6.2: Tabela com códigos de *Huffman* para o intervalo  $[-8, +8]$ .

Resíduo	Código	Resíduo	Código	Resíduo	Código
-8	11101010	-2	111110	4	11111110
-7	111011110	-1	110	5	11101011
-6	111111110	0	0	6	111011111
-5	11101110	1	10	7	1111111110
-4	1110100	2	11100	8	1111111111
-3	1111110	3	1110110	Excec.	11110

Fonte: Próprio autor.

têm mais ocorrências conforme discutido anteriormente. Na última linha dessa tabela está também o código de exceção, o qual é utilizado pelos resíduos que estão fora do intervalo especificado.

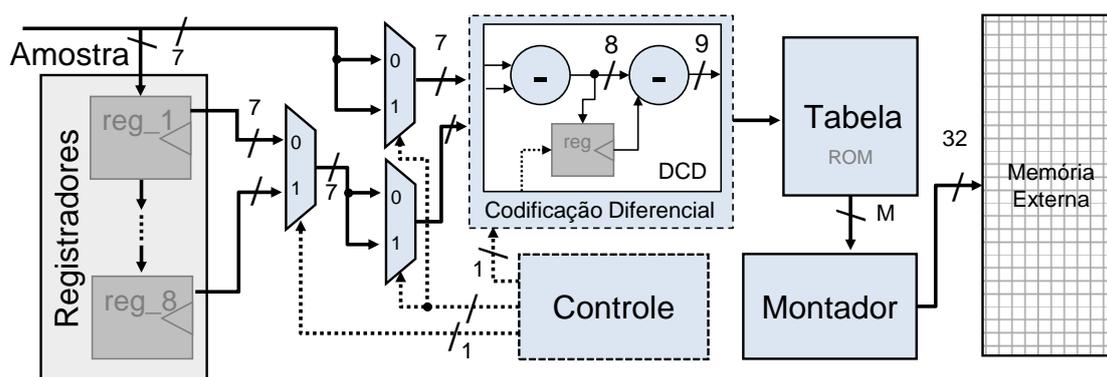
Como pode ser visto na Tabela 6.2, apenas 18 códigos de *Huffman* são utilizados pelo DDRFC-BD para realizar a compressão dos quadros de referência. A quantidade de códigos armazenados na tabela, i.e. o tamanho da tabela, é um fator importante, principalmente para a construção das arquiteturas de hardware do DDRFC-BD, uma vez que essa tabela ficará armazenada em uma memória ROM, tanto no codificador quanto no decodificador. Desta forma, quanto menor for o tamanho dessa tabela melhor será o desempenho da arquitetura de hardware em termos de dissipação de potência, área utilizada e taxa de processamento. Nas próximas seções as arquiteturas de hardware desenvolvidas para o DDRFC-BD serão apresentadas juntamente com os resultados de síntese dessas arquiteturas.

### 6.2.3 Arquitetura de hardware do DDRFC-BD

A Figura 6.6 apresenta a arquitetura de hardware desenvolvida para o codificador do compressor de quadros de referência. Para cada bloco codificado, o primeiro registrador é definido para zero. Isto é feito para preservar o valor da primeira amostra do bloco. Deste modo, não é necessário um processo especial para a primeira amostra. O *bit dropping* já é realizado na entrada das amostras, onde apenas os 7 MSBs são armazenados, uma vez que 3 configurações utilizam amostras com 7 bits e outra configuração com apenas 4 bits.

As entradas para o módulo codificador são sempre números positivos, uma vez que as amostras de luminância variam de 0 a 127, considerando 7 bits por amostra. O primeiro

Figura 6.6: Arquitetura de hardware do módulo codificador do DDRFC-BD.



Fonte: Próprio autor.

multiplexador faz a seleção entre a amostra da mesma coluna ou da coluna vizinha, e após, essa amostra e a amostra de entrada passam por mais dois multiplexadores. Nesses multiplexadores, caso a configuração *16-4bpp* seja utilizada, é realizado o segundo *bit dropping*, onde apenas os 4 MSBs passam para o próximo módulo, caso contrário, os 7 MSBs seguem. As saídas desses multiplexadores são as entradas para o módulo da codificação diferencial.

O módulo da codificação diferencial é composto por dois subtratores, como pode ser visto na Figura 6.6. Nesse módulo, dois subtratores são utilizados em série, um subtrator de 8 bits e um subtrator de 9 bits, evitando a ocorrência de *overflow* no cálculo. Além disso, um registrador de 7 bits é utilizado para armazenar o primeiro resíduo. A primeira codificação diferencial é realizada entre a amostra atual e a amostra selecionada pelo multiplexador. A segunda codificação diferencial é realizada entre o resíduo atual e o resíduo anterior, que está armazenado no registrador de 7 bits. Os resíduos gerados pela codificação diferencial são enviados para o módulo Tabela.

Na Tabela, o resíduo é traduzido para os códigos de *Huffman* correspondentes. Este módulo é composto por uma pequena memória ROM que armazena os códigos VLC, e o código de exceção utilizado para codificar os resíduos fora do intervalo especificado, uma vez que o algoritmo não considera a tabela completa. Se o código de exceção é utilizado, o resíduo é apenas anexado ao código de exceção. A tabela também contém o tamanho, em bits, dos códigos para cada amostra. Essa informação é utilizada pelo módulo Montador para escrever corretamente o código no seu *buffer* interno. Desta forma, a saída da Tabela possui 18 bits, sendo 4 bits para informar o tamanho do código e 14 bits para o código propriamente dito. O maior tamanho de código ocorre quando é utilizado o código de exceção, 5 bits, mais o resíduo de 9 bits. O menor tamanho de código possui

apenas 1 bit, quando ocorre o resíduo de valor 0, o qual ocorre na maioria dos casos conforme pode ser visto na Figura 6.5. Os códigos VLC são os mesmos apresentados na Tabela 6.2.

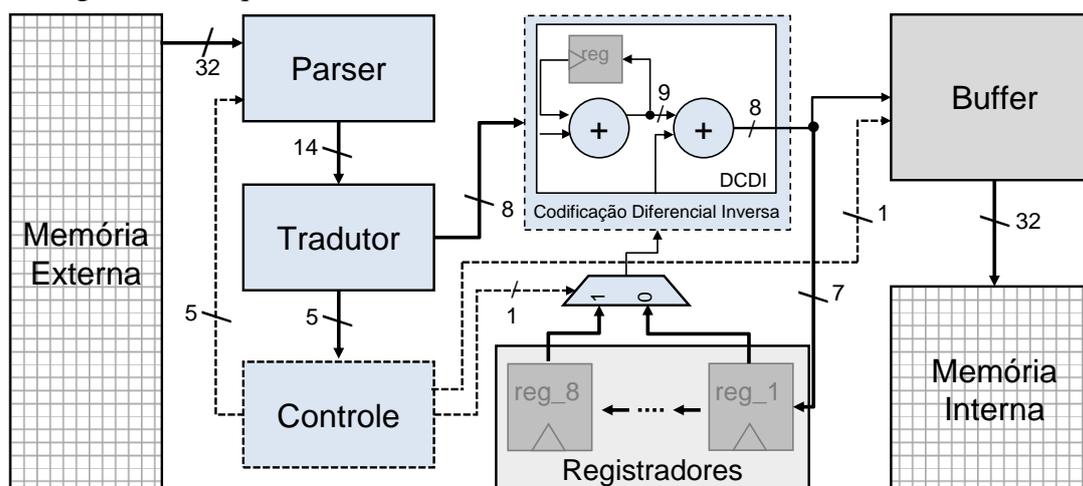
O módulo Montador é utilizado para normalizar a saída do codificador para a memória externa, uma vez que o codificador trabalha com codificação de comprimento variável, e memórias comerciais funcionam com palavras de tamanho fixo. Desta forma, sempre que 32 bits contínuos de informações são escritos no *buffer* circular, esta palavra é escrita na memória externa. O Montador foi projetado como um *buffer* circular, sendo o tamanho deste *buffer* definido pela soma do tamanho do maior código VLC com 32 que é o tamanho da saída. Assim, como o maior código possui 14 bits, o qual ocorre quando o código de exceção é utilizado, o *buffer* circular possui 46 bits, uma vez que o pior caso envolve escrever a palavra mais longa (14 bits) com a saída do codificador pronta para ser escrita na memória externa (32 bits).

O Controle possui uma máquina de estados finitos que define as amostras certas a serem subtraídas na codificação diferencial, os estados seguem os casos apresentados nas Equações 6.15 e 6.16. Além disso, o Controle especifica o tamanho do bloco utilizado. Dessa forma o início e o fim do bloco são conhecidos, essas são informações importantes, porque após a última amostra do bloco ser codificada, esta informação é escrita na memória externa; mesmo que o montador não esteja com a palavra de 32 bits completa (os outros bits não serão utilizados). Este processo gera uma fragmentação na memória externa; no entanto, permite acesso aleatório aos blocos da memória externa, uma vez que não existem dependências de dados entre os blocos codificados.

Enquanto a operação de escrita é realizada, o espaço restante no *buffer* é utilizado para receber mais entradas do codificador. Ao fazer isso, o sistema não fica parado durante as operações de escrita na memória. Todo esse processo de codificação foi projetado para usar um ciclo de relógio por amostra.

A arquitetura do decodificador DDRFC-BD possui dois núcleos de decodificação em paralelo, os quais são iguais e funcionam de forma independente, decodificando duas amostras por ciclo. A Figura 6.7 apresenta o diagrama de blocos de um desses núcleos da arquitetura do decodificador. Inicialmente, os blocos codificados são obtidos a partir da memória externa. O processo de decodificação ocorre coluna por coluna. O componente Parser é responsável por padronizar a entrada para o módulo Tradutor. O Parser recebe uma palavra de 32 bits como entrada, este módulo possui um *buffer* interno que funciona como uma fila para armazenar as entradas. Este *buffer* é necessário, uma vez que as

Figura 6.7: Arquitetura de hardware do módulo decodificador do DDRFC-BD.



Fonte: Próprio autor.

palavras codificadas apresentam comprimento variável. Este componente recebe também informações da unidade de Controle sobre o tamanho real da amostra codificada. Esta informação é importante, uma vez que é utilizada para atualizar o *buffer* interno desse componente, a fim de gerar a próxima palavra de forma correta. A saída do Parser é um código com tamanho do maior código presente na tabela. Este código é enviado para o módulo Tradutor. Além disso, o Parser emite um sinal para a unidade de Controle, informando se o *buffer* tem espaço suficiente para uma nova entrada.

O processo de decodificação do código é executado pelo módulo Tradutor. Esse módulo recebe o resíduo codificado, o qual é decodificado de acordo com a tabela de decodificação. Esta tabela está localizada no módulo Tradutor e foi projetada usando uma memória ROM, assim como no codificador. Nessa tabela, todos os possíveis códigos são testados em paralelo, retornando com valor válido apenas o código correto. Isso é possível devido a utilização da codificação de *Huffman*, a qual gera códigos com prefixos únicos. Além disso, tabela de decodificação contém o tamanho, em bits, do valor codificado. Essas informações são retornadas ao Parser através da unidade de Controle, de modo que o *buffer* interno ao Parser seja atualizado para a posição correta e, em seguida, uma nova amostra possa ser decodificada. A saída do módulo Tradutor é o resíduo com tamanho de 9 bits, sendo esse resíduo enviado ao módulo da codificação diferencial inversa.

A codificação diferencial inversa é composta por dois somadores em série, um somador de 9 bits e um somador de 8 bits. Essa codificação diferencial é realizada em dois passos: o primeiro, executa a soma entre o resíduo decodificado e o resíduo anterior, que está armazenado no registrador interno ao módulo, sendo que para a primeira amostra do

bloco esse registrador estará com o valor zero. No segundo passo, a soma é realizada entre o resíduo atual e a amostra selecionada pelo multiplexador do módulo banco de Registradores. Esse multiplexador selecionará durante a decodificação da primeira coluna, i.e. as primeiras 8 amostras do bloco, os valores que estarão na entrada “0” do multiplexador e para a decodificação restante do bloco, os valores contidos na entrada “1” do multiplexador serão selecionadas. Desta forma, a arquitetura segue os mesmos passos do algoritmo apresentado na seção anterior.

Após a codificação diferencial inversa, a amostra original é gerada e enviada para o *buffer* de saída e para o módulo Registradores. O módulo Registradores é utilizado para armazenar uma coluna inteira do bloco decodificado. Desta forma, este módulo possui capacidade de armazenamento de 56 bits, i.e. 8 amostras de, no máximo, 7 bits cada. Todo este processo, assim como no codificador, foi projetado para usar um ciclo de relógio por amostra decodificada. Nota-se que no decodificador não há o processo de *bit dropping*, assim como é realizado no codificador. O *bit dropping* inverso, i.e. o retorno dos LSBs removidos para os resíduos, deverá ser feito pelo hardware da ME. Desta forma, evita-se o armazenamento de bits, sabidamente com valor “0”, nas memórias internas. Na próxima seção serão apresentados os resultados de síntese lógica para biblioteca de células padrão das arquiteturas do DDRFC-BD.

#### 6.2.4 Síntese lógica das arquiteturas

As arquiteturas de hardware do codificador e do decodificador DDRFC-BD foram descritas em VHDL e sintetizadas para ASIC em tecnologia de células padrão. A síntese foi gerada visando a tecnologia 65nm da *ST Microelectronics* com 1.0V de tensão de alimentação, utilizando a ferramenta de síntese Genus<sup>TM</sup> da Cadence (CADENCE, 2020).

Para obter os resultados de síntese lógica foi empregada a metodologia de síntese apresentada em Goebel et al. (2016), Paim et al. (2019) e Rocha et al. (2020). Nesses trabalhos, os autores utilizam vetores reais extraídos do software de referência para gerar estímulos de chaveamento adequados para as arquiteturas de hardware. Desta forma, os resultados de síntese ficam mais próximos dos resultados reais, uma vez que as ferramentas de EDA (*Electronic design automation*), quando utilizadas com síntese lógica automática, i.e. sem a utilização de vetores reais, assumem uma probabilidade na atividade de chaveamento dos circuitos muito pessimista, e acabam não gerando os efeitos da propagação do sinal na dissipação de potência (GOEBEL et al., 2016).

Para obter esses vetores reais, foram utilizadas amostras de quadros de referência de duas sequências de vídeo, sendo  $512 \times 512$  amostras do quadro de referência da sequência de vídeo HD 1080p *BQTerrace* e  $512 \times 512$  amostras do quadro de referência da sequência UHD 2160p *Beauty*, ambos os quadros foram obtidos de simulação com QP 22. Essa quantidade de amostras foi utilizada para cada uma das quatro configurações, *48-7bpp*, *32-7bpp*, *16-7bpp* e *16-4bpp*, totalizando 8 vetores de entrada reais. Essas duas sequências de vídeo foram escolhidas por serem as que apresentaram as menores taxas de compressão dentre as sequências de teste de suas resoluções. Desta forma, são as entradas que mais geram estímulos nas arquiteturas e, conseqüentemente, maior dissipação de potência. Com esses vetores de entrada foram gerados arquivos *Toggle Count Format* (TCF), os quais contabilizam as atividades dos circuitos através da ferramenta de simulação Incisive<sup>TM</sup> da Cadence (CADENCE, 2020).

Além disso, foi utilizado o modo *Physically-Aware Layout Estimation* (PLE) na síntese lógica realizada pela ferramenta Genus. Este modo executa um planejamento básico para a síntese física do circuito, estimando comprimento de fios para conectar todas as células digitais, sendo fundamental para considerar os efeitos desses fios em termos de caminho crítico, área e consumo de energia. Os resultados das sínteses para células padrão são apresentados na Tabela 6.3. Essa tabela apresenta os resultados de potência em  $mW$ , de área em termos de portas lógicas NAND de 2 entradas (*gates*) e em  $(\mu m)^2$ , e de consumo de energia em  $pJ/amostra$ .

Os resultados são apresentados para cada uma das quatro configurações avaliadas pelo MERSy divididos entre codificador (*coder*) e decodificador (*decoder*). Além disso, foram definidos três objetivos de taxas de processamento: HD 1080p com 30 QPS, HD 1080p com 60 QPS e UHD 2160p com 30 QPS. Desta forma, exigindo das arquiteturas diferentes taxas de processamento e conseqüentemente diferentes frequências de operação. As frequências de operação para cada arquitetura e objetivo estão presentes na Tabela 6.3.

Conforme se pode observar nas informações apresentados na Tabela 6.3, os resultados de síntese do codificador para as três configurações que utilizam 7 bits são os mesmos. Esse resultado já era esperado, uma vez que para essas três configurações a quantidade de dados a serem codificados e o tamanho das amostras são os mesmos. Entretanto, para a configuração *16-4bpp* os resultados de síntese apresentaram pequena melhoria devido ao menor tamanho das amostras a serem codificadas, reduzindo a quantidade de operações para codificar as amostras.

Quanto aos resultados do decodificador, nota-se que há grande variação nos re-

Tabela 6.3: Resultados de síntese lógica para a tecnologia ST 65nm com 1.0V de tensão das arquiteturas de hardware do DDRFC-BD

<b>48-7bpp</b>	<i>1080p@30qps</i>		<i>1080p@60qps</i>		<i>2160p@30qps</i>	
	coder	decoder	coder	decoder	coder	decoder
Frequência ( <i>MHz</i> )	65	85	130	170	250	325
Potência ( <i>mW</i> )	0,267	1,227	0,524	2,345	1,026	4,570
Energia ( <i>pJ/amostra</i> )	4,295	7,458	4,208	7,128	4,124	7,115
Área ( $\mu m$ ) <sup>2</sup>	6434	21661	6543	21711	6893	21978
Área ( <i>gates</i> )	3093	10414	3146	10438	3314	10566
<b>32-7bpp</b>	<i>1080p@30qps</i>		<i>1080p@60qps</i>		<i>2160p@30qps</i>	
	coder	decoder	coder	decoder	coder	decoder
Frequência ( <i>MHz</i> )	65	65	130	130	250	260
Potência ( <i>mW</i> )	0,267	0,938	0,524	1,857	1,026	3,645
Energia ( <i>pJ/amostra</i> )	4,295	7,244	4,208	7,167	4,124	7,152
Área ( $\mu m$ ) <sup>2</sup>	6434	21661	6543	21711	6893	21978
Área ( <i>gates</i> )	3093	10414	3146	10438	3314	10566
<b>16-7bpp</b>	<i>1080p@30qps</i>		<i>1080p@60qps</i>		<i>2160p@30qps</i>	
	coder	decoder	coder	decoder	coder	decoder
Frequência ( <i>MHz</i> )	65	50	130	100	250	190
Potência ( <i>mW</i> )	0,267	0,722	0,524	1,423	1,026	2,742
Energia ( <i>pJ/amostra</i> )	4,295	7,552	4,208	7,447	4,124	7,233
Área ( $\mu m$ ) <sup>2</sup>	6434	21661	6543	21711	6893	21978
Área ( <i>gates</i> )	3093	10414	3146	10438	3314	10566
<b>16-4bpp</b>	<i>1080p@30qps</i>		<i>1080p@60qps</i>		<i>2160p@30qps</i>	
	coder	decoder	coder	decoder	coder	decoder
Frequência ( <i>MHz</i> )	65	50	130	100	250	190
Potência ( <i>mW</i> )	0,255	0,681	0,481	1,352	0,943	2,546
Energia ( <i>pJ/amostra</i> )	4,106	7,120	3,867	7,073	3,791	6,715
Área ( $\mu m$ ) <sup>2</sup>	6434	21661	6543	21711	6893	21978
Área ( <i>gates</i> )	3093	10414	3146	10438	3314	10566

Fonte: Próprio autor.

sultados. Isto se deve ao fato de que para o decodificador, a quantidade de informações a serem processadas está diretamente ligada ao tamanho da área de busca, uma vez que quanto maior a área de busca mais amostras serão trazidas da memória externa e, conseqüentemente, devem ser decodificadas. Como o MERSy utiliza quatro configurações com diferentes tamanhos de área de busca para três taxas de processamento, isso acaba refletindo nos resultados da síntese. Desta forma, nota-se nos resultados de frequência, potência dissipada e consumo de energia que os maiores gastos estão na configuração *48-7bpp*, principalmente para os resultados para vídeos UHD. Do mesmo modo, quanto mais restritiva é a configuração menor será a frequência de operação exigida e, conseqüentemente menor o gasto em energia, como por exemplo a configuração *16-4bpp*, a qual apresenta os menores resultados de frequência, dissipação de potência e consumo de energia.

A Tabela 6.3 também apresenta as frequências de operações utilizadas nas arquiteturas para atender a taxa de processamento necessária para cada configuração. Para definir a frequência de operação foi considerada a Equação 2.4 apresentada na Seção 2.4, e a capacidade de processamento, em amostras por ciclo, de cada arquitetura. Desta forma, é possível observar que as frequências definidas para o codificador e o decodificador são distintas. No codificador a quantidade de amostras processadas é a mesma, independente da configuração utilizada, uma vez que o tamanho do quadro de referência será o mesmo. A frequência de operação do codificador só é alterada quando a taxa de processamento muda, sendo que isso ocorre quando a quantidade de quadros a serem processados ou a resolução do vídeo aumentam. Entretanto, a frequência de operação do decodificador está diretamente relacionada a quantidade de áreas de busca a serem utilizadas e ao tamanho dessas áreas de busca. Desta forma, quando a configuração *48-7bpp* for utilizada, ela exigirá maior taxa de processamento que as outras configurações, pois ela utiliza a maior área de busca entre as configurações. Por outro lado, a configuração *16-4bpp* utiliza a menor área de busca entre as configurações, e conseqüentemente exigirá a menor frequência de operação entre as configurações do MERSy.

Além dos resultados de consumo de energia, a Tabela 6.3 também apresenta resultados de área para as arquiteturas. Os resultados de área são apresentados em  $(\mu m)^2$  e em quantidades de portas lógicas básicas do tipo NAND de 2 entradas, sendo essa uma métrica bastante utilizadas entre trabalhos da literatura, uma vez que permite comparativos entre diferentes arquiteturas sintetizadas para diferentes tecnologias. Nota-se na Tabela 6.3 que os resultados de área são iguais para todas as quatro configurações do

MERSy. Isso foi realizado de forma proposital, uma vez que a arquitetura de hardware será a mesma, mudando apenas a frequência de operação. Assim, foi mantido o tamanho da arquitetura para a configuração *48-7bpp*, a qual exige mais recursos de hardware para o pleno funcionamento.

No próximo capítulo serão apresentados e discutidos os resultados do MERSy, incluindo resultados de redução largura de banda de memória, redução do consumo de energia e impacto na eficiência de codificação. Os resultados de síntese das arquiteturas apresentados nessa seção serão utilizados para compor os resultados de consumo de energia da solução completa para as quatro configurações do MERSy.

### **6.3 Organização das memórias e reuso de dados**

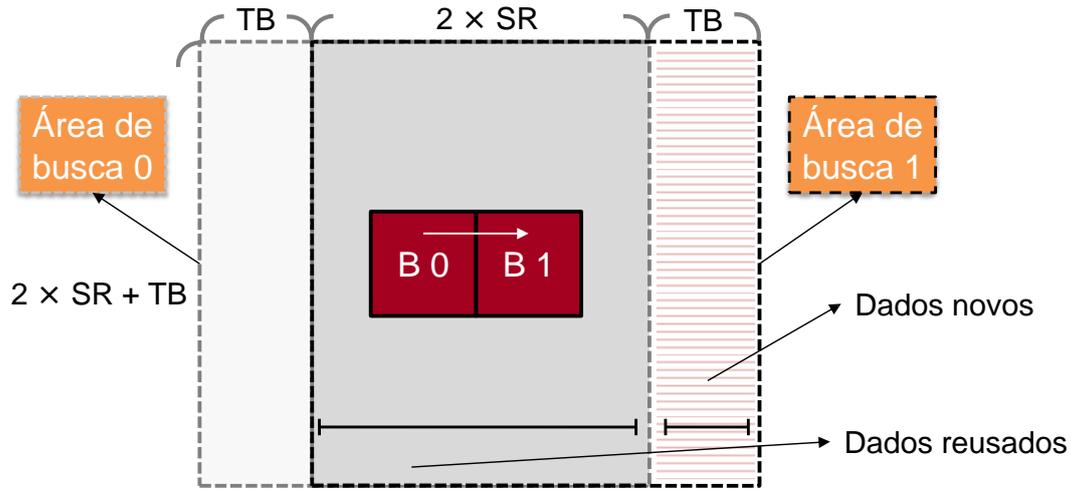
Nessa seção será apresentada a organização das memórias externa e internas utilizadas no MERSy. A Seção 6.3.1 apresenta a organização da memória interna, a qual suporta quatro níveis de configurações, *48-7bpp*, *32-7bpp*, *16-7bpp* e *16-4bpp*, onde cada uma dessas configurações demandam quantidades distintas de memória para armazenamento dos dados e em todas configurações o reuso de dados Level C é utilizado. Na Seção 6.3.2 é apresentada a organização da memória externa, a qual possibilita o acesso aleatório aos blocos codificados que estão armazenadas na memória.

#### **6.3.1 Organização da memória interna**

Nesta seção será apresentada e discutida a arquitetura proposta para a memória interna do MERSy. Para essa memória foi utilizada uma memória *scratchpad* (SPM), uma vez que o comportamento de todos acessos à memória é conhecido pela aplicação e, desta forma, não havendo necessidade de utilização de uma memória cache. Como mencionado anteriormente, o sistema utiliza 4 configurações, sendo elas: *48-7bpp*, *32-7bpp*, *16-7bpp* e *16-4bpp*, onde cada uma dessas configurações demandam quantidades distintas de memória para armazenamento dos dados e em todas configurações o reuso de dados Level C é utilizado.

O reuso de dados Level C é uma das técnicas de redução de largura de banda de memória mais utilizadas em arquiteturas de hardware para ME, uma vez que apresenta fácil implementação em hardware, provendo altas taxas de redução na comunicação com

Figura 6.8: Processo de reuso de dados Level C entre duas áreas de busca



Fonte: (TUAN; CHANG; JEN, 2002)

a memória externa. O Level C explora a localidade de dados de regiões sobrepostas entre áreas de busca.

A Figura 6.8 apresenta o processo de reuso de dados do Level C entre duas áreas de busca. Nesta figura,  $SR$  significa intervalo de busca e  $TB$  é o tamanho do bloco utilizado na codificação.  $B_0$  e  $B_1$  são os blocos centrais das áreas de busca 0 e 1, respectivamente. A região em cinza escuro representa a área reusada ( $LevelC_{DR}$ ) entre as duas áreas de busca, o tamanho dessa área reusada é dado pela Equação 6.19. Já a região branca (com linhas horizontais) representa os novos dados trazidos da memória externa, a quantidade de dados novos ( $LevelC_{DN}$ ) é dado pela Equação 6.20. Além disso, a Equação 6.21 indica o tamanho mínimo de memória ( $LevelC_{TM}$ ) necessária para armazenar a área de busca utilizando o Level C.

$$LevelC_{DR} = (2 \times SR + TB) \times (2 \times SR) \quad (6.19)$$

$$LevelC_{DN} = (2 \times SR + TB) \times TB \quad (6.20)$$

$$LevelC_{TM} = (2 \times SR + TB) \times (2 \times SR + TB) \quad (6.21)$$

Como pode ser observado nas Equações 6.19, 6.20 e 6.21 o tamanho do intervalo de busca ( $SR$ ) é o termo que causa mais impacto nas taxas de reuso e também na largura de banda da memória externa. As quatro configurações utilizadas pelo sistema proposto nessa tese reduzem o intervalo de busca de  $[-64, +64]$  para  $[-48, +48]$ ,  $[-32, +32]$  e

$[-16, +16]$ , reduzindo a largura de banda necessária à codificação. Além disso, as quatro configurações utilizam *bit dropping*, o que reduz ainda mais essa taxa.

Nesse sentido, a Tabela 6.4 apresenta um comparativo com dados de reuso do Level C entre uma configuração tradicional sem cortes na área de busca (coluna *64-8bpp*) e as quatro configurações utilizadas pelo MERSy. Na Tabela 6.4 são apresentados comparativos entre tamanho mínimo da memória, dados novos necessários a cada nova área de busca, os dados reusados, a redução no tamanho da memória em relação a configuração tradicional e a redução em dados novos trazidos da memória externa em relação a configuração tradicional. Essa configuração utiliza blocos  $64 \times 64$ , intervalo de busca  $[-64, +64]$  e oito bits por amostra.

Nota-se na Tabela 6.4 que todas as configurações utilizadas pelo MERSy atingem altas taxas de redução em relação a configuração tradicional. Em relação a redução do tamanho da memória interna necessária para armazenar os dados da área de busca, a configuração *48-7bpp* apresenta uma redução de 39,2% enquanto que a configuração *16-4bpp* apresenta uma redução de 87,5% no tamanho da memória interna. Considerando a redução em novos dados trazidos da memória externa, a configuração *48-7bpp* apresenta uma redução de 27,1% e a configuração *16-4bpp* atinge 75% de redução nesses acessos. Outro fato que pode ser observado na Tabela 6.4 é a relação entre dados novos e dados reusados. Na configuração tradicional 66,6% dos dados são reusados a cada nova área de busca. Essa taxa diminui com as configurações propostas, uma vez que essas configurações apresentam cortes no intervalo, e segundo visto nas equações anteriores, o intervalo de busca é termo o que mais impacta na taxa de reuso de dados.

Desta forma, a configuração *48-7bpp*, configuração que apresenta os menores cortes na área de busca, apresenta uma taxa de 60% de reuso de dados a cada nova área de busca. Na configuração *16-4bpp* essa taxa é reduzida para 33,3% a cada nova área de busca. Entretanto, mesmo apresentando uma taxa de reuso de dados, proporcional a sua área, menor que as demais configurações, a configuração *16-4bpp* é a que mais reduz acessos à memória externa.

Como pode ser observado na Tabela 6.4, as quatro configurações do sistema utilizam quantidades distintas de memória. Como a configuração *48-7bpp* é a que exige a maior quantidade de memória as outras configurações podem utilizar essa mesma memória para armazenar os dados. Entretanto, quando as configurações com mais cortes na área de busca são utilizadas nem todas as células de memória são necessárias.

Desta forma, para evitar o consumo de energia estática das células que não são

Tabela 6.4: Comparativo entre os dados das memórias internas utilizando somente o Level C e Level C mais as quatro configurações com cortes na área de busca e *bit dropping*.

<b>Memória</b>	<b>64-8bpp</b>	<b>48-7bpp</b>	<b>32-7bpp</b>	<b>16-7bpp</b>	<b>16-4bpp</b>
Tamanho (bytes)	36.864	22.400	14.336	8.064	4.608
Dados novos (bytes)	12.288	8.960	7.168	5.376	3.072
Dados reusados (bytes)	24.576	13.440	7.168	2.688	1.536
Redução da memória	0%	39,2%	61,1%	78,1%	87,5%
Redução de novos dados	0%	27,1%	41,7%	56,3%	75,0%

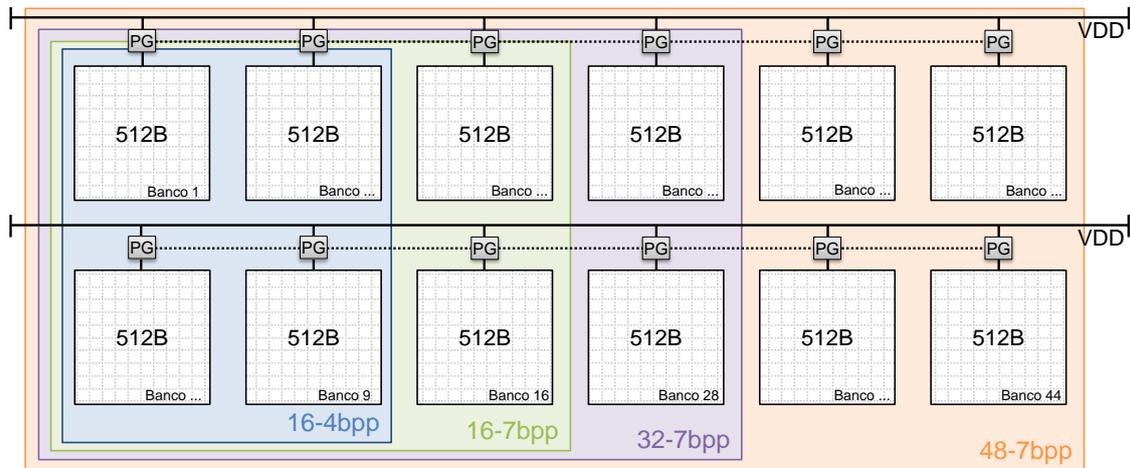
Fonte: Próprio autor.

utilizadas é realizado um *power gating* a nível de setores da memória interna. Esse desligamento dos setores da memória é implementado através da utilização de *sleep-state transistors* (JEONG et al., 2012), os quais desligam as linhas de alimentação dos bancos de memória.

A Figura 6.9 ilustra o projeto arquitetural da memória interna, a qual é composta por 44 bancos de memória com 512B de capacidade cada um. Desta forma, essa memória interna tem capacidade máxima de 22KB de armazenamento, que são suficientes para armazenar os dados mesmo na configuração com os menores cortes, *48-7bpp*. Quando são utilizadas as demais configurações os bancos que não serão utilizados são desligados. Cada circuito de *power gating* é associado a um banco e ao respectivo barramento. Quando um banco não for utilizado, tem sua alimentação interrompida, bem como a alimentação dos *buffers* em seu barramento. Além disso, *tri-state buffers* (MARTINS et al., 2018) são acionados às portas de entrada e saída dos barramentos, colocando os barramentos não utilizados em alta impedância. Desta forma, todos os circuitos associados ao banco são desligados, evitando o consumo de energia estática e dinâmica.

A Figura 6.9 ilustra a memória interna dividida em quatro regiões, uma para cada configuração. A região em laranja, que envolve todos os bancos de memória, é responsável por armazenar os dados quando a configuração *48-7bpp* é utilizada, nesse caso nenhum banco é desligado. A região em lilás, a qual envolve 28 bancos de memória com 512B cada um, é responsável por armazenar os dados da área de busca quando a configuração *32-7bpp* é utilizada. Nesse caso, 16 bancos completos são desligados. A configuração *16-7bpp*, destacada pela cor verde na figura, utiliza 16 bancos de memória e nesse caso, quando em uso, são desligados os 28 bancos. A configuração *16-4bpp*, cor azul na figura, utiliza 9 bancos de memória, sendo o restante dos bancos desligados quando em uso. O controle do MERSy é responsável pelo desligamento completo dos módulos através do *power gating*. Definida a configuração que estará em uso, o controle emite sinal

Figura 6.9: Arquitetura da memória interna composta por 44 bancos com 512B de capacidade cada.



Fonte: Próprio autor.

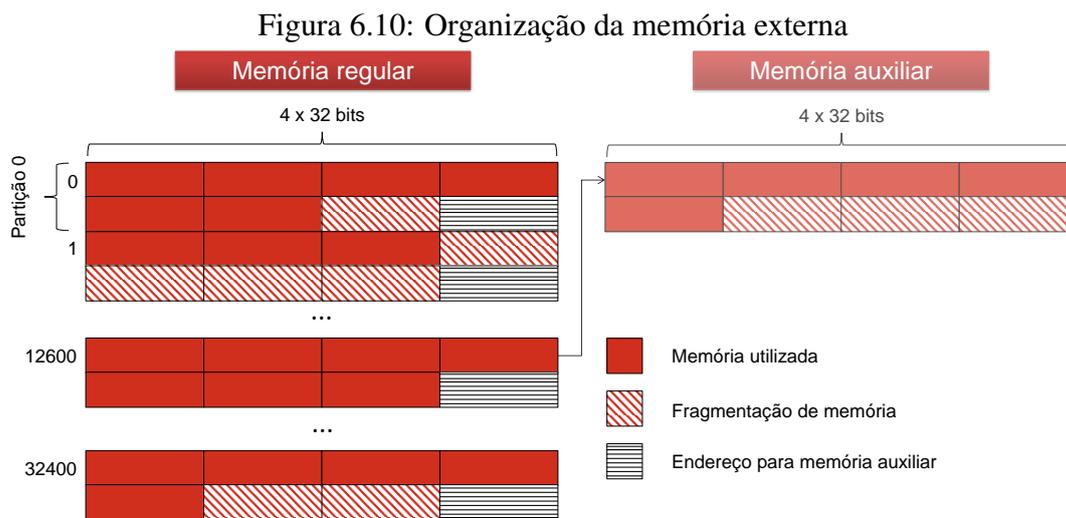
para desabilitar os bancos de memória que não serão utilizados.

Além disso, o Level C é implementado nos bancos de memória, independentemente da configuração que será utilizada. Entretanto, para que o reuso de dados funcione de maneira eficiente a SPM deve trabalhar como um *buffer* circular, onde uma *flag* indica o banco de memória com o início da área de busca. Essa sinalização evita que o Level C reescreva os dados que serão reusados cada vez que uma nova área de busca é processada. O controle do MERSy mantém a informação de qual banco de memória é o início da área de busca.

Todas as análises e o projeto da memória interna foram realizados a despeito da tecnologia de memória empregada. Nesse sentido, a Seção 6.4 irá apresentar e discutir os resultados de consumo de energia e largura de banda para diferentes tecnologias de memória, considerando a organização da memória interna proposta aqui e da memória externa que será apresentada na próxima seção.

### 6.3.2 Organização da memória externa

Os algoritmos rápidos de busca de ME podem realizar pesquisas em diferentes direções dentro dos quadros de referência. Entretanto, nos experimentos realizados nessa tese, a predição do TZS foi limitada ao bloco colocalizado, uma vez que essa estratégia favorece implementações em hardware e o uso de estratégias eficientes de reutilização de dados. Mesmo com essa estratégia este sistema requer uma organização de memória



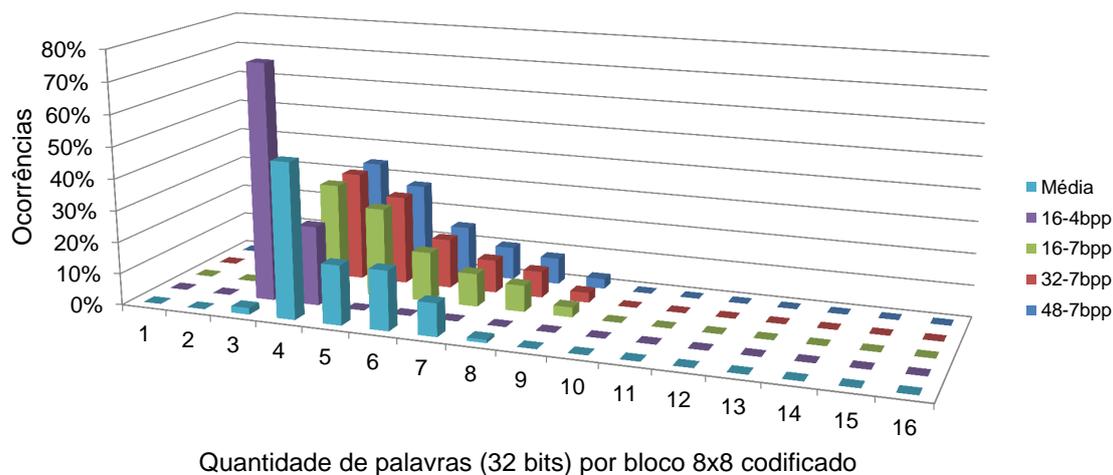
externa capaz de suportar o acesso aleatório aos blocos codificados, a fim de atender às necessidades da ME.

O tamanho de bloco utilizado pelo MERSy foi definido em  $8 \times 8$  amostras, sendo esse o menor tamanho de PU utilizado no HEVC. Esse tamanho de bloco facilita o acesso aleatório a pequenas partes do quadro de referência, otimizando o reuso de dados com o Level C na memória interna, facilitando a integração com o núcleo de processamento do TZS, sem custos adicionais com memórias temporárias. Além disso, permite altas taxas de compressão por parte do compressor de quadros de referência.

A Figura 6.10 apresenta a organização proposta para a memória externa. Essa organização de memória externa já foi proposta em um trabalho anterior do autor publicado em Silveira et al. (2019). Nessa organização, duas regiões da memória externa são usadas para endereçamento: a memória regular e a memória auxiliar. Quando utilizado um quadro de referência de resolução HD 1080p, a memória regular é dividida em 32.400 partições, onde cada partição é usada para armazenar um bloco de  $8 \times 8$  amostras codificadas. Para a resolução UHD 2160p são necessárias 129.600 partições. Desta forma, todo o quadro de referência pode ser armazenado.

Cada partição é composta por 8 palavras de 32 bits, com 4 palavras por linha, num total de 2 linhas por partição. Dessas 8 palavras, 7 são utilizadas para armazenar as amostras codificadas e uma palavra é usada para armazenar o endereço de uma região na memória auxiliar, caso seja necessária. Quando o bloco codificado ultrapassa o limite de 7 palavras na memória regular, o excesso de informações é enviado à memória auxiliar. O endereço para esta área na memória auxiliar fica escrito na última palavra da partição da memória regular, representado pelos retângulos texturizados (linhas horizontais) na

Figura 6.11: Histograma das ocorrências de palavras de 32 bits nos blocos  $8 \times 8$  codificado



Fonte: Próprio autor.

Figura 6.10.

A memória auxiliar tem partições de tamanho variável, onde cada partição é composta por um número inteiro de linhas de 128 bits, sendo o tamanho mínimo de uma linha com 4 palavras de 32 bits. Assim, a fragmentação nessa memória é reduzida, uma vez que a memória auxiliar não tem tamanho fixo de partição.

A Figura 6.11 apresenta o histograma de palavras de 32 bits utilizadas pelos blocos codificados. O histograma foi gerado considerando as 12 sequências de vídeo utilizadas nos experimentos apresentados no capítulo anterior, considerando as quatro configurações do MERSy. Este estudo forneceu suporte para definir os tamanhos da memória externa e das partições. Na Figura 6.11, 95% dos blocos codificados utilizam no máximo 8 palavras de 32 bits. Essa avaliação considerou mais de 380 milhões de blocos para as duas resoluções, incluindo os quatro QPs.

Devido a isso, definiu-se que seriam utilizadas 8 palavras por partição, sendo 7 palavras para armazenamento de dados e a última para endereçamento da memória auxiliar, caso seja necessário. Desta forma, 91,4% dos blocos codificados são armazenados apenas na memória regular, e 8,6% dos blocos codificados usam alguma área na memória auxiliar. Se nenhuma técnica de compressão fosse utilizada, seriam necessárias 16 palavras de 32 bits para armazenar um bloco de  $8 \times 8$ .

Uma análise do pior caso teórico neste sistema foi realizada. Para isso, foi considerado que cada amostra codificada utiliza o tamanho máximo em bits para ser codificado. Por exemplo, no caso do DDRFC-BD a maior amostra codificada tem 14 bits de tama-

nho, 9 bits do resíduo mais 5 bits do código de exceção. Desta forma, todas as partições da memória regular seriam utilizadas por completo e mais uma área na memória auxiliar. No total, esse quadro em resolução HD 1080p ocuparia 3,5MB de memória externa. No entanto, esse tamanho não seria um problema em termos de perda de dados, nem de organização de memória externa, porque com a utilização da memória auxiliar, que tem partições de tamanho variável, estas informações podem ser facilmente alocadas. Vale salientar que este pior caso teórico nunca ocorreu em nenhum dos testes realizados.

Devido à utilização desse esquema que possibilita a expansão da memória externa, e por permitir acesso aleatório a blocos individuais, esta abordagem pode ser utilizada em um sistema com múltiplas unidades do compressor DDRFC-BD funcionando em paralelo. Desta forma, o desempenho do compressor pode aumentar, sem inserir um novo gargalo no sistema.

#### 6.4 Caracterização das memórias para diferentes tecnologias

As tecnologias de memórias discutidas no Capítulo 3 apresentam características e composição distintas e isso, como já era esperado, acaba se refletindo na performance dessas memórias. Nesse sentido, essa seção apresenta e discute resultados de consumo de energia para as tecnologias de memórias DRAM, SRAM e NV-RAMs, as quais irão compor o sistema de memória do MERSy. Para os resultados da DRAM foi utilizado a ferramenta *System Power Calculator* da Micron (MICRON, 2019) para o modelo de uma memória LPDDR2 (*Low Power Double Data Rate 2*). Para a SRAM foi utilizada a ferramenta Cacti da HP (CACTI, 2018) e para as NV-RAMs foi utilizada a ferramenta de simulação NVSim (DONG et al., 2012). Essas ferramentas são bastante utilizadas pelos trabalhos relacionados e também por outros trabalhos que analisam o uso de memórias nos mais diversos sistemas, tais como computadores de propósito geral, servidores, embarcados, IoT (*Internet-of-Things*) e PIM (*Processing-In-Memory*) (DONG et al., 2012), IMEC (2018), Eisenman et al. (2018), Xie et al. (2019), Wang et al. (2020).

Nesse experimento foram avaliadas quatro tecnologias diferentes para a memória externa, sendo elas DRAM, STT-MRAM, ReRAM e PCRAM, onde as três últimas são NV-RAMs. Para as memórias internas, também foram utilizadas quatro tecnologias SRAM, STT-MRAM, ReRAM e PCRAM. Essas memórias foram avaliadas para as quatro configurações propostas nessa tese, *48-7bpp*, *32-7bpp*, *16-7bpp* e *16-4bpp*, e para uma configuração *baseline*, i.e. sem cortes na área de busca e sem *bit dropping*, a qual foi

intitulada de *64-8bpp*.

Como visto na seção anterior, das quatro configurações que utilizam a computação aproximada três delas associam circuitos de *power gating* aos bancos de memórias e seus respectivos barramentos. Desta forma, quando o banco não for utilizado, tem sua alimentação interrompida, bem como a alimentação dos *buffers* em seu barramento. Com isso a capacidade de armazenamento e o consumo de energia para leitura e escrita nessas memórias são impactados. Assim, para observar os impactos do *power gating* no consumo de energia foram obtidos resultados de energia para cada uma das configurações citadas anteriormente, considerando cada uma das tecnologias de memória.

Os resultados de consumo de energia para as quatro tecnologias de memória testadas como memória externa podem ser observados na Tabela 6.5. Nessa tabela os resultados de leitura e escrita, segunda e terceira colunas respectivamente, são apresentados em termos de energia consumida por *byte* acessado ( $pJ/B$ ), a energia estática é dada em  $mW$  e todas as memórias externas tem  $1GB$  de capacidade.

Como pode ser observado na Tabela 6.5, o maior consumo de energia para operações de leitura são observadas na tecnologia DRAM, onde para cada byte acessado são gastos  $184,5pJ$  de energia. Esse custo energético de leitura da DRAM é  $10\times$  maior quando comparado as NV-RAMs, chegando a ser  $14\times$  maior na comparação com o consumo de energia da tecnologia ReRAM, a qual consome apenas  $12,6pJ$  por acesso de leitura.

Nas operações de escrita na memória externa o consumo de energia da tecnologia DRAM está mais próximo de algumas NV-RAMs como STT-MRAM e ReRAM, mas ainda assim é um custo quase  $5\times$  maior que essas outras duas tecnologias. Entretanto, o maior consumo de energia nas operações de escrita é observado na tecnologia PCRAM, a qual utiliza indução térmica para realizar a mudança de fase no material da célula de memória, que pode ser cristalino ou amorfo. Devido a isso, o processo de escrita na tecnologia PCRAM é mais lento que nas demais tecnologias e consome maior energia sendo até  $130\times$  maior quando comparado a tecnologia STT-MRAM e  $27\times$  maior que a tecnologia DRAM.

A Tabela 6.5 também apresenta resultados de dissipação de potência estática para as quatro tecnologias de memória. Nesse ponto, observa-se que a tecnologia DRAM apresenta o melhor resultado com a menor dissipação de potência com  $173,7mW$ . As tecnologias ReRAM e PCRAM apresentam resultados  $2,6\times$  maior que a tecnologia DRAM, enquanto que a tecnologia STT-MRAM atinge resultado  $3,7\times$  maior que a DRAM. Destacam-

Tabela 6.5: Resultados de consumo de energia das memórias externas.

<b>Tecnologia</b>	<b>Leitura</b> ( $pJ/B$ )	<b>Escrita</b> ( $pJ/B$ )	<b>Estática</b> ( $mW$ )	<b>Capacidade</b>
LPDDR2 DRAM	184,58	60,04	173,75	1GB
STT-MRAM	14,14	12,72	650,74	
ReRAM	12,67	17,28	447,08	
PCRAM	18,51	1653,00	447,20	

Fonte: Próprio autor.

se nesses resultados, o consumo de energia das tecnologias STT-MRAM e ReRAM, as quais apresentam os menores consumos de energia para leitura e escrita na memória quando comparadas as tecnologias DRAM e PCRAM. Assim, é possível que a utilização dessas tecnologias como memória externa na codificação de vídeo tragam reduções no consumo de energia total da aplicação. Na sequência do texto esses dados poderão ser confirmados quando combinados com os dados de acessos à memória durante a codificação dos vídeos.

Os resultados de consumo de energia para as memórias internas podem ser observados na Tabela 6.6. Nessa tabela, além dos resultados de energia para as 4 tecnologias de memória testadas, SRAM, STT-MRAM, ReRAM e PCRAM, também são apresentados os resultados para cada uma das quatro configurações que utilizam computação aproximada. Para fins de comparação foram adicionados resultados para uma configuração onde não é utilizada computação aproximada, chamada *64-8bpp*. Para essa configuração, a memória deve comportar uma área de busca com intervalo  $[-64, +64]$  com 8 bits por amostra. Além disso, foram adicionados resultados de variância das tecnologias, a fim de observar a dispersão dos resultados de energia ao longo das configurações.

A Tabela 6.6 apresenta os dados de consumo de energia para operações de leitura e escrita na memória interna, além da potência estática dissipada por essas memórias. Nota-se nos resultados dessa tabela, que quanto mais restrita a configuração menor é o consumo de energia relacionada a ela. Isso ocorre devido a utilização do *power gating* empregado nessas configurações, o qual atua no desligamento da alimentação das células de memória e dos barramentos à elas conectados.

Analisando os resultados pelas configurações, nota-se que com a utilização da computação aproximada e do *power gating*, o qual é empregado a medida que uma configuração mais restrita é utilizada, o consumo de energia e a dissipação de potência apresentam reduções. Essas reduções ocorrem principalmente no consumo de energia das operações de leitura e na dissipação de potência estática. Nesse sentido, considerando os

Tabela 6.6: Resultados de consumo de energia das memórias internas.

Configuração	Tecnologia	Leitura (pJ/B)	Escrita (pJ/B)	Estática (mW)	Capacidade (bytes)
64-8bpp	SRAM	1,54	0,47	72,48	36.864
	STT-MRAM	0,46	2,29	13,75	
	ReRAM	0,47	6,88	12,45	
	PCRAM	1,25	1621,25	7,55	
48-7bpp	SRAM	0,94	0,45	35,38	22.400
	STT-MRAM	0,26	2,24	8,42	
	ReRAM	0,35	5,87	6,69	
	PCRAM	0,64	1620,88	4,97	
32-7bpp	SRAM	0,43	0,39	19,50	14.336
	STT-MRAM	0,19	2,19	6,15	
	ReRAM	0,28	5,81	4,43	
	PCRAM	0,42	1620,63	2,90	
16-7bpp	SRAM	0,25	0,17	9,95	8.064
	STT-MRAM	0,17	2,13	3,26	
	ReRAM	0,18	5,30	2,67	
	PCRAM	0,22	1620,50	1,76	
16-4bpp	SRAM	0,16	0,15	5,14	4.608
	STT-MRAM	0,14	2,10	2,15	
	ReRAM	0,15	5,09	1,48	
	PCRAM	0,12	1620,50	1,08	
Variância	SRAM	0,263	0,019	590,389	-
	STT-MRAM	0,013	0,005	17,107	-
	ReRAM	0,013	0,386	15,005	-
	PCRAM	0,164	0,081	5,539	-

Fonte: Próprio autor.

resultados de consumo de energia para as operações de leitura na configuração 48-7bpp, a tecnologia PCRAM é a que apresenta a maior redução, 49%, em relação a configuração 64-8bpp. As demais tecnologias também apresentam altas taxas de redução para essa mesma configuração, sendo 39% na SRAM, 43% na STT-MRAM e 26% na ReRAM.

Essas taxas aumentam ainda mais quando consideramos a configuração de tamanho mais restrita, 16-4bpp. Nessa configuração a PCRAM apresenta uma redução de 91% para o consumo de energia das operações de leitura em relação ao método tradicional. Para as outras tecnologias essa redução foi de 89% na SRAM, 69% na STT-MRAM e 68% na ReRAM. No cálculo da variância entre os resultados de consumo de energia entre as configurações, as tecnologias STT-MRAM e ReRAM foram as que apresentaram o menor resultado. Isso mostra que independentemente da configuração utilizada o

consumo de energia para as operações de leitura dessas tecnologias são próximos. Os resultados mais dispersos são encontrados na SRAM, que apresentou o maior resultado de variância.

Na quarta coluna da Tabela 6.6 são apresentados os resultados de consumo de energia para as operações de escrita nas memórias internas. Diferentemente dos resultados de consumo de energia de leitura, nas operações de escrita a SRAM é a tecnologia que menos consome energia em todos os cenários avaliados. Destaca-se aqui, o alto consumo de energia da tecnologia PCRAM, o qual chega a ser  $10000\times$  maior que a SRAM. Comparando as tecnologias STT-MRAM e SRAM, o consumo da STT-MRAM é  $7\times$  maior que a SRAM, em média, e na tecnologia ReRAM essa diferença aumenta para  $19\times$ . No entanto, esses resultados apresentam baixa variação entre as configurações, onde a maior variância é observada na tecnologia ReRAM e a menor na tecnologia STT-MRAM com apenas 0,005.

Conforme pode ser observado na Tabela 6.6 os resultados de consumo de energia para operações de leitura na memória SRAM são maiores que nas memórias NV-RAMs, independentemente da configuração utilizada. Porém, nas memórias de maior tamanho, configurações *48-7bpp* e *32-7bpp*, esta diferença fica acima de 60% quando comparada as tecnologias STT-MRAM e ReRAM. Para as memórias menores, configurações *16-7bpp* e *16-4bpp*, a diferença diminui para aproximadamente 28%. Na média para as quatro configurações a tecnologia STT-MRAM apresenta resultados de consumo de energia para operações de leitura 57% menores que a tecnologia SRAM. A tecnologia ReRAM apresenta resultados 44% menores e a tecnologia PCRAM resultados de consumo de energia 21% menores para essa mesma operação quando comparadas a tecnologia SRAM.

Os resultados de dissipação de potência estática são apresentados na quinta coluna da Tabela 6.6. Esses resultados apresentam tendência semelhante aos resultados de consumo de energia para operações de leitura. Nesses resultados também há altas taxas de redução quando uma configuração mais restrita é utilizada. A configuração *48-7bpp* apresenta uma redução de 51% na tecnologia SRAM quando comparada ao método tradicional. Nessa configuração as NV-RAMs também apresentam altas taxas de redução, sendo 39% na STT-MRAM, 46% na ReRAM e 34% na PCRAM. Na configuração *16-4bpp* essas taxas de redução são maiores e chegam a 93% na SRAM, 84% na STT-MRAM, 88% na ReRAM e 86% na PCRAM.

Além disso, nota-se nos resultados que a maior dissipação de potência estática é encontrada na tecnologia SRAM. A potência dissipada pela SRAM é em média  $3,5\times$

maior que a potência dissipada pelas memórias na tecnologia STT-MRAM e  $4,5\times$  maior que a ReRAM. Entretanto, a maior diferença é encontrada para a tecnologia PCRAM, onde a SRAM apresenta dissipação de potência estática  $6,5\times$  maior. Estes resultados se devem ao fato de que a tecnologia SRAM possui células de memória formadas por 6 transistores enquanto que as NV-RAMs utilizam apenas 1 transistor.

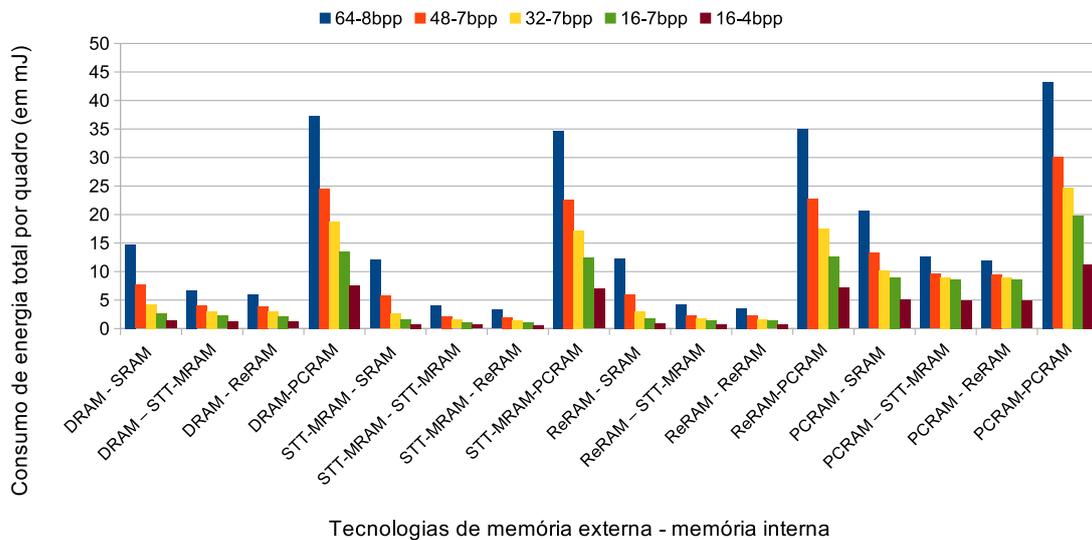
Como pode ser observado nas informações apresentadas nas Tabelas 6.5 e 6.6, a escolha das tecnologias para compor o sistema de memória depende também dos acessos à memória realizados pela aplicação, uma vez que não há uma tecnologia de memória melhor em todos os quesitos que foram avaliados. Desta forma, na próxima seção serão avaliados sistemas de memória compostos por memória interna e externa de diferentes tecnologias, contabilizando os acessos para vídeos 1080p e 2160p.

#### 6.4.1 Aplicação na codificação de vídeo

Essa seção apresenta resultados de avaliações da utilização do reuso de dados Level C, utilizando diferentes tecnologias de memória na hierarquia, incluindo as NV-RAMs: PCRAM, ReRAM e STT-MRAM, além das tradicionais DRAM e SRAM. Para obter esses resultados são utilizados os resultados de consumo de energia das tecnologias de memória apresentados na seção anterior e das informações dos *traces* para cada sequência de vídeo. Com as informações obtidas pelo *trace* foi contabilizado o número de acessos feitos a cada amostra na área de busca durante o processamento de 100 quadros de cada um dos 12 vídeos de testes, sendo 6 vídeos 1080p e 6 vídeos 2160p. As configurações do codificador de vídeo HM para extração dos *traces* são as mesmas apresentadas no Capítulo 5.

O *trace* gerado a partir dos dados acessados pela ME representam tanto os acessos internos quanto o número de acessos externos na memória. Com esses *traces*, é possível estimar com precisão o custo da memória interna e dos acessos à memória externa realizados pelo codificador de vídeo. O conjunto de 12 vídeos utilizadas para gerar os *traces* é composto por 6 vídeos HD 1080p: *BasketballDrive*, *BQTerrace*, *Cactus*, *Kimono*, *ParkScene* e *Tennis* e 6 vídeos UHD 2160p: *CampFire*, *CatRobot*, *DaylightRoad*, *FoodMarket*, *ParkRunning* e *Tango*. O *trace* considerou 100 quadros de cada vídeo para quatro diferentes QPs 22, 27, 32 e 37. As sequências de teste usadas nos experimentos apresentam diferentes características de movimento e brilho. Assim, uma ampla variação de diferentes decisões deve ser feita pelo codificador.

Figura 6.12: Resultados de consumo de energia total (dinâmica e estática) por quadro (em  $mJ$ ) para todas as combinações de tecnologias de memória para o reuso de dados Level C.



Fonte: Próprio autor.

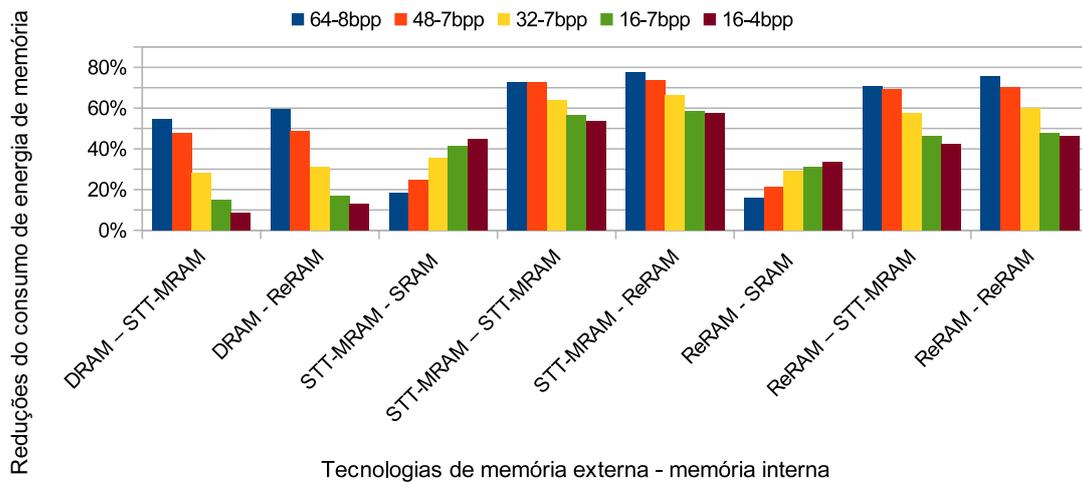
Esta avaliação considerou 16 configurações de hierarquia para o Level C, onde para cada uma dessas configurações foram testadas diferentes tecnologias de memória, DRAM, SRAM e NV-RAMs. Além disso, cada uma dessas hierarquias foram aplicadas nas quatro configurações de computação aproximada ( $48-7bpp$ ,  $32-7bpp$ ,  $16-7bpp$  e  $16-4bpp$ ) e na configuração tradicional, representada pela configuração  $64-8bpp$ , totalizando 80 diferentes sistemas de memória. Dessa forma, para cada *trace* de vídeo gerado essas configurações foram testadas.

A Figura 6.12 apresenta os resultados do consumo de energia por quadro codificado (em  $mJ$ ) para essas 80 configurações. Além disso, os resultados das configurações estão agrupados de acordo com a tecnologia de memória utilizada. Desta forma, cada um dos resultados apresentados nessa figura corresponde ao valor da média de consumo de energia para os 4 QPs dos 12 vídeos de teste. Assim, os resultados mais baixos representam configurações mais eficientes.

Nota-se na Figura 6.12, que dentro de cada composição, memória externa e memória interna, os maiores consumos de energia estão relacionados as configurações que mantêm mais bancos de memória ligados. Esse fato era esperado, uma vez que, de acordo com as Tabelas 6.5 e 6.6, o consumo de energia e a dissipação de potência são reduzidas nas configurações mais restritas, devido a redução na utilização de bancos de memória.

Além disso, é possível observar na Figura 6.12 que a utilização da tecnologia PCRAM como memória interna, independentemente da tecnologia da memória externa utilizada, gera os maiores resultados de consumo de energia. Ainda, a utilização da tec-

Figura 6.13: Comparativo entre o reuso de dados Level C tradicional (DRAM e SRAM) e o Level C NV-RAM excluindo resultados de configurações com PCRAM.



Fonte: Próprio autor.

nologia PCRAM como memória externa, também apresentou resultados de consumo de energia maiores que as demais configurações. Esses resultados se devem ao fato de que a tecnologia PCRAM apresenta resultados de consumo de energia para operações de escrita muito maiores que as demais tecnologias, ainda que na codificação de vídeo sejam realizados muito mais acessos de leitura do que os de escrita nas memórias.

Nota-se também na Figura 6.12 que todas as configurações que utilizaram a tecnologia PCRAM obtiveram resultados inferiores aos do *baseline* Level C, composto pelas tecnologias DRAM e SRAM. Desta forma, aumentado o consumo de energia do sistema em avaliação. Devido a esse desempenho insatisfatório da PCRAM e para melhorar a visualização das informações das outras tecnologias, a Figura 6.13 apresenta um comparativo dos resultados de consumo de energia entre o *baseline* Level C e as configurações de Level C que utilizam alguma NV-RAM em sua composição, sem contar com a PCRAM.

A Figura 6.13 apresenta os ganhos em redução de consumo de energia das configurações NV-RAM Level em relação ao *baseline* Level C. Nota-se nessa figura, que a utilização de NV-RAMs em apenas uma das memórias já introduz ganhos em redução de consumo de energia. Nas duas configurações DRAM - STT-MRAM e DRAM - ReRAM, a SRAM foi substituída pelas NV-RAMs. Na configuração DRAM - STT-MRAM os ganhos em energia variaram de 14% a 56%, enquanto que na configuração DRAM - ReRAM os ganhos em energia foram ligeiramente menores de 9% a 52%. Ainda, o ganho em energia é menor nas configurações mais restritas, pois o consumo de energia da SRAM fica próximo ao consumo de energia das NV-RAMs quando configurações com menor tamanho de memória são utilizadas.

Além disso, nota-se na Figura 6.13 um comportamento distinto dos demais nos resultados das configurações que utilizam a SRAM como memória interna nos reusos Level C junto as tecnologias STT-MRAM e ReRAM. Nessas duas abordagens de reuso de dados, quanto menor a memória interna maior é a redução de consumo de energia em relação ao *baseline* Level C. Isso ocorre porque nas configurações mais restritas o menor consumo de energia relacionado à memória interna, evidencia os ganhos da utilização das NV-RAMs como memória externa, principalmente, para esses dois casos, ganhos relacionados à escrita na memória externa, uma vez que o reuso de dados não reduz acessos de escrita nessa memória.

As quatro abordagens de reuso de dados que apresentam os maiores ganhos em energia são aquelas formadas exclusivamente por NV-RAMs, utilizando STT-MRAM e ReRAM tanto na memória externa quanto na memória interna. A utilização de STT-MRAM em ambas as memórias gerou ganhos em energia que variaram de 40% a 70% quando comparados ao *baseline* Level C. Na estratégia de reuso com as tecnologias STT-MRAM como memória externa e ReRAM como memória interna apresentou ganhos de energia de 45% a 75%. Quando a ReRAM foi utilizada em ambas memórias o ganho em energia foi de 50% a 74%. E na estratégia de reuso com ReRAM na memória externa e STT-MRAM na memória interna os ganhos de energia variaram de 55% a 77%, sendo essa combinação de tecnologias a que apresentou os melhores resultados. Apesar disso, as outras estratégias com NV-RAMs apresentaram resultados bastante próximos a esses, sendo também bons candidatos a compor sistemas de memórias.

Quando considerado os resultados apenas para as configurações com redução de área de busca a estratégia formada por ReRAM - STT-MRAM gerou ganhos de energia de 74% para a configuração *48-7bpp*, 66% para a *32-7bpp*, 58% para a *16-7bpp* e de 55% para a configuração *16-4bpp*. Devido a esse bom desempenho, a hierarquia de memória do MERSy será composta pelas tecnologias ReRAM na memória externa e STT-MRAM para memória interna. No próximo capítulo os resultados completos para o MERSy serão apresentados e discutidos.

## 7 RESULTADOS E DISCUSSÕES

Nesse capítulo serão apresentados e discutidos os principais resultados da tese, demonstrando através do MERSy que a utilização em conjunto das várias técnicas exploradas nessa tese, possibilitam alta redução de consumo de energia com baixo impacto na eficiência de codificação. Para isso, o MERSy utilizou reuso de dados Level C, composto por memórias NV-RAM *scratchpad* com a tecnologia STT-MRAM e memória externa também NV-RAM de tecnologia ReRAM. Os quadros de referência, que são armazenados nessas memórias durante a codificação do vídeo, são comprimidos de maneira eficiente através do DDRFC-BD. Além disso, o MERSy emprega computação aproximada com a utilização de *bit dropping* nas amostras dos quadros de referência e *power gating* nos bancos das memórias internas. Além das várias técnicas que são utilizadas em conjunto, o sistema utiliza quatro níveis de configuração, *48-7bpp*, *32-7bpp*, *16-7bpp* e *16-4bpp*, os quais permitem a redução do tamanho da área de busca, gerando uma economia ainda maior no consumo de energia do codificador de vídeo.

Para gerar esses resultados, o conjunto de vídeos de teste foi ampliado. Nas primeiras avaliações realizadas e discutidas no Capítulo 5 foram utilizadas 12 sequências de vídeo, sendo 6 sequências HD 1080p e 6 sequências UHD 2160p. Agora, para os resultados finais esse conjunto foi ampliado para 24 sequências de vídeo, sendo 12 vídeos HD 1080p: *BasketballDrive*, *BQTerrace*, *Cactus*, *Kimono*, *ParkScene*, *Tennis*, *BlueSky*, *Rolling*, *RushHour*, *Sunflower*, *Tractor* e *Traffic*; e 12 vídeos UHD 2160p: *CampFire*, *Ca-tRobot*, *DaylightRoad*, *FoodMarket*, *ParkRunning*, *Tango*, *Beauty*, *Bosphorus*, *CityAlley*, *HoneyBee*, *ShakeNDry* e *Twilight*. Para os obter os resultados foram utilizados 100 quadros de cada sequência para cada um dos quatro QPs. No Apêndice D são apresentados um quadro e uma descrição de cada uma dessas sequências de vídeo.

Na Seção 7.1 são apresentados e discutidos os resultados de redução de acessos às memórias externa e interna para as quatro configurações avaliadas no MERSy. Os resultados de redução no consumo de energia para essas configurações são apresentados na Seção 7.2. Na Seção 7.3 são avaliados os impactos em eficiência de codificação gerados pela utilização da computação aproximada empregada nessas configurações. Por último, na Seção 7.4, todos resultados obtidos para as quatro configurações são comparados a 12 trabalhos estado da arte encontrados na literatura.

## 7.1 Reduções de acessos às memórias externa e internas

A redução de acessos à memória externa ocorre com a utilização de três estratégias abordadas nessa tese, sendo elas: o reuso de dados Level C, a redução da área de busca e a compressão dos quadros de referência, sendo que as duas últimas estratégias também impactam nos acessos à memória interna. Além disso, com a compressão dos quadros de referência são reduzidos os acessos tanto no processo de leitura quanto de escrita de dados na memória externa.

A Tabela 7.1 apresenta os resultados de taxa de compressão obtidas pelo compressor DDRFC-BD. Para calcular as taxas de compressão foi utilizada a Equação 4.1. Os resultados são apresentados por vídeo, onde cada valor representa a média para os quatro QPs, e por QP, sendo o valor para cada QP a média de todos os vídeos que compõem aquela resolução. Esses resultados são apresentados para cada uma das configurações utilizadas no MERSy. Nota-se na primeira parte da tabela, que a taxa média de compressão para os vídeos de resolução HD 1080p que utilizam configurações com 7 bits por amostra foi de 65% e para a configuração *16-4bpp*, a qual utiliza 4 bits por amostra, foi de 81,6%. Essa melhora na taxa de compressão deve-se aos valores de resíduos mais próximos a “0” gerados por essa configuração. Fato esse que pode ser observado na Figura 6.4 da Seção 6.2.2.

Na segunda parte da Tabela 7.1 são apresentados os resultados de taxa de compressão para vídeos de resolução UHD 2160p. Nota-se nesses resultados que a taxa de compressão, assim como nos vídeos HD 1080p, para as configuração com 7 bits por amostra apresentaram resultados bastante semelhantes, com médias de 73,9% para as 12 sequências de vídeo. Na configuração *16-4bpp* a taxa de compressão aumentou para 83,8%, atingindo valores próximos a taxa de compressão máxima do DDRFC-BD, que é de 86%.

Desta forma, nota-se nesses resultados que a redução na área de busca interfere pouco na taxa de compressão, uma vez que as três configurações, *48-7bpp*, *32-7bpp* e *16-7bpp*, apresentaram resultados muito semelhantes. Entretanto, a utilização do *bit dropping* gera um impacto significativo na taxa de compressão. Isso fica ainda mais evidente quando observados os resultados das configurações *16-7bpp* e *16-4bpp*, as quais apresentam diferenças apenas na quantidade de bits por amostra. Enquanto que na configuração *16-7bpp* a taxa de compressão foi de 65% para vídeos HD 1080p e de 73,9 para vídeos UHD 2160p, na configuração *16-4bpp* essas taxas aumentaram para 81,6% e 83,8% nas

Tabela 7.1: Resultados de taxas de compressão obtidas pelo módulo DDRFC-BD para as 4 configurações do MERSy, utilizando 24 sequências de vídeo.

<b>Vídeos</b>		<b>48-7bpp</b>	<b>32-7bpp</b>	<b>16-7bpp</b>	<b>16-4bpp</b>
BasketballDrive		61,6%	61,6%	61,6%	80,5%
BQTerrace		51,4%	51,3%	51,3%	77,3%
Cactus		60,4%	60,4%	60,3%	80,6%
Kimono		71,6%	71,6%	71,6%	83,2%
ParkScene		56,0%	56,0%	56,1%	80,3%
Tennis		70,9%	71,0%	71,1%	83,6%
BlueSky		64,2%	64,1%	64,2%	81,3%
Rolling		78,4%	78,4%	78,4%	84,9%
RushHour		75,4%	75,3%	75,4%	84,0%
Sunflower		73,2%	73,2%	73,2%	83,5%
Tractor		54,0%	54,0%	54,1%	79,3%
Traffic		63,4%	63,4%	63,5%	81,0%
<b>Média</b>		65,03%	65,03%	65,06%	81,63%
<b>Desvio padrão</b>	1080p	8,91%	8,91%	8,92%	2,22%
CampFire		74,0%	74,1%	74,1%	84,2%
CatRobot		72,5%	72,5%	72,5%	83,7%
DaylightRoad		68,2%	68,3%	68,3%	82,9%
FoodMarket		79,0%	79,0%	79,0%	84,9%
ParkRunning		63,3%	63,3%	63,4%	81,6%
Tango		76,6%	76,6%	76,6%	84,5%
Beauty		74,4%	74,4%	74,5%	84,3%
Bosphorus		74,5%	74,5%	74,5%	83,0%
CityAlley		78,8%	78,7%	78,7%	85,1%
HoneyBee		76,6%	76,5%	76,5%	84,5%
ShakeNDry		71,6%	71,6%	71,6%	82,8%
Twilight		77,3%	77,3%	77,3%	84,9%
<b>Média</b>		73,90%	73,89%	73,91%	83,87%
<b>Desvio padrão</b>	2160p	5,42%	5,42%	5,41%	1,23%
	1080p	61,89%	61,90%	61,90%	81,01%
QP 22	2160p	70,43%	70,44%	70,44%	83,21%
	Todos	66,16%	66,17%	66,17%	82,11%
	1080p	64,39%	64,39%	64,42%	81,49%
QP 27	2160p	74,02%	74,01%	74,04%	83,88%
	Todos	69,20%	69,20%	69,23%	82,69%
	1080p	65,96%	65,95%	66,01%	81,82%
QP 32	2160p	75,16%	75,15%	75,15%	84,11%
	Todos	70,56%	70,55%	70,58%	82,96%
	1080p	67,88%	67,86%	67,93%	82,20%
QP 37	2160p	75,99%	75,97%	75,99%	84,27%
	Todos	71,93%	71,92%	71,96%	83,24%
<b>Média</b>		69,47%	69,46%	69,49%	82,75%
<b>Desvio padrão</b>	Todos	8,60%	8,60%	8,60%	2,12%

Fonte: Próprio autor.

respectivas resoluções.

Na terceira parte dos resultados da Tabela 7.1, onde os resultados são apresentados por QP, nota-se que quanto maior o QP maior é a taxa de compressão atingida. Isso se deve a um efeito semelhante ao causado pelo *bit dropping*, uma vez que com maior valor de QP maiores serão as perdas inseridas durante o processo de codificação do vídeo, gerando vídeos de menor qualidade com menos detalhes, os quais acabam facilitando a compressão. Desta forma, vídeos codificados com QP 22 atingem uma taxa de compressão média para as duas resoluções de 66% nas configurações com 7 bits por amostra e de 82% na configuração *16-4bpp*. Enquanto que os vídeos codificados com QP 37 atingem uma taxa de 71,9% nas configurações com 7 bits por amostra e 83,2% na configuração *16-4bpp*. O crescimento da taxa de compressão na configuração *16-4bpp* é sempre menor quando comparado as outras configurações, pois essa configuração já apresenta altas taxas de compressão desde o início, restando poucas possibilidades de aumento até o limite máximo da taxa de compressão.

Outro ponto a ser observado na Tabela 7.1 é a maior taxa de compressão obtida nos vídeos de resolução UHD 2160p quando comparado aos vídeos HD 1080p. Isso ocorre devido a maior similaridade entre as amostras dos blocos, uma vez que com maiores resoluções as cenas são representadas por maiores quantidades de amostras. Com esses resultados, a taxa de compressão média geral obtida pelo DDRFC-BD foi de 69,4% para as configurações *48-7bpp*, *32-7bpp* e *16-7bpp*, e de 82,7% para a configuração *16-4bpp*. Esses resultados de taxa de compressão irão compor junto com as outras estratégias, redução da área de busca e Level C, os resultados de redução de largura de banda de memória externa, os quais são apresentados na Tabela 7.2.

A Tabela 7.2 apresenta os resultados de redução de largura de banda de memória para as quatro configurações utilizadas no MERSy. Esses valores são resultados da redução em relação ao sistema *baseline*, o qual representa um sistema que utiliza intervalo de busca  $[-64, +64]$  com 8 bits por amostra e reuso de dados Level C. Desta forma, os resultados da Tabela 7.2 representam o ganho em redução da largura de banda de memória em relação a essa configuração *baseline*. Os resultados para a memória externa (Ext.) e para a memória interna (Int.) são apresentados por vídeo sendo a média ( $\bar{x}$ ) para os quatro QPs, por resolução do vídeo, representado pelo valor da média de todos os vídeos daquela resolução para os quatro QPs, e por QP com as médias de todos os vídeos. Além disso, são apresentados os resultados de desvio padrão ( $\sigma$ ) para esses conjuntos.

Os resultados de redução de largura de banda de memória apresentados na Tabela

Tabela 7.2: Resultados de redução de acessos às memórias externa e interna para as 4 configurações do MERSy.

Vídeos	48-7bpp		32-7bpp		16-7bpp		16-4bpp		
	Ext.	Int.	Ext.	Int.	Ext.	Int.	Ext.	Int.	
BasketballDrive	66,9%	30,5%	72,0%	46,5%	77,0%	66,1%	88,3%	80,1%	
BQTerrace	58,0%	19,7%	64,5%	29,3%	70,8%	43,4%	86,4%	60,7%	
Cactus	65,8%	20,2%	71,1%	30,0%	76,2%	44,1%	88,4%	48,6%	
Kimono	75,5%	26,7%	79,3%	40,5%	83,0%	58,5%	89,9%	73,1%	
ParkScene	62,1%	22,5%	67,9%	32,9%	73,6%	46,9%	88,2%	58,0%	
Tennis	74,9%	30,3%	78,8%	46,7%	82,6%	66,3%	90,2%	79,9%	
BlueSky	69,1%	12,2%	73,8%	33,2%	78,5%	57,8%	88,8%	64,2%	
Rolling	81,3%	16,4%	84,2%	35,1%	87,0%	56,9%	90,9%	28,2%	
RushHour	78,7%	13,7%	82,0%	28,4%	85,2%	47,6%	90,4%	25,5%	
Sunflower	76,9%	19,9%	80,4%	37,2%	83,9%	59,0%	90,1%	59,3%	
Tractor	60,3%	18,1%	66,4%	35,4%	72,4%	58,5%	87,6%	57,3%	
Traffic	68,4%	19,3%	73,3%	37,7%	78,1%	60,0%	88,6%	59,8%	
$\bar{x}$	1080p	69,8%	20,8%	74,5%	36,1%	79,0%	55,4%	89,0%	57,9%
$\sigma$		7,7%	5,9%	6,5%	6,1%	5,4%	7,7%	1,3%	16,6%
CampFire	77,5%	31,1%	80,8%	47,3%	84,1%	65,5%	90,3%	78,6%	
CatRobot	76,1%	13,9%	79,7%	27,4%	83,2%	44,7%	90,0%	49,1%	
DaylightRoad	72,4%	28,2%	76,5%	43,8%	80,6%	62,7%	89,5%	76,6%	
FoodMarket	81,8%	28,8%	84,5%	44,3%	87,2%	63,2%	90,8%	77,3%	
ParkRunning	68,2%	29,5%	72,8%	45,0%	77,6%	64,2%	88,7%	79,4%	
Tango	79,6%	30,1%	82,7%	46,1%	85,7%	65,2%	90,5%	78,7%	
Beauty	77,8%	14,7%	81,1%	31,0%	84,4%	50,9%	90,4%	18,3%	
Bosphorus	77,8%	4,2%	81,1%	15,8%	84,4%	33,0%	89,6%	-5,8%	
CityAlley	81,5%	-14,4%	84,3%	-8,1%	87,0%	0,9%	90,9%	-103%	
HoneyBee	79,6%	-7,5%	82,7%	-3,6%	85,6%	2,0%	90,5%	-149%	
ShakeNDry	75,3%	15,0%	79,0%	30,0%	82,6%	49,6%	89,5%	13,2%	
Twilight	80,3%	-21,2%	83,2%	-9,7%	86,1%	3,0%	90,7%	-113%	
$\bar{x}$	2160p	77,3%	12,7%	80,7%	25,8%	84,0%	42,1%	90,1%	8,2%
$\sigma$		4,7%	19,2%	4,0%	21,7%	3,3%	25,2%	0,8%	83,8%
22	1080p	67,1%	19,2%	72,2%	34,5%	77,1%	54,2%	88,6%	58,4%
	2160p	74,3%	6,3%	78,1%	20,6%	81,9%	38,9%	89,7%	5,0%
	Todos	70,7%	12,8%	75,2%	27,5%	79,5%	46,5%	89,2%	31,7%
27	1080p	69,3%	19,6%	74,0%	35,0%	78,6%	54,6%	88,9%	57,4%
	2160p	77,4%	10,5%	80,8%	23,9%	84,1%	40,7%	90,1%	-0,6%
	Todos	73,4%	15,1%	77,4%	29,4%	81,4%	47,6%	89,5%	28,4%
32	1080p	70,6%	21,4%	75,2%	36,7%	79,6%	55,9%	89,1%	57,5%
	2160p	78,4%	15,2%	81,6%	27,7%	84,8%	43,1%	90,3%	0,5%
	Todos	74,5%	18,3%	78,4%	32,2%	82,2%	49,5%	89,7%	29,0%
37	1080p	72,3%	22,9%	76,6%	38,1%	80,7%	57,0%	89,3%	58,3%
	2160p	79,1%	18,7%	82,2%	30,9%	85,3%	45,6%	90,4%	28,0%
	Todos	75,7%	20,8%	79,4%	34,5%	83,0%	51,3%	89,8%	43,1%
$\bar{x}$	Total	73,6%	16,7%	77,6%	30,9%	81,5%	48,8%	89,5%	33,1%
$\sigma$		7,4%	14,7%	6,2%	16,8%	5,1%	19,8%	1,2%	65,3%

Fonte: Próprio autor.

7.2 refletem a utilização de quatro técnicas utilizadas em conjunto: a redução da área de busca, através das quatro configurações, o reuso de dados Level C, a compressão de quadros de referência com o DDRFC-BD e o *bit dropping*, sendo este último implementado junto com o compressor. Nota-se nessa tabela, que mesmo a configuração menos restritiva, *48-7bpp*, altas taxas de redução são alcançadas. A taxa de redução de acessos à memória externa para a configuração *48-7bpp* com vídeos HD 1080p foi em média de 69,8%, e para a resolução UHD 2160p essa taxa foi de 77,3%, nessa mesma configuração. Os vídeos de UHD 2160p apresentam maiores taxas de redução de largura de banda de memória externa quando comparado as taxas obtidas nos vídeos HD 1080p, assim como nos resultados de compressão apresentados anteriormente. Essa diferença apresentada nos resultados de largura de banda da memória externa se devem justamente as maiores taxas de compressão obtidas para a resolução UHD 2160p, uma vez que as outras técnicas independem da resolução do vídeo.

A Tabela 7.2 também apresenta os resultados de redução de acessos à memória interna. Nota-se que as taxas desses resultados são menores quando comparadas as taxas de redução de acessos à memória externa, a qual gera maior impacto em consumo de energia. Essas taxas não apresentam semelhança por dois motivos: i) o compressor de quadros de referência atua somente nos dados que são enviados e trazidos da memória externa, e assim, são armazenados de forma íntegra na memória interna; e ii) ao maior esforço computacional realizado durante a codificação do vídeo quando uma configuração com computação aproximada é utilizada. Esse maior esforço, principalmente devido a utilização do *bit dropping* mais agressivo e da redução da área de busca, geram maiores perdas na eficiência de codificação, levando o codificador a realizar mais acessos à memória interna para minimizar essas perdas. Essa maior quantidade de acessos é decorrente da maior quantidade de blocos comparados pelo TZS para encontrar a menor distorção entre o bloco atual e o bloco candidato. Os resultados de eficiência de codificação serão abordados na Seção 7.3.

Devido a esses dois fatores, nota-se que em 4 sequências de vídeo UHD 2160p, *Bosphorus*, *CityAlley*, *HoneyBee* e *Twilight*, houveram mais acessos à memória interna com a utilização das 4 configurações testadas no MERSy, que quando utilizada a configuração com o *baseline*, resultados negativos na Tabela 7.2. Com isso, as taxas médias de redução de acessos à memória interna são sempre menores nos vídeos UHD 2160p, sendo que a maior diferença é encontrada na configuração *16-4bpp*. Nessa configuração, a média de redução de acessos à memória interna para os vídeos UHD 2160p é de 8,2%,

enquanto que para os vídeos HD 1080p essa taxa atinge 57,9%.

Nos resultados para as configurações apresentados na Tabela 7.2, sempre que uma configuração mais restrita é utilizada maior é a redução da largura de banda de memória externa. Isso era esperado, uma vez que há a diminuição do intervalo de busca e com isso menos informações precisam ser trazidas da memória externa para a memória interna. Nesse sentido, a configuração 48-7bpp atingiu uma taxa de redução de 69,8% para vídeos HD 1080p, enquanto que as demais configurações alcançaram uma redução média de 74,5%, 79% e 89% para as configurações 32-7bpp, 16-7bpp e 16-4bpp, respectivamente, para essas mesmas sequências de vídeo. Nos vídeos de resolução UHD 2160p as taxas foram de 77% para a configuração 48-7bpp, 80% para a 32-7bpp, 84% na configuração 16-7bpp e de 90% na 16-4bpp, sendo essa a configuração que atinge as maiores taxas de redução de largura de banda com o menor desvio padrão entre os resultados.

Avaliando os resultados da Tabela 7.2 por QP, observa-se que a menor taxa de redução de largura de banda é encontrada no QP 22 da configuração 48-7bpp que foi de 70,7% em média para todos os vídeos avaliados. Entretanto, apesar de ser a menor taxa entre todas as demais configurações representa uma grande redução, uma vez que esse valor representa um ganho em relação ao *baseline* Level C, o qual é bastante utilizado entre os trabalhos da literatura. Com o aumento do QP, a taxa de redução também aumenta, atingindo 89,8% na configuração 16-4bpp para o QP 37. Na média geral para todos os vídeos, considerando os quatro valores de QPs a taxa de redução de largura de banda de memória na configuração 48-7bpp foi de 73,6%, na configuração 32-7bpp de 77,6%, na 16-7bpp foi de 81,5% e 89,5% na configuração 16-4bpp. Desta forma, mostrando que a utilização dessas abordagens em conjunto geram altas taxas de redução de largura de banda de memória externa em codificadores de vídeo, impactando diretamente o consumo de energia do codificador, conforme será mostrado na próxima Seção.

## 7.2 Redução do consumo de energia

Nesta seção serão apresentados e discutidos os resultados de redução de consumo de energia relacionados a memória. Para gerar esses resultados foram utilizadas em conjunto todas as técnicas abordadas nessa tese, sendo elas: redução da área de busca, compressão de quadros de referência com o DDRFC-BD, *bit dropping*, implementado junto ao compressor, reuso de dados Level C, memória NV-RAMs na hierarquia de memória e *power gating* nos bancos da memória interna. Além disso, esse conjunto de técnicas foi

avaliado para quatro configurações, as quais fazem parte do MERSy, sistema desenvolvido para avaliar a utilização conjunta dessas técnicas.

A Tabela 7.3 apresenta os resultados de redução de consumo de energia para as quatro configurações MERSy, sendo que esses resultados representam a redução em relação ao sistema *baseline*. Esse sistema *baseline* utiliza intervalo de busca  $[-64, +64]$  com 8 bits por amostra, reuso de dados Level C, utilizando memórias DRAM como memória externa e SRAM para a memória interna. Os resultados são apresentados por vídeo sendo a média para os quatro QPs, por resolução do vídeo, representado pelo valor da média de todos os vídeos daquela resolução para os quatro QPs, e por QP com as médias de todos os vídeos.

Conforme pode ser visto na Tabela 7.3, a taxa de redução de consumo de energia para vídeos HD 1080p é de 87,5% na configuração 48-7bpp, chegando até 96,2% na configuração 16-4bpp com um crescimento linear entre as configurações. Para vídeos UHD 2160p, a diferença no resultado entre as configurações é um pouco menor, sendo que a configuração 48-7bpp atinge uma taxa de 88,5% e a configuração 16-4bpp atinge 95% de redução no consumo de energia. Além disso, o desvio padrão dos resultados obtidos pelos vídeos são em maioria inferiores a 1%, apenas a configuração 16-4bpp para os vídeos UHD 2160p apresentou uma taxa superior com 1,9%.

Esses dois pontos abordados anteriormente sobre os resultados para os vídeos UHD 2160p, i) taxa de redução de consumo energia inferior aos vídeos HD 1080p e ii) maior desvio padrão, que são evidenciados na configuração 16-4bpp, se devem ao maior esforço computacional realizado durante a codificação do vídeo quando essa configuração é utilizada. Como explicado na seção anterior, esse maior esforço computacional leva o codificador a realizar mais acessos à memória interna para minimizar as perdas geradas.

Seguindo na Tabela 7.3, nota-se que os resultados entre os QPs são bastante parecidos, havendo pouco melhoria em relação ao consumo de energia de memória quando um determinado QP é utilizado. Esse mesmo comportamento também pode ser observado nos resultados de redução de largura de banda de memória externa. Entretanto, quando avaliado as diferentes configurações para o mesmo QP, percebe-se uma melhoria significativa nos resultados de energia, gerando em média 2,5% a mais de economia de energia a cada configuração mais restrita utilizada.

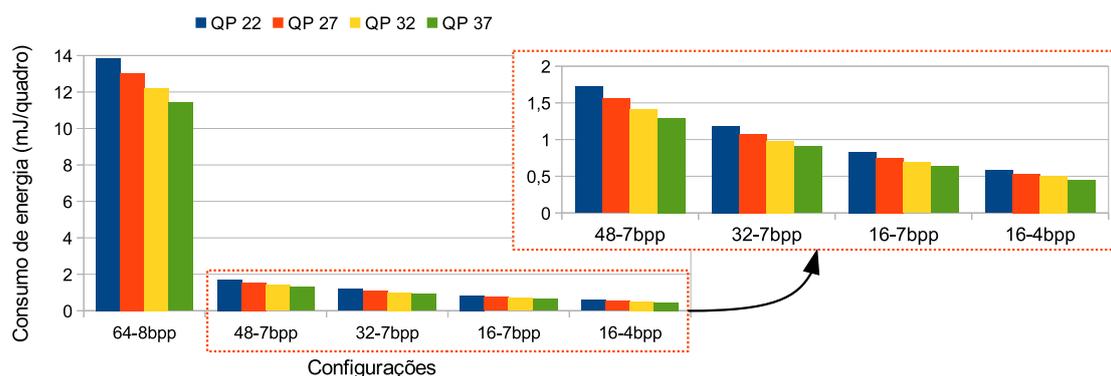
Desta forma, percebe-se na Tabela 7.3, que a utilização conjunta das técnicas apresentadas nessa tese foi altamente benéfica para a redução do consumo de energia do codificador de vídeo, uma vez que a menor taxa redução é encontrada na configuração 48-7bpp

Tabela 7.3: Resultados de redução de consumo de energia da memória para as 4 configurações do MERSy.

Vídeos		<b>48-7bpp</b>	<b>32-7bpp</b>	<b>16-7bpp</b>	<b>16-4bpp</b>
BasketballDrive		88,1%	92,1%	94,9%	97,2%
BQTerrace		86,6%	89,6%	92,3%	95,8%
Cactus		87,5%	90,3%	93,0%	95,8%
Kimono		88,3%	92,0%	94,7%	96,9%
ParkScene		87,2%	90,0%	92,7%	95,9%
Tennis		88,5%	92,5%	95,3%	97,4%
BlueSky		86,5%	91,2%	94,4%	96,4%
Rolling		88,4%	92,0%	94,9%	95,3%
RushHour		88,0%	91,5%	94,3%	95,2%
Sunflower		87,5%	91,9%	95,0%	96,3%
Tractor		86,4%	91,0%	94,2%	96,0%
Traffic		86,9%	91,6%	94,7%	96,2%
<b>Média</b>		87,5%	91,3%	94,2%	96,2%
<b>Desvio padrão</b>	1080p	0,8%	0,9%	1,0%	0,7%
CampFire		89,1%	92,7%	94,9%	97,0%
CatRobot		88,1%	91,0%	93,1%	95,4%
DaylightRoad		88,2%	92,0%	94,4%	96,9%
FoodMarket		89,0%	92,9%	95,3%	97,2%
ParkRunning		88,0%	92,1%	94,6%	97,3%
Tango		88,9%	92,9%	95,4%	97,3%
Beauty		88,0%	91,6%	93,9%	94,1%
Bosphorus		87,5%	90,7%	92,9%	93,2%
CityAlley		89,5%	91,5%	93,1%	93,4%
HoneyBee		89,4%	91,2%	92,8%	92,9%
ShakeNDry		87,6%	91,1%	93,4%	93,7%
Twilight		89,0%	91,2%	92,9%	93,2%
<b>Média</b>		88,5%	91,7%	93,9%	95,1%
<b>Desvio padrão</b>	2160p	0,9%	0,8%	1,0%	1,9%
	1080p	87,1%	91,0%	94,0%	96,2%
QP 22	2160p	87,7%	91,2%	93,5%	94,9%
	<i>Todos</i>	87,4%	91,1%	93,7%	95,5%
	1080p	87,4%	91,2%	94,1%	96,2%
QP 27	2160p	88,4%	91,7%	93,9%	95,0%
	<i>Todos</i>	87,9%	91,5%	94,0%	95,6%
	1080p	87,6%	91,4%	94,3%	96,2%
QP 32	2160p	88,8%	91,9%	94,0%	95,1%
	<i>Todos</i>	88,2%	91,7%	94,1%	95,6%
	1080p	87,9%	91,5%	94,4%	96,2%
QP 37	2160p	89,2%	92,1%	94,1%	95,6%
	<i>Todos</i>	88,5%	91,8%	94,3%	95,9%
<b>Média</b>		88,0%	91,5%	94,0%	95,7%
<b>Desvio padrão</b>	<i>Total</i>	1,0%	0,9%	1,0%	1,5%

Fonte: Próprio autor.

Figura 7.1: Resultados de redução no consumo de energia entre as quatro configurações.



Fonte: Próprio autor.

para o QP 22 com 87%. Esses resultados atingem 96% de redução do consumo de energia para vídeos codificados com o QP 37 na configuração *16-4bpp*. Na média geral, os ganhos de redução de energia em relação a configuração *baseline* atingem taxas bastante significativas. A configuração *48-7bpp* atinge uma taxa de 88% de economia em energia, enquanto que a configuração *32-7bpp* atinge uma taxa de 91,5%. Na configuração *16-7bpp* essa taxa de redução aumenta para 94% e para a configuração *16-4bpp* os resultados médios de economia de energia chegam a de 95,7%.

Para deixar mais clara a contribuição nos ganhos de redução de consumo de energia de cada uma das configurações, foi elaborado o gráfico apresentado na Figura 7.1. Na Figura 7.1 são apresentados os resultados de consumo de energia (em *mJ*) para o sistema *baseline* e para as quatro configurações utilizadas pelo MERSy. Esses resultados representam o consumo médio por quadro codificado, considerando todos as 24 sequências de vídeo, onde cada uma das quatro barras representam os resultados de cada QP utilizado.

Desta forma, nota-se na Figura 7.1 que as quatro configurações apresentam altas taxas de redução no consumo de energia de memória quando comparadas a configuração *baseline*, resultados apresentados em detalhes na Tabela 7.3. Além disso, nota-se na Figura 7.1 que entre as quatro configurações também há uma redução significativa no consumo de energia relacionado à memória.

Conforme pode ser visto na Figura 7.1, a configuração *32-7bpp* apresenta um ganho médio em redução de consumo de energia de 30,7% em relação a configuração *48-7bpp*, de 1,5*mJ* para 1,04*mJ*. Da mesma forma, a configuração *16-7bpp* apresenta um ganho médio na redução do consumo de energia de 30,1% em média, quando comparado a configuração *32-7bpp*, de 1,04*mJ* para 0,73*mJ*. E por último nessa figura, são apresentados os resultados de ganho da configuração *16-4bpp* em relação a configuração

*16-7bpp*, esses ganhos atingem 28,9% em média, onde a configuração *16-7bpp* consumo  $0,73mJ$  por quadro codificado, enquanto que na configuração *16-4bpp* são consumidos  $0,52mJ$  por quadro codificado. Esses resultados são as médias para todos os QPs e todas as sequências de vídeo.

Assim, com base nos resultados apresentados nessa seção, percebe-se as altas taxas de redução no consumo de energia geradas pelas abordagens apresentadas nessa tese. Além disso, nota-se que as quatro configurações avaliadas no MERSy, trazem reduções significativas no consumo de energia relacionada à memória do codificador, possibilitando ao codificador o uso eficiente do consumo de energia. Na próxima seção serão apresentados os resultados de eficiência de codificação para as quatro configurações avaliadas no MERSy.

### 7.3 Eficiência de codificação

Os resultados de eficiência de codificação já foram discutidos parcialmente na Seção 5.4.3, durante os experimentos com as técnicas de computação aproximada. Naquela seção as técnicas de redução de área de busca e *bit dropping* tiveram seus resultados apresentados de forma individual e em abordagem conjunta, contribuindo para a definição das quatro configurações avaliadas pelo MERSy.

No entanto, naquela avaliação o conjunto de vídeos utilizados para avaliar os experimentos era menor, 12 sequências de vídeo no total. Para os resultados finais é utilizado um conjunto de vídeos com 24 sequências, sendo 12 HD 1080p e 12 UHD 2160p. Desta forma, essa seção apresentará os resultados de eficiência de codificação, medidos em BD-rate, para esse novo conjunto de vídeos. A Tabela 7.4 apresenta os resultados eficiência de codificação, para as quatro configurações MERSy. Os resultados são apresentados por vídeo e média para cada uma das resoluções.

A Tabela 7.4 apresenta os resultados de variação em BD-rate para as quatro configurações avaliadas no MERSy. Esses resultados de BD-rate são em relação ao sistema *baseline* com intervalo de busca  $[-64, +64]$ . Os resultados são apresentados para os 24 vídeos, sendo 12 vídeos HD 1080p e 12 vídeos UHD 2160p. Além da média para os 24 vídeos testados, também são apresentados os resultados de BD-rate médio para os vídeos HD 1080p e UHD 2160p. Para cada grupo de vídeo é apresentado o desvio padrão, bem como o desvio padrão para 24 sequências de vídeos.

Nota-se nos resultados apresentados na Tabela 7.4 que as menores variações em

Tabela 7.4: Resultados médios da variação de BD-rate do MERSy com as 4 configurações para as 24 sequências de vídeo.

<b>Vídeos</b>		<b>48-7bpp</b>	<b>32-7bpp</b>	<b>16-7bpp</b>	<b>16-4bpp</b>
BasketballDrive		0,01%	0,12%	0,68%	3,26%
BQTerrace		0,18%	0,09%	0,31%	0,84%
Cactus		0,03%	0,09%	0,21%	0,87%
Kimono		0,09%	0,06%	0,01%	2,45%
ParkScene		-0,10%	-0,05%	0,00%	0,86%
Tennis		0,01%	0,13%	1,04%	3,22%
BlueSky		0,30%	0,45%	0,32%	0,88%
Rolling		0,30%	0,61%	1,64%	5,92%
RushHour		0,08%	0,20%	0,19%	4,43%
Sunflower		0,05%	0,36%	0,24%	2,88%
Tractor		0,06%	0,09%	0,79%	2,61%
Traffic		0,30%	0,75%	1,36%	2,54%
<b>Média</b>		0,11%	0,24%	0,57%	2,56%
<b>Desvio padrão</b>	1080p	0,13%	0,24%	0,52%	1,51%
CampFire		0,02%	-0,04%	-0,01%	-0,04%
CatRobot		0,03%	0,00%	-0,07%	2,73%
DaylightRoad		0,31%	0,57%	2,72%	7,30%
FoodMarket		0,24%	0,68%	1,64%	9,90%
ParkRunning		-0,05%	-0,01%	0,95%	4,05%
Tango		0,17%	0,56%	1,77%	9,50%
Beauty		0,07%	0,17%	0,10%	1,49%
Bosphorus		0,19%	0,16%	0,14%	1,74%
CityAlley		-0,12%	-0,23%	0,42%	6,33%
HoneyBee		-1,23%	0,26%	-0,16%	3,28%
ShakeNDry		-0,15%	-0,14%	-0,19%	-0,23%
Twilight		0,01%	-0,14%	0,53%	3,13%
<b>Média</b>		-0,04%	0,15%	0,65%	4,10%
<b>Desvio padrão</b>	2160p	0,38%	0,29%	0,89%	3,29%
<b>Média</b>		0,03%	0,20%	0,61%	3,33%
<b>Desvio padrão</b>	Todos	0,30%	0,27%	0,73%	2,67%

Fonte: Próprio autor.

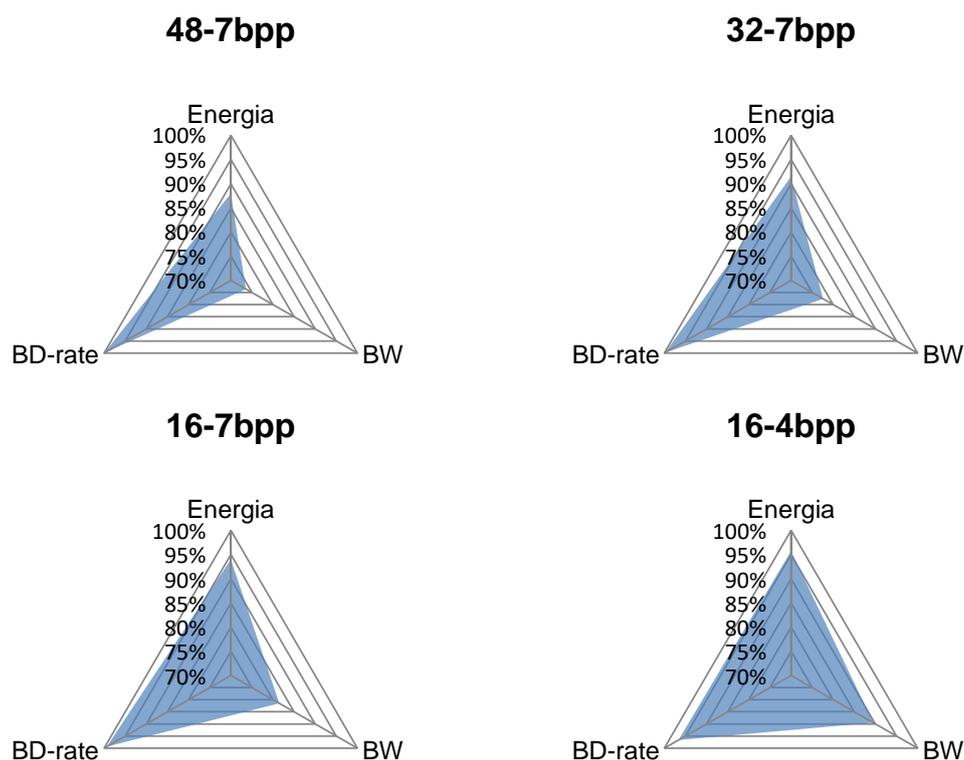
BD-rate estão, como esperado, na configuração *48-7bpp*, uma vez que essa configuração é a que mais preserva a área de busca original entre as quatro configurações utilizadas. Essa configuração registrou uma variação de BD-rate de 0,11% em média para o conjunto de vídeos HD 1080p. Para os vídeos UHD 2160p o resultados de BD-rate foi de -0,04% em média, i.e. houve uma melhoria na eficiência de codificação nessa configuração para os vídeos UHD 2160p. Na média geral para os 24 vídeos a configuração *48-7bpp* apresentou uma variação de BD-rate de apenas 0,03%.

Esses pequenos ganhos em eficiência de codificação observados na Tabela 7.4, ocorrem com mais frequência nos vídeos UHD 2160p. Isso ocorre porque durante o processo de codificação, cada CTU faz otimizações locais sem considerar o impacto nas próximas CTUs. Além disso, como visto na seção anterior, sobre os resultados de redução no consumo de energia, os ganhos em energia dos vídeos UHD 2160p eram menores que nos vídeos HD 1080p. Isso ocorreu devido ao maior esforço computacional exercido pelo TZS para encontrar blocos candidatos com menor distorção, uma vez que o algoritmo trabalha com decisões sub-ótimas. Essas novas buscas realizadas pelo TZS levaram a blocos com menor distorção do que aqueles encontrados na configuração *baseline*. Contudo, se percebe que esses casos são a exceção, a regra geral é quanto mais restritiva for a configuração utilizada maior será a variação em BD-rate, e isso vale tanto para os vídeos HD 1080p quanto para os vídeos UHD 2160p.

Nesse sentido, a configuração *32-7bpp* apresentou uma variação de BD-rate de 0,24% para os vídeos HD 1080p e de 0,15% para vídeos UHD 2160p, atingindo uma variação média para os 24 vídeos de 0,2%. Ambos resultados são maiores que aqueles gerados pela configuração *48-7bpp*. A configuração *16-7bpp* apresentou resultados de BD-rate um pouco maiores que os encontrados nas configurações anteriores, com uma variação em BD-rate de 0,57% para os vídeos HD 1080p e 0,65% para os vídeos UHD 2160p, atingindo uma variação média de 0,6% para os 24 vídeos.

A maior variação em BD-rate é encontrada na configuração *16-4bpp*, sendo essa a configuração mais restrita, esse resultado era esperado. Nessa configuração a variação em BD-rate para os vídeos HD 1080p foi de 2,56%, um valor 4,5× maior quando comparado a configuração *16-7bpp*. Para os vídeos UHD 2160p a variação de BD-rate na configuração *16-4bpp* foi de 4,1%, i.e. uma taxa 6,3× maior que a configuração anterior. Destaca-se nessa configuração a alta variação atingida por alguns vídeos, tais como *FoodMarket* e *Tango*. Para o vídeo *FoodMarket* a variação em BD-rate foi de 9,9% e de 9,5% para o vídeo *Tango*. Entretanto, essas duas seqüências de vídeos apresentaram uma

Figura 7.2: Comparativo entre os três principais resultados para as quatro configurações.



Fonte: Próprio autor.

das maiores taxa de redução em consumo de energia, atingido uma redução acima de 97% nessa configuração. Na média geral a configuração *16-4bpp* apresentou uma variação em BD-rate de 3,3% para os 24 vídeos.

Nesse sentido, a Figura 7.2 apresenta uma relação entre os resultados de redução de largura de banda de memória externa (*BW*), redução do consumo de energia (*Energia*) e eficiência de codificação (*BD-rate*). Para esses gráficos foram utilizados os resultados médios para as 24 sequências de vídeo para os 4 QPs. Para os resultados de BD-rate apresentados na figura foram utilizados os resultados de complemento para a eficiência de codificação. Esse complemento é obtido pela subtração:  $100\% - \text{BD-rate}$ . Desta forma, os três eixos dos gráficos da Figura 7.2 estão no mesmo sentido, onde quanto maior a área em azul melhores são os resultados obtidos.

Desta forma, nota-se na Figura 7.2 que as configurações mais restritas apresentam resultados mais eficientes, uma vez que, mesmo com a maior perda em eficiência de codificação, essas configurações introduzem altos ganhos nos outros dois eixos, redução do consumo de energia e redução da largura de banda de memória externa, aumentando sua eficiência. Comparando as configurações *48-7bpp* e *16-4bpp*, nota-se que os resultados de eficiência de codificação dessas configurações apresentam a maior diferença, 3,3%, sendo que a configuração *48-7bpp* introduz uma perda de eficiência de codificação de

apenas 0,03%, enquanto que a configuração *16-4bpp* essa perda atinge 3,33%. Entretanto, nos resultados de redução do consumo de energia a configuração *48-7bpp* atinge 88% de redução e na configuração *16-4bpp* essa taxa atinge 95,7%. Essa diferença é ainda maior nos resultados de redução de largura de banda de memória externa, onde a configuração *48-7bpp* atinge uma redução de 73,6%, e na configuração *16-4bpp* essa redução atinge 89,5%.

Com base nos resultados apresentados e discutidos, percebe-se que as estratégias apresentadas nessa tese geram resultados positivos para o codificador de vídeo, uma vez que os resultados das quatro configurações avaliadas pelo MERSy atingem altas taxas de redução do consumo de energia, apresentando baixo impacto na eficiência de codificação. Três dessas configurações atingem taxas de eficiência de codificação inferiores ou igual a 0,6%, e na configuração *16-4bpp* essa taxa aumenta para 3,3%, que, como apresentado na Figura 7.2, é um valor baixo diante dos altos ganhos na redução do consumo de energia.

#### 7.4 Comparativo com trabalhos relacionados

Nesta seção, os resultados obtidos pelo MERSy, apresentados anteriormente, são comparados com os trabalhos relacionados mais relevantes encontrados na literatura. Esses trabalhos relacionados foram apresentados e discutidos no Capítulo 4.

A Tabela 7.5 apresenta o comparativo entre os resultados das quatro configurações avaliados no MERSy e 12 trabalhos relacionados. Os resultados apresentados para cada uma das configurações são referentes aos resultados médios para as 24 sequências de vídeo, incluindo os 4 QPs. Nessa tabela, além dos resultados de redução no consumo de energia (coluna *Energia*), dos resultados de perdas na eficiência de codificação (coluna *BD-rate*) e dos resultados de redução de largura de banda de memória (coluna *BW*), são também destacadas quais as técnicas utilizadas nesses trabalhos, sendo elas: reuso de dados (coluna *RD*), compressão de quadros de referência (coluna *CQR*), computação aproximada (coluna *CA*) e utilização de tecnologia NV-RAM (coluna *NVM*).

Conforme pode ser visto na Tabela 7.5, nenhum dos 12 trabalhos utilizados nos comparativos utiliza as quatro estratégias de redução em conjunto. Além disso, somente quatro desses trabalhos apresentam resultados para os três pontos avaliados. Desta forma, prejudicando uma comparação justa com esses trabalhos. Além disso, nem todos os trabalhos apresentam os ganhos referentes à uma configuração *baseline* como foi apresentado nessa tese. Entretanto, os resultados serão comparados com os dados que cada trabalho

Tabela 7.5: Comparativos com trabalhos relacionados

<b>Estratégias</b>	<b>RD</b>	<b>CQR</b>	<b>CA</b>	<b>NVM</b>	<b>Energia</b>	<b>BD-rate</b>	<b>BW</b>
<i>MERSy 48-7bpp</i>	X	X	X	X	88,0%	0,03%	73,6%
<i>MERSy 32-7bpp</i>	X	X	X	X	91,5%	0,20%	77,6%
<i>MERSy 16-7bpp</i>	X	X	X	X	94,0%	0,61%	81,5%
<i>MERSy 16-4bpp</i>	X	X	X	X	95,7%	3,33%	89,5%
Pakdaman et al. (2018)	-	-	X	-	NA	1,1%	78,0%
Singh e Ahamed (2018)	X	-	X	X	NA	1,8%	54,0%
Li et al. (2015)	-	-	X	-	NA	0,2%	91,0%
Xu et al. (2018)	X	-	-	-	68%	0,0%	67,8%
Martins et al. (2018)	X	-	-	-	79,3%	0,0%	94,9%
Fan, Shang e Zeng (2015)	X	X	X	-	NA	0,2%	74,5%
Lian et al. (2018)	X	X	X	-	54,3%	1,3%	72,9%
Willème et al. (2018)	X	X	X	-	NA	14,0%	83,3%
Lian et al. (2016)	-	X	-	-	56,8%	0,0%	68,5%
Li et al. (2017)	-	X	-	-	NA	0,0%	55,2%
Lee et al. (2020)	-	X	-	-	NA	0,0%	55,4%
Penny et al. (2019)	X	-	-	X	57,6%	0,0%	NA

Fonte: Próprio autor.

apresenta.

Os trabalhos de Pakdaman et al. (2018) e Li et al. (2015) utilizam intervalo de busca dinâmico para reduzir acessos e consumo de energia na codificação do vídeo. Ambos apresentam altas taxas de redução de largura de banda de memória externa, em Pakdaman et al. (2018) essa taxa atinge 78% com uma perda de 1,1% em BD-rate, ambos resultados inferiores que aos da configuração *16-7bpp*, que atinge 81,5% com uma perda em BD-rate de 0,61%. O trabalho de Li et al. (2015) apresenta taxa de redução de largura de banda superior a todas as configurações testadas no MERSy, 91%, com uma perda em BD-rate de apenas 0,2%. Entretanto, esses resultados foram obtidos para o *Full Search*, o qual realiza uma quantidade muito alta de acessos à memória e possui um comportamento de acesso bastante linear, o que possibilita que até mesmo uma técnica simples gere resultados expressivos de redução de acessos à memória.

O trabalho de Singh e Ahamed (2018) emprega três estratégias, reuso de dados, computação aproximada através do *bit dropping* e subamostragem e memórias NV-RAM, para reduzir a largura de banda com a memória externa, atingindo uma taxa de redução de 54% com perda de 1,8%. O trabalho de Penny et al. (2019) também utiliza tecnologia NV-RAM aliada ao reuso de dados para reduzir o consumo de energia relacionado à memória. A abordagem proposta em Penny et al. (2019) atinge uma redução do consumo de energia de 57,6%, sem inserir perdas em eficiência de codificação. Nota-se nesses trabalhos que

mesmo utilizando várias estratégias a redução de largura de banda de memória e a redução no consumo de energia não são altos, sendo que qualquer configuração testada no MERSy atinge taxas melhores com perdas insignificantes em BD-rate, 0,03% no caso.

Os trabalhos de Xu et al. (2018) e Martins et al. (2018) utilizam apenas reuso de dados para reduzir o consumo de energia do codificador. Em Xu et al. (2018) a largura de banda de memória é reduzida em 67,8%, gerando uma economia de energia de 68%. Enquanto que em Martins et al. (2018) a redução em largura de banda de memória reportado pelos autores é de 94,9%, sendo essa a taxa mais alta encontrada no comparativo. Entretanto, devido a utilização de 8 memórias, as quais consomem mais energia que SPMs, a economia em energia foi de 79,3%, sendo esse superado pelas configurações do MERSy.

Os trabalhos de Lian et al. (2016), Li et al. (2017) e Lee et al. (2020) utilizam a compressão de quadros de referência sem perdas. Desta forma, essas soluções não alteram a eficiência de codificação. Por outro lado, minimiza a redução de largura de banda de memória e a redução no consumo de energia. Em Li et al. (2017) e Lee et al. (2020) apresentam uma redução de largura de banda de memória de 55%. Enquanto que em Lian et al. (2016) essa taxa chega a 68,5%, gerando uma economia de energia de 56,8%. Por serem soluções sem perdas em BD-rate, as taxas de reduções ficam limitadas, sendo inferiores as apresentadas pelas configurações do MERSy.

As estratégias apresentadas em Fan, Shang e Zeng (2015), Lian et al. (2018) e Willème et al. (2018) utilizam reuso de dados, compressão de quadros de referência e computação aproximada para reduzirem a largura de banda de memória e o consumo de energia do codificador. No trabalho de Fan, Shang e Zeng (2015) a redução em largura de banda foi de 74,5% com uma perda em BD-rate de 0,2%, mesma perda apresentada na configuração *32-7bpp*. Porém, nessa configuração o MERSy atinge uma redução em largura de banda de 77,6%, gerando um consumo de energia de 91,5%, o resultado de energia não é apresentado em Fan, Shang e Zeng (2015). Lian et al. (2018) apresenta uma redução de largura de banda de memória semelhante, 72,9%, com uma perda em BD-rate de 1,3%, gerando uma redução no consumo de energia de 54,3%. Nesse caso, qualquer uma das três configuração com *7bpp* do MERSy apresenta resultados superiores. O trabalho de Willème et al. (2018) apresenta alta taxa de redução de largura de banda, 83,3%. Entretanto, para obter essa redução insere muitas perdas em eficiência de codificação, 14%. Assim, a configuração *16-4bpp* apresenta resultados melhores tanto em BD-rate quanto em redução de largura de banda de memória.

Diante dos resultados apresentados e comparados aos trabalhos estado da arte,

percebe-se que as abordagens propostas e desenvolvidas nessa tese, impactam de forma positiva a área da codificação de vídeo, gerando soluções de alta eficiência energética com baixo impacto na eficiência de codificação, trazendo uma contribuição significativa para essa área de pesquisa.

## 8 CONCLUSÕES

O trabalho de pesquisa desenvolvido nessa tese resultou em várias contribuições para a redução do consumo de energia em codificadores de vídeo, principalmente o consumo de energia relacionado a hierarquia de memória. A grande quantidade de simulações e análises que foram realizadas levaram a descobertas importantes relacionadas: i) ao comportamento dos acessos realizados pelo TZS dentro de uma área de busca com conhecimento das regiões mais e menos acessadas; ii) a utilização de computação aproximada eficiente com baixo impacto na eficiência de codificação; iii) a utilização de tecnologia NV-RAM na hierarquia de memória dedicada ao codificador de vídeo; iv) a técnicas de redução de acessos à memória, gerando economia de energia no codificador; v) a organização da hierarquia de memória possibilitando acesso aleatório a blocos codificados, permitindo *power gating* em bancos de memória; e vi) a utilização eficiente de várias técnicas em conjunto, gerando um sistema capaz de atingir altas taxas de redução em consumo de energia com baixo impacto na eficiência de codificação. Nesse sentido, este capítulo conclui esta tese apresentando as principais contribuições e indicando possíveis trabalhos futuros nessa área de pesquisa.

### 8.1 Principais contribuições

No Capítulo 5 foram apresentadas análises sobre a distribuição dos acessos à área de busca realizadas pelo algoritmo de busca TZS. Estas análises possibilitaram um melhor entendimento sobre o comportamento dos acessos realizados pelo TZS dentro de uma área de busca e o conhecimento das regiões mais e menos acessadas. Estas análises e o domínio dessas informações foram de fundamental importância para o desenvolvimento dessa tese, uma vez que as técnicas de computação aproximada exploradas, utilizam essas informações para atingirem o máximo de eficiência, gerando alta redução no consumo de energia com o mínimo de perda na eficiência de codificação. Além da análise dos acessos à área de busca, também foi avaliado o impacto da computação aproximada na eficiência de codificação. Desta forma, três abordagens foram avaliadas: i) cortes na área de busca, ii) *bit dropping* iii) e a abordagem conjunta com cortes na área de busca e *bit dropping*.

Essas três técnicas permitiram a utilização de 26 configurações distintas envolvendo computação aproximada, sendo: 6 configurações para cortes na área de busca, 8 configurações de *bit dropping* e 12 configurações envolvendo as configurações que utili-

zam cortes na área de busca e *bit dropping*. Todas essas configurações foram simuladas para os 12 sequências de vídeos e cada vídeo para os 4 QPs, totalizando 1248 simulações com 100 quadros cada uma.

Para definir as melhores configurações a serem utilizadas pelo MERSy, foram avaliadas as 19 configurações mais promissoras entre as 3 técnicas exploradas. Para essas configurações foram gerados os *traces* de acessos às amostras para todos os vídeos e QPs. Para encontrar as melhores configurações com resultados de BD-rate e redução de acessos às amostras da área de busca, foi utilizada uma frente de Pareto. Desta forma, sendo possível chegar ao conjunto composto por 4 configurações: *48-7bpp*, *32-7bpp*, *16-7bpp* e *16-4bpp*.

O Capítulo 6 apresentou o sistema desenvolvido na tese para avaliar essas quatro configurações. Esse sistema foi intitulado de MERSy – *Memory Energy Reduction System*, o qual possibilita a utilização conjunta de várias técnicas para atingir alta redução de consumo de energia em codificadores de vídeo. Para o desenvolvimento do MERSy foram exploradas, desenvolvidas e avaliadas técnicas de reuso de dados com memórias *scratchpad*, utilizando memórias de diferentes tamanhos e tecnologias (voláteis e não voláteis), compressão de quadros de referência e computação aproximada através de *bit dropping*, *power gating* e redução da área de busca. Além disso, esse capítulo apresentou os algoritmos, arquiteturas e os resultados de síntese lógica do compressor de quadros de referência utilizado no MERSy, o DDRFC-BD. Ainda, foi apresentada a organização das memórias internas e externas, as quais permitem a utilização de *power gating* e acesso aleatório aos blocos codificados. Por último nesse capítulo, foram avaliadas hierarquias de memórias para codificadores de vídeos compostas por tecnologias NV-RAMs.

Os resultados envolvendo as quatro configurações avaliadas pelo MERSy foram apresentados no Capítulo 7. Nesse capítulo foram apresentados e discutidos os principais resultados da tese, demonstrando através do MERSy que a utilização em conjunto das várias técnicas exploradas nesse trabalho, possibilitam alta redução de consumo de energia com baixo impacto na eficiência de codificação. Para isso, as configurações do MERSy utilizaram reuso de dados Level C, composto por memórias internas NV-RAM de tecnologia STT-MRAM e memória externa também NV-RAM de tecnologia ReRAM, sendo que foi utilizado *power gating* nos bancos das memórias internas. Além disso, os quadros de referência foram comprimidos de maneira eficiente através do DDRFC-BD.

As quatro configurações avaliadas apresentaram altas taxas de redução no consumo de energia com baixo impacto na eficiência de codificação. A configuração *48-7bpp*

apresentou taxas de redução no consumo de energia de 88% com uma variação em BD-rate de 0,03%. A configuração *32-7bpp* atingiu uma redução no consumo de energia de 91,5% com 0,2% de variação em BD-rate. Já a configuração *16-7bpp* apresentou uma redução no consumo de energia de 94% e variação em BD-rate de 0,61%. A maior taxa de redução no consumo de energia foi obtida pela configuração *16-4bpp*, 95,7%. Entretanto, essa configuração é a que apresentou a maior perda em eficiência de codificação, 3,3%, mesmo assim uma taxa aceitável diante da alta redução possibilitada no consumo de energia. Os resultados dessas configurações foram comparados com resultados de 12 trabalhos estado da arte. Nessa comparação, se pode perceber que os resultados apresentados por essas configurações superam os resultados dos trabalhos estado da arte, confirmando a hipótese de tese defendida nesse trabalho, de que o consumo de energia relacionado à hierarquia de memória de codificadores de vídeo pode ser drasticamente reduzido com a utilização de computação aproximada e tecnologias de memórias NV-RAMs, gerando baixo impacto na eficiência de codificação.

## 8.2 Oportunidades de trabalhos futuros

Como pode ser visto ao longo dessa tese, a utilização conjunta de várias estratégias gerou reduções importantes no consumo de energia do codificador de vídeo, e mesmo que nesse trabalho várias dessas técnicas tenham sido exploradas, ainda há uma clara oportunidade de pesquisa no tema de redução de consumo de energia e uso eficiente do sistema de memória em codificadores de vídeo.

O fato de que novos codificadores de vídeo como H.266/VVC por exemplo, exigem ainda mais largura de banda com a memória externa quando comparado ao HEVC, faz com essas estratégias apresentadas nesse texto sejam imprescindíveis para uma eficiente utilização da codificação de vídeo. Desta forma, essa área pode ser melhorada ainda mais com desenvolvimento de técnicas de reuso de dados ainda mais eficientes; melhor uso da não-volatilidade das NV-RAMs; maior utilização da computação aproximada no processo de codificação e maior utilização da técnica de processamento em memória NV-RAM, para reduzir ainda mais o tráfego de dados entre codificador e memória externa.

## REFERÊNCIAS

AFONSO, V. et al. Energy-aware motion and disparity estimation system for 3D-HEVC with run-time adaptive memory hierarchy. **IEEE Transactions on Circuits and Systems for Video Technology**, v. 29, n. 6, p. 1878–1892, Junho 2019. ISSN 1558-2205.

AFONSO, V. et al. Hardware implementation for the HEVC fractional motion estimation targeting real-time and low-energy. **Journal of Integrated Circuits and Systems**, v. 11, n. 2, p. 106–120, 2016.

ALI, T. et al. High endurance ferroelectric hafnium oxide-based FeFET memory without retention penalty. **IEEE Transactions on Electron Devices**, v. 65, n. 9, p. 3769–3774, Setembro 2018. ISSN 0018-9383.

AOM. **AV1 Bitstream and Decoding Process Specification**. [S.l.], 2019. Disponível em: <<https://aomedia.org/av1-bitstream-and-decoding-process-specification/>>.

AV1. **AV1 reference software**. [S.l.], 2019. Disponível em: <<https://aomedia.googleusercontent.com/aom/>>.

BHATI, I. et al. Flexible auto-refresh: enabling scalable and energy-efficient DRAM refresh reductions. **International Symposium on Computer Architecture (ISCA)**, p. 235–246, Outubro 2015.

BI, X. et al. Cross-layer optimization for multilevel cell STT-RAM caches. **IEEE Transactions on Very Large Scale Integration (VLSI) Systems**, v. 25, n. 6, p. 1807–1820, Junho 2017.

BJONTEGAARD, G. **Calculation of average PSNR differences between RD curves**. [S.l.], 2001. 1-5 p.

BOSSEN, F. **Common Test Conditions and Software Reference Configurations**. [S.l.], 2012. 1-3 p.

BOSSEN, F. et al. HEVC complexity and implementation analysis. **IEEE Transactions on Circuits and Systems for Video Technology**, v. 22, n. 12, p. 1685–1696, 2012.

CACTI, H. **CACTI 6.5**. 2018. Disponível em: <<http://www.hpl.hp.com/research/cacti>>.

CADENCE. **Cadence EDA tools**. 2020. Disponível em: <<http://www.cadence.com>>.

CEBRIÁN-MÁRQUEZ, G.; MARTÍNEZ, J. L.; CUENCA, P. A motion-based partitioning algorithm for HEVC using a pre-analysis stage. **IEEE Transactions on Circuits and Systems for Video Technology**, v. 29, n. 5, p. 1448–1461, 2019.

CERVEIRA, A. et al. Memory assessment of versatile video coding. In: **2020 IEEE International Conference on Image Processing (ICIP)**. [S.l.: s.n.], 2020. p. 1–5.

CHANG, M. et al. A high-speed 7.2-ns read-write random access 4-Mb embedded resistive RAM (ReRAM) macro using process-variation-tolerant current-mode read schemes. **IEEE Journal of Solid-State Circuits**, v. 48, n. 3, p. 878–891, Março 2013. ISSN 0018-9200.

CHEN, C.-Y. et al. Level C+ data reuse scheme for motion estimation with corresponding coding orders. **IEEE Transactions on Circuits and Systems for Video Technology**, v. 16, n. 4, p. 553–558, 2006.

CHEN, Y.-G.; LEE, Y.-H.; CHEN, C.-C. An efficient multi-directional lossless recompression for video coding systems. In: **IEEE International Symposium on Bioelectronics and Bioinformatics**. [S.l.: s.n.], 2014. p. 1–4.

CISCO. **Cisco Visual Networking Index: Forecast and Trends, 2017-2022**. [S.l.], 2019. Disponível em: <[www.cisco.com](http://www.cisco.com)>.

CISCO. **Cisco Annual Internet Report - 2018–2023**. [S.l.], 2020. Disponível em: <[www.cisco.com](http://www.cisco.com)>.

CORREA, G. et al. Performance and computational complexity assessment of high-efficiency video encoders. **IEEE Transactions on Circuits and Systems for Video Technology**, v. 22, n. 12, p. 1899–1909, 2012.

CORREA, G. et al. Fast HEVC encoding decisions using data mining. **IEEE Transactions on Circuits and Systems for Video Technology**, v. 25, n. 4, p. 660–673, Abril 2015. ISSN 1051-8215.

DONG, X. et al. NVSim: A circuit-level performance, energy, and area model for emerging nonvolatile memory. **IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems**, v. 31, n. 7, p. 994–1007, Julho 2012. ISSN 0278-0070.

DÜNKEL, S. et al. A FeFET based super-low-power ultra-fast embedded nvm technology for 22nm FDSOI and beyond. In: **2017 IEEE International Electron Devices Meeting (IEDM)**. [S.l.: s.n.], 2017. p. 1971–1974. ISSN 2156-017X.

EISENMAN, A. et al. Reducing DRAM footprint with NVM in facebook. In: **Proceedings of the Thirteenth EuroSys Conference**. New York, NY, USA: Association for Computing Machinery, 2018. (EuroSys '18). ISBN 9781450355841. Disponível em: <<https://doi.org/10.1145/3190508.3190524>>.

FAN, Y.; SHANG, Q.; ZENG, X. In-block prediction-based mixed lossy and lossless reference frame recompression for next-generation video encoding. **IEEE Transactions on Circuits and Systems for Video Technology**, v. 25, n. 1, p. 112–124, Janeiro 2015.

FRUSTACI, F. et al. Approximate SRAMs with dynamic energy-quality management. **IEEE Transactions on Very Large Scale Integration (VLSI) Systems**, v. 24, n. 6, p. 2128–2141, 2016.

GHOSH, A.; DEHURI, S. Evolutionary algorithms for multi-criterion optimization: A survey. **International Journal of Computing and Information Sciences**, v. 2, n. 1, p. 38–57, 2004.

GOEBEL, J. et al. An hevc multi-size DCT hardware with constant throughput and supporting heterogeneous CUs. In: **IEEE International Symposium on Circuits and Systems (ISCAS)**. [S.l.: s.n.], 2016. p. 2202–2205.

GONG, N. et al. SPIDER: Sizing-priority-based application-driven memory for mobile video applications. **IEEE Transactions on Very Large Scale Integration (VLSI) Systems**, v. 25, n. 9, p. 2625–2634, Setembro 2017.

GRELLERT, M. et al. A multilevel data reuse scheme for motion estimation and its VLSI design. In: **2011 IEEE International Symposium of Circuits and Systems (ISCAS)**. [S.l.: s.n.], 2011. p. 583–586. ISSN 2158-1525.

GU, C.; ZENG, X.; FAN, Y. A 5.3 Gpixels/s frame memory recompression method for QHD video coding. In: **2018 14th IEEE International Conference on Solid-State and Integrated Circuit Technology (ICSICT)**. [S.l.: s.n.], 2018. p. 1–3.

GUO, L.; ZHOU, D.; GOTO, S. Lossless embedded compression using multi-mode DPCM amp; averaging prediction for HEVC-like video codec. In: **European Signal Processing Conference (EUSIPCO 2013)**. [S.l.: s.n.], 2013. p. 1–5. ISSN 2076-1465.

GUO, L.; ZHOU, D.; GOTO, S. A new reference frame recompression algorithm and its VLSI architecture for UHDTV video codec. **IEEE Transactions on Multimedia**, v. 16, n. 8, p. 2323–2332, Dezembro 2014. ISSN 1520-9210.

GUPTE, A. et al. Memory bandwidth and power reduction using lossy reference frame compression in video encoding. **IEEE Transactions on Circuits and Systems for Video Technology**, v. 21, n. 2, p. 225–230, 2011.

HONG, S.; AUCIELLO, O.; WOUTERS, D. **Emerging Non-Volatile Memories**. 1. ed. [S.l.]: Springer New York, 2014. ISBN 978-1-4899-7537-9.

HOSOMI, M. et al. A novel nonvolatile memory with spin torque transfer magnetization switching: spin-ram. In: **IEEE International Electron Devices Meeting, 2005. IEDM Technical Digest**. [S.l.: s.n.], 2005. p. 459–462. ISSN 0163-1918.

HUANG, Y.-W. et al. Survey on block matching motion estimation algorithms and architectures with new results. **Journal of VLSI Signal Processing Systems for Signal Image and Video Technology**, v. 42, p. 297–320, February 2006.

IMEC. **Next-gen memory ramping up**. [S.l.], 2018.

INTEL. **Intel Product Specifications**. [S.l.], 2021. Disponível em: <<https://ark.intel.com/>>.

IPSL. **Complexity and Performance Analysis of HEVC encoder**. [S.l.], 2012.

IRDS, I. **International Roadmap for Devices and Systems: Emerging Research Materials**. [S.l.], 2017.

ISO/IEC. **ISO/IEC 11172-2 MPEG-1 (1993): Information technology - Coding of moving pictures and associated audio for digital storage media at up to about 1,5 Mbit/s – Part 2: Video**. [S.l.], 1993. Disponível em: <<https://www.iso.org/standard/22411.html>>.

ISO/IEC. **ISO/IEC 14496-2 MPEG-4 Part 2 (2004): Information technology - Coding of audio-visual objects - Part 2: Visual**. [S.l.], 2004. Disponível em: <<https://www.iso.org/standard/39259.html>>.

ISO/IEC. **ISO/IEC 21122-1: Information technology - JPEG XS low-latency lightweight image coding system - Part 1: Core coding system.** [S.l.], 2019. Disponível em: <<https://www.iso.org/standard/74535.html>>.

ITU-R. **BT.2020: Parameter values for ultra-high definition television systems for production and international programme exchange.** [S.l.], 2020. Disponível em: <<https://www.itu.int/rec/R-REC-BT.2020>>.

ITU-T. **Recommendation H.262 (02/00): Information technology - Generic coding of moving pictures and associated audio information: Video.** [S.l.], 2000. Disponível em: <<https://www.itu.int/rec/T-REC-H.262/>>.

ITU-T. **Recommendation H.263 (01/05): Video coding for low bit rate communication.** [S.l.], 2005. Disponível em: <<https://www.itu.int/rec/T-REC-H.263/>>.

ITU-T. **Recommendation H.264 (01/12): Advanced video coding for generic audiovisual services.** [S.l.], 2012. Disponível em: <<http://www.itu.int/rec/T-REC-H.264-200305-S>>.

ITU-T. **Recommendation H.265: High Efficiency Video Coding and Audiovisual and Multimedia Systems.** [S.l.], 2013. Disponível em: <<http://www.itu.int/rec/T-REC-H.265-201304-I>>.

JACOB, B.; NG, S.; WANG, D. **Memory Systems: Cache, DRAM, Disk.** 1. ed. [S.l.]: Morgan Kaufmann, 2007. ISBN 978-0-12-379751-3.

JCT-VC. **Vision and Applications and Requirements for High Efficiency Video Coding (HEVC).** [S.l.], 2011. Disponível em: <<http://www.itu.int/rec/T-REC-H.264-200305-S>>.

JCT-VC. **High Efficiency Video Coding Software Repository.** [S.l.], 2014. Disponível em: <[https://hevc.hhi.fraunhofer.de/svn/svn\\_HEVCSoftware/](https://hevc.hhi.fraunhofer.de/svn/svn_HEVCSoftware/)>.

JEONG, K. et al. MAPG: Memory access power gating. In: **Design, Automation Test in Europe Conference (DATE).** [S.l.: s.n.], 2012. p. 1054–1059.

JOKAR, M. R.; ARJOMAND, M.; SARBAZI-AZAD, H. Sequoia: A high-endurance NVM-based cache architecture. **IEEE Transactions on Very Large Scale Integration Systems**, v. 24, n. 3, p. 954–967, 2016. ISSN 1063-8210.

JUNG, M. et al. Invited: Approximate computing with partially unreliable dynamic random access memory - approximate DRAM. **Design Automation Conference (DAC)**, p. 1–4, Outubro 2016.

JVET. **Developing a video compression algorithm with capabilities beyond HEVC.** [S.l.], 2019. Disponível em: <<https://www.itu.int/en/ITU-T/studygroups/2017-2020/16/Pages/video/jvet.aspx>>.

JVET. **JVET-G1001: Algorithm Description of Joint Exploration Test Model (JEM).** [S.l.], 2019. Disponível em: <<https://jvet.hhi.fraunhofer.de/>>.

JVET. **JVET-J1002: Algorithm Description of Versatile Test Model (VTM).** [S.l.], 2019. Disponível em: <<https://jvet.hhi.fraunhofer.de/>>.

KANG, U. et al. Co-architecting controllers and dram to enhance dram process scaling. In: **The Memory Forum**. [S.l.: s.n.], 2014. p. 1–4.

KECKLER, S. W. et al. Gpus and the future of parallel computing. **IEEE Micro**, v. 31, n. 5, p. 7–17, 2011.

KHAN, M. U. K.; SHAFIQUE, M.; HENKEL, J. AMBER: Adaptive energy management for on-chip hybrid video memories. **IEEE International Conference on Computer-Aided Design (ICCAD)**, p. 405–412, Novembro 2013.

KUHN, P. **Algorithms, Complexity Analysis and VLSI Architectures for Mpeg-4 Motion Estimation**. [S.l.]: Kluwer Academic Publisher, 1999. ISBN 978-1-4419-50888.

KULTURSAY, E. et al. Evaluating STT-RAM as an energy-efficient main memory alternative. **International Symposium on Performance Analysis of Systems and Software (ISPASS)**, p. 256–267, Julho 2013.

LAINEMA, J. et al. Intra coding of the HEVC standard. **IEEE Transactions on Circuits and Systems for Video Technology**, v. 22, n. 12, p. 1792–1801, Dezembro 2012. ISSN 1051-8215.

LEE, Y.-H.; CHEN, C.-C.; YOU, Y.-L. Design of VLSI architecture of autocorrelation-based lossless recompression engine for memory-efficient video coding systems. **Springer Circuits, Systems, and Signal Processing**, v. 3, n. 2, p. 459–482, 2014.

LEE, Y.-H. et al. Algorithm and VLSI architecture designs of a lossless embedded compression encoder for HD video coding systems. **Journal of Circuits, Systems and Computers**, v. 30, n. 4, p. 1–23, 2020. Disponível em: <<https://doi.org/10.1142/S021812662130004X>>.

LI, D.; ZHENG, W.; ZHANG, M. Architecture design for H.264/AVC integer motion estimation with minimum memory bandwidth. **IEEE Transactions on Consumer Electronics**, v. 53, n. 3, p. 1053–1060, Agosto 2007. ISSN 0098-3063.

LI, S. et al. Lossless image compression algorithm and hardware architecture for bandwidth reduction of external memory. **IET Image Processing**, v. 11, n. 6, p. 379–388, 2017. ISSN 1751-9659.

LI, Y. et al. An adaptive search range method for HEVC with the k-nearest neighbor algorithm. In: **Visual Communications and Image Processing (VCIP)**. [S.l.: s.n.], 2015. p. 1–4.

LIAN, X. et al. Lossless frame memory compression using pixel-grain prediction and dynamic order entropy coding. **IEEE Transactions on Circuits and Systems for Video Technology**, v. 26, n. 1, p. 223–235, Jan 2016. ISSN 1051-8215.

LIAN, X. et al. Parallel content-aware adaptive quantization-oriented lossy frame memory recompression for HEVC. **IEEE Transactions on Circuits and Systems for Video Technology**, v. 28, n. 4, p. 958–971, Abril 2018. ISSN 1051-8215.

LUO, F. et al. Gpu-based hierarchical motion estimation for high efficiency video coding. **IEEE Transactions on Multimedia**, v. 21, n. 4, p. 851–862, 2019.

MA, Z.; SEGALL, A. Frame buffer compression for low-power video coding. In: **18th IEEE International Conference on Image Processing**. [S.l.: s.n.], 2011. p. 757–760.

MARTINS, A. et al. Configurable cache memory architecture for low-energy motion estimation. In: **2018 IEEE International Symposium on Circuits and Systems (ISCAS)**. [S.l.: s.n.], 2018. p. 1–5. ISSN 2379-447X.

MARWEDEL, P. **Embedded Systems Foundations of Cyber-Physical Systems**. 2. ed. [S.l.]: Springer Dordrecht, 2011. ISBN 978-9-4007-0257-8.

MATIVI, A.; MONTEIRO, E.; BAMPI, S. Memory access profiling for HEVC encoders. In: **2016 IEEE 7th Latin American Symposium on Circuits Systems (LASCAS)**. [S.l.: s.n.], 2016. p. 243–246.

MICRON. **Micron MT42L32M32D1HE-18: 1Gb LPDDR2 SDRAM**. [S.l.], 2019. Disponível em: <<http://www.micron.com>>.

MICRON. **Micron DRAM**. [S.l.], 2021. Disponível em: <<https://www.micron.com/products/dram/>>.

MIKOLAJICK et al. Ferroelectric hafnium oxide for ferroelectric random-access memories and ferroelectric field-effect transistors. **MRS Bulletin**, Cambridge University Press, v. 43, n. 5, p. 340–346, 2018.

MONTEIRO, E. R. **Caracterização Energética da Codificação de Vídeo de Alta Eficiência (HEVC) em Processador de Propósito Geral**. 144 p. Tese (Doutorado) — Universidade Federal do Rio Grande do Sul, Instituto de Informática, 2017.

MRAM-INFO. **STT-MRAM: Introduction and market status**. [S.l.], 2018.

MUTLU, O.; SUBRAMANIAN, L. Research problems and opportunities in memory systems. **Supercomputing Frontiers and Innovations**, v. 1, n. 3, p. 19–55, 2015.

MÜLLER, J. et al. Ferroelectric hafnium oxide: A CMOS-compatible and highly scalable approach to future ferroelectric memories. In: **2013 IEEE International Electron Devices Meeting**. [S.l.: s.n.], 2013. p. 1081–1084. ISSN 2156-017X.

OHM, J.-R. et al. Comparison of the coding efficiency of video coding standards—including high efficiency video coding (HEVC). **IEEE Transactions on Circuits and Systems for Video Technology**, v. 22, n. 12, p. 1669–1684, Dezembro 2012.

PAIM, G. et al. Power-, area-, and compression-efficient eight-point approximate 2-d discrete tchebichef transform hardware design combining truncation pruning and efficient transposition buffers. **IEEE Transactions on Circuits and Systems I: Regular Papers**, v. 66, n. 2, p. 680–693, 2019.

PAKDAMAN, F. et al. Fast motion estimation algorithm with efficient memory access for HEVC hardware encoders. In: **European Workshop on Visual Information Processing (EUVIP)**. [S.l.: s.n.], 2018. p. 1–5. ISSN 2164-974X.

PAKDAMAN, F.; HASHEMI, M. R.; GHANBARI, M. A low complexity and computationally scalable fast motion estimation algorithm for HEVC. **Springer Multimedia Tools and Applications**, v. 79, n. 17, p. 11639–11669, 2020.

PENNY, W. et al. Energy-efficiency exploration of memory hierarchy using NVMs for HEVC motion estimation. In: **IEEE International Conference on Electronics, Circuits and Systems (ICECS)**. [S.l.: s.n.], 2019. p. 162–165.

PORTO, R. et al. Fast and energy-efficient approximate motion estimation architecture for real-time 4K UHD processing. **Journal of Real-Time Image Processing**, v. 18, n. 2, p. 723–737, Agosto 2020. ISSN 1861-8219. Disponível em: <<https://doi.org/10.1007/s11554-020-01014-6>>.

RICHARDSON, I. E. G. **H.264 and MPEG-4 Video Compression: Video Coding for Next generation Multimedia**. [S.l.]: Wiley, 2004. ISBN 0-470-84837-5.

RICHARDSON, I. E. G. **The H.264 Advanced Video Compression Standard**. 2. ed. [S.l.]: Wiley, 2011. ISBN 978-0-470-51692-8.

ROCHA, L. G. et al. Binary CorNET: Accelerator for HR estimation from wrist-ppg. **IEEE Transactions on Biomedical Circuits and Systems**, v. 14, n. 4, p. 715–726, 2020.

SAMPAIO, F. et al. Approximation-aware multi-level cells STT-RAM cache architecture. **International Conference on Compilers, Architecture, and Synthesis for Embedded Systems (CASES)**, p. 79–88, Outubro 2015.

SAMPAIO, F. M. et al. Hybrid scratchpad video memory architecture for energy-efficient parallel HEVC. **IEEE Transactions on Circuits and Systems for Video Technology**, v. 29, n. 10, p. 3046–3060, 2019.

SAYOOD, K. **Introduction to Data Compression**. 3. ed. [S.l.]: Morgan Kaufmann, 2005.

SHAFIQUE, M. et al. The eda challenges in the dark silicon era: Temperature, reliability, and variability perspectives. In: **Proceedings of the 51st Annual Design Automation Conference**. New York, NY, USA: ACM, 2014. (DAC '14), p. 185:1–185:6. ISBN 978-1-4503-2730-5. Disponível em: <<http://doi.acm.org/10.1145/2593069.2593229>>.

SHAFIQUE, M.; HENKEL, J. Low power design of the next-generation high efficiency video coding. **Asia and South Pacific Design Automation Conference (ASP-DAC)**, p. 274–281, Janeiro 2014.

SHAFIQUE, M. et al. Adaptive power management of on-chip video memory for multiview video coding. **Design Automation Conference (DAC)**, p. 866–875, 2012.

SILVEIRA, D.; PORTO, M.; BAMPI, S. Performance and energy consumption analysis of the x265 video encoder. In: **European Signal Processing Conference (EUSIPCO)**. [S.l.: s.n.], 2017. p. 1519–1523. ISSN 2076-1465.

SILVEIRA, D. et al. Efficient reference frame compression scheme for video coding systems: algorithm and VLSI design. **Journal of Real-Time Image Processing**, v. 16, n. 2, p. 391–411, Abril 2019. ISSN 1861-8219. Disponível em: <<https://doi.org/10.1007/s11554-015-0551-1>>.

- SINANGIL, M. E. et al. Memory cost vs. coding efficiency trade-offs for HEVC motion estimation engine. In: **2012 19th IEEE International Conference on Image Processing**. [S.l.: s.n.], 2012. p. 1533–1536.
- SINGH, K.; AHAMED, S. R. Low power motion estimation algorithm and architecture of HEVC/H.265 for consumer applications. **IEEE Transactions on Consumer Electronics**, v. 64, n. 3, p. 267–275, 2018. ISSN 0098-3063.
- SONG, C.; JU, L.; JIA, Z. Hybrid scratchpad and cache memory management for energy-efficient parallel hevc encoding. **International Conference on Computer Design (ICCD)**, p. 712–719, Outubro 2015.
- SULLIVAN, G. J. 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, Dezembro 2012.
- SULLIVAN, G. J.; TOPIWALA, P. N.; LUTHRA, A. The H.264/AVC advanced video coding standard: overview and introduction to the fidelity range extensions. In: . [s.n.], 2004. v. 5558, p. 1 – 21. Disponível em: <<https://doi.org/10.1117/12.564457>>.
- SZE, V.; BUDAGAVI, M.; SULLIVAN, G. J. **High Efficiency Video Coding (HEVC): Algorithms and Architectures**. Cham: Springer International Publishing, 2014. ISBN 978-3-319-06894-7.
- TANG, X.; DAI, S.; CAI, C. An analysis of tzsearch algorithm in jmvc. In: **The 2010 International Conference on Green Circuits and Systems**. [S.l.: s.n.], 2010. p. 516–520. ISSN null.
- TIKEKAR, M. et al. A 249-Mpixel/s HEVC video-decoder chip for 4K Ultra-HD applications. **IEEE Journal of Solid-State Circuits**, v. 49, n. 1, p. 61–72, Jan 2014. ISSN 0018-9200.
- TUAN, J.-C.; CHANG, T.-S.; JEN, C.-W. On the data reuse and memory bandwidth analysis for full-search block-matching vlsi architecture. **IEEE Transactions on Circuits and Systems for Video Technology**, p. 61–72, 2002.
- VANNE, J. et al. Comparative rate-distortion-complexity analysis of HEVC and AVC video codecs. **IEEE Transactions on Circuits and Systems for Video Technology**, v. 22, n. 12, p. 1885–1898, Dezembro 2012. ISSN 1051-8215.
- VERMA, M.; MARWEDEL, P. **Advanced Memory Optimization Techniques for Low-Power Embedded Processors**. 1. ed. [S.l.]: Springer Berlin, 2007. ISBN 13 978-1-4020-5897-4.
- WANG, Q. et al. A highly parallelized PIM-based accelerator for transaction-based blockchain in IoT environment. **IEEE Internet of Things Journal**, v. 7, n. 5, p. 4072–4083, 2020.
- WILLÈME, A. et al. Power-aware HEVC compression through asymmetric JPEG XS frame buffer compression. In: **IEEE International Conference on Image Processing (ICIP)**. [S.l.: s.n.], 2018. p. 3598–3602. ISSN 2381-8549.

WONG, H. S. P. et al. Metal-oxide RRAM. **Proceedings of the IEEE**, v. 100, n. 6, p. 1951–1970, Junho 2012. ISSN 0018-9219.

WONG, H. S. P. et al. Phase Change Memory. **Proceedings of the IEEE**, v. 98, n. 12, p. 2201–2227, Dezembro 2010. ISSN 0018-9219.

X265. **x265 Software**. 2019. Disponível em: <<http://x265.readthedocs.io/en/default/index.html>>.

XIE, M. et al. A novel STT-RAM-based hybrid cache for intermittently powered processors in IoT devices. **IEEE Micro**, v. 39, n. 1, p. 24–32, 2019.

XIE, Y. **Emerging Memory Technologies: Design, Architecture, and Applications**. 1. ed. [S.l.]: Springer New York, 2014. ISBN 978-1-4419-9551-3.

XU, C. et al. Overcoming the challenges of crossbar resistive memory architectures. In: **2015 IEEE 21st International Symposium on High Performance Computer Architecture (HPCA)**. [S.l.: s.n.], 2015. p. 476–488.

XU, Q.; MYTKOWICZ, T.; KIM, N. S. Approximate computing: A survey. **IEEE Design Test**, v. 33, n. 1, p. 8–22, Feb 2016. ISSN 2168-2356.

XU, W. et al. A novel data reuse method to reduce demand on memory bandwidth and power consumption for true motion estimation. **IEEE Access**, v. 6, p. 10151–10159, 2018. ISSN 2169-3536.

YU, H.; WANG, Y. **Design Exploration of Emerging Nano-scale Non-volatile Memory**. 1. ed. [S.l.]: Springer New York, 2014. ISBN 978-1-4939-0551-5.

ZATT, B. et al. **3D Video Coding for Embedded Devices: Energy Efficient Algorithms and Architectures**. 1. ed. [S.l.]: Springer-Verlag New York, 2013. ISBN 978-1-4614-6759-5.

ZATT, B. et al. Run-time adaptive energy-aware motion and disparity estimation in multiview video coding. **Design Automation Conference (DAC)**, p. 1026–1031, 2011.

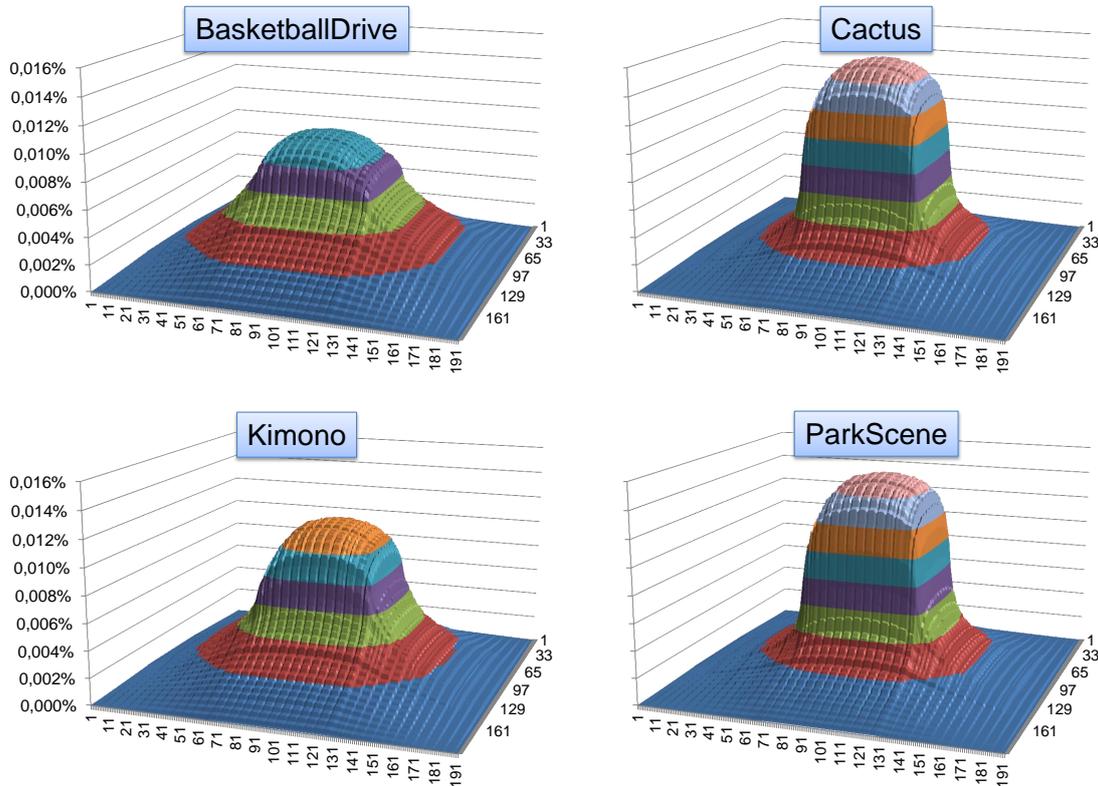
ZHENG, J. et al. A hardware-efficient block matching algorithm and its hardware design for variable block size motion estimation in ultra-high-definition video encoding. **ACM Trans. Des. Autom. Electron. Syst.**, ACM, New York, NY, USA, v. 24, n. 2, p. 15:1–15:21, jan. 2019. ISSN 1084-4309. Disponível em: <<http://doi.acm.org/10.1145/3290408>>.

ZHOU, D. et al. Reducing power consumption of HEVC codec with lossless reference frame recompression. In: **2014 IEEE International Conference on Image Processing (ICIP)**. [S.l.: s.n.], 2014. p. 2120–2124. ISSN 1522-4880.

## APÊNDICE A — HISTOGRAMAS DOS ACESSOS À ÁREA DE BUSCA

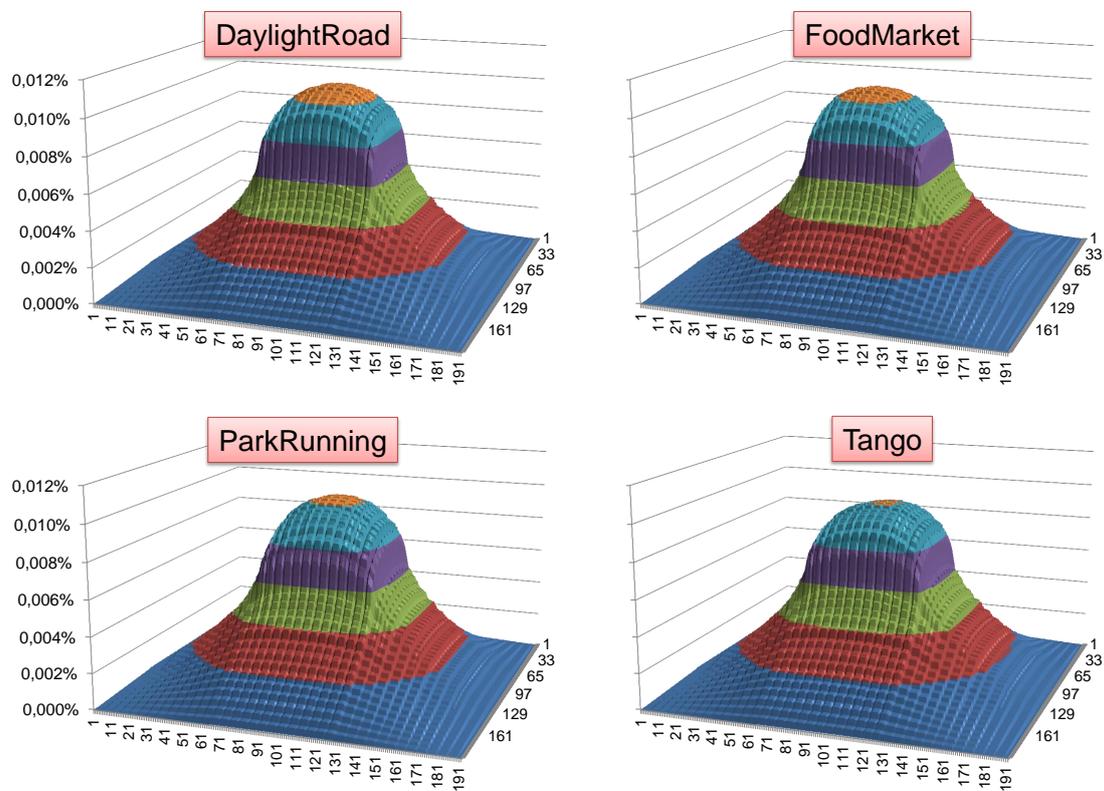
As Figuras A.1 e A.2 apresentam os histogramas para os vídeos do conjuntos de testes que apresentaram histogramas com resultados intermediários. A Figura A.1 apresenta os histogramas para os vídeos *BasketballDrive*, *Cactus*, *Kimono* e *ParkScene* de resolução HD 1080p. Na Figura A.2 são apresentados os histogramas para os vídeos *DaylightRoad*, *FoodMarket*, *ParkRunning* e *Tango* de resolução UHD 2160p.

Figura A.1: Histogramas com acessos à área de busca para quatro vídeos 1080p.



Fonte: Próprio autor.

Figura A.2: Histogramas com acessos à área de busca para quatro vídeos 2160p.



Fonte: Próprio autor.

## APÊNDICE B — RESULTADOS DE ACESSOS DAS TÉCNICAS DE COMPUTAÇÃO APROXIMADA

Este Capítulo de Apêndice apresenta os resultados de acessos para as três configurações de computação aproximada utilizadas nesta tese. As configurações apresentadas aqui são aquelas que tiveram melhores resultados e foram utilizadas nos comparativos de resultados na seção 5.4.4. Em todas as tabelas são apresentados resultados por QP e resolução, na última linha das tabelas aparecem os totais das médias para as resoluções e a média geral para todos os vídeos.

A Tabela B.1 apresenta os resultados de redução de acessos para os três tamanhos de área de busca avaliados, *48R*, *32R* e *16R*. Na Tabela B.2 são apresentados os resultados para as quatro configurações de *bit dropping* mais promissoras: *7bpp*, *6bpp*, *5bpp* e *4bpp*. As três últimas tabelas desse apêndice apresentam os resultados para as estratégias que empregam redução da área de busca e *bit dropping* em conjunto. Na Tabela B.3 são apresentados os resultados para o intervalo de busca  $[-48, +48]$  e os 4 níveis de *bit dropping*. Já na Tabela B.4 o intervalo utilizado foi de  $[-32, +32]$  com os 4 *bit dropping* e na Tabela B.5 o intervalo de busca foi de  $[-16, +16]$  juntamente com os 4 níveis de *bit dropping*.

Tabela B.1: Resultados de redução de acessos para as configurações com cortes na área de busca.

QPs	Vídeos	48R	32R	16R
22	1080p	16,55%	32,73%	53,48%
	2160p	18,23%	35,49%	56,94%
	Média	17,94%	35,01%	56,34%
27	1080p	16,68%	33,05%	53,91%
	2160p	19,01%	36,42%	57,78%
	Média	18,60%	35,82%	57,09%
32	1080p	17,96%	34,45%	55,10%
	2160p	23,67%	37,56%	58,70%
	Média	22,63%	36,99%	58,04%
37	1080p	19,13%	35,75%	56,18%
	2160p	21,01%	38,68%	59,50%
	Média	20,66%	38,13%	58,88%
Total	1080p	17,47%	33,87%	54,56%
	2160p	20,28%	36,87%	58,09%
	Média	19,78%	36,33%	57,46%

Fonte: Próprio autor.

Tabela B.2: Resultados de redução de acessos para as configurações com *bit dropping*.

QPs	Vídeos	7bpp	6bpp	5bpp	4bpp
22	1080p	11,87%	22,26%	32,07%	40,43%
	2160p	12,34%	18,78%	29,42%	44,04%
	Média	12,26%	19,39%	29,88%	43,41%
27	1080p	11,53%	21,83%	31,44%	39,56%
	2160p	10,79%	18,47%	28,11%	42,52%
	Média	10,92%	19,06%	28,70%	42,00%
32	1080p	12,35%	22,88%	31,84%	39,38%
	2160p	11,56%	19,48%	27,16%	40,29%
	Média	11,70%	20,10%	28,01%	40,12%
37	1080p	12,65%	23,73%	32,48%	38,97%
	2160p	12,01%	20,94%	27,24%	37,90%
	Média	12,13%	21,46%	28,22%	38,10%
Total	1080p	12,06%	22,61%	31,94%	39,64%
	2160p	11,68%	19,30%	28,10%	41,51%
	Média	11,74%	19,89%	28,79%	41,17%

Fonte: Próprio autor.

Tabela B.3: Resultados de redução de acessos para as configurações com *bit dropping* e cortes na área de busca de tamanho 48.

QPs	Vídeos	48-7bpp	48-6bpp	48-5bpp	48-4bpp
22	1080p	26,50%	35,19%	43,38%	50,54%
	2160p	26,84%	34,30%	44,59%	54,81%
	Média	26,78%	34,45%	44,38%	54,07%
27	1080p	26,80%	35,38%	43,28%	50,13%
	2160p	27,87%	34,22%	46,80%	53,89%
	Média	27,68%	34,42%	46,18%	53,23%
32	1080p	28,19%	36,90%	44,21%	50,40%
	2160p	29,29%	35,90%	42,01%	52,47%
	Média	29,09%	36,09%	42,41%	52,09%
37	1080p	29,26%	38,40%	45,42%	50,73%
	2160p	30,51%	37,87%	42,84%	51,15%
	Média	30,28%	37,97%	43,32%	51,07%
Total	1080p	27,57%	36,34%	43,99%	50,44%
	2160p	28,43%	35,38%	44,22%	53,27%
	Média	28,28%	35,55%	44,18%	52,77%

Fonte: Próprio autor.

Tabela B.4: Resultados de redução de acessos para as configurações com *bit dropping* e cortes na área de busca de tamanho 32.

<b>QPs</b>	<b>Vídeos</b>	<b>32-7bpp</b>	<b>32-6bpp</b>	<b>32-5bpp</b>	<b>32-4bpp</b>
22	1080p	40,78%	47,87%	54,61%	60,69%
	2160p	42,36%	48,06%	55,23%	64,99%
	Média	42,09%	48,03%	55,12%	64,24%
27	1080p	41,21%	48,16%	54,62%	60,49%
	2160p	43,49%	48,62%	54,95%	64,36%
	Média	43,08%	48,54%	54,89%	63,68%
32	1080p	42,61%	49,70%	55,61%	60,87%
	2160p	44,88%	50,26%	55,14%	63,39%
	Média	44,46%	50,16%	55,23%	62,93%
37	1080p	43,76%	51,18%	56,88%	61,39%
	2160p	46,12%	52,07%	56,15%	62,62%
	Média	45,68%	51,90%	56,29%	62,39%
Total	1080p	41,97%	49,09%	55,34%	60,83%
	2160p	44,01%	49,54%	55,32%	63,97%
	Média	43,65%	49,46%	55,32%	63,40%

Fonte: Próprio autor.

Tabela B.5: Resultados de redução de acessos para as configurações com *bit dropping* e cortes na área de busca de tamanho 16.

<b>QPs</b>	<b>Vídeos</b>	<b>16-7bpp</b>	<b>16-6bpp</b>	<b>16-5bpp</b>	<b>16-4bpp</b>
22	1080p	59,05%	64,10%	68,90%	73,49%
	2160p	61,63%	65,65%	70,64%	77,25%
	Média	61,18%	65,39%	70,34%	76,60%
27	1080p	59,56%	64,46%	69,04%	73,41%
	2160p	62,57%	66,23%	70,57%	76,90%
	Média	62,04%	65,92%	70,30%	76,28%
32	1080p	60,69%	65,63%	69,89%	73,77%
	2160p	63,63%	67,42%	70,86%	76,35%
	Média	63,10%	67,10%	70,69%	75,88%
27	1080p	61,70%	66,79%	70,92%	74,25%
	2160p	64,48%	68,63%	71,69%	75,91%
	Média	63,96%	68,29%	71,55%	75,60%
Total	1080p	60,15%	65,14%	69,60%	73,70%
	2160p	62,93%	66,83%	70,89%	76,67%
	Média	62,43%	66,52%	70,66%	76,14%

Fonte: Próprio autor.

## APÊNDICE C — RESULTADOS DE TAXA DE COMPRESSÃO PARA OS DIFERENTES TAMANHOS DE TABELA ESTÁTICA DE *HUFFMAN*

Nesse Apêndice são apresentados os resultados de taxa de compressão das 4 configurações considerando os 6 tamanhos de tabelas utilizadas. Os valores apresentados são os resultados médios para as 12 sequências de vídeos, sendo 6 HD 1080p e 6 UHD 2160p, para cada um dos 4 QPs utilizados.

A Tabela C.1 apresenta os resultados de compressão em relação a técnica de *bit dropping*, i.e. é ganho em compressão além do ganho obtido com o *bit dropping*. Nota-se nessa tabela que as 3 configurações que utilizam *7bpp* apresentam resultados semelhantes, independentemente do tamanho da tabela estática utilizada, com pequeno aumento na taxa de compressão conforme aumenta o tamanho da tabela. Entretanto, utilizando a tabela completa (resultados na coluna Completa) a taxa de compressão obtida não é maior. Isso se deve ao fato de que com a tabela completa muitos códigos tem tamanhos maiores que os códigos utilizados nas tabelas com cortes, onde um código de exceção é utilizado junto com o resíduo. Além disso, a configuração *16-4bpp* apresenta maiores taxas de compressão que as outras configurações. Isso já era esperado uma vez que essa configuração são gerados mais resíduos com valor zero ou próximo de zero, que são os valores de resíduos com os menores códigos de *Huffman*

A Tabela C.2 apresenta os resultados de compressão em relação as amostras originais, onde não é aplicado nenhum tipo de redução de informação. Desta forma, esses resultados já consideram a redução obtida com o *bit dropping*. Com isso, as taxas de redução apresentadas nessa tabela são maiores do que as taxas apresentadas na Tabela C.1. O comportamento dos resultados são semelhantes aos discutidos sobre a tabela anterior. Entretanto, a taxa de compressão é maior. Nas configurações com *7bpp* a taxa de compressão é aproximadamente 4,8% maior, em média. Já na configuração *16-4bpp* essa taxa aumenta para 17,9%, em média. Esse aumento se deve ao fato dessa configuração utilizar *bit dropping* de 4 bits.

Tabela C.1: Resultados de taxa de compressão em relação ao *bit dropping*, considerando as 4 configurações com 6 tamanhos de tabela estática.

Configuração	QPs	Res.	[-2, 2]	[-4, 4]	[-8, 8]	[-16, 16]	[-32, 32]	Completa
48-7bpp	22	1080	43,98	48,56	50,94	51,39	51,78	48,77
		2160	60,33	62,82	64,28	64,45	64,66	59,91
		$\bar{x}$	52,15	55,70	57,61	57,92	58,22	54,34
	27	1080	50,07	53,71	55,54	55,90	56,20	52,75
		2160	65,80	67,43	68,38	68,49	68,63	63,30
		$\bar{x}$	57,93	60,57	61,96	62,20	62,41	58,03
	32	1080	53,79	56,84	58,36	58,66	58,90	55,28
		2160	67,61	68,95	69,74	69,82	69,93	64,43
		$\bar{x}$	60,70	62,89	64,05	64,24	64,41	59,85
	37	1080	57,69	60,12	61,34	61,58	61,77	57,97
		2160	69,17	70,27	70,91	70,98	71,05	65,39
		$\bar{x}$	63,43	65,19	66,13	66,28	66,41	61,68
32-7bpp	22	1080	43,98	48,56	50,94	51,40	51,79	48,78
		2160	60,33	62,82	64,28	64,45	64,66	59,91
		$\bar{x}$	52,15	55,69	57,61	57,92	58,22	54,34
	27	1080	50,10	53,73	55,56	55,93	56,22	52,76
		2160	65,78	67,42	68,37	68,48	68,61	63,29
		$\bar{x}$	57,94	60,58	61,96	62,20	62,42	58,03
	32	1080	53,83	56,87	58,39	58,69	58,93	55,30
		2160	67,59	68,94	69,73	69,82	69,92	64,42
		$\bar{x}$	60,71	62,90	64,06	64,25	64,42	59,86
	37	1080	57,64	60,09	61,31	61,55	61,74	57,94
		2160	69,13	70,24	70,88	70,95	71,02	65,37
		$\bar{x}$	63,39	65,16	66,10	66,24	66,38	61,66
16-7bpp	22	1080	44,00	48,57	50,95	51,40	51,79	48,78
		2160	60,35	62,84	64,30	64,47	64,67	59,92
		$\bar{x}$	52,17	55,70	57,62	57,94	58,23	54,35
	27	1080	50,12	53,76	55,59	55,95	56,25	52,77
		2160	65,83	67,46	68,42	68,53	68,66	63,32
		$\bar{x}$	57,98	60,61	62,00	62,23	62,45	58,05
	32	1080	53,88	56,92	58,43	58,73	58,97	55,32
		2160	67,60	68,94	69,73	69,81	69,92	64,42
		$\bar{x}$	60,74	62,93	64,08	64,27	64,44	59,87
	37	1080	57,69	60,12	61,35	61,58	61,77	57,95
		2160	69,20	70,30	70,94	71,00	71,08	65,41
		$\bar{x}$	63,45	65,21	66,14	66,29	66,43	61,68
16-4bpp	22	1080	57,62	58,80	59,84	59,91	60,00	46,22
		2160	65,02	65,49	66,05	66,11	66,15	49,56
		$\bar{x}$	61,32	62,14	62,94	63,01	63,08	47,89
	27	1080	59,38	60,51	61,44	61,51	61,59	47,08
		2160	66,33	66,82	67,30	67,36	67,41	50,16
		$\bar{x}$	62,85	63,67	64,37	64,44	64,50	48,62
	32	1080	60,58	61,63	62,48	62,55	62,63	47,64
		2160	66,77	67,26	67,72	67,78	67,82	50,33
		$\bar{x}$	63,68	64,44	65,09	65,17	65,23	48,98
	37	1080	61,97	62,89	63,65	63,73	63,80	48,29
		2160	67,19	67,66	68,11	68,17	68,22	50,47
		$\bar{x}$	64,58	65,28	65,87	65,95	66,01	49,38

Fonte: Próprio autor.

Tabela C.2: Resultados de taxa de compressão em relação as amostras originais, considerando as 4 configurações com 6 tamanhos de tabela estática.

Configuração	QPs	Res.	[-2, 2]	[-4, 4]	[-8, 8]	[-16, 16]	[-32, 32]	Completa
48-7bpp	22	1080	50,98	54,99	57,07	57,47	57,81	55,18
		2160	65,29	67,47	68,74	68,90	69,07	64,92
		$\bar{x}$	58,13	61,23	62,91	63,18	63,44	60,05
	22	1080	56,31	59,50	61,10	61,42	61,68	58,65
		2160	70,07	71,50	72,34	72,43	72,55	67,89
		$\bar{x}$	63,19	65,50	66,72	66,92	67,11	63,27
	32	1080	59,57	62,23	63,57	63,83	64,04	60,87
		2160	71,65	72,83	73,53	73,60	73,69	68,88
		$\bar{x}$	65,61	67,53	68,55	68,71	68,86	64,87
	37	1080	62,98	65,11	66,18	66,38	66,55	63,22
		2160	73,02	73,98	74,55	74,60	74,67	69,72
		$\bar{x}$	67,99	69,54	70,36	70,49	70,60	66,47
32-7bpp	22	1080	50,99	54,99	57,07	57,47	57,81	55,18
		2160	65,28	67,47	68,75	68,90	69,08	64,92
		$\bar{x}$	58,14	61,23	62,91	63,18	63,44	60,05
	22	1080	56,33	59,52	61,12	61,44	61,70	58,67
		2160	70,06	71,49	72,33	72,42	72,54	67,88
		$\bar{x}$	63,20	65,50	66,72	66,92	67,12	63,27
	32	1080	59,61	62,26	63,59	63,85	64,06	60,89
		2160	71,64	72,82	73,52	73,59	73,68	68,87
		$\bar{x}$	65,62	67,54	68,55	68,72	68,87	64,88
	37	1080	62,94	65,08	66,15	66,35	66,52	63,20
		2160	72,99	73,96	74,52	74,58	74,65	69,70
		$\bar{x}$	67,96	69,52	70,33	70,46	70,58	66,45
16-7bpp	22	1080	51,00	55,00	57,08	57,48	57,82	55,19
		2160	65,30	67,49	68,76	68,91	69,09	64,93
		$\bar{x}$	58,15	61,24	62,92	63,19	63,45	60,06
	22	1080	56,36	59,54	61,14	61,46	61,72	58,68
		2160	70,11	71,53	72,36	72,46	72,57	67,91
		$\bar{x}$	63,23	65,54	66,75	66,96	67,15	63,29
	32	1080	59,64	62,30	63,63	63,89	64,10	60,91
		2160	71,65	72,82	73,52	73,59	73,68	68,87
		$\bar{x}$	65,65	67,56	68,57	68,74	68,89	64,89
	37	1080	62,98	65,11	66,18	66,39	66,55	63,21
		2160	73,05	74,01	74,57	74,63	74,70	69,73
		$\bar{x}$	68,01	69,60	70,38	70,50	70,62	66,47
16-4bpp	22	1080	78,81	79,40	79,92	79,96	80,00	73,11
		2160	82,51	82,74	83,02	83,06	83,08	74,78
		$\bar{x}$	80,66	81,07	81,47	81,50	81,54	73,95
	22	1080	79,69	80,26	80,72	80,76	80,80	73,54
		2160	83,16	83,41	83,65	83,68	83,70	75,08
		$\bar{x}$	81,43	81,83	82,18	82,22	82,25	74,31
	32	1080	80,29	80,82	81,24	81,28	81,32	73,82
		2160	83,39	83,63	83,86	83,89	83,91	75,17
		$\bar{x}$	81,84	82,22	82,55	82,58	82,61	74,49
	37	1080	80,98	81,44	81,82	81,86	81,90	74,15
		2160	83,60	83,83	84,05	84,09	84,11	75,24
		$\bar{x}$	82,29	82,64	82,94	82,97	83,00	74,69

Fonte: Próprio autor.

**APÊNDICE D — SEQUÊNCIAS DE VÍDEO**

Tabela D.1: Sequências HD 1080p utilizadas para os experimentos e para os resultados.

Sequência	Descrição da sequência de vídeo
	<p>BasketballDrive. Um grupo de jogadores jogam uma partida de basquete. Os jogadores se movem constantemente. A câmera fica em movimento sempre seguindo os jogadores.</p>
	<p>BQTerrace. Uma câmera distante mostra um terraço com pessoas. A câmera apresenta uma movimentação lenta e constante para cima, mostrando uma estrada onde carros se movem em ambas as direções.</p>
	<p>Cactus. A câmera está parada filmando uma cena com muitos objetos, incluindo um pequeno cacto que gira verticalmente. Além disso, as cartas de baralho ao fundo ficam girando e um pendulo com dois tigres fica girando rápido na cena.</p>
	<p>Kimono. Uma mulher caminha pela floresta vestindo um kimono. Na metade da sequência há uma mudança de cena, mostrando uma casa em estilo japonês.</p>
	<p>ParkScene. Parque arborizado com ciclistas em movimento. A câmera se movimenta lentamente.</p>
	<p>Tennis. A câmera em movimento atrás de uma grade mostra algumas pessoas se preparando para jogar tênis. Na metade da sequência a câmera muda para a quadra acompanhando as jogadoras.</p>

Fonte: Próprio autor.

Tabela D.2: Sequências HD 1080p utilizadas para os resultados.

Sequência	Descrição da sequência de vídeo
	<p>BlueSky. A câmera em movimento circular mostra o céu azul e a copa de uma árvore. Na metade final da sequência apenas a copa da árvore é mostrada.</p>
	<p>Rolling. A câmera está fixa em uma superfície onde tomates molhados rolam próximos a câmera.</p>
	<p>RushHour. A câmera está fixa no meio de uma avenida repleta de automóveis. Na metade final da sequência a câmera é movida para baixo mostrando alta movimentação desses veículos na horizontal.</p>
	<p>Sunflower. A câmera com pequena movimentação acompanha uma abelha coletando néctar e pólen em um girassol.</p>
	<p>Tractor. No início a câmera está focada na movimentação do trator e da plantadeira, na metade da sequência o foco é a plantadeira e no final da sequência a câmera reduz o zoom mostrando uma grande área agrícola.</p>
	<p>Traffic. A câmera acompanha o trânsito de veículos em uma rodovia. Na metade final da sequência há uma transição da rodovia para a copa de uma árvore que balança com o vento.</p>

Fonte: Próprio autor.

Tabela D.3: Sequências HD 2160p utilizadas para os experimentos e para os resultados.

Sequência	Descrição da sequência de vídeo
	<p>CampFire. Um grupo de pessoas está reunido próximo a uma fogueira, uma pessoas do grupo caminha entre o grupo e a fogueira. No final do vídeo a câmera se aproxima das pessoas.</p>
	<p>CatRobot. Nesta sequência a câmera está fixa e a cena é composta por lenços que estão sempre em movimento ao fundo, uma roleta no canto inferior que está girando, um pequeno robô e um livro aberto. Na parte final, o robô e as páginas do livro também se movimentam.</p>
	<p>DaylightRoad. A câmera está no interior de um veículo que está em movimento em uma avenida com outros carros também em movimento. Na cena há também muitos prédios e pessoas ao longo da avenida.</p>
	<p>FoodMarket. No início a câmera mostra os rostos de algumas pessoas em uma feira de rua. Após há uma transição para os produtos da feira e depois a câmera vai reduzindo o zoom, mostrando as pessoas e a feira.</p>
	<p>ParkRunning. A câmera acompanha duas pessoas correndo em uma pista dentro de um parque bastante arborizado. Outras pessoas também circulam nesse parque.</p>
	<p>Tango. A câmera está parada filmando um grupo de pessoas dançando em um lugar aberto. As pessoas estão bem próximas a câmera.</p>

Fonte: Próprio autor.

Tabela D.4: Sequências HD 2160p utilizadas para os resultados.

Sequência	Descrição da sequência de vídeo
	<p>Beauty. A câmera está fixa e bem próxima de um rosto feminino que faz movimentos lentos para lado enquanto o cabelo está sempre em movimento. O fundo da imagem é totalmente escuro.</p>
	<p>Bosphorus. A câmera acompanha um barco que está navegando em um lago. Ao fundo aparecem outros barcos, montanhas arborizadas e uma ponte com tráfego de veículos.</p>
	<p>CityAlley. A câmera está filmando uma torre entre prédios em uma estreita rua, fazendo uma movimentação lenta para cima.</p>
	<p>HoneyBee. A câmera está fixa filmando uma abelha que voa entre flores, praticamente o único movimento na cena é o da abelha.</p>
	<p>ShakeNDry. O vídeo mostra um cão se sacudindo para remover a água do corpo, os movimentos são rápidos, a câmera está fixa e tem como cenário de fundo um bosque. Na parte final do vídeo o cão começa a andar saindo da imagem.</p>
	<p>Twilight. A cena acompanha um bonde chegando a uma estação ao entardecer. A câmera está fixa, a cena é bastante escura e na parte final da cena nota-se movimentos de pessoas próximas ao bonde.</p>

Fonte: Próprio autor.

## APÊNDICE E — PUBLICAÇÕES

### Livro

1. **Dieison Silveira**: *Redução do consumo de energia em codificadores de vídeo digital: algoritmos e arquiteturas para a compressão de quadros de referência*. Editora: UFPEL - FAU - Fundação de Apoio Universitário, 2017.

### Revistas

1. **Dieison Silveira**, Livia Amaral, Guilherme Povala, Bruno Zatt, Luciano Agostini, Marcelo Porto, Sergio Bampi: *Low-Energy Motion Estimation Memory System with Dynamic Management*. Springer Journal of Real-Time Image Processing, pp. 1-16, Junho 2021. Disponível em: <<https://doi.org/10.1007/s11554-021-01138-3>>
2. **Dieison Silveira**, Guilherme Povala, Livia Amaral, Bruno Zatt, Luciano Agostini, Marcelo Porto: *Efficient reference frame compression scheme for video coding systems: algorithm and VLSI design*. Springer Journal of Real-Time Image Processing, v. 16, n. 2, pp. 391-411, Abril 2019. Disponível em: <<https://doi.org/10.1007/s11554-015-0551-1>>
3. **Dieison Silveira**, Bruno Zatt, Luciano Agostini, Marcelo Porto: *Reference frame context-adaptive variable-length coder: a real-time hardware-friendly approach for lossless external memory bandwidth reduction in current video-coding systems*. Springer Journal of Real-Time Image Processing, v. 14, n. 2, pp. 249-265, Fevereiro 2019. Disponível em: <<https://doi.org/10.1007/s11554-014-0443-9>>

### Conferências

1. **Dieison Silveira**, Ana Mativi, Marcelo Porto, Sergio Bampi: *Energy savings with non-volatile memory system for high definition video encoders*. International NEW Circuits and Systems Conference (NEWCAS), 2019. Disponível em: <<https://doi.org/10.1109/NEWCAS44328.2019.8961238>>
2. **Dieison Silveira**, Marcelo Porto, Sergio Bampi: *Performance and energy consumption analysis of the X265 video encoder*. European Signal Processing Conference (EUSIPCO), 2017. Disponível em: <<https://doi.org/10.23919/EUSIPCO.2017.8081463>>

3. **Dieison Silveira**, Sergio Bampi, Gabriel Moro, Eduardo H. M. da Cruz, Philippe O. A. Navaux, Lucas Schnorr: *System energy analysis for shared memory multiprocessing applications*. IEEE International Conference on Electronics, Circuits and Systems (ICECS), 2016. Disponível em: <<https://doi.org/10.1109/ICECS.2016.7841318>>
4. Livia Amaral, Guilherme Povala, Marcelo Porto, **Dieison Silveira**, Sergio Bampi: *Memory energy consumption analyzer for video encoder hardware architectures*. IEEE International Conference on Electronics, Circuits and Systems (ICECS), 2016. Disponível em: <<https://doi.org/10.1109/ICECS.2016.7841203>>
5. **Dieison Silveira**, Gabriel Moro, Eduardo H. M. da Cruz, Philippe O. A. Navaux, Lucas Schnorr, Sergio Bampi: *Energy consumption estimation in parallel applications: an analysis in real and theoretical models*. Simpósio em Sistemas Computacionais de Alto Desempenho (WSCAD), 2016. Disponível em: <<https://doi.org/10.5753/wscad.2016.14249>>

#### **Produção técnica**

1. **Dieison Silveira**, Vignesh V. Menon: *Performance and energy consumption analysis of x265*. x265 Blog: <<http://x265.org/performance-energy-consumption-analysis-x265/>>, 2018.