UNIVERSIDADE FEDERAL DE PELOTAS
Centro de Desenvolvimento Tecnológico
Programa de Pós-Graduação em Computação

Tese

Redução de Consumo Energético para Transformadas do Padrão *Versatile Video Coding* com Auxílio de Aprendizado de Máquina Supervisionado

Bianca Santos da Cunha da Silveira

Pelotas, 2025

Bianca Santos da Cunha da Silveira

Redução de Consumo Energético para Transformadas do Padrão *Versatile Video Coding* com Auxílio de Aprendizado de Máquina Supervisionado

Tese apresentada ao Programa de Pós-Graduação em Computação do Centro de Desenvolvimento Tecnológico da Universidade Federal de Pelotas, como requisito parcial à obtenção do título de Doutor em Ciência da Computação.

Orientador: Prof. Dr. Guilherme Ribeiro Corrêa
Coorientador: Prof. Dr. Cláudio Machado Diniz
Colaborador: Prof. Dr. Daniel Munari Vilchez Palomino

Pelotas, 2025

Universidade Federal de Pelotas / Sistema de Bibliotecas
Catalogação da Publicação

S587r Silveira, Bianca Santos da Cunha da

Redução de consumo energético para transformadas do padrão *Versatile Video Coding* com auxílio de aprendizado de máquina supervisionado [recurso eletrônico] / Bianca Santos da Cunha da Silveira ; Guilherme Ribeiro Corrêa, orientador ; Cláudio Machado Diniz, Daniel Munari Vilchez Palomino, coorientadores. — Pelotas, 2025.
154 f. : il.

Tese (Doutorado) — Programa de Pós-Graduação em Computação, Centro de Desenvolvimento Tecnológico, Universidade Federal de Pelotas, 2025.

1. VVC. 2. MTS. 3. Codificação de vídeo. 4. Aprendizado de máquina. I. Corrêa, Guilherme Ribeiro, orient. II. Diniz, Cláudio Machado, coorient. III. Palomino, Daniel Munari Vilchez, coorient. IV. Título.

CDD 005

Elaborada por Maria Inez Figueiredo Figas Machado CRB: 10/1612

Bianca Santos da Cunha da Silveira

Redução de Consumo Energético para Transformadas do Padrão *Versatile Video Coding* com Auxílio de Aprendizado de Máquina Supervisionado

Tese aprovada, como requisito parcial, para obtenção do grau de Doutor em Ciência da Computação, Programa de Pós-Graduação em Computação, Centro de Desenvolvimento Tecnológico, Universidade Federal de Pelotas.

Data da Defesa: 27 de junho de 2025

Banca Examinadora:

Prof. Dr. Guilherme Ribeiro Corrêa (orientador)
Doutor em Engenharia Electrotécnica e de Computadores pela Universidade de Coimbra.

Prof. Dr. Mateus Grellert da Silva
Doutor em Computação pela Universidade Federal do Rio Grande do Sul.

Prof. Dr. Fábio Luís Livi Ramos
Doutor em Computação pela Universidade Federal do Rio Grande do Sul.

Prof. Dr. Marcelo Schiavon Porto
Doutor em Computação pela Universidade Federal do Rio Grande do Sul.

AGRADECIMENTOS

Ao meu orientador Guilherme: obrigada por segurar essa barra comigo! A ti e aos coorientadores, Cláudio e Daniel, por compartilharem tanto conhecimento e por todas as revisões (algumas bem sofridas, confesso), mas sempre essenciais pra fazer esse projeto acontecer.

À Vania e ao Paulo, vulgo mamys e papito, obrigada por estarem sempre comigo, torcendo, apoiando, dando suporte nos momentos em que tudo parecia desmoronar. Sem vocês, nada disso faria sentido.

Ao Leo, meu parceiro de vida, obrigada pela paciência nas crises, pelo apoio técnico (que salvou algumas vezes!) e por estar por perto com tanto carinho quando tudo que eu queria era largar tudo.

Aos amigos que aturaram meus sumiços, a falta de paciência e ainda assim continuaram ali: vocês são incríveis. Lu, um obrigada extra a ti pelo capricho com minhas apresentações, gráficos e tudo mais que tu deste aquele toque de beleza e organização que faltava.

Ao pessoal do doutorado: obrigada por cada conversa, dica, risada e apoio. Murilo, tu foste fundamental , sempre presente, sempre disposto a ajudar, mesmo quando eu estava surtando. Alex, obrigada pelas palavras sempre tão gentis e pela ajuda nos momentos críticos. Adson, Luís, Roberta e Mário, valeu demais pelas contribuições, sugestões e por estarem comigo nessa jornada.

Às minhas meninas do coração, Bruna e Caroline. Bruna, minha pupila querida, obrigada por ser companhia, força e luz nos dias pesados. Carol, minha anja, obrigada por emprestar teu conhecimento e me ajudar a dar sentido a muita coisa aqui.

E a todo mundo que cruzou meu caminho, mesmo que por pouco tempo, mas deixou alguma contribuição nesse projeto: obrigada de coração.

RESUMO

SANTOS DA CUNHA DA SILVEIRA, Bianca. **Redução de Consumo Energético para Transformadas do Padrão *Versatile Video Coding* com Auxílio de Aprendizado de Máquina Supervisionado**. Orientador: Guilherme Ribeiro Corrêa. 2025. 154 f. Tese (Doutorado em Ciência da Computação) – Centro de Desenvolvimento Tecnológico, Universidade Federal de Pelotas, Pelotas, 2025.

O padrão de codificação de vídeo *Versatile Video Coding* foi lançado pelo *Joint Video Experts Team* em 2020, incluindo diversas ferramentas para melhorar a eficiência de compressão em relação a padrões anteriores. Uma das principais inovações é a *Multiple Transform Selection*, que permite ao codificador escolher entre diferentes tipos de transformadas para melhor se adequar às características locais do sinal de vídeo. A *Multiple Transform Selection* utiliza a transformada discreta do cosseno tipo II, a transformada discreta do cosseno tipo VIII e a transformada discreta do seno tipo VII, possibilitando ainda combinações distintas dessas transformadas nas direções horizontal e vertical. Embora essa flexibilidade proporcione ganhos em eficiência de compressão, ela também impõe um aumento significativo no custo computacional, já que diversas combinações de transformadas e tamanhos de blocos devem ser avaliadas pelo codificador. Diante desses desafios, esta tese propõe o desenvolvimento de arquiteturas de *hardware* dedicadas ao módulo da *Multiple Transform Selection* do codificador *Versatile Video Coding*, com foco na redução do consumo energético e na viabilidade de compressão em tempo real. O projeto é estruturado em três etapas principais: uma análise detalhada da usabilidade da *Multiple Transform Selection* no *software* de referência do *Versatile Video Coding*, a integração de modelos preditivos baseados em aprendizado de máquina ao fluxo de codificação, e a implementação de arquiteturas de *hardware* otimizadas a partir dos dados extraídos do codificador. Para reduzir a complexidade do processo de seleção das transformadas, foram desenvolvidos modelos preditivos utilizando algoritmos de aprendizado de máquina. Esses modelos foram treinados com dados extraídos diretamente do codificador de referência, e sua função é antecipar quais transformadas são mais prováveis de serem escolhidas em cada situação. Essa predição permite desabilitar transformadas desnecessárias, reduzindo o número de combinações testadas e, consequentemente, o tempo de processamento e o consumo energético. A arquitetura de *hardware* proposta foi projetada para suportar tanto o fluxo tradicional do *software* quanto o fluxo modificado com os modelos preditivos. A tese apresenta a metodologia de extração e seleção de *features*, o treinamento dos modelos, a integração ao codificador e os resultados de consumo energético e área para diferentes configurações e resoluções de vídeo. Os testes demonstram que, mesmo com

uma pequena perda de 0,89% na eficiência de codificação, a adoção dos modelos preditivos resultou em reduções expressivas de até 7,98%, em média, no tempo de processamento quando implementada no software de referência. Adicionalmente, foi discutido o potencial de implementação dos modelos preditivos em hardware, utilizando estruturas condicionais simples, possibilitando sua integração eficiente a sistemas embarcados com recursos computacionais limitados. Observa-se que a abordagem híbrida proposta, combinando aprendizado de máquina e arquitetura de *hardware* otimizada, representa uma estratégia promissora para a viabilização de codificadores *Versatile Video Coding* energeticamente eficientes, atingindo reduções de até 71,37% em consumo energético para resoluções de 4K. Esta contribuição é relevante tanto para aplicações em dispositivos portáteis quanto para cenários de compressão em tempo real em alta resolução.

Palavras-chave: VVC; MTS; Codificação de vídeo; Aprendizado de máquina.

ABSTRACT

SANTOS DA CUNHA DA SILVEIRA, Bianca. **Energy Consumption Reduction for Transforms in the Versatile Video Coding Standard Using Supervised Machine Learning**. Advisor: Guilherme Ribeiro Corrêa. 2025. 154 f. Thesis (Doctorate in Computer Science) – Technology Development Center, Federal University of Pelotas, Pelotas, 2025.

The Versatile Video Coding standard was released by the Joint Video Experts Team in 2020, introducing several tools aimed at improving compression efficiency compared to previous standards. One of the innovations is the Multiple Transform Selection, which enables the encoder to choose between different types of transforms to better adapt to the local characteristics of the video signal. Multiple Transform Selection employs the discrete cosine transform type II , discrete cosine transform type VIII, and discrete sine transform type VII, also allowing distinct combinations of these transforms in the horizontal and vertical directions. Although this flexibility improves compression efficiency, it also significantly increases computational complexity, since multiple combinations of transforms and block sizes must be evaluated by the encoder. In light of these challenges, this thesis proposes the development of dedicated hardware architectures for the Multiple Transform Selection module of the Versatile Video Coding encoder, focusing on reducing energy consumption and enabling real-time compression. The project is structured in three main stages: a detailed analysis of Multiple Transform Selection usability in the Versatile Video Coding reference software, the integration of machine learning-based predictive models into the encoding flow, and the implementation of optimized hardware architectures based on data extracted from the encoder. To reduce the complexity of the transform selection process, predictive models were developed using machine learning algorithms. These models were trained on data directly extracted from the Versatile Video Coding reference encoder, and their purpose is to anticipate which transforms are most likely to be selected in each situation. This prediction enables unnecessary transforms to be disabled, reducing the number of combinations tested and consequently lowering processing time and energy consumption. The proposed hardware architecture was designed to support both the traditional software execution flow and the modified flow that incorporates predictive models. The thesis presents the methodology for feature extraction and selection, model training, integration into the VTM encoder, and energy and area results across different configurations and video resolutions. The results demonstrate that, despite a slight increase in bitrate, the adoption of predictive models led to significant reductions in encoding time and energy consumption. The tests demonstrate that, even with a

small loss of 0.89% in coding efficiency, the adoption of predictive models resulted in significant reductions, averaging up to 7.98%, in processing time when implemented in the reference software. Additionally, the potential implementation of predictive models in hardware is discussed, using simple conditional structures, which allows their efficient integration into embedded systems with limited computational resources. The proposed hybrid approach, combining machine learning and optimized hardware architecture, represents a promising strategy for enabling energy-efficient Versatile Video Coding encoders, achieving energy consumption reductions of up to 71.37% for 4K resolutions. This contribution is relevant for both portable device applications and real-time, high-resolution video compression scenarios.

Keywords: VVC; MTS; Video Coding; Machine Learning.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

1D	*One Dimension*
2D	*Two Dimension*
ACP	*Análises de Componentes Principais*
AI	*All Intra*
AMT	*Adaptative Multiple Transform*
ASIC	*Application-Specific Integrated Circuit*
BD-BR	*Bjontegaard-Delta Rate*
CCLM	*Cross-component linear model*
CPU	*Central Processing Unit*
CTC	*Commom Test Conditions*
CU	*Coding Unit*
DCT	*Discrete Cosine Transform*
DST	*Discrete Sine Transform*
FHD	*Full High Definition*
FPGA	*Field-Programmable Gate Array*
fps	*frames per second*
GPP	*General-purpose processors*
GPU	*Graphical Processing Unit*
HEVC	*High Efficiency Video Coding*
IEEE	*Institute of Electrical and Electronics Engineers*
JEM	*Joint Exploration test Model*
JVET	*Joint Video Experts Team*
LDB	*Low Delay B*
LFNST	*Low Frequency Non-Separable Transform*
MPEG	*Moving Picture Experts Group*
MCM	*Multiple Constant Multiplication*

MTS *Multiple Transform Selection*
NUMA *Non-Uniform Memory Access*
PSNR *Peak signal to noise ratio*
PSO *Particle Swarm Optimization*
QP *Quantization parameter*
RA *Random Access*
RAG-n *N-Dimensional Reduced Adder Graph*
RAM *Random-Access Memory*
RDO *Rate Distortion Optimization*
RM *Regular Multipliers*
RRC *Regular Residual Coding*
SIMD *Single Instruction Multiple Data*
SMP *Symmetric Multi-Processor*
SPMD *Single Program Multiple Data*
SPS *Sequence Parameter Set*
TCC *Transform Coefficient Coding*
TCQ *Trellis-coded quantization*
TGM *Text and Graphics with Motion*
TSM *Transform Skip Mode*
TSRC *Transform Skip Residual Coding*
UBC *Unary Bitplane Coding*
URQ *Uniform-Reconstruction Quantizers*
VCEG *Video Coding Experts Group*
VHDL *VHSIC Hardware Description Language*
VTM *VVC Test Model*
VVC *Versatile Video Coding*

SUMÁRIO

1 INTRODUÇÃO

A crescente popularização dos vídeos digitais tem impulsionado transformações significativas no cenário multimídia, com a disseminação acelerada de aplicativos voltados à manipulação e transmissão de conteúdos audiovisuais. Esse avanço é acompanhado pelo aumento das expectativas dos usuários em relação à qualidade e ao tempo de carregamento dos vídeos, desafio que se intensifica com a adoção de formatos de alta resolução, como *Ultra-High-Definition* (UHD) 4K e 8K (Cisco, 2020). Segundo o relatório mais recente da Ericsson (Ericsson, 2024), no final de 2024, foi estimado que o tráfego de vídeo representaria 74% de todo o tráfego de dados móveis, evidenciando a crescente demanda por conteúdos audiovisuais em redes móveis globais. Esse crescimento não apenas evidencia a predominância desse formato na experiência digital dos usuários, mas também reforça sua posição como um dos principais responsáveis pela demanda crescente por infraestrutura de rede mais eficiente. Além do impacto no tráfego de dados, a elevada exigência de armazenamento impõe desafios adicionais. Um vídeo de cinco minutos em resolução *Full High Definition* (FHD-1920x1080) a 60 *frames* por segundo (*fps*), com 24 bits por *pixel*, pode ocupar aproximadamente 112 GB de espaço. Diante desse cenário, os métodos de compressão de vídeo desempenham um papel essencial na viabilização da transmissão e armazenamento eficientes desses conteúdos, reduzindo significativamente a quantidade de dados sem comprometer a qualidade visual.

Os codificadores de vídeo desempenham um papel fundamental na transmissão eficiente de vídeo digital, especialmente diante da crescente demanda por *streaming* em alta definição. Com a popularização de resoluções cada vez maiores, torna-se essencial o desenvolvimento de algoritmos avançados de compressão capazes de reduzir significativamente a taxa de bits sem comprometer a qualidade visual. Nesse contexto, a pesquisa em novos padrões de codificação tem se intensificado, buscando equilibrar eficiência de compressão e custo computacional. Entre os mais recentes avanços na área, destaca-se o *Versatile Video Coding* (VVC), um padrão de compressão de vídeo finalizado em julho de 2020. O VVC incorpora diversas novas ferramentas de codificação, permitindo uma redução de até 40% na taxa de bits em compara-

ção ao seu antecessor, o *High Efficiency Video Coding* (HEVC) (Sidaty; Hamidouche; Déforges; Philippe; Fournier, 2019). No entanto, esses ganhos de compressão vêm acompanhados por um aumento significativo no custo computacional do processo de codificação, exigindo estratégias avançadas de otimização algorítmica e soluções eficientes de implementação em *hardware* para viabilizar sua aplicação prática, especialmente em dispositivos com restrições de energia e processamento.

Apesar de o padrão VVC incorporar diversas inovações para aprimorar a eficiência da compressão, uma das mais relevantes é a *Multiple Transform Selection* (MTS), utilizada nas etapas de predição intraquadro e interquadros. Essa técnica amplia a flexibilidade do processo das transformadas, permitindo uma melhor adaptação às características do sinal residual, o que resulta em uma maior eficiência na codificação. O conjunto de transformadas disponíveis na MTS inclui as Transformadas Discretas do Cosseno tipo II e tipo VIII (DCT-II e DCT-VIII) e a Transformada Discreta do Seno tipo VII (DST-VII), nas versões diretas e inversas, com tamanhos de bloco de até 64 × 64 para a DCT-II e 32 × 32 para a DCT-VIII e DST-VII. A aplicação dessas transformadas segue um modelo separável, permitindo que a operação seja realizada independentemente nas direções vertical e horizontal. Dessa forma, quando a DST-VII é escolhida para a direção vertical, o codificador pode optar entre a DCT-VIII ou a DST-VII na direção horizontal. Já a DCT-II, quando selecionada, é aplicada simultaneamente em ambas as direções, garantindo um processamento mais eficiente para diferentes padrões de correlação espacial dos blocos de vídeo.

A introdução da técnica MTS no padrão VVC representou um avanço significativo na eficiência da compressão de vídeo. Esse progresso decorre do fato de que, durante a codificação, o sistema precisa avaliar e testar uma série de transformadas, considerando diferentes combinações e tamanhos de blocos, com o objetivo de identificar a configuração que melhor se adapta a cada quadro de vídeo. O processo de seleção das transformadas é crucial para alcançar uma compressão eficiente, já que a escolha correta pode reduzir substancialmente a quantidade de bits necessária para representar um vídeo sem comprometer a qualidade visual. No entanto, esse ganho em eficiência trouxe consigo um aumento considerável no custo computacional, principalmente devido ao incremento no número de operações aritméticas, como adições e multiplicações, que passaram a ser necessárias nas etapas de codificação. Esse aumento no número de operações aritméticas impõe desafios substanciais para a implementação em *software*, uma vez que a capacidade de processamento dos sistemas convencionais muitas vezes não é suficiente para lidar com a carga computacional imposta por essas operações. Assim, é imprescindível a consideração de soluções baseadas em *hardware* para garantir a aceleração do processo de codificação de vídeo, com o objetivo de reduzir o tempo de processamento sem prejudicar a eficiência da compressão.

A implementação eficiente da MTS em *hardware* exige um equilíbrio delicado entre desempenho e consumo de recursos, já que o suporte às múltiplas transformadas afeta diretamente a alocação de memória e os elementos lógicos necessários para a operação (Kammoun; Hamidouche; Philipp; Belghith; Massmoudi; Nezan, 2019). Nesse contexto, arquiteturas de *hardware* otimizadas se tornam fundamentais para viabilizar a codificação em tempo real, permitindo uma redução significativa na sobrecarga computacional, mas sem comprometer a eficiência geral do padrão VVC. Como evidenciado na revisão bibliográfica realizada ao longo do desenvolvimento deste trabalho, observou-se uma tendência na literatura em direção à criação de circuitos dedicados e aceleradores de *hardware* para a etapa de transformada no codificador VVC. Esses estudos destacam que tais implementações são capazes de contemplar os requisitos em termos de custo computacional não contemplados pelas soluções em *software*, promovendo maior eficiência e desempenho em sistemas de codificação de vídeo em tempo real.

Em paralelo a essas necessidades de implementação em *hardware*, a crescente demanda por serviços de vídeo, impulsionada principalmente pelo aumento do consumo de vídeos em alta definição, tem levado a indústria a buscar métodos para reduzir o espaço de otimização e o esforço computacional sem prejudicar em demasia a eficiência de compressão. Um dos maiores desafios nesse cenário tem sido reduzir o tempo investido nas etapas de codificação, sem comprometer a qualidade final do vídeo. Para lidar com esses desafios, algoritmos de aprendizado de máquina (ML) têm se tornado ferramentas cada vez mais importantes na área de multimídia, devido à sua capacidade de aprender com grandes volumes de dados e tomar decisões inteligentes baseadas nesse aprendizado, sem a necessidade de programação explícita para cada tarefa específica.

O aprendizado de máquina, com seu potencial para identificar padrões ocultos em grandes conjuntos de dados e adaptar-se a novos dados em tempo real, tem sido amplamente utilizado para otimizar o processo de codificação de vídeo. Ele pode ser aplicado para melhorar a seleção de parâmetros de codificação, como a escolha das transformadas e a adaptação dinâmica aos diferentes tipos de conteúdo de vídeo. Ao integrar algoritmos de aprendizado de máquina nos codificadores de vídeo, é possível aumentar a eficiência da compressão, ajustando as configurações de codificação em função das características do vídeo de entrada, o que ajuda a minimizar o número de bits necessários para representar o conteúdo visual sem comprometer a qualidade da imagem.

A interseção entre aprendizado de máquina e arquiteturas eficientes de *hardware* se torna particularmente relevante quando se considera o uso de algoritmos de aprendizado profundo, que podem ser implementados em *hardware* para acelerar ainda mais o processo de codificação. A implementação de tais técnicas em sistemas de

hardware especializados, como FPGAs (*Field-Programmable Gate Arrays*) ou ASICs (*Application-Specific Integrated Circuits*), permite que modelos de aprendizado sejam executados de maneira mais eficiente, com menos sobrecarga computacional e em tempo real. Isso é especialmente importante em contextos de codificação de vídeo em tempo real, onde a velocidade de processamento é crucial para garantir uma transmissão de vídeo de alta qualidade e baixa latência.

Como discutido anteriormente, os codificadores de vídeo, especialmente os baseados no padrão VVC, enfrentam desafios técnicos consideráveis devido ao custo computacional das operações exigidas. A utilização de técnicas de aprendizado de máquina para otimizar esses processos e a implementação dessas técnicas em arquiteturas de *hardware* eficientes surgem como uma solução promissora para lidar com a crescente demanda por vídeos de alta definição, sem comprometer a eficiência de compressão. As arquiteturas de *hardware* e as soluções baseadas em aprendizado de máquina, portanto, são peças-chave para garantir a viabilidade e a eficiência da codificação de vídeo em um mundo cada vez mais dominado por grandes volumes de dados e altas exigências de qualidade.

Neste capítulo, será apresentada a proposta central desta tese, que visa investigar técnicas de *hardware* eficientes para otimizar a etapa de transformadas no codificador VVC. A pesquisa explora diversos aspectos técnicos e teóricos, que serão detalhados ao longo das seções subsequentes. Primeiramente, são discutidas as motivações que impulsionaram a realização deste estudo, destacando sua importância dentro do contexto atual de evolução dos padrões de compressão de vídeo e as crescentes demandas por soluções mais eficientes. Além disso, é apresentada uma hipótese inicial, a qual serve como base para as investigações realizadas ao longo do estudo, oferecendo uma suposição que orienta as análises subsequentes.

A pesquisa é estruturada com o objetivo de testar e validar a hipótese proposta, estabelecendo objetivos gerais e específicos que guiam o desenvolvimento do estudo. Estes objetivos têm como foco a otimização das transformadas no processo de codificação de vídeo, com ênfase na otimização do uso de recursos computacionais e na redução do consumo energético. Para alcançar esses objetivos, são detalhados os métodos e abordagens adotadas, incluindo a coleta e análise de dados, bem como a interpretação dos resultados obtidos. Ao longo deste processo, são consideradas técnicas de aprendizado de máquina, como as árvores de decisão, para ajustar a escolha das transformadas de acordo com cada cenário de codificação, priorizando a redução do consumo energético, mesmo que isso resulte em escolhas sub-ótimas em termos de eficiência de compressão.

1.1 Motivação

A revisão da literatura, que será apresentada no capítulo 3 desta tese, revelou várias tendências e abordagens no campo da otimização do módulo de transformadas do padrão de codificação VVC, destacando um espaço significativo para novas contribuições, tanto no desenvolvimento de soluções em *hardware* quanto em *software*. Embora já existam algumas propostas apresentadas, a revisão da literatura revelou pontos específicos onde há oportunidades para contribuições significativas, as quais serão exploradas ao longo desta tese.

A análise aprofundada dos trabalhos existentes confirmou que o módulo de transformadas representa uma das etapas mais complexas e desafiadoras da codificação segundo o padrão VVC. Esta complexidade não se resume apenas às operações matemáticas envolvidas, mas também à necessidade crítica de selecionar, entre diversas combinações e tamanhos de transformadas possíveis, aquela que proporciona a melhor eficiência na compressão do vídeo. Essa tarefa de tomada de decisão, que deve ser feita em tempo real durante o processo de codificação, torna-se um fator limitante para o desempenho geral do sistema. Consequentemente, a maioria dos estudos identificados visa, de alguma forma, reduzir essa complexidade excessiva, seja por meio de abordagens implementadas em *software* ou em *hardware* dedicado.

Com base nessas constatações, torna-se evidente que a aplicação de técnicas baseadas em aprendizado de máquina surge como uma solução promissora para lidar com a complexidade do processo de decisão, aliada ao grande número de operações envolvidas nas transformadas. A crescente demanda por soluções que não apenas melhorem a eficiência do processo de codificação de vídeo, otimizando o uso de recursos computacionais e acelerando a escolha das transformadas mais adequadas, mas também reduzam o consumo de energia, destaca ainda mais a relevância de técnicas inteligentes. Os modelos de aprendizado de máquina, especialmente aqueles que incorporam modelos preditivos e algoritmos de otimização, oferecem uma abordagem poderosa para resolver problemas complexos que os métodos tradicionais de codificação de vídeo não conseguem abordar de forma eficiente.

Dessa forma, a motivação central desta tese é desenvolver um conjunto de soluções inovadoras em *hardware*, com foco na otimização do módulo de transformadas do VVC, combinando a redução de consumo energético e a minimização da área necessária para a implementação. A proposta é integrar algoritmos de aprendizado de máquina, por meio de modelos preditivos, para auxiliar no processo de decisão, de modo a selecionar a melhor transformada a ser aplicada pelo codificador em diferentes contextos.

1.2 Hipótese e Objetivo

1.2.1 Hipótese

Esta tese é guiada a partir da seguinte hipótese:

A integração de técnicas eficientes de projeto de hardware com modelos preditivos gerados por algoritmos de aprendizado de máquina pode desempenhar um papel estratégico na redução do custo computacional do módulo de transformadas do padrão VVC. Especificamente, acredita-se que essa abordagem é capaz de minimizar o custo associado ao processo de tomada de decisões desse módulo, contribuindo para a diminuição do consumo energético e da área de hardware em troca de uma pequena perda em eficiência de compressão.

1.2.2 Objetivo Geral

O objetivo central desta tese é investigar e desenvolver soluções em *hardware* de baixo consumo energético para o módulo de transformadas do padrão VVC, a partir da integração de modelos preditivos gerados por algoritmos de aprendizado de máquina. A proposta consiste na substituição do processo tradicional de tomada de decisão sobre qual transformada aplicar, que exige a avaliação exaustiva de múltiplas combinações, por um mecanismo preditivo mais leve e direcionado, capaz de reduzir significativamente a carga computacional dessa etapa do codificador.

Assim, este trabalho tem como objetivo desenvolver abordagens baseadas em modelos de árvores de decisão, treinados com características extraídas dos blocos de vídeo, visando permitir a seleção antecipada e eficiente da transformada a ser utilizada no processo de codificação. Busca-se, com isso, viabilizar a implementação de arquiteturas de *hardware* mais simples e com menor consumo energético, mesmo que isso implique em uma leve degradação na eficiência de compressão, assumida como um compromisso estratégico em favor da aplicabilidade prática.

1.2.3 Objetivos Específicos e Contribuições

Para alcançar o objetivo geral desta tese, os seguintes objetivos específicos foram definidos.

- Realizar uma análise aprofundada da literatura existente sobre soluções em *hardware* e *software* voltadas ao módulo de transformadas do padrão VVC, com ênfase em abordagens que visam a redução do custo computacional, do consumo de energia e da área ocupada, destacando estratégias que viabilizem implementações mais eficientes;

- Conduzir um estudo detalhado sobre a complexidade das transformadas no VVC, acompanhado de uma análise de usabilidade da técnica MTS no *software*

de referência do padrão, com o objetivo de avaliar o custo computacional de cada transformada e suas implicações para a eficiência de compressão;

- Propor soluções em *hardware* para o módulo MTS (tanto para as transformadas diretas quanto para as inversas) aplicado à codificação intraquadro e interquadros, capaz de suportar a variedade de tamanhos de blocos definidos no VVC (variando de 4×4 até 32×32, incluindo tamanhos retangulares), garantindo alta eficiência no processamento;
- Treinar modelos preditivos baseados em aprendizado de máquina, que serão responsáveis por selecionar a melhor combinação de transformadas a ser aplicada, considerando características específicas do conteúdo de vídeo;
- Desenvolver uma arquitetura de *hardware* para o módulo MTS, que incorpore os modelos preditivos treinados, com foco em maximizar a eficiência energética e minimizar o uso de área, garantindo a viabilidade do processo de codificação de vídeo em tempo real.

1.3 Organização da Tese

A organização desta tese é estruturada da seguinte forma. O Capítulo 2 apresenta os conceitos fundamentais relacionados à codificação de vídeo, abordando o codificador VVC e as transformadas permitidas segundo este padrão. Também é apresentada uma introdução ao aprendizado de máquina, contextualizando sua aplicação no âmbito da codificação de vídeo. O Capítulo 3 revisa os trabalhos existentes na literatura, com ênfase nos projetos de *software* e *hardware* voltados para o módulo de MTS. O Capítulo 4 se dedica à análise da complexidade e da taxa de usabilidade da MTS, destacando aspectos essenciais para a otimização deste módulo. No Capítulo 5, são discutidas as técnicas de aprendizado de máquina, incluindo a implementação do modelo preditivo, o processo de treinamento e os resultados obtidos com a integração do modelo no *software* de referência VTM. Os Capítulos 6 e 7 são voltados para as arquiteturas de *hardware* propostas para as transformadas direta e inversa do padrão VVC, sendo apresentados os resultados de desempenho de cada uma dessas arquiteturas. Finalmente, o Capítulo 8 traz as conclusões do estudo, além de expor as expectativas para futuros desenvolvimentos e pesquisas na área.

2 FUNDAMENTAÇÃO TEÓRICA

Este capítulo apresenta conceitos relacionados à codificação de vídeo digital, fornecendo o embasamento necessário para a compreensão do novo padrão de codificação de vídeo *Versatile Video Coding* e da pesquisa descrita ao longo desta tese. Inicialmente, serão discutidos os princípios básicos da codificação de vídeo, destacando as principais técnicas utilizadas para a redução da redundância espacial e temporal, essenciais para a eficiência dos *codecs* modernos.

Em seguida, será introduzido o padrão VVC, abordando suas principais inovações e melhorias em relação a padrões anteriores, como o *High Efficiency Video Coding* (HEVC). Dentre os avanços introduzidos, será detalhado o módulo de transformadas do VVC, com ênfase nas diferentes transformadas utilizadas para uma codificação eficiente.

Uma atenção especial será dada à ferramenta *Multiple Transform Selection* (MTS), uma inovação significativa no VVC que permite a seleção dinâmica de diferentes transformadas para melhor adaptabilidade às características do sinal de vídeo. Serão exploradas as vantagens e impactos dessa abordagem na eficiência da compressão.

Por fim, serão discutidas, de forma introdutória, as técnicas de aprendizado de máquina tipicamente aplicadas em codificadores de vídeo, destacando o potencial dessas abordagens para melhorar a eficiência de codificação e para diminuir o custo computacional, tornando o processamento mais rápido.

2.1 Codificação de Vídeo

A codificação de vídeo digital é um processo essencial para a redução da quantidade de dados necessária para transmissão ou armazenamento de sequências de vídeo. Esse processo é fundamental para aplicações que demandam eficiência em termos de largura de banda e capacidade de armazenamento. A codificação de vídeo considera o vídeo representado na sua forma digital, conforme abordado na próxima subseção.

2.1.1 Representação do Vídeo Digital

Um vídeo digital é composto por uma sequência de imagens individuais, conhecidas como quadros (*frames*), que, quando reproduzidas em uma determinada taxa de quadros por segundo (*frames per second - fps)*, geram a ilusão de movimento ao olho humano. Esse princípio se baseia na persistência da visão, onde o sistema visual humano integra informações ao longo do tempo para perceber a continuidade de uma cena. A Figura 1 apresenta um exemplo de uma sequência de vídeo digital, ilustrando a composição de um vídeo a partir de *frames* individuais.

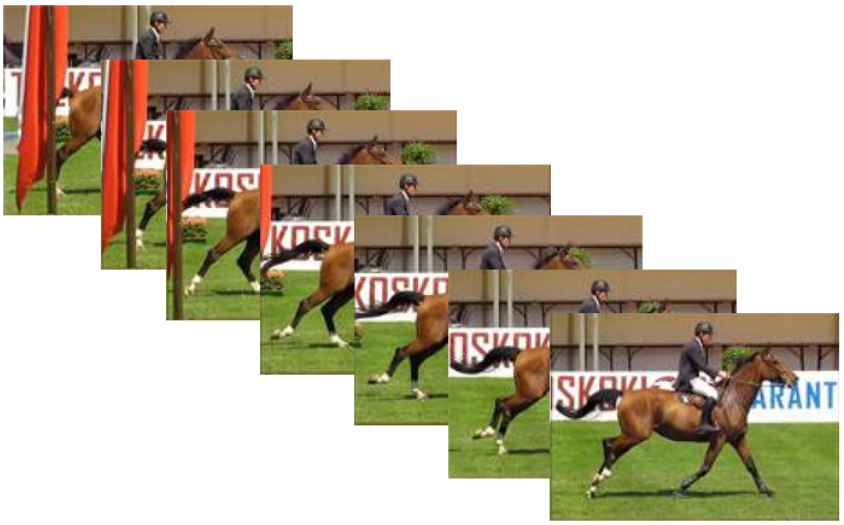

Figura 1 – Sequência de imagens de um vídeo digital.

Um *frame* em um vídeo digital é composto por uma matriz de elementos chamados *pixels*, que representam a menor unidade de uma imagem. Cada *pixel* contém informações essenciais sobre luminosidade e cor, que determinam a aparência visual da imagem quando exibida em uma tela. A Figura 2 ilustra um recorte ampliado de uma imagem, destacando a estrutura dos *pixels* e sua organização dentro do *frame*. Essa representação permite compreender melhor a composição das imagens digitais e a importância da resolução na qualidade final do vídeo.

Figura 2 – Conjunto de *pixels* numa imagem.

A resolução espacial de uma imagem é determinada pela quantidade total de *pixels* presentes em um *frame*, o que se configura como um fator essencial para a qualidade visual de um vídeo. Um maior número de *pixels* resulta em uma maior capacidade de captura de detalhes na imagem, proporcionando maior nitidez e definição. Com o avanço das tecnologias, as resoluções de vídeo têm aumentado consideravelmente, exigindo um volume significativo de dados para serem representadas. As resoluções de vídeo atualmente em uso variam desde a *High Definition* (HD) com 1280x720 *pixels*, passando pela *Full High Definition* (FHD) com 1920x1080 *pixels*, até a *Quad High Definition* (2K) de 2560x1440 *pixels*. A resolução conhecida como *Ultra High Definition* inclui o 4K (3840x2160 *pixels*) e, por fim, a 8K (7680x4320 *pixels*), que representa a resolução mais alta entre as mencionadas. Essas resoluções são ilustradas na Figura 3.

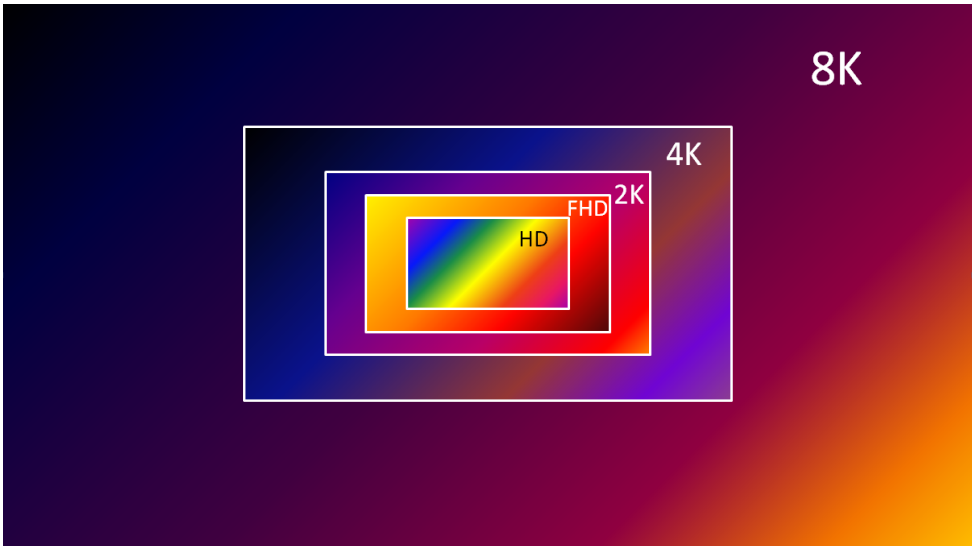

Figura 3 – Diferentes resoluções espaciais de vídeo.

Conforme discutido anteriormente, os *pixels* podem ser compostos por amostras de cor e luminosidade. Nos vídeos digitais, a representação das cores é comumente realizada a partir de três componentes primárias: vermelho, verde e azul (modelo RGB). Essa escolha baseia-se na fisiologia do sistema visual humano, que apresenta uma resposta tricromática mediada por três tipos de células cone na retina, cada uma com sensibilidade máxima a diferentes faixas do espectro eletromagnético visível (Poynton, 2012). Por meio da combinação aditiva desses três estímulos, é possível reproduzir uma ampla gama de cores percebidas pelo observador humano (Stone, 2016). Como resultado, o modelo RGB tornou-se uma convenção amplamente adotada na codificação e exibição de imagens e vídeos digitais. O sistema utilizado para representar as cores de forma digital é denominado espaço de cores.

Existem diversos espaços de cores, mas os mais proeminentes são o RGB (*Red, Green, Blue*) e o YCbCr (Y-luminância, Cb-crominância azul e Cr-crominância vermelha). O espaço de cor RGB, que representa as cores por meio de três matrizes

separadas, é amplamente utilizado, especialmente em dispositivos como monitores e câmeras. Contudo, o espaço de cor mais utilizado em codificação de vídeo é o YCbCr. Esse espaço se destaca devido à separação clara entre as informações de luminosidade (Y) e as de cor (Cb e Cr), permitindo o tratamento distinto dessas duas componentes.

Essa separação é vantajosa, pois o olho humano possui maior sensibilidade à variação de luminosidade do que à variação de cor (Gonzalez, 2009). Essa característica é explorada nos padrões de compressão de vídeo, que aumentam a eficiência de codificação ao reduzir a quantidade de dados necessários para representar as informações de cor. Isso é possível por meio de um processo denominado sub-amostragem de cores, onde a resolução das amostras de cor pode ser reduzida sem comprometer significativamente a qualidade da imagem.

Os formatos de sub-amostragem mais comuns usados na codificação de vídeo são o 4:4:4, o 4:2:2 e o 4:2:0, os quais são ilustrados na Figura 4. Esses formatos variam na maneira como os componentes de cor (Cb e Cr) são amostrados em relação à componente de luminosidade (Y), sendo que, em formatos com menor taxa de amostragem, há uma redução na quantidade de dados de cor sem perdas perceptíveis na qualidade visual.

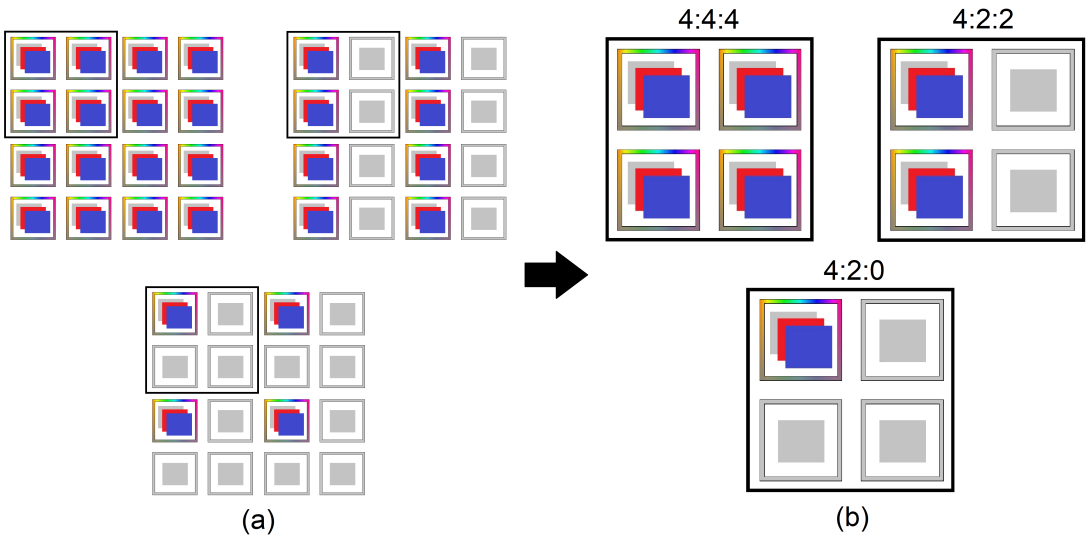

Figura 4 – Formatos de sub-amostragem. (a) Bloco 4×4; (b) 4 amostras do bloco.

No formato de sub-amostragem 4:4:4, não há redução na quantidade de amostras de cor, pois para cada quatro amostras de luminância (Y), existem quatro amostras de crominância azul (Cb) e quatro amostras de crominância vermelha (Cr). Isso implica que a resolução de cor e a resolução de luminosidade são mantidas iguais. Por outro lado, no formato 4:2:2, a sub-amostragem de cor é aplicada de forma que, para cada 4 amostras de Y, existem apenas duas amostras de Cb e duas amostras de Cr. Finalmente, no formato 4:2:0, a redução nas amostras de cor é ainda mais acentuada, com uma amostra de Cb e uma amostra de Cr para cada quatro amostras de Y.

Essa redução nas amostras de cor nos formatos 4:2:2 e 4:2:0 resulta em uma diminuição significativa na quantidade de dados necessários para representar as informações de cor, o que tem um impacto direto na taxa de compressão. Em particular, um codificador que utilize o formato 4:2:2 utilizará apenas metade das amostras de crominância (Cb e Cr) em comparação com o formato 4:4:4, levando a um aumento na taxa de compressão de vídeo. Esse ganho em compressão é alcançado sem causar uma degradação perceptível na qualidade visual da imagem, uma vez que a sensibilidade do olho humano a variações nas componentes de cor é inferior à sua sensibilidade a variações na luminosidade.

Dessa forma, a técnica de sub-amostragem de cores se revela uma abordagem eficiente para a compressão de vídeo, permitindo uma significativa redução no volume de dados necessário para representar a informação sem comprometer a qualidade perceptível para o observador. A eficiência dessa técnica se baseia no fato de que parte das informações de cor pode ser descartada sem afetar substancialmente a experiência visual, o que torna a compressão mais eficaz.

2.1.2 Compressão de Vídeo

Os quadros de um vídeo digital apresentam uma grande semelhança entre si, o que torna desnecessária a representação de muitas das informações de um quadro para o outro. Esse fenômeno é denominado redundância e os dados redundantes não agregam novas informações relevantes para a representação do vídeo. Por exemplo, ao considerar um vídeo de um professor explicando um conteúdo em frente a um quadro negro, e assumindo que a câmera está estática com o movimento restrito ao professor, o fundo (o quadro negro) permanece inalterado em todos os quadros. Essas informações estáticas de um quadro para o outro são consideradas redundantes, pois não oferecem nenhuma nova informação.

A redundância é, portanto, um alvo de otimização nos processos de compressão de vídeo. Os codificadores de vídeo exploram três tipos principais de redundância para reduzir a quantidade de dados a ser representada e armazenada:

- Redundância Espacial: Refere-se à distribuição de *pixels* dentro de um mesmo quadro de vídeo. Também conhecida como redundância intraquadro, ela se manifesta quando *pixels* próximos possuem valores muito semelhantes, permitindo que uma informação seja representada de maneira mais eficiente.

- Redundância Temporal: Está relacionada à similaridade entre quadros consecutivos. Muitas vezes, os blocos de *pixels* em um quadro não mudam significativamente de um quadro para o seguinte. Essa redundância é também chamada de redundância interquadros e pode ser explorada para reduzir a quantidade de dados necessária para representar sequências de imagens em movimento.

- Redundância Entrópica: Este tipo de redundância está associado à distribuição de probabilidades dos símbolos que compõem o vídeo codificado. A entropia é uma medida da quantidade média de informação transmitida por símbolo (Shi; Sun, 1999), e a redundância entrópica surge quando certos símbolos ou padrões aparecem com maior frequência, permitindo uma codificação mais eficiente.

Com o objetivo de reduzir a quantidade de informação armazenada ou transmitida, são empregados diversos algoritmos de compressão de vídeo, que podem ser classificados em duas categorias principais:

- Compressão sem perdas (*lossless*): Esta categoria de técnicas reduz a quantidade de dados sem eliminar qualquer informação original, permitindo a reconstrução exata do vídeo. A compressão sem perdas é ideal para situações em que a preservação completa da qualidade é crucial.

- Compressão com perdas (*lossy*): Utiliza técnicas de remoção de informações redundantes e irrelevantes para reduzir significativamente o tamanho do arquivo. Embora essa abordagem possa resultar em uma degradação perceptível da qualidade da imagem, ela é frequentemente empregada em cenários onde a redução do tamanho do arquivo é prioritária, como em transmissões de vídeo e armazenamento em mídias de capacidade limitada.

Diante do exposto, observa-se que a compressão de vídeo desempenha um papel crucial em canais de comunicação que lidam com volumes elevados de dados, como ocorre em serviços de *streaming*. Esse processo é essencial para reduzir a quantidade de dados necessária para representar um vídeo digital, aproveitando características intrínsecas aos próprios vídeos. Uma dessas características, como mencionado anteriormente, é a redundância, que pode ser explorada para diminuir o tamanho do arquivo sem comprometer a qualidade perceptível da imagem.

Os codificadores de vídeo disponíveis atualmente precisam seguir padrões específicos durante as etapas de codificação para garantir a interoperabilidade entre dispositivos que processam vídeos digitais. Esses padrões são fundamentais para que diferentes sistemas possam se comunicar e compartilhar vídeos de maneira eficiente e compatível. Ao longo dos anos, diversos padrões de codificação foram desenvolvidos, com o objetivo de melhorar a qualidade, a eficiência de compressão e a compatibilidade entre os dispositivos.

A Figura 5 ilustra as principais etapas de um codificador de vídeo genérico, destacando os processos envolvidos na compressão de vídeos digitais. Essas etapas incluem predição, transformadas/quantização, codificação de entropia e filtros, com a utilização de técnicas que exploram redundâncias espaciais, temporais e entrópicas, conforme discutido anteriormente.

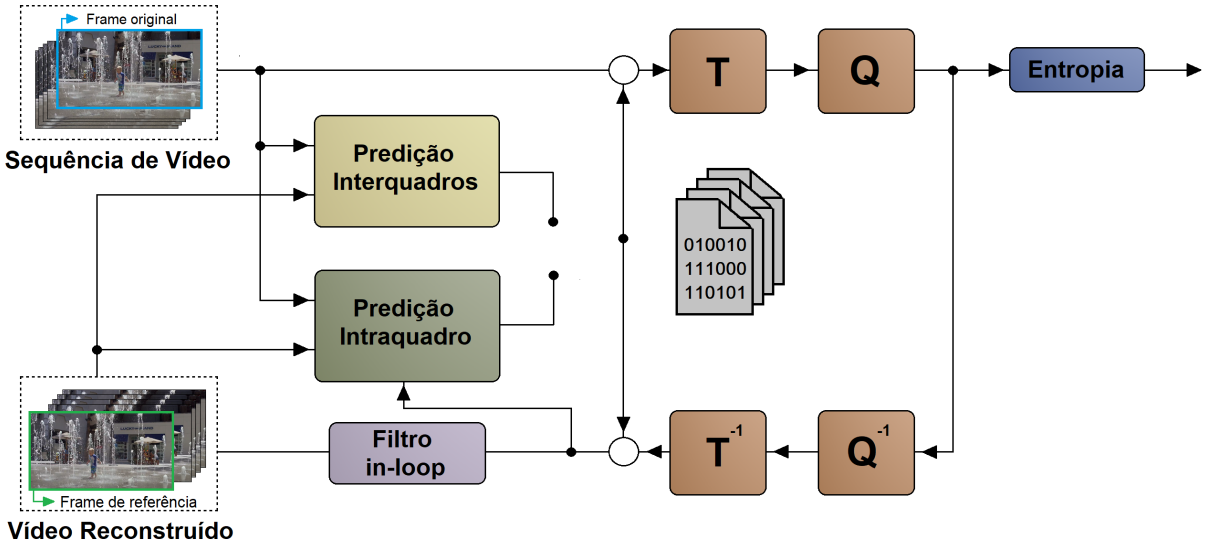

Figura 5 – Modelo genérico de codificador de vídeo híbrido. (Palau; Cunha silveira; Domanski; Loose; Cerveira; Sampaio; Palomino; Porto; Corrêa; Agostini, 2021).

Como ilustrado na Figura 5, os principais módulos que compõem um codificador de vídeo incluem predições intraquadro e interquadros, transformadas direta e inversa, quantizações direta e inversa, filtros e codificação de entropia. O processo de codificação de um vídeo digital inicia-se com o particionamento dos quadros (*frames*) em blocos menores, uma estratégia que visa otimizar a eficiência do processo de compressão. O particionamento em blocos facilita a análise e a codificação, permitindo um tratamento mais eficaz das redundâncias espaciais e temporais presentes no vídeo.

O tamanho desses blocos pode variar conforme o padrão de codificação adotado. Padrões de compressão mais avançados podem permitir blocos maiores, o que se revela vantajoso em vídeos com resoluções mais altas. Blocos menores possibilitam uma maior precisão na codificação de detalhes finos, melhorando a eficiência de compressão e a qualidade visual dos vídeos em resoluções superiores. Assim, a escolha do tamanho de bloco adequado depende diretamente das características do vídeo a ser comprimido, como sua resolução e o tipo de conteúdo.

A **predição intraquadro** tem como objetivo a redução da redundância espacial dentro de um quadro de vídeo. Nesta etapa, é realizada a predição de blocos candidatos, que são gerados a partir de blocos vizinhos da imagem atual. O processo de predição intraquadro utiliza modos de predição que indicam a direção das amostras com maior similaridade em relação às amostras do bloco que está sendo codificado. Esses modos são fundamentais para otimizar a compressão, pois permitem que os blocos em questão sejam representados de maneira mais eficiente, aproveitando as informações já presentes nas regiões vizinhas do quadro. Na Figura 6, são apresentados exemplos desses modos de predição intraquadro, que demonstram diferentes abordagens para estimar a direção de similaridade entre os blocos, com o intuito de

reduzir a quantidade de dados necessários para representar a informação visual no vídeo. A escolha adequada do modo de predição pode impactar diretamente a eficiência da compressão, especialmente em vídeos com características espaciais complexas.

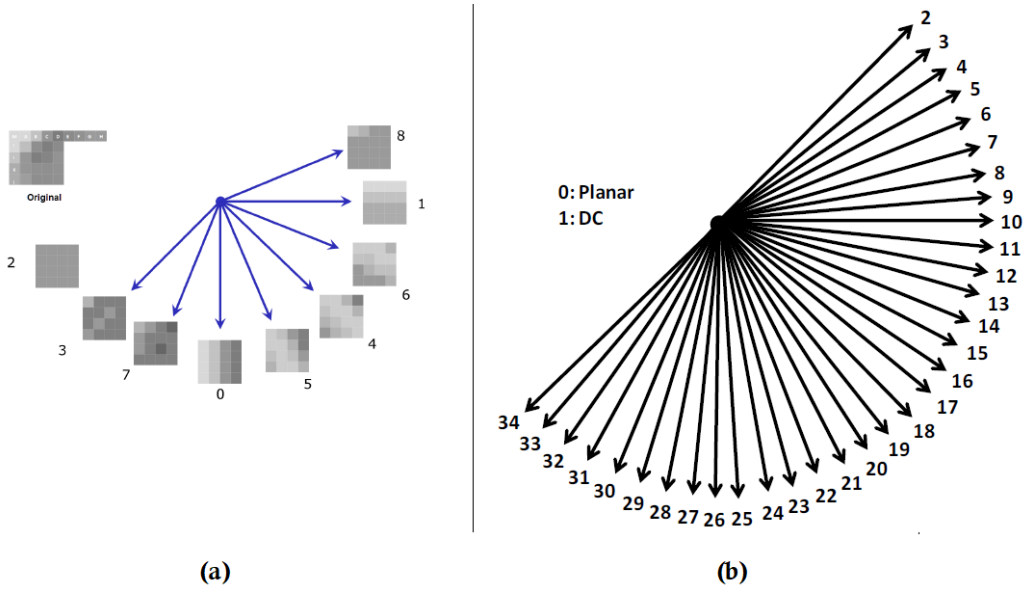

Figura 6 – Modos de predição intraquadro: (a) H.264.AVC (Seidel et al., 2014) (b) HEVC (Corrêa, 2014).

O módulo de **predição interquadros** está relacionado à exploração das redundâncias temporais entre quadros vizinhos de um vídeo digital. Para gerar essa predição, são utilizadas duas ferramentas principais: a Estimação de Movimento (*Motion Estimation* - ME) e a Compensação de Movimento (*Motion Compensation* - MC). A ME realiza a comparação entre um bloco do quadro atual e blocos dos quadros anteriores ou posteriores, com o objetivo de identificar a melhor similaridade entre eles. Quando o bloco mais similar é encontrado, é gerado um Vetor de Movimento (*Motion Vector* - MV), que indica o deslocamento entre a posição do bloco atual e o bloco correspondente no quadro de referência. Este MV é então utilizado pela MC, que reconstrói o quadro previsto copiando os blocos do quadro de referência para um *buffer* do quadro reconstruído. Esse processo é fundamental para a redução de dados redundantes entre os quadros e para aumentar a eficiência da compressão. A Figura 7 ilustra um exemplo desse processo de predição interquadros, demonstrando como a Estimação e a Compensação de Movimento trabalham em conjunto para prever e reconstruir os quadros subsequentes.

Após a execução das predições intraquadro e interquadros, os blocos resultantes são subtraídos do bloco original, gerando um bloco de resíduo. Esse bloco de resíduo contém as diferenças entre as amostras originais e as amostras preditas e será utilizado nas etapas subsequentes da compressão. O resíduo é uma representação compacta das discrepâncias entre a imagem original e a sua predição, permitindo uma codificação mais eficiente. A Figura 8(a) apresenta um exemplo de um quadro de refe-

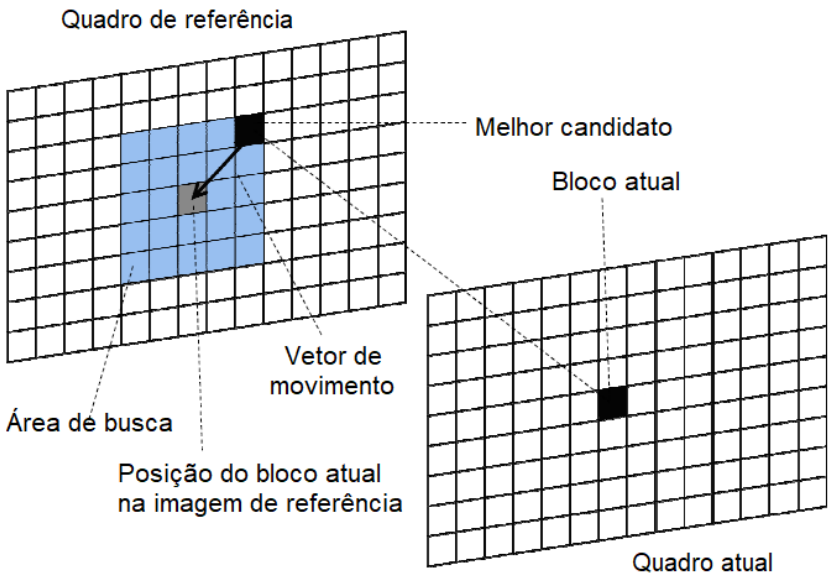

Figura 7 – Predição interquadros (Diniz, 2015).

rência, enquanto a Figura 8(b) ilustra um quadro predito. Por fim, a Figura 8(c) mostra o quadro de resíduo, que contém as variações a serem codificadas de forma mais eficiente nas etapas seguintes.

O módulo das **transformadas** envolve a conversão de blocos do domínio espacial para o domínio das frequências. Nesta etapa da codificação de vídeo, o que é transformado não são os blocos da imagem original, mas sim os blocos de resíduos (como ilustrado na Figura 8(c)). Este processo é fundamental, pois explora a característica do sistema visual humano de ter maior sensibilidade às baixas frequências em comparação às altas frequências. No domínio das frequências, a informação essencial de uma imagem de resíduos tende a se concentrar em um número reduzido de coeficientes, o que possibilita uma representação mais compacta e eficiente desses dados (Bross; Chen; Ohm; Sullivan; Wang, 2021).

Como a etapa de transformadas é um dos focos principais desta tese, ela será abordada com mais detalhes nas seções subsequentes.

A **quantização** é uma etapa crucial que ocorre logo após a etapa de transformadas no processo de codificação de vídeo. Durante essa fase, a eliminação de dados começa a ser efetivamente aplicada. Os coeficientes gerados na etapa de transformadas são, de maneira geral, submetidos a um processo de redução que envolve a divisão e o arredondamento dos valores, de forma a aproximar os coeficientes das altas frequências a zero. Essa aproximação resulta na perda de detalhes das altas frequências, enquanto mantém os componentes de baixa frequência, que são mais relevantes para a percepção visual humana. A quantização pode ser compreendida como um processo de mapeamento dos coeficientes transformados para um conjunto de valores discretos. Esse mapeamento é controlado por parâmetros de escala que

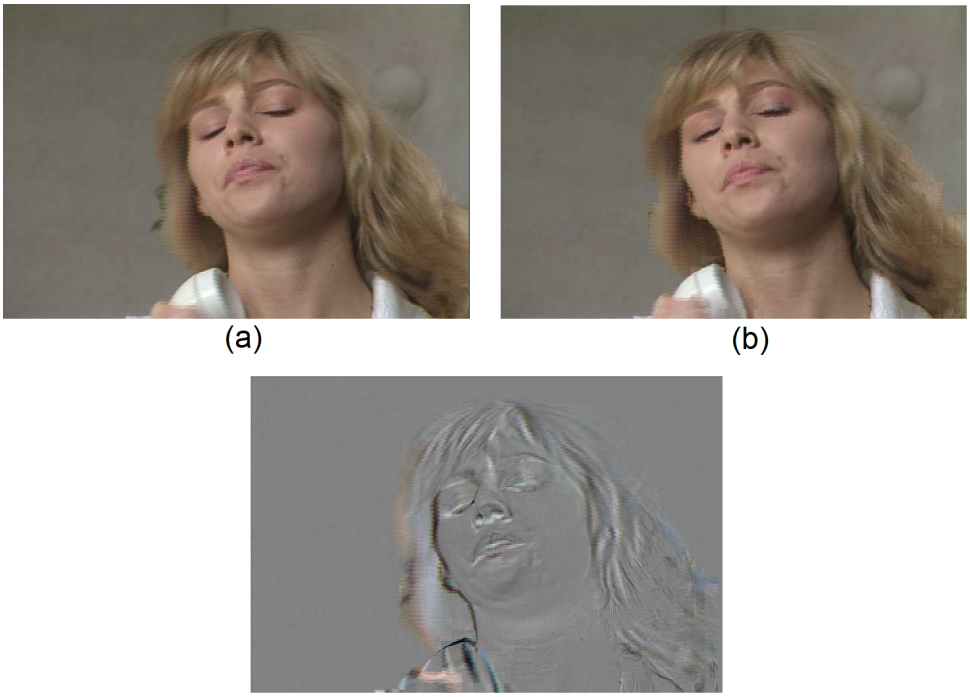

Figura 8 – (a) Quadro de referência; (b) Quadro predito; (c) Quadro de resíduo.

determinam a precisão com a qual os coeficientes serão representados. A aplicação desses parâmetros permite o ajuste da taxa de compressão, permitindo que o codificador atinja uma taxa de bits desejada para a sequência de vídeo em questão. É importante destacar que a quantização é uma das etapas responsáveis pela introdução da perda de qualidade no processo de compressão com perdas. Isso ocorre porque, ao aproximar os coeficientes de alta frequência a zero, detalhes sutis e informações de menor importância são descartados, resultando em uma diminuição na qualidade da imagem. No entanto, esse processo é essencial para a redução do tamanho do arquivo de vídeo, sendo um compromisso entre a preservação da qualidade visual e a eficiência na compressão. A Figura 9 ilustra as matrizes geradas durante os processos de transformadas e quantização de um bloco de vídeo. Essas matrizes demonstram a distribuição dos coeficientes transformados e a sua subsequente quantização, evidenciando a eliminação das altas frequências em benefício da eficiência de compressão.

A **codificação de entropia** é uma etapa do processo de compressão que ocorre após a quantização dos coeficientes. Nessa fase, os coeficientes quantizados e demais informações geradas pelo processo de codificação (vetores de movimento, modos de predição) são codificados para permitir a transmissão ou o armazenamento eficiente do vídeo. O principal objetivo da codificação de entropia é representar os dados de maneira compacta, utilizando algoritmos que aplicam um comprimento variável aos códigos, de forma a otimizar o número de bits para representar os dados provenientes dos demais módulos do codificador de vídeo (Palau; Cunha silveira; Domanski;

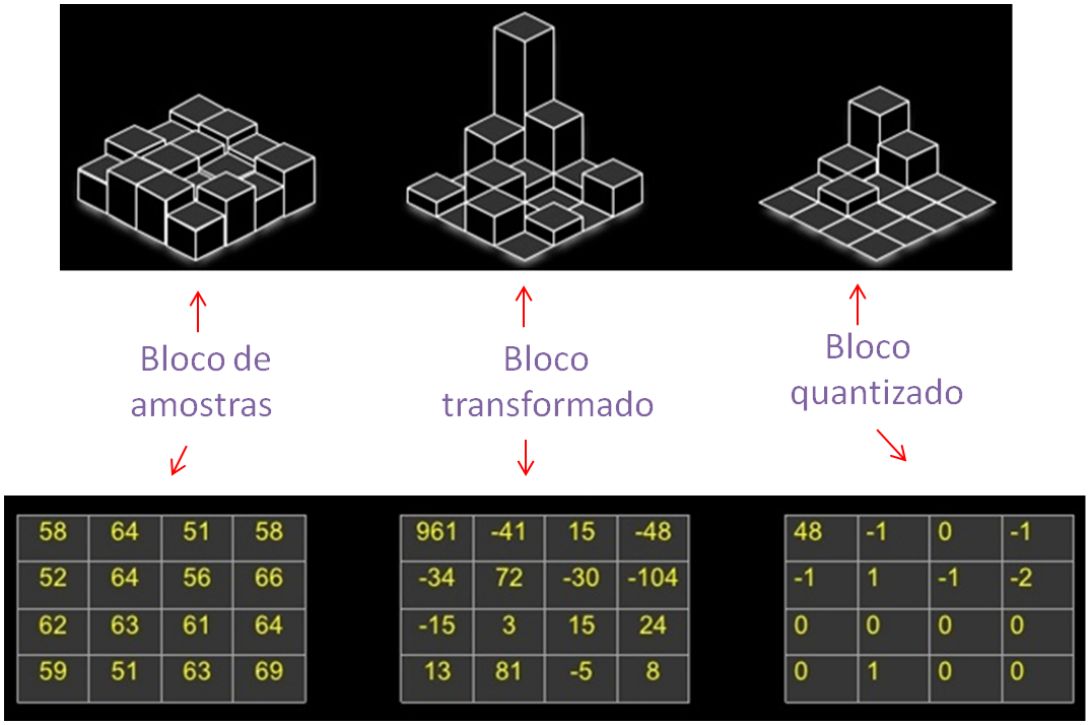

Figura 9 – Transformada e Quantização.

Loose; Cerveira; Sampaio; Palomino; Porto; Corrêa; Agostini, 2021). Essa codificação sem perdas assegura que a informação original possa ser recuperada sem qualquer degradação, embora os dados sejam comprimidos. Os algoritmos de codificação de entropia exploram a distribuição e a probabilidade de ocorrência dos coeficientes quantizados e das demais informações geradas pelo processo de codificação. Como muitas vezes esses coeficientes gerados durante a quantização são esparsos (isto é, contêm um grande número de valores nulos ou próximos de zero), a codificação de entropia é particularmente eficaz nesse contexto.

A última etapa da codificação de vídeo digital envolve a aplicação de **filtros**, que têm como objetivo suavizar os efeitos indesejados da quantização, como artefatos visíveis, distorções e blocos. Esses filtros são aplicados aos quadros antes que sejam utilizados novamente como quadros de referência na predições interquadros subsequentes. O processo de filtragem ajuda a melhorar a qualidade visual do vídeo reconstruído, preservando características visuais importantes e minimizando ruídos artificialmente gerados pela quantização. Filtros espaciais ou adaptativos são comumente empregados para ajustar as transições entre blocos de *pixels*, garantindo uma compressão eficiente e mantendo a integridade visual dos quadros.

2.2 *Versatile Video Coding*

Como discutido na introdução desta tese, o *Versatile Video Coding* é um dos mais recentes e avançados padrões de codificação de vídeo. O desenvolvimento do VVC

ocorreu em duas fases distintas: a fase de exploração, que se estendeu de 2015 a 2017, e a fase de padronização, realizada entre 2018 e 2020. Este padrão foi criado por um comitê denominado *Joint Video Experts Team* (JVET), composto pelos grupos *ITU-T Video Coding Expert Group* (VCEG) e *ISO Motion Picture Expert Group* (MPEG). O objetivo principal do JVET foi superar, de maneira significativa, o desempenho e os recursos de compressão do padrão anterior, *High Efficiency Video Coding* (HEVC), com a intenção de proporcionar uma alternativa que pudesse, eventualmente, substituir o HEVC de maneira universal (Wien; Baroncini; Boyce; Segall; Suzuki, 2017).

O VVC foi projetado para lidar com uma gama mais ampla de requisitos de compressão, com o foco na eficiência em resoluções de vídeo mais altas, como 4K, 8K, e além, e na melhoria da codificação em ambientes com restrições de largura de banda, como serviços de *streaming* e plataformas de comunicação em tempo real. Sua arquitetura e ferramentas inovadoras permitem um desempenho superior na compressão de vídeo, mantendo a qualidade da imagem mesmo em taxas de bits significativamente mais baixas.

O processo de codificação de vídeo para o novo padrão VVC segue o modelo geral do codificador genérico descrito na seção 2.1.1. Contudo, em comparação com seu predecessor, o padrão HEVC, o VVC incorpora diversas ferramentas novas e aprimoradas em cada etapa da codificação, visando otimizar a eficiência e a qualidade da compressão. Algumas das inovações mais notáveis no VVC são as seguintes:

- **Particionamento das *Coding Tree Units* (CTU)**: A CTU é a unidade básica de particionamento em codificadores de vídeo modernos. CTUs representam a divisão de um quadro em blocos de diferentes dimensões, permitindo que o vídeo seja codificado de maneira eficiente. Cada CTU pode ser subdividida em blocos menores, denominados *Coding Units* (CUs)). No VVC, o tamanho das CTUs foi expandido para até 128×128 *pixels*, permitindo divisões binárias, ternárias e quaternárias (CUs de tamanhos 4×4 até 128×128), o que oferece maior flexibilidade na adaptação ao conteúdo visual.

- **Número de ângulos na predição intraquadro**: Os ângulos de predição intraquadro são direções usadas para prever os valores dos *pixels* em um bloco, com base nos *pixels* vizinhos. A ideia é aproveitar a redundância espacial de um bloco, prevendo os valores de *pixels* a partir de seus vizinhos. O número de ângulos de predição define as direções possíveis para essa previsão. Quanto maior o número de ângulos, mais flexível e precisa se torna a predição, permitindo que o codificador se adapte melhor às diferentes texturas e padrões presentes no quadro. No VVC, o número de ângulos de predição intraquadro foi ampliado para 67, em comparação aos padrões anteriores, proporcionando uma codifica-

ção mais precisa das variabilidades espaciais nos quadros.

- **Número de quadros de referência para predição interquadros**: Os quadros de referência são quadros anteriores ou posteriores em um vídeo usados para prever os valores dos *pixels* em um quadro atual. Durante a predição interquadros, o codificador compara um quadro com os quadros de referência e usa essa comparação para calcular os movimentos dos blocos de *pixels* entre os quadros. O objetivo é reduzir a redundância temporal, codificando apenas as diferenças entre os quadros, em vez de codificar cada quadro de forma independente. No VVC, o número de quadros de referência foi aumentado para 16, permitindo uma predição mais precisa e eficiente.

- **Tipos de transformadas**: No novo padrão de codificação, foram introduzidas transformadas adicionais, como a DCT-VIII (*Discrete Cosine Transform* de ordem 8) e a DST-VII (*Discrete Sine Transform* de ordem 7), visando melhorar a eficiência de compressão. Além disso, o tamanho dos blocos utilizados no processo de transformada foi ampliado para até 64×64 *pixels*, incluindo também blocos de tamanho retangular. Essas mudanças proporcionam maior flexibilidade e eficiência durante o processo das transformadas, permitindo uma adaptação mais precisa às características das imagens e vídeos a serem codificados.

- **Parâmetro de quantização**: Os parâmetros de quantização são valores utilizados para controlar a redução de precisão dos coeficientes após as transformadas. Eles determinam o nível de compressão, ajustando a granularidade da quantização dos coeficientes transformados. Quanto maior o valor do parâmetro de quantização, maior a perda de informação e maior a compressão, mas isso pode resultar em uma redução na qualidade do vídeo. O objetivo é equilibrar a taxa de compressão com a preservação da qualidade visual. No VVC, estes parâmetros de quantização foram expandidos para 63 níveis, permitindo um controle mais preciso da redução de dados durante a codificação.

- **Incorporação de Filtros**: O padrão VVC introduziu quatro tipos distintos de filtros, aplicados para suavizar artefatos visuais resultantes das etapas anteriores de codificação, como a quantização. Esses filtros desempenham um papel crucial na melhoria da qualidade perceptual do vídeo comprimido, proporcionando uma experiência visual superior ao reduzir distorções e defeitos causados durante o processo de compressão.

Assim, essas inovações permitem ao VVC oferecer melhorias significativas em relação ao HEVC, tanto em termos de eficiência de compressão quanto em capacidade de adaptação a diferentes tipos de conteúdo e requisitos de codificação.

2.3 Transformadas do VVC

A *Multiple Transform Selection* representa uma das inovações mais significativas do padrão de codificação de vídeo VVC. Ao contrário do seu antecessor, que utilizava exclusivamente a DCT-II, o VVC introduz a utilização de múltiplos tipos de transformadas para a codificação residual dos blocos, tanto para os quadros do tipo I (intra) quanto para quadros do tipo P e B (que admitem predições intra e interquadros). Além da DCT-II, o VVC incorpora duas novas transformadas: a DCT-VIII e a DST-VII. Adicionalmente, uma quarta transformada, denominada *Low-Frequency Non-Separable Transform* (LFNST), foi introduzida, sendo aplicada antes da etapa de quantização.

O codificador VVC utiliza as transformadas discretas do cosseno e do seno, tanto em sua versão direta quanto inversa, no contexto de uma codificação e decodificação eficientes de vídeos. Estas transformadas desempenham um papel crucial no processo de compressão e reconstrução de dados de vídeo. A principal distinção matemática entre a transformada direta e inversa consiste na direção do processamento dos dados. Enquanto a transformada direta converte os valores dos *pixels* de uma imagem ou quadro para o domínio da frequência, a transformada inversa é responsável por reverter esse processo, convertendo os coeficientes de volta ao domínio espacial. Esse processo permite a reconstrução da imagem ou vídeo original a partir dos coeficientes previamente codificados.

Durante o processo de codificação no VVC, as transformadas diretas são aplicadas aos blocos de resíduos de predição para converter a informação do domínio espacial para o domínio da frequência. Essa transformação visa otimizar a compressão ao identificar as frequências dominantes nos dados, permitindo a eliminação ou quantização mais eficiente das frequências menos significativas ao sistema visual humano. A escolha da melhor transformada para cada bloco, portanto, envolve um processo de testes exaustivos, uma vez que o codificador deve determinar qual transformada minimiza a redundância e maximiza a eficiência de compressão para as características específicas dos dados de entrada.

Por outro lado, no processo de decodificação, a transformada inversa é utilizada para reverter os coeficientes quantizados ao domínio espacial, recuperando a imagem ou o vídeo de forma aproximada. No entanto, como a transformada a ser aplicada já foi definida no codificador, o processo de decodificação é substancialmente mais simples. Não é necessário realizar uma nova análise ou seleção de transformada, assim tornando o processo de reconstrução da imagem ou vídeo mais eficiente, com menor custo computacional. A decodificação, portanto, pode ser otimizada para garantir a reconstrução eficiente dos dados sem a necessidade de cálculos tão intensivos quanto os realizados durante a codificação.

Portanto, as etapas de aplicação das transformadas no processo de codificação do

VVC se diferenciam principalmente pelo esforço computacional envolvido. A transformada direta, sendo uma das etapas mais complexas da codificação, requer um processo de avaliação contínua para determinar a melhor transformada para cada bloco de resíduos. Em contraste, a transformada inversa no decodificador é uma etapa mais simplificada, uma vez que a escolha da transformada já foi realizada no codificador. Esse contraste entre a complexidade da codificação e a simplicidade da decodificação reflete as demandas de processamento nos dois lados do sistema de codificação e decodificação.

A complexidade da codificação, particularmente na aplicação das transformadas diretas, está intimamente ligada à flexibilidade na escolha do tamanho e formato dos blocos residuais. Como mencionado, a partição das CTUs em tamanhos menores de blocos permite divisões binárias, ternárias e quaternárias, gerando blocos que não são restritos a formas quadradas, mas também retangulares. Essa flexibilidade no particionamento dos blocos é crucial para otimizar a eficiência da compressão, pois diferentes tamanhos de blocos podem ser mais adequados para diferentes tipos de conteúdo. A Figura 10 ilustra um exemplo de uma divisão de um bloco 64x64 em tamanhos menores, tanto quadrados quanto retangulares.

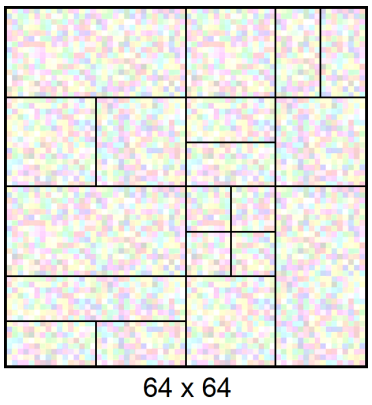

Figura 10 – Exemplo de partições de um bloco 64×64.

A ferramenta MTS introduzida no padrão VVC permite a utilização e combinação dos três tipos de transformadas em uma dimensão (1D), resultando em transformadas 2D mescladas. Essas transformadas podem ser aplicadas de maneira separável, o que significa que as direções vertical e horizontal da matriz são processadas de forma independente. Essa abordagem possibilita a combinação dos diferentes tipos de transformadas, proporcionando maior flexibilidade no processo de codificação. A Figura 11 exemplifica essa característica.

2.3.1 Transformada Discreta do Cosseno

A Transformada Discreta do Cosseno (DCT) é uma das ferramentas mais utilizadas na área de processamento de imagem devido à sua principal característica de

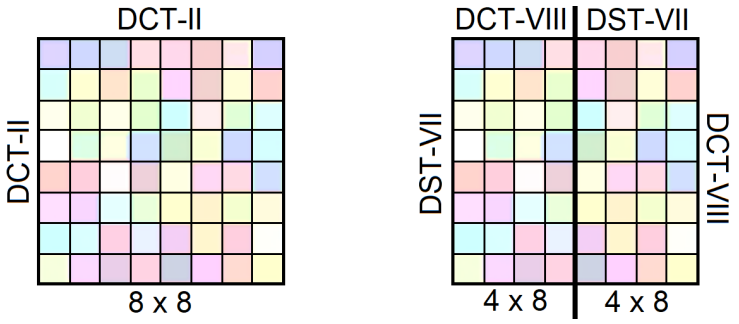

Figura 11 – Exemplo de combinações de transformadas na MTS.

concentrar a maior parte da informação do sinal em componentes de baixas frequências. Na codificação de vídeo, isso permite descartar pequenas componentes de alta frequência na etapa de quantização, levando a significativas taxas de compressão. A DCT transforma as amostras de uma imagem do domínio espacial para o domínio das frequências, isto é, ela expressa uma sequência finita de pontos de dados em termos de uma soma de funções cosseno que oscilam em diferentes frequências.

Existem 8 tipos diferentes de transformadas discretas do cosseno, porém a mais utilizada na codificação de vídeo é a DCT-II. Na equação 1 é apresentada a função básica desta transformada, onde *N* é o tamanho do bloco da transformada, *i* = 0, 1,..., *N*-1 é o índice de saída, e *j* = 0, 1,..., *N*-1 é o índice do vetor de entrada. O valor de ω_0 é definido em (2).

$$T_i(j) = \omega_0 \cdot \sqrt{\frac{2}{N}} \cdot \cos\left(\frac{\Pi \cdot i \cdot (2j+1)}{2N}\right) \tag{1}$$

$$\omega_0 = \begin{cases} \sqrt{\frac{2}{N}} & i = 0 \\ 1 & i \neq 0 \end{cases} \tag{2}$$

Como dito anteriormente, além da DCT-II, mais duas transformadas foram acrescentadas ao codificador VVC, sendo uma delas a DCT-VIII. A equação 3 apresenta a função básica desta transformada, onde *N* é o tamanho do bloco da transformada, *i* = 0, 1,..., *N*-1 é o índice de saída, e *j* = 0, 1,..., *N*-1 é o índice do vetor de entrada.

$$T_i(j) = \sqrt{\frac{4}{2N+1}} \cdot \cos\left(\frac{\Pi \cdot (2i+1) \cdot (2j+1)}{4N+2}\right) \tag{3}$$

2.3.2 Transformada Discreta do Seno

A Transformada Discreta do Seno (DST) permite representar um sinal digital em termos de uma soma de sinusoides considerando diferentes frequências e amplitudes. Assim como a DCT, a DST tem algumas características desejáveis para o processamento de imagens, pois possui grande capacidade de compactação de energia e coeficientes que são reais, simétricos e ortogonais.

Uma das inovações do padrão VVC foi a inclusão deste tipo de transformada na codificação de vídeo, mais precisamente a DST tipo VII. A função básica desta transformada pode ser vista na equação 4. *N* é o tamanho do bloco da transformada, *i* = 0, 1,..., *N*-1 é o índice de saída, e *j* = 0, 1,..., *N*-1 é o índice do vetor de entrada.

$$T_i(j) = \sqrt{\frac{4}{2N+1}} \cdot \sin\left(\frac{\Pi \cdot (2i+1) \cdot (j+1)}{2N+1}\right) \tag{4}$$

2.3.3 *Multiple Transform Selection* – MTS

Como mencionado anteriormente, a técnica de MTS é um método avançado de codificação que permite a aplicação de diferentes tipos de transformadas de forma independente nas dimensões horizontal e vertical. O processo inicia-se com a aplicação de uma transformada unidimensional (1D) na direção horizontal sobre um bloco residual de tamanho $N \times M$, gerando um bloco intermediário de mesma dimensão. Em seguida, uma nova transformada 1D é aplicada na direção vertical sobre esse bloco intermediário, resultando no bloco final transformado. O diferencial da MTS em relação às transformadas tradicionais está na flexibilidade na escolha dos tipos de transformadas para cada direção, permitindo que o codificador selecione de forma adaptativa a melhor combinação para cada bloco. Essa seleção pode otimizar a compactação da informação ao considerar a distribuição de energia do sinal em diferentes orientações, melhorando a eficiência da codificação.

No VTM, a técnica de MTS apresenta duas variantes: MTS explícito e MTS implícito (Hamidouche; Biatek; Abdoli; François; Pescador; Radosavljević; Menard; Raulet, 2022). No modo explícito, diferentes transformadas (DST-VII ou DCT-VIII) são testadas para cada bloco, o custo de codificação é calculado para cada combinação e, em seguida, a melhor opção é escolhida e sinalizada no bitstream. Essa abordagem permite seu uso tanto para blocos codificados por predição intraquadro quanto interquadros. Por outro lado, no modo implícito, a transformada não é selecionada por meio de testes; em vez disso, é escolhida com base em regras fixas, determinadas pela estrutura do bloco e pelos modos de predição intraquadro. Como a escolha depende apenas de informações já disponíveis no processo de codificação, não há necessidade de sinalização no bitstream. As combinações possíveis entre os modos explícito e implícito, bem como suas respectivas sinalizações, são apresentadas na Tabela 1.

Com o objetivo de garantir maior clareza e uniformidade na terminologia empregada ao longo da tese, a *flag* responsável por indicar o modo explícito ou implícito de seleção das transformadas será referida, a partir deste ponto, como *MTS-MODO*. A padronização dessa nomenclatura é essencial para evitar ambiguidade nas referências futuras e facilitar a compreensão do leitor durante a descrição das análises e discussões subsequentes.

No VVC, o controle da MTS explícito é realizado por meio de *flags* de habilitação

Tabela 1 – Aplicação dos modos explícito e implícito da MTS

MTS-MODO	Intra	Inter
1	Explícito	-
2	Implícito	Explícito
3	Explícito	Explícito
4	Implícito	-

definidas no *Sequence Parameter Set* (SPS), um conjunto de parâmetros da sequência de vídeo responsável por configurar diversos aspectos da codificação (Fraunhofer, 2020). Essas *flags* determinam se a técnica MTS pode ser aplicada e, caso ativada, possibilitam a escolha entre diferentes tipos de transformadas ao longo das dimensões horizontal e vertical. A habilitação dessas *flags* segue as especificações do VTM, onde o comportamento da MTS explícita é ajustado conforme apresentado na Tabela 2.

Tabela 2 – Combinações de transformadas permitidas na MTS

MTS	Hor_*flag*	Ver_*flag*	1a Transformada	2a Transformada
Desabilitada	-	-	DCT-II	DCT-II
Habilitada	0	0	DST-VII	DST-VII
Habilitada	0	1	DCT-VIII	DST-VII
Habilitada	1	0	DST-VII	DCT-VIII
Habilitada	1	1	DCT-VIII	DCT-VIII

A Tabela 2 apresenta a configuração das transformadas aplicadas em cada dimensão de acordo com a ativação do MTS. Quando o MTS está desativado, a única transformada empregada em ambas as direções (horizontal e vertical) é a DCT-II, garantindo um processamento padronizado sem variação na escolha das transformadas. Por outro lado, ao ativar o MTS, seja no modo explícito ou implícito, duas *flags* adicionais, Hor_*flag* e Ver_*flag*, são introduzidas no processo de codificação. Essas *flags* permitem selecionar de forma independente a transformada a ser aplicada em cada dimensão, possibilitando a utilização de alternativas como a DST-VII e a DCT-VIII.

O processo de seleção das transformadas no VVC não depende apenas da *flag* MTS, mas também de outros parâmetros, como a *flag* ISP (*Intra Sub-Partition*). Quando essa *flag* está ativada, o particionamento dos blocos ocorre em sub-blocos retangulares, o que leva à desativação automática do MTS. Nessa condição, o conjunto de transformadas disponíveis fica restrito a determinadas combinações da DCT-II e DST-VII, caracterizando o que é conhecido como MTS implícita. A Tabela 3 detalha essa configuração, evidenciando que, além das combinações convencionais listadas na Tabela 2, a MTS implícita também permite o uso das transformadas híbridas DCT-

II_DST-VII e DST-VII_DCT-II. A seleção dessas transformadas ocorre de acordo com o modo de predição intraquadro adotado pelo codificador, considerando as variações planar, angular e DC.

Tabela 3 – Combinações de transformadas com ISP habilitado

ISP	Tamanho do Bloco	Modo Intra	Vertical	Horizontal
1	largura=1 altura=1		Nenhuma	
	largura e altura = 2 largura e haltura >32		DCT-II	
	4×4 até 32×32	Planar Ang(31,32,34,36,37)	DST-VII	DST-VII
		DC Ang(33,35)	DCT-II	DCT-II
		Ang(2,4,6,...28,30) (39,41,43,...63,65)	DST-VII	DCT-II
		Ang(3,5,7,...27,29) (38,40,42,...64,66)	DCT-II	DST-VII

Diante do exposto, este processo de seleção de transformadas no VVC, especialmente considerando a variedade de tipos de transformadas e suas combinações, bem como o aumento no tamanho dos blocos, resulta em um elevado número de operações durante a codificação. Com o objetivo de atingir a melhor eficiência de compressão, o codificador deve testar as diversas alternativas, o que pode resultar em um alto custo computacional, especialmente quando se trata de vídeos com maior resolução e complexidade de conteúdo. A necessidade de avaliar todas as possíveis combinações de transformadas para determinar a mais eficiente para cada bloco residual torna a etapa de transformadas uma das mais exigentes em termos de recursos computacionais.

2.4 Visão Geral de Aprendizado de Máquina

O aprendizado de máquina é considerado uma área da inteligência artificial baseada no estudo de algoritmos e modelos estatísticos que permitem desenvolver sistemas computacionais capazes de aprender e executar uma tarefa específica sem serem explicitamente programados. Eles fazem previsões sobre os dados a partir de diferentes abordagens de aprendizado, facilitando desta maneira a geração de resultados confiáveis e repetíveis (Mahesh, 2020).

O rápido aumento na quantidade de dados gerada e armazenada, assim como a demanda de serviços de vídeos e o crescimento de aplicações que usam esse tipo de tecnologia, vem sendo desafiador para a área multimídia. Nesse sentido, os algoritmos de aprendizado de máquina têm se tornado um componente-chave das soluções de digitalização que têm atraído grande atenção na área digital (Ray, 2019) como uma alternativa que permite fornecer previsões baseadas em dados, facilitando a otimi-

zação da codificação de vídeo e oferecendo novas oportunidades para atualizar as tecnologias de codificação (Zhang; Kwong; Wang, 2020).

Algumas das aplicações mais recentes de aprendizado de máquina em visão computacional incluem identificação de objetos, classificação de objetos e extração de informações utilizáveis de imagens, documentos gráficos e vídeos (Zhang; Kwong; Wang, 2020).

É importante salientar que não existe uma única metodologia que possa ser usada nas diferentes áreas onde o aprendizado de máquina é aplicado. O tipo de algoritmo a ser utilizado vai depender do tipo de problema que se deseja resolver, do número de variáveis que serão incluídas e do tipo de modelo que melhor se adapta ao problema (Mahesh, 2020).

No contexto da codificação de vídeo, o aprendizado de máquina tem sido utilizado para auxiliar na tomada de decisões complexas realizadas pelo codificador, com o objetivo de equilibrar a eficiência de codificação e o custo computacional. Essas técnicas permitem, por exemplo, selecionar de forma inteligente a melhor ferramenta de predição ou o modo mais adequado de codificação para cada bloco de vídeo. Além disso, modelos de aprendizado também vêm sendo aplicados na melhoria da qualidade visual por meio de filtros de pós-processamento e no desenvolvimento de novas abordagens de predição de quadros, blocos e *pixels*.

Neste trabalho, o foco está na aplicação de técnicas de aprendizado de máquina para otimizar a seleção da transformada no módulo de transformadas, buscando reduzir o consumo energético e o custo computacional, ainda que com possível impacto na qualidade da imagem reconstruída, priorizando a viabilidade da implementação em *hardware*.

Existem vários métodos ou estilos de aprendizado de máquina. Os mais conhecidos são os estilos de aprendizado primário que incluem o aprendizado supervisionado, não supervisionado e por reforço (Mahadevkar; Khemani; Patil; Kotecha; Vora; Abraham; Gabralla, 2022; Bonaccorso, 2017).

2.4.1 Aprendizado Supervisionado

O aprendizado supervisionado é aplicado principalmente quando se pretende prever uma variável que depende de uma lista de variáveis independentes, considerando que tanto os dados de entrada quanto os de saída são fornecidos durante o treinamento do modelo. Ou seja, este aprendizado é caracterizado por inferir uma função ou identificar padrões a partir de dados de treinamento rotulados baseados em um conjunto de exemplos. Os algoritmos utilizados neste tipo de aprendizado são treinados para prever valores ou classificar dados, de modo que o modelo se ajusta aos valores fornecidos por meio de uma validação, com o intuito de atingir os melhores resultados. Este tipo de aprendizado é classificado em duas categorias: classificação e

regressão. Os algoritmos usados nos problemas de classificação distribuem os dados em categorias de acordo com padrões observados, enquanto os algoritmos de regressão são usados para entender a associação existente entre variáveis independentes e uma variável dependente definida.

Alguns dos algoritmos de aprendizado supervisionado mais comuns incluem:

- **Árvores de Decisão:** As árvores de decisão são consideradas algoritmos de aprendizado supervisionado usados tanto para problemas de classificação como de regressão. Como seu nome indica, é um método amplamente usado na tomada de decisões. De maneira geral, as árvores de decisão são uma representação gráfica e hierárquica de escolhas que apresentam resultados possíveis em forma de árvore, buscando encontrar a melhor divisão para o conjunto de dados. Por esta razão, a ideia da árvore de decisão é criar um modelo capaz de predizer o valor de uma variável de saída, aprendendo regras básicas de decisão a partir dos atributos fornecidos no treinamento. As árvores geradas são compostas por um nó raiz, ramificações, nós internos ou de decisão, e nós folhas ou terminais.

- ***Random Forests*:** Conhecidas, em português, como Florestas Aleatórias. É um algoritmo que combina a saída de várias árvores de decisão para chegar a uma única predição ou resultado com maior precisão. Tem como característica a incorporação do componente aleatório na partição dos nós.

- **Redes Neurais:** Redes Neurais são modelos que permitem simular o comportamento do cérebro humano, ensinando o sistema computacional a processar dados de uma maneira similar a como funcionam as redes neurais biológicas. Esse tipo de algoritmo usa nós ou unidades interconectadas por ligações numa estrutura de camadas. As redes neurais são algoritmos de aprendizado que se ajustam à precisão, permitindo a classificação e agrupamento de dados a grandes velocidades. O algoritmo aprende a partir dos dados de treinamento e dos erros cometidos no processo, tendendo a melhorar com o tempo, ou seja, aprende da experiência, sendo capaz de modificar seu comportamento e se ajustar aos dados do entorno.

- **Modelos de regressão linear:** Neste tipo de modelagem tem-se um conjunto de dados rotulados, sendo que o valor da variável de saída será determinado pelos valores de entrada. Na regressão linear, a ideia principal é tentar ajustar uma linha reta ao conjunto de dados sempre que exista uma relação linear entre as variáveis de estudo, ou seja, através de uma equação linear modelar a variável dependente e independente do estudo. Cabe salientar que a variável dependente na regressão linear é contínua.

- **Modelos de regressão logística:** Este tipo de regressão fornece um resultado binomial, dada a probabilidade de um evento acontecer ou não baseado nos valores das variáveis de entrada (variáveis independentes ou preditoras). Na regressão logística, diferente da regressão linear, a variável de previsão ou dependente é categórica.

2.4.2 Aprendizado Não Supervisionado

Diferentemente do aprendizado supervisionado, o aprendizado não supervisionado trabalha a partir de dados não rotulados, onde o algoritmo tenta identificar padrões usando só os dados de entrada, sem outras informações ou resultados definidos previamente, criando agrupamentos que permitam entender os padrões identificados. Neste caso, três classificações são conhecidas: *clustering*, associação e redução de dimensionalidade.

Dentre os algoritmos usados no aprendizado não supervisionado se encontram:

- **Agrupamento de k-médias:** Técnica usada frequentemente para resolver problemas de agrupamento. Permite detectar grupos de objetos semelhantes que estejam relacionados entre si, gerando agrupamentos, também chamados *clusters*.

- **Modelos de variáveis latentes:** Trata-se de um modelo estatístico que contém variáveis latentes, ou seja, variáveis que não são diretamente observadas, mas que afetam ou influenciam as variáveis de resposta. As variáveis latentes são usadas para representar o efeito de fatores não observáveis por meio de uma ou mais variáveis indicadoras (observadas). Este tipo de modelo é útil para capturar propriedades complexas ou conceituais que são difíceis de quantificar ou medir diretamente.

- **Análises de componentes principais (ACP):** Técnica estatística que se caracteriza por reduzir a dimensionalidade de um conjunto de dados, preservando o máximo de variabilidade possível. A ACP converte um conjunto de variáveis possivelmente correlacionadas em um conjunto menor de variáveis, conhecidas como componentes principais, que se caracterizam por não estarem mais correlacionadas.

2.4.3 Aprendizado por Reforço

O aprendizado por reforço é uma abordagem do aprendizado de máquina na qual um agente aprende a tomar decisões por meio da interação com um ambiente, buscando maximizar uma função de recompensa. Diferentemente do aprendizado supervisionado, que requer dados rotulados, o aprendizado por reforço baseia-se na

experiência adquirida durante a prática, utilizando tentativa e erro para encontrar a melhor solução para um problema proposto.

Essa técnica é particularmente eficaz em cenários que envolvem múltiplas decisões sequenciais, como jogos, controle de robôs e, mais recentemente, sistemas de codificação de vídeo. Por exemplo, o algoritmo AlphaGo, desenvolvido pela DeepMind, utilizou aprendizado por reforço para aprender estratégias de jogo superiores às humanas (Silver; Huang; Maddison; Guez; Sifre; Van den driessche; Schrittwieser; Antonoglou; Panneershelvam; Lanctot et al., 2016).

No contexto da codificação de vídeo, o aprendizado por reforço tem sido aplicado para resolver problemas de otimização combinatória, como a seleção de modos de predição, a ordenação de operações de codificação e a escolha adaptativa de transformadas. Ao utilizar este aprendizado, o codificador pode aprender, por exemplo, quais transformadas testar e em que ordem, de forma a reduzir o número de operações realizadas, diminuindo o custo computacional e energético sem comprometer significativamente a qualidade da reconstrução (Li; Zhang; Zhu; Luo; Kwong, 2019).

Além disso, o aprendizado por reforço tem sido empregado para o controle adaptativo de taxa de bits, onde o agente aprende a alocar recursos de forma eficiente entre quadros ou blocos, considerando restrições de largura de banda e qualidade perceptual (He; Jin; Tian; Liu; Deng; Shi, 2024). Há também abordagens que utilizam este aprendizado para decidir dinamicamente a profundidade da árvore de particionamento, otimizando a granularidade da divisão dos blocos de vídeo conforme o conteúdo.

Portanto, o aprendizado por reforço oferece uma estrutura promissora para a tomada de decisão inteligente e adaptativa em codificadores de vídeo de próxima geração, permitindo abordagens mais flexíveis, eficientes e energeticamente viáveis.

2.5 Resumo do Capítulo

Este capítulo apresentou uma visão abrangente sobre os principais conceitos relacionados à codificação de vídeo digital e suas inovações recentes, com ênfase no padrão *Versatile Video Coding* (VVC). Inicialmente, discutiu-se como a codificação de vídeo permite reduzir a quantidade de dados necessária para a transmissão e o armazenamento de vídeos, por meio de técnicas como subamostragem de cores e compressão que explora redundâncias espaciais, temporais e estatísticas (entrópicas).

Em seguida, foram detalhadas as principais melhorias do VVC em relação ao seu antecessor, o HEVC, incluindo o aumento do tamanho das unidades de codificação e a introdução de novas transformadas. Entre essas inovações, destacou-se a técnica de *Multiple Transform Selection* (MTS), que permite aplicar diferentes transformadas na codificação de blocos residuais, buscando otimizar a eficiência de compressão

conforme as características do conteúdo.

Adicionalmente, foi abordada a aplicação do aprendizado de máquina no contexto da codificação de vídeo, evidenciando sua relevância na melhoria do desempenho dos sistemas. Foram discutidas as principais abordagens de aprendizado (supervisionado, não supervisionado e por reforço), ressaltando suas aplicações em diferentes tipos de problemas relacionados à compressão e otimização de recursos computacionais.

3 REVISÃO DA LITERATURA

A maioria dos codificadores de vídeo desenvolvidos até hoje normalmente é baseada em um esquema de codificação que utiliza a família da DCT, portanto é muito comum encontrar trabalhos na literatura com soluções em *hardware* para este tipo de transformada. Nesta seção serão apresentadas algumas soluções em *hardware* e *software*, encontradas na literatura, para a etapa das transformadas do VVC.

Para realizar a revisão da literatura, foram adotadas algumas técnicas para o mecanismo de busca de trabalhos relacionados ao tema deste trabalho. Primeiro, foi necessário definir os elementos essenciais de estudo e, assim, decidir o objetivo geral da revisão.

Objetivo da revisão: Identificar trabalhos que tragam soluções em *hardware* e *software* para o módulo das transformadas do codificador de vídeo VVC.

Após definido o objetivo de busca, o próximo passo foi escolher palavras-chave para formar a *string* de busca listada abaixo:

```
(transform OR transforms OR MTS OR "multiple transform selection" OR DCT OR DST) AND
("versatile video coding" OR VVC)
```

O fluxograma da revisão pode ser visto na Figura 12. Os mecanismos de busca adotados foram:

- *IEEE Xplore Digital Library*[1];
- *Google Scholar Library*[2];

O primeiro resultado desta revisão levou em consideração artigos publicados entre os anos de 2018 a 2022, devido ao fato de que o padrão VVC começou a ser desenvolvido em meados de 2018. Os resultados obtidos incluem 164 artigos na base de dados da *IEEE Xplore* e 325 da base de dados do *Google Scholar*, resultando em um total de 489 artigos.

[1] Disponível em `http://ieeexplore.ieee.org`
[2] Disponível em `http://scholar.google.com.br`

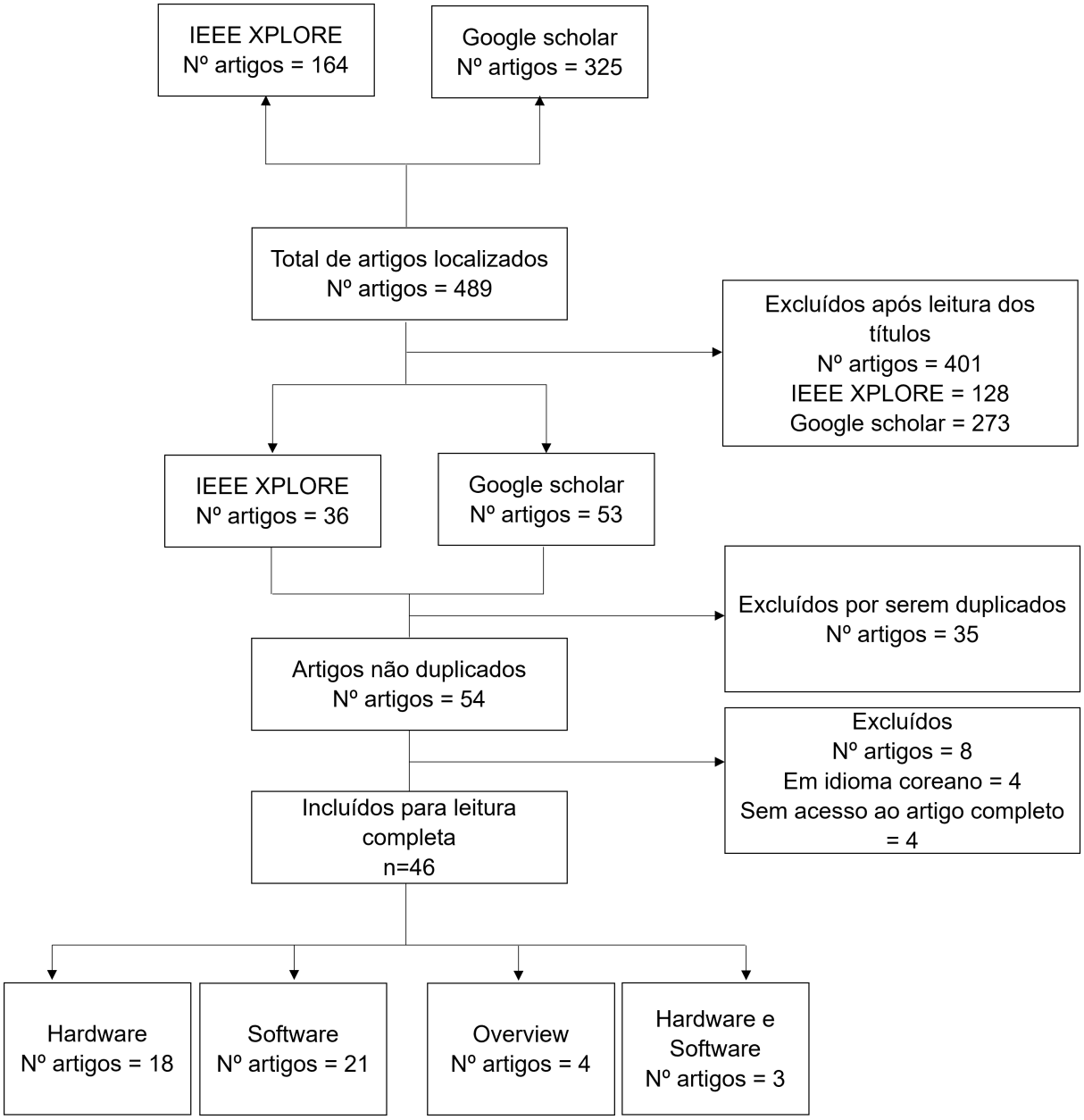

Figura 12 – Fluxograma das etapas de revisão da literatura.

Para refinar mais o resultado de busca, a *string* foi explorada apenas no título dos artigos, resultando em 36 artigos na base de dados *IEEE Xplore* e 52 artigos na plataforma *Google Scholar*, totalizando 88 artigos. Deste total, 8 artigos foram excluídos, 4 por serem no idioma coreano e 4 por indisponibilidade de acesso ao artigo completo. Os 80 artigos resultantes foram catalogados por título, resumo, ano e conferência e assim observou-se uma duplicação de alguns deles. Sendo assim, foi realizado um último refinamento para exclusão destes artigos duplicados. Por fim, restaram 45 artigos para a realização desta revisão bibliográfica.

Os trabalhos selecionados foram divididos em soluções de *hardware* e soluções em *software* para o módulo das transformadas do padrão VVC. Para realizar esta categorização, foi realizada a leitura dos *abstracts*, da introdução e da conclusão de

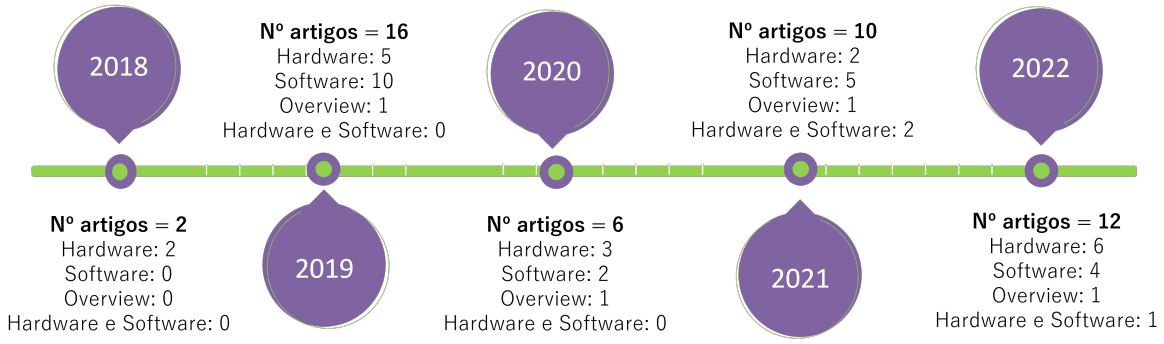

Figura 13 – Linha temporal dos artigos encontrados na revisão da literatura.

todos os artigos. O gráfico da Figura 13 apresenta o número de artigos encontrados por ano e divididos por classes.

3.1 Soluções em *hardware* para as Transformadas do VVC

Nesta seção, são apresentados os trabalhos disponíveis na literatura que propõem arquiteturas de *hardware* para o módulo de transformadas do padrão de codificação de vídeo VVC.

Os trabalhos encontrados na literatura abordam diversas soluções para a implementação de transformadas em *hardware*, com ênfase nas transformadas DCT-II, DCT-VIII, DST-VII e LFNST, para diferentes tamanhos de blocos, variando de 4×4 a 64×64, incluindo tamanhos quadrados e retangulares. A maioria dos estudos propõe arquiteturas otimizadas para FPGA, com técnicas como a substituição de multiplicadores por somas e deslocamentos, *pipeline*, e uso de memórias de transposição, visando reduzir a complexidade computacional e melhorar a eficiência de área e consumo de energia. Algumas abordagens incluem a decomposição de matrizes para reduzir a área (como nas DCT-VIII e DST-VII), enquanto outras exploram a reutilização de *hardware* para diferentes tipos de transformadas e aproveitam a similaridade entre os coeficientes das matrizes, como no caso das DCT-II. Além disso, várias soluções também foram projetadas para suportar vídeos 2K e 4K, com alta taxa de quadros, e incluem metodologias escaláveis para otimizar o *throughput* e a latência, atendendo aos requisitos de padrões como MPEG, AVC, HEVC e VVC.

O estudo apresentado em Mert; Kalali; Hamzaoglu (2017) foi conduzido durante a fase de exploração do padrão VVC. Nele, é proposta a implementação de uma arquitetura de *hardware* capaz de executar as operações das transformadas DCT-II, DCT-V, DCT-VIII, DST-I e DST-VII para blocos de dimensões 4×4 e 8×8. A abordagem adotada para a implementação consiste na substituição dos multiplicadores convencionais por operações baseadas em somas e registradores de deslocamento, visando a redução do custo computacional e do consumo de recursos. Além disso, foi realizada

uma comparação entre a arquitetura proposta e as implementações de *hardware* das transformadas bidimensionais DCT/IDCT presentes no codificador HEVC original.

Ainda durante a fase de desenvolvimento do padrão VVC, o estudo apresentado em Kammoun; Hamidouche; Belghith; Nezan; Masmoudi (2018) propõe a implementação das cinco transformadas previamente mencionadas, abrangendo blocos de tamanhos variando de 4×4 até 32×32, incluindo todas as dimensões quadradas intermediárias. A abordagem adotada considera a otimização dos recursos de *hardware* disponíveis na plataforma FPGA, explorando técnicas específicas para a sua eficiente implementação.

O estudo apresentado em Garrido; Pescador; Chavarrías; Lobo; Sanz (2019) propõe uma arquitetura de alto desempenho para a implementação de todas as transformadas bidimensionais (2D) da MTS, incluindo DCT-II, DCT-VIII e DST-VII, para blocos de tamanhos 4×4, 8×8, 16×16 e 32×32. A arquitetura utiliza a técnica de pipeline e memórias de transposição para otimizar o processamento. Posteriormente, essa proposta foi expandida em Garrido; Pescador; Chavarrías; Lobo; Sanz; Paz (2020), incorporando blocos de tamanhos maiores, de 4×4 até 64×64, além da inclusão de formatos retangulares, ampliando a flexibilidade da implementação.

O trabalho apresentado em Yibo; Jiro; Heming; Xiaoyang; Yixuan (2019) propõe uma arquitetura unificada para a implementação das transformadas DCT-VIII e DST-VII, abrangendo todos os tamanhos de blocos, tanto quadrados quanto retangulares. A proposta faz uso do algoritmo *N-Dimensional Reduced Adder Graph* (RAG-n), cuja principal finalidade é minimizar a quantidade de somadores, reduzindo a complexidade computacional. Além disso, uma memória de transposição é empregada para armazenar blocos de diferentes tamanhos. A arquitetura desenvolvida foi inicialmente implementada para uma única dimensão (1D) e posteriormente expandida para duas dimensões (2D) no estudo apresentado em Fan; Zeng; Sun; Katto; Zeng (2019), no qual técnicas adicionais de pipeline foram incorporadas.

O estudo apresentado em Kammoun; Hamidouche; Philipp; Belghith; Massmoudi; Nezan (2019) propõe uma implementação de *hardware* baseada no método de aproximação do módulo da MTS. Duas soluções arquiteturais são propostas para a realização das transformadas. A primeira consiste em uma arquitetura eficiente e unificada, com *pipeline*, para a implementação da DCT-II direta e inversa, suportando tamanhos de blocos de 4, 8, 16 e 32. Essa abordagem busca reduzir a complexidade computacional e otimizar a alocação lógica dos recursos do FPGA. A segunda solução propõe uma implementação 2D aproximada das transformadas direta e inversa da DST-VII e DCT-VIII. Os resultados de síntese indicam que a arquitetura desenvolvida é capaz de processar vídeos em resoluções 2K e 4K a taxas de 377 e 94 quadros por segundo, respectivamente, utilizando apenas 18% dos *Adaptive Logic Modules* (ALMs), 40% dos registradores e 34% dos blocos de DSP do dispositivo.

O estudo apresentado em Damak; Houidi; Ayed; Masmoudi (2020) propõe a implementação de uma arquitetura de *hardware* eficiente para a transformada DCT-II. A solução desenvolvida suporta tamanhos de blocos variando de 4×4 a 64×64, utilizando multiplexadores para selecionar dinamicamente o tamanho do bloco por meio de uma entrada externa. A avaliação da arquitetura foi realizada exclusivamente em um FPGA, apresentando resultados relacionados à ocupação de área e à frequência mínima de operação.

Em Farhat; Hamidouche; Grill; Menard; Déforges (2020) é realizada uma comparação entre duas arquiteturas de *hardware* para a MTS 1D. A primeira abordagem é baseada na técnica de *Multiple Constant Multiplication* (MCM), que substitui multiplicações por somas e deslocamentos, enquanto a segunda utiliza apenas 32 multiplicadores, denominados *Regular Multipliers* (RM), compartilhados entre todas as transformadas da arquitetura. Uma memória ROM é empregada para armazenar os coeficientes. Ambas as arquiteturas foram projetadas para o módulo das transformadas inversas e suportam blocos variando de 4 até 64 pontos. No estudo subsequente é proposta uma arquitetura multi-padrão para o módulo das transformadas inversas, capaz de processar os padrões MPEG, AVC, HEVC e VVC (Farhat; Hamidouche; Grill; Ménard; Déforges, 2021). A abordagem abrange tanto transformadas separáveis quanto não separáveis para todos os tamanhos de blocos permitidos, adotando a técnica de *Regular Multipliers*, que se mostrou mais eficiente na comparação realizada no trabalho anterior. A arquitetura apresenta uma estrutura escalável, permitindo o aumento do *throughput* e da latência mediante a ampliação do número de multiplicadores. A solução implementa as transformadas DCT-II, DCT-VIII, DST-VII e LFNST em duas dimensões (2D).

Considerando que a DCT-II é a transformada mais selecionada pelo codificador do VVC, o estudo apresentado em Imen; Fatma; Amna; Masmoudi (2021) propõe uma arquitetura específica para sua implementação. O trabalho desenvolve um *design* otimizado para os tamanhos de bloco 4×4 e 8×8, baseado nas equações extraídas do *software* de referência VTM. A abordagem emprega exclusivamente operações de soma e deslocamento, eliminando a necessidade de multiplicadores, com o objetivo de reduzir a complexidade computacional e o consumo de recursos de *hardware*.

No trabalho de Ben jdidia; Belghith; Masmoudi (2022) é proposta uma arquitetura para a DST-VII, abrangendo tamanhos de bloco de 8×8 até 32×32, incluindo blocos retangulares. A arquitetura é projetada para suportar três técnicas diferentes para o cálculo da transformada 1D: (1) cálculo com multiplicadores, (2) cálculo com somas e deslocamentos, e (3) cálculo aproximado baseado na DCT-II. A escolha entre esses três métodos é feita por meio de um sinal externo de controle. Além disso, a arquitetura 1D é reutilizada para o cálculo de uma segunda transformada 1D, utilizando uma realimentação da saída para a entrada, o que aumenta a eficiência do processamento.

Como mencionado anteriormente, o módulo das transformadas é composto pelas transformadas primárias e secundárias. A LFNST é uma transformada secundária, aplicada ao bloco previamente transformado pela transformada primária. O trabalho em (Goebel; Costa; Agostini; Zatt; Porto, 2022) propõe uma arquitetura eficiente para o cálculo da LFNST. O *design* do *hardware* foi otimizado para receber uma linha de cada bloco a cada ciclo de *clock*, aproveitando o fato de que muitas arquiteturas de transformadas primárias entregam os resultados linha por linha em cada ciclo. Em seguida, o trabalho em Goebel; Agostini; Zatt; Porto (2022) utiliza a mesma técnica, mas com a introdução de dois domínios de *clock* distintos, a fim de melhorar a relação entre área e consumo de energia da arquitetura. Um domínio é utilizado para o recebimento e entrega das amostras, enquanto o outro é destinado ao processamento das operações.

Para reduzir a área das transformadas, o trabalho em Hao; Zheng; Fan; Xiang; Zhang; Sun (2022) propõe a decomposição das matrizes das transformadas DCT-VIII e DST-VII em duas matrizes mais simples, denominadas *"Low-value matrix"* (matriz de baixo valor) e *"Error matrix"* (matriz de erro). Essas matrizes são multiplicadas pelo bloco residual, gerando valores intermediários que são somados para produzir o resultado final da operação. No caso da DCT-II, o algoritmo *Butterfly* foi utilizado para realizar o cálculo da transformada.

Em (Zhang; Shi; Zhang, 2022) também é proposta uma arquitetura unificada da MTS, porém apenas para uma dimensão. O projeto consiste em uma unidade de quatro transformadas de 4 *poitns* como *hardware* básico. Ela utiliza computação paralela e multiplexação por divisão de tempo para aumentar a taxa de reutilização da arquitetura de *hardware*.

A Tabela 4 fornece um resumo das principais características dos trabalhos analisados. Os resultados obtidos a partir dessas pesquisas serão discutidos no capítulo 7 desta tese.

3.2 Soluções em *software* para as Transformadas do VVC

Nesta seção, são apresentados os principais trabalhos que propuseram novas soluções de *software* para o módulo das transformadas do codificador do padrão de vídeo VVC. O objetivo é sintetizar os achados desses estudos, destacando as soluções apresentadas pelos autores. Embora o VVC ofereça desempenho de codificação superior ao HEVC, a codificação de vídeo ainda é um processo demorado devido à alta complexidade computacional das novas ferramentas introduzidas. Como resultado, diversos métodos e algoritmos foram propostos, com ênfase em otimizar o tempo de execução, sem comprometer a eficiência da codificação.

Uma das estratégias propostas para reduzir a complexidade computacional é a uti-

Tabela 4 – Características dos trabalhos relacionados com soluções em *hardware*

	Transformadas suportadas	Tamanho Blocos	Dimensões	Tecnologia
(KAMMOUN et al., 2018)	DCT-II/V/VIII DST-I/VII	NxM N=4,8,16,32 M=4,8,16,32	2D	FPGA 20nm
(GARRIDO et al., 2019)	DCT-II/VIII DST-VII	NxN N=4,8,16,32	2D	FPGA 20nm/14nm
(YIBO et al., 2019)	DCT-VIII DST-VII	NxM N=4,8,16,32 M=4,8,16,32	1D	ASIC 65nm
(FAN et al., 2019)	DCT-VIII DST-VII	NxM N=4,8,16,32 M=4,8,16,32	2D	ASIC 65nm
(KAMMOUN et al., 2019)	DCT-II/VIII IDCT-II/VIII DST-VII IDST-VII	NxM N=4,8,16,32 M=4,8,16,32	2D	FPGA
(KAMMOUN et al., 2019)	DCT-II/VIII IDCT-II/VIII DST-VII IDST-VII	NxN N=4,8,16,32	2D	FPGA 20nm
(GARRIDO et al., 2020)	DCT-II/VIII DST-VII	NxM N=4,8,16,32,64 M=4,8,16,32,64	2D	FPGA 20nm
(FARHAT et al., 2020)	IDCT-II/VIII IDST-VII	N N=4,8,16,32,64	1D	ASIC 28nm
(IMEN et al., 2021)	DCT-II	NxN N=4,8	2D	FPGA
(FARHAT et al., 2021)	IDCT-II/VIII IDST-VII LFNST	NxM N=4,8,16,32,64 M=4,8,16,32,64	2D	ASIC 28nm
(BEN et al., 2022)	DST-VII	NxM N=8,16,32,64 M=8,16,32,64	2D	FPGA
(GOEBEL et al., 2022)	LFNST	NxM N=4,8 M=4,8	2D	ASIC 40nm
(HAO et al., 2022)	DCT-II/VIII DST-VII	N N=4,8,16,32,64	1D	ASIC 65nm
(GOEBEL et al., 2022)	LFNST	NxM N=4,8 M=4,8	2D	ASIC 45nm
(ZHANG et al., 2022)	DCT-II/VIII DST-VII	NxM N=4,8,16,32,64 M=4,8,16,32,64	1D	FPGA 14nm

lização de métodos rápidos, como o apresentado por Zhang; Zhao; Li; Li; Liu (2019). Neste trabalho, os autores propõem um método rápido denominado *Adaptive Multiple Transforms* (AMT), que utiliza um algoritmo do tipo *butterfly* para as transformadas DST-VII e DCT-VIII. A abordagem visa reduzir o número de operações, mantendo quase o mesmo desempenho de codificação. Para isso, as propriedades de ortogonalidade das transformadas foram exploradas por meio de ajustes nas matrizes. A solução foi implementada para tamanhos de 16, 32 e 64 pontos, com testes de codificação e decodificação em 26 sequências de vídeo variando de 416x240 a 3840x2160, incluindo vídeos com conteúdo misto natural e de tela. Os resultados mostraram uma redução média de 7%, 5% e 6% no tempo de decodificação para os modos *All Intra (AI)*, *Random Access* (RA) e *Low Delay* B (LDB), respectivamente. Em termos de codificação, as economias de tempo foram de 3%, 6% e 8%, com os melhores resultados observados em sequências específicas, como *RaceHorses* (21%), *FoodMarket4* (8%) e *SlideShow* (22%). Com base nesses resultados, os autores concluem que o método proposto proporciona uma maior eficiência de tempo em comparação com o *codec VTM-1.1*, sem comprometer o desempenho de codificação.

Em Zhang; Zhao; Li; Li; Luo; Liu; Li (2020), os autores apresentam uma versão estendida do trabalho anterior, com uma abordagem mais atualizada de acordo com o desenvolvimento do VVC. Neste artigo, os métodos rápidos são descritos em detalhes, incluindo uma análise mais profunda das transformadas DST-VII e DCT-VIII. A partir das características dessas transformadas, os autores propõem um algoritmo rápido parcial com suporte para implementação dual, oferecendo uma redução aproximada de 50% na contagem de operações. Para avaliar a eficácia, foram realizados testes comparando o desempenho do método com o VTM-3.0 e outros métodos relevantes. Os resultados mostram que, ao executar repetidamente a transformada DCT-VIII de 16 pontos, o método proposto economizou 57,52% e 67,76% do tempo de execução para as transformadas direta e inversa, respectivamente. Para a transformada de 32 pontos, as economias foram de 48,65% e 62,84%. Além disso, os testes sob diferentes condições de codificação, com MTS ativado, demonstraram uma economia de tempo de decodificação de até 9% em All Intra e 3% em *Low Delay* B. Com essas melhorias, os autores concluem que os métodos rápidos proporcionam uma significativa redução no tempo de execução do *software*, mantendo a eficiência de codificação em comparação com o VVC *Test Model VTM-3.0*.

O trabalho de Fu; Zhang; Mu; Chen (2019a) aborda a complexidade computacional da MTS no contexto do VVC, destacando o impacto do processo de *Rate Distortion Optimization* (RDO), que é essencial para avaliar várias transformadas e obter ganhos de codificação. Para reduzir essa complexidade, os autores propõem um novo algoritmo rápido de dois estágios que utiliza estatísticas de codificação espacial e informações da transformada primária para antecipar a finalização do processo RDO da

MTS. Os experimentos foram realizados com seis sequências de vídeo de diferentes resoluções e texturas, testando trinta *frames* em condições de teste JVET comuns. O algoritmo reordena as transformadas candidatas com base nas frequências de escolha dos blocos vizinhos, permitindo que a MTS seja interrompida mais rapidamente quando uma transformada menos provável for detectada. Os resultados indicaram uma redução de até 23% no tempo de execução, com um aumento marginal de 0,16% no BD-BR. Os autores concluem que o algoritmo proposto é robusto, funcionando bem em diversas condições de sequências de vídeo, incluindo aquelas de alta resolução ou alta largura de banda.

O artigo de Wang; Wang; Yang; Luo; Liang; Huang (2022a) propõe um algoritmo rápido para a seleção de múltiplas transformadas no codificador VVC, com o objetivo de reduzir o tempo de *Rate Distortion Optimization* (RDO) da MTS. Os autores exploram a relação entre o núcleo (*kernel*) da transformada escolhida e a distribuição dos coeficientes residuais no bloco de codificação. No processo de codificação do VVC, o codificador geralmente precisa testar todas as combinações de *kernels* e calcular o custo de taxa-distorção para determinar a melhor transformada. O algoritmo rápido proposto usa características da distribuição residual para selecionar os *kernels* ideais, acelerando o processo de busca e reduzindo a complexidade. Implementado no *software* de referência VTM13.0, o algoritmo foi testado em condições da CTC com quatro classes de sequências de vídeo e diferentes parâmetros de quantização (QP). Os resultados mostraram uma redução de 25% no tempo de codificação, mantendo o desempenho semelhante em termos de BD-BR. Essa redução foi confirmada pela análise teórica, já que o algoritmo rápido realiza o RDO apenas duas ou três vezes, reduzindo a complexidade de codificação das transformadas em 40% a 60%.

O artigo de Saldanha; Sanchez; Marcon; Agostini (2022a) propõe um esquema de decisão rápida para a seleção de transformadas na predição intraquadro do VVC, utilizando árvores de decisão para otimizar o processo de codificação. O objetivo é reduzir o tempo de codificação sem comprometer a eficiência da compressão. O estudo realizou análises sobre a escolha das transformadas na MTS e na LFNST, identificando uma correlação entre o contexto de codificação e seus atributos para definir classificadores de árvores de decisão que determinam quando essas transformadas podem ser removidas. A mineração de dados foi usada para treinar os classificadores, utilizando informações estatísticas coletadas de 8 sequências de vídeo de diferentes resoluções e características. Os resultados indicaram que, ao usar as árvores de decisão para MTS, houve uma economia de 5% no tempo de codificação com um leve aumento de 0,21% no BD-BR. Para a LFNST, a redução na complexidade de codificação foi de 6,40%, com um aumento de 0,23% no BD-BR. Quando ambas as soluções foram combinadas, a economia de tempo foi de 10,99%, com um aumento de 0,43% no BD-BR. Os autores concluem que o esquema proposto oferece uma significativa redução

no tempo de codificação com impacto mínimo na eficiência de codificação.

O trabalho de Nguyen; Bross; Keydel; Schwarz; Marpe; Wiegand (2019) descreve detalhadamente um esquema unificado para a MTS que incorpora o *Transform Skip Mode* (TSM) para blocos de 32×32 da transformada luma, com o objetivo de reduzir o tempo de codificação sem comprometer a eficiência. O esquema proposto visa resolver o aumento do tempo de codificação, introduzindo um novo algoritmo para a decisão de modo da MTS, permitindo que o tempo de codificação se mantenha semelhante ao de uma configuração que usa uma extensão direta do TSM. Os resultados mostraram uma melhoria na eficiência de compressão, com ganhos de cerca de 2% para sequências da Classe F e 8% para sequências *Text and Graphics with Motion* (TGM), tanto na configuração *All-Intra* quanto na *Random-Access*.

O trabalho de Hamidouche; Philippe; Mohamed; Kammoun; Menard; Déforges (2019) aborda a complexidade das aproximações DST-VII e DCT-VIII, propondo uma solução baseada na reutilização da arquitetura DCT-II existente para otimizar o cálculo dessas transformadas. Diferente da DCT-II, que possui algoritmos rápidos bem estabelecidos, a DST-VII e a DCT-VIII dependem de multiplicações de matrizes mais complexas e carecem de implementações eficientes. A solução proposta modela as aproximações como um problema de otimização inteira, ajustando a banda-matriz para minimizar o erro enquanto mantém a ortogonalidade. A integração dessa abordagem ao padrão VVC resultou em uma redução de 25% no número de multiplicações e na memória necessária para armazenar os coeficientes transformados, sem comprometer o desempenho de codificação. Esse método é especialmente vantajoso para implementações em *hardware*, como em plataformas embarcadas com recursos limitados.

Nesta mesma linha, o trabalho de Hamidouche; Philippe; Fezza; Haddou; Pescador; Menard (2022) investiga a aproximação das transformadas DST-VII e DCT-VIII, visando reduzir a complexidade e a utilização de memória no bloco MTS, especialmente em plataformas com recursos limitados. A proposta modela a aproximação da DST-VII como um problema de otimização contínua com restrições, utilizando o *kernel* da DCT-II para manter o ganho de codificação da MTS enquanto minimiza o número de multiplicações e o uso de memória. A solução foi implementada no *software* de referência VTM3.0, focando nas DST-VII e DCT-VIII para blocos de tamanhos 16, 32 e 64. O desempenho foi avaliado em termos de BD-BR, complexidade (multiplicações/adições) e memória necessária. Os resultados mostraram uma redução de até 92% no número de multiplicações, preservando os ganhos de codificação e reduzindo a memória. Os autores destacam que a abordagem pode ser aplicada ao decodificador H.266/VVC em dispositivos de consumo com recursos limitados, como em termos de energia e memória.

No estudo de Said; Egilmez; Chao (2019), os autores propõem uma nova técnica

para projetar aproximações de baixa complexidade de transformadas trigonométricas, visando reduzir o custo computacional e de memória em transformadas de grandes tamanhos, como 32 pontos ou mais. Eles utilizam uma família de transformadas eficientes intimamente relacionadas à DCT-II, como DCT-III, DST-II e DST-III, além de transformadas ortogonais simples chamadas ajustes. A técnica é composta por dois tipos de ajustes: um pré-ajuste com matrizes de banda esparsa, seguido por uma DCT-II, DCT-III, DST-II ou DST-III no codificador, e um pós-ajuste com matrizes esparsas de bloco. Os métodos foram aplicados para reduzir a complexidade das DST-VII e DCT-VIII, testados no VTM-3.0, com o objetivo de melhorar a eficiência computacional. Os resultados mostraram que as aproximações para as DST-VII e DCT-VIII de 32 e 64 pontos mantiveram o desempenho de compressão praticamente inalterado. Além disso, a abordagem baseada em sub-blocos 8×8 ofereceu uma melhor compensação entre desempenho e complexidade, reduzindo as multiplicações por coeficiente de 64 para 36 e a memória necessária em 1 kbyte. A técnica proposta demonstrou uma significativa redução na complexidade computacional, sem comprometer a eficiência da codificação.

O estudo de Abdallah; Belghith; Masmoud (2019) visa aprimorar a taxa de compressão do padrão HEVC, com foco na aplicação de um esquema de Transformada Múltipla Adaptativa (AMT). Este esquema utiliza quatro transformadas das famílias DCT/DST: DST-I, DST-VII, DCT-V e DCT-VIII, e é empregado para predições interquadros e intraquadro. O trabalho propõe que, dependendo de um sinalizador, diferentes transformadas sejam selecionadas. Quando o sinalizador está em 0, a DCT-II e DST-VII são usadas para predição intraquadro, enquanto apenas a DCT-II é aplicada para o resíduo de predição interquadros. Quando o sinalizador é igual a 1, um subconjunto contendo DCT-II, DST-VII e DCT-VIII é utilizado. O algoritmo desenvolvido pelos autores remove as transformadas DCT-V e DST-I, que são menos frequentemente usadas, e utiliza apenas DCT-II, DST-VII e DCT-VIII, tanto para predições intraquadro quanto interquadros. Os resultados experimentais demonstraram que o algoritmo proposto reduziu a complexidade em 41,05%, com um leve aumento na taxa de bits (0,008%) e uma mínima perda de qualidade, com variação de 0,004 dB no PSNR.

O estudo de Schwarz; Nguyen; Marpe; Wiegand; Karczewicz; Coban; Dong (2019) propõe a substituição da quantização escalar convencional na codificação de vídeo HEVC pela quantização codificada em treliça (*Trellis-coded quantization* - TCQ), além da utilização de dependências estatísticas adicionais entre índices de quantização para a codificação de entropia. O TCQ apresenta a vantagem de ser semelhante à quantização escalar com quantizadores de reconstrução uniforme (*Uniform-reconstruction quantizers* - URQs), permitindo sua integração direta com técnicas modernas de codificação de entropia e algoritmos de decisão de codificador. Para avaliar a contribuição de eficiência da codificação dos coeficientes das transformadas mo-

dificadas e do TCQ, foram realizadas duas versões de testes: uma com mudanças apenas na codificação de entropia (VTM-1 + *transform coefficient coding* - TCC) e outra com a adição do TCQ (VTM-1 + TCC + TCQ). Os experimentos foram conduzidos para configurações *All Intra*, *Random Access* e *Low Delay*. Os resultados mostraram economias médias de taxa de bits de 4,9% (AI), 3,4% (RA) e 2,8% (LD). A inclusão apenas da codificação de coeficientes das transformadas resultou em economias de 1,5%, 1,0% e 0,8% para AI, RA e LD, respectivamente. Os autores concluem que a eficiência de codificação dos *codecs* de vídeo híbridos modernos pode ser substancialmente aprimorada com a substituição dos quantizadores escalares convencionais pelo quantizador codificado em treliça.

O estudo de Medina; Saha; Floriano; Chavarrías; Pescador (2019) propôs uma abordagem inicial para a migração de funcionalidades do VVC para plataformas embarcadas, utilizando OpenMP. A implementação focou no decodificador HEVC, incorporando algumas funcionalidades do VVC, começando pela Transformada Múltipla Adaptativa (AMT), com a intenção de implementar posteriormente a nova MTS. Três plataformas foram testadas: um processador de propósito geral (GPP1) de alto desempenho com 16 núcleos a 3,4 GHz, uma plataforma embarcada heterogênea (GPP2) com 8 núcleos a 2,26 GHz e uma GPU, e um *embedded*-PC GPP3 com 8 núcleos a 1,4 GHz. A eficiência da decodificação foi avaliada nas três plataformas, e o desempenho da AMT foi analisado especificamente no GPP2. As sequências de teste JCT-VC foram codificadas com processamento paralelo *Wavefront* (WPP) nas configurações HEVC-only e HEVC com AMT, utilizando parâmetros de quantização de 27 e 32. Os resultados mostraram que a solução proposta superou a implementação anterior em 10% no GPP1 para transformadas HEVC e até 20% para AMT. Desempenhos semelhantes foram observados nas plataformas GPP2 e GPP3. Os autores concluíram que o OpenMP se mostrou uma solução competitiva, com uma melhoria média de 10% no desempenho e aceleração, sugerindo que essa abordagem pode servir como ponto de partida para a implementação futura de um decodificador VVC completo utilizando OpenMP.

Um trabalho similar de Vázquez; Saha; Morillas; Lapastora; Oso (2019) propôs a migração das transformadas do VVC para uma plataforma heterogênea baseada em Unidade de Processamento Gráfico (GPU), aproveitando a capacidade de paralelização dessa tecnologia para lidar com a alta complexidade computacional das transformadas, especialmente a AMT. Os autores argumentaram que, embora a AMT tenha um custo maior de complexidade em comparação ao HEVC, essa sobrecarga pode ser gerenciada eficientemente pela GPU. Os testes foram conduzidos em uma plataforma composta por 8 núcleos de CPU ARM a 2,26 GHz e 12 núcleos de GPU, utilizando sequências de vídeo com parâmetros de quantização 27 e 32 fornecidas pelo JCT-VC. Foram testadas as transformadas DST-I e DST-VII de tamanho 32×32, bem como

diferentes subconjuntos da AMT, como DST-I (Vertical) com DST-I (Horizontal), DST-I (Vertical) com DST-VII (Horizontal) e DST-VII (Vertical) com DST-I (Horizontal). Os resultados mostraram que os tempos de processamento na GPU foram, no mínimo, 13 vezes mais rápidos para o DST-I e 5 vezes mais rápidos para o DST-VII, em comparação com a CPU. Para a solução combinada Vertical + Horizontal, a GPU foi pelo menos 10 vezes mais eficiente. Os autores concluíram que os resultados indicam um desempenho significativamente superior ao migrar os algoritmos de uma arquitetura baseada em CPU para GPU, com um aumento de até 11 vezes na performance.

O trabalho de Abdoli; Henry; Brault; Dufaux; Duhamel (2019) propõe uma abordagem de codificação de coeficientes de transformada para blocos 4×4 no VVC, visando melhorar a compressão de conteúdo de tela, um tipo de conteúdo cada vez mais comum. Devido ao alto nível de detalhe das texturas presentes em blocos 4×4, a codificação desses blocos requer uma atenção especial. Os autores introduziram o algoritmo *Unary Bitplane Coding* (UBC), que usa amplitudes binarizadas e códigos unários para representar cada bloco com planos de bits. Essa representação permite explorar informações contextuais dos blocos codificados e agrupá-las conforme suas características estatísticas durante a codificação de entropia. Para lidar com o grande número de situações contextuais, foi aplicado o algoritmo *K-Means* para categorizar os *bins* de acordo com seu comportamento estatístico. Os resultados experimentais mostraram que, ao substituir a codificação de transformadas convencional pelo UBC, foram obtidos ganhos de 2,8% em BD-BR no modo de acesso aleatório e 3,4% em todos os modos intraquadro.

O trabalho de Nguyen; Bross; Schwarz; Marpe; Wiegand (2020) apresenta um esquema de codificação residual dedicado para blocos de transformadas codificados em TSM, uma abordagem promissora para melhorar a eficiência de codificação com impacto reduzido na complexidade. Em vez de modificar a codificação residual regular (RRC), os autores propuseram o *Transform Skip Residual Coding* (TSRC), que mantém conceitos essenciais do RRC, alterando apenas aspectos que melhoram a eficiência de compressão. O TSRC foi implementado na versão 3 do *software* de referência VTM, avaliando a eficiência de codificação com condições de teste JVET. Os resultados experimentais, com diferentes valores do parâmetro de quantização (QP), mostraram melhorias na eficiência de compressão, com ganhos de cerca de 9% em uma configuração *all-intra* e 5% em uma configuração de *random access* para conteúdo de tela. Esses resultados indicam que a reutilização da arquitetura da codificação residual, com modificações na ordem de codificação e modelagem de contexto, pode melhorar significativamente a eficiência da compressão.

O trabalho de Jdidia; Amor; Belghith; Masmoudi (2020) avalia uma aproximação da DST-VII como solução eficiente para reduzir a contagem de operações e o tempo de execução, mantendo um desempenho de codificação similar ao da transformada

exata. Isso se deve ao fato de que a DST-VII envolve multiplicação de matrizes complexas. A avaliação da qualidade do vídeo foi feita usando *Peak Signal to Noise Ratio* (PSNR) e *Global Score Reduced Reference Video Quality Assessment based on Human Visual System* (GSRV H). A matriz de transformada proposta foi integrada ao algoritmo do *software* de referência 7.1 e comparada com o original em termos de PSNR e GSRV H. Os experimentos foram realizados com sequências de vídeo da classe A (resolução 3840×2160), para os QPs 22, 27, 32 e 37, nas configurações *All Intra* (AI) e *Random Access*. Os resultados mostraram que o algoritmo proposto reduziu a complexidade computacional e melhorou tanto a qualidade objetiva quanto a perceptiva do vídeo em comparação com o algoritmo original.

Com o objetivo de economizar tempo de codificação e tornar o VVC mais adequado para aplicações em tempo real, o trabalho de He; Xiong; Yang; He; Chen (2022a) propôs uma seleção de transformada múltipla de baixa complexidade combinada com o algoritmo de segmentação de árvore *multi-type* para otimizar o VVC para aplicações em tempo real, visando economizar tempo de codificação. A proposta baseia-se na similaridade entre *sub-Coding Units* (CUs) e na correlação entre o custo *Rate-Distortion* (RD) da transformada primária e da MTS. Foi desenvolvido um método para estimar o custo RD da última *child* CU, evitando a verificação de RD da MTS quando a soma dos custos de RD das *child* CUs fosse maior ou igual ao custo da *parent* CU. Além disso, um método de terminação antecipada da MTS foi proposto para acelerar ainda mais a codificação, utilizando informações das CUs vizinhas. Os experimentos mostraram que o algoritmo reduziu o tempo de codificação em 26,4%, mantendo um desempenho similar (aumento de 0,13% no BD-BR), evidenciando uma menor complexidade computacional, adequada para aplicações em tempo real. O algoritmo pode ser combinado com outros métodos para otimizar ainda mais o processo de codificação.

O artigo de Liu; Shi; Li; Zhang; Ming; Fang (2021) propõe um novo método de ocultação de informações para fluxos de vídeo compactados no VVC, utilizando blocos de luminância e crominância da predição intraquadro. O método explora as ferramentas MTS e o *Cross-component linear model* (CCLM) para preservar a qualidade de reconstrução, eficiência de compactação e melhorar a capacidade de ocultação. O processo é dividido em duas fases: ocultação e extração de informações. Os autores avaliaram o desempenho com quatro sequências de vídeo de diferentes resoluções e complexidades de textura, comparando com três métodos de ocultação projetados para o HEVC. Os resultados indicaram que o método proposto oferece alta capacidade de ocultação, boa qualidade de vídeo e impacto mínimo na eficiência de compactação, superando os métodos existentes em termos de capacidade, taxa de bits e PSNR.

O trabalho de Chan; Im (2021) propõe a construção de transformadas ortogonais discretas baseadas na *Discrete Tchebichef Transform* (DTT) para atender às necessidades de codificação do VVC. A DTT, que utiliza polinômios ortogonais de *Tchebichef*,

é aplicada na compressão de imagens e codificação de vídeo. Os autores desenvolveram um novo método para gerar matrizes ortogonais discretas e aplicaram-nas em 64 primeiros quadros de sequências de vídeo com diferentes resoluções usando o VTM. Os testes mostraram que a DTT proposta melhorou a eficiência de codificação, reduzindo a taxa de bits e melhorando a qualidade do vídeo sem aumentar o tempo de codificação. A abordagem oferece uma solução prática para reduzir a complexidade computacional e o consumo de energia, substituindo a DCT linear no VVC.

O artigo de Egilmez; Singh; Coban; Karczewicz; Zhu; Yang; Said; Cohen (2021) explora soluções de redes de transformadas para suportar formatos de cor subamostrados em arquiteturas de codificação de imagem/vídeo baseadas em aprendizagem profunda (*Deep Learning based End-to-end image/video Coding* – DLEC), focando no formato YUV 4:2:0. Os autores propõem duas abordagens: a primeira modifica a estrutura dos canais de entrada e saída com um método predefinido de divisão, adaptando redes DLEC existentes para dados YUV 4:2:0; a segunda constrói redes específicas que operam diretamente em dados YUV 4:2:0 sem alteração nos canais. Foram realizados testes com dados YUV 4:2:0 convertidos de vídeos RGB, e os modelos treinados mostraram um desempenho superior em comparação com HEVC, atingindo uma melhoria de 10% na BD-BR, embora ainda 16% menos eficientes que o VVC.

Observa-se que diversas abordagens têm sido exploradas para melhorar a eficiência das transformadas no VVC, focando em reduzir a complexidade computacional. Uma dessas abordagens envolve a modificação de esquemas de codificação residual, com o objetivo de manter a eficiência de compressão, enquanto minimiza o impacto na implementação. Técnicas de aproximação de transformadas, como a substituição de matrizes complexas por versões de baixa complexidade Hamidouche; Philippe; Mohamed; Kammoun; Menard; Déforges (2019), têm-se mostrado eficazes em reduzir operações aritméticas, mantendo a qualidade visual e a eficiência da codificação. Outras soluções combinam a seleção de transformadas múltiplas e métodos de segmentação para otimizar o processo de codificação He; Xiong; Yang; He; Chen (2022a), resultando em uma redução significativa no tempo de codificação, sem sacrificar a qualidade do vídeo. A integração de técnicas de ocultação de informações também tem sido uma área de interesse, com métodos que preservam a qualidade do vídeo e a eficiência de compressão, ao mesmo tempo em que aumentam a capacidade de ocultação Liu; Shi; Li; Zhang; Ming; Fang (2021). Além disso, transformadas discretas de polinômios ortogonais Chan; Im (2021) têm sido usadas para substituir transformadas tradicionais, proporcionando uma redução no consumo de energia e na complexidade computacional. Soluções baseadas em *deep learning* também estão sendo exploradas, especialmente para formatos YUV 4:2:0 Egilmez; Singh; Coban; Karczewicz; Zhu; Yang; Said; Cohen (2021), onde novas arquiteturas de rede são projetadas para operar diretamente com esses dados, melhorando a eficiência de co-

dificação em comparação com métodos baseados em formatos RGB. Essas técnicas, em conjunto, visam otimizar o VVC para aplicações em tempo real, proporcionando melhores desempenhos de codificação e menores custos computacionais.

3.3 Resumo do Capítulo

Este capítulo apresentou uma revisão das técnicas utilizadas na implementação de transformadas no VVC, abrangendo soluções tanto em hardware quanto em software. Os trabalhos analisados abordam diversas abordagens para implementar transformadas como DCT-II, DCT-VIII, DST-VII e LFNST, aplicáveis a blocos de tamanhos variados, de 4×4 a 64×64. Muitas dessas soluções focam na otimização para FPGAs, utilizando técnicas como a substituição de multiplicadores por somas e deslocamentos, pipeline e memórias de transposição, com o objetivo de reduzir a complexidade computacional e melhorar a eficiência de área e consumo de energia. Algumas estratégias, como a decomposição de matrizes, são utilizadas para reduzir a área, enquanto outras exploram a reutilização de hardware para diferentes transformadas, aproveitando a similaridade entre os coeficientes das matrizes. Além disso, várias abordagens são projetadas para suportar vídeos 2K e 4K, com alta taxa de quadros, atendendo aos requisitos de padrões como MPEG, AVC, HEVC e VVC.

No campo de software, diversas estratégias foram exploradas para melhorar a eficiência do VVC, incluindo modificações nos esquemas de codificação residual e o uso de técnicas de aproximação de transformadas, como a substituição de matrizes complexas por versões de baixa complexidade, visando reduzir operações aritméticas sem comprometer a qualidade visual. Outras soluções combinam a seleção de transformadas múltiplas e métodos de segmentação para otimizar a codificação, resultando em menores tempos de codificação sem perda de qualidade. Técnicas de ocultação de informações também têm sido investigadas, permitindo melhorar a capacidade de ocultação sem afetar a qualidade do vídeo ou a eficiência de compressão. A utilização de transformadas discretas de polinômios ortogonais tem proporcionado uma redução no consumo de energia e na complexidade computacional, enquanto soluções baseadas em *deep learning*, especialmente para o formato YUV 4:2:0, têm mostrado melhorias na eficiência de codificação. Essas técnicas visam otimizar o VVC para aplicações em tempo real, resultando em melhor desempenho e redução no custo computacional.

Apesar desses avanços, ainda existem lacunas relevantes que precisam ser exploradas, sobretudo em relação à eficiência energética. No hardware, estratégias como *data gating*, *clock gating* e domínios de *clock* independentes são pouco aplicadas, mesmo com seu potencial para redução de potência. No software, observa-se uma ausência de mecanismos de decisão baseados em aprendizado de máquina que possam adaptar dinamicamente a complexidade da MTS conforme o conteúdo codi-

ficado. Diante disso, o uso de técnicas de aprendizado de máquina desponta como uma oportunidade promissora para apoiar a tomada de decisão no módulo de transformadas. Essa abordagem pode ser integrada ao projeto de arquiteturas de hardware, resultando em soluções mais adaptativas e energeticamente eficientes, conectando de forma inovadora os domínios de software e hardware no contexto do padrão VVC.

4 ANÁLISE DE COMPLEXIDADE E USABILIDADE DA MTS

Como previamente mencionado, a técnica de *Multiple Transform Selection* introduzida no novo padrão de codificação de vídeo VVC representa um aumento significativo na complexidade computacional, em função da incorporação de múltiplos tipos de transformadas. Essa complexidade é alta devido à necessidade de avaliar diversas combinações dessas transformadas durante o processo de codificação, o que resulta em um alto custo computacional. Para avaliar o impacto dessa técnica, foram realizadas algumas análises utilizando o *software* de referência *VVC Test Model* (Fraunhofer, 2020). O objetivo da MTS é melhorar o desempenho na etapa de transformadas, permitindo a escolha de diferentes transformadas para cada dimensão do bloco residual. Este estudo também investiga como os modos explícito e implícito de MTS afetam a eficiência de compressão e identificará qual modo apresenta maior consumo computacional, oferecendo subsídio para possíveis otimizações no processo de codificação.

Para as análises em ambiente de *software*, foi utilizado o *VVC Test Model* (VTM), nas versões 10.0 (Fraunhofer, 2020) e 23.0 (Jvet, 2024). O VTM é o *software* de referência oficial do padrão VVC, desenvolvido para pesquisa e validação de técnicas de codificação. Ele foi empregado neste trabalho com o objetivo de compreender e documentar a implementação das transformadas utilizadas no processo de compressão de vídeo, além de permitir a realização dos experimentos.

O impacto das modificações no processo de codificação foi avaliado utilizando métricas quantitativas, como o tempo de execução do codificador e a métrica *Bjontegaard-Delta-Rate* (BD-BR) (Bjontegaard, 2001). O BD-BR quantifica a diferença na taxa de bits necessária para alcançar a mesma qualidade visual entre dois métodos de codificação, sendo utilizado para medir a eficiência da compressão. Quanto maior o valor do BD-BR, menor a eficiência, indicando aumento no volume de dados transmitidos para a mesma qualidade.

Além do BD-BR, algumas análises na literatura utilizam o PSNR (*Peak Signal-to-Noise Ratio*) como métrica complementar (Winkler, 2005). O PSNR mede a qualidade de reconstrução dos vídeos comprimidos em relação aos originais, baseando-se na diferença entre os *pixels* correspondentes. Valores mais altos de PSNR estão as-

sociados a maior fidelidade visual. As simulações foram conduzidas com diferentes valores do parâmetro de quantização (QP), especificamente QP = 22, 27, 32, 37, como especificado na Seção 2.2. O QP controla o grau de compressão: valores menores resultam em maior qualidade de vídeo e menor compressão, enquanto valores maiores priorizam a compressão em detrimento da qualidade.

Todas as análises desta tese seguiram o conjunto de condições de teste padronizadas *Common Test Conditions* (CTC) (Boyce; Suehring; Li, 2018), que define parâmetros e conjuntos de sequências de vídeo para garantir a comparação consistente entre diferentes estudos. Em casos específicos, no entanto, nem todas as sequências do CTC foram utilizadas. Nessas situações, a seleção de vídeos baseou-se nas métricas de Informação Espacial (SI) e Informação Temporal (TI), que classificam as sequências conforme seu grau de detalhamento espacial e variação temporal. O SI avalia a quantidade de detalhes presentes em uma imagem, refletindo a complexidade visual da cena, enquanto o TI mede a variação entre quadros consecutivos, indicando a quantidade de movimento (Itu, 2012).

4.1 Análise do Impacto da MTS na Codificação de Vídeo

A primeira etapa deste estudo teve como objetivo analisar o impacto da ferramenta *Multiple Transform Selection* no processo de codificação de vídeo segundo o padrão VVC. Para isso, foi realizada uma série de experimentos no *software* de referência do padrão, o *Versatile Video Coding Test Model*, versão 10.0.

Durante a análise preliminar do codificador, foi identificada a presença de uma *flag* responsável pela habilitação da MTS. Quando desabilitada, esta *flag* impede a execução dos testes de combinações de transformadas previstas no padrão, restringindo o processo de codificação à aplicação da transformada DCT-II em ambas as dimensões, comportamento semelhante ao adotado no padrão HEVC. Por outro lado, quando habilitada, a MTS realiza a avaliação de múltiplas combinações de transformadas, selecionando aquela que proporciona melhor desempenho em termos de taxa-distorção.

O estudo inicial foi conduzido com o propósito de quantificar o impacto dessa funcionalidade no desempenho do codificador. Para isso, foram selecionadas cinco sequências de vídeo recomendadas nas CTC do VVC (conforme descrito na Fundamentação Teórica). A escolha das sequências considerou os índices de informações espaciais (SI) e temporais (TI), conforme discutido na Seção 2.5, com o objetivo de incluir vídeos que apresentassem ampla diversidade nesses parâmetros. Essa abordagem buscou garantir que o conjunto de testes fosse o mais representativo e abrangente possível.

A Figura 14 apresenta o gráfico com a distribuição das sequências do conjunto CTC em termos de SI e TI, evidenciando a variedade de conteúdos considerados. As

sequências selecionadas e suas principais características encontram-se resumidas na Tabela 5.

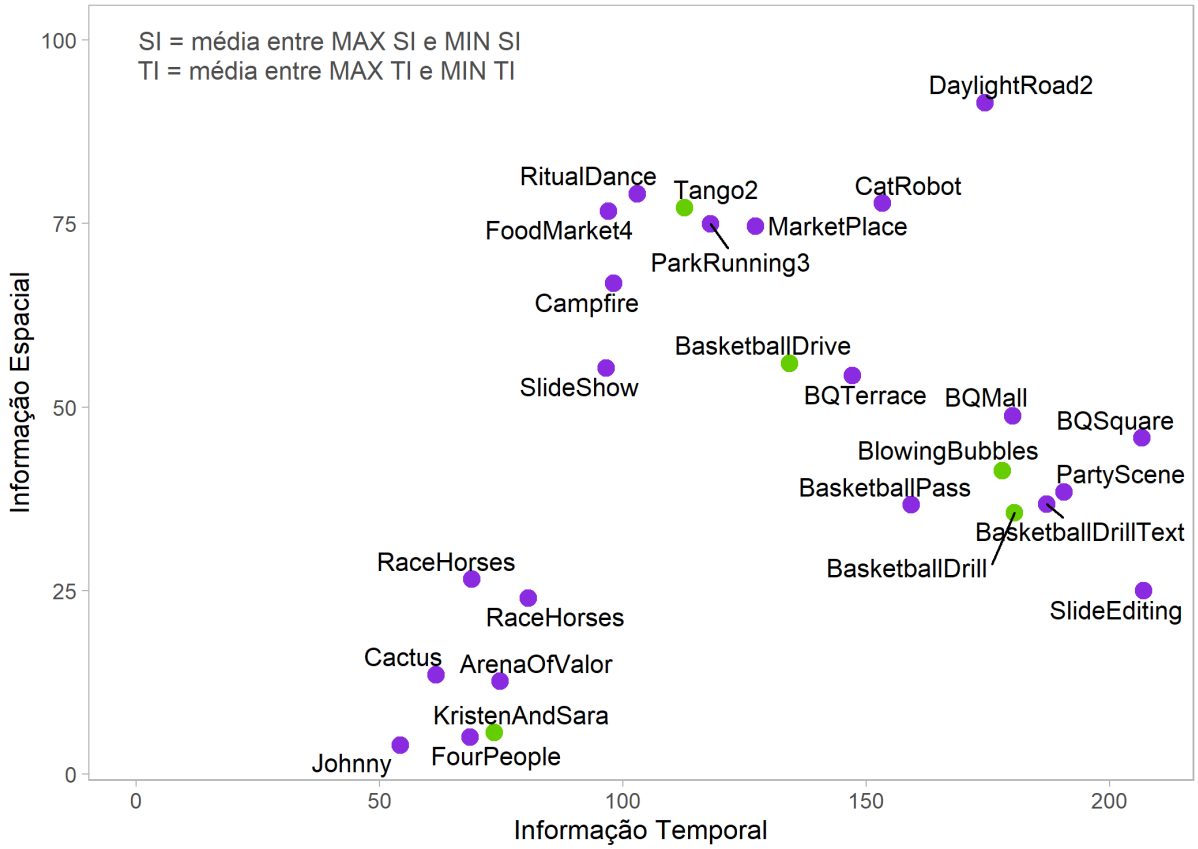

Figura 14 – Índices de informação espacial e temporal das sequências de vídeos.

Foi utilizada a configuração do SPS *All-Intra* e com a *flag TemporalSubsampleRatio* igual a oito. Esta *flag* indica que um quadro é codificado a cada oito quadros, o que é empregado para acelerar o experimento e para coletar dados de quadros mais distantes temporalmente, visto que quadros vizinhos tendem a apresentar informações muito semelhantes.

Tabela 5 – Sequências de vídeos de teste

Vídeo	SI	TI	Resolução	Quadros
BlowingBubbles	178,03	41,31	416x240	5
BasketballDrill	180,54	35,56	832x480	5
KristenAndSara	73,61	5,61	1280x720	5
BasketballDrive	134,24	55,95	1920x1080	5
Tango2	112,75	77,18	4096x2160	5

4.1.1 Resultados Obtidos

Como o principal objetivo desta análise foi avaliar o impacto da habilitação da ferramenta MTS no desempenho do codificador, foram consideradas duas métricas fundamentais: o tempo de codificação e a eficiência de codificação, quantificada pela métrica BD-BR. As avaliações foram realizadas comparando dois cenários distintos: com a MTS habilitada e com a MTS desabilitada.

Para garantir a representatividade e a comparabilidade dos resultados, os testes foram conduzidos utilizando quatro valores do parâmetro de quantização (QP = 22, 27, 32 e 37), conforme as recomendações estabelecidas pelas CTC. Para cada valor de QP e para cada sequência de vídeo, a *flag* de habilitação da MTS foi alternada entre ativada e desativada, permitindo a coleta sistemática dos resultados nas duas configurações.

A Figura 15 apresenta o tempo médio de codificação obtido para cada sequência de vídeo testada, enquanto a Tabela 6 resume os valores percentuais de BD-BR e a redução do tempo de codificação quando a MTS é desabilitada em comparação à sua ativação.

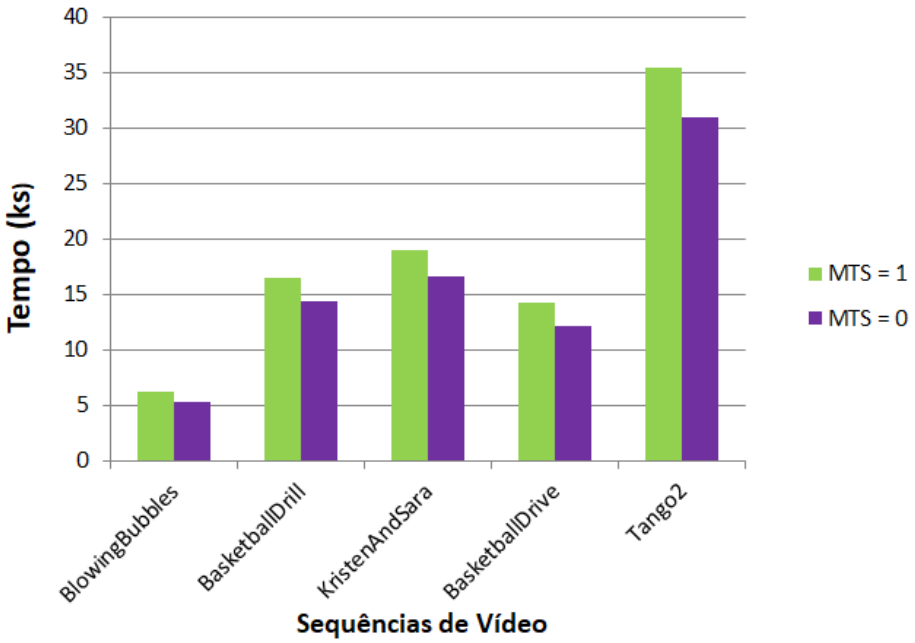

Figura 15 – Resultado de tempo de execução.

Os resultados obtidos evidenciam que a desativação da MTS proporciona uma redução média de 12,5% no tempo total de codificação, indicando um impacto significativo no custo computacional do codificador. Entretanto, essa simplificação vem acompanhada de um aumento médio de 1,06% no BD-BR, o que implica uma diminuição na eficiência de codificação.

Esse aumento médio de 1,06% no BD-BR implica uma necessidade proporcional de aumento na taxa de bits para manter a qualidade do vídeo, contrariando os princí-

Tabela 6 – Variação de BD-BR e redução de tempo com MTS desabilitada

Sequência de Vídeo	BD-BR (%)	T (%)
BlowingBubbles 240p	0,827	14,52
BasketballDrill 480p	0,778	13,09
KristenAndSara 720p	1,494	11,69
BasketballDrive 1080p	0,902	12,69
Tango2 4k	1,316	10,73
Média	**1,0634**	**12,544**

pios da compressão eficiente, sobretudo em aplicações que exigem alta qualidade e largura de banda restrita.

Além disso, a MTS é uma inovação central no VVC, oferecendo flexibilidade na seleção de transformadas que melhor se adaptam às características espaciais e temporais dos blocos de vídeo. Sua desativação comprometeria essa adaptabilidade, impactando negativamente o desempenho em diversos conteúdos. Os resultados, portanto, confirmam que, ainda que eleve a complexidade computacional, a MTS é fundamental para manter a eficiência de compressão. Descartá-la não é uma solução viável, pois acarretaria perdas significativas.

4.2 Análise das Escolhas da MTS

Após a avaliação do impacto do uso da MTS em termos de complexidade computacional e eficiência de codificação, foi conduzida uma análise complementar com o objetivo de identificar as combinações de transformadas mais frequentemente selecionadas pelo codificador. Esta investigação é essencial para compreender o comportamento interno do processo de seleção da MTS e identificar tendências de escolha que possam orientar otimizações futuras.

Como apresentado anteriormente no capítulo 2, quando a MTS está habilitada, um conjunto de transformadas pode ser empregado para a execução da etapa. No *software* de referência é possível encontrar um parâmetro chamado MTS-IDX (MTS *Index*) que determina qual destes conjuntos de transformadas será aplicado. Na Tabela 7 podemos ver o efeito da configuração deste parâmetro em termos de quais transformadas são habilitadas em cada orientação do bloco.

Se o MTS-IDX for igual a 0, o conjunto de transformadas escolhidas será a DCT-II aplicada na horizontal e vertical. Caso seja igual a 1, será aplicado o modo SKIP (nenhuma transformada) e assim sucessivamente para os outros índices. Para a extração das informações de quais combinações de transformadas foram escolhidas, foi

Tabela 7 – Índice da MTS

MTS-IDX	Horizontal	Vertical
0	DCT-II	DCT-II
1	SKIP	
2	DST-VII	DST-VII
3	DCT-VIII	DST-VII
4	DST-VII	DCT-VIII
5	DCT-VIII	DCT-VIII

necessária uma manipulação do *software* de referência VTM 10.0. Primeiro, foi localizado onde são escolhidas as transformadas para os diferentes tamanhos de blocos. Após, foram introduzidos contadores para calcular o número de ocorrências deste índice. Portanto, algumas condições tiveram que ser consideradas. Por exemplo, como a MTS é aplicada apenas para as componentes de luminância do quadro, a contagem dos índices da MTS foi desprezada para as componentes de crominância, já que o índice escolhido seria sempre igual a 0 (DCT-II aplicada na horizontal e na vertical). Uma outra condição foi a contagem das transformadas apenas quando o parâmetro ISP é igual a 0, pois, como apresentado anteriormente, quando este parâmetro é habilitado, a MTS é automaticamente desabilitada.

4.2.1 Resultados Obtidos

A análise foi restrita a blocos de tamanhos quadrados, uma vez que esta etapa inicial da pesquisa teve caráter exploratório, visando uma primeira compreensão do comportamento das técnicas avaliadas sem a complexidade adicional imposta pelos blocos retangulares. Os resultados são apresentados nos gráficos das Figuras 16, 17, 18 e 19. Para cada tamanho de bloco, foi quantificado o número de vezes em que cada conjunto de transformadas foi selecionado pelo *software* de referência durante o processo de codificação. Com o intuito de permitir uma comparação equitativa entre os diferentes tamanhos de blocos (considerando que blocos maiores, por conterem mais *pixels*, possuem um número naturalmente maior de amostras) os valores foram normalizados. Essa normalização utilizou como referência a quantidade de amostras de um bloco 64×64 (4096 amostras), o maior tamanho permitido pelo padrão.

Observa-se que os resultados para blocos 64×64 não são apresentados graficamente, uma vez que, para esses blocos, o codificador aplica exclusivamente a combinação de transformadas MTS-IDX = 0 (combinação da DCT-II nas duas dimensões). As ocorrências correspondentes, entretanto, foram incluídas nos cálculos de taxa de seleção geral para manter a coerência estatística.

Adicionalmente, cabe destacar que a configuração MTS-IDX = 5 (DCT-VIII aplicada tanto na direção horizontal quanto na vertical), embora declarada como um possível valor de MTS-IDX no *software* de referência, não foi considerada na análise. Durante todos os experimentos, esta combinação não foi escolhida pelo codificador em nenhuma circunstância e, portanto, não gerou ocorrências contabilizáveis.

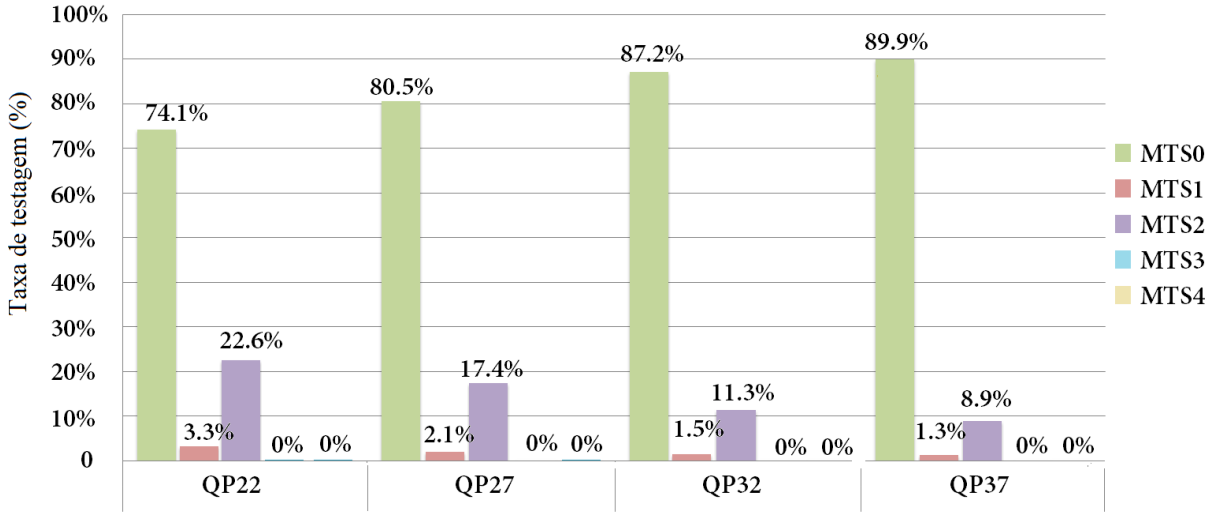

Figura 16 – Taxa de escolha de cada índice de MTS em blocos 4×4.

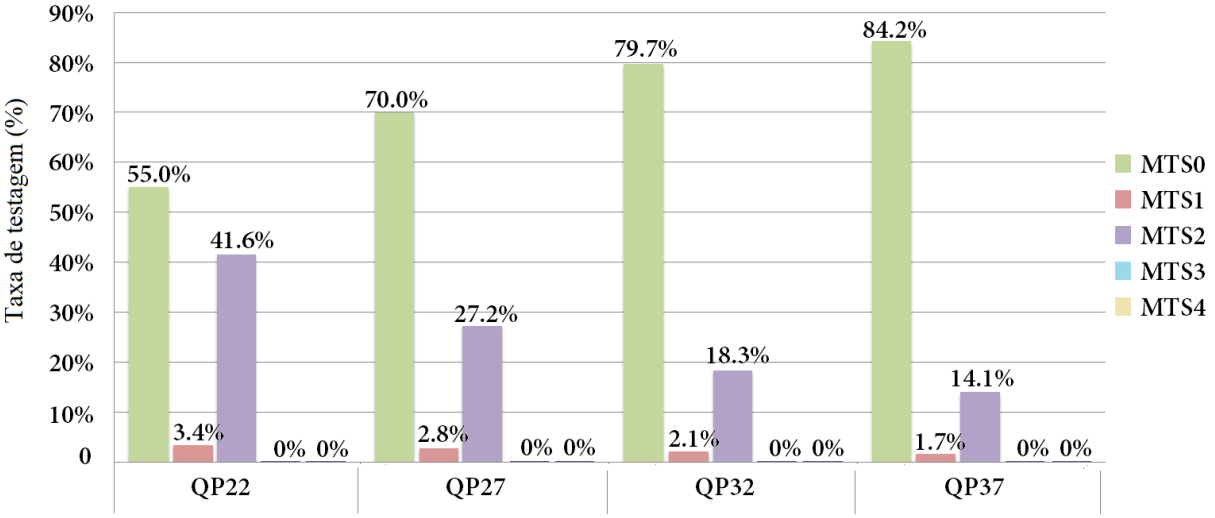

Figura 17 – Taxa de escolha de cada índice de MTS em blocos 8×8.

Os resultados obtidos indicam que a combinação de transformadas MTS-IDX = 0 (DCT-II aplicada nas direções horizontal e vertical) foi a mais frequentemente selecionada em quase todos os tamanhos de blocos e valores de QP. A única exceção observada ocorreu para blocos de tamanho 16×16 com QP igual a 22, onde a MTS-IDX = 2 apresentou maior representatividade.

As taxas de seleção da MTS-IDX = 0 foram de 69,7%, 80,4%, 84,9% e 86,2% para os QPs 22, 27, 32 e 37, respectivamente. Para blocos de 4×4, 8×8 e 16×16, observou-se uma tendência de aumento na utilização da MTS-IDX = 0 à medida que o QP se

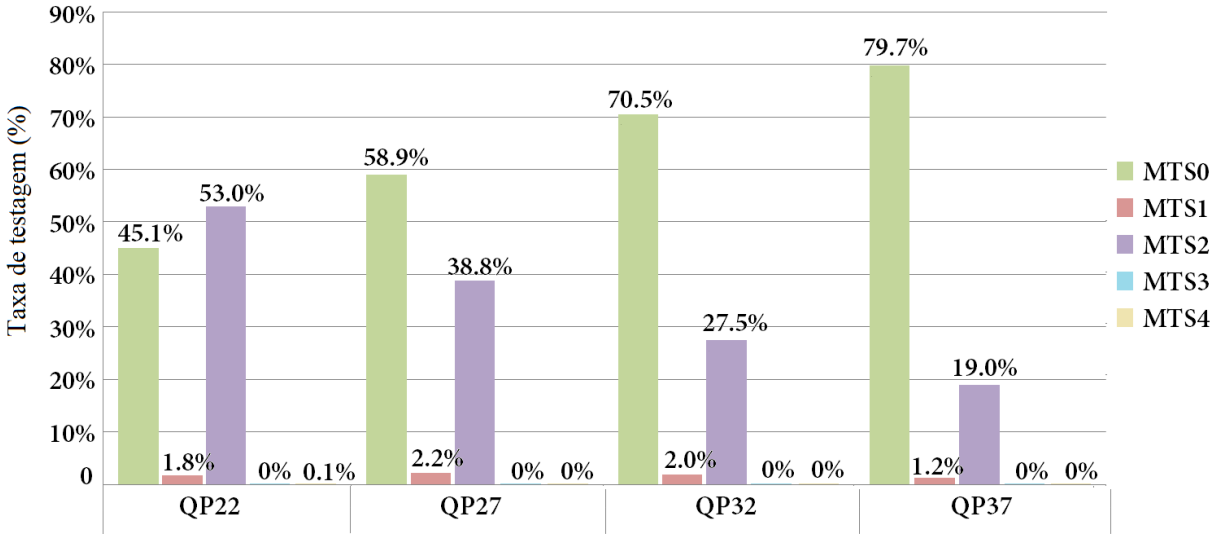

Figura 18 – Taxa de escolha de cada índice de MTS em blocos 16×16.

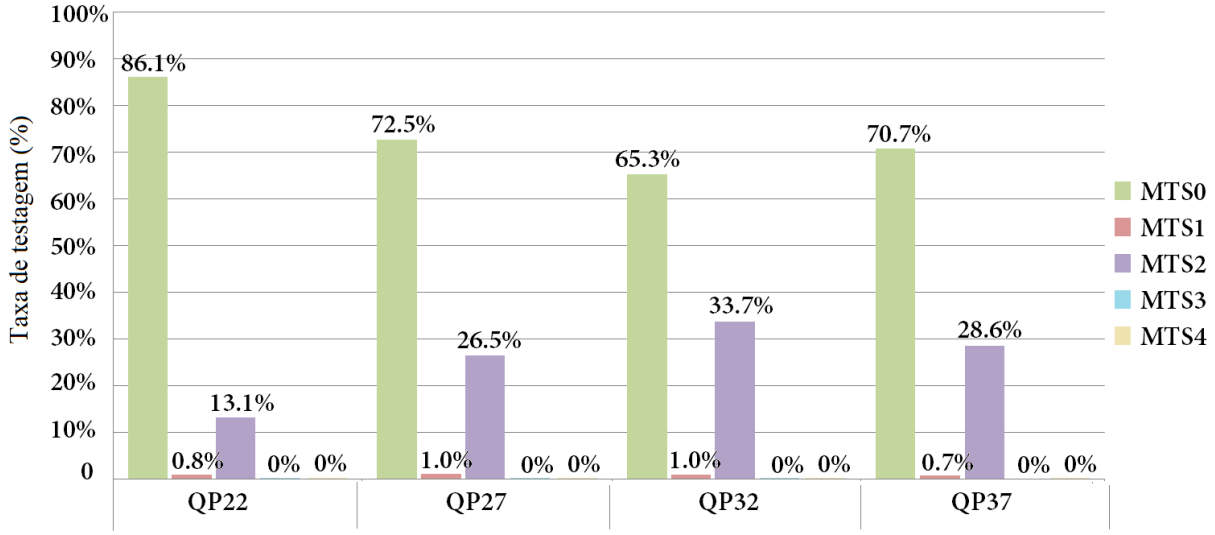

Figura 19 – Taxa de escolha de cada índice de MTS em blocos 32×32.

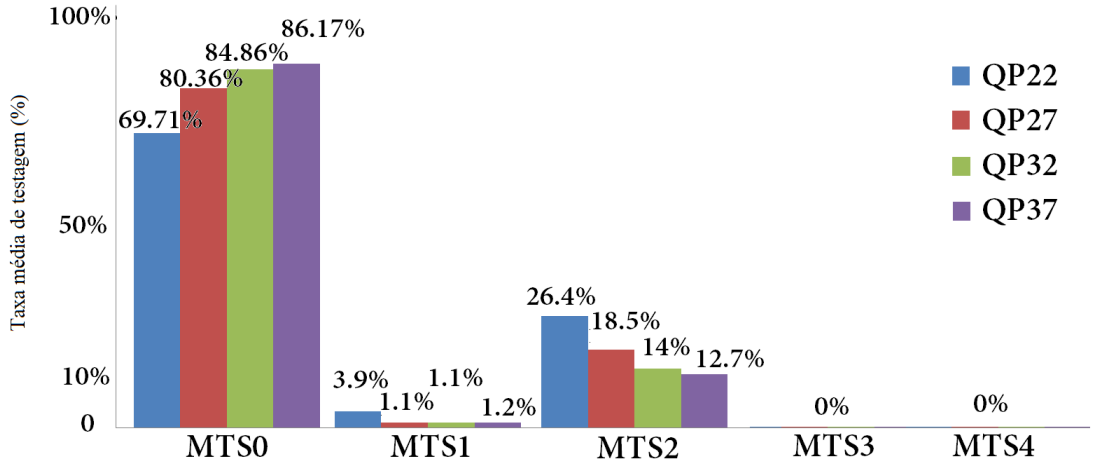

Figura 20 – Taxa média de escolha da MTS.

eleva. Por outro lado, para esses mesmos tamanhos de blocos, a seleção da MTS-IDX = 2 foi mais frequente nos menores valores de QP, indicando que transformadas mais complexas tendem a ser escolhidas quando o codificador prioriza qualidade sobre taxa de compressão.

A Figura 20 apresenta a taxa média de seleção de cada grupo de transformadas nas cinco sequências de vídeo analisadas. Esta representação permite visualizar a predominância relativa das combinações em diferentes cenários de codificação.

Os dados desta análise revelam padrões claros na escolha das transformadas, refletindo as preferências do codificador em função do tamanho do bloco e da configuração de QP. Embora esta etapa não tenha o objetivo de propor alterações no fluxo de execução, os resultados fornecem uma base importante para compreender o comportamento da MTS e identificar combinações potencialmente prioritárias. A identificação desses padrões é particularmente relevante porque evidencia que, apesar da grande quantidade de combinações possíveis, o codificador tende a selecionar com maior frequência um subconjunto restrito de transformadas. Essa constatação sugere que a etapa de testagem de combinações, responsável por uma parcela significativa do custo computacional do VVC, pode ser otimizada com estratégias que priorizem as combinações mais utilizadas ou eliminem testagens de combinações raramente escolhidas. Esse entendimento fundamenta as etapas seguintes do estudo, que buscarão propor mecanismos de redução de complexidade baseados nos padrões de uso observados.

4.3 Análise do Impacto dos Diferentes Modos Explícito e Implícito

Neste estudo, foi investigado o impacto dos diferentes modos de operação da MTS (*MTS-MODO*, conforme definido na Tabela 1, na eficiência de compressão do padrão VVC. Para isso, foram avaliadas diversas configurações de codificação, abrangendo os

seguintes modos: *MTS-MODO* = 1 (Intra explícito), 2 (Intra implícito e Inter explícito), 3 (Intra explícito e Inter explícito) e 4 (Intra implícito). Os experimentos foram realizados utilizando o *software* de referência VTM, versão 23.0 (Jvet, 2024), nas configurações *Random Access* e *All Intra*, aplicadas aos 60 primeiros quadros de todas as sequências recomendadas pela CTC. Essas sequências foram escolhidas por representarem uma ampla variedade de conteúdos, com diferentes níveis de textura e movimento, o que possibilita uma avaliação abrangente do desempenho do codificador sob diversas condições de codificação. As características detalhadas das sequências utilizadas encontram-se na Tabela 8.

Tabela 8 – Sequência de vídeos

Classe	Sequência	Resolução	SI	TI
A1	Tango2	4096x2160	112.75	77.18
A1	FoodMarket4	3840x2160	97.07	76.67
A1	Campfire	3840x2160	98.14	66.84
A2	CatRobot	3840x2160	153.37	77.81
A2	DaylightRoad2	3840x2160	174.44	91.47
A2	ParkRunning3	3840x2160	117.99	74.93
B	MarketPlace	1920x1080	127.30	74.67
B	RitualDance	1920x1080	102.94	79.08
B	Cactus	1920x1080	61.67	13.52
B	BasketballDrive	1920x1080	134.24	55.95
B	BQTerrace	1920x1080	147.15	54.26
C	RaceHorsesC	832x480	69.01	26.52
C	BQMall	832x480	180.18	48.78
C	PartyScene	832x480	190.72	38.44
C	BasketballDrill	832x480	180.54	35.56
D	RaceHorses	416x240	80.61	23.98
D	BQSquare	416x240	206.69	45.81
D	BlowingBubbles	416x240	178.03	41.31
D	BasketballPass	416x240	159.32	36.65
E	FourPeople	1280x720	68.56	5.01
E	Johnny	1280x720	54.31	3.94
E	KristenAndSara	1280x720	73.61	5.61
F	ArenaOfValor	1920x1080	74.72	12.64
F	BasketballDrillText	832x480	187.09	36.77
F	SlideEditing	1280x720	206.99	24.98
F	SlideShow	1280x720	96.59	55.34

4.3.1 Resultados Obtidos para *Random Access*

Os resultados obtidos nos experimentos estão apresentados na Tabela 9, onde se comparam a eficiência e o tempo de codificação dos diferentes modos de MTS-MODO em relação à codificação de referência. Para este comparativo, a codificação de referência foi o modo MTS igual a 0, ou seja, quando o MTS está desativado. As colunas BD-BR e tempo foram organizadas para refletir os resultados dos modos de MTS analisados. A fim de facilitar a visualização, os vídeos foram agrupados de acordo com sua classe, calculando-se as médias de BD-BR e tempo para cada grupo.

Tabela 9 – Resultados de eficiência de codificação e tempo para configuração *Random Access*

	MTS-MODO=1		**MTS-MODO=2**		**MTS-MODO=3**		**MTS-MODO=4**	
Classe	**BD-BR (%)**	**Tempo (%)**	**BD-BR (%)**	**Tempo (%)**	**BD-BR (%)**	**Tempo (%)**	**BD-BR (%)**	**Tempo (%)**
A1	-0,58	3,60	-0,37	7,84	-0,59	12,57	-0,31	1,76
A2	-0,89	7,24	-0,95	8,09	-1,15	15,09	-0,63	0,82
B	-0,69	8,06	-0,76	6,20	-1,02	13,73	-0,67	0,46
C	-0,53	11,14	-0,46	6,87	-0,65	17,37	-0,36	0,50
D	-0,52	10,03	-0,48	6,06	-0,48	15,16	-0,39	0,46
E	-1,21	5,50	-0,89	1,39	-1,23	7,83	-0,95	-0,23
F	-0,27	9,28	0,13	3,86	-0,30	12,11	0,13	0,73
Média	**-0,71**	**7,84**	**-0,54**	**5,76**	**-0,77**	**13,41**	**-0,45**	**0,64**

A métrica BD-BR se destaca nesta análise, pois fornece uma avaliação quantitativa da eficiência de codificação (Bjøntegaard, 2001). Através da comparação das variações no BD-BR entre os diferentes modos de MTS, é possível identificar quais oferecem melhorias relevantes na compressão, quando comparados à codificação de referência. Além disso, a análise do tempo de codificação é essencial para avaliar o custo computacional relacionado à implementação de cada MTS-MODO.

Os resultados observados nos experimentos confirmam a expectativa de que os modos de MTS promovem melhorias na eficiência de codificação, como evidenciado pela maioria dos valores de BD-BR negativos apresentados na Tabela 9. Esses resultados são consistentes com a teoria de que, ao ativar a MTS, o uso de múltiplas transformadas na codificação possibilita uma adaptação mais precisa às características específicas dos blocos de vídeo. Esse processo resulta em uma representação mais eficiente dos dados, o que leva a vídeos de melhor qualidade com uma taxa de bits reduzida, indicando uma melhoria significativa na compressão final.

Entre os modos testados, a MTS-MODO = 3 se destaca, apresentando a maior

redução no BD-BR, com uma média de -0,77%. Isso sugere que a combinação de transformadas explícitas na predição intra e inter é particularmente eficaz para melhorar a eficiência de compressão, proporcionando uma melhor qualidade de vídeo com uma taxa de bits mais baixa. No entanto, a implementação desse modo resultou em um aumento no tempo de codificação, com uma média de 13,41%. Esse dado ilustra o *trade-off* entre a eficiência de codificação e o tempo de processamento. Considerando que a MTS-MODO = 3 não apenas apresenta a maior eficiência de codificação, mas também o maior custo computacional, ela oferece uma oportunidade valiosa para otimizações no futuro, onde o esforço computacional adicional poderia ser explorado para reduzir o tempo de codificação.

Em contraste, a MTS implícita (MTS-MODO = 4) apresenta um custo computacional consideravelmente menor. Os resultados indicam que, ao utilizar a MTS implícita, o tempo de codificação aumentou apenas 0,64% em relação à codificação original, enquanto houve uma melhoria de -0,45% na eficiência de compressão. Este comportamento sugere que, embora o ganho em eficiência seja modesto, o custo computacional envolvido é significativamente menor. Por outro lado, quando a MTS implícita foi combinada com a MTS explícita (MTS-MODO = 2), o aumento no tempo de codificação foi considerável, atingindo 5,76%. Nesse cenário, o ganho em eficiência foi de apenas -0,09% em relação ao MTS-MODO = 4. Esses resultados indicam que a MTS explícita, particularmente na predição inter, é responsável pelo aumento substancial no custo computacional, sendo que os ganhos em BD-BR não justificam esse custo adicional.

Finalmente, os MTS-MODO = 1 e MTS-MODO = 2 emergem como opções equilibradas, proporcionando um compromisso eficaz entre a eficiência de compressão e o tempo de codificação. Esses modos intermediários são particularmente valiosos, pois destacam a importância de considerar os *trade-offs* entre a eficiência de codificação e os recursos computacionais necessários ao optar por diferentes configurações de MTS. A utilização desses modos permite uma abordagem flexível e eficiente, adaptável a uma ampla gama de cenários de codificação de vídeo, atendendo a diferentes requisitos de aplicação.

4.3.2 Resultados Obtidos para *All Intra*

Dando continuidade à avaliação da ferramenta MTS, foi realizada uma nova série de experimentos considerando a configuração *All Intra* do codificador de referência VTM. Essa configuração, por desabilitar a predição entre quadros (*inter-frame*), restringe a análise às estruturas puramente *intra-frame*. Dessa forma, os modos MTS-MODO = 2 (Intra implícito e Inter explícito) e MTS-MODO = 3 (Intra e Inter explícitos) não foram incluídos nesta etapa, uma vez que dependem da ativação da codificação *inter-frame* para funcionarem adequadamente. Cabe observar que, caso esses modos

fossem utilizados sob a configuração *All Intra*, o comportamento resultante seria funcionalmente equivalente aos MTS-MODO = 1 (Intra explícito) e MTS-MODO = 4 (Intra implícito), respectivamente. Isso se deve ao fato de que, na ausência da predição temporal, os elementos associados à codificação *inter-frame* deixam de influenciar o processo, tornando as distinções entre os modos irrelevantes. Dessa forma, restringir a análise aos modos compatíveis com a codificação *intra-frame* garante a validade e a consistência dos resultados obtidos. A Tabela 10 apresenta os valores obtidos para as métricas de eficiência de compressão (BD-BR) e tempo de codificação, permitindo avaliar o impacto da seleção de transformadas em um cenário sem predição temporal.

Tabela 10 – Resultados de eficiência de codificação e tempo para configuração *All Intra*

	MTS-MODO=1		**MTS-MODO=4**	
Classe	**BD-BR (%)**	**Tempo (%)**	**BD-BR (%)**	**Tempo (%)**
A1	-1,04	20,12	-0,60	1,71
A2	-1,34	19,07	-0,99	2,36
B	-1,40	19,34	-0,86	2,07
C	-0,88	18,85	-0,50	1,23
D	-0,82	18,93	-0,44	1,02
E	-1,47	17,14	-1,11	1,20
F	-0,49	15,33	-0,06	1,22
Média	**-1,07**	**18,40**	**-0,65**	**1,55**

Os resultados demonstram que a utilização da MTS em sua forma explícita (MTS-MODO = 1) proporcionou o maior ganho de compressão, com uma redução média de 1,07% no BD-BR. No entanto, esse ganho foi acompanhado por um acréscimo significativo no tempo de codificação, que atingiu 18,4% (o valor mais elevado entre todos os modos e configurações avaliados). Tal comportamento reforça o perfil da MTS explícita como uma alternativa mais indicada para aplicações em que a eficiência de compressão é priorizada, mesmo que isso implique em maior custo computacional.

Em contrapartida, o modo implícito de MTS (MTS-MODO = 4) apresentou um equilíbrio mais favorável entre complexidade e desempenho. Com um aumento moderado de 1,55% no tempo de codificação e uma redução de 0,65% no BD-BR, essa configuração se mostra uma alternativa viável para cenários com restrições de tempo de processamento, como em aplicações em tempo real ou em dispositivos com recursos computacionais limitados.

A configuração *All Intra* revelou comportamentos consistentes com os observados na configuração *Random Access*, embora com particularidades importantes. A ausência de predição temporal nesse modo permitiu uma exploração mais aprofundada

das transformadas disponíveis, o que resultou em uma utilização mais intensiva da ferramenta MTS. Como consequência, observou-se uma melhora mais acentuada na eficiência de compressão, refletida em reduções mais significativas no BD-BR quando comparadas à configuração *Random Access*.

Esses dados ressaltam a importância de selecionar o modo de operação da MTS conforme os requisitos específicos da aplicação. Enquanto o modo implícito favorece um compromisso entre eficiência de compressão e tempo de codificação, o modo explícito é mais eficaz na redução da taxa de bits, porém à custa de maior complexidade computacional. Essa dicotomia evidencia um dilema clássico de projeto entre desempenho e custo, apontando para a necessidade de estratégias que otimizem a MTS explícita. Melhorias nesse sentido podem viabilizar sua adoção em cenários práticos que demandem alta eficiência de compressão, mas operem sob restrições de tempo ou de processamento. As próximas seções aprofundam essa problemática por meio de uma análise focada no comportamento da MTS explícita.

4.4 Análise de Tempo de Execução da etapa de Transformadas no VVC

As análises apresentadas na Seção 4.3 consideraram métricas de eficiência de compressão (BD-BR) e tempo de codificação com base no impacto global da ativação da ferramenta MTS sobre o tempo total de codificação, isto é, contemplando todo o codificador. No entanto, tais medições não discriminam a fração de tempo especificamente atribuída à etapa de transformada, o que limita a compreensão sobre o papel individual dessa etapa no desempenho geral do sistema.

Nesta pesquisa, é essencial isolar e mensurar o tempo dedicado exclusivamente à etapa de transformadas, uma vez que isso permite avaliar com maior precisão seu impacto direto no desempenho da codificação. Com base nessa necessidade, esta nova análise foi desenvolvida com o objetivo de investigar o comportamento temporal da etapa de transformada no contexto da codificação.

A avaliação dessa etapa de forma isolada também possibilita estimar o limite superior de ganhos que poderiam ser obtidos caso técnicas de otimização fossem aplicadas exclusivamente sobre ela. Em termos práticos, quanto maior for sua participação relativa no tempo total de codificação, maior será o potencial benefício de estratégias de aceleração. No entanto, é importante ressaltar que, mesmo em cenários ideais de otimização integral da etapa de transformadas, os ganhos obtidos estarão limitados pela sua contribuição proporcional ao tempo de codificação como um todo, ou seja, há um limite intrínseco aos impactos possíveis.

Para viabilizar essa análise, o *software* de referência VTM 23.0 foi modificado de modo a registrar de forma precisa o tempo consumido pela etapa das transformadas.

Essa adaptação permitiu uma avaliação mais detalhada da sua relevância dentro do processo de codificação e de seu potencial de otimização. A fim de assegurar consistência metodológica com os experimentos anteriores, foram mantidas as mesmas sequências de vídeo e configurações utilizadas nas análises descritas na Seção 4.3.

4.4.1 Resultados para o tempo de execução das transformadas

Nos próximos parágrafos, são analisadas as proporções do tempo total de codificação especificamente atribuídas à etapa de transformadas, considerando separadamente as configurações *All Intra* e *Random Access*. Os resultados obtidos estão expressos em termos percentuais e organizados de acordo com os diferentes valores do parâmetro de quantização (QP), permitindo observar como essa etapa se comporta em função da qualidade de codificação exigida. Os gráficos das Figuras 21 e 22 apresentam os resultados para as duas configurações.

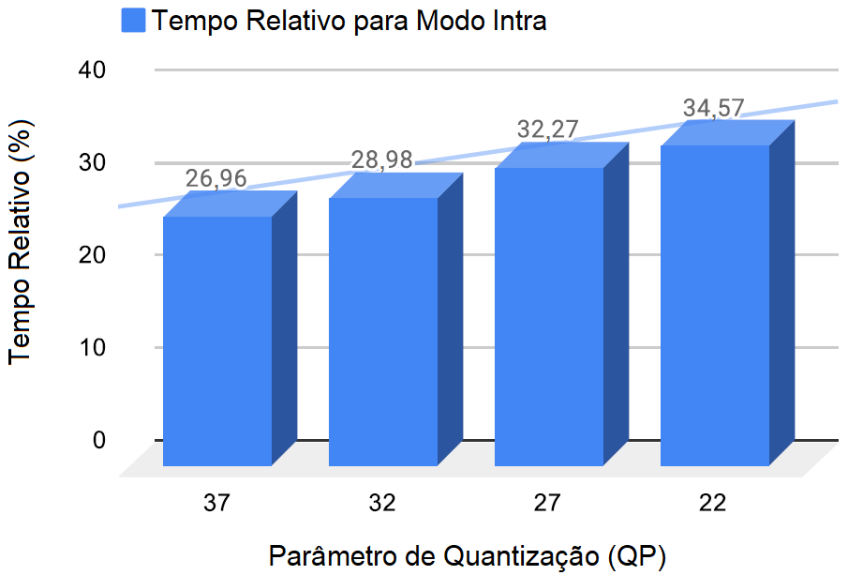

Figura 21 – Tempo de execução da etapa das transformadas para a configuração *All Intra*.

Na configuração *All Intra*, os dados revelam que a etapa de transformadas representa uma parcela significativa do tempo total de codificação. Como ilustrado na Figura 21, para QP = 37, aproximadamente 26,96% do tempo de codificação é dedicado à aplicação das transformadas, enquanto os 73,04% restantes correspondem às demais operações do pipeline do codificador. Observa-se, ainda, uma tendência crescente: à medida que o QP diminui (indicando maior qualidade e detalhamento do vídeo) o tempo alocado à etapa de transformadas aumenta. Para QP = 22, essa proporção atinge 34,57%, evidenciando que a codificação de vídeos com maior riqueza de detalhes demanda mais intensamente a aplicação da ferramenta MTS, tanto em sua forma explícita quanto implícita.

Esse mesmo padrão de crescimento é verificado na configuração *Random Access*, conforme apresentado na Figura 22. Entretanto, as proporções observadas são, de

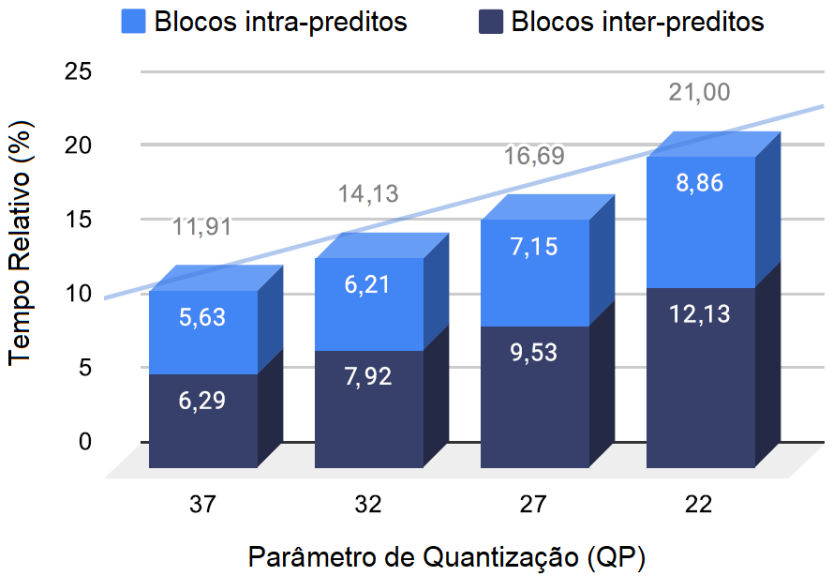

Figura 22 – Tempo de execução da etapa das transformadas para a configuração *Random Access*.

modo geral, inferiores às da configuração *All Intra*. Por exemplo, para QP = 22, a etapa de transformadas responde por 21% do tempo total de codificação, uma diferença de mais de 10 pontos percentuais em relação ao valor observado na configuração *All Intra*. Esse contraste indica que otimizações aplicadas exclusivamente à etapa de transformadas tendem a produzir ganhos mais expressivos na configuração *All Intra*, enquanto seus efeitos na configuração *Random Access* serão mais limitados em termos de impacto no tempo total de codificação.

Outro aspecto relevante da configuração *Random Access* é a divisão do tempo de processamento da etapa de transformadas entre blocos gerados por predições intra-quadro e interquadros. Como detalhado na Figura 22, para QP = 22, dos 21% do tempo total dedicado à etapa de transformadas, 12,13% referem-se a blocos inter-preditos, enquanto 8,86% correspondem a blocos intra-preditos. Isso demonstra que a maior parte da carga computacional da etapa de transformadas nessa configuração está concentrada nos blocos interquadros, o que destaca a importância de se considerar separadamente os dois tipos de predição em estudos de otimização.

Esses resultados evidenciam que análises restritas à configuração *All Intra* podem negligenciar uma parcela significativa do comportamento do codificador em cenários reais, onde a predição temporal é amplamente utilizada. Como consequência, estratégias de otimização desenvolvidas com base apenas nessa configuração podem apresentar desempenho inferior ao esperado quando aplicadas em codificadores configurados para *Random Access*, devido às diferenças no padrão de uso das transformadas entre as duas abordagens.

Portanto, para que avanços na etapa das transformadas sejam de fato significativos e aplicáveis a uma ampla gama de aplicações, é imprescindível que as estratégias

de otimização considerem ambas as configurações. Isso garante que as soluções propostas sejam eficientes e adaptáveis a diferentes perfis de codificação. Os achados desta análise reforçam o papel crítico da etapa de transformadas no desempenho global do codificador e evidenciam fatores que devem ser cuidadosamente considerados no desenvolvimento de soluções de otimização.

4.5 Análise da Taxa de Testagem da MTS para o Modo Explícito

Dentre os diferentes modos avaliados, o modo MTS-MODO = 3 (que ativa a MTS explícita tanto para predições intra quanto interquadros), destaca-se por apresentar o maior tempo de codificação, ainda que também tenha proporcionado os melhores resultados em termos de eficiência de compressão. Esse comportamento reforça seu potencial como alvo para estratégias de otimização, especialmente em contextos onde a redução de complexidade e o consumo energético são fatores críticos.

Considerando o objetivo central desta tese, que é a redução do consumo energético na etapa de transformadas por meio de soluções baseadas em *hardware* de baixo consumo e aprendizado de máquina, a MTS explícita surge como uma oportunidade estratégica de intervenção. Seu elevado custo computacional, evidenciado nas análises realizadas nas Seções 4.3.1 e 4.3.2, indica que ganhos expressivos podem ser obtidos caso sua execução seja otimizada.

Nesse sentido, foi conduzida uma análise específica voltada ao comportamento do modo MTS-MODO = 3, com foco na identificação de padrões de escolha entre as combinações de transformadas mais frequentemente testadas pelo codificador. A compreensão desses padrões abre caminho para o desenvolvimento de abordagens heurísticas capazes de reduzir o tempo de execução da etapa de transformadas, contribuindo diretamente para a diminuição do consumo energético sem comprometer significativamente a eficiência de compressão.

Para a realização da análise estatística, as informações extraídas do *software* de referência foram associadas às transformadas avaliadas em cada bloco candidato, e não apenas àquelas efetivamente aplicadas no bloco final selecionado pelo codificador. Essa abordagem foi adotada com o propósito de ampliar o escopo da análise, permitindo a consideração de um número significativamente maior de amostras. Mesmo quando um bloco candidato não é escolhido na versão final do vídeo codificado, o codificador ainda precisa processá-lo para avaliar a aplicabilidade de uma transformada e, caso aplicável, determinar qual combinação resulta no menor custo de codificação.

Esse nível de detalhamento possibilita uma compreensão mais profunda do processo de tomada de decisão interna do codificador, oferecendo informações sobre o comportamento do módulo de transformadas. Tais dados se mostram fundamentais para o desenvolvimento de estratégias de otimização, especialmente aquelas voltadas

à redução de complexidade e consumo energético.

Os resultados referentes às taxas de seleção das transformadas para cada bloco candidato são apresentados de forma segmentada, considerando-se a distribuição das transformadas em função dos valores do parâmetro de quantização (QP), do tipo de predição (intra e interquadros) e do tamanho dos blocos analisados.

4.5.1 Resultados para a taxa de seleção das transformadas distribuídas por valor de QP

Como discutido anteriormente, o parâmetro de quantização (QP) exerce um papel central no processo de codificação de vídeo, sendo responsável por controlar o nível de compressão e, consequentemente, a qualidade do vídeo reconstruído. É nessa etapa que ocorre a maior perda de informação, uma vez que a quantização reduz a precisão dos coeficientes gerados pela transformada, impactando diretamente a fidelidade do sinal reconstruído. Diante dessa relevância, esta análise tem como objetivo investigar a taxa de seleção das transformadas aplicadas aos blocos escolhidos no processo de codificação, classificando os resultados conforme os diferentes valores de QP utilizados (22, 27, 32 e 37). Essa abordagem permite compreender como o nível de quantização influencia a escolha das transformadas, fornecendo subsídios para futuras estratégias de otimização. Os resultados obtidos podem ser vistos na Tabela 11.

Tabela 11 – Distribuição percentual das transformadas por QP

Tipo de Transformada	**QP (%)**				**Total por**
	22	**27**	**32**	**37**	**Transformada (%)**
MTS desabilitada	5,28	3,15	2,03	1,21	**11,67**
DCT-II e DCT-II	34,23	15,58	7,98	4,06	**61,84**
SKIP	0,94	0,24	0,10	0,04	**1,31**
DST-VII e DST-VII	12,41	5,23	2,47	1,21	**21,32**
DCT-VIII e DST-VII	0,84	0,37	0,19	0,08	**1,48**
DST-VII e DCT-VIII	0,91	0,38	0,18	0,07	**1,53**
DCT-VIII e DCT-VIII	0,53	0,24	0,07	0,01	**0,85**

Observa-se uma tendência clara de redução no uso de transformadas à medida que o valor do QP se eleva. Essa tendência indica que, em cenários com menor compressão (QP mais baixos), o codificador explora com maior intensidade a aplicação de transformadas durante o processo de decisão. Entre os diferentes valores analisados, o QP = 22 apresenta a maior incidência, com destaque para a combinação DCT-II_DCT-II, que foi selecionada em 34% dos blocos candidatos. Em contrapartida,

à medida que o QP aumenta, a frequência de seleção dessa transformada diminui significativamente: 15,58% para QP = 27, 7,98% para QP = 32 e apenas 4,06% para QP = 37.

Esse comportamento não é exclusivo da DCT-II_DCT-II, as outras combinações de transformadas apresentam padrão semelhante, o que confirma uma correlação direta entre a intensidade da quantização e a redução do número de transformadas aplicadas. Em síntese, valores mais altos de QP resultam em menor variação das transformadas.

Os dados indicam que, em valores baixos de QP, como QP = 22, a configuração MTS desabilitada ocorre com maior frequência, sugerindo que a não aplicação de múltiplas transformadas ainda pode ser uma escolha eficiente em cenários de alta fidelidade. À medida que o QP aumenta, essa ocorrência diminui consideravelmente, como observado em QP = 37, onde a seleção de transformadas alternativas se torna mais comum.

Essa tendência evidencia uma correlação entre o nível de quantização e a variabilidade na escolha das transformadas: quanto maior o QP, maior a necessidade de explorar diferentes combinações para compensar a perda de qualidade causada pela compressão.

4.5.2 Resultados para a taxa de seleção das transformadas distribuídas por tipo de predição

A análise da distribuição das transformadas em função do tipo de predição é importante para compreender o comportamento do codificador diante de diferentes contextos estruturais do vídeo. Cada tipo de predição apresenta características distintas: enquanto a predição intraquadro explora redundâncias espaciais dentro do mesmo quadro, a predição interquadros busca redundâncias temporais entre quadros consecutivos. Como consequência, a escolha da transformada mais eficiente pode variar significativamente entre essas duas modalidades. Avaliar separadamente as taxas de utilização das transformadas para blocos intra e inter permite identificar padrões de uso diferenciados, o que contribui para o desenvolvimento de estratégias de otimização mais direcionadas. A Tabela 12 apresenta a porcentagem da distribuição das transformadas através do tipo de predição.

Os dados revelam que a transformada DCT-II, aplicada nas direções horizontal e vertical, é amplamente dominante no processo de codificação, sendo responsável por 61,84% das escolhas do codificador. Esse resultado está alinhado com as expectativas, uma vez que a DCT-II constitui a base das transformadas utilizadas em codificadores modernos, em função de sua alta eficiência na compactação de energia. Quando analisada por tipo de predição, verifica-se que 39,25% dessas ocorrências correspondem a blocos preditos usando predição intraquadro, enquanto 22,59% es-

Tabela 12 – Distribuição percentual das transformadas por tipo de predição

Tipo de Transformada	Intra (%)	Inter (%)	Total por Transformada (%)
MTS não permitida	1,91	9,76	**11,67**
SKIP	0,52	0,79	**1,31**
DCT-II e DCT-II	39,25	22,59	**61,84**
DST-VII e DST-VII	17,76	3,56	**21,32**
DCT-VIII e DST-VII	0,08	1,39	**1,48**
DST-VII e DCT-VIII	0,23	1,31	**1,53**
DCT-VIII e DCT-VIII	0,00	0,85	**0,85**

tão associados à predição interquadros. Esse leve predomínio nos blocos intra sugere uma maior adequação da DCT-II_DCT-II a conteúdos com alta correlação espacial.

A segunda combinação mais recorrente é a DST-VII, também aplicada nas duas direções, que representa 21,32% das escolhas. Destes, 17,76% ocorrem em blocos intraquadro e apenas 3,56% em blocos interquadros, indicando que essa transformada tende a ser particularmente eficiente na representação de padrões estruturais mais frequentes em blocos intra. Dessa forma, tanto a DCT-II_DCT-II quanto a DST-VII_DST-VII se destacam como as transformadas com maior impacto sobre o desempenho do codificador, motivo pelo qual quaisquer alterações nos critérios de seleção dessas combinações devem ser cuidadosamente avaliadas. Embora a otimização dessas transformadas possa reduzir o tempo de processamento na etapa de transformação, mudanças inadequadas podem comprometer a qualidade de reconstrução ou aumentar a taxa de bits.

Em contrapartida, combinações híbridas como DCT-VIII_DST-VII e DST-VII_DCT-VIII apresentam taxas de uso significativamente menores, de 1,48% e 1,53%, respectivamente, sugerindo que sua aplicação é restrita a cenários específicos. A combinação DCT-VIII_DCT-VIII, por sua vez, foi observada em apenas 0,85% dos casos, o que evidencia sua baixa relevância no processo decisório do codificador.

Além das transformadas principais, dois casos específicos merecem destaque: Skip e MTS não permitida. O primeiro ocorre em 1,31% dos casos e representa situações em que o codificador avalia que a aplicação de uma transformada não trará ganho significativo em compressão, optando por omiti-la. Essa decisão resulta em economia computacional, já que a etapa de transformada é uma das mais custosas do pipeline de codificação. O segundo caso, MTS não permitida, aparece em 11,67% das ocorrências e corresponde a situações em que o codificador, embora com a MTS globalmente habilitada, não realiza a avaliação de transformadas alternativas como

DST-VII ou DCT-VIII. Nesses casos, aplica-se diretamente a DCT-II nas duas direções, o que reduz o custo computacional em comparação com a ativação completa da MTS explícita, embora ainda demande mais recursos do que o caso de Skip. É relevante destacar que o comportamento MTS não permitida não é consequência de desativação da ferramenta, mas sim de restrições impostas por outras ferramentas do codificador ou pelo próprio tamanho do bloco.

4.5.3 Resultados para a taxa de seleção das transformadas distribuídas por tamanho de bloco

A análise da distribuição das transformadas em função do tamanho dos blocos de codificação tem como objetivo a compreensão de como o codificador adapta suas decisões à granularidade espacial do conteúdo. Blocos de diferentes dimensões apresentam características estatísticas distintas: blocos maiores tendem a representar regiões mais homogêneas, enquanto blocos menores capturam variações locais e detalhes com maior precisão. Essa heterogeneidade afeta diretamente a escolha da transformada mais eficiente em termos de compressão e qualidade de reconstrução. Ao examinar quais combinações de transformadas são preferencialmente aplicadas a cada faixa de tamanho de bloco, torna-se possível identificar padrões decisórios do codificador que podem orientar futuras estratégias de otimização, tanto no desenvolvimento de heurísticas quanto na implementação de arquiteturas de *hardware* com foco em eficiência computacional.

A Tabela 13 organiza os resultados de distribuição das transformadas de acordo com o tamanho dos blocos, permitindo uma análise segmentada por granularidade espacial. Para fins de simplificação, blocos retangulares foram agrupados conforme sua maior dimensão. Por exemplo, blocos de 32×16 são classificados na categoria 32×32, enquanto blocos de 4×8 foram incluídos no grupo 8×8.

Tabela 13 – Distribuição percentual das transformadas por tamanho de bloco

Tipo de Transformada	**Tamanho de Bloco (%)**					**Total por Transformada (%)**
	4x4	**8x8**	**16x16**	**32x32**	**64x64**	
MTS desabilitada	0,00	1,09	2,33	1,95	6,19	**11,56**
SKIP	0,08	0,48	0,51	0,24	0,00	**1,31**
DCT-II e DCT-II	3,53	19,71	23,91	14,69	0,00	**61,84**
DST-VII e DST-VII	1,60	7,55	8,17	4,00	0,00	**21,32**
DCT-VIII e DST-VII	0,01	0,46	0,66	0,35	0,00	**1,48**
DST-VII e DCT-VIII	0,02	0,51	0,67	0,34	0,00	**1,54**
DCT-VIII e DCT-VIII	0,00	0,28	0,39	0,18	0,00	**0,85**

A análise revela um padrão claro: blocos maiores fazem uso predominante da DCT-II, enquanto blocos menores apresentam maior diversidade de transformadas, incluindo DST-VII, DCT-VIII e suas combinações. Esse comportamento reflete as características estruturais dos blocos. Blocos grandes, geralmente mais homogêneos, se beneficiam da DCT-II pela sua eficiência na compactação de energia. Já blocos menores, por conterem mais detalhes e bordas, tendem a ser melhor representados por transformadas alternativas, que preservam nuances espaciais com maior fidelidade.

Cabe destacar que a aplicação da MTS é limitada a blocos com tamanho máximo de 32×32. Assim, blocos superiores a esse limite (como os de 64×64 e 128×128) não passam pelo processo de seleção explícita de transformadas, sendo automaticamente classificados sob a configuração MTS desabilitada. Nesses casos, apenas a DCT-II é aplicada, sem avaliação de outras combinações possíveis. Tal limitação está possivelmente associada à eficácia reduzida de transformadas como DST-VII e DCT-VIII em blocos de grandes dimensões, nos quais a variação espacial é menos acentuada.

Dos 11,67% de casos identificados como MTS desabilitada, a maioria corresponde a blocos de 64×64 (6,19%) e 128×128 (0,11%), o que confirma que a restrição por tamanho é o principal fator para a não ativação da MTS nesses cenários.

Nos contextos em que a MTS explícita é aplicada, a combinação DCT-II_DCT-II permanece como a mais recorrente, sobretudo em blocos de 16×16 (23,91%) e 8×8 (19,71%). Em contraste, sua incidência em blocos de 4×4 é bem menor (3,59%). Essa tendência também se verifica para outras combinações de transformadas, embora com menor expressividade, o que sugere que blocos de 8×8 e 16×16 exercem maior influência no processo de decisão da transformada, possivelmente por representarem uma proporção significativa dos blocos candidatos durante a codificação.

4.6 Resumo do Capítulo

Este capítulo apresentou uma análise do impacto da ferramenta MTS na codificação de vídeo do padrão VVC, considerando aspectos de complexidade computacional, eficiência de compressão e comportamento do codificador em diferentes configurações e cenários. As diversas investigações realizadas buscaram compreender os custos e benefícios da ativação da MTS, sobretudo em seu modo explícito, e identificar padrões de uso que possam subsidiar estratégias futuras de otimização com foco na redução do consumo energético. Inicialmente, foi conduzida uma análise comparativa entre os modos com MTS habilitada e desabilitada. Verificou-se que, embora a desativação da MTS reduza em média 12,5% o tempo de codificação, há um aumento de 1,1% no BD-BR. Esse resultado confirma a importância da MTS para o desempenho do VVC, mas também evidencia seu alto custo computacional. Na sequência, avaliou-se o comportamento dos diferentes modos de operação da MTS. O modo MTS-MODO

= 3, que ativa explicitamente a ferramenta tanto para blocos intra quanto interquadros, apresentou o maior ganho de compressão (-0,77% no BD-BR), mas também o maior tempo de codificação (+13,41%), sendo identificado como o principal alvo para estratégias de otimização. A configuração All Intra se mostrou mais exigente computacionalmente que a Random Access, com maior tempo relativo dedicado à etapa de transformadas. A análise do tempo específico da etapa de transformadas revelou que, em QP baixos (maior qualidade), essa etapa pode representar até 35% do tempo total de codificação. A importância da MTS explícita foi reforçada nesse cenário, sendo responsável por significativa parte da complexidade. A distribuição do uso das transformadas foi investigada por tipo de predição, tamanho de bloco e valores de QP, revelando que a DCT-II é a mais utilizada, seguida pela DST-VII, com predomínio nos blocos intra e em tamanhos menores. Além disso, transformadas híbridas apresentaram baixa incidência, indicando potencial de simplificação. Com base nesses achados, conclui-se que a MTS explícita é essencial para a eficiência do VVC, mas impõe desafios consideráveis em termos de complexidade e energia. Assim, o capítulo justifica e direciona a etapa seguinte da tese: o desenvolvimento de estratégias de aceleração e otimização da MTS por meio de modelos preditivos e abordagens de *hardware* de baixo consumo.

5 DECISÃO RÁPIDA PARA MTS ATRAVÉS DE APRENDIZADO DE MÁQUINA

Conforme abordado na seção anterior, a utilização da MTS explícita para blocos preditos tanto com predição intraquadro quanto interquadros (MTS = 3) resulta em melhorias na eficiência de codificação, mas também impõe um aumento significativo no custo computacional. Esse aumento abre caminho para a busca por estratégias de otimização. Diversos estudos, como os encontrados em (Wang; Wang; Yang; Luo; Liang; Huang, 2022b; Wang; Feng; Zhang; Yang, 2024; He; Xiong; Yang; He; Chen, 2022b; Fu; Zhang; Mu; Chen, 2019b; Saldanha; Sanchez; Marcon; Agostini, 2022b; Bonnineau; Puri; Naser; Poirier; Léannec, 2024), concentram-se principalmente na redução da complexidade associada à MTS em blocos intra (MTS = 1), desconsiderando a complexidade em blocos inter. Contudo, conforme evidenciado pela análise realizada, o principal desafio reside no alto custo computacional gerado pela MTS explícita, tanto para blocos das predições intra quanto inter.

Neste contexto, este capítulo apresenta como proposta a utilização de técnicas de aprendizado de máquina para acelerar o processo de decisão da MTS em codificadores VVC. A ideia central é utilizar dados obtidos durante a fase de transformada explícita da codificação para treinar modelos preditivos que, ao identificar padrões nos dados, possam antecipar quais transformadas devem ser testadas durante o processo de codificação. Ao aplicar esse método, é possível acelerar a seleção de transformadas, resultando em uma redução substancial no tempo de codificação.

Para a implementação dessa proposta, foram empregados modelos de árvore de decisão treinados com a biblioteca Scikit-Learn (Pedregosa; Varoquaux; Gramfort; Michel; Thirion; Grisel; Blondel; Prettenhofer; Weiss; Dubourg; Vanderplas; Passos; Cournapeau; Brucher; Perrot; Duchesnay, 2011), uma ferramenta amplamente reconhecida no desenvolvimento de modelos de aprendizado supervisionado. Todos os experimentos realizados neste estudo utilizaram essa biblioteca para treinar os modelos, avaliando seu desempenho na predição de transformadas que deveriam ser testadas em cada bloco durante a codificação. O uso de técnicas de aprendizado de máquina, em especial árvores de decisão, apresenta-se como uma abordagem pro-

missora para lidar com a complexidade do processo de codificação. Essa estratégia permite acelerar a tomada de decisão sobre as transformadas a serem testadas, contribuindo para a redução do tempo total de codificação, sem comprometer a eficiência de codificação proporcionada pela MTS explícita.

A avaliação dos modelos de aprendizado foi realizada por meio de métricas de acurácia e *F1-Score*. A acurácia corresponde à proporção de previsões corretas realizadas pelo modelo em relação ao total de amostras avaliadas. No entanto, como essa métrica pode ser insensível à presença de desbalanceamento entre classes, o *F1-Score* também foi empregado. O *F1-Score* é a média harmônica entre a precisão (proporção de verdadeiros positivos entre todas as previsões positivas) e a revocação (proporção de verdadeiros positivos identificados em relação a todas as instâncias realmente positivas). Essa métrica fornece uma avaliação mais equilibrada do desempenho, especialmente em cenários de classes desbalanceadas (Powers, 2011). Mesmo que os dados utilizados na análise já estejam balanceados, o uso do *F1-Score* se justifica pelo fato de a maioria dos trabalhos da área empregar essa métrica como base de comparação.

5.1 Implementação e Fluxo de Execução dos Modelo Preditivos

Para o desenvolvimento dos modelos preditivos propostos neste estudo, um conjunto de 62 *features* (características) foi extraído, com o objetivo de construir o banco de dados utilizado no treinamento. Devido à grande quantidade de dados gerados durante a coleta e aos requisitos de armazenamento, optou-se por extrair as *features* de blocos de 32×32 *pixels*. Essa abordagem permitiu reduzir o tamanho geral dos dados, mantendo, contudo, a eficiência na representação das informações necessárias para o treinamento.

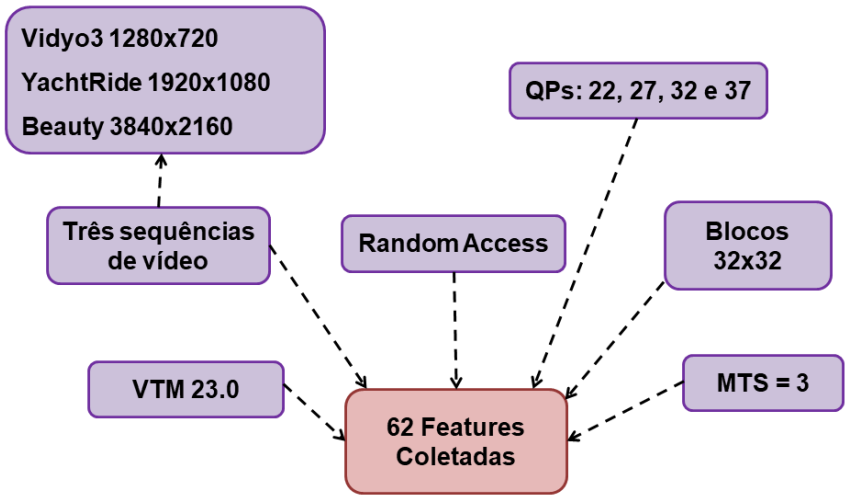

Figura 23 – Ambiente para coleta de dados do modelo.

A fim de gerar os dados brutos para a criação dos modelos preditivos, foram utili-

zadas três sequências de vídeo: *Vidyo3_1280x720_60*, *YachtRide_1920x1080* e *Beauty_3840x2160*, que estão definidas nos documentos CTC (Zhao; Chen; Choi; Cock; Grange; He; Helle; Ye; Zhang, 2022; Bossen, 2013; Daede; Norkin; Brailovskiy, 2020). Essas sequências foram selecionadas com base em suas resoluções distintas e no valor da Informação Espacial-Informação Temporal (SITI) (ITU-T, 2021), com o intuito de representar uma gama diversificada de cenários e proporcionar uma análise mais abrangente e representativa. Cada uma dessas sequências foi codificada em diferentes valores de QP (22, 27, 32 e 37) com o modo MTS configurado para 3, que corresponde à utilização das transformadas explícitas tanto para blocos de predição intra quanto para predição inter. Vale ressaltar que as ferramentas LFNST, SBT e ISP foram desativadas durante a coleta dos dados, com o intuito de evitar que essas ferramentas influenciassem a sinalização da MTS habilitada, que controla a ativação das transformadas explícitas. O diagrama da Figura 23 apresenta o ambiente utilizado para a coleta das *features*.

No processo de codificação realizado pelo codificador, inicialmente é gerada uma lista de modos de transformadas possíveis para um bloco. O modo de transformada padrão é o DCT-II_DCT-II e transformadas adicionais são inseridas quando a transformada explícita está ativada. Esse fluxo original do VTM pode ser visto na Figura 24.

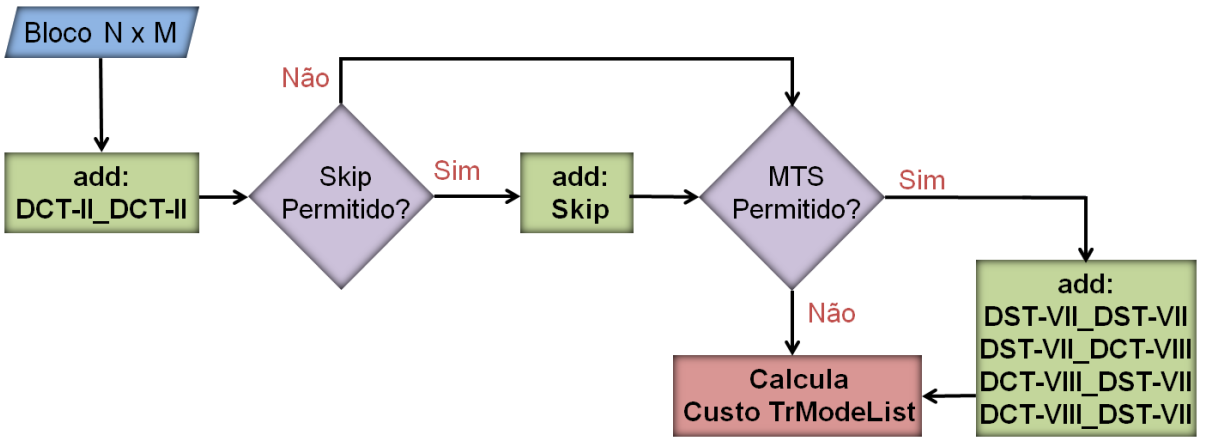

Figura 24 – Fluxo de execução do VTM original.

Para acelerar o processo de decisão sobre quais transformadas devem ser testadas durante a etapa de codificação, propõe-se um fluxo de execução otimizado. Nesse fluxo, o **Modelo A** tem como função inicial determinar a necessidade de utilização de transformadas adicionais no bloco em codificação. Caso o **Modelo A** indique que não há tal necessidade, o fluxo adota a estratégia denominada *shortlist* (classe S), na qual é avaliado exclusivamente o custo associado à transformada DCT-II_DCT-II. Por outro lado, se o **Modelo A** indicar a necessidade de explorar transformadas adicionais, o fluxo direciona a execução para o **Modelo B** (classe B), responsável por selecionar as combinações de transformadas que devem ser efetivamente testadas. O **Modelo**

B realiza esta escolha entre duas alternativas principais: (i) testar unicamente a combinação DST-VII_DST-VII (classe C); ou (ii) testar as combinações DST-VII_DCT-VIII, DCT-VIII_DST-VII e DCT-VIII_DCT-VIII (classe A).

A implementação do fluxo proposto é ilustrada na Figura 25, onde as decisões padrão do VTM são destacadas em cinza, enquanto os modelos sugeridos são representados em laranja. Devido às diferenças significativas entre os blocos preditos intra e inter, as *features* foram coletadas separadamente para cada tipo de bloco. Como resultado, quatro modelos distintos foram propostos: dois modelos para os blocos da predição intra (**Modelo Intra A** e **Modelo Intra B**) e dois para os blocos da predição inter (**Modelo Inter A** e **Modelo Inter B**). Esses modelos visam oferecer uma abordagem mais eficiente e adaptada ao tipo de predição.

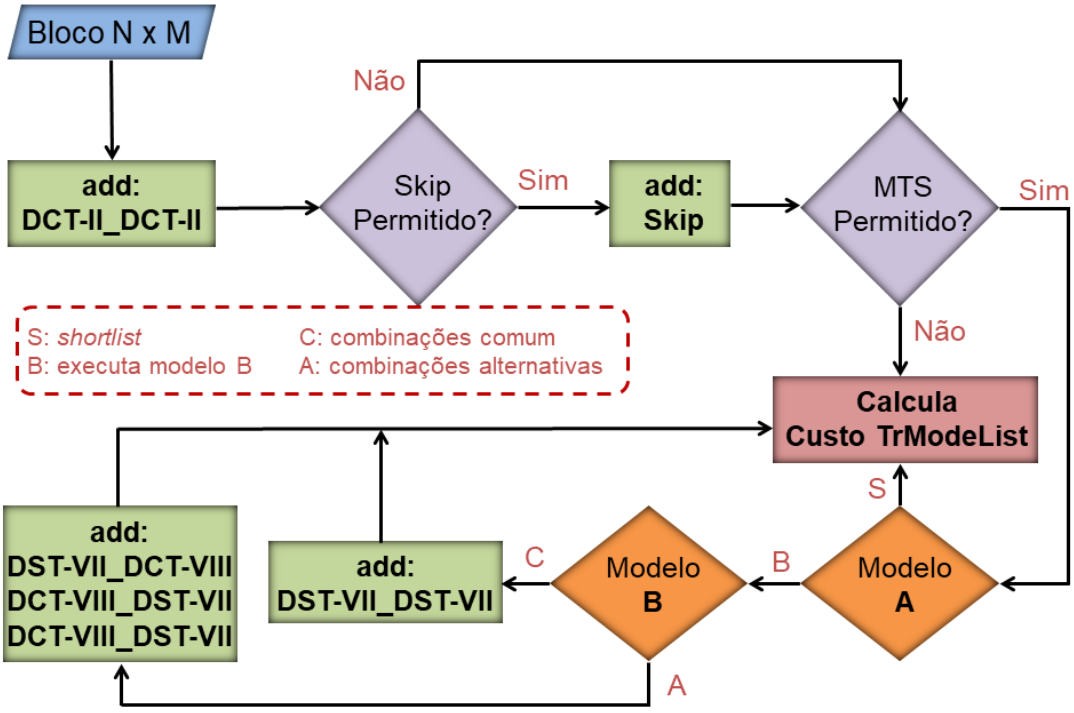

Figura 25 – Fluxo de execução proposto para a MTS com os modelos preditivos.

5.2 Produção de Conjuntos de Dados e Seleção de *Features*

A produção do conjunto de dados (*datasets*) seguiu uma série de etapas meticulosas para garantir a qualidade e a consistência das informações utilizadas no treinamento dos modelos preditivos. Após a coleta dos dados e a definição dos modelos, iniciou-se o processo de limpeza, que envolveu a remoção de duplicatas e colunas constantes, assegurando que as variáveis fossem representativas e não viessem a introduzir redundâncias ou informações irrelevantes. A normalização dos dados foi, então, realizada, com o objetivo de uniformizar as escalas das variáveis e garantir que os modelos de aprendizado de máquina não fossem influenciados por variáveis com

escalas muito discrepantes.

Além disso, os valores nulos presentes no conjunto de dados foram tratados adequadamente para evitar que esses dados incompletos afetassem negativamente a qualidade do treinamento. Foi adotada uma abordagem rigorosa para garantir que a integridade do conjunto de dados fosse mantida, o que incluiu a substituição de valores faltantes por estratégias baseadas em imputação ou exclusão, dependendo do caso. As variáveis também foram ajustadas para o formato exigido pelos algoritmos de aprendizado de máquina, incluindo transformações necessárias para representar adequadamente as informações categóricas e contínuas.

Por fim, uma etapa crucial foi o balanceamento das classes de MTS, a fim de garantir uma distribuição equitativa entre as diferentes classes de transformadas possíveis, evitando o viés dos modelos em favor de classes mais frequentes. Este processo de balanceamento foi realizado utilizando técnicas como *undersampling* (reduz a quantidade de amostras da classe majoritária). Essas preparações foram fundamentais para garantir que o conjunto de dados fosse robusto, consistente e representativo, formando a base ideal para o treinamento eficiente dos modelos preditivos propostos.

Para a seleção das *features* mais relevantes, foi empregada a técnica de Eliminação Recursiva de Características com Validação Cruzada (RFE-CV, do inglês *Recursive Feature Elimination with Cross-Validation*). O RFE-CV é um método amplamente utilizado em aprendizado de máquina para a seleção de variáveis, que combina a eliminação recursiva de características com a validação cruzada, visando otimizar a performance do modelo preditivo. O processo consiste em treinar o modelo inicial utilizando todas as *features* disponíveis e, em seguida, iterativamente eliminar a *feature* menos relevante, com base na importância atribuída a ela pelo modelo. Após a remoção de uma *feature*, o processo é repetido até que o número desejado de *features* seja alcançado. A validação cruzada é integrada a cada iteração para garantir que a remoção das variáveis não prejudique a capacidade preditiva do modelo, oferecendo uma abordagem robusta para a seleção de variáveis que maximiza a generalização do modelo.

Ao aplicar a técnica RFE-CV, o conjunto de dados foi reduzido a 16 *features* mais relevantes para os modelos propostos. Essas *features* podem ser vistas na Tabela 14.

As *featutres* (xi) e (xii) foram baseadas em (Wang; Feng; Zhang; Yang, 2024), pois capturam informações importantes relacionadas aos padrões dos resíduos das linhas e colunas do bloco, que têm impacto direto na eficiência da transformada. Já as *features* (xiii) a (xvi) seguiram a metodologia proposta por (Wang; Wang; Yang; Luo; Liang; Huang, 2022b), uma vez que são fundamentais para identificar padrões de resíduos específicos nas regiões do bloco, o que tem grande influência no processo de predição e na escolha das transformadas mais adequadas. A Tabela 15 apresenta as *features* extraídas para cada um dos modelos, fornecendo uma visão detalhada

Tabela 14 – *Features* resultantes após a aplicação da RFE-CV

Índice	***Features***
(i)	resolução do vídeo
(ii)	número do quadro atual
(iii)	coordenada *x* dentro do quadro
(iv)	coordenada *y* dentro do quadro
(v)	profundidade do bloco na árvore de partição
(vi)	parâmetro de quantização
(vii)	QP atual utilizado para codificação
(viii)	modo de predição intra
(ix)	índice de múltiplas referências para predição intra
(x)	soma absoluta dos valores de resíduos do bloco
(xi)	soma absoluta dos valores de resíduos da última linha do bloco
(xii)	soma absoluta dos valores de resíduos da última coluna do bloco
(xiii)	resíduos no canto superior esquerdo do bloco
(xiv)	resíduos no canto superior direito do bloco
(xv)	resíduos no canto inferior direito do bloco
(xvi)	resíduos no canto inferior esquerdo do bloco

das variáveis que influenciam o processo de transformadas e codificação dos blocos.

5.3 Treinamento dos Modelos

Para otimizar o desempenho dos modelos preditivos, foi realizada uma busca de hiperparâmetros utilizando a técnica de *Random Search* (Busca Aleatória), com um total de 500 iterações. Hiperparâmetros são parâmetros cujos valores não são aprendidos diretamente pelo modelo, mas definidos antes do processo de treinamento. Eles influenciam o comportamento do algoritmo de aprendizado, como a taxa de aprendizado, o número de camadas de uma rede neural ou a profundidade de uma árvore de decisão. A escolha adequada desses valores é crucial para maximizar a precisão e a eficiência do modelo. A técnica de *Random Search* consiste em amostrar aleatoriamente um conjunto de hiperparâmetros dentro de um espaço predefinido, em vez de testar todas as combinações possíveis (o que pode ser computacionalmente dispendioso). Durante a busca, os quatro conjuntos de dados foram divididos em 75% para desenvolvimento e 25% para validação. Para garantir a robustez dos resultados

Tabela 15 – *Features* usadas nos modelos

Modelo	***Features***
Inter A	(i), (ii), (iii), (iv), (vi), (x), (xi), (xii)
Inter B	(iv), (vi), (x), (xi), (xii), (xiii), (xiv), (xvi)
Intra A	(i), (ii), (iii), (iv), (v), (vi), (vii), (viii), (ix), (x), (xi), (xii), (xiii), (xiv), (xv), (xvi)
Intra B	(i), (iii), (iv), (viii), (x), (xi), (xii)

e minimizar o risco de *overfitting* (quando o modelo se ajusta excessivamente aos dados de treinamento), utilizou-se a técnica de Validação Cruzada com 5 dobras (*5-fold Cross-Validation*). Essa técnica envolve dividir o conjunto de dados em 5 partes iguais, treinando o modelo em 4 dessas partes e avaliando-o na parte restante, repetindo o processo 5 vezes, de modo que cada parte seja utilizada como validação uma vez. O processo gerou um total de 2.500 modelos treinados e testados. Os melhores hiperparâmetros encontrados durante o processo de busca podem ser consultados na Tabela 16.

Tabela 16 – Melhores hiperparâmetros encontrados

Hiperparâmetros	**Intra A**	**Intra B**	**Inter A/B**
min_samples_split	150	75	225
min_samples_leaf	70	30	30
max_leaf_nodes	300	300	300
max_features	13	10	7
max_depth	80	20	40
criterion	‘gini’	‘gini’	‘gini’

Conforme exposto no Capítulo 2, as análises foram conduzidas com base em duas métricas principais. A acurácia foi utilizada para mensurar a proporção de previsões corretas realizadas pelo modelo em relação ao total de casos avaliados. Adicionalmente, o F1-Score foi empregado como métrica complementar, permitindo uma avaliação mais equilibrada do desempenho do modelo, sobretudo em contextos com possíveis desequilíbrios entre as classes.

Os resultados obtidos durante a avaliação dos modelos preditivos indicam que o Modelo Intra A alcançou uma acurácia de 77%, com valores de F1-Score de 0,76 para a classe S (*shortlist*) e 0,78 para a classe B (executar modelo B).

O Modelo Intra B obteve uma acurácia de 68%, com F1-Scores de 0,69 para a classe C (combinações comuns) e 0,67 para a classe A (combinações alternativas). O

desempenho mais baixo deste modelo pode ser atribuído à dificuldade em lidar com as combinações alternativas de transformadas, que exigem maior complexidade na tomada de decisão.

O Modelo Inter A se destacou como o melhor modelo, atingindo 78% de acurácia e F1-Scores de 0,77 para a classe S e 0,79 para a classe B, demonstrando uma boa capacidade de previsão para ambos os casos, com um equilíbrio adequado entre precisão e revocação. Por outro lado, o Modelo Inter B apresentou o pior desempenho, com 67% de acurácia e F1-Scores de 0,67 para ambas as classes, indicando que o modelo teve dificuldades significativas em lidar com os dados, possivelmente devido a uma menor generalização ou ao impacto de um conjunto de parâmetros não ideal.

5.4 Resultados com a integração dos Modelos ao VTM

Para avaliar a eficácia do método proposto, a versão 23.0 do *software* de referência VTM (Jvet, 2024) foi adaptada para integrar os modelos treinados. A avaliação envolveu a codificação de 17 vídeos provenientes de (Bossen, 2021), que abrangem sequências com resoluções espaciais em HD, Full HD e 4K. Esses vídeos foram processados tanto pela versão original quanto pela versão modificada do VTM, o que possibilitou uma análise comparativa entre os tempos de execução e a eficiência de codificação. A medição do tempo foi realizada utilizando a configuração *Random Access*, com o parâmetro MTS ajustado para 3 e a desativação do LFNST. A exclusão do LFNST seguiu a metodologia estabelecida em (Wang; Wang; Yang; Luo; Liang; Huang, 2022b), uma vez que a aplicação do MTS não ocorre em blocos que utilizam LFNST.

Os resultados apresentados na Tabela 17 indicam que a aceleração proposta proporciona uma redução média de 7,98% no tempo total de codificação, acompanhada de um aumento médio de 0,89% no BD-BR. Quando o foco recai sobre o processamento de blocos preditos por inter, observa-se uma redução média de 22,91% no tempo de codificação. De maneira análoga, para blocos preditos por intra, a redução do tempo de codificação atinge 45,33%.

Até onde se sabe, nenhuma outra proposta se concentrou de maneira específica na aceleração da codificação VVC para transformadas explícitas de blocos preditos por intra e inter, o que torna as comparações diretas com outras abordagens desafiadoras. Diversos estudos anteriores, como os de (Wang; Wang; Yang; Luo; Liang; Huang, 2022b) e (Fu; Zhang; Mu; Chen, 2019b), empregaram versões mais antigas do VTM, com foco exclusivo na codificação intra. Em contrapartida, a pesquisa de (Bonnineau; Puri; Naser; Poirier; Léannec, 2024) explora a MTS implícita, o que limita a comparação com a abordagem proposta, uma vez que esta se concentra na MTS explícita. Uma metodologia semelhante à proposta neste trabalho é descrita em

Tabela 17 – Resultados de eficiência de codificação e tempo para cada resolução

Resolução	BD-BR (%)	Tempo Total (%)	Tempo Inter (%)	Tempo Intra (%)
720p	0.75	-6.18	-12.50	-43.93
1080p	0.96	-8.57	-25.96	-49.06
4k	0.96	-9.18	-30.28	-43.00
Média	**0.89**	**-7.98**	**-22.91**	**-45.33**

(Saldanha; Sanchez; Marcon; Agostini, 2022b), que também utiliza árvores de decisão para otimizar a MTS. No entanto, uma diferença fundamental reside no fato de que (Saldanha; Sanchez; Marcon; Agostini, 2022b) incorpora uma decisão rápida para LFNST e utiliza uma versão mais antiga do VTM (10.0) com configuração All-Intra. Por outro lado, esta abordagem foca exclusivamente na MTS explícita e adota a versão mais recente do VTM (23.0), configurada para *Random Access*, permitindo a avaliação de blocos preditos tanto por intra quanto por inter. Embora o método proposto tenha apresentado um leve aumento no BD-BR (0,89% contra 0,43% em (Saldanha; Sanchez; Marcon; Agostini, 2022b)), ele abrange uma gama mais ampla de aplicações. Além disso, a redução no tempo de codificação permanece em uma margem competitiva, com uma diminuição de 7,98% contra 11% observados em (Saldanha; Sanchez; Marcon; Agostini, 2022b).

5.5 Resumo do Capítulo

Este capítulo apresentou a proposta e a implementação de um método para aceleração da etapa da MTS no codificador VVC, com foco no modo explícito aplicado a blocos intra e inter. Inicialmente, foi discutida a motivação para a otimização da MTS explícita, cujo elevado custo computacional representa um dos principais gargalos no processo de codificação, especialmente em dispositivos com restrições energéticas.

A estratégia proposta baseou-se na aplicação de modelos preditivos construídos por meio de aprendizado de máquina supervisionado, mais especificamente com o uso de árvores de decisão. Quatro modelos distintos foram desenvolvidos para tratar separadamente blocos com predição intra e inter, dividindo a tarefa entre a decisão de utilizar transformadas adicionais (Modelo A) e a seleção das combinações a serem testadas (Modelo B). A metodologia contemplou desde a coleta e pré-processamento de dados até a seleção das features mais relevantes via RFE-CV, culminando em um conjunto de modelos robustos e ajustados.

A integração dos modelos ao *software* de referência VTM 23.0 permitiu uma ava-

liação comparativa com a versão original. Os resultados demonstraram uma redução média de 7,98% no tempo total de codificação, com impacto moderado na eficiência de compressão (aumento de 0,89% no BD-BR). Destaque especial foi observado na codificação de blocos intra e inter, com reduções de 45,33% e 22,91%, respectivamente, na etapa de transformada.

Além de demonstrar o potencial da proposta em *software*, este capítulo reforça que os modelos preditivos desenvolvidos são suficientemente simples para serem integrados em codificadores implementados em *hardware*. Uma vez que os modelos consistem em estruturas de decisão hierárquicas baseadas em condições (IF/ELSE), sua implementação pode ser realizada com baixo custo em termos de área e energia. Nesse cenário, tais modelos podem ser empregados como unidades de controle para habilitar ou desabilitar o uso de determinadas transformadas, contribuindo diretamente para a redução do consumo energético do decodificador.

Por fim, destaca-se que a metodologia de desenvolvimento adotada neste capítulo (desde a construção dos modelos até sua integração ao codificador) é generalizável. Isso significa que, além do VTM, os mesmos princípios podem ser aplicados a outros codificadores de vídeo, sejam eles em *software* ou *hardware*, tornando-se uma abordagem promissora para otimizações futuras no domínio da compressão de vídeo.

6 ARQUITETURA DE *HARDWARE* PARA TRANSFORMADAS INVERSAS

No processo de codificação e decodificação do padrão VVC, as transformadas inversas desempenham um papel essencial na reconstrução dos blocos codificados. Embora sua aplicação seja mais frequentemente associada à etapa de decodificação do vídeo, as transformadas inversas também são utilizadas no codificador de vídeo, especificamente no laço de reconstrução (apresentado na Fig. 5). Como mencionado previamente, essas transformadas têm a função de converter os coeficientes de frequência de volta para o domínio espacial, permitindo assim a reconstrução dos resíduos, que serão somados às amostras preditas para gerar a imagem decodificada. Dessa forma, a correta implementação das transformadas inversas garante a fidelidade da reconstrução tanto no codificador quanto no decodificador.

É importante destacar a diferença de comportamento entre as transformadas diretas e inversas no que se refere ao impacto em custo computacional. As transformadas diretas são significativamente mais exigentes, pois são aplicadas repetidamente durante o processo de decisão, sendo testadas em múltiplas combinações por bloco candidato. Tal característica as torna um dos principais gargalos computacionais do codificador. Em contrapartida, as transformadas inversas não requerem essa testagem exaustiva: uma vez selecionada a transformada direta mais adequada, a transformada inversa correspondente é aplicada de forma única ao bloco final, apenas para fins de reconstrução.

Ainda assim, é essencial desenvolver arquitetura de *hardware* para a etapa de transformada inversa, pois seu processamento deve ser realizado de forma precisa e eficiente, garantindo a integridade da reconstrução do vídeo. Isso é especialmente relevante em sistemas embarcados e aplicações com restrições de consumo de energia ou latência. A implementação arquitetural deve, portanto, ser otimizada para conciliar alta taxa de processamento, baixo custo computacional e flexibilidade no suporte às diferentes transformadas previstas pelo padrão VVC.

Neste capítulo, propõe-se uma arquitetura de *hardware* dedicada à execução das transformadas inversas suportadas pelo padrão VVC. A solução projetada oferece su-

porte a diferentes tamanhos e formatos de blocos, incluindo DCT-II inversa (IDCT-II), DCT-VIII inversa (IDCT-VIII) e DST-VII inversa (IDST-VII), com foco na reutilização de blocos funcionais e na compatibilidade com a ferramenta de *Multiple Transform Selection*. A seguir, são detalhadas as características funcionais, o fluxo de processamento e a implementação da arquitetura proposta.

No desenvolvimento em *hardware*, as arquiteturas foram descritas em VHDL (*VHSIC Hardware Description Language*) (Society, 2009), uma linguagem padrão para a descrição de sistemas digitais. O VHDL permite a especificação do comportamento funcional e estrutural dos circuitos, sendo amplamente utilizado tanto em simulação quanto em síntese de *hardware*.

As arquiteturas foram projetadas para síntese como circuitos integrados de aplicação específica (*Application-Specific Integrated Circuit* – ASIC). Um ASIC é um chip dedicado a uma função específica, ao contrário do FPGA (*Field-Programmable Gate Array*), que pode ser reprogramado após a fabricação. A escolha por ASIC justifica-se pela maior eficiência em área, consumo energético e desempenho quando comparados ao FPGA, que, embora ofereça flexibilidade, apresenta maior *overhead* em termos de consumo energético e tempo de processamento (Weste; Harris, 2011).

A ferramenta Quartus II (Intel, 2022) foi utilizada para a descrição inicial dos projetos, enquanto a validação funcional foi realizada por meio do simulador ModelSim, ferramenta amplamente empregada para verificação de designs VHDL. Posteriormente, a síntese lógica foi realizada utilizando o *Cadence RTL Compiler* versão 14.02 (Cadence encounter, 2022), empregando a tecnologia de células padrão TSMC de 40nm. Esta tecnologia foi escolhida por proporcionar um bom equilíbrio entre velocidade de operação, capacidade de integrar um grande número de transistores em uma área reduzida e pela sua ampla adoção na indústria e na pesquisa acadêmica. Os resultados das sínteses foram avaliados considerando as métricas de área projetada e consumo energético. A área projetada refere-se ao espaço ocupado pelos circuitos no silício, impactando diretamente nos custos de fabricação e na capacidade de integração de sistemas complexos. O consumo energético, por sua vez, é uma métrica crítica para sistemas embarcados e aplicações que demandam alta eficiência energética, sendo determinante para a sustentabilidade e viabilidade de soluções em ambientes com restrições de energia (foco desta tese).

6.1 Arquitetura Proposta

A arquitetura proposta foi projetada para processar blocos de diferentes tamanhos, variando de 4×4 a 64×64 para a IDCT-II, e de 4×4 a 32×32 para as transformadas IDCT-VIII e IDST-VII. Ela também é capaz de lidar com todos os formatos de blocos, tanto quadrados quanto retangulares, conforme as especificações do VVC. Além

disso, a arquitetura permite a combinação dessas transformadas, tanto na primeira quanto na segunda dimensão, conforme as definições da ferramenta MTS (consultar Tabela 2 na seção 2.3.3).

O processo de cálculo pode ser abordado como uma multiplicação matricial, conforme ilustrado nas Equações 5 e 6.

$$Z_{1D} = CX^t \tag{5}$$

$$X_{1D} = Z^tC \tag{6}$$

As equações representam o processo de cálculo da transformada inversa bidimensional (2D), realizado em duas etapas sucessivas. Na primeira etapa, expressa por 5, aplica-se a transformada inversa em uma das direções (geralmente na vertical), em que *X* representa a matriz de coeficientes quantizados, X^t é sua transposta, *C* é a matriz de base da transformada inversa (como IDCT ou IDST), e Z_{1D} é o resultado intermediário. Na etapa seguinte, representada pela equação 6, realiza-se a transformação na direção restante (geralmente horizontal), onde Z^t é a transposta do resultado intermediário e X_{1D} representa os resíduos reconstruídos no domínio espacial. Esse processo é essencial para reverter os dados do domínio da frequência e permitir a reconstrução das imagens no decodificador.

Com base nessa representação matricial, a Figura 26 apresenta um diagrama de blocos simplificado, destacando os principais componentes da arquitetura: o bloco 1D, o *buffer* de transposição e o bloco 2D.

Figura 26 – Diagrama em blocos simplificado: 1D, buffer de transposição, 2D.

A Figura 26 ilustra o fluxo de processamento simplificado da etapa de transformada inversa em duas dimensões, conforme utilizado na arquitetura proposta. O processo inicia-se com a leitura dos coeficientes residuais (indicados como src0 até src63), que são os coeficientes no domínio da frequência resultantes do processo de quantização e codificação.

Esses coeficientes são primeiramente submetidos a uma transformada unidimensional (1D) na direção horizontal (ou vertical, dependendo da ordem). Esta primeira etapa aplica a matriz de transformada inversa correspondente, convertendo parcialmente os dados para o domínio espacial.

Em seguida, os resultados intermediários são armazenados no *buffer* de transposição, um componente responsável por reorganizar os dados, permitindo que a segunda transformada 1D seja aplicada na direção ortogonal à primeira. Esse processo de transposição é fundamental, pois garante que a aplicação das duas transformadas unidimensionais (1D) em direções ortogonais correspondam corretamente a uma transformada bidimensional (2D).

Finalmente, os dados transpostos passam pela segunda etapa de transformada 1D, completando o processo bidimensional. O resultado final dessa segunda etapa corresponde aos coeficientes reconstruídos no domínio espacial, representados como dst0 até dst63, que serão utilizados na reconstrução dos blocos da imagem no decodificador.

A arquitetura completa pode ser vista na Figura 27. Ela foi projetada para operar de forma parcialmente paralela, processando uma linha completa da matriz de entrada a cada ciclo de *clock*. O número de coeficientes a serem processados por ciclo depende do tamanho do bloco selecionado, que pode ser 4, 8, 16, 32 ou 64. Para acomodar o maior tamanho de bloco possível, ou seja, 64 coeficientes, o número de conexões de entrada e saída foi dimensionado adequadamente. Na Figura 27, essas entradas e saídas são representadas pelos sinais *src* e *dst*, respectivamente.

A entrada do bloco de cálculo da primeira dimensão é composta pelos coeficientes transformados obtidos após o processo de quantização inversa. O resultado do processamento nesta primeira dimensão gera um bloco intermediário, cujos coeficientes são armazenados em um *buffer* de transposição. Esses coeficientes armazenados no *buffer* são então utilizados como entrada para o cálculo na segunda dimensão. Este processamento na segunda dimensão pode ser entendido como a execução de um segundo bloco 1D, representando uma etapa adicional no fluxo de processamento da arquitetura.

A seleção do tipo de transformada e do tamanho de bloco utilizados na etapa de transformada inversa depende do contexto (codificador ou decodificador) em que essa operação ocorre. No decodificador, essas informações são explicitamente sinalizadas no bitstream, ou seja, os parâmetros da transformada (tipo e dimensão do bloco) já foram definidos durante a codificação e são utilizados diretamente, sem a necessidade de testagens adicionais. Isso implica que, do ponto de vista computacional, a aplicação da transformada inversa no decodificador não representa um gargalo em termos de consumo energético ou tempo de processamento, visto que não há múltiplas combinações a serem avaliadas.

No entanto, como dito anteriormente, no codificador, o cenário é substancialmente diferente. Durante o processo de seleção da melhor transformada direta, diversas combinações são testadas no módulo de MTS, a fim de encontrar aquela que oferece o menor custo taxa-distorção. Para cada uma dessas combinações candidatas, a

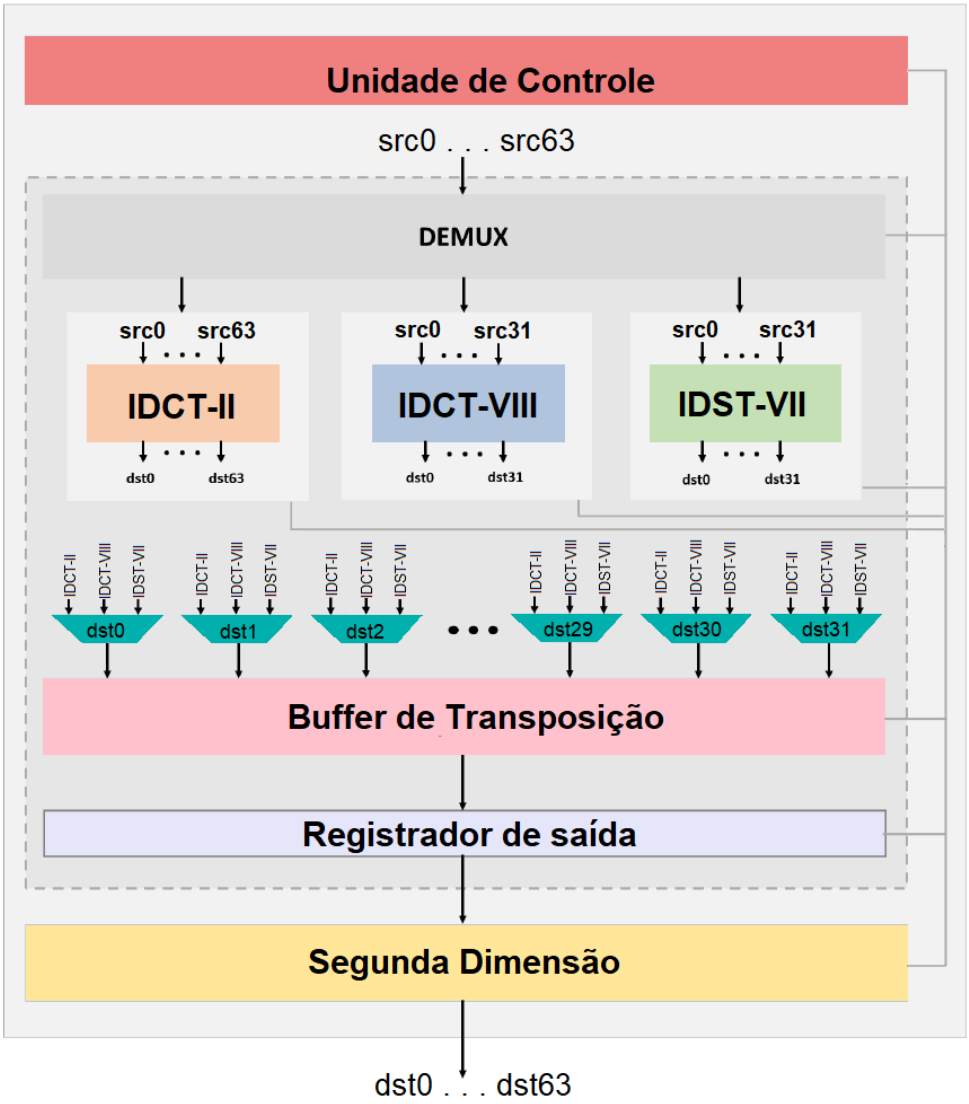

Figura 27 – Arquitetura proposta para as transformadas inversas de duas dimensões.

respectiva transformada inversa também precisa ser aplicada no laço de reconstrução, para que o codificador possa calcular os resíduos e estimar corretamente o custo de distorção. Dessa forma, se N transformadas forem testadas para um mesmo bloco, N transformadas inversas também serão executadas, aumentando significativamente o custo computacional da etapa.

Esse comportamento destaca um ponto crítico: embora a transformada inversa não seja diretamente responsável pela seleção de combinações, ela é executada repetidamente durante a codificação, o que pode levar a um aumento expressivo no consumo de energia. Neste contexto, os modelos preditivos apresentados no Capítulo 5 desempenham um papel estratégico. Ao reduzirem o número de combinações testadas durante a etapa de transformada direta, esses modelos impactam positivamente também a etapa de transformada inversa, contribuindo para a diminuição global do custo computacional e energético do processo de codificação.

As entradas do sistema são conectadas a um demultiplexador (conforme ilustrado

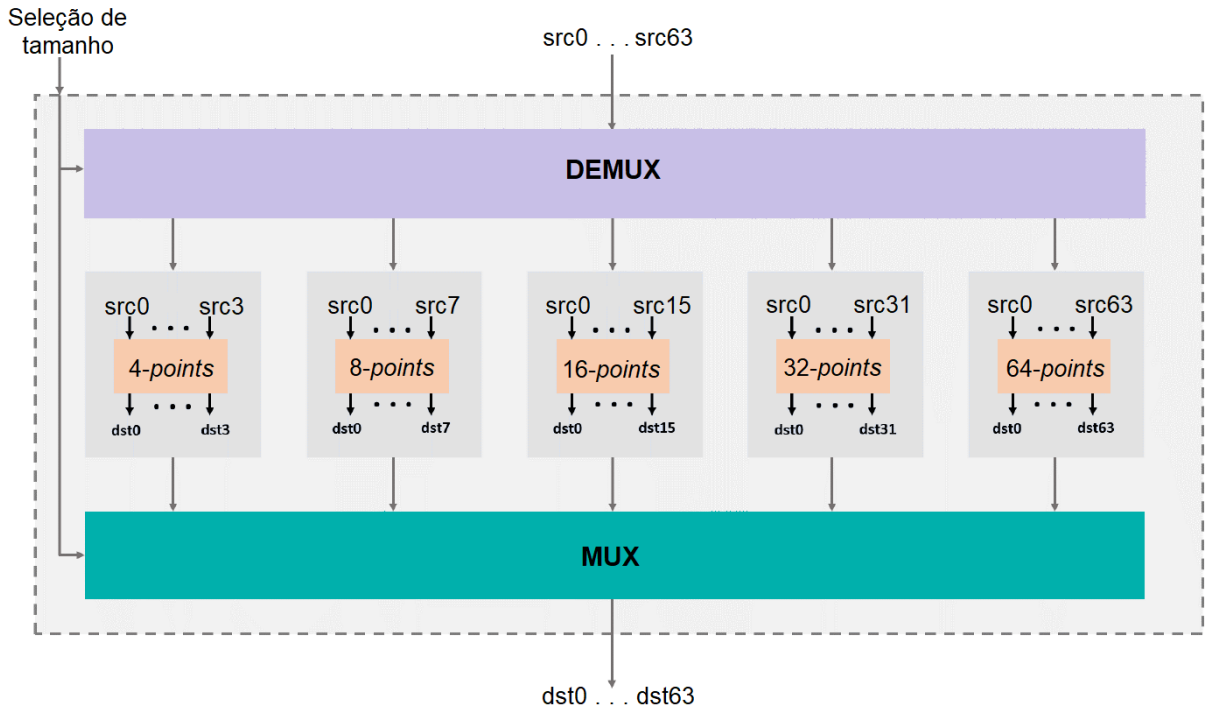

Figura 28 – *Hardware* da DCT-II para todos os tamanhos de bloco.

na Figura 27), que tem a função de selecionar o módulo adequado, correspondente ao tipo de transformada (IDCT-II, IDCT-VIII ou IDST-VII) escolhido para o processamento da matriz de entrada. Uma vez selecionado o módulo apropriado, a informação de entrada é direcionada para esse módulo, onde será processada de acordo com a transformada definida.

Cada coeficiente gerado pelos módulos das três transformadas é direcionado a um multiplexador, cuja função é encaminhar as saídas para o *buffer* de transposição. Para essa tarefa, é utilizado um total de 32 multiplexadores. As saídas são identificadas como *dst0* a *dst63*, sendo que, a partir de *dst32*, a conexão é realizada diretamente ao *buffer* de transposição, dado que a partição de tamanho 64 é aplicável exclusivamente à IDCT-II. Assim, o multiplexador desempenha o papel de determinar a origem de cada coeficiente, baseando-se no tipo de transformada à qual ele pertence.

A Figura 28 apresenta a estrutura interna do bloco dedicado ao cálculo da IDCT-II. A arquitetura é projetada para suportar múltiplos tamanhos de bloco, incluindo 4, 8, 16, 32 e 64. O controle de direcionamento das entradas e saídas é realizado por meio de um demultiplexador e um multiplexador, respectivamente. O demultiplexador distribui os coeficientes de entrada para o módulo correspondente ao tamanho do bloco selecionado, enquanto o multiplexador encaminha os resultados do cálculo para as próximas etapas do processamento.

Cada módulo de cálculo da IDCT-II inclui uma entrada de habilitação, permitindo que o processamento ocorra apenas quando o tamanho do bloco correspondente for ativado, otimizando o consumo de energia e recursos computacionais. Essa estra-

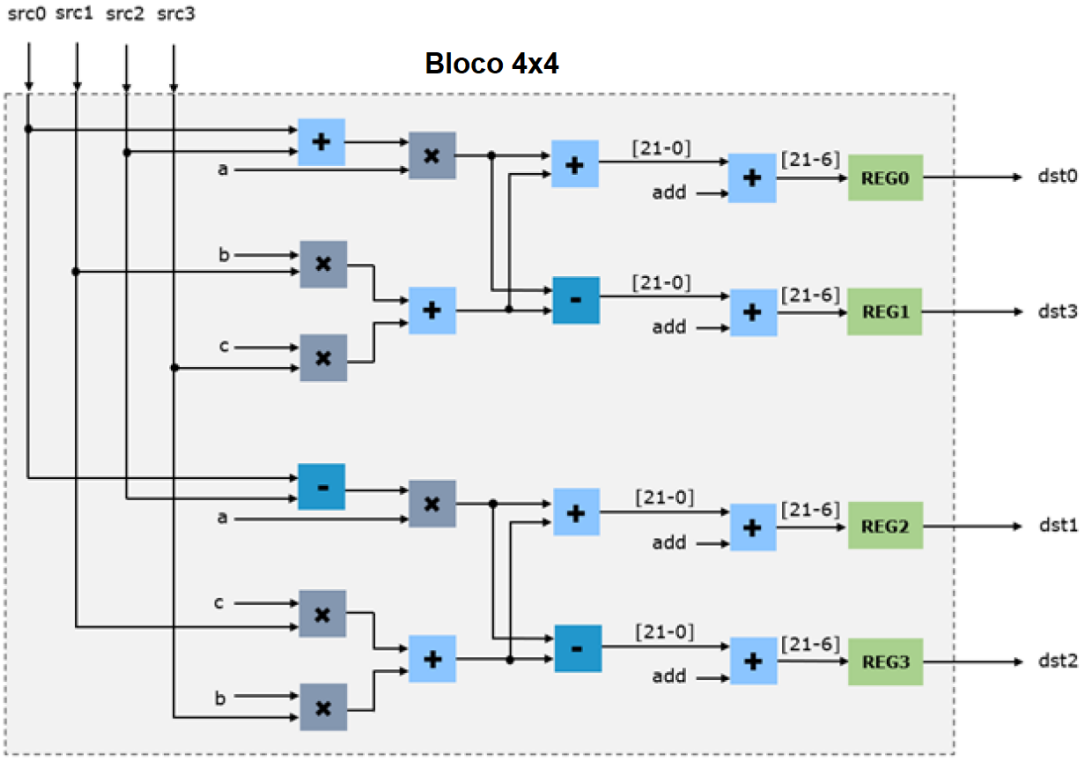

Figura 29 – *Hardware* da DCT-II para blocos de 4-pontos.

tégia de ativação seletiva é fundamental para garantir eficiência no processamento, reduzindo a complexidade de *hardware* desnecessária quando blocos menores são processados.

A mesma abordagem arquitetural é adotada para os módulos responsáveis pelo cálculo das transformadas IDCT-VIII e IDST-VII. No entanto, diferentemente da IDCT-II, essas transformadas possuem um limite máximo de tamanho de bloco de 32. Essa restrição se deve à natureza das transformadas e às especificações do padrão VVC, que definem a aplicação dessas transformadas a blocos menores.

A Figura 29 ilustra o design de *hardware* para uma linha de um bloco de 4 pontos, detalhando a estrutura interna utilizada para a computação da transformada inversa. O processamento é baseado na multiplicação dos coeficientes de entrada pelos valores da matriz de coeficientes da transformada, representada pela variável C na equação 5. Os sinais a, b e c indicam os coeficientes extraídos dessa matriz, os quais são utilizados na operação matricial para a reconstrução dos blocos transformados.

No cálculo da primeira dimensão, após a aplicação da matriz da transformada, um valor fixo de 64 é adicionado ao resultado final. Esse valor, representado pelo sinal add na Figura 29, é essencial para a operação de arredondamento e compensação numérica, reduzindo erros de precisão no processamento. Além disso, o valor final é deslocado sete posições para a direita, o que equivale a uma divisão por 128, garantindo a normalização do número de bits correto dos coeficientes gerados.

Na segunda dimensão, a estrutura de processamento segue um princípio seme-

lhante, mas com ajustes nos valores de arredondamento e deslocamento. O valor de add utilizado nessa etapa é 512, e o resultado final é deslocado dez posições para a direita, ou seja, dividido por 1024. Esse aumento no fator de deslocamento está relacionado à necessidade de maior normalização devido ao acúmulo de operações sucessivas ao longo das duas dimensões.

O processamento da transformada inversa é realizado de forma estruturada, operando linha por linha sobre a matriz de entrada. O tamanho do bloco processado é diretamente determinado pelo número de colunas dessa matriz, pois esse valor deve corresponder ao número de linhas da matriz de coeficientes da transformada. Dessa forma, a configuração da matriz de entrada não apenas define o tamanho do bloco transformado, mas também influencia o número de ciclos de *clock* necessários para completar o cálculo.

Por outro lado, o número de linhas da matriz de entrada determina quantas vezes um determinado bloco será processado. Esse fator é crucial, pois impacta diretamente no tempo total de execução da transformada. Por exemplo, ao selecionar a IDCT-VIII como a transformada aplicada na primeira dimensão, se a matriz de entrada possuir 4 linhas e 8 colunas, o bloco da IDCT-VIII de tamanho 8 será ativado por exatamente 4 ciclos de *clock*. Esse comportamento reflete a relação direta entre a altura do bloco e o tempo de processamento: uma matriz com 4 linhas exigirá 4 ciclos de *clock*, enquanto matrizes com 8, 16, 32 ou 64 linhas requererão 8, 16, 32 ou 64 ciclos de *clock*, respectivamente.

Essa variação no tempo de processamento exige uma estrutura de controle para garantir a correta ativação dos módulos de cálculo. Para isso, a unidade de controle foi projetada com 67 estados distintos, garantindo uma operação sequencial precisa. Entre esses estados, três são especificamente dedicados ao gerenciamento dos módulos de entrada e saída, assegurando a correta sincronização dos dados ao longo do pipeline de processamento. Essa abordagem permite que a arquitetura opere de forma eficiente, ajustando dinamicamente os tempos de execução conforme a configuração da matriz de entrada, otimizando o uso dos recursos de *hardware* e garantindo um alto desempenho no processamento das transformadas inversas.

O *buffer* de transposição desempenha um papel fundamental na arquitetura ao armazenar temporariamente os coeficientes resultantes da transformada na primeira dimensão antes de serem processados na segunda dimensão. Sua estrutura é projetada para garantir a correta organização e reordenação dos dados, permitindo que a segunda etapa do processamento ocorra sem perda de desempenho. A Figura 30 apresenta um diagrama esquemático desse *buffer*.

A estrutura do *buffer* de transposição é composta por 64 registradores sequenciais (uma vez que operam de forma ordenada, seguindo um fluxo contínuo de armazenamento e recuperação de coeficientes), sendo cada um deles capaz de armazenar uma

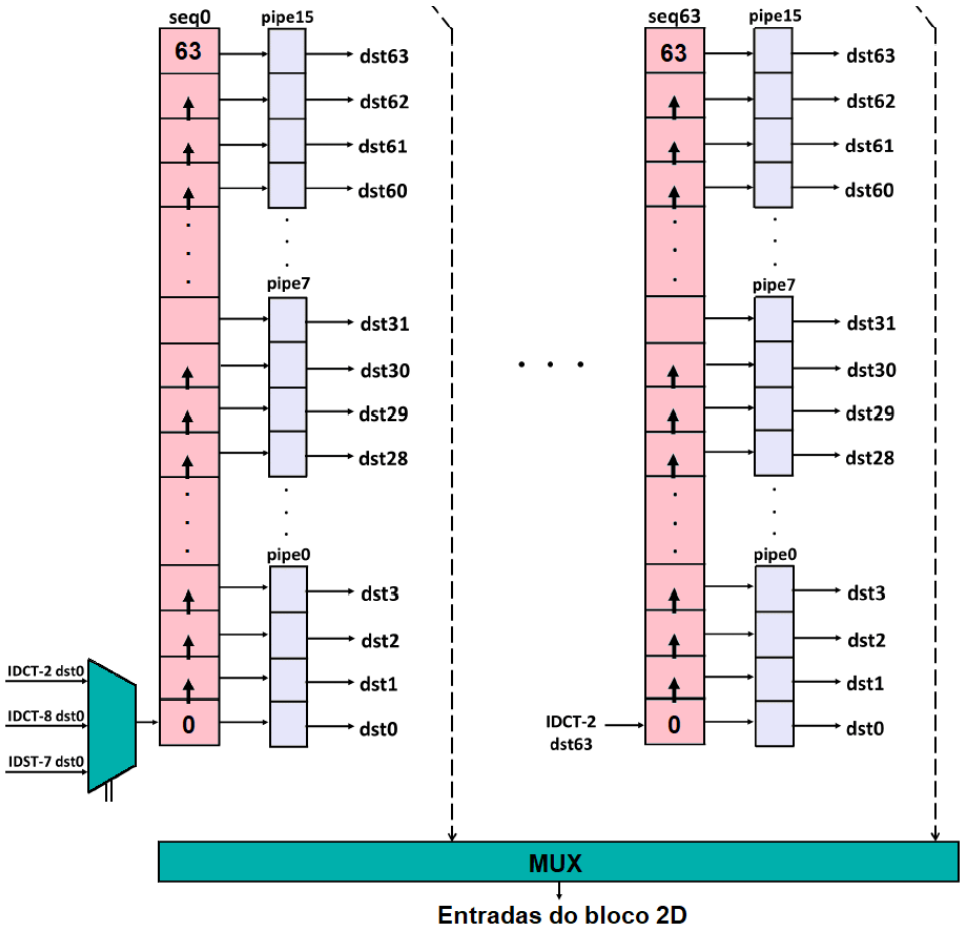

Figura 30 – Conjunto de *buffers* de transposição.

matriz de dimensões 64×64 bits. Isso significa que cada registrador sequencial possui 64 posições, e cada posição pode conter um coeficiente representado com 18 bits de precisão. A principal função desses registradores é capturar um coeficiente de cada linha do bloco intermediário, garantindo que os dados sejam armazenados de maneira ordenada e acessíveis para a etapa subsequente de cálculo.

O mecanismo de armazenamento do *buffer* opera de forma sequencial. À medida que novos coeficientes chegam, o conteúdo previamente armazenado é deslocado por uma posição, garantindo que os novos valores sejam corretamente inseridos no local apropriado. Esse deslocamento ocorre até que o registrador seja completamente preenchido de acordo com o número previamente determinado de linhas do bloco. Dessa maneira, ao final do processo, o *buffer* contém uma versão transposta da matriz original, pronta para ser utilizada na próxima etapa de processamento.

É importante ressaltar que esse conjunto de registradores forma, na prática, uma memória de transposição. Essa abordagem permite que a arquitetura lide com blocos de diferentes tamanhos e formatos, garantindo compatibilidade com as especificações das normas do padrão VVC. Além disso, o *buffer* de transposição foi projetado para minimizar a latência associada à reordenação dos dados. Como a transformada 2D exige que os coeficientes sejam reorganizados antes do processamento, a eficiência

do *buffer* impacta diretamente no desempenho global da arquitetura. Dessa forma, estratégias de otimização foram implementadas para garantir que o acesso aos coeficientes ocorra de maneira paralela sempre que possível, reduzindo o tempo total de processamento.

O conjunto de registradores sequenciais está vinculado a 16 registradores de pipeline de 4 posições. Esses registradores de pipeline desempenham um papel essencial na sincronização e manutenção dos coeficientes intermediários durante a transição entre a primeira e a segunda dimensão do cálculo. A ativação dos registradores de pipeline ocorre por meio de um sinal de carga específico, que é acionado ao término da operação da primeira dimensão. Esse sinal assegura que os valores processados sejam mantidos estáveis até que a segunda dimensão inicie seu processamento, garantindo a integridade e consistência dos dados ao longo da operação.

A geração desse sinal de carga está diretamente associada ao número de linhas da matriz de entrada. Como a estrutura da matriz pode variar de acordo com o tamanho do bloco escolhido, o circuito de controle é responsável por identificar a configuração adequada e ajustar a ativação dos registradores de pipeline em conformidade. Esse mecanismo assegura que a transição entre as dimensões ocorra de forma eficiente, evitando a perda ou sobreposição prematura dos coeficientes armazenados.

A transmissão das linhas reorganizadas para o bloco de cálculo 2D é gerenciada por um multiplexador, cuja função é selecionar e direcionar os coeficientes armazenados para os módulos responsáveis pela segunda dimensão. Esse multiplexador opera de maneira coordenada com o sinal de carga dos registradores de pipeline, garantindo que os coeficientes corretos sejam fornecidos ao bloco de cálculo no momento exato.

Além da função primária de reordenar e armazenar os coeficientes, a inclusão dos registradores de pipeline na arquitetura do *buffer* de transposição traz benefícios adicionais. Primeiramente, eles reduzem a latência associada à transmissão de dados entre as etapas do cálculo, minimizando a necessidade de acessos simultâneos à memória e permitindo que o processamento ocorra de maneira contínua. Em segundo lugar, esses registradores contribuem para a estabilidade do circuito, evitando variações indesejadas nos coeficientes intermediários e garantindo um fluxo de dados mais previsível.

6.2 Resultados

A implementação da arquitetura para as transformadas inversas seguiu a abordagem das transformadas diretas, utilizando a linguagem de descrição de *hardware* VHSIC (VHDL) e sintetizada para ASIC com o Cadence RTL Compiler 14.02, empregando células padrão TSMC de 40 nm. As operações foram derivadas do *software* de referência VVC VTM, que utiliza estrutura de borboleta parcial (partial butterfly) para

otimização e notação matricial para representação das transformadas.

Nesta etapa, não foram utilizados dados reais do VVC VTM para a síntese em ASIC. A validação da arquitetura baseou-se na extração direta de dados de entrada e saída do VTM, mediante funções incorporadas ao *software* para capturar as informações relevantes. Com esses dados, foram desenvolvidas descrições de testbench em VHDL para gerar estímulos à arquitetura, simulando seu comportamento com base nas saídas esperadas do VTM.

A verificação incluiu uma análise comparativa detalhada entre os resultados obtidos pela arquitetura e aqueles fornecidos pelo VTM. Este processo é necessário para a validação funcional, assegurando que a arquitetura atende aos requisitos de desempenho e conformidade do padrão VVC.

O cálculo da frequência de operação necessária para a síntese lógica do sistema é dado pela equação 7. Foram consideradas resoluções de vídeo Full HD (1920×1080 *pixels*), UHD 4K (3840×2160 *pixels*) e 8K (7680×4320 *pixels*), operando a 30 e 60 quadros por segundo (fps). Nesta equação, a variável resolução corresponde ao número total de *pixels* por quadro de vídeo, enquanto color_subsample descreve a taxa de subamostragem das componentes cromáticas azul e vermelha do vídeo de entrada. A variável ciclos representa o número de ciclos necessários para processar cada bloco de dados, *fps* define a taxa de quadros do vídeo, e número_de_amostras corresponde à quantidade de amostras presentes em um bloco de vídeo.

$$f = \frac{\text{resolução} \times color\_subsample \times ciclos \times fps}{número\_de\_amostras} \tag{7}$$

A equação em questão sintetiza os principais parâmetros envolvidos no cálculo da frequência alvo, conectando a resolução do vídeo, a técnica de subamostragem cromática e a exigência temporal do processamento. Ao considerar esses fatores, é possível dimensionar a frequência de operação necessária para garantir o processamento adequado dos dados.

Para o cálculo das frequências alvo, foi adotado o pior cenário de processamento, caracterizado por um bloco de entrada de 64×64 amostras, em que a partição selecionada para a divisão dos blocos é de 4×4, ou seja, o menor tamanho de bloco é utilizado para todas as partições. Neste cenário, o processamento de cada linha do bloco exige 16 ciclos de clock, totalizando 1024 ciclos para concluir a transformada na primeira dimensão (ou bloco intermediário).

A Tabela 18 apresenta os resultados da síntese lógica, detalhando tanto a área ocupada quanto o consumo total de potência para diferentes taxas de processamento, juntamente com suas respectivas frequências-alvo.

O circuito sintetizado apresentou uma área total de 7,74 kGates, e o consumo de potência foi calculado com base nas configurações padrão de comutação fornecidas

Tabela 18 – Resultado de síntese

Resolução	Frequência (MHz)	Área (kGates)	Potência Total (mW)
Full HD@30 fps	23,33	7,74	33.75
Full HD@60 fps	46,66		58,21
UHD 4K@30 fps	93,31		126,28
UHD 4K@60 fps	186,62		213,48
UHD 8K@30 fps	373,25		382,61
UHD 8K@60 fps	746,49		778,81

pela ferramenta de síntese. Como era esperado, a dissipação de potência aumentou com o aumento da frequência de operação, refletindo a necessidade de mais recursos computacionais à medida que a taxa de processamento se eleva.

Uma análise adicional envolveu a determinação da frequência máxima suportada pela arquitetura, que foi estabelecida em 1,25 GHz. No entanto, ao operar nesta frequência, a dissipação de potência alcançou valores consideravelmente elevados, cerca de 1.110 mW, o que indicou que, embora a frequência máxima fosse alcançável, ela não era a mais eficiente em termos de consumo de energia. Esse resultado sugere que a vazão necessária para a maior resolução, neste caso, a resolução UHD 8K a 60 *fps*, poderia ser alcançada a uma frequência significativamente inferior, tornando desnecessária a operação na frequência máxima.

Especificamente, para a maior resolução considerada (UHD 8K a 60 *fps*), a arquitetura proposta foi capaz de atingir a frequência de 746,49 MHz, com uma dissipação total de potência de 778,81 mW. Esse desempenho mostra a eficiência do projeto, que é capaz de entregar o desempenho desejado com uma dissipação de potência substancialmente menor do que a observada na frequência máxima. A grande área ocupada pelo circuito é atribuída à implementação do *buffer* de 64×64 e do bloco de tamanho 64 utilizado na Transformada Discreta Cosseno Inversa (IDCT-II) em ambas as dimensões. Estes blocos, por sua vez, requerem um número elevado de registradores, além de um grande volume de operações de multiplicação e adição, que são fundamentais para o processamento de vídeos de alta resolução.

6.3 Resumo do Capítulo

Este capítulo apresentou o desenvolvimento de uma arquitetura de *hardware* dedicada à execução das transformadas inversas previstas no padrão *Versatile Video Coding*, com foco na eficiência computacional, baixo consumo energético e compati-

bilidade com diferentes tamanhos e formatos de blocos. As transformadas inversas, embora tradicionalmente associadas ao processo de decodificação, também desempenham papel fundamental na etapa de codificação, especialmente no laço de reconstrução, onde são utilizadas para o cálculo do custo taxa-distorção. A principal diferença entre transformadas diretas e inversas foi destacada: enquanto as diretas são testadas de forma exaustiva para cada bloco candidato, as inversas são aplicadas uma única vez ao bloco selecionado. Contudo, durante a codificação, cada teste de transformada direta exige a execução correspondente da transformada inversa, elevando o custo computacional do codificador. Neste contexto, os modelos preditivos apresentados no Capítulo 5 tornam-se relevantes também para esta etapa, pois, ao reduzir o número de combinações testadas, impactam positivamente o consumo energético global do codificador. A arquitetura proposta oferece suporte às transformadas IDCT-II, IDCT-VIII e IDST-VII, com blocos variando de 4×4 a 64×64 para a IDCT-II, e até 32×32 para as demais, além de permitir combinações horizontais e verticais como exigido pela ferramenta MTS. O processamento é realizado em duas etapas unidimensionais, intercaladas por um *buffer* de transposição otimizado. O projeto considera ativação seletiva de módulos de transformada conforme o tamanho do bloco, e é implementado com controle de fluxo eficiente e suporte a paralelismo parcial. O *buffer* de transposição foi projetado com uma estrutura composta por registradores sequenciais e pipelines, otimizando a reordenação dos dados entre as duas dimensões da transformada e reduzindo a latência. A arquitetura foi descrita em VHDL e sintetizada em tecnologia ASIC de 40 nm, com validação funcional baseada em estímulos extraídos do codificador de referência VTM. Os resultados de síntese mostraram que a arquitetura atinge frequências superiores a 700 MHz, sendo capaz de processar vídeos em resoluções de até 8K a 60 fps com dissipação de potência inferior a 800 mW. A análise evidenciou que a principal fonte de área e consumo está nos blocos de maior dimensão e no *buffer* de transposição. Ainda assim, a proposta se mostrou eficiente e viável para aplicações com altas demandas de processamento e restrições energéticas.

Neste capítulo, foram adotadas as mesmas técnicas e ferramentas de desenvolvimento descritas no Capítulo 6. As arquiteturas foram modeladas em VHDL, validadas funcionalmente com o simulador *ModelSim* e sintetizadas com o *Cadence RTL Compiler*, utilizando a tecnologia de células padrão TSMC de 40nm. A análise dos resultados também seguiu os mesmos critérios, com foco nas métricas de área projetada e consumo energético. Assim como no capítulo anterior, a ênfase está na avaliação da viabilidade das arquiteturas quanto à eficiência em termos de ocupação de silício e redução de potência, aspectos essenciais para aplicações embarcadas e cenários com restrições energéticas.

7 ARQUITETURA DE *HARDWARE* PARA TRANSFORMADAS DIRETAS

As transformadas diretas no VVC desempenham um papel fundamental ao converter os coeficientes de resíduo de um bloco do domínio temporal para o domínio da frequência, facilitando a compressão dos dados. Diferentemente das transformadas inversas, que são aplicadas no codificador e decodificador, as transformadas diretas são executadas exclusivamente no codificador, influenciando diretamente a eficiência da compressão. Conforme discutido ao longo desta tese, essa etapa do VVC é computacionalmente intensiva, pois, para que a melhor eficiência de compressão seja atingida, é exigida a avaliação exaustiva de todas as combinações possíveis de tipos de transformadas e tamanhos de blocos suportados pelo padrão. Diante desse contexto, este capítulo se dedica ao desenvolvimento e à análise de arquiteturas de *hardware* otimizadas para a implementação das transformadas diretas.

7.1 Arquitetura da DCT-II

Esta seção apresenta um conjunto de arquiteturas para a implementação do módulo MTS no codificador de vídeo VVC, englobando os três tipos de transformadas unidimensionais suportadas pela ferramenta, bem como suas possíveis combinações na composição das transformadas bidimensionais. As arquiteturas desenvolvidas garantem compatibilidade com uma ampla gama de tamanhos de blocos, incluindo configurações quadradas e retangulares, nos formatos 4×4, 4×8, 4×16, 4×32, 8×4, 8×8, 8×16, 8×32, 16×4, 16×8, 16×16, 16×32, 32×4, 32×8, 32×16 e 32×32. A Figura 31 ilustra as matrizes de coeficientes da DCT-II para diferentes tamanhos de bloco (4, 8, 16 e 32 *points*), cujos valores numéricos detalhados encontram-se no Apêndice B.

Observa-se que a matriz correspondente a 32 *points* contém, em sua estrutura, a matriz de 16 *points*, a qual, por sua vez, incorpora a matriz de 8 *points*, que também é parcialmente composta pela matriz de 4 *points*. Essa hierarquia estrutural permite a identificação de operações redundantes, possibilitando sua reutilização nas matrizes de maior dimensão. Essa propriedade pode ser explorada para reduzir a complexi-

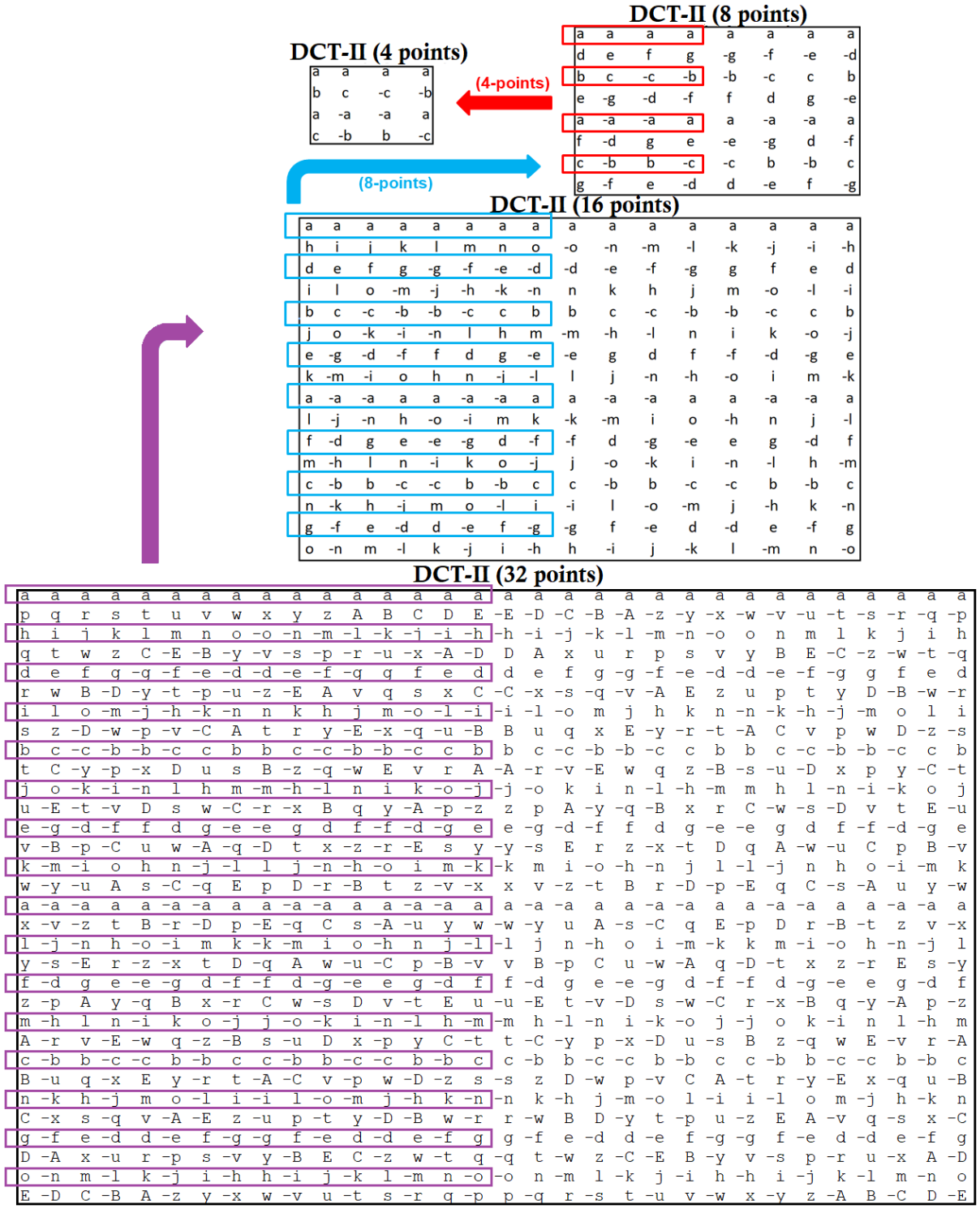

Figura 31 – Coeficientes da DCT-II para 4, 8, 16 e 32 *points*.

dade computacional e otimizar o consumo de recursos das operações na implementação do módulo de transformadas. A Figura 32 ilustra essa reutilização de operações, evidenciando a possibilidade de compartilhamento de cálculos entre diferentes tamanhos de bloco, contribuindo para uma arquitetura mais eficiente em termos de área e desempenho.

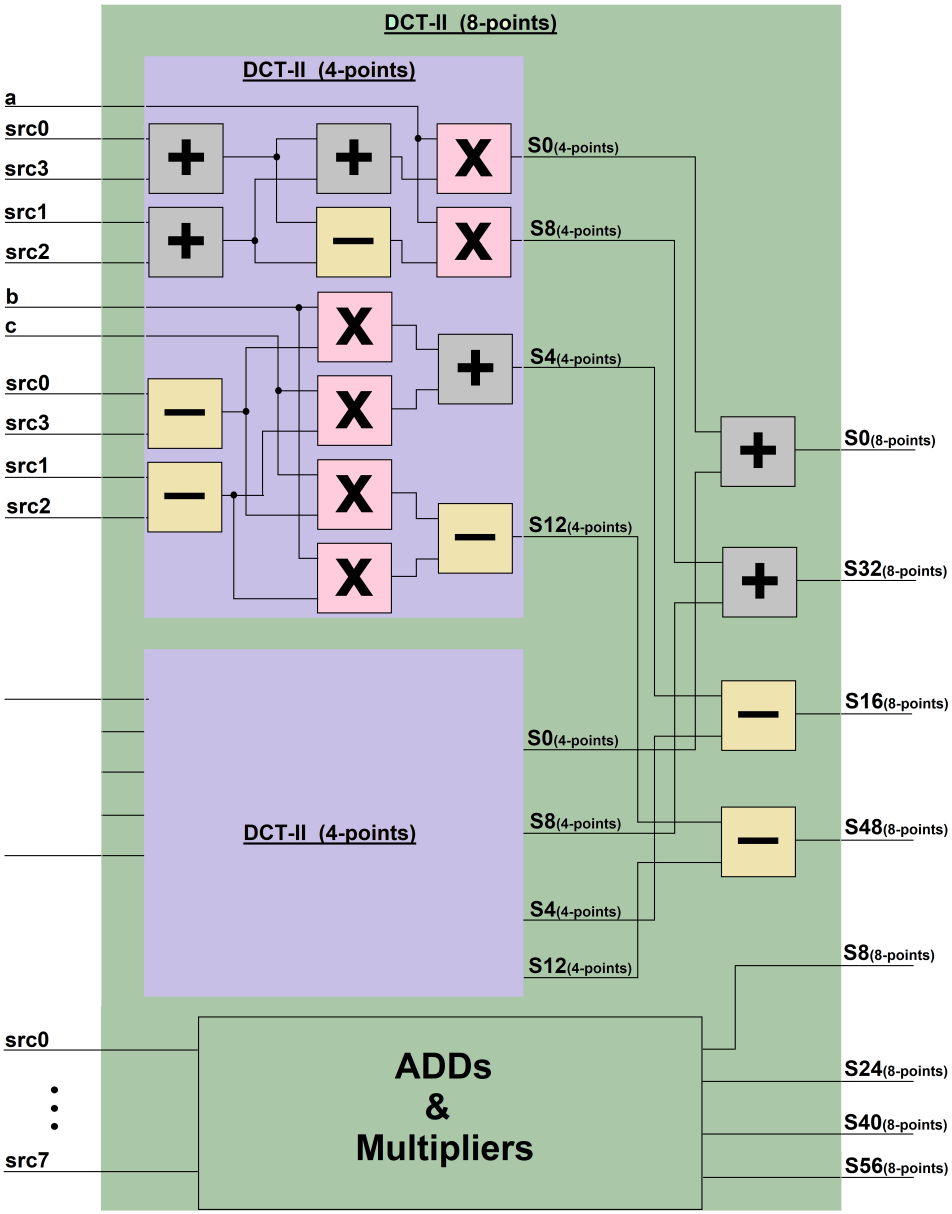

Figura 32 – Reutilização de operação entre as transformadas DCT-II de 4 e 8 *points*.

Na Figura 32, os blocos destacados em lilás representam a DCT-II de 4 *points*, cujas entradas (*src0* a *src3*) geram quatro saídas que são reutilizadas no cálculo da DCT-II de 8 *points* (destacada em verde na mesma figura). Ao combinar duas DCT-II de 4 *points*, obtém-se um conjunto inicial de quatro saídas para a DCT-II de 8 *points*. No entanto, as quatro saídas remanescentes dessa transformada ainda exigem cálculos adicionais, sendo necessários somadores e multiplicadores específicos para sua

obtenção. Esse mesmo princípio hierárquico se estende às transformadas DCT-II de 16 e 32 *points*, onde a reutilização ocorre a partir das transformadas imediatamente inferiores, ou seja, a DCT-II de 8 e 16 *points*, respectivamente.

7.2 Arquitetura das DCT-VIII e DST-VII

No caso dos coeficientes das transformadas DCT-VIII e DST-VII, não é possível identificar padrões de similaridade entre diferentes dimensões de matrizes que permitam a reutilização direta de cálculos, como ocorre na DCT-II. No entanto, ao comparar diretamente as matrizes dessas duas transformadas, observa-se uma notável correspondência estrutural entre seus coeficientes. Em diversas instâncias, os valores apresentam um comportamento espelhado, frequentemente acompanhados de inversões de sinal. Essa característica sugere a possibilidade de otimizações no processo de implementação, explorando a relação entre as operações envolvidas. A Figura 33(a) ilustra esse fenômeno, apresentando um exemplo das matrizes de coeficientes para DCT-VIII e DST-VII de 4 *points*, juntamente com as respectivas operações associadas.

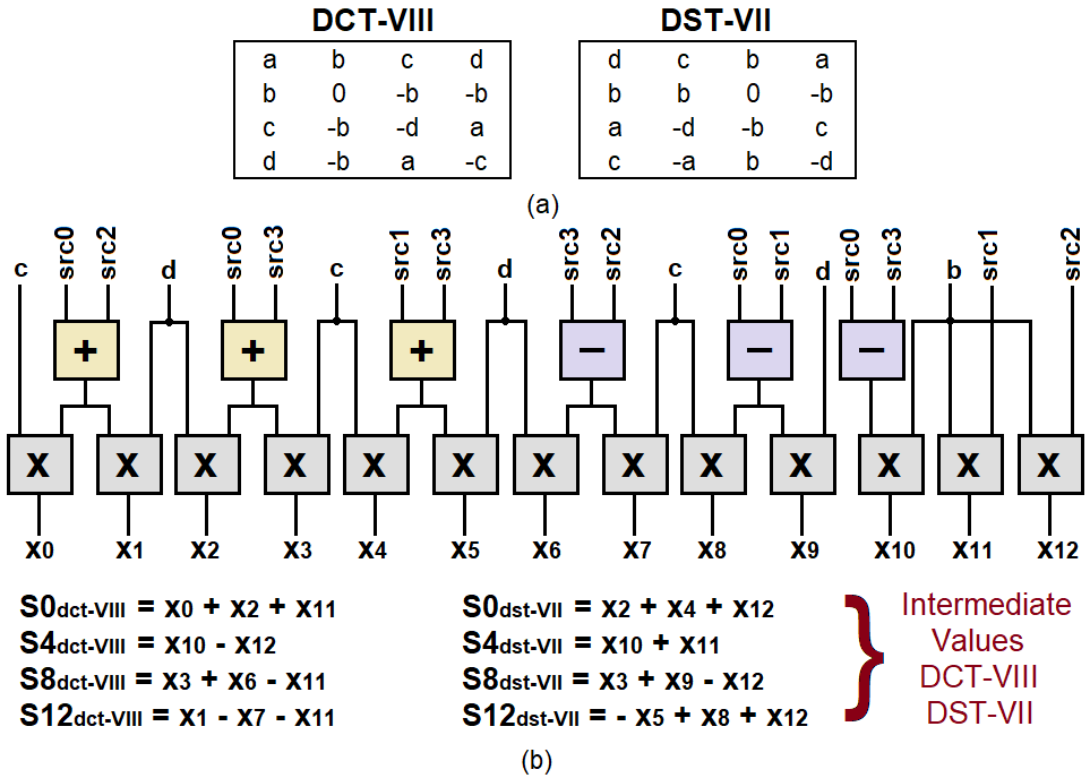

Figura 33 – DCT-VIII e DST-VII (a) coeficientes e (b) operações.

Ao analisar a estrutura matricial das transformadas DCT-VIII e DST-VII, observa-se que os coeficientes da primeira linha da DCT-VIII apresentam uma disposição espelhada em relação à primeira linha da DST-VII. Além disso, na segunda linha de ambas as transformadas, verifica-se não apenas a reflexão dos coeficientes, mas também a inversão de seus sinais. Essa característica permite explorar a redundância algébrica

entre as duas transformadas, possibilitando a reutilização eficiente de operadores matemáticos. Com base nessa observação, foi desenvolvida uma arquitetura otimizada para a MTS, na qual um número significativo de operadores é compartilhado entre os diferentes tipos de transformadas, reduzindo assim o custo de implementação. A Figura 33(b) ilustra essa abordagem, evidenciando a estratégia adotada para a reutilização dos operadores.

7.3 Arquitetura Proposta para MTS

A arquitetura proposta para a MTS unificada integra as duas arquiteturas discutidas nas seções anteriores, consolidando um design otimizado para a implementação das transformadas no VVC. Essa arquitetura, ilustrada na Figura 34, foi desenvolvida para processar blocos residuais de até 32×32, permitindo a divisão em sub-blocos conforme as diversas estruturas de particionamento definidas pelo padrão. Assim como o cálculo da transformada inversa, o cálculo da transformada direta é conduzido de maneira sistemática, operando sequencialmente sobre as linhas da matriz de entrada. O tamanho do bloco processado é diretamente determinado pelo número de colunas dessa matriz, pois este valor deve corresponder ao número de linhas da matriz de coeficientes da transformada. Consequentemente, a estrutura da matriz de entrada não apenas define a dimensão do bloco a ser transformado, mas também impacta diretamente a quantidade de ciclos de clock necessários para a conclusão do processamento.

A estrutura recebe 32 entradas (*src0* a *src31*), representando a primeira linha de amostras do bloco residual, além de duas entradas correspondentes às dimensões do bloco largura (*WIDTH*) e altura (*HEIGHT* e uma entrada adicional que especifica o índice da MTS (*MTS-IDX*). Este índice indica o tipo de combinação de transformadas que será utilizada nas direções horizontal e vertical do bloco. Todas essas informações são encaminhadas ao módulo de controle, responsável por habilitar dinamicamente o núcleo da transformada apropriada com base no valor de *MTS-IDX* e no tamanho da partição do bloco atual, garantindo flexibilidade no processamento das transformadas.

Todas as saídas das transformadas são direcionadas a um multiplexador (destacado em verde escuro na Figura 34), responsável por encaminhar os coeficientes ao *buffer* de transposição de acordo com a transformada selecionada (representado nas cores rosa e cinza na Figura 34). Esse *buffer* é composto por 32 registradores de deslocamento (com 32 posições cada) e 128 registradores paralelos (com 4 posições cada), cuja função é armazenar temporariamente os coeficientes gerados pela transformada horizontal e redistribuí-los coluna por coluna para a transformada vertical (bloco *Second 1D*, em amarelo na Figura 34).

A estrutura desse *buffer* foi projetada para assegurar a correta organização e reor-

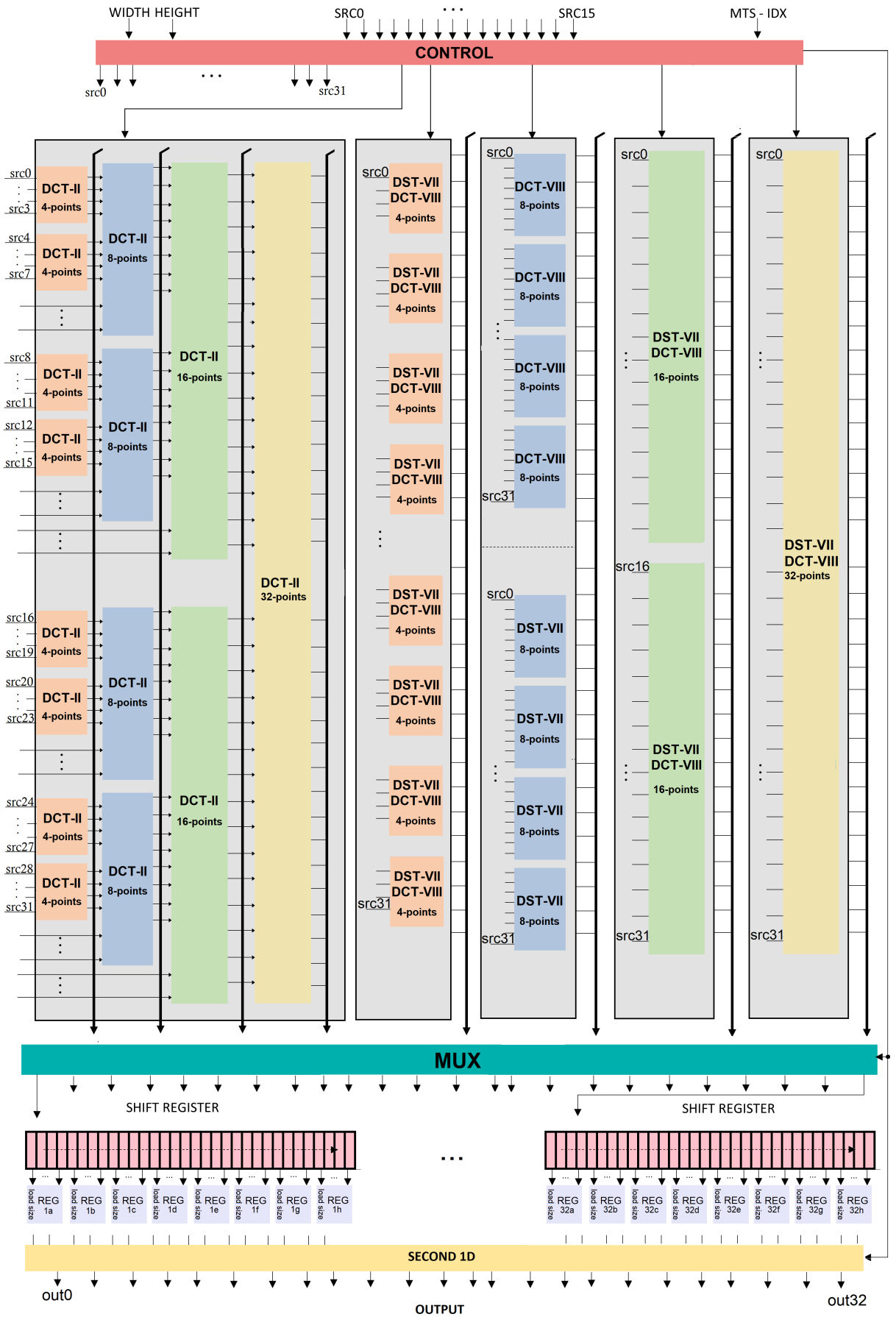

Figura 34 – Projeto de *hardware* da MTS proposta.

denamento dos coeficientes intermediários, permitindo que a segunda etapa da transformação ocorra de maneira eficiente e sem comprometer o desempenho do sistema. A cada ciclo de *clock*, os registradores de deslocamento (indicados em rosa na Figura 34) armazenam os coeficientes da saída horizontal, considerando o tamanho do bloco previamente definido. Por exemplo, se o bloco residual for de dimensão 4×4, apenas as quatro primeiras posições dos registradores de deslocamento serão preenchidas. Em seguida, o módulo de controle transfere esses valores para os registradores paralelos (*1a* até *31a*) por meio do sinal de carregamento (*load size*), garantindo que a transformação na segunda dimensão seja realizada de forma ordenada e precisa.

Por exemplo, considerando um bloco de tamanho 4×4 e um índice *MTS-IDX* = 0, as 32 amostras de entrada (correspondentes à primeira linha do bloco residual) são distribuídas entre os oito módulos DCT-II de 4 *points* (destacados em laranja na Figura 34). Cada um desses módulos processa a transformada 1D e, em um único ciclo de *clock*, gera as seguintes saídas:

- Posições 0, 32, 64 e 96 da primeira coluna do bloco 32×32;
- Posições 4, 36, 68 e 100 da quinta coluna do bloco 32×32;
- Posições 8, 40, 72 e 104 da nona coluna do bloco 32×32;
- Posições 12, 44, 76 e 108 da décima terceira coluna do bloco 32×32;
- Posições 16, 48, 80 e 112 da décima sétima coluna do bloco 32×32;
- Posições 20, 52, 84 e 116 da vigésima primeira coluna do bloco 32×32;
- Posições 24, 56, 88 e 120 da vigésima quinta coluna do bloco 32×32;
- Posições 28, 60, 92 e 124 da vigésima nona coluna do bloco 32×32.

No segundo ciclo de *clock*, as próximas 32 amostras são carregadas, correspondendo à segunda linha do bloco residual 32×32. Esse processo se repete, e ao final do quarto ciclo, a arquitetura terá concluído a transformada para oito blocos de 4×4, abrangendo as quatro primeiras linhas do bloco 32×32.

Para completar o cálculo da transformada 1D em todo o bloco residual 32×32, a arquitetura requer um total de 36 ciclos de *clock*, dos quais 32 ciclos são utilizados para processar todas as linhas do bloco, enquanto 4 ciclos adicionais são necessários para preencher o pipeline. A partir do quarto ciclo de *clock*, os valores correspondentes às primeiras quatro linhas do bloco intermediário 32×32 já estão disponíveis, permitindo que esses coeficientes sejam imediatamente utilizados como entradas para a segunda etapa da transformada 1D, correspondendo à dimensão vertical do processamento. A latência total da arquitetura para computar ambas as dimensões do bloco 32×32 é de

68 ciclos de *clock*. Após essa latência inicial, a transformada 2D completa pode ser processada em 32 ciclos de *clock*, garantindo eficiência na execução das operações.

7.4 Implementações em *hardware*

Nesta seção, são descritas as diferentes versões de implementação em *hardware* da arquitetura proposta, incluindo aquela com integração dos modelos preditivos baseados em aprendizado de máquina. Também são apresentados os resultados obtidos em cada uma das etapas do desenvolvimento. Todas as versões foram descritas em VHDL e submetidas ao processo de síntese lógica utilizando a ferramenta *Cadence RTL Compiler* 14.02. A síntese foi realizada com tecnologia de células padrão de 40 nm da TSMC, possibilitando uma análise detalhada dos parâmetros físicos das arquiteturas, como área ocupada, consumo de energia e desempenho.

7.4.1 Arquitetura para processamento de todos tamanhos de bloco e todos modos MTS

No software de referência do codificador VVC, a etapa de MTS impõe a necessidade de avaliação exaustiva de todas as combinações possíveis entre tipos de transformadas e tamanhos de blocos, o que eleva significativamente os requisitos computacionais e o consumo energético do sistema. Para a realização da síntese voltada à estimativa de dissipação de potência e cálculo de área, adotou-se como referência o pior cenário de operação, definido pela exigência de suportar, em tempo real, a execução da arquitetura considerando as 16 combinações distintas de tamanhos de blocos e as 5 possíveis combinações de transformadas previstas pela ferramenta MTS. Esse cenário foi utilizado para o cálculo da frequência-alvo, assegurando que a arquitetura seja capaz de atender aos requisitos de diferentes resoluções e taxas de quadros de vídeo em aplicações práticas.

As frequências-alvo foram determinadas com base na taxa mínima de processamento exigida para garantir a codificação em tempo real de diferentes resoluções de vídeo, conforme definido pela Eq. (8). As resoluções e taxas de quadros selecionadas neste estudo são: HD 720p a 20, 30 e 60 fps; FHD 1080p a 30, 50 e 60 fps; e UHD 4K a 30, 50 e 60 fps. Essas taxas foram escolhidas por refletirem os valores efetivamente utilizados nas sequências de teste padronizadas pela CTC. A resolução de vídeo refere-se à dimensão espacial de cada quadro (altura × largura), enquanto a taxa de quadros (*frames per second* – fps) representa a frequência com que os quadros são exibidos.

$$f = \frac{largura \cdot altura \cdot fps \cdot ciclos}{amostras} \tag{8}$$

O parâmetro *ciclos* é definido como o número de ciclos de *clock* necessários para

processar um bloco de 32×32, multiplicado pelo total de combinações de transformadas (5) e pelo número de variações possíveis de tamanhos de bloco (16). Já o parâmetro *amostras* corresponde à quantidade total de amostras em um bloco 32×32, resultando em 1024 amostras.

7.4.1.1 Resultados de Síntese

Nesta análise, a dissipação de potência foi estimada considerando a atividade de comutação padrão da ferramenta Cadence RTL Compiler, que assume uma taxa de transição de 20% em cada entrada da arquitetura (PERLEBERG et al., 2018). Essa abordagem permite obter uma estimativa preliminar do consumo energético, garantindo uma avaliação inicial da viabilidade da implementação em termos de eficiência energética e desempenho. Para a estimativa da área da arquitetura, adotou-se como métrica o número equivalente de portas lógicas NAND de duas entradas. Essa abordagem permite uma comparação justa com outros trabalhos que utilizam diferentes tecnologias de fabricação, uma vez que a área total é obtida pelo quociente entre a área ocupada pelas células padrão e a área de uma única porta NAND-2.

A Tabela 19 apresenta os resultados da potência total extraída da síntese realizada para cada *throughput* e sua respectiva frequência-alvo.

Tabela 19 – Resultados de síntese para o pior caso

Resolução	Frequência (MHz)	Área (kgates)	Potência (mW)
HD@20fps	46,08	1544	331,67
HD@30fps	69,12		687,74
HD@60fps	138,24		642,28
FHD@30fps	155,52		974,00
FHD@50fps	259,20		1037,60
FHD@60fps	311,04		1170,16
4K@30fps	622,08		3174,44

Os resultados na Tabela 19 mostram que, como esperado, a potência total aumenta à medida que a frequência aumenta, então é natural que a maior potência total seja para o *throughput* 4K@30fps. Vale ressaltar também que, como a frequência dobra de valor, a potência total também apresenta um comportamento semelhante, exceto em 4K@30fps, onde a potência não aumenta proporcionalmente à frequência.

Para possibilitar uma comparação equitativa entre os resultados obtidos com vídeos codificados em diferentes taxas de quadros por segundo, foi adotada uma estratégia de normalização energética. A métrica escolhida para essa análise foi a energia por amostra (ou por *pixel* processado), a qual expressa a quantidade de energia con-

sumida pela arquitetura de *hardware* para processar individualmente cada *pixel* de vídeo.

Além dessa normalização por pixel, todos os valores foram também ajustados com base em uma referência comum: a taxa de 60 quadros por segundo para vídeos na maior resolução definida pela CTC (3840×2160 *pixels*). Essa padronização visa garantir que os resultados de consumo energético reflitam apenas as características da arquitetura, independentemente da duração do vídeo ou da frequência de quadros original de cada sequência. Tal abordagem é especialmente relevante em aplicações de vídeo em tempo real, nas quais dispositivos operam sob diferentes restrições de taxa de quadros, resoluções e requisitos energéticos. A Tabela 20 apresenta estes resultados.

Tabela 20 – Resultados de energia e média para o pior caso

Resolução de vídeo	**Energia por *Pixel* (*n*J)**	**Energia por *frame* (*m*J)**	**Média (*n*J)**
HD (1280×720 @20fps)	17,99	16,58	**18,20**
HD (1280×720 @30fps)	24,87	22,92	
HD (1280×720 @60fps)	11,62	10,70	
FHD(1920×1080 @30fps)	15,66	32,47	**11,70**
FHD (1920×1080 @50fps)	10,00	20,76	
FHD (1920×1080 @60fps)	9,4	19,50	
4K (3840×2160 @30fps)	12,76	105,81	**12,76**
4K (3840×2160 @50fps)	Ñ suporta	Ñ suporta	
4K (3840×2160 @60fps)	Ñ suporta	Ñ suporta	

A Tabela 20 apresenta os resultados de consumo energético por *pixel* e por quadro para diferentes resoluções e taxas de quadros, além da média de energia por *pixel* calculada para cada resolução. Os dados indicam que o consumo energético por *pixel* não depende apenas da resolução, mas também da taxa de quadros (fps), que influencia diretamente a frequência de operação necessária para processar os vídeos em tempo real. Para a resolução HD, observa-se que o valor de energia por *pixel* varia de forma expressiva conforme o fps: 17,99 nJ para 20 fps, 24,87 nJ para 30 fps e 11,62 nJ para 60 fps. Esses resultados sugerem que, em taxas mais baixas, apesar da menor exigência de processamento por segundo, o aproveitamento energético é menos eficiente devido ao maior tempo de atividade relativo por *frame*. Em contrapartida, taxas mais elevadas permitem um melhor paralelismo e utilização da arquitetura, reduzindo o custo energético por *pixel*.

A resolução Full HD apresentou o melhor desempenho em termos de eficiência energética, com uma média de 11,70 nJ por *pixel*, destacando-se como a mais eficiente entre os cenários analisados. Isso indica que, para a arquitetura proposta, há um ponto de equilíbrio favorável entre carga computacional e frequência de operação nessa resolução. Para a resolução 4K, a arquitetura foi capaz de operar apenas até 30 fps, registrando um consumo de 12,76 nJ por *pixel*. As configurações de 50 fps e 60 fps não foram suportadas, o que evidencia uma limitação estrutural da arquitetura para frequências mais elevadas exigidas por essas resoluções. Mesmo assim, os resultados em 4K a 30 fps demonstram que a arquitetura mantém um desempenho competitivo, ainda que com maior custo energético em relação ao FHD.

De maneira geral, observa-se que a eficiência energética da arquitetura é superior em resoluções intermediárias, como o FHD, particularmente em taxas de quadros mais elevadas dentro da faixa suportada. Este comportamento ocorre porque, em resoluções mais altas, o consumo fixo associado aos circuitos de controle e suporte da arquitetura, que permanecem ativos independentemente da carga de processamento, é diluído em um maior volume de pixels processados. Em contrapartida, em resoluções mais baixas, como o HD, esse consumo fixo passa a representar uma fração mais significativa do consumo total, resultando em um aumento relativo do consumo energético por *pixel*. Já em resoluções muito altas, como o 4K a 50 ou 60 fps, os requisitos de frequência tornam-se proibitivos, o que reforça a necessidade de estratégias complementares de otimização. Esses resultados oferecem subsídios importantes para o dimensionamento de arquiteturas futuras e para a adoção seletiva de mecanismos de aceleração, como os modelos preditivos, em contextos com maiores exigências de desempenho e restrições energéticas.

7.4.2 Arquitetura para processamento de blocos 32×32 e todos modos MTS

Na segunda implementação da arquitetura, buscou-se refletir de forma mais realista um cenário de codificação em tempo real. Em particular, apesar de a arquitetura ainda suportar qualquer tamanho de bloco, considerou-se o cenário em que o tamanho dos blocos de entrada está restrito a 32×32 *pixels*, uma vez que este foi também o tamanho utilizado no processo de coleta de dados para treinamento de modelos preditivos. Como consequência, o conjunto de combinações de transformadas e tamanhos de blocos considerados no cálculo da frequência-alvo foi significativamente reduzido, o que permitiu recalcular essa frequência de operação de maneira mais condizente com o perfil real de uso em aplicações de tempo real. Essa abordagem intermediária viabilizou uma segunda implementação da arquitetura, com menores requisitos de processamento em comparação ao pior caso. Assim, esta etapa serve como ponte entre a arquitetura original e a terceira implementação, que incorpora as decisões preditivas, permitindo uma avaliação mais progressiva e fundamentada dos ganhos

obtidos ao longo do processo de otimização.

A equação empregada para o cálculo da frequência permanece inalterada em relação àquela descrita em 8. No entanto, ao contrário da etapa anterior, na qual foram consideradas até 16 combinações de tamanhos de blocos para a transformada (representando o pior caso da MTS), neste novo cenário adotou-se como referência apenas o tamanho de bloco 32×32.

Adicionalmente, as novas estimativas também consideram a taxa mínima de processamento necessária para diferentes resoluções e taxas de quadros por segundo, conforme detalhado a seguir:

- HD: 20 fps, 30 fps e 60 fps
- Full HD: 30 fps, 50 fps e 60 fps
- Ultra HD 4K: 30 fps, 50 fps e 60 fps

7.4.2.1 Resultados de síntese

Assim como na primeira etapa do trabalho, a dissipação de potência da arquitetura foi estimada com base na atividade de comutação padrão da ferramenta *Cadence RTL Compiler*, a qual adota uma taxa de transição de 20% para as portas lógicas. Para a estimativa da área, foi utilizada como métrica o número equivalente de portas NAND de duas entradas, permitindo uma avaliação comparativa da complexidade do *hardware*. A Tabela 21 apresenta os resultados de potência e frequência, considerando as diferentes resoluções de vídeo.

Tabela 21 – Resultados de síntese

Resolução de vídeo	Power (mW)	Frequência (MHz)	Área (KGates)
HD (1280×720 @20fps)	36,78	2,88	1544
HD (1280×720 @30fps)	65,02	4,32	
HD (1280×720 @60fps)	124,14	8,64	
FHD(1920×1080 @30fps)	138,30	9,72	
FHD (1920×1080 @50fps)	222,60	16,2	
FHD (1920×1080 @60fps)	228,71	19,44	
4K (3840×2160 @30fps)	350,13	38,88	
4K (3840×2160 @50fps)	505,59	64,80	
4K ($3840 \times 2160$0 @60fps)	839,89	77,76	

Nesta análise também foi adotada a energia por *pixel* para efeitos de comparação. A Tabela 22 apresenta os resultados obtidos para o consumo de energia, já normali-

zados segundo os critérios estabelecidos, assim como a média correspondente para cada resolução analisada.

Tabela 22 – Resultados de energia e média

Resolução de vídeo	Energia por *Pixel* (nJ)	Energia normalizada (mJ)	Média (mJ)
HD (1280×720 @20fps)	1,99	993,11	**2,20**
HD (1280×720 @30fps)	2,35	1170,43	
HD (1280×720 @60fps)	2,24	1117,23	
FHD(1920×1080 @30fps)	2,22	1106,43	**2,07**
FHD (1920×1080 @50fps)	2,15	1068,48	
FHD (1920×1080 @60fps)	1,84	914,85	
4K (3840×2160 @30fps)	1,41	700,26	**1,44**
4K (3840×2160 @50fps)	1,22	606,71	
4K (3840×2160 @60fps)	1,69	839,89	

A Tabela 22 apresenta os resultados energéticos obtidos com a segunda implementação da arquitetura, em que se considerou a execução apenas para blocos de 32×32. Embora a arquitetura permaneça inalterada para todas as resoluções e taxas de quadros, a frequência de operação varia de acordo com a demanda de processamento imposta por cada cenário. Nesse contexto, observa-se que a energia por *pixel* tende a ser menor em resoluções mais altas (como 4K), pois o maior volume de dados processados por segundo permite diluir o consumo energético total ao longo de um número maior de *pixels*. Isso resulta em uma menor energia média por *pixel*, mesmo que a arquitetura seja executada mais vezes por segundo. Em resoluções mais baixas, como HD, a frequência de operação é relativamente alta para um volume menor de dados, o que acentua o impacto do *overhead* fixo e contribui para um consumo proporcionalmente maior por *pixel*. Esses resultados evidenciam que uma possível simplificação da MTS e a uniformização do tamanho de bloco podem contribuir para ganhos energéticos relevantes, sobretudo em cenários de maior resolução, onde o uso da arquitetura é mais intensivo e eficiente.

7.4.3 Arquitetura para processamento de blocos 32×32 e decisão de modo MTS com modelos preditivos

Nesta última etapa do trabalho, os modelos preditivos baseados em aprendizado de máquina foram integrados ao codificador de referência VVC com o objetivo de obter dados reais de execução, que pudessem ser utilizados como base para a avaliação

da arquitetura de *hardware* proposta. A partir dessa integração, foram extraídas informações relevantes diretamente do software VTM modificado, incluindo o número de transformadas testadas, o tamanho do bloco, o MTS-INDEX e os coeficientes da matriz de resíduo, que serviram para o recálculo das frequências-alvo de operação. Essas frequências, empregadas nas estimativas de desempenho e consumo energético, foram determinadas conforme a equação (8), considerando blocos fixos de tamanho 32×32. Diferentemente da etapa na qual foi adotado o pior caso da MTS (com 16 combinações distintas de tamanhos), esta abordagem representa de forma mais realista o comportamento da arquitetura com a aplicação das estratégias preditivas. Os dados obtidos nesta fase foram, então, utilizados como entradas reais para a síntese da arquitetura em *hardware*, possibilitando uma análise mais precisa dos parâmetros físicos, como potência, área e energia.

Para fins de comparação, foram implementadas duas arquiteturas de *hardware* baseadas nos fluxos de execução do VTM original e modificado (conforme representado nas Figuras 24 e 25 da Seção 5.1). Essas implementações têm como objetivo principal avaliar o impacto da introdução dos modelos preditivos nos parâmetros físicos da arquitetura, especialmente em relação à área ocupada e ao consumo energético.

A implementação em *hardware* concentrou-se nos fluxos de execução correspondentes ao VTM original e modificado, considerando que os modelos preditivos, por serem compostos por decisões condicionais simples, podem ser facilmente integrados de forma modular em versões subsequentes. Dessa forma, priorizou-se a análise estrutural e de desempenho da arquitetura, mantendo a compatibilidade com a lógica de decisão introduzida pelos modelos de aprendizado de máquina.

Como parte do processo de avaliação, utilizou-se o conjunto de vídeos definido pela CTC, com diferentes resoluções e características. Foram extraídos valores de 24 quadros de cada sequência de vídeo, porém, para a etapa de síntese, foi selecionado apenas o primeiro quadro de cada vídeo para servir como entrada real. Essa escolha foi motivada por limitações práticas: a extração e o processamento de dados reais para todos os quadros resultariam em arquivos de entrada de grande tamanho, com elevado custo de armazenamento e longos tempos de execução na ferramenta de síntese, tornando o processo inviável em termos computacionais. Apesar disso, a seleção de um único *frame* ainda permite uma análise estatisticamente representativa, uma vez que este contém um número considerável de blocos 32×32 e apresenta padrões típicos de variação espacial que se repetem ao longo dos quadros subsequentes.

Para ambas versões original e a modificada com o modelo preditivo foi realizada a extração de uma variável denominada *"MTS_count"*, a qual representa o número total de vezes em que o índice de MTS foi acionado durante a codificação dos blocos 32×32. No fluxo original do VTM, o cálculo da frequência-alvo parte do pressuposto de

que o índice é chamado 100% das vezes, isto é, para cada bloco 32×32 são testadas todas as cinco transformadas possíveis. Já na versão modificada, a decisão sobre quais transformadas devem ser testadas é tomada pelo modelo preditivo, o qual pode reduzir significativamente o número de chamadas ao índice e, portanto, o número de operações realizadas pela arquitetura. A frequência-alvo da versão modificada foi calculada a partir da redução percentual de contagens da variável *MTS_count*. A partir do valor de *MTS_count*, foi possível quantificar diretamente a economia proporcionada pela integração do modelo.

7.4.3.1 Resultados de Síntese

Conforme discutido no Capítulo 5, os blocos codificados nos modos de predição intraquadro e interquadros apresentam características estruturais e estatísticas significativamente distintas, o que justifica a análise separada de seus impactos sobre a arquitetura. Dessa forma, os resultados de síntese lógica referentes à frequência de operação e ao consumo de potência foram organizados de maneira individualizada para cada tipo de predição. As Tabelas 23 e 24 apresentam, respectivamente, os resultados obtidos para os modos interquadros e intraquadro, permitindo uma comparação detalhada entre as diferentes sequências de vídeo processadas. Essa separação possibilita uma avaliação mais precisa do comportamento da arquitetura em cenários específicos de codificação, além de destacar eventuais variações na eficiência energética e no desempenho em função do tipo de predição adotado.

Como apresentado nas tabelas anteriores, os resultados de frequência de operação e de potência estimada são detalhados para cada sequência de vídeo, em ambos os modos de predição intraquadro e interquadros. No entanto, para uma visualização mais clara e impactante do efeito da integração dos modelos preditivos sobre o consumo energético da arquitetura, optou-se por representar graficamente os dados das reduções de potência e frequência.

Tabela 23 – Resultados de síntese para modo inter

Vídeo (resolução)	Fluxo Original		Fluxo Modificado	
	Power (mW)	Frequência (MHz)	Power (mW)	Frequência (MHz)
RaceHorses (416×240 @30fps)	18,35	0,50	17,75	0,26
BasketballPass (416×240 @50fps)	21,63	0,83	19,73	0,44
BlowingBubbles (416×240 @50fps)	21,14	0,83	19,16	0,44
BQSquare (416×240 @60fps)	25,10	1,00	22,16	0,60
RaceHorsesC (832×480 @30fps)	26,19	1,87	19,87	0,97
BasketballDrill (832×480 @50fps)	53,13	3,12	25,70	1,66
BasketballDrillText (832×480 @50fps)	53,40	3,12	26,44	1,70
PartyScene (832×480 @50fps)	55,73	3,12	25,66	1,66
BQMall (832×480 @60fps)	56,90	3,74	28,86	1,90
SlideShow (1280×720 @20fps)	33,97	2,94	23,91	1,61
SlideEditing (1280×720 @30fps)	54,05	4,42	32,00	3,12
KristenAndSara (1280×720 @60fps)	101,98	8,83	37,96	4,62
Johnny (1280×720 @60fps)	101,58	8,83	39,29	4,48
FourPeople (1280×720 @60fps)	102,65	8,83	38,21	4,28
Cactus (1920×1080 @50fps)	192,10	16,32	74,88	7,54
BasketballDrive (1920×1080 @50fps)	183,28	16,32	68,96	7,52
RitualDance (1920×1080 @60fps)	228,16	19,58	98,49	10,53
ArenaOfValor (1920×1080 @60fps)	218,59	19,58	95,67	10,41
MarketPlace (1920×1080 @60fps)	252,81	19,58	121,42	11,32
BQTerrace (1920×1080 @60fps)	236,23	19,58	101,41	9,74
Campfire (3840×2160 @30fps)	464,95	39,17	196,27	20,11
ParkRunning3 (3840×2160 @50fps)	738,50	65,28	331,77	36,44
CatRobot (3840×2160 @60fps)	862,23	78,34	362,49	39,02
DaylightRoad2 (3840×2160 @60fps)	905,87	78,34	332,16	33,55
FoodMarket4 (3840×2160 @60fps)	784,79	78,34	339,74	41,62
Tango2 (3840×2160 @60fps)	875,56	78,34	354,41	39,32

Tabela 24 – Resultados de síntese para modo intra

Vídeo (resolução)	Fluxo Original		Fluxo Modificado	
	Power (mW)	Frequência (MHz)	Power (mW)	Frequência (MHz)
RaceHorses (416×240 @30fps)	20,41	0,50	18,46	0,17
BasketballPass (416×240 @50fps)	21,10	0,83	17,97	0,13
BlowingBubbles (416×240 @50fps)	21,93	0,83	17,89	0,28
BQSquare (416×240 @60fps)	24,62	1,00	19,32	0,29
RaceHorsesC (832x480 @30fps)	26,36	1,87	19,37	0,71
BasketballDrill (832×480 @50fps)	52,48	3,12	21,88	1,04
BasketballDrillText (832×480 @50fps)	52,14	3,12	22,10	1,11
PartyScene (832×480 @50fps)	54,04	3,12	22,21	1,09
BQMall (832×480 @60fps)	55,98	3,74	22,40	1,06
SlideShow (1280×720 @20fps)	27,92	2,94	21,45	1,48
SlideEditing (1280×720 @30fps)	57,58	4,42	25,13	1,62
KristenAndSara (1280×720 @60fps)	90,50	8,83	26,79	3,08
Johnny (1280×720 @60fps)	84,75	8,83	28,57	3,46
FourPeople (1280×720 @60fps)	97,07	8,83	25,43	2,17
Cactus (1920×1080 @50fps)	176,21	16,32	44,50	7,23
BasketballDrive (1920×1080 @50fps)	161,11	16,32	59,53	7,89
RitualDance (1920×1080 @60fps)	200,96	19,58	46,94	9,03
ArenaOfValor (1920×1080 @60fps)	214,80	19,58	53,73	8,79
MarketPlace (1920×1080 @60fps)	232,42	19,58	50,03	7,37
BQTerrace (1920×1080 @60fps)	215,86	19,58	87,50	12,34
Campfire (3840×2160 @30fps)	335,96	39,17	81,31	19,15
ParkRunning3 (3840×2160 @50fps)	744,87	65,28	201,81	37,15
CatRobot (3840×2160 @60fps)	761,89	78,34	198,42	36,72
DaylightRoad2 (3840×2160 @60fps)	788,17	78,34	242,45	44,61
FoodMarket4 (3840×2160 @60fps)	640,12	78,34	228,16	47,75
Tango2 (3840×2160 @60fps)	754,54	78,34	217,48	41,13

As Figuras 35 e 36 ilustram, respectivamente, a redução percentual no dissipação de potência e nas frequências-alvo observadas após a incorporação do modelo preditivo à arquitetura original. Essa abordagem foi adotada para evidenciar de maneira mais direta a economia energética introduzida pela modificação proposta, facilitando a comparação entre diferentes resoluções de vídeo e modos de predição.

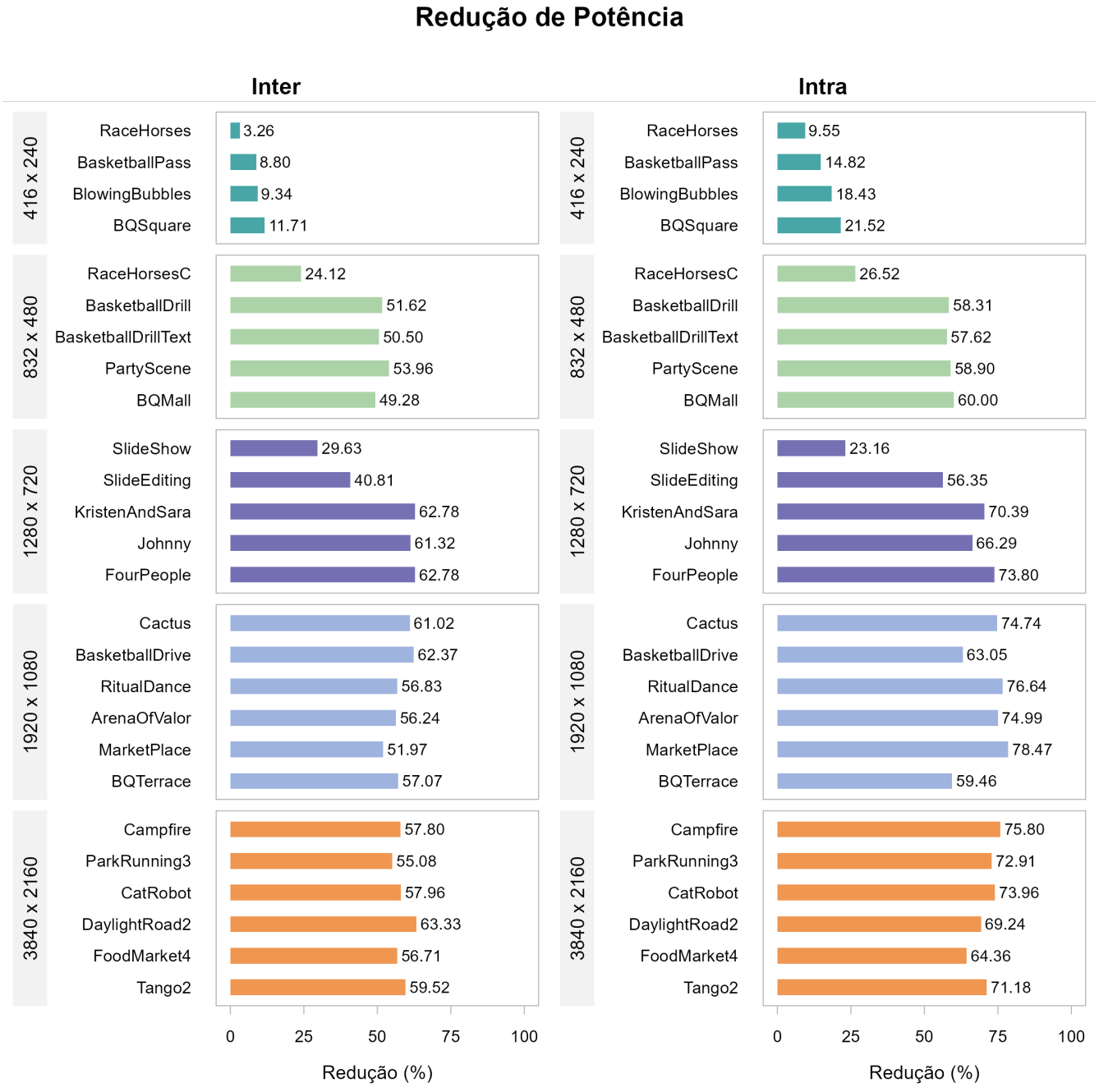

Figura 35 – Redução de potência com a integração do modelo.

Ao analisar os resultados apresentados nas Figuras 35 e 36, observa-se que a introdução dos modelos preditivos no processo de codificação gerou reduções significativas tanto na potência estimada quanto na frequência de operação das arquiteturas.

No que se refere à redução de potência, destacam-se as resoluções mais altas,

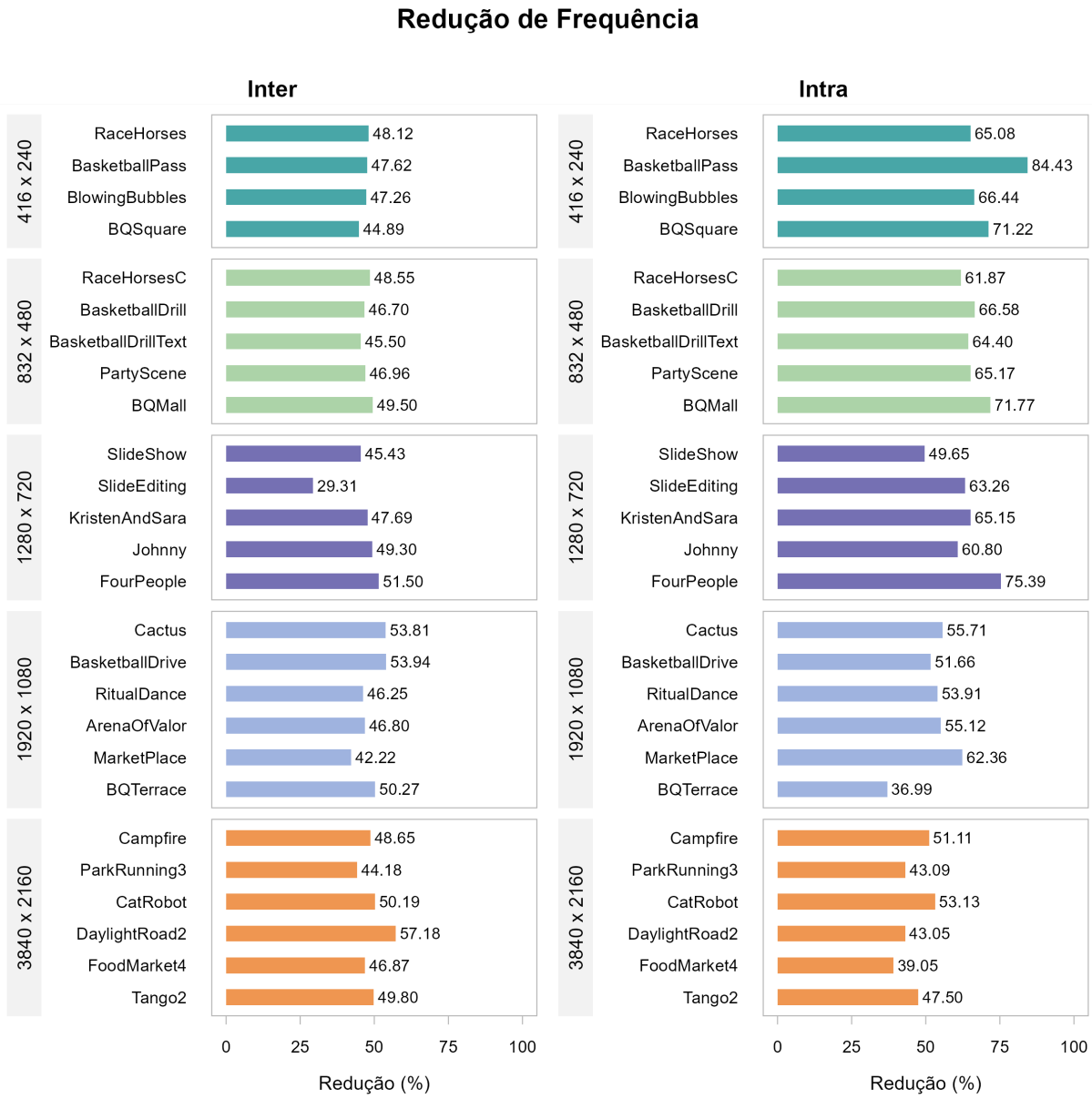

Figura 36 – Redução de frequência com a integração do modelo.

como 3840×2160 (4K) e 1920×1080 (Full HD), com valores superiores a 70% em várias sequências no modo Intra. Por exemplo, na resolução 4K, as sequências Campfire e ParkRunning3 apresentaram reduções superiores a 75%, evidenciando a eficácia do modelo.

Para a resolução 1920×1080, o desempenho do modelo também se mostrou expressivo, com destaque para as sequências Marketplace e RitualDance, cujas reduções ultrapassaram 75%. Já em resoluções menores, como 416×240, os ganhos, embora ainda presentes, foram mais modestos, com médias de redução de potência na faixa de 3% a 26%. Isso pode ser atribuído ao menor volume de dados e à menor complexidade desses vídeos, que naturalmente exigem menos recursos computacionais, reduzindo o potencial de otimização com o modelo.

Quanto à redução de frequência, um comportamento semelhante é observado. As

maiores reduções estão associadas às resoluções mais altas, o que é coerente com o fato de que o modelo preditivo atua reduzindo o número de operações computacionais intensivas, diminuindo a exigência de clock para manter o throughput necessário. Em particular, as sequências BasketballPass, Tango2 e Johnny demonstraram reduções superiores a 60% na frequência-alvo.

Outro ponto relevante é a consistência dos ganhos nas diferentes resoluções. A presença de reduções significativas em praticamente todas as resoluções evidencia que o modelo proposto é eficiente e generalizável, funcionando bem em conteúdos variados e em diferentes faixas de complexidade.

Essa análise comprova que a integração de um modelo preditivo para antecipação do índice de transformada não apenas reduz o tempo de codificação, como também diminui substancialmente o consumo de energia e a exigência de frequência de operação dos circuitos.

Para a avaliação do consumo energético médio da arquitetura, nesta etapa também foi adotada a técnica de normalização da potência por sequência de vídeo, a fim de permitir uma comparação mais justa entre as diferentes amostras. Entretanto, considerando que o objetivo principal é comparar, para cada resolução, a versão original da arquitetura com sua correspondente versão modificada, optou-se por utilizar como métrica a energia consumida por quadro. Para isso, os valores de potência estimada foram ajustados com base na resolução de cada vídeo e em uma taxa de processamento de 60 *fps*, valor amplamente adotado em aplicações de vídeo em tempo real. A escolha por normalizar os dados por resolução garante que a comparação entre as versões original e modificada da arquitetura seja realizada sob condições iguais de esforço computacional, uma vez que a resolução está diretamente relacionada ao volume de dados processados por quadro. A Tabela 25 apresenta esses resultados de energia para os modos inter e intra.

7.4.4 Arquiteturas para os Fluxos do VTM Original e Modificado

As arquiteturas implementadas com base nos fluxogramas de execução do VTM original e do VTM modificado (vide Figuras 24 e 25) também foram avaliadas quanto à frequência de operação, área ocupada e energia consumida isoladamente da arquitetura da MTS.

7.4.4.1 Resultados de Síntese

A Tabela 26 apresenta os resultados obtidos para ambas as versões, considerando diferentes resoluções e taxas de quadros. São listados os valores correspondentes de frequência, área e potência total dissipada, permitindo uma análise comparativa dos impactos da modificação no fluxo de codificação.

Observa-se que a arquitetura modificada, embora tenha apresentado um aumento

Tabela 25 – Resultados de energia para os modos inter e intra

Resolução de vídeo	Modo	Média de Energia normalizada (mJ)		Redução de Energia (%)
		Original	Modificada	
416×240	Inter	28,28	26,08	**7,77**
	Intra	29,26	24,82	**15,20**
832×480	Inter	60,80	32,39	**46,72**
	Intra	59,82	28,11	**53,01**
HD (1280×720)	Inter	103,25	50,23	**51,35**
	Intra	94,24	39,08	**58,53**
FHD (1920×1080)	Inter	231,04	98,27	**57,47**
	Intra	211,47	60,51	**71,39**
4K (3840×2160)	Inter	874,09	363,23	**58,45**
	Intra	751,75	215,22	**71,37**

significativo na área total (45 Kgates para 66 Kgates), apresentou redução na frequência de operação e potência dissipada em todas as resoluções avaliadas. Essa redução está diretamente associada à simplificação do processo de seleção de transformadas promovida pelo uso dos modelos preditivos. Ao evitar a testagem exaustiva de múltiplas combinações, o fluxo modificado permite que a arquitetura opere em frequências mais baixas para atender à mesma demanda de throughput, resultando, consequentemente, em menores níveis de consumo de potência. Por exemplo, para a resolução HD a 60 fps, a frequência de operação foi reduzida de 8,64 MHz no fluxo original para 3,61 MHz na versão modificada, com uma correspondente diminuição de potência de 0,005 mW para 0,002 mW. Esse comportamento se mantém nas resoluções mais altas, como 4K a 60 fps, em que a frequência caiu de 77,76 MHz para 40,16 MHz, e a potência de 0,042 mW para 0,019 mW.

Esses resultados evidenciam o potencial do uso de estratégias baseadas em aprendizado de máquina para otimizar a execução do módulo de transformadas, não apenas reduzindo o consumo energético, mas também diminuindo as exigências de frequência sem comprometer a funcionalidade. No entanto, o aumento da área é um fator a ser considerado, já que a lógica adicional associada ao novo fluxo de controle, ainda que composta majoritariamente por instruções condicionais simples, contribui para o crescimento do circuito. De forma geral, a arquitetura modificada demonstra um balanço positivo entre área, potência e frequência, revelando-se vantajosa para aplicações onde o consumo energético é um fator crítico, como em dispositivos em-

Tabela 26 – Resultados de frequência, área e potência para o fluxo de execução do VTM original e modificado

Resolução	Original			Modificada		
	Frequência (MHz)	Área (Kgates)	Power (mW)	Frequência (MHz)	Área (Kgates)	Power (mW)
HD 30fps	4,32	45	0,005	2,32	66	0,000
HD 60fps	8,64		0,005	3,61		0,002
FHD 30fps	9,72		0,005	9,72		0,002
FHD 60fps	19,44		0,008	9,87		0,002
4K 30fps	38,88		0,020	19,49		0,009
4K 60fps	77,76		0,042	40,16		0,019

barcados e sistemas com restrições térmicas ou de bateria.

7.4.5 Comparação com Trabalhos Relacionados

A Tabela 27 apresenta uma comparação qualitativa entre a arquitetura proposta e trabalhos relacionados encontrados na literatura.

Um fato importante que se deve ressaltar é que a maioria das arquiteturas apresentadas anteriormente apenas são sintetizadas para componentes FPGA e não para componentes *Application-Specific Integrated Circuit* (ASIC), além de não utilizarem dados reais extraídos de vídeos para avaliação de consumo energético. Isto deve ser levado em consideração, pois torna os resultados menos realistas para uma avaliação de eficiência energética das arquiteturas.

A maioria das arquiteturas de *hardware* encontradas na literatura suportam apenas tamanhos de blocos quadrados e são sintetizadas apenas para FPGA. Este é o caso dos trabalhos de Mert; Kalali; Hamzaoglu (2017), Kammoun; Hamidouche; Philipp; Belghith; Massmoudi; Nezan (2019), e Garrido; Pescador; Chavarrías; Lobo; Sanz (2019). O trabalho de Zeng; Sun; Katto; Fan (2021) suporta blocos retangulares, mas implementa apenas o DCT-VIII e DST-VII. Os dois trabalhos que apresentam resultados de síntese para ASIC são Fan; Zeng; Sun; Katto; Zeng (2019) e Farhat; Hamidouche; Grill; Menard; Déforges (2020). A solução de Fan; Zeng; Sun; Katto; Zeng (2019) inclui apenas as transformadas DCT-VIII e DST-VII, e apresenta resultados para a tecnologia de 65 nm. A arquitetura de *hardware* de Farhat; Hamidouche; Grill; Menard; Déforges (2020) suporta todos os tamanhos de bloco permitidos no módulo MTS, mas tem como alvo o módulo de transformadas inversas. Além disso, os resultados de área e frequência são apresentados apenas para as transformadas 1D. A arquitetura proposta neste trabalho tem como alvo o codificador, que suporta todas

as combinações de transformadas permitidas na ferramenta MTS e suporta todos os tamanhos de blocos de 4×4 a 32×32, incluindo formas retangulares.

Na arquitetura apresentada nesta seção, o módulo MTS do codificador VVC é totalmente contemplado. O projeto proposto é capaz de processar blocos residuais de todas as formas quadradas e retangulares iguais ou menores que 32×32 (ou seja, 4×4, 4×8, 4×16, 4×32, 8×4, 8×8, 8×16, 8×32, 16×4, 16×8, 16×16, 16×32, 32×4, 32×8, 32×16 e 32×32). Além disso, a arquitetura suporta todas as combinações de transformadas 2D permitidas na ferramenta MTS para blocos preditos com predição intraquadro e interquadros e é capaz de processar até vídeos 4K de 60 fps em tempo real.

Tabela 27 – Comparação de trabalhos relacionados

	Resolução	Freq. max (MHz)	Área (Kgates)	Potência (mW)	Síntese
(KAMMOUN et al., 2018)	2K@50fps	147	Não reportado	Não reportado	FPGA
(GARRIDO et al., 2019)	4K@23fps	576	Não reportado	Não reportado	FPGA 14nm
(YIBO et al., 2019)	Não reportado	384	228,6	61,81	ASIC 65nm
(FAN et al., 2019)	Não reportado	250	560	69,5	ASIC 65nm
(KAMMOUN et al., 2019)	4K@94fps	257	Não reportado	Não reportado	FPGA
(KAMMOUN et al., 2019)	4K@273fps	228	Não reportado	Não reportado	FPGA 20nm
(GARRIDO et al., 2020)	4K@64fps	200	Não reportado	Não reportado	FPGA
(FARHAT et al., 2020)	4K@48fps	600	96,85	Não reportado	ASIC 28nm
(IMEN et al., 2021)	Não reportado	165	Não reportado	Não reportado	FPGA
(FARHAT et al., 2021)	4K@30fps	600	163,67	Não reportado	ASIC 28nm
(BEN et al., 2022)	4K@25fps	78	Não reportado	Não reportado	FPGA
(GOEBEL et al., 2022)	4K@60fps	186,6	99,13	38,5	ASIC
(HAO et al., 2022)	Não reportado	384	Não reportado	Não reportado	ASIC 65nm
(GOEBEL et al., 2022)	4K@60fps	746,8 186,6	57,3	32,22	ASIC
(ZHANG et al., 2022)	8K@44fps	366	Não reportado	Não reportado	FPGA
Arquitetura Proposta	4K60fps	622,08	1544	362,49	ASIC 40nm

7.5 Resumo do capítulo

Este capítulo abordou o desenvolvimento e a análise de arquiteturas de *hardware* dedicadas à implementação das transformadas diretas do padrão *Versatile Video Coding*, com foco na ferramenta *Multiple Transform Selection*. Diferentemente das transformadas inversas, as transformadas diretas são aplicadas exclusivamente no codificador e exercem grande impacto sobre a eficiência da compressão, sendo uma das etapas mais exigentes do ponto de vista computacional. O capítulo apresentou soluções arquiteturais para os três tipos de transformadas unidimensionais suportadas pela MTS (DCT-II, DCT-VIII e DST-VII), incluindo a reutilização de operações em estruturas hierárquicas para a DCT-II e estratégias de compartilhamento de operadores entre DCT-VIII e DST-VII. A arquitetura proposta foi projetada para suportar blocos residuais de até 32×32, abrangendo todas as combinações de transformadas 2D permitidas pelo padrão. Três implementações distintas foram desenvolvidas: (i) uma baseada no pior caso da MTS, considerando todas as combinações de tamanhos de bloco e transformadas; (ii) uma versão com restrição ao tamanho de bloco 32×32, alinhada com o processo de treinamento dos modelos preditivos; e (iii) uma implementação que integra dados reais extraídos do codificador VTM modificado com aprendizado de máquina e também restrita ao tamanho de bloco 32×32. Foram realizados testes de síntese lógica para cada uma dessas implementações, utilizando tecnologia ASIC de 40 nm. Os resultados mostraram que a energia por *pixel* é mais eficiente em resoluções mais altas, como 4K, enquanto resoluções mais baixas sofrem maior impacto do *overhead* fixo da arquitetura. A terceira implementação evidenciou os benefícios da integração dos modelos preditivos: houve significativa redução na frequência de operação e no consumo de potência, com destaque para ganhos superiores a 70% em resoluções como FHD e 4K. Por fim, os resultados foram comparados com os principais trabalhos da literatura, demonstrando que a arquitetura proposta se destaca por oferecer suporte completo à MTS, por ser sintetizada para ASIC e por utilizar dados reais extraídos de vídeos, o que a torna mais realista e aplicável a cenários de compressão em tempo real e sistemas embarcados com restrições energéticas.

8 CONCLUSÃO

Esta tese propôs e investigou uma abordagem híbrida para otimização do módulo de transformadas do codificador *Versatile Video Coding* (VVC), integrando soluções de *hardware* de baixo consumo com técnicas de aprendizado de máquina supervisionado. A motivação central da pesquisa surgiu do reconhecimento de que a ferramenta *Multiple Transform Selection*, embora fundamental para a eficiência de compressão do padrão VVC, impõe uma carga computacional elevada devido à necessidade de testar todas as combinações possíveis de transformadas e tamanhos de blocos. Esse processo, intensivo em recursos, compromete a viabilidade do uso do codificador em contextos de tempo real e em dispositivos com restrições energéticas.

A hipótese que guiou o desenvolvimento desta tese foi que a integração de arquiteturas eficientes de *hardware* com modelos preditivos gerados por algoritmos de aprendizado de máquina poderia reduzir significativamente o custo computacional do módulo de transformadas do codificador VVC. A proposta partiu do princípio de que decisões condicionais inteligentes, aplicadas na seleção de transformadas, seriam capazes de eliminar combinações desnecessárias, reduzindo assim o número de operações e o consumo energético, ainda que com uma leve perda de eficiência de compressão.

O trabalho foi desenvolvido em três grandes etapas. Na primeira, foi conduzida uma análise aprofundada da MTS no *software* de referência VTM, quantificando seu impacto na complexidade do processo de codificação. Essa etapa revelou que a MTS pode representar até 35% do tempo total de codificação em cenários de alta qualidade. Além disso, observou-se que, embora existam diversas combinações de transformadas previstas pelo padrão, um subconjunto específico é dominante na maioria dos casos. Esses dados fundamentaram a proposta de que uma seleção preditiva poderia eliminar redundâncias.

A segunda etapa da pesquisa envolveu o desenvolvimento de modelos preditivos utilizando algoritmos de árvores de decisão. Quatro modelos foram treinados com dados extraídos diretamente do VTM, diferenciando blocos intra e inter. Os modelos foram integrados ao VTM e, após testes, demonstraram uma redução significativa

na quantidade de transformadas testadas. Os resultados mostraram uma diminuição média de 7,98% no tempo total de codificação e reduções de até 45,33% no tempo de codificação de blocos intra, com aumento médio de apenas 0,89% no BD-BR. Esses números confirmam que a seleção antecipada de transformadas é eficaz para reduzir o custo computacional com impacto mínimo na eficiência de compressão.

A terceira etapa concentrou-se na implementação das arquiteturas de *hardware*. Três versões distintas foram desenvolvidas: (i) uma baseada no pior caso de operação, incluindo todas as combinações possíveis de transformadas e tamanhos de blocos; (ii) uma versão restrita a blocos de tamanho 32×32, com busca entre todas as combinações de transformadas; e (iii) uma arquitetura baseada nos dados extraídos da execução do VTM com os modelos preditivos integrados. As arquiteturas foram descritas em VHDL e sintetizadas em tecnologia ASIC de 40 nm, utilizando a ferramenta *Cadence RTL Compiler*. A avaliação dos resultados mostrou que as versões ajustadas aos modelos preditivos apresentaram redução de até 50% na potência dissipada e área inferior, quando comparadas à versão do pior caso.

Além disso, foi discutida a viabilidade de implementação direta dos modelos preditivos em *hardware*. Como os modelos são compostos por estruturas simples de decisão (baseadas em *if-else*), sua implementação em circuitos lógicos é trivial do ponto de vista estrutural e pode ser integrada de forma modular ao fluxo do codificador. Isso torna o sistema mais eficiente e com potencial de adaptação para aplicações reais em sistemas embarcados e dispositivos com restrições de energia.

Com os resultados alcançados, é possível afirmar que a hipótese proposta foi confirmada. A combinação de estratégias inteligentes de aprendizado de máquina com arquiteturas de *hardware* otimizadas demonstrou ser eficaz para mitigar a complexidade computacional da MTS no codificador VVC. A abordagem resultou em economia energética significativa e redução de área de *hardware*, mantendo níveis aceitáveis de eficiência de compressão.

Essa solução híbrida representa uma contribuição relevante para o estado da arte em compressão de vídeo, especialmente para aplicações que exigem codificação em tempo real, como transmissões ao vivo, videoconferência e dispositivos móveis. Além disso, a metodologia adotada pode ser estendida para outros módulos do codificador ou para diferentes padrões de compressão, consolidando o uso de aprendizado de máquina como ferramenta estratégica no projeto de sistemas digitais eficientes.

Assim, conclui-se que a tese atingiu seu objetivo central de propor e validar soluções de baixo consumo energético para o módulo de transformadas do VVC, fundamentadas na combinação entre *hardware* dedicado e aprendizado de máquina supervisionado. As contribuições deste trabalho reforçam o papel das abordagens híbridas como caminho promissor na evolução dos sistemas de codificação de vídeo do futuro.

Como trabalhos futuros, destaca-se a possibilidade de estender as técnicas pro-

postas para blocos de diferentes tamanhos, ampliando a generalização dos resultados obtidos. No âmbito do hardware, a adoção de técnicas de clock gating poderá contribuir para uma maior eficiência energética das arquiteturas desenvolvidas, especialmente em cenários de variação de carga e de resolução. No software, a aplicação dos modelos preditivos desenvolvidos a outros modos de MTS (tanto explícito quanto implícito) representa uma oportunidade relevante para a redução da complexidade computacional em etapas ainda não exploradas nesta tese. Essas direções reforçam o potencial das soluções aqui apresentadas para aplicações futuras em sistemas de codificação de vídeo de alto desempenho.

REFERÊNCIAS

ABDALLAH, B.; BELGHITH, F.; MASMOUD, N. Low-complexity transform algorithm for versatile video coding. In: IEEE INTERNATIONAL CONFERENCE ON DESIGN & TEST OF INTEGRATED MICRO & NANO-SYSTEMS (DTS), 2019., 2019. **Anais...** [S.l.: s.n.], 2019. p.1–3.

ABDOLI, M.; HENRY, F.; BRAULT, P.; DUFAUX, F.; DUHAMEL, P. Transform coefficient coding for screen content in Versatile Video Coding (VVC). In: ICASSP 2019-2019 IEEE INTERNATIONAL CONFERENCE ON ACOUSTICS, SPEECH AND SIGNAL PROCESSING (ICASSP), 2019. **Anais...** [S.l.: s.n.], 2019. p.1792–1796.

BEN JDIDIA, S.; BELGHITH, F.; MASMOUDI, N. A high-performance two-dimensional transform architecture of variable block sizes for the VVC standard. **Journal of Real-Time Image Processing**, [S.l.], v.19, n.6, p.1081–1090, 2022.

BJONTEGAARD, G. Calculation of average PSNR differences between RD-curves. **VCEG-M33**, [S.l.], 2001.

BJØNTEGAARD, G. **Calculation of average PSNR differences between RD-curves**. 2001. Disponível em: <http://wftp3.itu.int/av-arch/video-site/0104_Aus/VCEG-M33.doc>. Acesso em: 2020-09-13.

BONACCORSO, G. **Machine learning algorithms**. [S.l.]: Packt Publishing Ltd, 2017.

BONNINEAU, C.; PURI, S.; NASER, K.; POIRIER, T.; LéANNEC, F. L. Low-Complexity Transform Design Using Hybrid Intra MTS. In: PICTURE CODING SYMPOSIUM (PCS), 2024. **Proceedings...** [S.l.: s.n.], 2024. p.1–5.

BOSSEN, F. **Common test conditions and software reference configurations**. 2013. `http://phenix.it-sudparis.eu/jct/doc/end/user/current/document.php?id=7281`.

BOSSEN, F. **VTM common test conditions and software reference configurations for SDR video**. 2021. `https://jvetexperts.org/doc_end_user/current_document.php?id=10545`.

BOYCE, J.; SUEHRING, K.; LI, X. JVET-J1010: JVET common test conditions and software reference configurations. **JVET, San Diego, CA, USA, Tech. Rep. JVET-J1010**, [S.l.], 2018.

BROSS, B.; CHEN, J.; OHM, J.-R.; SULLIVAN, G. J.; WANG, Y.-K. Developments in international video coding standardization after avc, with an overview of versatile video coding (vvc). **Proceedings of the IEEE**, [S.l.], v.109, n.9, p.1463–1493, 2021.

CADENCE ENCOUNTER, R. **Cadence Encounter, RTL**. 2022. Disponível em: www.cadence.com.

CHAN, K.-H.; IM, S.-K. Discrete Tchebichef Transform for Versatile Video Coding. In: INTERNATIONAL CONFERENCE ON MULTIMEDIA RETRIEVAL, 2021., 2021. **Proceedings...** [S.l.: s.n.], 2021. p.623–626.

CISCO. **Cisco Annual Internet Report (2018–2023) White Paper**. 2020. Disponível em: <www.cisco.com>. Acesso em: 2023-09-05.

CORRÊA, G. R. **Computational Complexity Reduction and Scaling for High Efficiency Video Encoders**. 2014. Tese (Doutorado em Ciência da Computação) — Universidade de Coimbra.

DAEDE, T.; NORKIN, A.; BRAILOVSKIY, I. **Video Codec Testing and Quality Measurement**. 2020. `https://tools.ietf.org/html/draft-ietf-netvc-testing-09`.

DAMAK, T.; HOUIDI, S.; AYED, M. B.; MASMOUDI, N. Adaptive Multiple Transforms Hardware Architecture for Versatile Video Coding. **International Journal of Computer and Information Engineering**, [S.l.], v.14, n.3, p.72–77, 2020.

DINIZ, C. M. **Dedicated and reconfigurable hardware accelerators for high efficiency video coding standard**. 2015. Tese (Doutorado em Ciência da Computação) — Universidade Federal do Rio Grande do Sul.

EGILMEZ, H. E.; SINGH, A. K.; COBAN, M.; KARCZEWICZ, M.; ZHU, Y.; YANG, Y.; SAID, A.; COHEN, T. S. Transform network architectures for deep learning based end-to-end image/video coding in subsampled color spaces. **IEEE Open Journal of Signal Processing**, [S.l.], v.2, p.441–452, 2021.

ERICSSON. **Ericsson Mobility Report - November 2024**. 2024. Acesso em: [data de acesso], `https://www.ericsson.com/en/reports-and-papers/mobility-report/reports/november-2024`.

FAN, Y.; ZENG, Y.; SUN, H.; KATTO, J.; ZENG, X. A pipelined 2d transform architecture supporting mixed block sizes for the vvc standard. **IEEE Transactions on Circuits and Systems for Video Technology**, [S.l.], v.30, n.9, p.3289–3295, 2019.

FARHAT, I.; HAMIDOUCHE, W.; GRILL, A.; MENARD, D.; DÉFORGES, O. Lightweight hardware implementation of VVC transform block for ASIC decoder. In: IEEE INTERNATIONAL CONFERENCE ON ACOUSTICS, SPEECH AND SIGNAL PROCESSING (ICASSP), 2020. **Anais...** [S.l.: s.n.], 2020. p.1663–1667.

FARHAT, I.; HAMIDOUCHE, W.; GRILL, A.; MÉNARD, D.; DÉFORGES, O. Lightweight Hardware Transform Design for the Versatile Video Coding 4K ASIC Decoders. **IEEE Transactions on Consumer Electronics**, [S.l.], v.67, n.4, p.329–340, 2021.

FRAUNHOFER. **VTM reference software, VVC test model (VTM)**. 2020. Disponível em: <https://vcgit.hhi.fraunhofer.de/jvet/VVCSoftware_VTM.>.

FU, T.; ZHANG, H.; MU, F.; CHEN, H. Two-stage fast multiple transform selection algorithm for VVC intra coding. In: IEEE INTERNATIONAL CONFERENCE ON MULTIMEDIA AND EXPO (ICME), 2019., 2019. **Anais...** [S.l.: s.n.], 2019. p.61–66.

FU, T.; ZHANG, H.; MU, F.; CHEN, H. Two-Stage Fast Multiple Transform Selection Algorithm for VVC Intra Coding. In: IEEE INTERNATIONAL CONFERENCE ON MULTIMEDIA AND EXPO (ICME), 2019., 2019. **Anais...** [S.l.: s.n.], 2019. p.61–66.

GARRIDO, M. J.; PESCADOR, F.; CHAVARRÍAS, M.; LOBO, P. J.; SANZ, C. A 2-D multiple transform processor for the versatile video coding standard. **IEEE Transactions on Consumer Electronics**, [S.l.], v.65, n.3, p.274–283, 2019.

GARRIDO, M. J.; PESCADOR, F.; CHAVARRÍAS, M.; LOBO, P. J.; SANZ, C.; PAZ, P. An FPGA-Based Architecture for the Versatile Video Coding Multiple Transform Selection Core. **IEEE Access**, [S.l.], v.8, p.81887–81903, 2020.

GOEBEL, J.; AGOSTINI, L.; ZATT, B.; PORTO, M. Low-Frequency Non-Separable Transform Hardware System Design for the VVC Encoder. In: SBC/SBMICRO/IEEE/ACM SYMPOSIUM ON INTEGRATED CIRCUITS AND SYSTEMS DESIGN (SBCCI), 2022., 2022. **Anais...** [S.l.: s.n.], 2022. p.1–6.

GOEBEL, J.; COSTA, V.; AGOSTINI, L.; ZATT, B.; PORTO, M. A High-Throughput Design for the H. 266/VVC Low-Frequency Non-Separable Transform. In: IEEE INTERNATIONAL SYMPOSIUM ON CIRCUITS AND SYSTEMS (ISCAS), 2022., 2022. **Anais...** [S.l.: s.n.], 2022. p.1798–1802.

GONZALEZ, R. C. **Digital image processing**. [S.l.]: Pearson education india, 2009.

HAMIDOUCHE, W.; BIATEK, T.; ABDOLI, M.; FRANçOIS, E.; PESCADOR, F.; RADOSAVLJEVIć, M.; MENARD, D.; RAULET, M. Versatile Video Coding Standard: A Review From Coding Tools to Consumers Deployment. **IEEE Consumer Electronics Magazine**, [S.l.], v.11, n.5, p.10–24, 2022.

HAMIDOUCHE, W.; PHILIPPE, P.; FEZZA, S. A.; HADDOU, M.; PESCADOR, F.; MENARD, D. Hardware-friendly multiple transform selection module for the VVC standard. **IEEE Transactions on Consumer Electronics**, [S.l.], 2022.

HAMIDOUCHE, W.; PHILIPPE, P.; MOHAMED, C.-E.; KAMMOUN, A.; MENARD, D.; DÉFORGES, O. Hardware-friendly dst-vii/dct-viii approximations for the versatile video coding standard. In: PICTURE CODING SYMPOSIUM (PCS), 2019., 2019. **Anais...** [S.l.: s.n.], 2019. p.1–5.

HAO, Z.; ZHENG, Q.; FAN, Y.; XIANG, G.; ZHANG, P.; SUN, H. An Area-efficient Unified Transform Architecture for VVC. In: IEEE INTERNATIONAL SYMPOSIUM ON CIRCUITS AND SYSTEMS (ISCAS), 2022., 2022. **Anais...** [S.l.: s.n.], 2022. p.2012–2016.

HE, L.; XIONG, S.; YANG, R.; HE, X.; CHEN, H. Low-Complexity Multiple Transform Selection Combining Multi-Type Tree Partition Algorithm for Versatile Video Coding. **Sensors**, [S.l.], v.22, n.15, p.5523, 2022.

HE, L.; XIONG, S.; YANG, R.; HE, X.; CHEN, H. Low-Complexity Multiple Transform Selection Combining Multi-Type Tree Partition Algorithm for Versatile Video Coding. **Sensors**, [S.l.], v.22, n.15, 2022.

HE, S.; JIN, B.; TIAN, S.; LIU, J.; DENG, Z.; SHI, C. Hierarchical Reinforcement Learning-Based Adaptive Initial QP Selection and Rate Control for H. 266/VVC. **Electronics (2079-9292)**, [S.l.], v.13, n.24, 2024.

IMEN, W.; FATMA, B.; AMNA, M.; MASMOUDI, N. DCT-II transform hardware-based acceleration for VVC standard. In: IEEE INTERNATIONAL CONFERENCE ON DESIGN & TEST OF INTEGRATED MICRO & NANO-SYSTEMS (DTS), 2021., 2021. **Anais...** [S.l.: s.n.], 2021. p.1–5.

INTEL. **Intel® Quartus® Prime Software**. 2022. Disponível em:https://www.intel.com/content/www/us/en/products/details/fpga/development-tools/quartus-prime.html.

ITU. **Methodology for the Subjective Assessment of the Quality of Television Pictures**. Geneva, Switzerland: International Telecommunication Union, Radiocommunication Sector (ITU-R), 2012. Recommendation BT.500-13, Accessed 2025.

ITU-T. **Subjective Video Quality Assessment Methods for Multimedia Applications**. 2021. `https://www.itu.int/rec/T-REC-P.910-200804-I`.

JDIDIA, S. B.; AMOR, M. B.; BELGHITH, F.; MASMOUDI, N. Subjective evaluation of approximate discrete Sine transform for the versatile video coding standard. In:

INTERNATIONAL CONFERENCE ON ADVANCED TECHNOLOGIES FOR SIGNAL AND IMAGE PROCESSING (ATSIP), 2020., 2020. **Anais...** [S.l.: s.n.], 2020. p.1–5.

JVET. **VVC Software VTM, VTM-23.0**. 2024. Disponível em: <https://vcgit.hhi.fraunhofer.de/jvet/VVCSoftware$_VTM/-/tree/VTM-23.0?ref_type=tags$>.

KAMMOUN, A.; HAMIDOUCHE, W.; BELGHITH, F.; NEZAN, J.-F.; MASMOUDI, N. Hardware design and implementation of adaptive multiple transforms for the versatile video coding standard. **IEEE Transactions on Consumer Electronics**, [S.l.], v.64, n.4, p.424–432, 2018.

KAMMOUN, A.; HAMIDOUCHE, W.; PHILIPP, P.; BELGHITH, F.; MASSMOUDI, N.; NEZAN, J.-F. Hardware Acceleration of Approximate Transform Module for the Versatile Video Coding Standard. In: EUROPEAN SIGNAL PROCESSING CONFERENCE (EUSIPCO), 2019., 2019. **Anais...** [S.l.: s.n.], 2019. p.1–5.

LI, N.; ZHANG, Y.; ZHU, L.; LUO, W.; KWONG, S. Reinforcement learning based coding unit early termination algorithm for high efficiency video coding. **Journal of Visual Communication and Image Representation**, [S.l.], v.60, p.276–286, 2019.

LIU, X.; SHI, K.; LI, A.; ZHANG, H.; MING, J.; FANG, H. A Novel Information Hiding Method for H. 266/VVC Based on Selections of Luminance Transform and Chrominance Prediction Modes. In: IEEE INTERNATIONAL CONFERENCE ON SYSTEMS, MAN, AND CYBERNETICS (SMC), 2021., 2021. **Anais...** [S.l.: s.n.], 2021. p.3158–3163.

MAHADEVKAR, S. V.; KHEMANI, B.; PATIL, S.; KOTECHA, K.; VORA, D.; ABRAHAM, A.; GABRALLA, L. A. A Review on Machine Learning Styles in Computer Vision-Techniques and Future Directions. **IEEE Access**, [S.l.], 2022.

MAHESH, B. Machine learning algorithms-a review. **International Journal of Science and Research (IJSR).[Internet]**, [S.l.], v.9, p.381–386, 2020.

MEDINA, R.; SAHA, A.; FLORIANO, M.; CHAVARRÍAS, M.; PESCADOR, F. Porting Adaptive Multiple Transforms of a Versatile Video Coding decoder using OpenMP. In: IEEE 9TH INTERNATIONAL CONFERENCE ON CONSUMER ELECTRONICS (ICCE-BERLIN), 2019., 2019. **Anais...** [S.l.: s.n.], 2019. p.138–139.

MERT, A. C.; KALALI, E.; HAMZAOGLU, I. High performance 2D transform hardware for future video coding. **IEEE Transactions on Consumer Electronics**, [S.l.], v.63, n.2, p.117–125, 2017.

NGUYEN, T.; BROSS, B.; KEYDEL, P.; SCHWARZ, H.; MARPE, D.; WIEGAND, T. Extended transform skip mode and fast multiple transform set selection in VVC. In: PICTURE CODING SYMPOSIUM (PCS), 2019., 2019. **Anais...** [S.l.: s.n.], 2019. p.1–5.

NGUYEN, T.; BROSS, B.; SCHWARZ, H.; MARPE, D.; WIEGAND, T. Residual coding for transform skip mode in versatile video coding. In: DATA COMPRESSION CONFERENCE (DCC), 2020., 2020. **Anais...** [S.l.: s.n.], 2020. p.83–92.

PALAU, R. d. C. N.; CUNHA SILVEIRA, B. S. da; DOMANSKI, R. A.; LOOSE, M. B.; CERVEIRA, A. A.; SAMPAIO, F. M.; PALOMINO, D.; PORTO, M. S.; CORRÊA, G. R.; AGOSTINI, L. V. Modern Video Coding: Methods, Challenges and Systems. **Journal of Integrated Circuits and Systems**, [S.l.], v.16, n.2, p.1–12, 2021.

PEDREGOSA, F.; VAROQUAUX, G.; GRAMFORT, A.; MICHEL, V.; THIRION, B.; GRISEL, O.; BLONDEL, M.; PRETTENHOFER, P.; WEISS, R.; DUBOURG, V.; VANDERPLAS, J.; PASSOS, A.; COURNAPEAU, D.; BRUCHER, M.; PERROT, M.; DUCHESNAY, E. Scikit-learn: Machine Learning in Python. **Journal of Machine Learning Research**, [S.l.], v.12, p.2825–2830, 2011.

POWERS, D. M. W. Evaluation: From Precision, Recall and F-Measure to ROC, Informedness, Markedness and Correlation. **Journal of Machine Learning Technologies**, [S.l.], v.2, n.1, p.37–63, 2011.

POYNTON, C. **Digital video and HD**: Algorithms and Interfaces. [S.l.]: Elsevier, 2012.

RAY, S. A quick review of machine learning algorithms. In: INTERNATIONAL CONFERENCE ON MACHINE LEARNING, BIG DATA, CLOUD AND PARALLEL COMPUTING (COMITCON), 2019., 2019. **Anais...** [S.l.: s.n.], 2019. p.35–39.

SAID, A.; EGILMEZ, H. E.; CHAO, Y.-H. Low-complexity transform adjustments for video coding. In: IEEE INTERNATIONAL CONFERENCE ON IMAGE PROCESSING (ICIP), 2019., 2019. **Anais...** [S.l.: s.n.], 2019. p.1188–1192.

SALDANHA, M.; SANCHEZ, G.; MARCON, C.; AGOSTINI, L. Fast transform decision scheme for VVC intra-frame prediction using decision trees. In: IEEE INTERNATIONAL SYMPOSIUM ON CIRCUITS AND SYSTEMS (ISCAS), 2022., 2022. **Anais...** [S.l.: s.n.], 2022. p.1948–1952.

SALDANHA, M.; SANCHEZ, G.; MARCON, C.; AGOSTINI, L. Fast Transform Decision Scheme for VVC Intra-Frame Prediction Using Decision Trees. In: IEEE INTERNATIONAL SYMPOSIUM ON CIRCUITS AND SYSTEMS (ISCAS), 2022., 2022. **Anais...** [S.l.: s.n.], 2022. p.1948–1952.

SCHWARZ, H.; NGUYEN, T.; MARPE, D.; WIEGAND, T.; KARCZEWICZ, M.; COBAN, M.; DONG, J. Improved quantization and transform coefficient coding for the emerging versatile video coding (VVC) standard. In: IEEE INTERNATIONAL CONFERENCE ON IMAGE PROCESSING (ICIP), 2019., 2019. **Anais...** [S.l.: s.n.], 2019. p.1183–1187.

SEIDEL, I. et al. Análise do impacto de pel decimation na codificação de vídeos de alta resolução. , [S.l.], 2014.

SHI, Y. Q.; SUN, H. **Image and video compression for multimedia engineering**: Fundamentals, algorithms, and standards. [S.l.]: CRC press, 1999.

SIDATY, N.; HAMIDOUCHE, W.; DÉFORGES, O.; PHILIPPE, P.; FOURNIER, J. Compression Performance of the Versatile Video Coding: HD and UHD Visual Quality Monitoring. In: PICTURE CODING SYMPOSIUM (PCS), 2019., 2019. **Anais...** [S.l.: s.n.], 2019. p.1–5.

SILVER, D.; HUANG, A.; MADDISON, C. J.; GUEZ, A.; SIFRE, L.; VAN DEN DRIESSCHE, G.; SCHRITTWIESER, J.; ANTONOGLOU, I.; PANNEERSHELVAM, V.; LANCTOT, M. et al. Mastering the game of Go with deep neural networks and tree search. **nature**, [S.l.], v.529, n.7587, p.484–489, 2016.

SOCIETY, I. C. **IEEE Standard VHDL Language Reference Manual**. 2009. n.IEEE Std 1076-2008. Originally developed by Intermetrics, IBM, and Texas Instruments for the U.S. Department of Defense.

STONE, M. **A field guide to digital color**. [S.l.]: CRC press, 2016.

VÁZQUEZ, M. F.; SAHA, A.; MORILLAS, R. M.; LAPASTORA, M. C.; OSO, F. P. del. Work-in-Progress: Porting new Versatile Video Coding transforms to a heterogeneous GPU-based technology. In: INTERNATIONAL CONFERENCE ON COMPLIERS, ARCHITECTURES AND SYNTHESIS FOR EMBEDDED SYSTEMS (CASES), 2019., 2019. **Anais...** [S.l.: s.n.], 2019. p.1–2.

WANG, Y.; FENG, S.; ZHANG, W.; YANG, F. VVC Intra Coding Complexity Optimization Based on Early Skipping of the Secondary Transform. **IEEE Signal Processing Letters**, [S.l.], v.31, p.366–370, 2024.

WANG, Z.; WANG, J.; YANG, J.; LUO, C.; LIANG, F.; HUANG, K. A Fast Transform Algorithm for VVC Intra Coding. In: INTERNATIONAL CONFERENCE ON COMMUNICATIONS, CIRCUITS AND SYSTEMS (ICCCAS), 2022., 2022. **Anais...** [S.l.: s.n.], 2022. p.237–240.

WANG, Z.; WANG, J.; YANG, J.; LUO, C.; LIANG, F.; HUANG, K. A Fast Transform Algorithm for VVC Intra Coding. In: INTERNATIONAL CONFERENCE ON COMMUNICATIONS, CIRCUITS AND SYSTEMS (ICCCAS), 2022., 2022. **Anais...** [S.l.: s.n.], 2022. p.237–240.

WESTE, N. H. E.; HARRIS, D. **CMOS VLSI Design**: A Circuits and Systems Perspective. 4th.ed. [S.l.]: Addison-Wesley, 2011.

WIEN, M.; BARONCINI, V.; BOYCE, J.; SEGALL, A.; SUZUKI, T. Preliminary joint call for evidence on video compression with capability beyond HEVC. In: DOCUMENT JVET-E1002 5TH JVET MEETING, 2017. **Anais...** [S.l.: s.n.], 2017.

WINKLER, S. **Digital Video Quality**: Vision Models and Metrics. [S.l.]: John Wiley & Sons, 2005.

YIBO, F.; JIRO, K.; HEMING, S.; XIAOYANG, Z.; YIXUAN, Z. A minimal adder-oriented 1D DST-VII/DCT-VIII hardware implementation for VVC standard. In: IEEE INTERNATIONAL SYSTEM-ON-CHIP CONFERENCE (SOCC), 2019., 2019. **Anais...** [S.l.: s.n.], 2019. p.176–180.

ZENG, Y.; SUN, H.; KATTO, J.; FAN, Y. Approximated Reconfigurable Transform Architecture for VVC. In: IEEE INTERNATIONAL SYMPOSIUM ON CIRCUITS AND SYSTEMS (ISCAS), 2021. **Anais...** [S.l.: s.n.], 2021. p.1–5.

ZHANG, J.; SHI, W.; ZHANG, H. Study on versatile video coding multiple transform selection of hardware architecture based on FPGA. **Multimedia Tools and Applications**, [S.l.], p.1–16, 2022.

ZHANG, Y.; KWONG, S.; WANG, S. Machine learning based video coding optimizations: A survey. **Information Sciences**, [S.l.], v.506, p.395–423, 2020.

ZHANG, Z.; ZHAO, X.; LI, X.; LI, L.; LUO, Y.; LIU, S.; LI, Z. Fast DST-VII/DCT-VIII with dual implementation support for versatile video coding. **IEEE Transactions on Circuits and Systems for Video Technology**, [S.l.], v.31, n.1, p.355–371, 2020.

ZHANG, Z.; ZHAO, X.; LI, X.; LI, Z.; LIU, S. Fast adaptive multiple transform for versatile video coding. In: DATA COMPRESSION CONFERENCE (DCC), 2019., 2019. **Anais...** [S.l.: s.n.], 2019. p.63–72.

ZHAO, X.; CHEN, J.; CHOI, K.; COCK, J. D.; GRANGE, A.; HE, Y.; HELLE, P.; YE, Y.; ZHANG, L. **AOM Common Test Conditions v2.0**. 2022. `https://aomedia.org/docs/CWG-B075o_AV2_CTC_v2.pdf`.

Apêndices

APÊNDICE A – Principais Contribuições

Este apêndice lista as produções resultantes deste trabalho que foram publicadas e divulgadas em eventos científicos e periódicos ao longo do doutorado.

A.1 Artigos em Periódicos

- Roberta Palau, **Bianca Silveira**, Robson Domanski, Marta Loose, Arthur Cerveira, Felipe Sampaio, Daniel Palomino, Marcelo Porto, Guilherme Corrêa, Luciano Agostini. *MODERN VIDEO CODING: METHODS, CHALLENGES AND SYSTEMS*. **In: Journal of Integrated Circuits and Systems**, vol. 16, n.2, 2021.
- Caroline Camargo, **Bianca Silveira**, Guilherme Corrêa. *A COMPREHESIVE ANALYSIS OF THE MULTIPLE TRANSFORM SELECTION TOOL IN VERSATILE VIDEO CODING*. **In: Journal of Integrated Circuits and Systems**, vol 20, n.2, 2025 (aceito para publicação).
- **Bianca Silveira**, Caroline Camargo, Murilo Perleberg, Daniel Palomino, Cláudio Diniz, Guilherme Corrêa. *MACHINE LEARNING-ASSISTED ENERGY-EFFICIENT HARDWARE ACCELERATORS FOR TRANSFORM MODULE IN THE VERSATILE VIDEO CODING STANDARD*. **In: IEEE Transactions on Circuits and Systems I**, 2025 (submetido).

A.2 Artigos Publicados em Eventos Qualificados

- **Bianca Silveira**, Luis Neto, Daniel Palomino, Cláudio Diniz, Guilherme Correa. *MULTIPLE TRANSFORM SELECTION HARDWARE DESIGN FOR 4K@60FPS REAL-TIME VERSATILE VIDEO CODING*. **In: ISCAS - IEEE International Symposium on Circuits and Systems**, 2022.
- Bruna Garcia, **Bianca Silveira**, Luis Neto, Daniel Palomino, Cláudio Diniz, Guilherme Correa. *MULTI-SIZE INVERSE DCT-II HARDWARE DESIGN FOR THE VVC DECODER*. **In: LASCAS - Latin American Symposium on Circuits and Systems**, 2023.
- **Bianca Silveira**, Daniel Palomino, Cláudio Diniz, Guilherme Correa. *A HARDWARE DESIGN FOR MULTI-TRANSFORM MODULE OF THE VERSATILE VI-*

DEO CODING STANDARD. **In: SBCCI - Symposium on Integrated Circuits and Systems Design**, 2023.

- Caroline Camargo, **Bianca Silveira**, Guilherme Correa. *COMPUTIONAL COST ANALYSIS AND REDUCTION OF VVC MULTIPLE TRANSFORM SELECTION*. **In: LASCAS - Latin American Symposium on Circuits and Systems**, 2025.
- Caroline Camargo, **Bianca Silveira**, Guilherme Corrêa. *MACHINE LEARNING-DRIVEN MULTIPLE TRANSFORM SELECTION FOR LOW-COMPLEXITY VVC ENCODING*. **In: ISCAS - International Symposium on Circuits and Systems**, 2025.

A.3 Resumos Publicados em Outros Eventos

- **Bianca Silveira**, Daniel Palomino, Cláudio Diniz, Guilherme Correa. *ANÁLISE DA TAXA DE UTILIZAÇÃO DE TRANSFORMADAS EM QUADROS INTRA CODIFICADOS SEGUNDO O PADRÃO VVC*. **In: XXIII ENPOS - Encontro de Pós Graduação UFPel**, Pelotas, 2021.
- **Bianca Silveira**, Luis Neto, Daniel Palomino, Cláudio Diniz, Guilherme Correa. *ARQUITETURA DE HARDWARE DA MTS PARA O CODIFICADOR VERSATILE VIDEO CODING*. **In: XXIV ENPOS - Encontro de Pós Graduação UFPel, Pelotas**, 2022.

A.4 Apresentações de Trabalhos

- **Bianca Silveira**, Daniel Palomino, Cláudio Diniz, Guilherme Correa.*ANALYSIS OF TRANSFORMS USAGE IN VVC INTRA CODING*. **In: 5th DPVSA - Digital Processing of Visual Signals and Applications**, Pelotas, 2021.
- **Bianca Silveira**, Daniel Palomino, Cláudio Diniz, Guilherme Correa.*MULTIPLE TRANSFORM SELECTION HARDWARE DESIGN FOR VVC*. **In: 6th DPVSA - Digital Processing of Visual Signals and Applications**, Pelotas, 2022.
- **Bianca Silveira**, Luis Neto, Daniel Palomino, Cláudio Diniz, Guilherme Correa.*MULTIPLE TRANSFORM SELECTION HARDWARE DESIGN FOR 4K@60FPS REAL-TIME VERSATILE VIDEO CODING*. **In: SIM - Simpósio Sul de Microeletrônica**, 2022.

APÊNDICE B – Coeficientes das transformadas

Este apêndice apresenta os valores dos coeficientes das matrizes das DCT-II, DCT-VIII e DST-VII extraídos do *software* de referência VTM.

- Coeficientes DCT-II de 4-points

  { 64, 64, 64, 64}

  { 83, 36, -36, -83}

  { 64, -64, -64, 64}

  { 36, -83, 83, -36}

- Coeficientes DCT-II de 8-points

  { 64, 64, 64, 64, 64, 64, 64, 64}

  { 89, 75, 50, 18, -18, -50, -75, -89}

  { 83, 36, -36, -83, -83, -36, 36, 83}

  { 75, -18, -89, -50, 50, 89, 18, -75}

  { 64, -64, -64, 64, 64, -64, -64, 64}

  { 50, -89, 18, 75, -75, -18, 89, -50}

  { 36, -83, 83, -36, -36, 83, -83, 36}

  { 18, -50, 75, -89, 89, -75, 50, -18}

- Coeficientes DCT-II de 16-points

  { 64, 64, 64, 64, 64, 64, 64, 64, 64, 64, 64, 64, 64, 64, 64, 64}

  { 90, 87, 80, 70, 57, 43, 25, 9, -9, -25, -43, -57, -70, -80, -87, -90}

  { 89, 75, 50, 18, -18, -50, -75, -89, -89, -75, -50, -18, 18, 50, 75, 89}

  { 87, 57, 9, -43, -80, -90, -70, -25, 25, 70, 90, 80, 43, -9, -57, -87}

  { 83, 36, -36, -83, -83, -36, 36, 83, 83, 36, -36, -83, -83, -36, 36, 83}

  { 80, 9, -70, -87, -25, 57, 90, 43, -43, -90, -57, 25, 87, 70, -9, -80}

  { 75, -18, -89, -50, 50, 89, 18, -75, -75, 18, 89, 50, -50, -89, -18, 75}

  { 70, -43, -87, 9, 90, 25, -80, -57, 57, 80, -25, -90, -9, 87, 43, -70}

{ 64, -64, -64, 64, 64, -64, -64, 64, 64, -64, -64, 64, 64, -64, -64, 64}

{ 57, -80, -25, 90, -9, -87, 43, 70, -70, -43, 87, 9, -90, 25, 80, -57}

{ 50, -89, 18, 75, -75, -18, 89, -50, -50, 89, -18, -75, 75, 18, -89, 50}

{ 43, -90, 57, 25, -87, 70, 9, -80, 80, -9, -70, 87, -25, -57, 90, -43}

{ 36, -83, 83, -36, -36, 83, -83, 36, 36, -83, 83, -36, -36, 83, -83, 36}

{ 25, -70, 90, -80, 43, 9, -57, 87, -87, 57, -9, -43, 80, -90, 70, -25}

{ 18, -50, 75, -89, 89, -75, 50, -18, -18, 50, -75, 89, -89, 75, -50, 18}

{ 9, -25, 43, -57, 70, -80, 87, -90, 90, -87, 80, -70, 57, -43, 25, -9}

- Coeficientes DCT-VIII de 4-points

  { 84, 74, 55, 29}

  { 74, 0, -74, -74}

  { 55, -74, -29, 84}

  { 29, -74, 84, -55}

- Coeficientes DCT-VIII de 8-points

  { 86, 85, 78, 71, 60, 46, 32, 17}

  { 85, 60, 17, -32, -71, -86, -78, -46}

  { 78, 17, -60, -86, -46, 32, 85, 71}

  { 71, -32, -86, -17, 78, 60, -46, -85}

  { 60, -71, -46, 78, 32, -85, -17, 86}

  { 46, -86, 32, 60, -85, 17, 71, -78}

  { 32, -78, 85, -46, -17, 71, -86, 60}

  { 17, -46, 71, -85, 86, -78, 60, -32}

- Coeficientes DCT-VIII de 16-points

  { 88, 88, 87, 85, 81, 77, 73, 68, 62, 55, 48, 40, 33, 25, 17, 8}

  { 88, 81, 68, 48, 25, 0, -25, -48, -68, -81, -88, -88, -81, -68, -48, -25}

  { 87, 68, 33, -8, -48, -77, -88, -81, -55, -17, 25, 62, 85, 88, 73, 40}

  { 85, 48, -8, -62, -88, -77, -33, 25, 73, 88, 68, 17, -40, -81, -87, -55}

  { 81, 25, -48, -88, -68, 0, 68, 88, 48, -25, -81, -81, -25, 48, 88, 68}

  { 77, 0, -77, -77, 0, 77, 77, 0, -77, -77, 0, 77, 77, 0, -77, -77}

  { 73, -25, -88, -33, 68, 77, -17, -88, -40, 62, 81, -8, -87, -48, 55, 85}

{ 68, -48, -81, 25, 88, 0, -88, -25, 81, 48, -68, -68, 48, 81, -25, -88}

{ 62, -68, -55, 73, 48, -77, -40, 81, 33, -85, -25, 87, 17, -88, -8, 88}

{ 55, -81, -17, 88, -25, -77, 62, 48, -85, -8, 88, -33, -73, 68, 40, -87}

{ 48, -88, 25, 68, -81, 0, 81, -68, -25, 88, -48, -48, 88, -25, -68, 81}

{ 40, -88, 62, 17, -81, 77, -8, -68, 87, -33, -48, 88, -55, -25, 85, -73}

{ 33, -81, 85, -40, -25, 77, -87, 48, 17, -73, 88, -55, -8, 68, -88, 62}

{ 25, -68, 88, -81, 48, 0, -48, 81, -88, 68, -25, -25, 68, -88, 81, -48}

{ 17, -48, 73, -87, 88, -77, 55, -25, -8, 40, -68, 85, -88, 81, -62, 33}

{ 8, -25, 40, -55, 68, -77, 85, -88, 88, -87, 81, -73, 62, -48, 33, -17}

- Coeficientes DST-VII de 4-points

  { 29, 55, 74, 84 }

  { 74, 74, 0, -74 }

  { 84, -29, -74, 55 }

  { 55, -84, 74, -29 }

- Coeficientes DST-VII de 8-points

  { 17, 32, 46, 60, 71, 78, 85, 86}

  { 46, 78, 86, 71, 32, -17, -60, -85}

  { 71, 85, 32, -46, -86, -60, 17, 78}

  { 85, 46, -60, -78, 17, 86, 32, -71}

  { 86, -17, -85, 32, 78, -46, -71, 60}

  { 78, -71, -17, 85, -60, -32, 86, -46}

  { 60, -86, 71, -17, -46, 85, -78, 32}

  { 32, -60, 78, -86, 85, -71, 46, -17}

- Coeficientes DST-VII de 16-points

  { 8, 17, 25, 33, 40, 48, 55, 62, 68, 73, 77, 81, 85, 87, 88, 88}

  { 25, 48, 68, 81, 88, 88, 81, 68, 48, 25, 0, -25, -48, -68, -81, -88}

  { 40, 73, 88, 85, 62, 25, -17, -55, -81, -88, -77, -48, -8, 33, 68, 87}

  { 55, 87, 81, 40, -17, -68, -88, -73, -25, 33, 77, 88, 62, 8, -48, -85}

  { 68, 88, 48, -25, -81, -81, -25, 48, 88, 68, 0, -68, -88, -48, 25, 81}

  { 77, 77, 0, -77, -77, 0, 77, 77, 0, -77, -77, 0, 77, 77, 0, -77}

{ 85, 55, -48, -87, -8, 81, 62, -40, -88, -17, 77, 68, -33, -88, -25, 73}

{ 88, 25, -81, -48, 68, 68, -48, -81, 25, 88, 0, -88, -25, 81, 48, -68}

{ 88, -8, -88, 17, 87, -25, -85, 33, 81, -40, -77, 48, 73, -55, -68, 62}

{ 87, -40, -68, 73, 33, -88, 8, 85, -48, -62, 77, 25, -88, 17, 81, -55}

{ 81, -68, -25, 88, -48, -48, 88, -25, -68, 81, 0, -81, 68, 25, -88, 48}

{ 73, -85, 25, 55, -88, 48, 33, -87, 68, 8, -77, 81, -17, -62, 88, -40}

{ 62, -88, 68, -8, -55, 88, -73, 17, 48, -87, 77, -25, -40, 85, -81, 33}

{ 48, -81, 88, -68, 25, 25, -68, 88, -81, 48, 0, -48, 81, -88, 68, -25}

{ 33, -62, 81, -88, 85, -68, 40, -8, -25, 55, -77, 88, -87, 73, -48, 17}

{ 17, -33, 48, -62, 73, -81, 87, -88, 88, -85, 77, -68, 55, -40, 25, -8}