UNIVERSIDADE FEDERAL DO PAMPA

GABRIEL BITENCOURT CARDOSO

ANÁLISE E DESIGN EFICIENTE PARA A GERAÇÃO DOS ELEMENTOS SINTÁTICOS RESIDUAIS SEGUNDO O PADRÃO VVC

Bagé
2026

GABRIEL BITENCOURT CARDOSO

ANÁLISE E DESIGN EFICIENTE PARA A GERAÇÃO DOS ELEMENTOS SINTÁTICOS RESIDUAIS SEGUNDO O PADRÃO VVC

Trabalho de Conclusão de Curso apresentado ao curso de Bacharelado em Engenharia de Computação como requisito parcial para a obtenção do grau de Bacharel em Engenharia de Computação.

Orientador: Julio Saraçol Domingues Júnior

Bagé
2026

Ficha catalográfica elaborada automaticamente com os dados fornecidos pelo(a) autor(a) através do Módulo de Biblioteca do Sistema GURI (Gestão Unificada de Recursos Institucionais).

C268a Bitencourt Cardoso, Gabriel

Análise e design eficiente para a geração dos elementos sintáticos residuais segundo o padrão VVC / Gabriel Bitencourt Cardoso.

73 f.: il.

Orientador: Julio Saraçol Domingues Júnior

Trabalho de Conclusão de Curso (Graduação) - Universidade Federal do Pampa, Engenharia de Computação, 2026.

1. Codificação de vídeo. 2. VVC. 3. Elementos sintáticos residuais. 4. Transformada. 5. Transform-skip. 6. Codificação de entropia. I. Título.

SERVIÇO PÚBLICO FEDERAL
MINISTÉRIO DA EDUCAÇÃO
Universidade Federal do Pampa

GABRIEL BITENCOURT CARDOSO

ANÁLISE E DESIGN EFICIENTE PARA A GERAÇÃO DOS ELEMENTOS SINTÁTICOS RESIDUAIS SEGUNDO O PADRÃO VVC

Trabalho de Conclusão de Curso apresentado ao Curso de Bacharelado em Engenharia de Computação da Universidade Federal do Pampa, como requisito parcial para obtenção do Título de Bacharel em Engenharia de Computação.

Trabalho de Conclusão de Curso defendido e aprovado em: 03 de Julho de 2026.

Banca examinadora:

Prof. Dr. Julio Saraçol Domingues Júnior
Orientador
(UNIPAMPA)

Prof. Dr. Bruno Silveira Neves
(UNIPAMPA)

Prof. Dr. Carlos Michel Betemps
(UNIPAMPA)

Assinado eletronicamente por **JULIO SARACOL DOMINGUES JUNIOR**, **PROFESSOR DO MAGISTERIO SUPERIOR**, em 13/07/2026, às 10:43, conforme horário oficial de Brasília, de acordo com as normativas legais aplicáveis.

Assinado eletronicamente por **CARLOS MICHEL BETEMPS**, **PROFESSOR DO MAGISTERIO SUPERIOR**, em 13/07/2026, às 14:29, conforme horário oficial de Brasília, de acordo com as normativas legais aplicáveis.

Assinado eletronicamente por **BRUNO SILVEIRA NEVES**, **PROFESSOR DO MAGISTERIO SUPERIOR**, em 13/07/2026, às 22:16, conforme horário oficial de Brasília, de acordo com as normativas legais aplicáveis.

A autenticidade deste documento pode ser conferida no site https://sei.unipampa.edu.br/sei/controlador_externo.php?acao=documento_conferir&id_orgao_acesso_externo=0, informando o código verificador **2085461** e o código CRC **CD9C0B21**.

Referência: Processo nº 23100.010883/2026-81 SEI nº 2085461

Dedico este trabalho aos meus pais, meus avós e minha cachorra Athena.

AGRADECIMENTO

Gostaria de agradecer minha família que mesmo distante sempre me apoiou e se fez presente, além disso gostaria de agradecer meus amigos por sempre me ajudarem.

“If I have seen farther than others, it is because I stood on the shoulders of giants.”
— Sir Isaac Newton

RESUMO

O crescente consumo de conteúdo em vídeo, impulsionado por plataformas de *streaming* e mídias sociais, demanda o desenvolvimento de técnicas eficientes de compressão e transmissão de dados. Nesse contexto, destaca-se o padrão *Versatile Video Coding* (VVC/H.266), que oferece taxas de compressão até 50% superiores ao seu antecessor, o HEVC. No entanto, essa eficiência gera um aumento considerável na complexidade computacional, tornando lenta a codificação em dispositivos de uso geral e inviabilizando aplicações em tempo real sem *hardware* dedicado. Este trabalho tem como principal objetivo: realizar uma análise estatística detalhada dos Elementos Sintáticos Residuais (RSEs) no VVC para os modos Transformada (*Transform-based*) e *Transform-skip* e, a partir dela, propor e implementar arquiteturas de *hardware* dedicadas para sua geração. A análise estatística foi realizada com o VVC *Test Model* (VTM) em diversas sequências de teste e parâmetros de quantização (QP), confirmou que os RSEs são majoritários ou têm contribuição significativa para os dados de entrada da codificação de entropia (CABAC), sendo cruciais para a vazão do codificador. Os resultados também evidenciaram que o modo Transformada é majoritário e o *Transform-skip* pode chegar a 26.53% de uso em certas sequências. Sendo os RSES majoritários nota-se a necessidade de uma arquitetura de *hardware* dedicada para a geração desses elementos. As arquiteturas propostas foram descritas em VHDL e sintetizadas utilizando o nó tecnológico TSMC de 40 nm. Operando a uma frequência de 1 GHz, o sistema completo consumiu apenas 3,23 mW de potência e ocupou uma área de 11,8 K gates, enquanto a 1,5 GHz ocupou 15,5 K gates e consumiu 6,11 mW de potência. Em comparação com trabalhos semelhantes na literatura voltados ao VVC, a solução desenvolvida destacou-se por uma economia de área de 70,2% até 77,3% e por dissipar menos da metade da potência na frequência de 1 GHz.

Palavras-chave: Codificação de vídeo; VVC; Elementos sintáticos residuais; Transformada; Transform-skip; Codificação de entropia.

ABSTRACT

The increasing consumption of video content, driven by streaming platforms and social media, demands the development of efficient data compression and transmission techniques. In this context, the Versatile Video Coding (VVC/H.266) standard stands out, offering compression rates up to 50% higher than its predecessor, HEVC. However, this efficiency results in a considerable increase in computational complexity, slowing down encoding on general-purpose devices and making real-time applications unfeasible without dedicated hardware. The main objective of this work is to perform a detailed statistical analysis of the Residual Syntax Elements (RSEs) in VVC for the Transform-base and Transform-skip modes and, based on this analysis, to propose and implement dedicated hardware architectures for their generation. The statistical analysis, performed using the VVC Test Model (VTM) across various test sequences and quantization parameters (QP), confirmed that RSEs are predominant or make a significant contribution to the entropy coding (CABAC) input data, being crucial for the encoder's throughput. The results also evidenced that, while the Transform mode is predominant, the Transform-skip mode can reach up to 26.53% usage in certain sequences. Given that RSEs are predominant, the need for a dedicated hardware architecture for generating these elements is evident. The proposed architectures were described in VHDL and synthesized using the TSMC 40 nm technology node. Operating at a frequency of 1 GHz, the complete system consumed only 3.23 mW of power and occupied an area of 11.8 K gates, while at 1.5 GHz it occupied 15.5 K gates and consumed 6.11 mW of power. Compared to similar works in the literature targeting VVC, the developed solution stood out by achieving area savings from 70.2% to 77.3% and dissipating less than half the power at 1 GHz.

Keywords: Video coding; VVC; Residual syntax elements; Hardware architecture; Transform; Transform-skip; Entropy coding.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

AV1	*AOMedia Video 1*
ASIC	*Application-Specific Integrated Circuit*
BAE	*Binary Arithmetic Encoder*
BN	Binarização
BR	*Base Range*
CABAC	*Context-Adaptive Binary Arithmetic Coding*
CG	*Coding group*
CUs	*Coding units*
CPU	Unidade Central de Processamento
CTU	*Conding Tree Units*
GPU	Unidade de Processamento Gráfico
HEVC	*High-Efficiency Video Coding*
HR	*High Range*
ISP	*Intra Sub-Partitions*
ISO	*International Organization for Standardization*
ITU-T	*International Telecommunication Union - Telecommunication Standardization Sector*
LD	*Low-Delay*
LR	*Low Range*
MBBS	*Multiple Bypass Bin Scheme*
MTT	*Multi-tipe Tree*
MRL	*Multiple Reference Line*
MIP	*Matrix-based Intra Prediction*
MRSET	*Multi Residual Syntax Element Treatment*
QP	*Quantization Parameters*

QT	*Quad-Tree*
RA	*Random-Access*
RAM	*Random Access Memory*
RGB	*Red, Green, Blue*
RSEs	Elementos Sintáticos Residuais
RSEG	*RSEs Generation*
RTL	*Register Transfer Level*
SEs	Elementos Sintáticos
SSD	*Solid State Drive*
TB	*Transform-based*
TU	Unidade da Transformada
TS	*Transform-skip*
TSMC	*Taiwan Semiconductor Manufacturing Company*
VHDL	*VHSIC Hardware Description Language*
YCbCr	*Luminance (Y), Chrominance Blue (Cb), Chrominance Red (Cr)*
JPEG	*Joint Photographic Experts Group*
MPEg	*Moving Picture Experts Group*
VTM	*VVC Test Model*
VVC	*Versatile Video Coding*

SUMÁRIO

1 INTRODUÇÃO

Atualmente, o consumo de conteúdos em vídeo está cada vez mais presente no cotidiano das pessoas, como, por exemplo, em propagandas, plataformas de *streaming*, redes sociais, videoconferências, cursos a distância e *lives*. De acordo com medidas recentes, estima-se que os vídeos representem entre 30% e 60% do tráfego total de dados em dispositivos conectados, variando conforme a região e o tipo de rede utilizada (Ericsson, 2022). Segundo pesquisas recentes, o conteúdo de vídeo constitui aproximadamente 40% do volume total de *downstream* em redes fixas (aumenta para 50% considerando também o conteúdo de televisão) e 35% em redes móveis (Sandvine, 2024), o que se traduz em *exabytes* de dados por mês.

A evolução da tecnologia de video nos últimos anos, como o aumento da resolução (4K, 8K), faz com que a demanda por maior qualidade se torne cada vez mais importante; portanto, torna-se indispensável o desenvolvimento de técnicas e arquiteturas que otimizem a compressão e a transmissão de dados, reduzindo o volume de informação sem que seja necessário comprometer a qualidade visual percebida. Nesse cenário, destacam-se os *codecs* de última geração que lidam melhor com vídeos de qualidade mais alta em comparação com os padrões de codificação anteriores.

Dessa forma, destaca-se o padrão *Versatile Video Coding* (VVC), conhecido como H.266, desenvolvido em parceria pelo Setor de Padronização das Telecomunicações da União Internacional de Telecomunicações (ITU-T) e pela ISO/IEC (ITU-T; ISO/IEC, 2020). O VVC representa o estado da arte em compressão de vídeo, alcançando taxas de compressão até 50% superiores as do seu antecessor, o *High Efficiency Video Coding* (HEVC ou H.265) (ITU-T; ISO/IEC, 2013), mantendo a mesma qualidade subjetiva de imagem. Isso significa que, para um mesmo nível de qualidade visual, o VVC consegue reduzir pela metade o volume de dados necessário.

Os ganhos em compressão vêm acompanhados de um aumento considerável na complexidade computacional, tanto na etapa de codificação quanto na de decodificação. Essa complexidade representa um obstáculo importante, especialmente em dispositivos com recursos limitados ou aplicações em tempo real.

1.1 Motivação

O VVC tem um alto poder de compactação, ocasionando uma alta complexidade computacional para esse padrão de codificação de video, refletindo em maiores tempos de processamento. O trabalho Uhrina *et al.* (2024) mostra que o tempo de codificação do VVC chega a ser de 27 a 174 vezes maior que o tempo de codificação do AV1 (Alliance for Open Media, 2025), que é outro codificador de vídeo de última geração e foi implementado com o objetivo de ser livre de *royalties*, desenvolvido pela *AOMedia*, que é composta por empresas como *Netflix*, Amazon, dentre outras. Diante disso, é evidente que o *software* de referência do VVC apresenta alta complexidade computacional, sendo pouco otimizado para execução em processadores de uso geral. Mesmo em computadores de uso geral, sua operação se torna inviável, conforme demonstra o trabalho de Uhrina *et al.* (2024), que utilizou uma máquina de alto desempenho (descritas na Tabela 1) apenas para executar o codificador e, mesmo assim, seu tempo de codificação foi muito maior que o do AV1 podendo chegar em até 174,5 vezes mais tempo para codificar a mesma sequência de video. Dessa forma, a execução em *software* do *codec* VVC em dispositivos móveis torna-se ainda mais impraticável, tanto pelas limitações de hardware desses equipamentos (capacidade de processamento, memória e consumo de energia), quanto pelo tempo necessário para processar um vídeo completo. Logo, é evidente que são necessários aceleradores de hardware para as partes mais críticas do processo.

Tabela 1 – Descrição do computador dos testes

Processador	AMD Ryzen 9 5950X 16-Core 4000.0 MHz
SSD	Samsung SSD 980 Pro 1TB
RAM	64 GB DDR4 SDRAM
Placa de vídeo	NVIDIA GeForce RTX 3080 Founders Edition LHR
Sistema operacional	Microsoft Windows 10 Education 64-bit

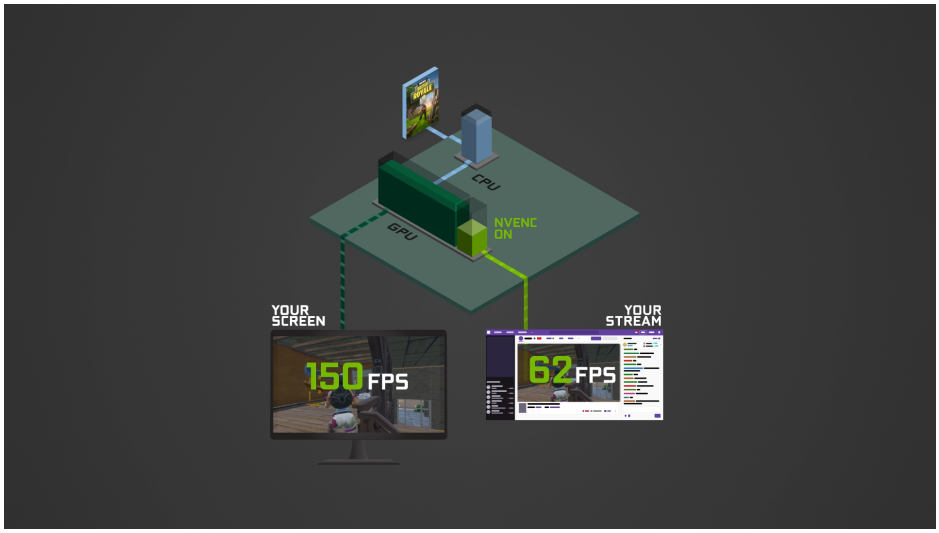
Fonte: O autor, adaptado de (UHRINA *et al.*, 2024).

Os aceleradores em *hardware* podem ser implementados de forma a ser sintetizada em um ASIC (do inglês, *Application-Specific Integrated Circuit*). Um exemplo prático é da adoção desse tipo de abordagem que é encontrado nas placas de vídeo da NVIDIA, que incluem módulos dedicados de codificação de vídeo, o NVENC (NVIDIA Developer, 2025) que fornece codificação de vídeo para os *codecs* H.264, HEVC (H.265) e AV1. Esses módulos constituem blocos de *hardware* independentes da GPU propriamente dita, sendo responsáveis por realizar a codificação de vídeo sem sobrecarregar o processador

ou os núcleos de processamento gráfico.

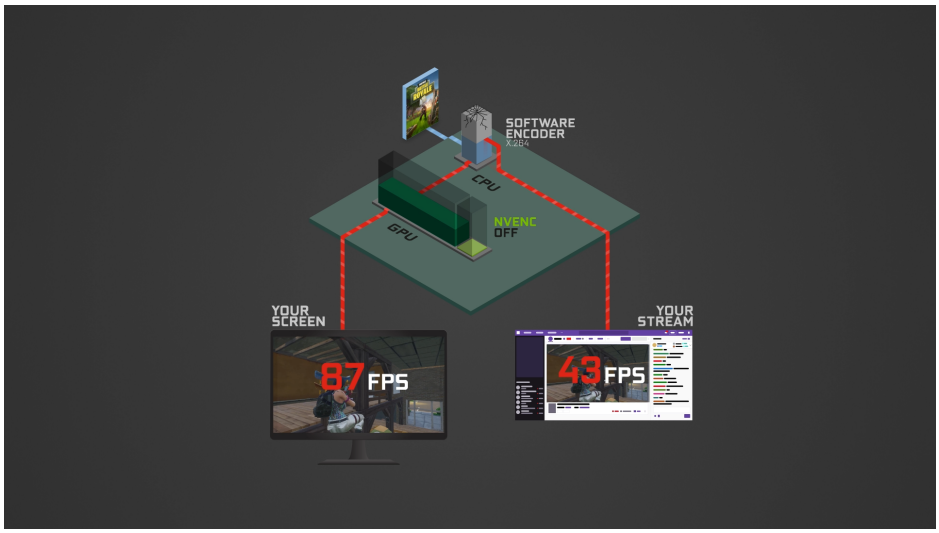
Essa estratégia permite que o sistema direcione os recursos computacionais principais para outras tarefas. No caso das GPU GeForce®, o objetivo é maximizar o desempenho da GPU e da CPU, estes não precisam gastar recursos para codificar video de maneira ineficiente. A diferença prática na alocação de recursos entre o codificador de vídeo implementado em *hardware* (Figura 1) e a implementação em *software* (Figura 2) ilustra esse contraste. Na Figura 2, observa-se que os recursos da CPU são utilizados tanto para a codificação do vídeo quanto para a execução de outras tarefas. Entretanto, por se tratar de um processador de uso geral, a CPU opera com eficiência limitada, pois sua arquitetura de propósito geral demanda ciclos intensivos de software para executar algoritmos complexos, resultando em um uso inadequado de recursos que poderiam ser alocados em outras tarefas do sistema. Já na Figura 1, a presença de um hardware dedicado, como o NVENC, garante uma eficiência superior. Por se tratar de um ASIC (*Application-Specific Integrated Circuit*), esse componente é projetado exclusivamente para a codificação de vídeo, realizando operações de predição e transformada através de caminhos de dados fixos e dedicados. Isso minimiza a latência e o consumo energético, permitindo que a CPU permaneça disponível para o processamento de outras atividades de forma mais adequada.

Figura 1 – Codificador da NVIDIA (NVENC)



Fonte: O autor, adaptado de NVIDIA (2025).

Figura 2 – Codificação em video em software



Fonte: O autor, adaptado de NVIDIA (2025).

Seguindo o mesmo princípio, um *codec* implementado como ASIC poderia ser integrado diretamente em dispositivos como televisores, placas de vídeo e *smartphones*. Isso torna o *codec* viável do ponto de vista de desempenho em tempo real, uma vez que a execução dedicada aumenta a eficiência em relação a soluções puramente baseadas em *software*.

Nos outros formatos de codificação, existem trabalhos sobre o AV1 (GOMES *et al.*, 2023) e o HEVC (RAMOS *et al.*, 2019), que realizaram uma análise estatística. Os elementos sintáticos residuais tendiam a ser maioria dos dados de entrada para a codificação de entropia, logo pelo trabalho se tratar de um novo padrão de codificação de vídeo é valido o questionamento se os elementos sintáticos residuais são a maioria dos dados de entrada do padrão de codificação de vídeo do VVC. Essa hipótese foi confirmada por meio da análise estatística deste trabalho, logo é válida a implementação de um *hardware* específico para a geração dos elementos sintáticos residuais do VVC.

1.2 Objetivo Geral

Este trabalho tem como objetivo principal responder ao seguinte questionamento: os elementos sintáticos residuais representam a maior parte dos dados no processo de codificação de vídeo do padrão VVC? Caso esse questionamento esteja correto, ou seja, os elementos sintáticos residuais são predominantes no VVC é possível propor aceleradores em *hardware* dedicados (para gerar os RSEs do VVC) e a partir desses módulos do codificador realizar extrações de parâmetros da síntese ASIC para realizar comparações

desses módulos com propostas de outros padrões ou soluções similares. Para validar essa hipótese e propor uma solução, este trabalho realiza uma análise estatística dos elementos residuais nos modos de Transformada (*Transform-based*) e *Transform-skip*. Além disso, são implementadas arquiteturas de *hardware* específicas para o processamento desses dados.

1.2.1 Objetivos Específicos

Para ter uma visão melhor e progressiva dos objetivos específicos deste trabalho, estes são expostos abaixo:

- Realizar a análise estatística dos elementos residuais;
- Implementar arquiteturas de *hardware* para geração dos elementos sintáticos residuais para o modo *Transform-based* e para o modo *Transform-skip*;
- Validar as arquiteturas de *hardware* para geração dos elementos sintáticos residuais para o modo *Transform-skip* e modo *Transform-based*;
- Realizar a síntese das arquiteturas de *hardware* para geração dos elementos sintáticos residuais para o modo *Transform-skip* e modo *Transform-based*;
- Comparar as arquiteturas de *hardware* para geração dos elementos sintáticos residuais para o modo *Transform-skip* e modo *Transform-based* com trabalhos correlatos;

1.3 Organização do Texto

No **Capítulo 2**, apresentam-se os fundamentos essenciais de codificação de vídeo, abordando conceitos de processamento, compressão e funcionamento geral de *codecs* híbridos, além de algumas características do VVC. O **Capítulo 3** descreve detalhadamente os elementos sintáticos residuais do padrão VVC, incluindo seus modos de operação e características específicas. No **Capítulo 4** é apresentada a metodologia científica adotada. A Revisão da Literatura, contendo trabalhos relevantes e soluções correlatas, é discutida no **Capítulo 5**. O **Capítulo 6** apresenta os resultados obtidos na análise estatística dos elementos sintáticos residuais, bem como a discussão desses resultados. O **Capítulo 7** detalha as arquiteturas propostas em *hardware* para a geração dos elementos

sintáticos residuais do VVC, tanto para o modo *Transform-based* quanto para o modo *Transform-skip*, apresentando também a validação funcional e os resultados de síntese. Por fim, o **Capítulo 8** reúne as considerações finais, destacando os resultados obtidos e as conclusões do trabalho. Além disso, contém as publicações do autor deste trabalho.

2 REFERENCIAL TEÓRICO DE CODIFICAÇÃO DE VÍDEO

Este capítulo apresenta os fundamentos teóricos necessários para a compreensão dos processos de codificação de vídeo digital, servindo como base conceitual para o desenvolvimento do trabalho. Inicialmente, são discutidos os conceitos básicos relacionados a vídeos digitais, incluindo a composição por *frames*, taxa de *frames* e representação de cores, bem como os principais espaços de cores e técnicas de subamostragem utilizadas na redução de dados. Em seguida, são abordados os princípios da codificação e compressão de vídeo, destacando os mecanismos de exploração de redundâncias espaciais e temporais, as etapas dos *codecs* híbridos, como predição, transformada e quantização. Por fim, o capítulo apresenta uma visão geral do padrão de codificação de vídeo VVC (H.266), descrevendo suas principais características.

2.1 Conceitos Básicos

Esta seção apresentará alguns conceitos básicos que são importantes para o entendimento de imagem, codificação de vídeo e as arquiteturas apresentadas ao longo da monografia.

2.1.1 Quadro e taxa de amostragem

Para compreender os processos de codificação, é fundamental definir os elementos primários que compõem o vídeo digital. De acordo com Manoel (2007) e Ries (2008), um vídeo consiste em uma sucessão temporal de imagens estáticas, denominadas *frames* ou quadros. Quando exibidos em alta velocidade, exploram a persistência retiniana do olho humano para criar a ilusão de movimento contínuo (Figura 3). Cada quadro, por sua vez, é formado por uma matriz bidimensional de *pixels*, que são as menores unidades de informação visual, responsáveis por armazenar os dados de brilho e cor. A qualidade e a fluidez da experiência visual dependem diretamente de duas grandezas: a resolução espacial, que dita a quantidade de *pixels* por quadro e afeta a nitidez; e a resolução temporal, ou taxa de quadros por segundo (FPS), que define a suavidade da percepção de movimento.

Figura 3 – Exemplo de sequência de vídeo em frames

Fonte: (HitPaw, 2025)

Outro conceito importante são as cores que compõem os vídeos, conforme ressalta Iain E. Richardson em Richardson (2010):

> *"A maioria das aplicações de vídeo digital depende da exibição de vídeo colorido e, portanto, precisa de um mecanismo para capturar e representar informações de cor. Uma imagem monocromática requer apenas um número para indicar o brilho ou a luminância de cada amostra espacial. Imagens coloridas, por outro lado, requerem pelo menos três números por posição de pixel para representar a cor com precisão. O método escolhido para representar brilho, luminância ou luma e cor é descrito como um espaço de cores."* (RICHARDSON, 2010, p.12).

A seguir, na Figura 4, está um exemplo de imagem monocromática conforme os conceitos expostos acima.

Figura 4 – Imagem monocromática

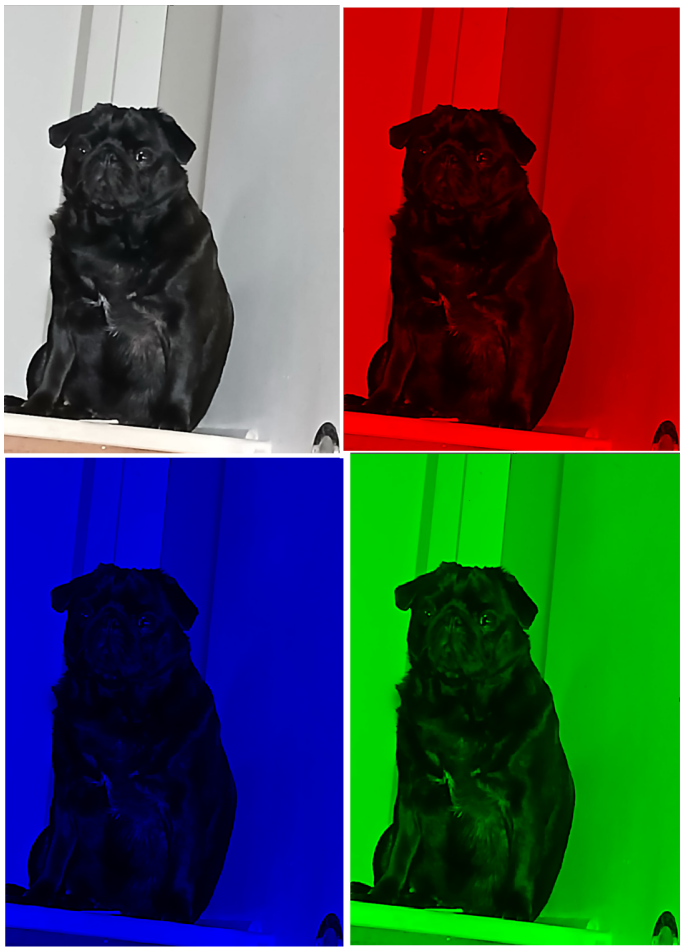
Fonte: (RICHARDSON, 2010)

2.1.2 Espaço de cores

O conceito de cor está presente nos vídeos e é essencial para produzir significado e realismo. Para representar brilho e cor, essas informações precisam ser descritas por pelo menos três valores para cada posição de *pixel*, garantindo precisão na reprodução visual.

O sistema de cores RGB é composto pelos três componentes que representam as seguintes cores: vermelho, verde e azul, como apresenta a Figura 5, na qual é possível visualizar uma decomposição da imagem RGB.

Figura 5 – Decomposição de imagem RGB Athena



Fonte: O autor.

> *"A combinação de vermelho, verde e azul em proporções variáveis pode criar qualquer cor."* (RICHARDSON, 2010, p. 12).

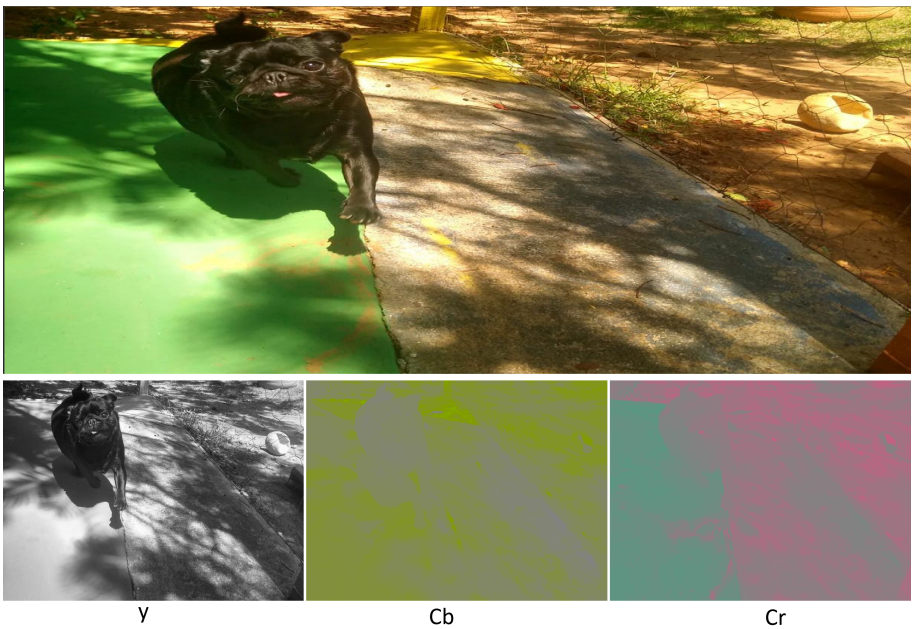
Segundo Richardson (2010), o modelo RGB é o padrão primário para a captura e exibição em sistemas digitais. O processo de captura nesse espaço de cores consiste em separar a

luz da cena em três matrizes de sensores distintas, sensíveis aos comprimentos de onda do vermelho, verde e azul. Da mesma forma, os dispositivos de exibição modernos, como monitores e televisores, operam com base nesse sistema, emitindo luz proporcionalmente nessas três frequências para sintetizar o espectro de cores visível ao usuário.

Apesar de sua adequação para captura e exibição, o formato RGB apresenta forte correlação entre seus canais, o que o torna ineficiente para processos de compressão. Para contornar essa redundância, Richardson (2010) destaca a conversão para o espaço de cores YCbCr (frequentemente denotado como Y:Cr:Cb). Nesse modelo, a informação visual é separada em um canal de luminância (Y), que carrega os dados de brilho e percepção de detalhes espaciais, e dois canais de crominância (Cb e Cr), que armazenam as informações relativas às diferenças de cores azul e vermelha, respectivamente (Figura 6).

A grande vantagem do espaço YCbCr reside na exploração das características do sistema visual humano. Como o olho humano possui uma sensibilidade significativamente maior a variações de brilho (luminância) do que a variações de cor (crominância), torna-se possível reduzir a resolução espacial dos canais Cb e Cr sem que ocorra uma degradação perceptível na qualidade subjetiva da imagem. Essa técnica, conhecida como subamostragem de croma, configura-se como um dos princípios fundamentais e mais eficazes para a redução inicial da taxa de dados em sistemas de codificação de vídeo (RICHARDSON, 2010).

Figura 6 – Decomposição de imagem YCbCr Athena

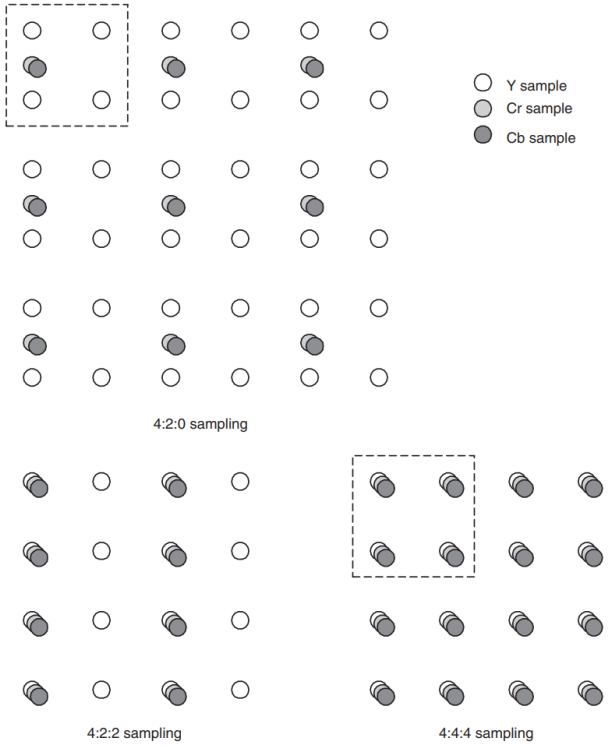


Fonte: O autor.

2.1.3 Subamostragem

De acordo com Richardson (2010), os formatos de amostragem Y:Cr:Cb, também chamados de esquemas de chroma *subsampling*, são métodos cruciais usados na compressão digital de vídeo e imagens para reduzir a quantidade de dados. Esses formatos aproveitam a menor sensibilidade do olho humano à cor do que ao brilho (luminância, Y) para reduzir seletivamente a resolução espacial dos componentes de crominância CrCb em comparação com o componente Y. Os esquemas são tipicamente representados por uma notação de três dígitos, J:a:b, que descreve a frequência relativa de amostragem dos componentes de crominância em uma grade de *pixels*. Os formatos mais comuns estão representados na Figura 7 e são o 4:4:4 (amostragem total sem compressão de croma, com a mesma resolução para Y, Cr e Cb), o 4:2:2 (amostragem da croma pela metade na horizontal, frequentemente usada em estúdios de *broadcast*) e o 4:2:0 (amostragem da croma pela metade tanto na horizontal quanto na vertical, sendo o mais comum em formatos de consumo como JPEG e MPEG) (RICHARDSON, 2010).

Figura 7 – Amostra dos padrões 4:2:0, 4:2:2 e 4:4:4

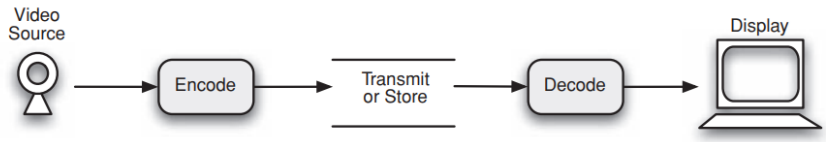


Fonte: (RICHARDSON, 2010)

2.2 Codificação e Compressão de video

Segundo Richardson (2010): A compressão é o ato ou processo de compactar dados em um número menor de *bits*. A compressão de vídeo (codificação de vídeo) é o processo de converter vídeo digital em um formato adequado para transmissão ou armazenamento, reduzindo geralmente o número de *bits* para representar o vídeo. A compressão envolve um par de sistemas complementares: um compressor (codificador) e um descompressor (decodificador) como mostra a Figura 8. O codificador converte os dados de origem em um formato comprimido que ocupa um número reduzido de *bits*, antes da transmissão ou armazenamento, e o decodificador converte o formato comprimido de volta em uma representação dos dados de vídeo originais. O par codificador/decodificador é frequentemente descrito como um *Codec*.

Figura 8 – Encoder e Decoder



Fonte: (RICHARDSON, 2010).

De acordo com Richardson (2010), a codificação de vídeo é uma forma de compressão de vídeo, porém existem várias técnicas e alguns mecanismos que se aproveitam da redundância de dados nos vídeos que são imperceptíveis ao olho humano, como alguns dados do sistema Y:Cr:Cb, assim diminuindo o máximo possível os dados a serem transmitidos.

A compressão de dados é obtida removendo redundâncias, ou seja, componentes que não são necessários para a reprodução fiel dos dados. Muitos tipos de dados contêm redundância estatística e podem ser comprimidos de forma eficaz usando compressão sem perdas, de modo que os dados reconstruídos na saída do decodificador sejam uma cópia perfeita dos dados originais. Infelizmente, a compressão sem perdas de imagens e vídeos proporciona apenas um nível moderado de compressão.

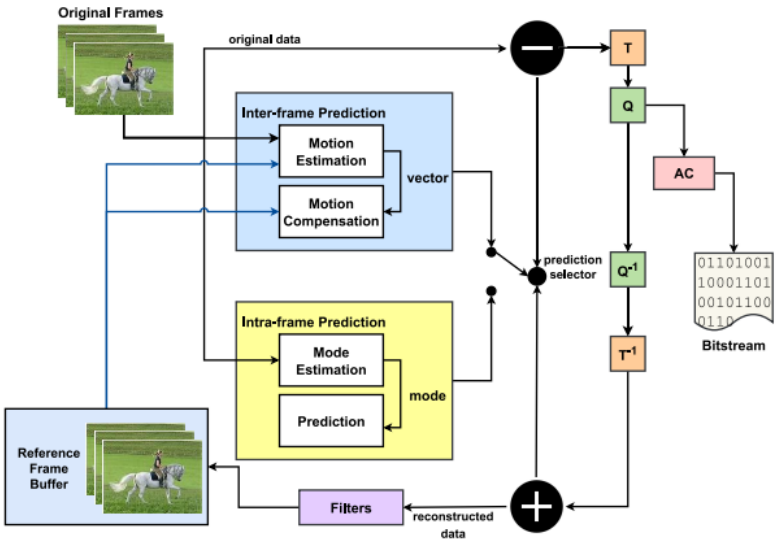
Para obter compressões maiores, é necessária a compressão com perdas. Em um sistema de compressão com perdas, os dados descomprimidos não são idênticos aos dados de origem, e taxas de compressão muito maiores podem ser alcançadas à custa de perda na qualidade visual. Sistemas de compressão de vídeo com perdas baseiam-se no princípio de remover redundâncias subjetivas, isto é, elementos da imagem ou sequência de vídeo

que podem ser excluídos sem afetar significativamente a percepção de qualidade visual pelo observador.

Conforme Richardson (2010), a maioria dos métodos de codificação de vídeo explora tanto a redundância temporal quanto a espacial para alcançar compressão. No domínio temporal, geralmente, há uma alta correlação ou similaridade entre quadros de vídeo capturados em instantes próximos. Quadros temporalmente adjacentes, ou seja, sucessivos no tempo, são frequentemente altamente correlacionados, especialmente quando a taxa de amostragem temporal ou taxa de *frames* é alta. No domínio espacial, normalmente há grande correlação entre *pixels* (amostras) que estão próximos uns dos outros; isto é, os valores das amostras vizinhas costumam ser muito semelhantes. Os *codecs* modernos utilizam uma arquitetura híbrida, combinando técnicas de predição que exploram redundâncias entre *pixels* e quadros para gerar um resíduo e de Transformada que convertem esse resíduo para o domínio da frequência, permitindo uma compressão mais eficiente como apresenta a Figura 9.

Os *codecs* modernos utilizam uma arquitetura híbrida, combinando técnicas de predição que exploram redundâncias entre *pixels* e quadros para gerar um resíduo e de Transformada que convertem esse resíduo para o domínio da frequência, permitindo uma compressão mais eficiente.

Figura 9 – Representação simplificada de um codec de vídeo híbrido tradicional.



Legenda: T:Transformada; Q: Quantização; Q^{-1}: Quantização reversa; T^{-1}: Transformada reversa; AC: Codificação de Entropia.

Fonte: (GOMES *et al.*, 2025).

A Figura 9 apresenta as etapas que o vídeo original deve seguir para ser transformado em um *bitstream* (representação codificada do vídeo) :

- Primeiramente, o processo começa pela separação do *frame* de entrada em blocos de tamanhos variados, usualmente dependentes do nível de detalhe (áreas com mais detalhes são divididas em blocos menores).
- *Intra* e *Inter-frame* predição, que utilizam informações de blocos já codificados para predizer o bloco atual.
- Transformada, que converte blocos residuais (valores restantes após a predição) para o domínio da frequência.
- Quantização, que reduz a precisão dos coeficientes da transformada, removendo componentes de alta frequência pouco perceptíveis ao olho humano.
- Codificação de entropia, que codifica os coeficientes quantizados em um *bitstream* com base na distribuição probabilística dos dados.
- A transformada e quantização reversas são aplicadas no processo de decodificação para reconstrução do *frame*.
- O *loop* de filtros é utilizado durante a codificação para reduzir artefatos gerados pelo processo de compressão (GOMES *et al.*, 2025).

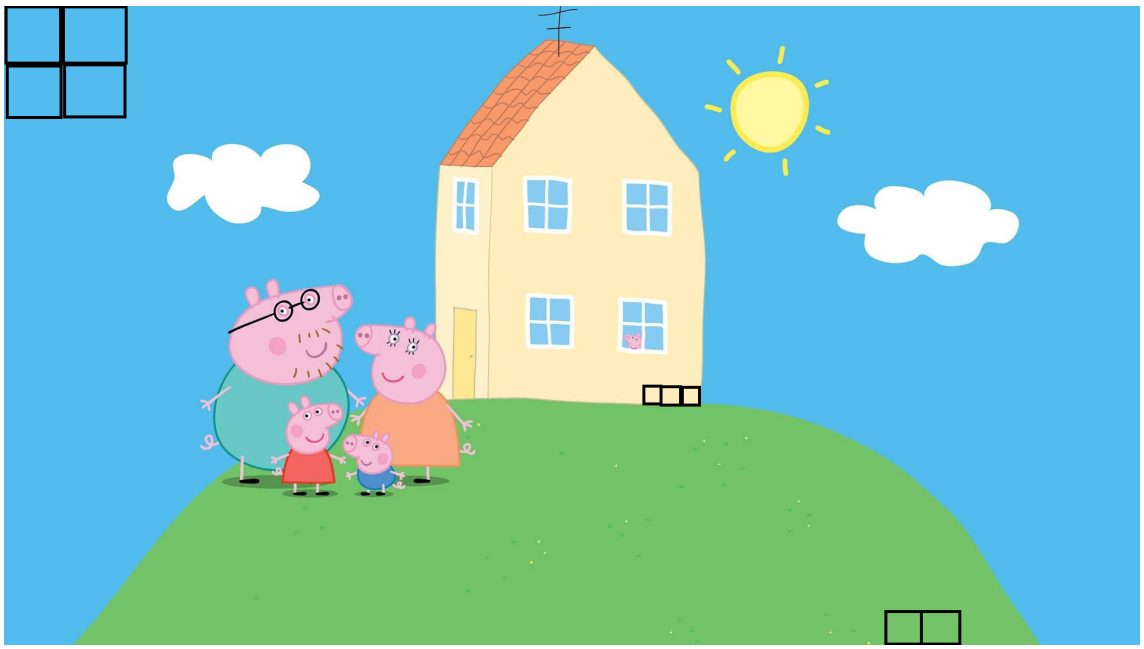
2.2.1 Predições (*Intra e Inter-frame*)

A arquitetura de um codificador de vídeo híbrido baseia-se primordialmente na etapa de predição, cujo objetivo é minimizar a quantidade de dados a serem transmitidos ao explorar correlações existentes no próprio conteúdo visual. Em vez de codificar os valores absolutos dos *pixels*, o sistema gera uma estimativa (predição) para o bloco atual com base em informações previamente processadas, calculando e transmitindo apenas a diferença (resíduo) entre o bloco original e a sua estimativa (RICHARDSON, 2010). Essa estimativa pode ser formulada de duas maneiras: explorando o contexto espacial do mesmo quadro (predição intra) ou o contexto temporal de quadros adjacentes (predição inter).

A predição *intra-frame* atua exclusivamente no domínio espacial, processando cada quadro de forma independente, sem referência a instantes temporais passados ou futuros. O princípio dessa técnica é a exploração das redundâncias presentes dentro da própria imagem, aproveitando a alta probabilidade de que amostras vizinhas

compartilhem características de cor e luminância semelhantes. A Figura 10 ilustra esse conceito de forma prática: áreas extensas, como o céu azul ou as paredes da casa, apresentam blocos de *pixels* com valores quase idênticos, permitindo que a predição espacial alcance alta eficiência ao propagar esses valores direcionalmente. Os quadros codificados inteiramente com essa técnica são denominados quadros I (*Intra-coded frames*) e atuam como âncoras essenciais para o acesso aleatório e a decodificação da sequência de vídeo.

Figura 10 – Frame do desenho Peppa Pig.

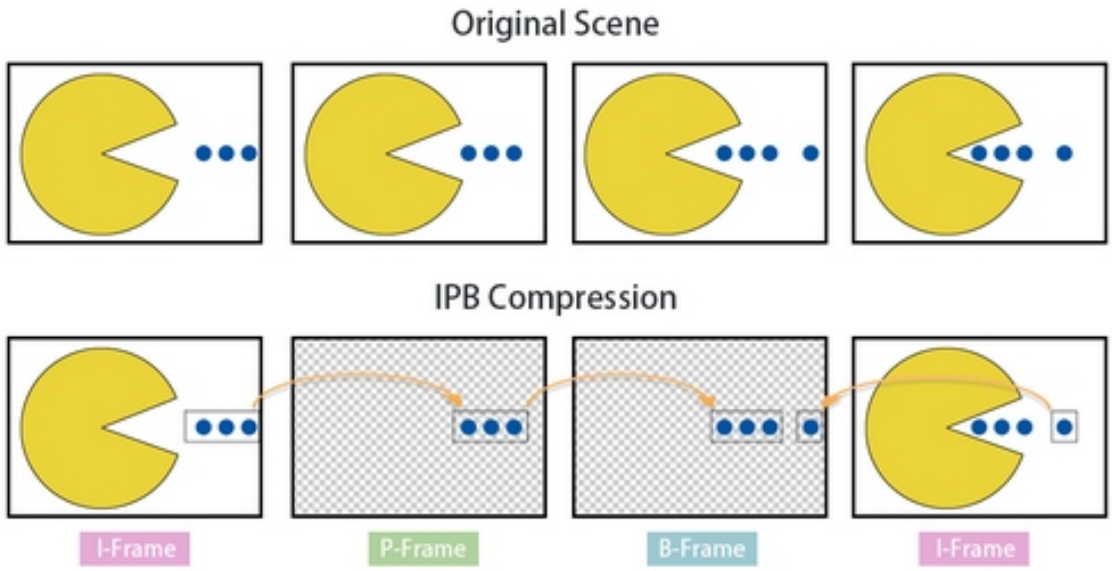


Fonte: O autor.

Por outro lado, a predição *inter-frame* visa explorar a redundância temporal inerente às sequências de vídeo, operando sob a premissa de que quadros consecutivos costumam apresentar variações mínimas, muitas vezes resultantes de pequenos deslocamentos de objetos ou de movimentos da própria câmera. Nesse processo, conhecido como compensação de movimento, o codificador busca o melhor casamento possível para o bloco atual inspecionando um ou mais quadros de referência que já foram codificados e decodificados. O resultado dessa busca geométrica é traduzido em um vetor de movimento, garantindo que apenas a diferença (resíduo) da subtração entre o bloco atual e sua predição inter seja de fato transmitida. Dependendo do sentido adotado nessa busca, os quadros podem ser classificados como quadros P (*Predictive*), que utilizam predições restritas ao passado, ou quadros B (*Bi-predictive*), que fundem predições provenientes do passado e do futuro para extrair uma estimativa altamente

precisa. A Figura 11 exemplifica o mapeamento temporal derivado dessa mecânica (RICHARDSON, 2010).

Figura 11 – Quadros de vídeo subsequentes (da esquerda para a direita) com quadros I, P e B.



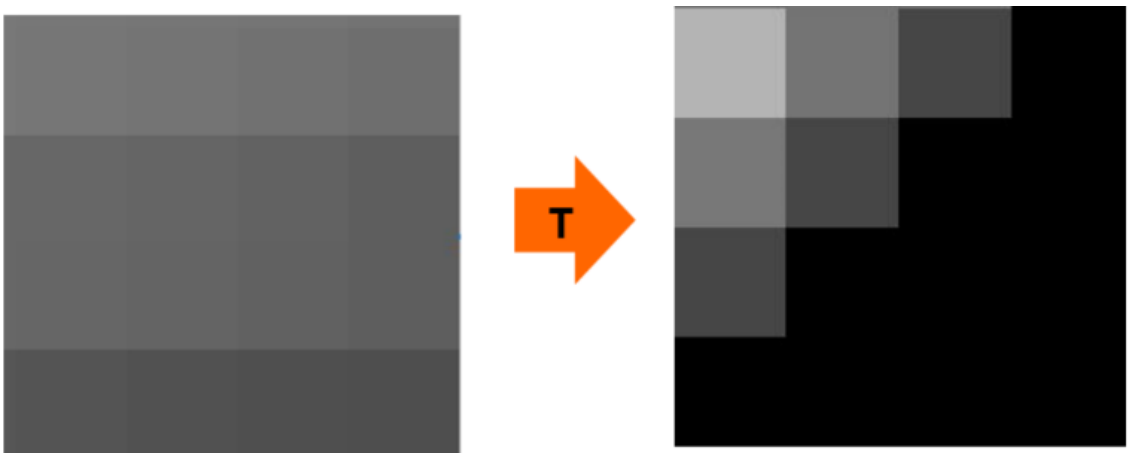
Fonte: (Canon Professional eXchange, 2025).

2.2.2 Transformadas

A transformada é uma etapa matemática essencial nos sistemas de codificação de vídeo, projetada para reduzir a correlação espacial entre os sinais residuais de um bloco de imagem. Matematicamente se trata de uma operação linear que reorganiza as amostras de entrada que geralmente possuem forte correlação com seus vizinhos em uma nova matriz de dados. Os elementos resultantes dessa operação são chamados de coeficientes de transformada. O objetivo central dessa conversão é derivar coeficientes que sejam quase totalmente não correlacionados entre si. Para compreender o papel da transformada na arquitetura de compressão, é importante diferenciá-la da etapa de predição que tenta remover a redundância estatística lidando com a correlação entre as amostras de um bloco de referência e as do bloco atual, a transformada atua de forma complementar, explorando e mitigando a correlação restante entre as amostras internas do próprio bloco residual. É importante destacar que, isoladamente, a transformada é um processo sem perdas; ou seja, a aplicação da transformada inversa aos coeficientes permite reconstruir a fonte de

entrada original com exatidão. A verdadeira compressão do codificador ocorre porque essa descorrelação eficiente da entrada cria um cenário ideal para as etapas subsequentes. Ao produzir coeficientes sem correlação, a transformada viabiliza a etapa de quantização que é um método de codificação com perdas que reduz os níveis dos coeficientes para alcançar altas taxas de compressão e permite que a codificação de entropia final seja aplicada com máxima eficiência (GAO; MA, 2014). A Figura 12 apresenta um exemplo de transformada em um bloco 4x4 de resíduos.

Figura 12 – Exemplo de transformada em um bloco 4x4 de resíduos



Fonte: (RAMOS *et al.*, 2019).

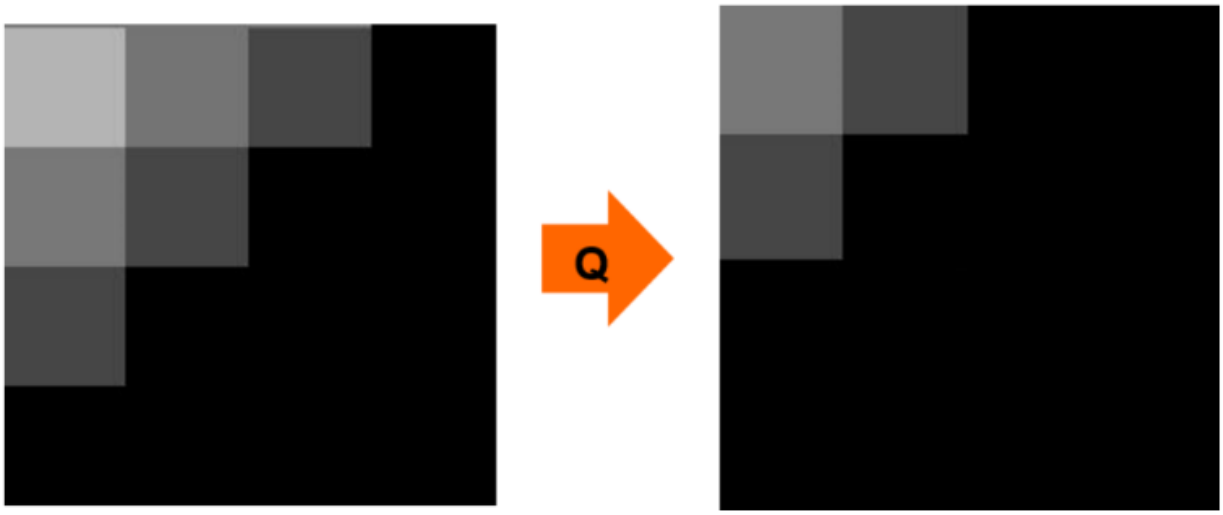
2.2.3 Quantização

No contexto da codificação de vídeo, a quantização atua processando a informação residual do bloco de codificação para remover as frequências que são menos relevantes para a visão humana. Como resultado direto desse descarte de informações, trata-se de um processo com perdas. A intensidade dessa etapa é controlada pelo Parâmetro de Quantização (QP). Valores menores de QP conseguem preservar os detalhes da imagem, enquanto valores mais altos garantem uma maior taxa de compressão, o que, consequentemente, ocorre à custa de perdas na qualidade da imagem (SALDANHA *et al.*, 2021).

Ao final das aproximações da matriz quantizada (representada na Figura 13), o bloco atinge um estado matemático chamado esparso: com um contingente massivo dominado pelo algarismo zero, o que resulta na ideal conformação de base para a futura submissão ao compressor estatístico de entropia na etapa derradeira de formatação da

stream.

Figura 13 – Exemplo de quantização em um bloco 4x4 de residuos



Fonte: (RAMOS *et al.*, 2019).

2.2.4 Codificação de Entropia

A codificação de entropia converte uma série de símbolos que representam elementos da sequência de video em um *bitstream* comprimido para transmissão ou armazenamento. O CABAC (do inglês, *Context-Adaptive Binary Arithmetic Coding*) (MARPE; SCHWARZ; WIEGAND, 2003) é o método de codificação de entropia mais eficiente utilizado nos padrões de vídeo modernos, como H.264/AVC, HEVC e VVC. Seu objetivo é gerar um *bitstream* altamente compacto por meio de um processo aritmético que atribui intervalos de probabilidade a cada símbolo. A característica *"context-adaptive"* significa que as probabilidades não são fixas: elas mudam conforme o comportamento local do vídeo, observando elementos vizinhos, modos de predição, posições de coeficientes e padrões já codificados. Isso garante que cada binário produzido seja codificado segundo o contexto provável, permitindo que os símbolos comuns sejam representados com menos *bits* e aumentando a eficiência da compressão. Esse mecanismo torna o CABAC especialmente eficaz para elementos com grande variabilidade estatística, como coeficientes quantizados e sinais de significância.

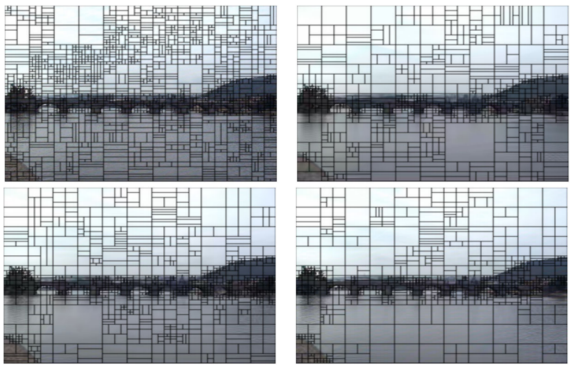
2.3 Padrão de Codificação de Vídeo VVC.

O *Versatile Video Coding* (VVC), também conhecido como H.266, representa o mais recente avanço em padrões de codificação de vídeo, estabelecido em parceria pela ITU-T e pela ISO/IEC (ITU-T; ISO/IEC, 2020). Criado para suceder o *High Efficiency Video Coding* (HEVC/H.265), o VVC é capaz de alcançar taxas de compressão de até 59,9% superiores às de seu antecessor, mantendo a mesma qualidade visual subjetiva (UHRINA *et al.*, 2024). Esse salto em eficiência é importante no cenário atual, onde o consumo de vídeos em resoluções crescentes, como 4K e 8K, além de plataformas de *streaming* e redes sociais, está cada vez mais dominando o tráfego de dados globais. Além disso, o VVC é altamente versátil e eficiente em um maior número de aplicações do que os padrões de codificação de vídeo anteriores, como em transmissões de televisão, serviços de *streaming* de vídeo e comunicação em tempo real (ITU-T; ISO/IEC, 2020).

O VVC, assim como seus predecessores (H.264/AVC e H.265/HEVC), é um *codec* híbrido de compressão. Um sistema de codificação de vídeo é chamado de híbrido porque combina técnicas de predição (para reduzir redundância temporal e espacial) com técnicas de transformação de domínio e quantização (para compactar o sinal residual). Esse esquema, conforme representado na Figura 9, é composto por etapas de predição e transformação, seguidas por um processo de codificação por entropia.

O VVC adota uma nova estrutura de particionamento chamada *Multi-type Tree* (MTT), que estende a *Quad-Tree* (QT) utilizada no HEVC ao introduzir particionamentos binários e ternários, tanto na horizontal quanto na vertical. Inicialmente, os quadros são divididos em *Coding Tree Units* (CTU) com tamanho máximo de 128x128 *pixels*. Uma CTU pode ser particionada em várias *Coding Units* (CUs). As CU podem englobar a região inteira de 128x128, ou até mesmo ter várias CUs, que podem ser de tamanhos: 64x64, 32x32, 16x16, 8x8 e 4x4. Esse particionamento é realizado pela MTT em um processo recursivo no qual uma CU é dividida em outras CUs, até o tamanho mínimo da *Coding Unit* que pode ser de 4x4 *pixels* (BUBOLZ, 2021). Na Figura 14 é possível visualizar imagens particionadas de formas diferentes.

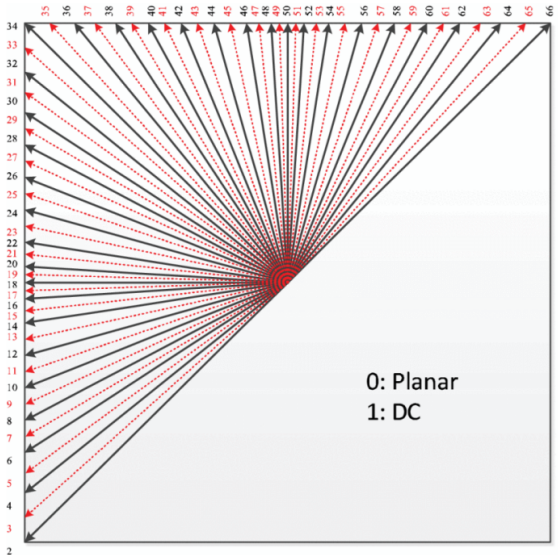
Figura 14 – Exemplo de particionamento



Fonte: O autor, adaptado de(XU *et al.*, 2025).

A predição intra-quadro, responsável por explorar a redundância espacial usando *pixels* vizinhos como referência, teve seus modos expandidos significativamente no VVC, passando de 35 modos no HEVC para um total de 67 modos. Esses modos são divididos em 2 modos não angulares (Planar e DC) e 65 modos angulares, o que oferece uma granularidade muito maior para capturar diferentes direções de bordas e texturas dentro dos blocos de imagem. Os modos angulares buscam realizar a predição através da extrapolação das amostras de referência em diferentes direções, enquanto os não angulares utilizam médias ou interpolações. A Figura 15 representa os 67 modos de predição do VVC.

Figura 15 – Modos de Predição Intra-Quadro no VVC



Fonte: (BUBOLZ, 2021).

Além do aumento nos modos angulares, o VVC introduziu três ferramentas inovadoras de predição intra não convencionais: *Intra Sub-Partitions* (ISP), *Multiple Reference Line* (MRL) e *Matrix-based Intra Prediction* (MIP). O ISP divide o bloco

horizontalmente ou verticalmente em tamanhos iguais; o MRL permite o uso de linhas de referência adicionais (índices 0, 1 e 3) além da linha vizinha imediata; e o MIP utiliza matrizes obtidas por treinamento de redes neurais para realizar predições não lineares, sendo muito eficaz para padrões complexos onde os métodos tradicionais falham.

O funcionamento do CABAC, que é o codificador de entropia utilizado pelo VVC, pode ser dividido em três processos principais: binarização, modelagem de contexto e codificação aritmética. Na binarização, cada símbolo sintático (como modos, *flags*, amplitudes de coeficientes, vetores de movimento) é convertido em uma sequência de *bins* (dígitos binários intermediários). Esses *bins* funcionam como uma camada de abstração que padroniza a representação dos dados, permitindo que o codificador trate diferentes tipos de informação sob uma lógica probabilística uniforme. Em seguida, a modelagem de contexto estima a probabilidade de cada *bin* observando o padrão ao redor; o codificador mantém diversos modelos que se atualizam dinamicamente conforme os *bins* são processados, permitindo que as probabilidades reflitam o conteúdo real do vídeo. Por fim, na codificação aritmética, um intervalo numérico é continuamente subdividido segundo as probabilidades de cada *bin*, resultando em um fluxo binário compacto. Ao longo desse processo, o codificador também trata *bins* chamados *bypass*, codificados sem contexto para símbolos com distribuição uniforme. Essa arquitetura com estimativa adaptativa, múltiplos contextos, renormalização contínua do intervalo e divisão em *bins* regulares e *bypass*, torna o CABAC altamente eficiente, mas também computacionalmente mais complexo que métodos tradicionais de comprimento variável, sendo um dos principais responsáveis pelos elevados ganhos de compressão das gerações recentes de *codecs* de vídeo.

O foco deste trabalho esta antes da codificação de entropia, está na análise estatística dos RSEs e um *hardware* dedicado por gerar os elementos sintáticos residuais que atua como a ponte entre a saída do bloco de quantização e a entrada do codificador CABAC. No próximo capítulo serão apresentados os principais conceitos sobre os elementos sintáticos residuais do VVC.

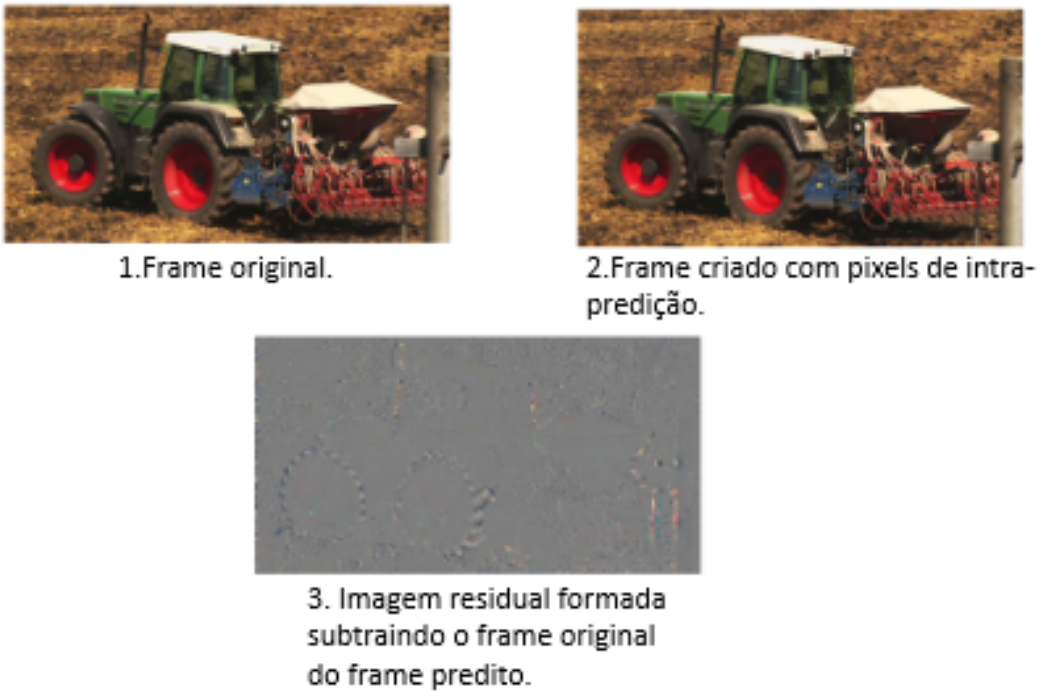
3 ELEMENTOS SINTÁTICOS RESIDUAIS DO VVC

O padrão de codificação de vídeo VVC define um conjunto abrangente de elementos sintáticos responsáveis por descrever, organizar e transmitir as informações necessárias para a reconstrução das imagens no decodificador. Esses elementos sintáticos residuais constituem a estrutura lógica do *bitstream* tendo sido distribuídos em diferentes níveis hierárquicos, desde parâmetros globais da sequência até informações específicas de cada unidade de codificação.

3.1 Elementos sintáticos residuais

Os padrões e formatos de codificação de vídeo utilizam um esquema híbrido, no qual são combinadas etapas de predição e transformada, seguidas por um processo de codificação de entropia, como é explicado no capítulo anterior. Na predição, os *pixels* do quadro são organizados em blocos e busca-se um bloco semelhante em regiões do vídeo que já foram processadas. As discrepâncias entre esses blocos resultam na geração dos chamados dados residuais como apresenta a Figura 16.

Figura 16 – Exemplo de resíduos



Fonte: O autor, adaptado de Cetinkaya (2022).

Após a etapa de predição, ocorre a Transformada na qual os resíduos são reorganizados de acordo com seus componentes de frequência. Em seguida, ocorre a quantização, responsável por atenuar ou descartar componentes de alta frequência, já que o sistema visual humano é menos sensível a eles. Após essas duas etapas, os resíduos passam a ser representados como coeficientes. A última fase do processo é a codificação de entropia. Nela, todas as informações geradas anteriormente são estruturadas e convertidas em Elementos Sintáticos (SEs) antes de entrarem no módulo de entropia. No padrão VVC, utiliza-se exclusivamente o CABAC (MARPE; SCHWARZ; WIEGAND, 2003), que binariza todos os SEs, ou seja, converte-os em sequências de *bits* com apenas dois valores possíveis, chamados *bins*. Esses *bins* são classificados em duas categorias: regular, que abrange aqueles com redundância estatística; e *bypass*, que reúne os símbolos cujos *bins* têm probabilidade igual de ocorrência. Os alfabetos correspondem aos símbolos ou palavras-código usados na camada superior do motor de codificação aritmética.

Elementos sintáticos residuais (RSEs) no VVC podem ser em dois modos: elementos sintáticos residuais para o modo *Transform-based* e para o modo *Transform-skip*, os dois tem algumas semelhanças e diferenças que serão explicadas a seguir. Para ambos os esquemas, a primeira coisa a observar é que a granularidade dos coeficientes é a mesma da transformação bidimensional aplicada a eles.

3.2 Elementos sintáticos residuais do Modo *Transform-based*- Tb-RSEs

Vamos ilustrar com um exemplo, considerando que esse é o primeiro bloco em uma TU (do inglês, *Transform Unit*) que é a unidade onde o resíduo da predição é transformado e quantizado. O primeiro RSE a ser determinado é o ***last*** que é representado por um quadrado lilás no canto superior direito da Figura 17(a). Esse elemento indica a posição do primeiro coeficiente significativo (diferente de zero) encontrado na leitura dos dados (de baixo para cima e da direita para a esquerda). Após a obtenção do ***last***, os demais RSEs são gerados para os coeficientes remanescentes. Com exceção do **rem** (que carrega um valor numérico absoluto), os RSEs listados a seguir comportam-se como sinalizadores (*flags*), recebendo o valor 1 caso a condição que representam seja verdadeira, ou 0 caso contrário:

- **Sig**: indica se o coeficiente é zero (não significativo) ou não-zero(significativo).

- **Gt1**: (do inglês, *greater than one*) Indica se o valor absoluto do coeficiente é maior que um. Ele é gerado para todo coeficiente significativo, ou seja , o **sig** é usado como validade.
- **Par**: (do inglês, *parity*) Indica a paridade de um valor. Ele é gerado para todo coeficiente com valor absoluto maior que um, ou seja, o **gt1** é usado como validade.
- **Gt3**: (do inglês, *greater than three*) Indica se o valor absoluto do coeficiente é maior que três. Ele é gerado para todo coeficiente com valor absoluto maior que um, ou seja, o **gt1** é usado como validade.

Os quatro RSEs anteriores são *flags*, ou sejam, possuem um *bit*, e implicam na criação de *bins* regulares. Os próximos dois são responsáveis pela produção dos *bins bypass*:

- **Sign**: indica o sinal de um coeficiente (positivo ou negativo).
- **Rem**: Coeficientes maiores que três em módulo (valor absoluto) são subtraídos por quatro e dividido por dois gerando a informação remanescente do coeficiente.

O padrão VVC possui um limite de *bins* regulares por unidade de Transformada (TU), com o nome *budget*. Na Figura 17, foram escolhidos valores didáticos para exemplificar a geração dos Tb-RSEs e o funcionamento do *budget*. A Figura 17(a) apresenta a ordem de varredura (leitura) de uma TU no modo *Transform-based* através das setas vermelhas, que partem do canto inferior esquerdo e possuem sentido diagonal descendente, com sentido da direita para a esquerda.

No modo *Transform-based*, como explicado anteriormente nesta seção, os RSEs são gerados a partir do *last*. O *budget* possui o valor inicial 28 e é decrementado conforme a geração de *bins* regulares. Na primeira coluna da tabela representada na Figura 17(b), é possível observar que o resíduo tem o valor -1. A partir disso a *flag sig* receberá o valor 1 (pois o resíduo é diferente de zero) e a *flag gt1* receberá o valor 0 (uma vez que o módulo do resíduo não é maior que 1). Ou seja, como dois *bins* regulares foram gerados, é necessário subtrair duas unidades (valor 2) do *budget*.

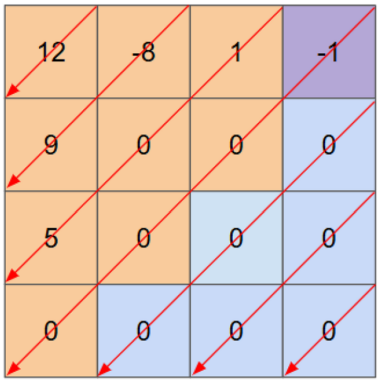
O resíduo seguinte tem o valor 0, resultando na geração de apenas um *bin* regular (o *sig*). Logo, deve-se subtrair uma unidade do *budget* (que no momento é 26), atualizando seu valor para 25, e assim sucessivamente.

Mais à direita na tabela, ao analisar o resíduo de valor -8, a *flag sig* recebe o valor 1, pois o coeficiente é significativo. A *flag gt1* recebe 1, visto que o resíduo é maior em módulo que um; a *flag par* recebe o valor 0, por ser um número par; e a *flag gt3* recebe

1, pois o coeficiente é maior em módulo que três. A *flag sign* recebe o valor 1, indicando que o número é negativo. O elemento *rem* recebe o valor 2, uma vez que o módulo do coeficiente (que é oito) sofre uma subtração de quatro unidades e, em seguida, é dividido por dois.

O *budget*, que anteriormente possuía o valor 16, é decrementado em quatro unidades, pois quatro *bins* regulares foram gerados para esse resíduo (*sig*, *gt1*, *par* e *gt3*). Vale lembrar que *sign* e *rem* são codificados como *bins bypass* e, consequentemente, não influenciam no *budget*. Quando o limite do *budget* é ultrapassado, nenhum dos Tb-RSEs anteriores é gerado (apenas o *sign*), e um novo RSE *bypass-based*, chamado ***DecAbsLevel*** (**dec**), é ativado.

Figura 17 – Exemplo de geração elementos sintáticos residuais do modo *Transform-based*



(a)

Resíduos	-1	0	0	0	1	0	5	-8	9	12
SIG	1	0	0	0	1	0	1	1	1	1
GT1	0	-	-	-	0	-	1	1	1	1
PAR	-	-	-	-	-	-	1	0	1	0
GT3	-	-	-	-	-	-	1	1	1	1
SIGN	1	-	-	-	0	-	0	1	0	0
REM	-	-	-	-	-	-	0	2	2	4
Budget	26	25	24	23	21	20	16	12	8	4

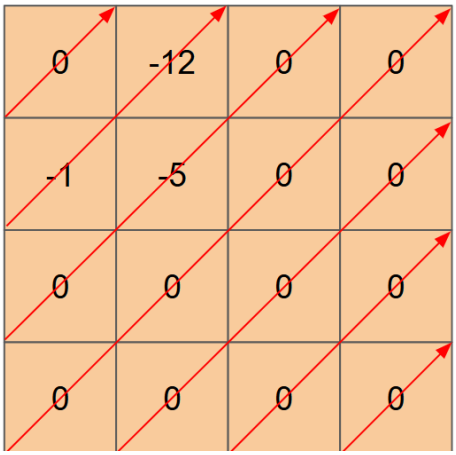
(b)

Fonte: O autor, adaptado de Cardoso *et al.* (2025).

3.3 Elementos sintáticos residuais do Modo *Transforskip*- Ts-RSEs

No modo *Transform-skip* não existe um RSEs análogo ao ***last*** do modo *Transform-based* e a ordem de leitura do CG é do canto superior esquerdo para o canto inferior da direita como apresenta a Figura 18.

Figura 18 – Ordem de Leitura modo Transformskip



Fonte: O autor, adaptado de Cardoso *et al.* (2025).

Existe outro modo de geração dos elementos sintáticos residuais além da *Transform-based*, que é o modo *Transform-skip*.

- Primeira etapa (do inglês, *first pass*)
  - **Sig**: indica se o coeficiente é zero (não significativo) ou não-zero (significativo).
  - **Gt1**: (*greater than one*) Indica se o valor absoluto do coeficiente é maior que um. Ele é gerado para todo coeficiente significativo, ou seja , o **sig** é usado como validade.
  - **Par**: *(parity)* Indica a paridade de um valor. Ele é gerado para todo coeficiente com valor absoluto maior que um, ou seja, o **gt1** é usado como validade.
  - **Sign**: indica o sinal de um coeficiente (positivo ou negativo).
  - **Gt3**: (*greater than three*) Indica se o valor absoluto do coeficiente é maior que três. Ele é gerado para todo coeficiente maior que um, ou seja , o **Gt1** é usado como validade.
- Segunda etapa (do inglês, *second pass*)
  - **Gt5**: (do inglês, *greater than five*) Indica se o valor absoluto do coeficiente é maior que cinco. Ele é gerado para todo coeficiente maior que três, ou seja , o **Gt3** é usado como validade.
  - **Gt7**: (do inglês, *greater than seven*) Indica se o valor absoluto do coeficiente é maior que sete. Ele é gerado para todo coeficiente maior que cinco, ou seja , o **Gt5** é usado como validade.

- **Gt9**: (do inglês, *greater than nine*) Indica se o valor absoluto do coeficiente é maior que nove. Ele é gerado para todo coeficiente maior que sete, ou seja , o **Gt7** é usado como validade.
- **Rem**: (i) uma subtração do valor absoluto por dois seguida de uma divisão por dois, caso exista apenas o gt1, ou (ii) uma subtração do valor absoluto por dez, seguida de uma divisão por dois, caso o gt9 tenha ocorrido para o coeficiente.

A Figura 19 ilustra um exemplo de como os Ts-RSEs são gerados.

Figura 19 – Exemplo da geração de RSE para o modo *Transform-skip*

Resíduos	0	-1	-12	0	-5	0	...	0
SIG	0	1	1	0	1	0	0	0
GT1	-	0	1	-	1	-	-	-
SIGN	-	1	1	-	1	-	-	-
PAR	-	-	1	-	0	-	-	-

(a)

Resíduos	-12	-5
GT3	1	-
GT5	1	-
GT7	1	-
GT9	1	-
REM	1	1

(b)

Fonte: O autor, adaptado de Cardoso *et al.* (2025).

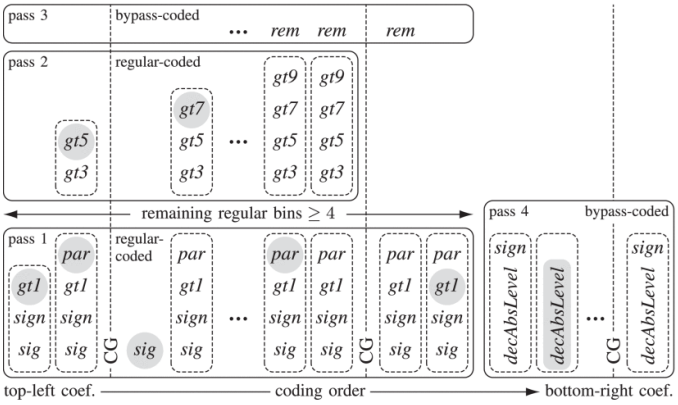
Para começar, até quatro *Regular-coded* RSEs podem ser produzidos por coeficiente: **sig**, **sign**, **par** e **gt1**. Todos estes têm o mesmo significado dos elementos sintáticos residuais do modo *Transform-based* com o mesmo nome. A diferença é que no modo TS (*Transform-Skip*) os resíduos **sign** são correlatos com a posição dos vizinhos, e os Tb-RSE correspondentes são categorizados como *bins* regulares nesse caso. Essa primeira etapa é chamada de *first pass* e está exemplificada na Figura 19(a). Além disso, o limite (*budget*) de *bins* regulares também é aplicado para os quatro Ts-RSEs. No caso de haver menos de 4 *bins* disponíveis do *budget* por grupo de coeficientes de uma unidade de Transformada (TU), o **dec** e o **sign**, que novamente são *bypass-coded* são os únicos RSEs possíveis que podem ser gerados. No entanto, caso ainda exista *budget* disponível para mais *bins* regulares no grupo de coeficientes e todos os resíduos foram lidos, quatro novos *regular-coded* TS-RSEs são produzidos para resíduos que são maiores que um: **gt3**, **gt5**, **gt7** e **gt9**. Essa segunda etapa é chamada de *second pass*. Cada um desses quatro elementos representa se o valor absoluto de um coeficiente é maior que três, cinco, sete e nove, respectivamente. Esses RSEs também decrementam do *budget* sendo "verdadeiros" ou "falsos". Logo o *budget* pode ultrapassar o limite enquanto processa esses elementos,

e, nesse caso, nenhum deles é gerado. Essa situação é ilustrada na Figura 19(b), enquanto o elemento '-12' é processado não tem mais *budget* disponível para o próximo resíduo, que era maior que um, e, com isso, nenhuma *flag* **gts** é criada para o '-5'.

Segundo a existência de **gt1** ou **gt9** para um determinado resíduo, o resíduo também é gerado, mas agora ele corresponde a (i) uma subtração do valor absoluto por dois, seguida por uma divisão por dois, caso exista somente **gt1** ou (ii) uma subtração do valor absoluto por dez, seguida de uma divisão por dois, caso o **gt9** tenha ocorrido para o coeficiente. É possível observar que esses dois comportamentos ocorreram em nosso exemplo na Figura 19(b): para o elemento '-12', o **gt9** ocorreu, tanto por haver budget suficiente quanto pelo valor absoluto do resíduo ser maior que nove, e então o resíduo associado foi decrementado de dez seguido por uma divisão por dois; agora para o valor -5, como o limite permitido para *budget* foi ultrapassado e existe apenas o RSE **gt1** associado a ele, o respectivo comportamento do resíduo **rem** nesse caso é o valor decrementado por quatro e em seguida é dividido por dois, ou seja, o mesmo comportamento do modo *Transform-based*.

A Figura 20 do artigo Schwarz *et al.* (2021) que ajuda a ilustrar e resumir as etapas.

Figura 20 – Exemplo da geração de RSE para o modo *Transform-skip*



Fonte: (SCHWARZ *et al.*, 2021).

4 METODOLOGIA

A metodologia científica adotada neste trabalho foi estruturada conforme a classificação detalhada por Freitas (2013), contemplando as dimensões quanto à natureza, aos objetivos, aos procedimentos técnicos e à forma de abordagem do problema. Essa classificação permite enquadrar o tipo de pesquisa e compreender de forma sistemática os métodos utilizados em cada etapa do desenvolvimento do projeto.

4.1 Caracterização da pesquisa do trabalho

Para garantir o rigor científico e o direcionamento adequado do estudo, a pesquisa foi categorizada em quatro dimensões fundamentais. Essa estruturação permite esclarecer os caminhos adotados para a resolução do problema, conectando a base teórica ao desenvolvimento prático da arquitetura em *hardware*. A seguir, detalham-se as classificações quanto à natureza, aos objetivos, aos procedimentos técnicos e à abordagem do problema.

Esta pesquisa caracteriza-se como aplicada, uma vez que, conforme Freitas (2013), objetiva gerar conhecimentos para aplicação prática dirigidos à solução de problemas específicos. O estudo volta-se para o desenvolvimento de uma solução tecnológica, especificamente uma arquitetura de *hardware* dedicada à geração dos elementos sintáticos residuais no processo de codificação de vídeo do padrão VVC (*Versatile Video Coding*). O foco prático visa contribuir para o aumento do desempenho e da eficiência energética em sistemas de codificação.

A pesquisa apresenta um caráter exploratório e descritivo. É exploratória, pois investiga e delimita um novo enfoque voltado para os elementos residuais do VVC em *hardware*, possibilitando a formulação de arquiteturas otimizadas. Simultaneamente, é descritiva ao envolver a observação, o registro e a análise estatística dos elementos residuais presentes no processo de codificação, caracterizando seu comportamento e relevância dentro do *codec* VVC de forma padronizada.

Em termos de procedimentos técnicos, o trabalho combina pesquisa bibliográfica e pesquisa experimental. Inicialmente, a pesquisa bibliográfica foi empregada para o levantamento de fundamentos teóricos sobre codificação de vídeo, transformadas, compressão e arquiteturas de *hardware*. Na sequência, o trabalho assume um caráter experimental e de desenvolvimento computacional, por meio de simulações no *software*

de referência do VVC (VTM) para coleta de dados sob condições controladas e o posterior desenvolvimento da arquitetura em nível de transferência de registradores (RTL) em VHDL.

A abordagem metodológica do problema é quantitativa. Os resultados são fundamentados na coleta, mensuração e análise estatística de dados numéricos obtidos das modificações no *software* de referência VTM. A avaliação ocorre com base em métricas objetivas, considerando percentuais dos elementos sintáticos residuais, o tempo em modo *Transform-based* ou *Transform-skip*, tipos de *bin* (regular ou *bypass*) e a ocorrência consecutiva do tipo de *bin*, variando-se sequências de teste, resoluções e parâmetros de quantização (QP).

4.2 Delimitação do Estudo

Embora a codificação de vídeo VVC abranja um vasto conjunto de ferramentas de compressão, este estudo delimita seu escopo exclusivamente à análise e otimização dos elementos sintáticos residuais. A aplicação foca na viabilidade de implementações em *hardware* dedicado, onde as restrições de latência e consumo energético são fatores críticos.

4.3 Etapas da Pesquisa

O desenvolvimento desta pesquisa ocorreu de forma estruturada, englobando estudo teórico, experimentação no *software* de referência, projeto de *hardware* e validação funcional, conforme detalhado nas subseções a seguir.

4.3.1 Revisão da Literatura e Trabalhos Correlatos

A revisão teórica e o levantamento de trabalhos correlatos foram conduzidos com o objetivo de mapear o estado da arte sobre o padrão de codificação VVC, especificamente no que tange à geração de elementos sintáticos residuais e o projeto de arquiteturas em *hardware*.

A estratégia de busca utilizou bases de dados científicas amplamente reconhecidas na área de engenharia e ciência da computação, com destaque para a base **IEEE Xplore**,

além do auxílio de plataformas como *Google Scholar*, *ACM Digital Library* e *Springer*.

Para a busca, foram empregadas combinações das seguintes *strings* (palavras-chave):

- *"Versatile Video Coding"OR "VVC"AND "hardware architecture"*
- *"hardware"AND"VVC"AND "residual coding"*
- *"hardware"AND "residual syntax elements"*
- *"transform skip"AND "VVC"AND "hardware"*
- *"entropy coding"AND "residual"AND "VVC"*
- *"Residual syntax elements"AND"architecture"*
- *"Video compression Standard"*
- *"Quantization"AND "Transformation"*
- *"Quantization"AND "Transformation"AND"VVC"*
- *"Video compression Standard"*

Essa etapa fundamentou o embasamento teórico básico e a seleção de arquiteturas correlatas que serviram de comparação para a solução proposta.

Os critérios de escolha do referencial teórico foram menos rigorosos, priorizando fontes dentro do tema de codificação de vídeo, mas que tivessem os conceitos fundamentais com uma linguagem clara e de fácil compreensão. Para os trabalhos correlatos, foram estabelecidos critérios de seleção para filtrar os resultados obtidos nas buscas, como critérios de inclusão, priorizando trabalhos focados em implementações de *hardware* voltadas para a geração de elementos sintáticos residuais, binarização ou aceleração do codificador de entropia (CABAC), com ênfase nos padrões de vídeo, como HEVC, AV1 e VVC. Além disso, buscou-se trabalhos que apresentassem resultados de síntese em nível RTL, para serem utilizados para a comparação com as arquiteturas propostas. Por outro lado, foram adotados como critérios de exclusão artigos com abordagens puramente em *software* que não fornecessem métricas de *hardware*, trabalhos voltados a blocos do codificador não correlacionados aos resíduos.

4.3.2 Estudo Experimental no VTM e Análise de Dados

A segunda etapa envolveu a modificação do *software* de referência do VVC (VTM) para a extração de métricas com o uso de contadores no código. Foram

executadas simulações variando as configurações (resoluções, sequências e parâmetros de quantização) para gerar relatórios detalhados. Em seguida, os dados gerados passaram por uma análise estatística para quantificar a ocorrência dos elementos residuais e as características dos *bins*.

4.3.3 Desenvolvimento Computacional em Nível RTL

A partir dos dados coletados e interpretados, procedeu-se ao desenvolvimento tecnológico. Esta fase consistiu no projeto, descrição e implementação da arquitetura de *hardware* dedicada para geração dos elementos sintáticos residuais do VVC, utilizando a linguagem VHDL. O foco foi otimizar o caminho de dados para a geração eficiente dos elementos residuais.

4.3.4 Síntese e Validação

Por fim, o código VHDL desenvolvido foi submetido a ferramentas de síntese lógica. O desempenho da arquitetura, medido em frequência de operação, área ocupada e eficiência energética, foi validado e comparado com os trabalhos correlatos encontrados durante a revisão da literatura.

5 TRABALHOS CORRELATOS

Neste capítulo, são apresentados os trabalhos correlatos encontrados na literatura que abordam a implementação em *hardware* e a análise estatística de elementos sintáticos residuais em diferentes padrões de codificação de vídeo, como HEVC, AV1 e VVC. A análise detalhada desses trabalhos serve como base para identificar lacunas tecnológicas e fundamentar as decisões de projeto da arquitetura proposta neste trabalho.

5.1 *Low-power HEVC Binarizer architecture for the CABAC block targeting UHD video processing*

Alonso em Alonso *et al.* (2017) propõem uma arquitetura de *hardware* de baixo consumo de energia para o bloco de binarização (*Binarizer*) do CABAC no padrão HEVC, visando o processamento de vídeos na resolução 8K UHD. O objetivo dos autores foi reduzir o consumo de energia do binarizador sem comprometer o desempenho necessário para codificação em tempo real. A metodologia consistiu, primeiramente, na realização de uma análise estatística utilizando sequências de vídeo recomendadas pelo padrão (PeopleOnStreet em resolução WQXGA e BasketballDrive em 1080p HD), rodadas no *software* de referência HM 16.6, com configurações *LowDelay* e *RandomAccess* e parâmetros de quantização (QP) 22 e 37. Essa análise revelou que o método de binarização *Fixed Length* (FL) corresponde a aproximadamente 89% a 93% do total de chamadas de binarização, com até 11 chamadas consecutivas do mesmo método. A partir dessa constatação, os autores aplicaram a técnica de *Operand Isolation* baseada em portas AND para desativar as lógicas de binarização não utilizadas no ciclo atual em uma arquitetura de quatro núcleos (*multi-core*), sendo que o método FL, por ser extremamente simples e o mais chamado, foi excluído da técnica para não aumentar o consumo. Quatro versões da arquitetura foram descritas em VHDL e sintetizadas para a tecnologia de 65nm da ST, nas condições de pior caso (processo, 0,95V e 125°C). Como resultado, a versão de quatro núcleos com baixo consumo (FLP-BIN) atingiu uma frequência máxima de 834 MHz e um rendimento médio de 8,34 *bins/cycle*, com consumo total médio de 1,87 mW, representando uma economia de energia de aproximadamente 22% em relação à versão sem a técnica. A simulação de *gate-level* com sequências de vídeo reais confirmou que o *design* atende aos requisitos para vídeos 8K UHD no nível 6.2 *high tier*, sendo o único trabalho na literatura recente a focar em *design* de baixo consumo para o bloco

binarizador.

5.2 *Residual syntax elements analysis and design targeting high-throughput HEVC CABAC*

Ramos em Ramos *et al.* (2019) introduz um esquema denominado MRSET (*Multi Residual Syntax Element Treatment*) para evitar a falta de dados (*starvation*) nas arquiteturas de CABAC de alto rendimento do HEVC. O objetivo principal foi criar uma solução capaz de gerar os elementos sintáticos (SEs) residuais em paralelo, de modo a acompanhar a alta taxa de consumo dos codificadores de entropia mais recentes, que processam de 4 a 4,9 *bins* por ciclo. Na metodologia, os autores primeiramente analisaram o *software* de referência HM 16.6 e identificaram que os SEs residuais de transformada constituem a maioria dos dados de entrada para o CABAC, variando de 56% a 94% do total de *bins* gerados. A partir disso, observaram que a abordagem do HM, que utiliza múltiplos *loops* e múltiplos acessos à mesma posição de memória, é ineficiente para *hardware*. A solução proposta (MRSET) combina acesso paralelo a múltiplos coeficientes transformados com a geração simultânea de todos os SEs associados a cada posição, em um único acesso à memória. Cinco versões *multi-core* (de 1 a 16 núcleos) foram descritas em VHDL e sintetizadas para tecnologia de 65nm da ST. Os resultados demonstraram que a versão com 4 núcleos (4-MRSET) apresentou o melhor compromisso entre área (3,67 *Kgates*), frequência máxima (668 MHz) e capacidade de entrega de *bins*, processando 2,67 *Giga residues/s*. Adicionalmente, técnicas de *Clock-Gating* e *Operand Isolation* foram aplicadas na versão de baixo consumo (LP-MRSET), obtendo ganhos de aproximadamente 26% a 30% em economia de energia com impacto insignificante em área (cerca de 2%) e degradação de frequência.

5.3 *An efficient hardware implementation of residual data binarization in hevc cabac encoder*

Tran em Tran *et al.* (2020) apresenta uma implementação em *hardware* eficiente para a binarização de dados residuais no codificador CABAC do HEVC. O objetivo principal foi alcançar baixo custo de área e baixo consumo de energia, sem sacrificar a alta taxa de processamento necessária para aplicações UHD. Na metodologia, os autores

observaram que os *bins* de unidades de transformada (*Transform Unit bins*) ocupam em média 75% e até 94% no pior caso da carga total do binarizador. A partir disso, propuseram uma otimização no algoritmo de geração de SEs residuais que substitui as múltiplas varreduras tradicionais (que requerem múltiplos acessos à memória do bloco de transformada) por uma única varredura no grupo de coeficientes, determinando todos os quatro SEs com *flags* (SIG, COEFF1, COEFF2 e SIGN) em um único *datapath*. Adicionalmente, combinaram a binarização das coordenadas X e Y do último coeficiente significativo em um único módulo *Truncated Rice*, economizando área. A arquitetura foi modelada em VHDL, testada com diferentes casos de blocos 4×4 de dados residuais e sintetizada utilizando *Synopsys Design Compiler* com tecnologia *NanGate* de 45nm. Os resultados revelaram um processamento médio de 3,5 SEs/ciclo (3,05 *bins/cycle*) a uma frequência máxima de 500 MHz, com um custo de área de 9,45 *Kgates* (6,41 *Kgates* para o núcleo binarizador) e consumo de apenas 0,239 mW (0,184 mW para o núcleo). A arquitetura destacou-se pela altíssima eficiência energética de 8.288 Mbins/mW, sendo a mais eficiente em termos de *power-efficiency* dentre os trabalhos comparados.

5.4 *AV1 Residual Syntax Elements Assessment and Efficient VLSI Architecture*

Gomes em Gomes *et al.* (2023) propõem a primeira arquitetura de *hardware* da literatura dedicada à geração eficiente de elementos sintáticos residuais para o padrão AV1, denominada RSEG (*Residual Syntax Elements Generator*). O objetivo foi acelerar a etapa de codificação de coeficientes residuais do AV1, uma vez que, assim como em *codecs* anteriores, os SEs residuais tendem a compor a maior parte dos dados processados pelo codificador de entropia. Na metodologia, os autores realizaram uma extensa análise estatística utilizando o *software* de referência libaom v.3.6.0, codificando 60 *frames* de 11 sequências de vídeo do *dataset* objective-2-fast em três resoluções (720p, 1080p e 2160p) com quatro valores de parâmetro de quantização CQ (20, 32, 43 e 55). Os resultados confirmaram que os SEs residuais compõem uma parcela considerável do *bitstream* do AV1, chegando a 80% para CQ 20, enquanto aproximadamente 52% dos blocos são ignorados (*SKIP*) em QPs mais altos. Além disso, a análise de tamanhos de bloco de transformada revelou que o tamanho mais frequente é 64 (blocos 8×8), tornando-se ainda mais prevalente com QPs maiores. Com base nessas análises, a RSEG foi projetada como uma arquitetura *pipeline* de 5 estágios, operando em blocos de 64 coeficientes quantizados, capaz de tratar *arrays* de até 1024 QTCs. O AV1 utiliza um sistema de

codificação por mapa de níveis (*level map*) que decompõe cada coeficiente em quatro símbolos: *Base Range* (BR), *Low Range* (LR), *High Range* (HR) e *Sign bit*. A RSEG gera todos esses SEs em um fluxo *pipeline* que produz 7 SEs por ciclo de *clock*. Os resultados da síntese em tecnologia TSMC de 40nm indicaram que a RSEG atinge uma frequência de 1,1 GHz, consumindo 10,92 mW e ocupando aproximadamente 6,93 Kgates, sendo capaz de entregar cerca de 7 *Giga* SEs por segundo, rendimento mais que suficiente para suprir os codificadores de entropia do AV1 existentes na literatura, como o exemplo de Bitencourt *et al.* (BITENCOURT, 2023), que opera a 590 MHz.

5.5 *Hardware Implementation of A High-Accuracy and High-Throughput Rate Estimation Unit for VVC Residual Coding.*

Liu em Liu *et al.* (2025) descrevem uma unidade de estimação de taxa de alta precisão e alto rendimento voltada para a codificação residual do padrão VVC. O objetivo foi resolver o gargalo de *throughput* gerado pela alta dependência de dados e complexidade introduzidas pelas novas ferramentas do VVC na etapa de Otimização de Taxa-Distorção (RDO). Os autores identificaram que o VVC introduz desafios adicionais em relação ao HEVC: múltiplos tipos de grupos de coeficientes (CGs de 4×4 e 8×2), quatro passadas de codificação (com dependências de atualização de estado e do limite MCCB), e modelagem de contexto avançada com dupla taxa de atualização de probabilidade. Na metodologia, propuseram dois algoritmos de otimização: (i) *Localized State Update* (LSU), que limita as atualizações seriais de contexto a apenas 4 coeficientes de baixa frequência por CG, e (ii) MCCB *Dependency Removal* (MCCBDR), que permite o processamento paralelo dos diferentes grupos de SEs eliminando a dependência do limite de *bins* regulares. Além disso, introduziram uma compressão da tabela de estimação de taxa (de 1,3 KB para 0,04 KB) e um esquema otimizado de armazenamento de informações estatísticas locais (de 1,28 KB para 0,18 KB para TBs 32×32). O *hardware*, implementado em Verilog e sintetizado com GlobalFoundries 28nm, instancia 18 estimadores de taxa em paralelo para suportar diferentes partições MTT do VVC. Como resultado, o *design* suporta a codificação em tempo real de vídeos 8K (7680×4320 a 30fps) operando a 500 MHz, com uma perda de apenas 0,29% no BD-rate em relação ao *software* de referência VTM-19.2. Os autores destacam que este é o primeiro trabalho de implementação em *hardware* de estimação de taxa para o VVC.

5.6 *High-Performance Binary Arithmetic Encoder with Multiple Bypass Bin Scheme for VVC CABAC*

Vínicius em Faria *et al.* (2025) propõe a arquitetura de *hardware* **VArchBAE**, projetada especificamente para o Codificador Aritmético Binário (*Binary Arithmetic Encoder* – BAE) do estágio CABAC (*Context Adaptive Binary Arithmetic Coding*) no padrão de compressão de vídeo *Versatile Video Coding* (VVC/H.266). Diferentemente de seus predecessores como o HEVC, que utilizam tabelas de busca aproximadas, o padrão VVC exige o cálculo por multiplicação explícita para a atualização de intervalo tornando o estágio BAE um gargalo computacional crítico. Para otimizar a vazão de processamento, a arquitetura incorpora o esquema de múltiplos desvios MBBS (do inglês, *Multiple Bypass Bin Scheme*), o qual permite processar até dois *bins* do tipo *bypass* simultaneamente por ciclo. Uma importante conclusão extraída dos resultados é que o uso de MBBS para VVC BAE aumenta a produtividade de bins por ciclo em aproximadamente 7,608%, em média, em comparação com a versão base. O *hardware* foi implementado em VHDL sob uma estrutura de *pipeline* de 4 estágios e sintetizada para a tecnologia TSMC de 40 nm, a VArchBAE atinge uma frequência de operação de $1111MHz$, com vazão média de $1{,}195Mbins/s$ e consumo de potência de apenas $2{,}04^4mW$. Esses resultados atestam a viabilidade da arquitetura proposta para viabilizar o processamento em tempo real de fluxos de vídeo em resolução 8K a 120 quadros por segundo (*fps*), sendo a primeira solução de *hardware* para o BAE do padrão VVC reportada na literatura.

5.7 Análise Comparativa

A Tabela 2 apresenta um resumo comparativo das características dos trabalhos descritos acima. Ao analisá-la, é possível observar as diferentes abordagens adotadas por cada autor de acordo com o padrão de vídeo alvo e o foco da implementação.

Tabela 2 – Comparação entre trabalhos correlatos e a arquitetura proposta

Trabalho	**Formato de vídeo**	**Etapa da codificação**	**Vazão de Entrada**	**Nó tecnológico**	**Frequência**	**Área**	**Consumo**
(ALONSO *et al.*, 2017)	HEVC	Binarização	4 ES / Ciclo	65 nm	834 Mhz	11.85 K	1.87 mW
(TRAN *et al.*, 2020)	HEVC	Binarização + RSEG	3.5 ES / Ciclo	45 nm	500 Mhz	9.45 K	0.239 mW
(RAMOS *et al.*, 2019)	HEVC	RSEG	4 Coefs / Ciclo	65 nm	668 Mhz	3.67 K	8.46 mW
(GOMES *et al.*, 2023)	AV1	RSEG	1 Coef / Ciclo	40 nm	1.1 GHZ	6.93 K	11.52 mW
(LIU *et al.*, 2025)	Baseado no VVC	Binarização + RSEG + Estimador de taxa	12 Coefs / Ciclo	28 nm	500 Mhz	52.03 K	7.30 mW
(FARIA *et al.*, 2025)	VVC	Codificação Aritimética	1195 Mbins/s	40 nm	1.1 GHZ	5.15 K	$2.04^4 mW$
Transformada-RSE*	**VVC**	**RSEG**	1 Coef / Ciclo	**40 nm**	1 GHz	8.98 K	2.42 mW
Transformada-RSE*	**VVC**	**RSEG**	1 Coef / Ciclo	**40 nm**	1.5 GHz	10.80 K	4.60 mW
Transformskip-RSE*	**VVC**	**RSEG**	1 Resíduo / Ciclo	**40 nm**	1 GHz	2.82 K	0.81 mW
Transformskip-RSE*	**VVC**	**RSEG**	1 Resíduo / Ciclo	**40 nm**	1.5 GHz	4.27 K	1.51 mW

Fonte: O autor, adaptado de Cardoso *et al.* (2025).
Legenda: ES: Elementos sintáticos; Coef: Coeficientes; RSEG: Gerador de elementos sintáticos residuais; *:Este trabalho.

Os três primeiros trabalhos, de Alonso *et al.*, Ramos *et al.* e Tran *et al.*, concentram-se no padrão HEVC e abordam o problema da geração e binarização dos SEs residuais para o CABAC. Alonso *et al.* focam na redução do consumo de energia por meio de *Operand Isolation*, enquanto Ramos *et al.* priorizam o processamento paralelo de múltiplos resíduos para evitar a *starvation* do codificador de entropia, e Tran *et al.* buscam o melhor compromisso entre área e eficiência energética. Todos os três trabalhos utilizam análises estatísticas de sequências de vídeo reais para embasar suas decisões arquiteturais, uma metodologia que também é adotada neste trabalho de conclusão.

Gomes *et al.* trazem a geração de SEs residuais para o padrão AV1, que utiliza um sistema de codificação por mapa de níveis (*level map*) distinto do HEVC e do VVC. Sua arquitetura RSEG é a primeira da literatura dedicada a esse propósito no AV1 e demonstra a importância da análise estatística para guiar o projeto de *hardware*. O trabalho de Liu *et al.* aborda o padrão VVC, porém com foco na estimação de taxa para RDO, e não na geração direta dos SEs residuais para o codificador de entropia. Embora ambos lidem com o VVC, o escopo é distinto do proposto neste trabalho de conclusão.

Por sua vez, o trabalho de Faria et al. também direciona seus esforços para o padrão VVC, mas com foco específico na aceleração do Codificador Aritmético Binário (BAE) dentro do estágio CABAC. Através da arquitetura VArchBAE e da adoção do esquema MBBS, os autores buscam aumentar a vazão processando múltiplos *bins bypass*

simultaneamente. Faria et al. otimiza o motor aritmético da codificação de entropia com um ganho de desempenho obtido (cerca de 7,6% a mais de bins por ciclo).

Em suma, nota-se que a literatura existente tem focado no desenvolvimento de arquiteturas de geração de elementos sintáticos residuais para os padrões HEVC e AV1, enquanto para o VVC ainda há uma lacuna significativa. Embora Liu *et al.* abordem a codificação residual do VVC, seu foco é na estimação de taxa para a otimização de taxa-distorção, e não na geração dos SEs em si para o fluxo do CABAC. Portanto, o presente trabalho se posiciona visando preencher essa lacuna, propondo uma arquitetura dedicada à geração dos elementos sintáticos residuais do padrão VVC. A proposta arquitetural deste trabalho de conclusão utilizará como base os princípios de análise estatística, paralelismo de *hardware* e redução de acessos à memória discutidos por Ramos *et al.*, Tran *et al.* e Gomes *et al.*, adaptando-os às novas e mais complexas regras de particionamento, modelagem de contexto e codificação de coeficientes introduzidas pelo VVC.

6 ANÁLISE ESTATÍSTICA DOS ELEMENTOS RESIDUAIS

Em padrões anteriores de codificadores de video, como o HEVC (**??**) e o AV1 (GOMES *et al.*, 2023), foi confirmado que os RSEs (Elementos sintáticos residuais) são majoritários ou têm contribuição significativa para os dados de entrada da codificação de entropia. Assim, surgiu a necessidade de fazer uma análise estatística dos elementos sintáticos residuais no VVC. Com esses dados torna-se viável propor uma arquitetura de *hardware* dedicada para a geração dos RSEs, sendo eles gerados em *hardware* isso permitiria que os dados de entrada fossem suficientes para que CABAC não ficasse ocioso e, com isso, viabilizar uma possível capacidade em tempo real do fluxo de codificação para esse *codec*. Outro questionamento é qual o impacto no tempo total para os modos *Transform-based* e *Transform-skip*, pois cada um desses modos possui RSEs diferentes e consequentemente *hardwares* diferentes. Outra dúvida é a porcentagem de *bins* regulares e *bins bypass* nos dados de entrada do CABAC e, a partir desses dados, estudar a possibilidade de aumentar a vazão do *encoder* a partir da avaliação da quantidade de *bins* regulares e *bypass* que ocorrem sequencialmente.

A análise estatística foi realizada em 30 quadros de seis sequências de video de teste de acordo com a norma Boyce *et al.* (2018), utilizando o VVC *Test Model* (VTM) versão 19.2 (Fraunhofer-Gesellschaft, 2023), com resoluções desde SD até 4k, aplicado os parâmetros de quantização QP recomendados em (BOYCE *et al.*, 2018), 22, 27, 32 e 37. Com exceção para as sequências 4K, onde apenas os limites de QP recomendados inferior e superior foram aplicados. Além disso, dependendo da sequência de teste, a configuração *Low-Delay* (LD) ou *Random-Access* (RA) foi escolhida com base na recomendação do teste, ou seja, as sequências 4K Tango2 e CatRobot usaram RA, enquanto todas as outras usaram LD. O *software* VVC *Test Model* (VTM) versão 19.2 foi modificado, foram inseridos contadores no *software* e saídas em *logs* com as informações de interesse. Alguns vídeos com as menores resoluções (480p) levaram horas para serem codificados no VTM, enquanto as sequencias de 4K chegaram a levar 5 dias. Após os *logs* serem gerados eles foram transferidos para planilhas nas quais o cálculo dos percentuais foram efetuados e analisados, os resultados serão apresentados na Tabela 3 a seguir.

Tabela 3 – Análise Estatística dos Elementos Sintáticos

Sequências	QP	Elementos Sintáticos		Uso da Transformada		Tipo de bin		Ocorrência consecutiva do tipo de bin	
		Residuais	Não-Residuais	Tb	Ts	Regular	Bypass	Regular	Bypass
PartyScene (832x480)	22	78.08%	21.92%	76.31%	23.69%	80.76%	19.24%	18.82	4.48
	27	72.37%	27.63%	73.47%	26.53%	81.63%	18.37%	16.93	3.80
	32	65.99%	34.01%	72.27%	27.73%	81.88%	18.12%	15.36	3.40
	37	57.39%	42.61%	80.50%	19.50%	82.27%	17.73%	14.29	3.08
FourPeople (1280x720)	22	62.85%	37.15%	96.15%	3.85%	82.23%	17.77%	16.51	3.56
	27	59.26%	40.74%	97.38%	2.62%	82.18%	17.82%	15.73	3.41
	32	55.67%	44.33%	98.56%	1.44%	82.65%	17.35%	15.39	3.23
	37	50.58%	49.42%	99.03%	0.97%	82.93%	17.07%	14.95	3.08
BQTerrace (1920x1080)	22	78.08%	21.92%	76.31%	23.69%	80.76%	19.24%	18.82	4.48
	27	68.69%	31.31%	70.36%	29.64%	83.07%	16.93%	17.12	3.48
	32	64.06%	35.94%	76.39%	23.61%	83.04%	16.96%	15.40	3.14
	37	59.18%	40.82%	84.42%	15.58%	83.58%	16.42%	15.01	2.94
CatRobot (3840x2160)	22	65.24%	34.76%	94.48%	5.52%	81.60%	18.40%	16.18	3.64
	37	53.10%	46.90%	98.44%	1.56%	82.52%	17.48%	14.96	3.17
Tango2 (3840x2160)	22	58.47%	41.53%	98.85%	1.15%	82.38%	17.62%	16.92	3.62
	37	49.67%	50.33%	99.87%	0.13%	81.79%	18.21%	15.02	3.34
Média	-	**61.06%**	**38.94%**	**89.45%**	**10.55%**	**82.25%**	**17.75%**	**15.56**	**3.40**

Fonte: O autor, adaptado de Cardoso *et al.* (2025).

A Tabela 3 apresenta detalhadamente os resultados quantitativos da avaliação estatística para as cinco sequências de teste codificadas, discriminadas sob diferentes parâmetros de quantização (QP). Ao examinar as colunas correspondentes à proporção de Elementos Sintáticos, verifica-se claramente que os Elementos Sintáticos Residuais (RSEs) constituem a ampla maioria dos dados trafegados. Na sequência de menor resolução, *PartyScene* (832×480), por exemplo, a taxa de elementos residuais atinge o pico de 78,08% quando submetida a um QP de 22 (alta qualidade). Nota-se uma correlação direta com o nível de compressão: à medida que o valor do QP aumenta para 37, a presença de resíduos cai para 57,39%. Em sequências de altíssima resolução, como a *Tango2* (3840×2160), o comportamento se mantém, variando de 58,47% a 49,67%, o que comprova que, independentemente da resolução, os RSEs exigem uma carga massiva de processamento no *encoder*.

Em uma análise subsequente focada nas colunas de "Uso da Transformada", investigou-se a frequência com que o codificador recorre à Transformada convencional (*Transform-based - Tb*) em detrimento do modo *Transform-skip* (*Ts*). Os dados demonstram uma dominância absoluta do modo *Tb*, que apresenta uma média geral de 89,45% de utilização. Sequências como *FourPeople* e *Tango2* chegam a registrar mais de 98% e 99% de uso da transformada convencional em QPs mais altos (37). Por outro lado, sequências com texturas mais complexas, como *BQTerrace*, apresentam

uma taxa um pouco maior do modo *Ts* (atingindo até 29,64% em QP 27), mas ainda assim o *Tb* permanece majoritário. Esse cenário consolida a justificativa para focar o desenvolvimento arquitetural na otimização da geração de elementos oriundos da transformada convencional.

Avançando para a classificação dos dados de entrada do codificador de entropia (CABAC), a tabela detalha a ocorrência dos tipos de *bin*. O volume de *bins* regulares é expressivamente superior, dominando o fluxo de vídeo com uma média de 82,25% de todos os *bins* processados, enquanto os *bins bypass* respondem pelos 17,75% restantes. Contudo, é fundamental observar a dinâmica dos *bins bypass* sob variação do parâmetro de quantização: em todas as sequências analisadas, quando o QP diminui (ou seja, quando há menor perda de qualidade), a porcentagem de *bins bypass* gerados aumenta. Na sequência *PartyScene*, por exemplo, os *bins bypass* representam 19,24% com QP 22, sofrendo leve redução para 17,73% no QP 37.

Por fim, a última seção da tabela avalia a métrica crítica de "Ocorrência consecutiva", que mensura quantos *bins* do mesmo tipo são processados em sequência ininterrupta. Os dados revelam que os *bins* regulares são entregues em grandes agrupamentos ininterruptos, com picos de até 18,82 *bins* consecutivos (observados em *PartyScene* e *BQTerrace* com QP 22) e uma média geral consolidada de 15,56. Em contrapartida, os *bins bypass* apresentam sequências muito mais curtas, agrupando-se em média apenas 3,40 vezes consecutivas. Embora os agrupamentos de *bypass* sejam menores, a constatação estatística de que eles ocorrem em blocos sequenciais (chegando a 4,48 em certos cenários) é reveladora. Esse comportamento sequencial demonstra que existe um claro e viável potencial para se projetar um *hardware* dedicado ao CABAC do VVC capaz de processar múltiplos *bins bypass* em um único ciclo de *clock*, abrindo caminho para ganhos significativos no rendimento global da codificação.

7 ARQUITETURAS PROPOSTAS PARA GERAÇÃO DE RSES NO VVC

Conforme discutido anteriormente, uma parcela significativa dos dados fornecidos para a codificação de entropia é composta pelos elementos sintáticos residuais. Portanto, acelerar o processamento dos dados residuais é essencial para evitar que o CABAC não fique ocioso pela falta de dados.

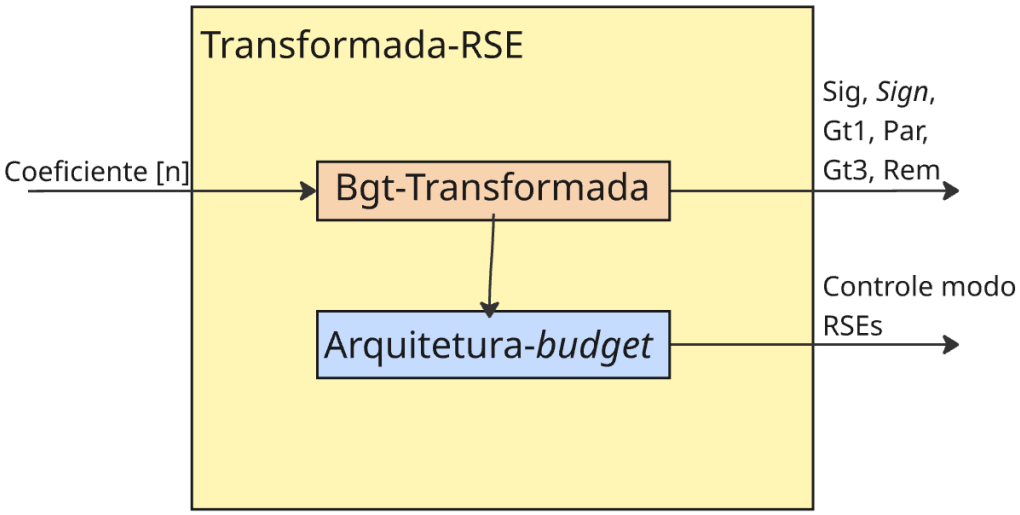
Neste capítulo, são propostas duas arquiteturas para a geração dos elementos sintáticos residuais do VVC, tanto para o modo *Transform-based* quanto para o modo *Transform-skip*. A proposta fundamenta-se na análise estatística deste trabalho, que demonstra a importância dos RSEs para a codificação VVC. Como solução, são propostos aceleradores em *hardware* compatíveis com o padrão VVC, capazes de gerar RSEs em ambos os modos e fornecer dados suficientes para maximizar a utilização do CABAC. A implementação dos aceleradores se deu por dois objetivos:

- Compatibilidade total com o padrão VVC;
- Acesso único à memória, onde os coeficientes/resíduos são lidos apenas uma vez para produzir todos os RSEs relacionados;

7.1 Arquitetura para o Modo *Transform-based*

O *design* Transformada-RSE, foi implementado para o processamento de coeficientes para o modo *Transform-based* e atua na geração de RSEs dentro do limite do orçamento de *bins* regulares, conforme Fraunhofer-Gesellschaft (2023), sendo composta por dois módulos: módulo Bgt-Transformada e pelo módulo *Arquitetura-Budget*. Foi feita uma abstração e um diagrama geral desta arquitetura pode ser visualizado na Figura 21.

Figura 21 – Diagrama Arquitetura para o Modo *Transform-based*



Fonte: O autor

7.1.1 Arquitetura Transformada-RSE (Bgt-Transformada)

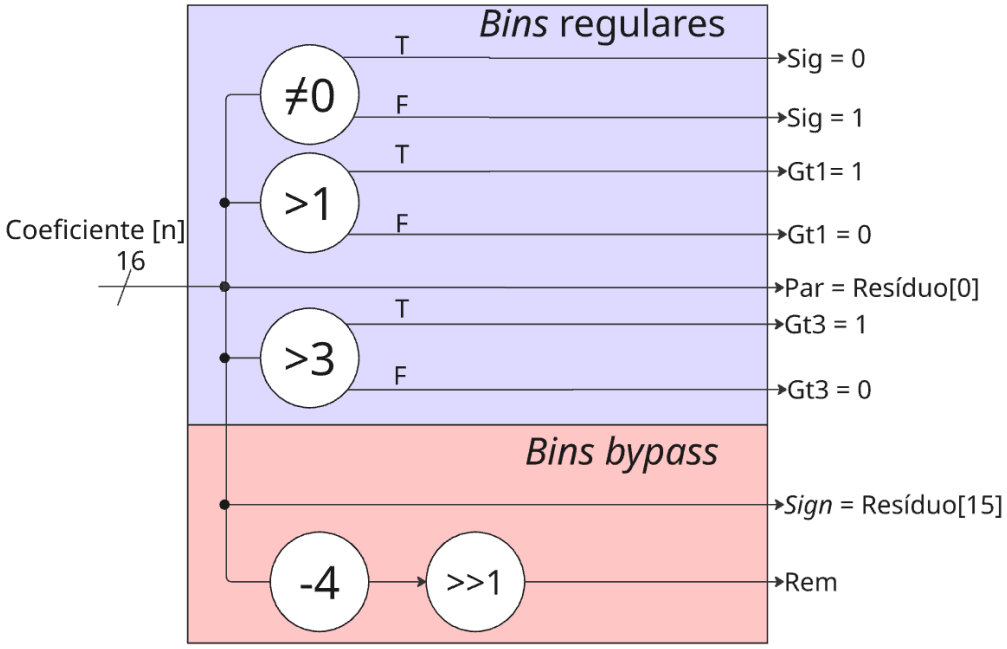
A estrutura do módulo Bgt-Transformada é responsável por receber um coeficiente e gerar, em paralelo, todos os elementos sintáticos residuais para esse coeficiente. É assumido que o elemento *'last'* já foi previamente processado pelas etapas anteriores, sendo esse e o elemento dec os RSEs que essa arquitetura não gera.

A lógica para cada RSE é explicada a seguir. Na Figura 22 é possível ver uma abstração desse módulo:

- **sig:** Uma simples comparação do coeficiente com zero. Esse sinal é a validade, ou seja, pré-requisito para os RSEs 'gt1' e *'sign'*.
- **gt1:** É gerado a partir do módulo do coeficiente; o valor é comparado a uma constante de valor um. Se for maior, a *flag* gt1 recebe o valor 1. Este RSE funciona como validação para o processamento de 'par' e 'gt3'.
- **par:** Corresponde diretamente ao *bit* menos significativo do coeficiente lido.
- **gt3:** É gerado a partir do módulo do coeficiente; o valor é comparado a uma constante de valor três. Se for maior, a *flag* gt3 recebe o valor 1. Este RSE funciona como validação para o processamento de 'rem'.
- **sign:** Indica se o coeficiente tem valor positivo ou negativo, o que pode ser gerado diretamente pelo *bit* mais significativo, pois o coeficiente está em complemento de 2.
- **rem:** É gerado a partir de uma subtração do coeficiente absoluto por quatro, seguida

de um deslocamento lógico à direita *(shift-right)* em uma posição.

Figura 22 – Módulo Bgt-Transformada



Fonte: O autor

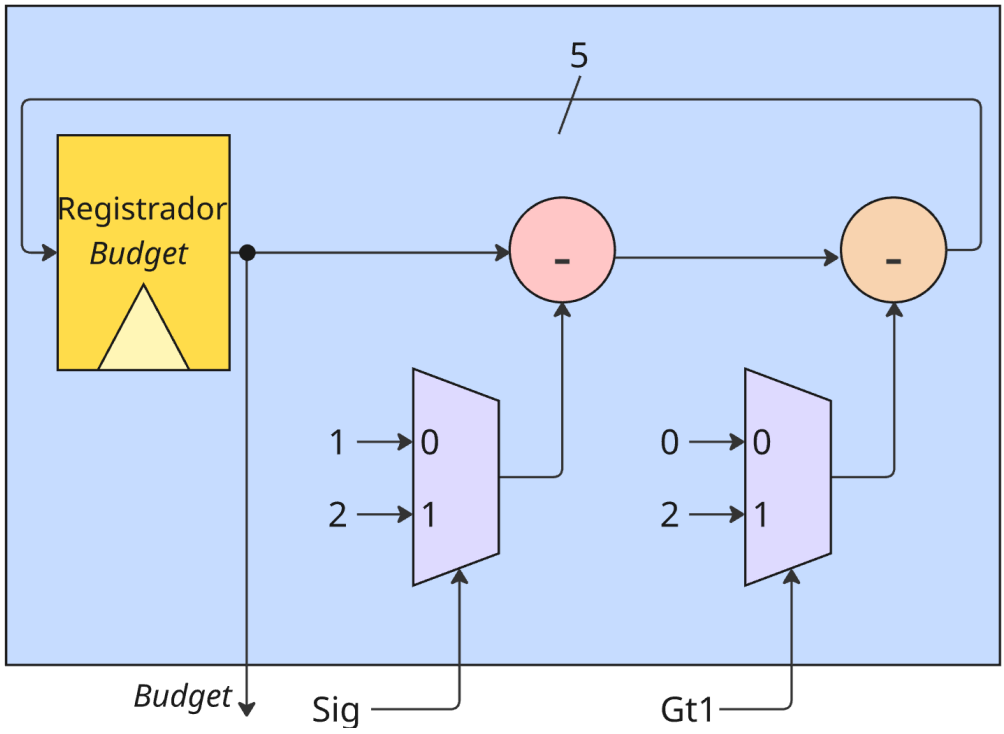
7.1.2 Unidade de Controle do *budget* (Arquitetura-*budget*)

A seguir, será explicado o módulo responsável pelo controle do *budget*, que é composto por dois subtratores, dois multiplexadores e um registrador que é inicializado com o valor padrão do VVC (28 *bins* regulares) conforme Fraunhofer-Gesellschaft (2023). Este valor é redefinido toda vez que um novo *Coding Group* (CG) de uma *Transform Unit* (TU) é acessado.

O decréscimo deste registrador depende da ocorrência dos elementos sintáticos residuais regulares recém-gerados (‘sig‘, ‘gt1‘, ‘par‘ e ‘gt3‘). A lógica segue as regras ilustradas na Figura 23: se o RSE ‘sig‘ tiver o valor 1, o registrador *budget* sofre um decréscimo inicial pelo valor 2 (pois pelo menos o ‘sig‘ e o ‘gt1‘ existirão para o coeficiente, ou seja, foram gerados dois *bins* regulares); caso contrário, o decréscimo é apenas de um (apenas sig foi gerado , ou seja, apenas um bin regular foi gerado, logo é necessário subtrair apenas o valor um do valor armazenado no registrador do *budget)*. Logo após essa primeira subtração, verifica-se o ‘gt1‘, se esse RSE tiver valor 1, um segundo decréscimo de valor 2 ocorre, pois os RSEs ‘par‘ e ‘gt3‘ também foram gerados (em outras palavras mais dois *bins* regulares foram gerados e isso deve ser contabilizado

no registrador do *budget*). Logo, esse módulo é composto por um par de subtratores com decréscimos selecionados por meio de multiplexadores controlados pelos valores dos RSE de ‘sig‘ e ‘gt1‘ que são utilizados como seletores dos multiplexadores.

Figura 23 – Módulo Arquitetura-*budget*



Fonte: O autor

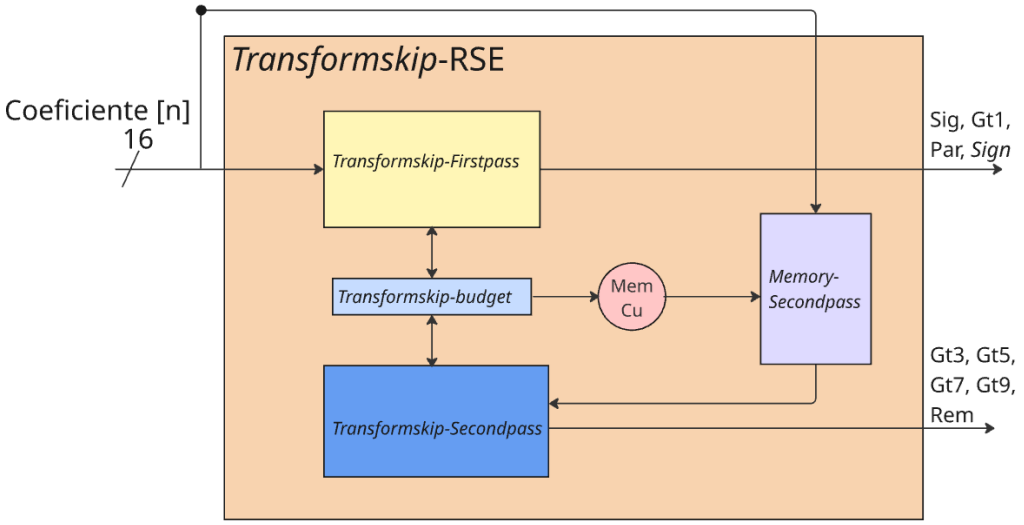
7.2 Arquitetura para o Modo *Transform-skip* (*Transformskip*-RSE)

O modo *Transform-skip* apresenta regras de geração de RSEs consideravelmente diferentes do modo *Transform-based*, exigindo até duas etapas (*first-pass* e *second-pass*) pelos coeficientes dentro de um CG, e não faz uso do elemento ‘*last*‘. A arquitetura, está abstraída em módulos e ilustrada na Figura 24. A arquitetura encarregada do modo *Transformskip* é denominada *Transformskip-RSE*, ela reflete essas diferenças e é composta por dois módulos fundamentais: *Transformskip-Firstpass* e *Transformskip-Secondpass*, conectadas por uma unidade lógica unificada *Transformskip-Budget* que determina se há *budget* suficiente para a segunda etapa (*second-pass*).

Caso haja *budget* o processamento pode seguir para uma segunda etapa de geração dos RSEs, o *design* foi implementado em vhdl e contém *Transformskip-RSE* e dentro dela estão a *Transformskip-Firstpass*, *Transformskip-Secondpass*, *Transformskip-Budget*, uma

interface controladora de Memória (*Mem CU*) responsável pela reiteração de acessos dos resíduos processados na primeira etapa (composta por dois contadores: o contador de escrita e o contador de leitura) e uma memória denominada *memory-secondpass*.

Figura 24 – Arquitetura *Transformskip*-RSE



Fonte: O autor

7.2.1 Módulo de Primeira Etapa (*Transformskip-Firstpass*)

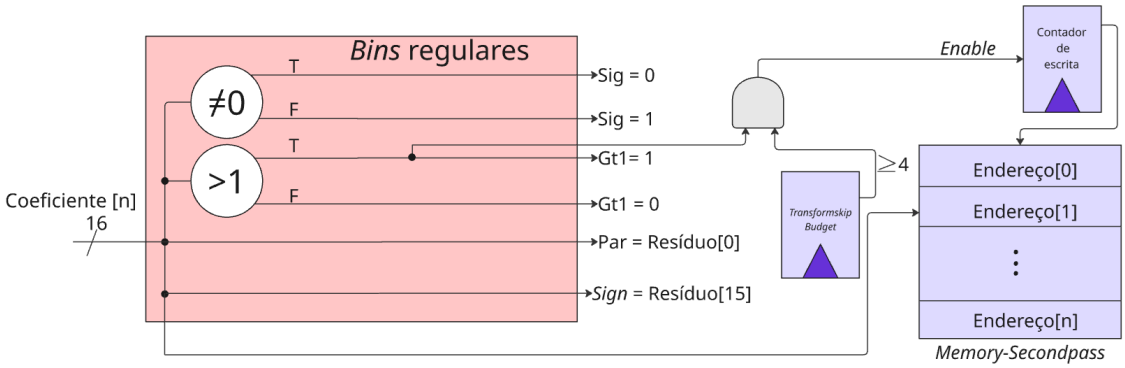
O módulo *Transformskip-Firstpass* processa os primeiros elementos sintáticos residuais que todo coeficiente apresentará os *bins* regulares (dentre estes apenas o rem não é um *bin* regular) sendo eles gerados de forma semelhante ao modo *Transform-based*:

- **sig:** Uma simples comparação do coeficiente com uma constante de valor zero. Esse sinal é a validade, ou seja, um pré-requisito para os elementos sintáticos residuais ‘gt1‘ e ‘*sign*‘.
- **gt1:** É gerado a partir do módulo do coeficiente; o valor é comparado a uma constante de valor um. Se for maior, o RSE gt1 recebe o valor 1. Este RSE funciona como validação, ou seja, é um pré-requisito para o processamento de ‘par‘ e ‘gt3‘.
- **par:** Corresponde diretamente ao *bit* menos significativo do coeficiente lido.
- **sign:** Indica se o coeficiente tem valor positivo ou negativo, o que pode ser gerado diretamente pelo *bit* mais significativo, pois o coeficiente está em complemento de 2.

Para o modo *Transformskip*, é importante destacar que o ‘*sign*‘ passa a pertencer à categoria de codificação regular, ou seja, ele é categorizado como um *bin* regular

e ao ser gerado, leva a um decremento no registrador *budget* diferente do modo *Transform-based*. O circuito é implementado e é similar à Transformada-RSE, usando lógicas combinacionais para gerar os elementos sintáticos residuais, que foram explicados acima. Uma adição substancial do *Transformskip-RSE* é que todos os coeficientes onde o RSE ‘gt1‘ é gerado com o valor 1 e há um valor maior ou igual a quatro no registrador *budget* precisam ter seus valores preservados em uma memória para a segunda etapa (*second-pass*). A Unidade *Mem CU* escreve tais coeficientes em uma memória de armazenamento intermediário (baseada em registradores), iterando um contador denominado contador de escrita para controlar até onde a memória foi preenchida com coeficientes, que terão RSEs gerados na segunda etapa. A Figura 25 apresenta uma abstração do módulo da primeira etapa do modo *Transformskip*.

Figura 25 – Módulo *Transformskip-Firstpass*



Fonte: O autor

7.2.2 Lógica de Controle do *Budget* (*Transformskip-Budget*)

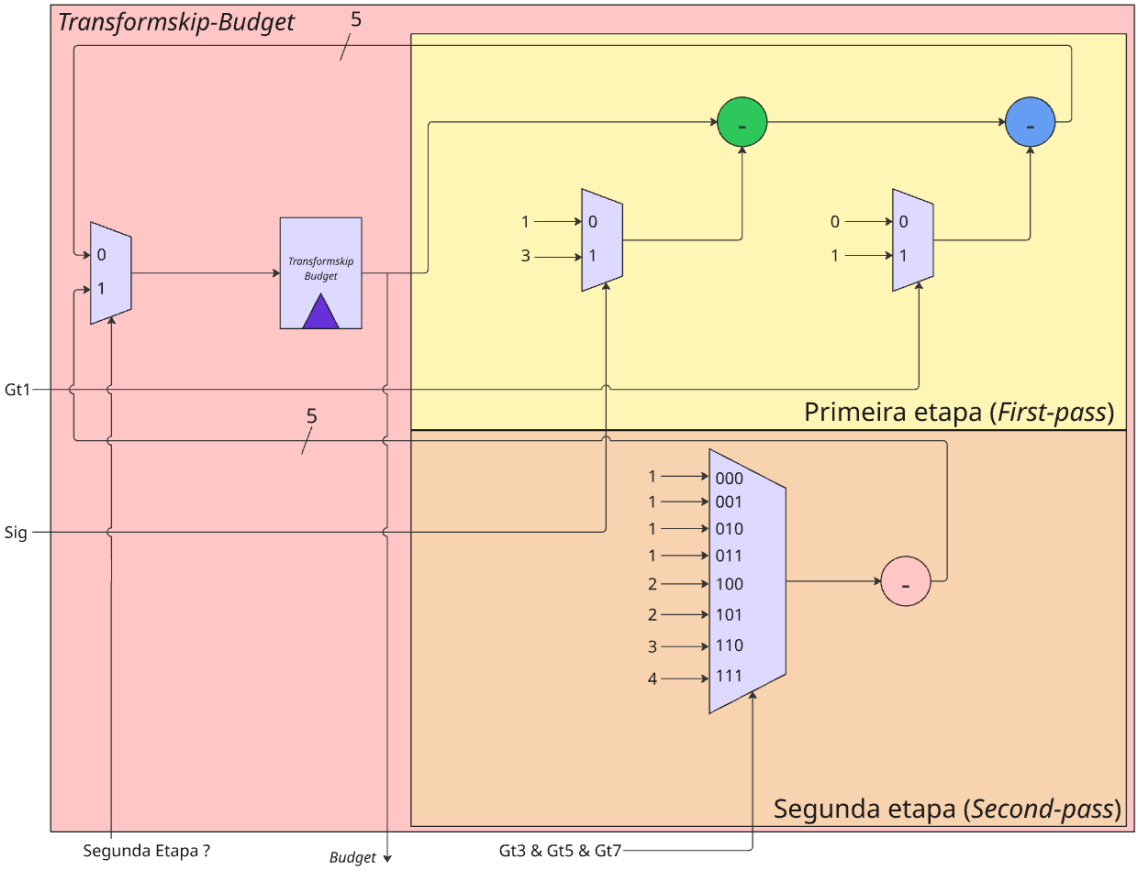
O monitoramento (controle) do *budget* (*Transformskip-Budget*) para o modo *Transformkskip* é mais elaborado, subdividindo suas deduções para decremento a partir das duas etapas presentes na arquitetura (Figura 26).

Durante a primeira etapa, se o RSE ‘sig‘ for gerado e tiver o valor 1, a arquitetura imediatamente decresce o valor do orçamento em três (levando em conta que, sendo o coeficiente significativo serão gerados os RSE: ‘*sign*‘ e ‘gt1‘), e caso contrário, em um pois apenas o ’sig’ será gerado com o valor 0. Um novo decréscimo unitário subsequente ocorrerá caso o coeficiente seja maior ou igual a um em módulo, ou seja, o RSE ‘gt1‘ recebe o valor 1 (isso implica na geração futura do RSE ‘par‘). É possível observar essa parte da arquitetura na região amarela da Figura 26).

Após realizar a primeira etapa, existindo coeficientes maiores ou iguais a um e havendo orçamento disponível no *budget* após o término da primeira etapa, a arquitetura inicia sua segunda etapa. Aqui, as instâncias de decremento assumem outras regras, sendo subtraído um determinado valor do registrador de acordo com as ocorrências dos RSEs combinados: ‘gt3‘, ‘gt5‘, e ‘gt7‘. O seletor deste multiplexador subtrai quantidades incrementais de 1 a 4 guiado por uma lógica de concatenação desses RSEs (é possível observar essa parte da arquitetura na região laranja da Figura 26):

- É repassado o valor 1 ao subtrator se gt3 for zero, ou seja, apenas o gt3 foi gerado, o multiplexador fornece ao subtrator a constante de valor 1 da posição zero do multiplexador e é selecionada quando a concatenação de gt3, gt5 e gt7 é "000".
- É repassado o valor 2 ao subtrator se gt3 for 1 e gt5 for 0, ou seja, apenas gt3 e gt5 foram gerados, o multiplexador fornece ao subtrator a constante de valor 2 da posição cinco do multiplexador e é selecionada quando a concatenação de gt3, gt5 e gt7 é "100".
- É repassado o valor 3 ao subtrator se gt3 for 1, gt5 for 1 e gt7 for 0, ou seja, gt3, gt5 e gt7 foram gerados, o multiplexador fornece ao subtrator a constante de valor 3 da posição cinco do multiplexador e é selecionada quando a concatenação de gt3, gt5 e gt7 é "110".
- É repassado o valor 4 ao subtrator se gt3 for 1, gt5 for 1 e gt7 for 1, ou seja, os RSEs: gt3, gt5, gt7 e gt9; foram gerados, o multiplexador fornece ao subtrator a constante de valor 4 da posição oito do multiplexador e é selecionada quando a concatenação de gt3, gt5 e gt7 é "111".
- No multiplexador também foram levadas em conta possíveis anomalias; por exemplo , não é possível existir o gt7 se o valor de gt3 for 0 e gt5 for 0, dentre outros casos semelhantes, evitando assim falhas de decréscimo no registrador do *budget*. Esse caso corresponde a posição dois do multiplexador, selecionada quando a concatenação de gt3, gt5 e gt7 é "001".

Figura 26 – Módulo *Transformskip-Budget*



Fonte: O autor

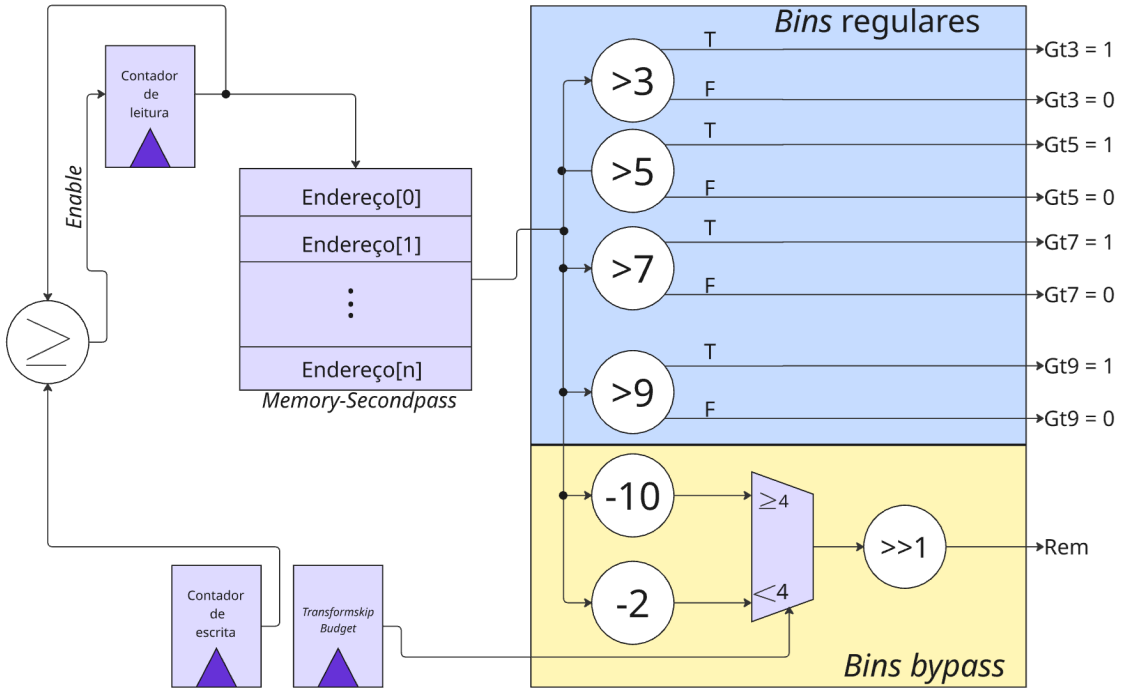
7.2.3 Módulo da Segunda Etapa (Transformskip-Secondpass)

O módulo *Transformskip-Secondpass* processa os coeficientes de valor em módulo superior a 1, que foram armazenados anteriormente na memória *Memory-Secondpass*. A leitura dos endereços dessa memória é realizada por meio de dois contadores: o contador de escrita e o contador de leitura. O contador de leitura é incrementado sucessivamente até igualar o valor armazenado no contador de escrita. Caso haja *budget* disponível, podem ser gerados os seguintes RSES:

- **gt3:** É gerado a partir do módulo do coeficiente; o valor é comparado a uma constante de valor 3. Se for maior, a *flag* gt3 recebe o valor 1. Este RSE funciona como validação para o processamento de gt5.
- **gt5:** É gerado a partir do módulo do coeficiente; o valor é comparado a uma constante de valor 5. Se for maior, a *flag* gt5 recebe o valor 1. Este RSE funciona como validação para o processamento de gt7.

- **gt7:** É gerado a partir do módulo do coeficiente; o valor é comparado a uma constante de valor 7. Se for maior, a *flag* gt7 recebe o valor 1. Este RSE funciona como validação para o processamento de gt9.
- **gt9:** É gerado a partir do módulo do coeficiente; o valor é comparado a uma constante de valor 9. Se for maior, a *flag* gt9 recebe o valor 1.
- **rem:** Tem duas possíveis regras caso a condição de que o *budget* não pode ter estourado, ou seja, ter o valor maior que 4 é necessário para que o RSE "rem" ser gerado, caso contrário não será gerado:
  - É gerado a partir de uma subtração do coeficiente absoluto por quatro, seguida de um deslocamento lógico à direita (*shift-right*) em uma posição quando o coeficiente for maior em módulo que um, ou seja, *gt1* tiver o valor 1.
  - É gerado a partir de uma subtração do coeficiente absoluto por dez, seguida de um deslocamento lógico à direita (*shift-right*) em uma posição quando o coeficiente for maior em módulo que nove, ou seja, *gt9* tiver o valor 1.

Figura 27 – Módulo *Transformskip-Secondpass*



Fonte: O autor

7.3 Validação funcional

Para garantir o funcionamento correto e a total conformidade com o padrão VVC, as arquiteturas implementadas (Transformada-RSE e *Transformskip-RSE*) foram submetidas a um processo de validação funcional em nível RTL (*Register Transfer Level*). Essa etapa foi conduzida utilizando a linguagem de descrição de *hardware* VHDL, por meio do desenvolvimento de *testbenchs* específicos e simulações realizadas no *software* ModelSim. A metodologia de verificação adotou o *software* de referência do VVC (VTM) como modelo de referência, foram utilizadas determinadas entradas no *testbench* para testar se as saídas estavam em conformidade com o *software*.

7.4 Resultados de Síntese e Discussões

Para validar a viabilidade de hardware das estruturas propostas, as arquiteturas *Tb-RSE-arch* e *Ts-RSE-arch* foram descritas em VHDL e sintetizadas utilizando o PDK TSMC 40nm com a ferramenta *Cadence® Genus™*. A análise focou no pior cenário *(worst-case corner)*: 0,81V de tensão de alimentação e 125°C de temperatura, estipulando metas flexíveis de frequência operacional em 1 GHz e 1,5 GHz.

Tabela 4 – Comparação entre trabalhos correlatos e a arquitetura proposta

Trabalho	**Formato de vídeo**	**Etapa da codificação**	**Vazão de Entrada**	**Nó tecnológico**	**Frequência**	**Área**	**Consumo**
(ALONSO *et al.*, 2017)	HEVC	Binarização	4 ES / Ciclo	65 nm	834 Mhz	11.85 K	1.87 mW
(TRAN *et al.*, 2020)	HEVC	Binarização + RSEG	3.5 ES / Ciclo	45 nm	500 Mhz	9.45 K	0.239 mW
(RAMOS *et al.*, 2019)	HEVC	RSEG	4 Coefs / Ciclo	65 nm	668 Mhz	3.67 K	8.46 mW
(GOMES *et al.*, 2023)	AV1	RSEG	1 Coef / Ciclo	40 nm	1.1 GHZ	6.93 K	11.52 mW
(LIU *et al.*, 2025)	Baseado no VVC	Binarização + RSEG + Estimador de taxa	12 Coefs / Ciclo	28 nm	500 Mhz	52.03 K	7.30 mW
(FARIA *et al.*, 2025)	VVC	Codificação Aritimética	1195 Mbins/s	40 nm	1.1 GHZ	5.15 K	$2.04^4 mW$
Transformada-RSE*	**VVC**	**RSEG**	1 Coef / Ciclo	**40 nm**	1 GHz	8.98 K	2.42 mW
Transformada-RSE*	**VVC**	**RSEG**	1 Coef / Ciclo	**40 nm**	1.5 GHz	10.80 K	4.60 mW
Transformskip-RSE*	**VVC**	**RSEG**	1 Resíduo / Ciclo	**40 nm**	1 GHz	2.82 K	0.81 mW
Transformskip-RSE*	**VVC**	**RSEG**	1 Resíduo / Ciclo	**40 nm**	1.5 GHz	4.27 K	1.51 mW

Fonte: O autor, adaptado de Cardoso *et al.* (2025).

Legenda: ES: Elementos sintáticos; Coef: Coeficientes; RSEG: Gerador de elementos sintáticos residuais; * : Este trabalho.

Analisando os trabalhos correlatos, a arquitetura que mais se aproxima da proposta neste projeto é a de Liu *et al.* (2025) por ser para o mesmo padrão de codificação de vídeo. Não é possível fazer uma comparação direta com os outros por serem baseados em outros *codecs*. Conforme a Tabela 4, a taxa de coeficientes por ciclo de Liu *et al.* (2025) é superior à obtida neste trabalho. No entanto, esse ganho de desempenho resulta em um custo muito maior em área e consumo de energia. O trabalho de Faria *et al.* (2025) conclui que o uso de MBBS (do inglês, *Multiple Bypass Bin Scheme*) para VVC BAE (do inglês, *Binary Arithmetic Encoder*) aumenta a produção de *bins* por ciclo em aproximadamente apenas 7,6%. Logo, esse pequeno ganho de vazão não justifica o aumento drástico nos custos de *hardware*.

Ao comparar diretamente com Liu et al. (2024) que também foca no padrão VVC, a vantagem do presente trabalho em área é promissora. Somando a ocupação dos dois módulos (*Transformada-RSE* e *Transformskip-RSE*), a versão de 1 GHz ocupa 11,8 *Kgates*. Isso representa uma economia de área de 77,3% em relação ao *design* de Liu et al. (52,03 *Kgates*), que é 4,4 vezes maior. Mesmo na versão de 15,5 *Kgates*, a redução de área chega a 70,2%, tornando a arquitetura de Liu 3,3 vezes maior que a proposta neste trabalho.

Para avaliar o consumo de potência, considerou-se o cenário de operação conjunta, simulando um sistema RSEG completo capaz de processar blocos com *Transform-based* e no modo *Transform Skip*. Operando as duas arquiteturas simultaneamente a 1 GHz, o consumo total é de 3,23 mW (sendo 2,42 mW da *Transformada-RSE* e 0,81 mW da *Transformskip-RSE*). No cenário de alto desempenho (1,5 GHz), o consumo conjunto atinge 6,11 mW.

Os ganhos energéticos desta solução, por focar estritamente na geração dos RSEG, destacam-se frente a Liu *et al.* (2025), que dissipa 7,30 mW a apenas 500 MHz. Isso significa que, operando o sistema completo no dobro da frequência (1 GHz), o consumo deste trabalho ainda representa menos da metade da potência dissipada pela solução de Liu. Mesmo rodando ao triplo da velocidade (1,5 GHz), o consumo total de 6,11 mW continua inferior ao Liu *et al.* (2025). Além das vantagens em área e consumo energético, existem duas distinções técnicas importantes: a arquitetura desenvolvida neste trabalho suporta a geração de RSEs para os modos *Transform-based* e *Transform Skip*, enquanto Liu *et al.* (2025) atende apenas ao modo *Transform-based*. Além disso, Liu *et al.* (2025) apresenta uma solução apenas baseada no VVC, ao passo que o presente trabalho é inteiramente compatível com as normas do padrão VVC.

8 CONSIDERAÇÕES FINAIS

O crescente consumo de vídeos em altas resoluções exige o uso de padrões de codificação mais eficientes, como o VVC, que, apesar de oferecer ganhos expressivos de compressão, apresenta um aumento considerável na complexidade computacional. Este trabalho teve como objetivo central investigar a viabilidade e o impacto do processamento dos Elementos Sintáticos Residuais (RSEs) nesse *codec*, propondo arquiteturas em *hardware* dedicadas para acelerar essa etapa e evitar a ociosidade (*starvation*) do codificador de entropia CABAC.

A análise estatística, realizada por meio do VTM, confirmou a hipótese de que os RSEs constituem a maior parcela dos dados processados no fluxo de entrada da codificação de entropia. Observou-se que o modo Transformada é amplamente predominante, representando, em média, 89,45% do uso. No entanto, o modo *Transform-skip* demonstrou relevância estatística em sequências de vídeo com texturas mais complexas, alcançando até 29,64% de utilização, o que justificou a necessidade de desenvolver *hardware* dedicado para ambos os modos.

A partir desses dados, é válido implementação de duas arquiteturas em *hardware* (Transformada-RSE e *Transformskip-RSE*). Ambas foram projetadas para garantir acesso único à memória, lendo os resíduos apenas uma vez para gerar todos os RSEs correspondentes(com exeção do elemento sintático residual Dec). Os *hardwares* foram descritos com total compatibilidade com o padrão VVC.

Os resultados de síntese, baseada no nó tecnológico TSMC de 40 nm, operando a uma frequência de 1 GHz, o sistema completo consumiu apenas 3,23 mW de potência e ocupou uma área de 11,8 *Kgates* e a de 1.5 Ghz ocupou uma área de 15,5 *Kgates* e consumiu 6,11 mW de potência. Em comparação com o trabalho encontrado mais semelhante na literatura voltada ao VVC, a arquitetura deste trabalho destacou-se da seguinte forma: obteve uma economia de área de 70,2% até 77,3% e dissipou menos da metade da potência da solução concorrente na frequência de 1 Ghz e mesmo no cenário de maior desempenho com 1.5 Ghz ainda dissipava menos potência que o trabalho de (LIU *et al.*, 2025), operando com o triplo da frequência.

Como trabalhos futuros, propõe-se o desenvolvimento de uma arquitetura capaz de gerar totalmente os elementos sintáticos residuais tanto para o modo de Transformada quanto para o modo *Transformskip*. Além disso, a ampliação da análise estatística atual por meio da inclusão de um número maior de sequências de teste, com mais valores de

QP e mais quadros (*frames*) por sequência codificados, visando consolidar e ampliar os resultados obtidos.

8.1 Publicações pelo autor

CARDOSO, G. B.; TOMM, D. F.; GOMES, J. S.; WUERDIG, R. N.; BAMPI, S.; RAMOS, F. L. L. *Statistical Analysis of VVC Residual and Entropy Coding aiming Efficient Hardware Design.* In: ***2024 IEEE 15th Latin America Symposium on Circuits and Systems (LASCAS)***. IEEE, 2024. p. 1-5. DOI: <https://doi.org/10.1109/LASCAS60203.2024.10506158>.

CARDOSO, G. B.; HUARACHI, H. M. C.; TOMM, D. F.; GOMES, J. S.; WÜRDIG, R. N.; DE MORAES, M. R.; BAMPI, S.; RAMOS, F. L. L. *Hardware accelerator for VVC/H.266 residual syntax elements: a unified and resource-efficient architecture.* ***Journal of Real-Time Image Processing***, v. 22, n. 4, p. 151, 2025. DOI: <https://doi.org/10.1007/s11554-025-01733-8>.

CARDOSO, G. B.; GOMES, J. S.; ROBAINA, R. P.; BAMPI, S.; RAMOS, F. L. L. *NNCodec Neural Network Compression Assessment and Binarization Step Hardware Design.* In: **2025 38th SBC/SBMicro/*IEEE Symposium on Integrated Circuits and Systems Design* (SBCCI)**. IEEE, 2025. p. 1-5. DOI: <https://doi.org/10.1109/SBCCI66862.2025.11218673>.

REFERÊNCIAS

Alliance for Open Media. **AV1 Video Codec — Specifications**. 2025. Disponível em: https://aomedia.org/specifications/av1/.

ALONSO, C. de M. *et al.* Low-power hevc binarizer architecture for the cabac block targeting uhd video processing. In: **Proceedings of the 30th Symposium on Integrated Circuits and Systems Design: Chip on the Sands**. [S.l.: s.n.], 2017. p. 30–35.

BITENCOURT, T. P. **Architecture exploration and VLSI design of multi-symbol arithmetic encoders for the AV1 coding format**. Dissertação (Dissertação de Mestrado) — Universidade Federal do Rio Grande do Sul (UFRGS), Programa de Pós-Graduação em Microeletrônica (PGMICRO), Porto Alegre, RS, 2023. Disponível em: http://hdl.handle.net/10183/262529.

BOYCE, J. *et al.* **JVET-J1010: JVET common test conditions and software reference configurations**. [S.l.], 2018.

BUBOLZ, T. L. A. **Aceleração do particionamento de quadros intra no codificador Versatile Video Coding (VVC) utilizando redes neurais profundas**. Dissertação (Dissertação de Mestrado) — Universidade Federal de Pelotas (UFPel), Programa de Pós-Graduação em Computação (PPGC), Pelotas, RS, 2021. Disponível em: http://guaiaca.ufpel.edu.br/handle/prefix/8073.

Canon Professional eXchange. **EOS Movie Compression Options: All-I and IPB**. 2025. Disponível em: https://www.canon.com.hk/cpx/en/technical/va_EOS_Movie_Compression_Options_All_I_and_IPB.html. Acesso em: 28 nov. 2025.

CARDOSO, G. B. *et al.* Hardware accelerator for vvc/h. 266 residual syntax elements: a unified and resource-efficient architecture. **Journal of Real-Time Image Processing**, Springer, v. 22, n. 4, p. 151, 2025.

CETINKAYA, E. **WaveOne Research Proposes A New Machine Learning-Based Algorithm For Video Coding, Learned End-To-End For The Low-Latency Mode**. 2022. MarkTechPost. Disponível em: https://www.marktechpost.com/2022/09/12/waveone- research- proposes- a- new- machine- learning- based- algorithm- for- video-coding- learned- end- to- end- for- the- low- latency- mode/. Acesso em: 30 nov. 2025.

Ericsson. **Ericsson Mobility Report**. [S.l.], 2022. Disponível em: https://www.ericsson.com/en/reports-and-papers/mobility-report.

FARIA, V. M. *et al.* High-performance binary arithmetic encoder with multiple bypass bin scheme for vvc cabac. In: **2025 IEEE 16th Latin America Symposium on Circuits and Systems (LASCAS)**. [S.l.: s.n.], 2025. v. 1, p. 1–5.

Fraunhofer-Gesellschaft. **VVC Reference Software: VTM-19.2**. 2023. Disponível em: https://vcgit.hhi.fraunhofer.de/jvet/VVCSoftware_VTM/-/tags/VTM-19.2.

FREITAS, C. C. P. E. C. de. **Metodologia do Trabalho Científico: Métodos e Técnicas da Pesquisa e do Trabalho Acadêmico**. Editora Feevale, 2013. ISBN 9788577171583. Disponível em: https://books.google.com.br/books?id=zUDsAQAAQBAJ.

GAO, W.; MA, S. Transform and quantization. In: **Advanced Video Coding Systems**. Cham: Springer International Publishing, 2014. p. 79–93. ISBN 978-3-319-14243-2. Disponível em: https://doi.org/10.1007/978-3-319-14243-2_5.

GOMES, J. S. *et al.* End-to-end neural video compression: A review. **IEEE Open Journal of Circuits and Systems**, v. 6, p. 120–134, 2025.

GOMES, J. S. *et al.* Av1 residual syntax elements assessment and efficient vlsi architecture. In: IEEE. **2023 36th SBC/SBMicro/IEEE/ACM Symposium on Integrated Circuits and Systems Design (SBCCI)**. [S.l.], 2023. p. 1–6.

HitPaw. **O que é taxa de quadros e como alterá-la com passos fáceis**. 2025. Disponível em: https://www.hitpaw.com.br/video-tips/what-is-frame-rate-and-how-to-change-it.html.

ITU-T; ISO/IEC. **High Efficiency Video Coding, ITU-T Recommendation H.265 and ISO/IEC 23008-2**. [S.l.], 2013. Disponível em: https://www.itu.int/rec/T-REC-H.265-201304-S/en.

ITU-T; ISO/IEC. **Infrastructure of audiovisual services – Coding of moving video (H.266)**. [S.l.], 2020.

LIU, C. *et al.* Hardware implementation of a high-accuracy and high-throughput rate estimation unit for vvc residual coding. **IEEE Transactions on Circuits and Systems for Video Technology**, v. 35, n. 3, p. 2832–2843, 2025.

MANOEL, E. T. M. **Codificação de Vídeo H.264 – Estudo de codificação mista de macroblocos**. Dissertação (Dissertação (Mestrado em Engenharia Elétrica)) — Universidade Federal de Santa Catarina, Florianópolis, 2007.

MARPE, D.; SCHWARZ, H.; WIEGAND, T. Context-based adaptive binary arithmetic coding in the h.264/avc video compression standard. **IEEE Transactions on Circuits and Systems for Video Technology**, v. 13, n. 7, p. 620–636, 2003.

NVIDIA. **GeForce Broadcasting – Sua melhor solução para livestreaming**. 2025. Disponível em: https://www.nvidia.com/pt-br/geforce/broadcasting/.

NVIDIA Developer. **Video Codec SDK**. 2025. NVIDIA Developer. Disponível em: https://developer.nvidia.com/video-codec-sdk.

RAMOS, F. L. L. *et al.* Residual syntax elements analysis and design targeting high-throughput hevc cabac. **IEEE Transactions on Circuits and Systems I: Regular Papers**, IEEE, v. 67, n. 2, p. 475–488, 2019.

RICHARDSON, I. E. **The H.264 Advanced Video Compression Standard**. 2. ed. Chichester: Wiley, 2010.

RIES, M. **Video Quality Estimation for Mobile Video Streaming**. Dissertação (Dissertação (Mestrado em Engenharia)) — Tampere University of Technology, Finlândia, 2008.

SALDANHA, M. *et al.* Performance analysis of vvc intra coding. **Journal of Visual Communication and Image Representation**, v. 79, p. 103202, 2021. ISSN 1047-3203. Disponível em: https://www.sciencedirect.com/science/article/pii/S1047320321001292.

Sandvine. **2024 Global Internet Phenomena Report**. [S.l.], 2024. Disponível em: https://www.sandvine.com/phenomena.

SCHWARZ, H. *et al.* Quantization and entropy coding in the versatile video coding (vvc) standard. **IEEE Transactions on Circuits and Systems for Video Technology**, v. 31, n. 10, p. 3891–3906, 2021.

TRAN, D.-L. *et al.* An efficient hardware implementation of residual data binarization in hevc cabac encoder. **Electronics**, MDPI, v. 9, n. 4, p. 684, 2020.

UHRINA, M. *et al.* Performance comparison of vvc, av1, hevc, and avc for high resolutions. **Electronics**, MDPI, v. 13, n. 5, p. 953, 2024.

XU, Q. *et al.* Detection of h. 266/vvc video transcoding based on refined block partition and filtering modes statistics in coding domain. **Applied Intelligence**, Springer, v. 55, n. 2, p. 121, 2025.