Método Rápido para Codificação de Nuvens de Pontos Dinâmicas Baseado em Predição de Particionamento de Blocos

Gustavo Rehbein
ghrehbein@inf.ufpel.edu.br
Universidade Federal de Pelotas
Video Technology Research Group – Vitech
Graduate Program in Computing
Pelotas, Brazil

Guilherme Corrêa
gcorrea@inf.ufpel.edu.br
Universidade Federal de Pelotas
Video Technology Research Group – Vitech
Graduate Program in Computing
Pelotas, Brazil

Cristiano Santos
cfdsantos@inf.ufpel.edu.br
Universidade Federal de Pelotas
Video Technology Research Group – Vitech
Graduate Program in Computing
Pelotas, Brazil

Marcelo Porto
porto@inf.ufpel.edu.br
Universidade Federal de Pelotas
Video Technology Research Group – Vitech
Graduate Program in Computing
Pelotas, Brazil

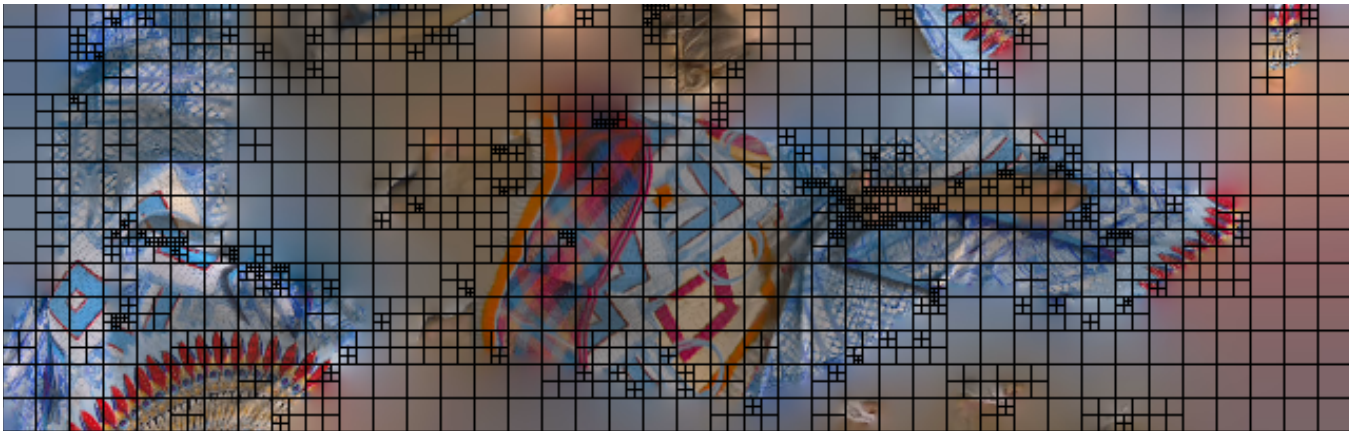

Figure 1: Quadro do subfluxo de vídeo de atributo do V-PCC e sua estrutura de particionamento de blocos.

ABSTRACT

Point clouds are a versatile 3D data representation format that captures spatial and attribute information about objects or environments. It has been widely used in applications like autonomous driving, augmented and virtual reality, and 3D scanning. However, the large volume of data involved in point cloud processing poses challenges in terms of storage, transmission, and real-time processing. The Video-based Point Cloud Compression (V-PCC) standard addresses these challenges by employing 2D video compression techniques to encode dynamic 3D point clouds. Despite its effectiveness, V-PCC demands a high computational cost, particularly in encoding geometry and attribute video sub-streams. This paper presents a machine learning-based approach to accelerate the V-PCC encoding, focusing on the Coding Unit partitioning process in the geometry and attribute sub-streams deployed in the reference software (TMC2). The proposed method significantly reduces the encoding complexity by using decision tree models with negligible impact on coding efficiency. Experimental results demonstrate an average encoding time reduction of 36.12% for geometry and attribute sub-streams, with a minimal impact on coding efficiency between -0.43% and 0.83% in terms of BD-Rate.

KEYWORDS

point clouds, machine learning, V-PCC, complexity reduction, random access, all-intra

1 INTRODUÇÃO

Nuvens de pontos constituem um formato versátil de representação de dados 3D, capaz de capturar informações espaciais sobre objetos ou ambientes por meio de uma coleção de pontos no espaço. Cada ponto contém coordenadas geométricas (x, y, z) e frequentemente inclui atributos adicionais, como cor, intensidade ou normais de superfície. Esse formato é fundamental em aplicações como condução autônoma, realidade aumentada, realidade virtual e escaneamento

In: Proceedings of the Brazilian Symposium on Multimedia and the Web (WebMedia'2025). Rio de Janeiro, Brazil. Porto Alegre: Brazilian Computer Society, 2025.

ISSN 2966-2753

3D de objetos. A capacidade de representar cenas complexas com precisão torna as nuvens de pontos valiosas para sistemas em tempo real. Entretanto, o grande volume de dados envolvido impõe desafios significativos em termos de armazenamento, transmissão e processamento em tempo real.

A compressão é fundamental para o processamento eficiente de conteúdo 3D. Para isso, o *Motion Picture Experts Group* (MPEG) definiu dois padrões de compressão de nuvens de pontos (PCC): o *Video-based PCC* (V-PCC) [15], destinado a nuvens de pontos dinâmicas (Categoria 2), e o *Geometry-based PCC* (G-PCC), voltado tanto para nuvens de pontos estáticas quanto para aquelas adquiridas dinamicamente (Categorias 1 e 3). Enquanto o G-PCC realiza a compressão diretamente sobre as informações 3D, o V-PCC emprega técnicas de compressão de vídeo 2D para codificar nuvens de pontos dinâmicas em 3D. Ao projetar a nuvem de pontos 3D em quadros de imagem 2D e aplicar métodos tradicionais de compressão de vídeo, o V-PCC alcança uma compressão significativa, mantendo a fidelidade geométrica e visual [6].

Este trabalho propõe uma implementação acelerada do V-PCC baseada em aprendizado de máquina, utilizando modelos de árvores de decisão para reduzir a complexidade da compressão de nuvens de pontos no V-PCC. O modelo é aplicado na etapa de particionamento de blocos durante a codificação dos subfluxos de vídeo de geometria e atributos (Figura 1) no Test Model Category 2 (TMC2) [13], o software de referência para o padrão V-PCC. Os resultados demonstram uma redução média de 35,06% no tempo total de codificação, 28,23% para o subfluxo de geometria e 43,18% para o subfluxo de atributos na configuração *Random Access* (RA). O impacto médio na eficiência de codificação, em termos de BD-Rate, variou de -0,07% a 0,00% para geometria e de 0,49% a -0,53% para atributos. Para a configuração *All-Intra* (AI), as reduções foram de 37,17% no tempo total, 35,32% para geometria e 39,35% para atributos. O impacto médio na eficiência de codificação foi de -0,13% a -0,12% para geometria e de 0,40% a -0,39% para atributos, conforme as métricas avaliadas.

Este artigo está organizado da seguinte forma: a Seção 2 apresenta o padrão V-PCC e seu funcionamento; a Seção 3 discute trabalhos relacionados; a Seção 4 descreve o método proposto; a Seção 5 apresenta os resultados experimentais; e, por fim, a Seção 6 conclui o trabalho.

2 VIDEO-BASED POINT CLOUD COMPRESSION (V-PCC)

O V-PCC é um padrão para compressão de nuvens de pontos dinâmicas (Categoria 2) proposto pelo MPEG [18]. O conceito central do V-PCC é utilizar codificadores de vídeo já estabelecidos para a codificação de nuvens de pontos. Para isso, os quadros da nuvem de pontos dinâmicas são inicialmente projetados para o espaço 2D, dividindo a nuvem em *patches* e realizando projeções ortogonais sobre um plano virtual. Esses *patches* são organizados em uma imagem 2D, que é então codificada por um codificador de vídeo. Por padrão, o software de referência do V-PCC (TMC2) utiliza o HEVC [20] como codificador de vídeo [6]. A arquitetura do codificador do V-PCC é apresentada na Figura 2.

Após a projeção 2D, apenas as coordenadas x e y de cada ponto são mantidas, tornando necessário armazenar as informações do eixo z (ou profundidade) de cada *patch* em uma imagem 2D separada. Para tratar casos em que mais de um ponto seja projetado para a mesma coordenada 2D, o V-PCC permite o uso de duas camadas, chamadas *near-layer* e *far-layer*. Cada *patch* é projetado para ambas as *layers*, sendo que o *near-layer* armazena os pontos com menor profundidade e o *far-layer* armazena os pontos sobrepostos de maior profundidade [15]. Assim, cada quadro 3D gera dois quadros bidimensionais.

Para reconstruir a nuvem de pontos, duas imagens são utilizadas para representar os *patches* 3D: uma imagem de Geometria, que preserva as informações de profundidade, e uma imagem de Atributos, que armazena os atributos dos *patches* (tipicamente informações de cor RGB). Uma imagem binária adicional, denominada mapa de ocupação, indica as localizações dos pontos válidos dentro da imagem 2D [15]. A Figura 3 ilustra esses três subfluxos de vídeo utilizados no V-PCC. Um processo de suavização é aplicado ao fundo das imagens de geometria e atributos, conforme exemplificado nas Figuras 3b e 3c. Esse processo é repetido para todos os quadros comprimidos da nuvem de pontos. Após a conclusão da projeção 2D, esses três subfluxos de vídeo (Figura 3) são codificados utilizando o codificador *High-Efficiency Video Coding* (HEVC). Os subfluxos codificados, juntamente com as informações de metadados dos *patches*, constituem o *bitstream* comprimido do V-PCC.

2.1 High-Efficiency Video Coding

O HEVC (H.265) [20] foi lançado em 2013 pelo grupo MPEG como sucessor do AVC (H.264). As ferramentas e técnicas introduzidas melhoraram a eficiência de codificação em aproximadamente 50% em relação ao seu predecessor [7]. Esse avanço se deve, em parte, ao uso dos *Coding Tree Units* (CTUs) como esquema de particionamento de blocos, que emprega *quad-trees* recursivas [8] para subdividir blocos em unidades menores, chamadas *Coding Units* (CUs), até tamanhos de 8×8 para amostras de luminância e 4×4 para crominância, com tamanho máximo de 64×64. A Figura 5 ilustra o particionamento de um CTU de 64×64 e sua respectiva estrutura de *Coding Tree*. As subdivisões (ou divisões) das CUs são indicadas por 1's, enquanto 0's denotam níveis sem subdivisões adicionais.

Cada CU pode ainda ser subdividida em duas ou mais Unidades de Predição, ou *Prediction Unit* (PU). Cada PU é predita separadamente e armazena informações sobre a predição, como vetores de movimento, modo de particionamento e modo de codificação. O HEVC suporta oito tipos de particionamento de PUs, apresentados na Figura 4, sendo dois quadráticos e seis retangulares. O particionamento $2N \times 2N$ não apresenta nenhuma subdivisão em relação à CU utilizada. O modo MSM permite que uma PU herde diretamente os parâmetros de movimento (vetores de movimento e referências) de blocos vizinhos já codificados, sem a necessidade de transmitir explicitamente novos vetores de movimento. Todas as possibilidades de particionamento de CTU e PU são avaliadas utilizando o processo de Otimização de Taxa-Distorção (*Rate-Distortion Optimization*, RDO). O RDO é responsável por selecionar a melhor configuração de particionamento que equilibra a qualidade do vídeo reconstruído (distorção) e a taxa de *bits* necessária para representá-lo.

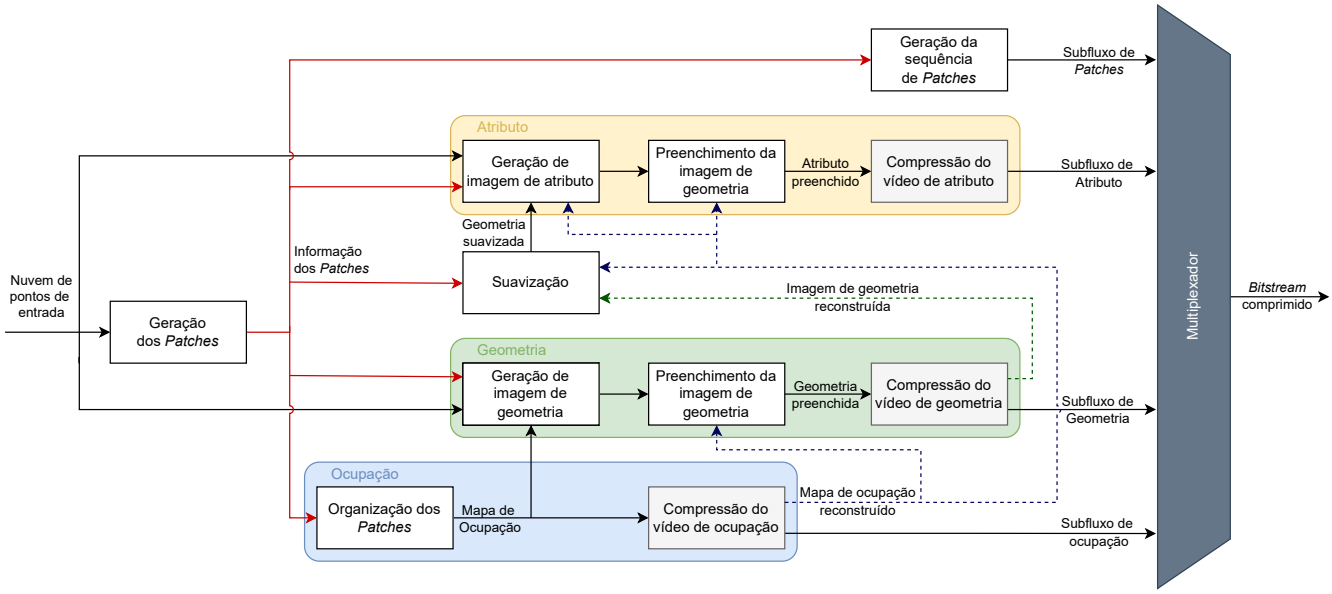

Figura 2: Diagrama de fluxo do V-PCC [15].

(a) **(b)** **(c)**

Figura 3: Exemplo de imagem de Ocupação (a), Geometria (b) e Atributos (c) extraídas de codificações do *software* de referência do V-PCC.

Embora o HEVC seja mais eficiente que o AVC, ele também é significativamente mais complexo [2]. Essa complexidade é ainda mais acentuada no contexto do V-PCC, onde três diferentes subfluxos de vídeo são gerados e codificados durante a compressão de uma nuvem de pontos dinâmica. De fato, ao utilizar o software de referência do HEVC fornecido com o TMC2 (HM 16.20 com *SCC*), até 90% do tempo total de codificação do V-PCC é consumido pelo processo de codificação do HEVC [17]. No entanto, essa porcentagem pode ser menor ao utilizar um codificador mais rápido, como o *x264* [11], com menor eficiência de codificação, ou até maior ao empregar um codificador mais avançado, como o VVC [3]. Portanto, reduzir a complexidade do HEVC no contexto do V-PCC é fundamental.

3 TRABALHOS RELACIONADOS

Alguns trabalhos propuseram métodos para reduzir o tempo de codificação da etapa de codificação de vídeo no V-PCC. Em [4], é apresentado um algoritmo de baixa complexidade para codificação intra no V-PCC, explorando correlações entre as sequências projetadas em 2D (ocupação, geometria, atributos) e introduzindo informações de projeção cruzada para aprimorar a predição de

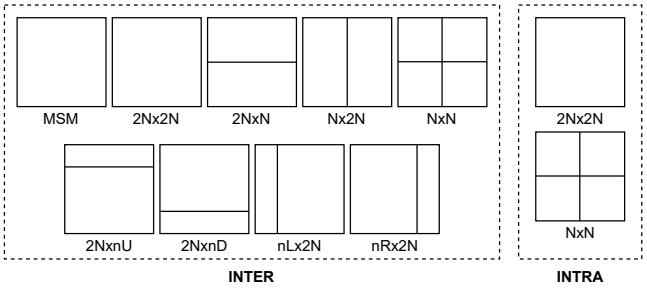

Figura 4: Tipos de particionamento de PUs no HEVC.

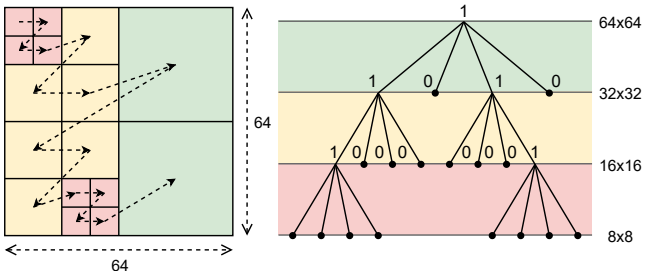

Figura 5: Exemplo de uma CTU 64×64 e sua estrutura de Coding Tree, subdividido em CUs menores.

partição das CUs. Os autores também utilizam um método de aprendizado guiado por taxa-distorção para decisões de partição, avaliado no modo AI.

Um método rápido para decisão do tamanho da CU nos subfluxos de geometria, utilizando agrupamento hierárquico não supervisionado, é proposto em [9], com um limiar adaptativo de ligação para aprimorar a determinação do tamanho da CU de acordo com os parâmetros de quantização. O método é avaliado sob a configuração AI. O *Blocky Occupancy Flag* é introduzido em [10], utilizando mapas de ocupação para identificar áreas relevantes de codificação e permitir a terminação antecipada da *Rate-Distortion Optimization* (RDO) das CUs em regiões não ocupadas. Essa abordagem também explora a homogeneidade espacial 2D/3D das imagens de geometria e modifica o processo de RDO para HEVC/V-PCC, sendo avaliada no modo RA.

Um método rápido e adaptativo de particionamento de CU, que integra informações de ocupação, redes neurais convolucionais (CNNs) para classificação inicial e otimização Bayesiana para refinar o particionamento em regiões de baixa confiança, é descrito em [19]. Por fim, [16] desenvolve um método rápido de decisão de modo de codificação baseado em uma rede Perceptron portátil e leve, com extração de características simples e uma função de perda adaptativa ponderada por taxa-distorção (RD), avaliada sob a configuração RA.

De modo geral, embora esses trabalhos apresentem estratégias promissoras para a redução do tempo de codificação no V-PCC, a maioria foca em subfluxos específicos, configurações particulares ou emprega modelos mais complexos. Até onde sabemos, nenhum deles realiza uma avaliação abrangente considerando todas as sequências do CTC e ambos os modos RA e AI, utilizando modelos de aprendizado de máquina simples e facilmente integráveis.

4 MÉTODO PROPOSTO

O método proposto tem como objetivo acelerar a etapa de codificação de vídeo do *software* de referência do V-PCC utilizando modelos de aprendizado de máquina para determinar a melhor estrutura de uma CTU durante o processo de particionamento dos blocos inter-quadros do HEVC. Uma vez identificada a profundidade ótima de uma CU, não há necessidade de realizar testes adicionais em níveis inferiores. Isso reduz o número de possibilidades avaliadas durante o processo de RDO, diminuindo assim o tempo de codificação. O padrão HEVC suporta até quatro profundidades de *Coding Tree*: 64×64, 32×32, 16×16 e 8×8. Como a subdivisão de uma CU não é possível no último nível, o método foca na determinação da profundidade nos níveis anteriores.

Para implementar o método, foi criado um conjunto de dados utilizando informações extraídas das codificações dos subfluxos de vídeo do V-PCC. O *software* de referência do HEVC versão 16.20, utilizado no TMC2, foi modificado para facilitar a coleta de características. As características coletadas estão listadas a seguir:

- Profundidade média utilizada nas CTUs vizinhas já codificadas (CTUs acima, à esquerda, acima à esquerda, acima à direita e CTUs colocalizadas nos primeiros quadros das duas listas de referência do HEVC).
- Particionamento da PU selecionada para a CU sendo avaliada ($2N \times 2N$, $2N \times N$, $N \times 2N$, $N \times N$, $2N \times nU$, $2N \times nD$, $nL \times 2N$ ou $nR \times 2N$).
- Razão entre os custos RD das PU $2N \times 2N$ e *Merge Skip Mode* (MSM).
- Custo RD dos modos de PU $2N \times 2N$, $2N \times N$ e MSM, respectivamente.
- Diferença normalizada entre os custos RD dos modos $2N \times 2N$ e MSM (Calculada conforme a Equação 1).
- *Flag* indicando se a CU foi predita com o modo *skip*.
- *Flag* indicando se a CU foi predita com o modo MSM.

$$DifNorm(2N \times 2N, MSM) = \left| \frac{RD(2N \times 2N) - RD(MSM)}{RD(MSM)} \right| \quad (1)$$

Para criar o conjunto de dados de treinamento para cada abordagem, foram utilizadas as sequências *BlueBackpack*, *BlueSpin*, *BlueSquat*, *CasualSpin*, *CasualSquat*, *ElegantDance*, *ElegantWave*, *FlowerDance*, *FlowerWave* e *Gymnast* do conjunto de dados UVG-VPC [5], com precisão de 11 bits. Os primeiros 64 quadros dessas sequências foram codificados com o TMC2 versão 22.1 [13] utilizando a configuração temporal RA e as cinco configurações de taxa do V-PCC (r1, r2, r3, r4 e r5), conforme descrito nas Condições Comuns de Teste (CTC) [12] do V-PCC. A configuração de taxa r1 oferece o menor bitrate, enquanto r5 proporciona a melhor qualidade com bitrate mais elevado.

Além das características previamente listadas, foi coletada uma variável-alvo para indicar se a CTU processada apresenta os melhores resultados de RDO. Um valor 0 ou *false* indica que a CTU selecionada está em um nível de profundidade diferente. Esses dados permitem o treinamento de um modelo de aprendizado de máquina para acelerar o processo de particionamento de blocos. Caso os

Tabela 1: Espaço de busca utilizado para o ajuste de hiperparâmetros.

	Hiperparâmetro	Espaço de busca	Tamanho
a.	Criterion	{'gini', 'entropy'}	2
b.	min_samples_split	[10–90 (passo 20)] ∪ [100–1150 (passo 250)]	10
c.	min_samples_leaf	[10–90 (passo 20)] ∪ [100–1150 (passo 250)]	10
d.	max_depth	[2–14 (passo 1)]	13
e.	max_leaf_nodes	[10–90 (passo 20)] ∪ [100–900 (passo 200)]	10
f.	max_features	[1–11 (passo 1)]	11
	Total		286.000

Tabela 2: Resultados do ajuste de hiperparâmetros utilizando busca aleatória.

Dataset	Hiperparâmetro (Como na Tabela 1)						F1-Score
	a	b	c	d	e	f	
Geometria_16x16	gini	50	10	14	900	10	0,85
Geometria_32x32	gini	350	70	13	900	10	0,81
Geometria_64x64	gini	350	10	14	900	11	0,81
Atributo_16x16	gini	10	30	12	900	5	0,91
Atributo_32x32	gini	10	70	14	900	11	0,90
Atributo_64x64	gini	10	70	14	900	11	0,89

dados de uma CTU sejam classificados como 1 ou *true*, isso indica que a profundidade atual da CTU é ótima, permitindo a dispensa de subdivisões ou CTUs de níveis inferiores. Os dados coletados foram organizados em seis conjuntos de dados, conforme o subfluxo e o tamanho da CTU: Geometria_64x64, Geometria_32x32, Geometria_16x16, Atributo_64x64, Atributo_32x32 e Atributo_16x16.

Devido à natureza dos dados coletados, a maioria dos exemplos corresponde a 0 ou *false*, representando CTUs não selecionadas pelo processo de RDO. Portanto, foi necessário realizar uma etapa de balanceamento dos dados para cada conjunto. O número de exemplos da classe minoritária para cada sequência e configuração de taxa foi anotado, e o menor valor foi utilizado para subamostrar as demais sequências e taxas. Essa abordagem garante um número igual de exemplos *true* e *false* para cada sequência e configuração de taxa no conjunto de dados. Esse processo é realizado de forma independente para cada um dos seis conjuntos mencionados anteriormente.

Após isso, modelos de aprendizado de máquina foram treinados utilizando o algoritmo de árvore de decisão da biblioteca SciKit-Learn [14]. O algoritmo de árvore de decisão foi escolhido por sua flexibilidade e inferência de baixa complexidade. Uma etapa de ajuste de hiperparâmetros foi realizada utilizando o algoritmo RandomizedSearchCV, também da SciKit-Learn [14]. Nesta etapa, um processo de busca aleatória foi conduzido. O espaço de busca é apresentado na Tabela 1. O RandomizedSearchCV foi configurado para 2.860 iterações e 5 *folds* de validação cruzada. As configurações de hiperparâmetros selecionadas são mostradas na Tabela 2. Os modelos finais para cada conjunto de dados foram treinados usando os hiperparâmetros selecionados e implementados em uma versão modificada do software de referência TMC2 que incorpora o método proposto. Como o método proposto afeta apenas a etapa de RDO do codificador de vídeo, nenhuma alteração é necessária no lado do decodificador do V-PCC.

5 RESULTADOS EXPERIMENTAIS

Para avaliar o método proposto, foram realizadas avaliações utilizando as sequências de teste V-PCC CTC [12]: sequências de 10 bits (*longdress*, *loot*, *queen*, *redandblack*, *soldier*) e 11 bits (*dancer*, *basketball_player*) (Figura 6). Embora os modelos tenham sido treinados com dados de 11 bits, as sequências de 10 bits foram incluídas para avaliar o desempenho em diferentes níveis de precisão. Todas as sete sequências foram codificadas com a configuração temporal RA e AI em cinco configurações de taxa, utilizando os primeiros 64 quadros de cada sequência. Os experimentos foram conduzidos utilizando o software de referência TMC2, comparando seu desempenho original com a versão modificada que incorpora o método proposto.

Para cada experimento, o *Peak Signal-to-Noise Ratio* (PSNR) foi calculado para a geometria das nuvens de pontos reconstruídas utilizando as métricas ponto a ponto (D1) e ponto a plano (D2) [12]. O PSNR das informações de atributos foi obtido a partir dos canais de Luminância (Luma), Crominância Azul (Cb) e Crominância Vermelha (Cr). Essas métricas foram calculadas com a ferramenta *Distortion Metric*, conforme especificado nas CTC [12]. Adicionalmente, os tempos de codificação dos subfluxos de vídeo foram registrados para avaliar o desempenho da codificação. Os experimentos utilizaram a versão 22.1 do TMC2 [13] em um sistema com processador Intel Xeon E5-4650 e 512 GB de RAM.

A Tabela 3 apresenta os resultados de BD-Rate [1] para as nuvens de pontos reconstruídas com a configuração RA. Também inclui os resultados de BD-Rate considerando as taxas de bits para geometria (GeomRate) e atributos (AttrRate), bem como a taxa de bits total (TotalRate) da nuvem de pontos comprimida.

Os resultados de redução de tempo de codificação mostram que a abordagem proposta conseguiu reduzir, em média, em 35,06% o tempo total de codificação, em 28,23% para o subfluxo de geometria e em 43,18% para o subfluxo de atributos. Os valores de redução de tempo de codificação variaram entre as sequências de teste, com a sequência *queen* apresentando a maior redução (45,73% no tempo total, 40,84% no subfluxo de geometria e 51,00% no subfluxo de atributos) e a sequência *dancer* apresentando a menor redução (28,21% no tempo total, 22,24% no subfluxo de geometria e 35,05% no subfluxo de atributos).

Os resultados médios de BD-Rate foram de 0,49% para Luma, -0,53% para Cb e -0,06% para Cr, considerando o Attr.BD-AttrRate. Observando os resultados obtidos para cada sequência de teste, a sequência *queen* apresentou o maior aumento de BD-Rate para Luma (1,71%), enquanto as demais sequências apresentaram valores próximos ou abaixo da média. Esse valor discrepante pode ser justificado pelo fato de a sequência *queen* ser a única criada a partir de conteúdo sintético, o que pode dificultar a generalização dos modelos treinados predominantemente com dados de cenas reais. Para os canais de cor Cb e Cr, os resultados médios indicam uma pequena diminuição de BD-Rate, chegando a -1,82% para o canal Cb da sequência *dancer*. Esses resultados mostram que a abordagem não prejudicou a qualidade da nuvem de pontos reconstruída e também não prejudicou a taxa de *bits* do conteúdo comprimido.

Além disso, os resultados para o *Attr.BD-TotalRate* também foram próximos dos valores observados para o Attr.BD-AttrRate, indicando que o impacto da abordagem nos atributos se mantém mesmo

Tabela 3: Resultados experimentais de BR-Rate e redução de tempo de codificação obtidos com o método proposto utilizando a configuração temporal *Random Access*.

Sequência	Geom.BD-GeomRate		Attr.BD-AttrRate			Geom.BD-TotalRate		Attr.BD-TotalRate			ΔTempo		
	D1	D2	Luma	Cb	Cr	D1	D2	Luma	Cb	Cr	ΔT_{Total}	ΔT_{Geom}	ΔT_{Atrib}
longdress	-0,03%	0,07%	0,24%	-0,05%	0,25%	-0,01%	0,05%	0,22%	-0,05%	0,17%	-29,10%	-25,13%	-33,86%
loot	-0,26%	-0,31%	0,12%	0,98%	-0,96%	-0,15%	-0,21%	-0,02%	0,54%	-0,69%	-37,64%	-25,39%	-53,34%
queen	-0,04%	0,13%	1,71%	-1,36%	0,71%	-0,39%	-0,31%	1,47%	-0,57%	0,80%	-45,73%	-40,84%	-51,00%
redandblack	0,01%	0,21%	0,34%	-0,12%	0,02%	-0,10%	0,05%	0,40%	0,08%	0,13%	-30,11%	-23,82%	-37,61%
soldier	-0,54%	-0,44%	0,13%	-1,05%	-1,20%	-0,29%	-0,28%	-0,01%	-0,59%	-0,93%	-43,04%	-35,31%	-52,11%
basketball	0,24%	0,13%	0,50%	-0,31%	1,36%	0,09%	-0,02%	0,50%	-0,12%	0,92%	-31,59%	-24,90%	-39,27%
dancer	0,16%	0,20%	0,37%	-1,82%	-0,61%	-0,01%	0,02%	0,41%	-1,01%	-0,32%	-28,21%	-22,24%	-35,05%
Média	-0,07%	0,00%	0,49%	-0,53%	-0,06%	-0,12%	-0,10%	0,42%	-0,25%	0,01%	-35,06%	-28,23%	-43,18%

Tabela 4: Resultados experimentais de BR-Rate e redução de tempo de codificação obtidos com o método proposto utilizando a configuração temporal *All-Intra*.

Sequência	Geom.BD-GeomRate		Attr.BD-AttrRate			Geom.BD-TotalRate		Attr.BD-TotalRate			ΔTempo		
	D1	D2	Luma	Cb	Cr	D1	D2	Luma	Cb	Cr	ΔT_{Total}	ΔT_{Geom}	ΔT_{Atrib}
longdress	-0,12%	-0,13%	0,21%	-0,28%	-0,15%	-0,11%	-0,14%	0,19%	-0,26%	-0,14%	-33,67%	-33,41%	-34,09%
loot	-0,12%	-0,13%	0,17%	0,21%	-0,16%	-0,09%	-0,11%	0,14%	0,14%	-0,17%	-38,39%	-34,81%	-42,59%
queen	-0,15%	-0,14%	1,25%	-0,41%	-0,31%	-0,43%	-0,42%	0,99%	-0,21%	-0,17%	-38,09%	-38,52%	-37,56%
redandblack	-0,14%	-0,15%	0,21%	-0,39%	0,02%	-0,08%	-0,12%	0,18%	-0,32%	0,01%	-38,40%	-36,32%	-40,78%
soldier	-0,08%	-0,07%	0,14%	-0,49%	-0,48%	-0,11%	-0,11%	0,13%	-0,38%	-0,44%	-32,77%	-31,28%	-34,57%
basketball	-0,13%	-0,13%	0,47%	-0,95%	-0,71%	-0,27%	-0,28%	0,43%	-0,67%	-0,46%	-39,40%	-36,01%	-43,34%
dancer	-0,14%	-0,11%	0,36%	-0,39%	-0,56%	-0,23%	-0,19%	0,32%	-0,30%	-0,44%	-39,49%	-36,90%	-42,50%
Média	-0,13%	-0,12%	0,40%	-0,39%	-0,34%	-0,19%	-0,20%	0,34%	-0,29%	-0,26%	-37,17%	-35,32%	-39,35%

quando se considera o *bitrate* total do *bitstream*. Vale lembrar que o Attr.BD-TotalRate leva em conta não apenas o subfluxo de atributos, mas também o subfluxo de geometria, refletindo o impacto global da abordagem sobre a eficiência de compressão do V-PCC. Os resultados médios de BD-Rate relacionados aos subfluxos de geometria foram próximos de zero (respectivamente -0,07% para D1 e 0,00% para D2), mostrando que praticamente não houve impacto negativo na eficiência de codificação da geometria. Além disso, ao analisar o Geom.BD-TotalRate, observa-se o mesmo comportamento.

A maioria dos trabalhos recentes na literatura focados na redução da complexidade do V-PCC relata resultados apenas para a configuração temporal AI. Experimentos adicionais foram realizados utilizando a mesma configuração e modelos treinados, também sob a configuração AI, para permitir uma comparação justa com trabalhos relacionados.

Vale ressaltar que o modo AI no padrão V-PCC refere-se à organização e à consistência temporal dos *patches*, e não diretamente ao modo de predição utilizado no codificador de vídeo. De fato, a implementação padrão do HEVC no TMC2 permite o uso de blocos

(a) longdress (b) loot (c) queen (c) redandblack (d) soldier (e) basketball player (f) dancer

Figura 6: Primeiro quadro de cada sequência de teste da CTC do V-PCC.

Tabela 5: Comparação com trabalhos relacionados.

Trabalho	Geom.BD-GeomRate		Attr.BD-AttrRate			Geom.BD-TotalRate		Attr.BD-TotalRate			ΔTime		
	D1	D2	Luma	Cb	Cr	D1	D2	Luma	Cb	Cr	ΔT_{Tempo}	ΔT_{Geom}	ΔT_{Atrib}
Nosso (RA)	-0,07%	0,00%	0,49%	-0,53%	-0,06%	-0,12%	-0,10%	0,42%	-0,25%	0,01%	-35,06%	-28,23%	-43,18%
Lin et al.[10]	0,67%	0,82%	-1,15%	-6,85%	-7,28%	0,02%	0,02%	-0,31%	-4,08%	-4,31%	-55,43%	–	–
Que; Li[16]	-0,4%	-0,4%	0,1%	-0,8%	-0,2%	0,2%	0,0%	-0,5%	-1,0%	-0,6%	-43,13%	–	–
Nosso (AI)	-0,04%	0,05%	0,02%	-0,17%	-0,05%	0,21%	0,35%	-0,04%	-0,16%	-0,09%	-42,73%	-46,99%	-37,88%
Li et al.[9]	0,2%	0,2%	–	–	–	0,0%	0,1%	–	–	–	–	-56,7%	–
Gao et al.[4]*	0,15%	0,3%	0,14%	-0,27%	-0,6%	0,08%	0,33%	0,16%	-0,14%	-0,41%	-57,8%	-58,25%	-58,51%
Song et al.[19]	0,42%	0,56%	0,12%	0,28%	-0,43	–	–	–	–	–	-54,75%	57,37%	54,43%

* Valores médios para um subconjunto diferente de sequências recomendadas do CTC.

com modo *inter* para codificação de quadros de *far-layer* na configuração AI [15]. Assim, embora o método proposto seja aplicado apenas aos blocos preditos por *inter* nos subfluxos de vídeo, ele também terá impacto ao utilizar a configuração AI do V-PCC.

Os resultados apresentados na Tabela 4 são semelhantes aos obtidos com a configuração RA, com uma leve melhora na eficiência de codificação da geometria. Esse aumento de eficiência sugere que o uso de blocos maiores nos quadros do *far-layer* (onde o processo de RDO pode ter sido encerrado prematuramente) não degrada significativamente a qualidade da nuvem de pontos reconstruída, nem aumenta o bitrate dos subfluxos. Como o *far-layer* auxilia em casos onde múltiplos pontos são projetados sobre a mesma amostra [15], e a maioria das sequências de teste do CTC apresenta superfícies suaves bem definidas, é provável que a maior parte dos pontos não se sobreponha. Dessa forma, o *far-layer* permanece amplamente inutilizado na nuvem de pontos reconstruída.

Os resultados médios de BD-Rate para Luma, Cb e Cr foram de 0,40%, -0,39% e -0,34%, respectivamente, considerando o Attr.BD-AttrRate. Esses valores indicam que o método proposto preserva a eficiência de compressão dos subfluxos de atributos, com impacto mínimo na qualidade dos dados reconstruídos. Em relação ao tempo de codificação, observou-se uma redução média de 37,17% no tempo total, 35,32% para o subfluxo de geometria e 39,35% para o subfluxo de atributos, demonstrando que o método é eficaz também sob a configuração All-Intra.

Para os subfluxos de geometria, os valores médios de BD-Rate foram de -0,13% para D1 e -0,12% para D2, mostrando que a abordagem não compromete a eficiência de compressão da geometria. O Geom.BD-TotalRate apresentou resultados próximos de zero, reforçando que o impacto sobre a taxa de bits da geometria é desprezível. De modo geral, os resultados obtidos com a configuração AI confirmam que o método proposto mantém o desempenho observado no modo RA, com ganhos consistentes em redução de tempo e sem prejuízo relevante à eficiência de codificação dos subfluxos de vídeo. Vale destacar que, em ambos os casos (RA e AI), a sequência de teste *queen* apresenta o maior BD-Rate para o subfluxo de atributos, o que eleva o resultado médio. Isso pode ser explicado pela natureza do conteúdo, já que *queen* é a única sequência composta por informação sintética.

A Tabela 5 apresenta uma comparação com trabalhos relacionados que focam no particionamento de CU do HEVC no contexto do V-PCC [4, 9, 10, 16, 19]. São exibidos os valores médios de BD-Rate e redução de tempo de codificação para cada método. Ao comparar os resultados obtidos com os trabalhos relacionados, observa-se que o método proposto apresenta desempenho superior em termos de eficiência de codificação da geometria. Enquanto os métodos de [10], [9],[4] e [19] resultam em BD-Rate positivo para as métricas GeomRate e TotalRate, indicando perda de eficiência, o método proposto apresenta BD-Rate negativo ou próximo de zero, evidenciando ganho ou manutenção da eficiência de compressão.

Para os atributos, os resultados do método proposto são próximos de zero ou levemente negativos/positivos, indicando impacto mínimo na eficiência de codificação. Destaca-se que, para o canal Cb, o método iguala ou supera o ganho de eficiência observado em [4], além de apresentar leve melhora para o canal Cr. Por outro lado, observa-se uma pequena perda de eficiência para o canal Luma, porém inferior à registrada em outros trabalhos.

Em relação à redução do tempo de codificação, embora alguns métodos da literatura apresentem valores ligeiramente superiores, o método proposto mantém reduções expressivas e competitivas, tanto nas configurações RA quanto AI. É importante ressaltar que os resultados foram obtidos considerando todas as sequências recomendadas pelo CTC e ambas as configurações temporais, o que torna a avaliação mais abrangente e robusta em relação à maioria dos trabalhos comparados, que utilizam subconjuntos de sequências ou apenas uma configuração.

De modo geral, os resultados indicam que o método proposto oferece um melhor equilíbrio entre redução de complexidade e eficiência de compressão, especialmente para os subfluxos de geometria, sem comprometer significativamente a qualidade dos atributos. Além disso, a abrangência dos testes reforça a robustez e aplicabilidade da abordagem em diferentes cenários do V-PCC.

6 CONCLUSÃO

Este trabalho apresentou uma solução para redução do custo computacional do codificador V-PCC utilizando aprendizado de máquina para acelerar a codificação de nuvens de pontos dinâmicas. Um conjunto de dados foi criado a partir da extração de informações de sequências de teste existentes e modelos de aprendizado de máquina foram treinados e integrados ao software de referência do V-PCC. Os experimentos avaliaram o impacto do método proposto, resultando em uma redução média de 35,06% no tempo de codificação total para a configuração temporal RA e 37,17% para a configuração AI, sem comprometer a eficiência de compressão de forma significativa. Os resultados mostraram que o método mantém a eficiência de codificação dos subfluxos de geometria e atributos, com BD-Rate próximo de zero, indicando que a abordagem não prejudica a qualidade da nuvem de pontos reconstruída. Além disso, os resultados

foram comparados com trabalhos relacionados, mostrando que o método proposto obteve resultados competitivos em termos de redução de complexidade e eficiência de compressão. Destaca-se que este trabalho, dentre os trabalhos focados na redução do tempo de processamento da etapa de codificação de vídeo, foi o primeiro a realizar uma avaliação abrangente utilizando ambos os modos temporais, RA e AI, e considerando todas as sequências recomendadas pelo conjunto de CTC do V-PCC, reforçando a robustez e a aplicabilidade da abordagem proposta.

AGRADECIMENTOS

Os autores deste trabalho gostariam de agradecer à Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) Código Financeiro 001, ao Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq) e à Fundação de Amparo à Pesquisa do Estado do Rio Grande do Sul (FAPERGS) pelo financiamento desta pesquisa.

REFERÊNCIAS

[1] Gisle Bjontegaard. 2001. Calculation of average PSNR differences between RD-curves. *ITU SG16 Doc. VCEG-M33* (2001).
[2] Frank Bossen, Benjamin Bross, Karsten Suhring, and David Flynn. 2012. HEVC Complexity and Implementation Analysis. *IEEE Transactions on Circuits and Systems for Video Technology* 22, 12 (2012), 1685–1696. https://doi.org/10.1109/TCSVT.2012.2221255
[3] Benjamin Bross, Ye-Kui Wang, Yan Ye, Shan Liu, Jianle Chen, Gary J Sullivan, and Jens-Rainer Ohm. 2021. Overview of the versatile video coding (VVC) standard and its applications. *IEEE Transactions on Circuits and Systems for Video Technology* 31, 10 (2021), 3736–3764.
[4] Wei Gao, Hang Yuan, Ge Li, Zhu Li, and Hui Yuan. 2023. Low Complexity Coding Unit Decision for Video-Based Point Cloud Compression. *IEEE Transactions on Image Processing* 33 (2023), 149–162.
[5] Guillaume Gautier, Alexandre Mercat, Louis Fréneau, Mikko Pitkänen, and Jarno Vanne. 2023. UVG-VPC: Voxelized Point Cloud Dataset for Visual Volumetric Video-based Coding. In *2023 15th International Conference on Quality of Multimedia Experience (QoMEX)*. IEEE, 244–247.
[6] Danillo Graziosi, Ohji Nakagami, Satoru Kuma, Alexandre Zaghetto, Teruhiko Suzuki, and Ali Tabatabai. 2020. An overview of ongoing point cloud compression standardization activities: Video-based (V-PCC) and geometry-based (G-PCC). *APSIPA Transactions on Signal and Information Processing* 9 (2020), e13.
[7] Dan Grois, Detlev Marpe, Amit Mulayoff, Benaya Itzhaky, and Ofer Hadar. 2013. Performance comparison of H.265/MPEG-HEVC, VP9, and H.264/MPEG-AVC encoders. In *2013 Picture Coding Symposium (PCS)*. 394–397.
[8] Il-Koo Kim, Junghye Min, Tammy Lee, Woo-Jin Han, and JeongHoon Park. 2012. Block partitioning structure in the HEVC standard. *IEEE Transactions on Circuits and Systems for Video Technology* 22, 12 (2012), 1697–1706.
[9] Yue Li, Jun Huang, Chaofeng Wang, and Hongyue Huang. 2024. Unsupervised learning-based fast CU size decision for geometry videos in V-PCC. *Journal of Real-Time Image Processing* 21, 1 (2024), 11.
[10] Ting-Lan Lin, Hong-Bin Bu, Yan-Cheng Chen, Jun-Rui Yang, Chi-Fu Liang, Kun-Hu Jiang, Ching-Hsuan Lin, and Xiao-Feng Yue. 2021. Efficient quadtree search for HEVC coding units for V-PCC. *IEEE Access* 9 (2021), 139109–139121.
[11] Loren Merritt and Rahul Vanam. 2006. x264: A high performance H. 264/AVC encoder. *http://neuron2. net/library/avc/overview_x264_v8_5.pdf* (2006).
[12] MPEG. 2020. Common Test Conditions for V3C and V-PCC. *ISO/IEC JTC 1/SC 29/WG 11* (2020).
[13] MPEG. 2024. Video Point Cloud Compression - VPCC - mpeg-pcc-tmc2 test model candidate software. https://github.com/MPEGGroup/mpeg-pcc-tmc2.
[14] Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, et al. 2011. Scikit-learn: Machine learning in Python. *the Journal of machine Learning research* 12 (2011), 2825–2830.
[15] Marius Preda. 2020. V-PCC codec description. https://www.mpeg.org/wp-content/uploads/mpeg_meeti ngs/134_OnLine/w20352.zip.
[16] Shicheng Que and Yue Li. 2023. Portable Perceptron Network-Based Fast Mode Decision for Video-Based Point Cloud Compression. *CAAI Artificial Intelligence Research* 2 (2023).
[17] Gustavo Rehbein, Eduardo Costa, Guilherme Corrêa, Cristiano Santos, and Marcelo Porto. 2024. A Machine-Learning-Driven Fast Video-based Point Cloud Compression (V-PCC). In *Proceedings of the 30th Brazilian Symposium on Multimedia and the Web* (Juiz de Fora/MG). SBC, Porto Alegre, RS, Brasil, 20–27. https://doi.org/10.5753/webmedia.2024.242069
[18] R Schaefer. 2017. Call for proposals for point cloud compression V2. In *ISO/IEC JTC1 SC29/WG11 MPEG, 117th Meeting. Hobart, TAS.*
[19] Wenjun Song, Xinqi Liu, and Qiuwen Zhang. 2025. Fast Coding Unit Partitioning Method for Video-Based Point Cloud Compression: Combining Convolutional Neural Networks and Bayesian Optimization. *Electronics* 14, 7 (2025), 1295.
[20] Gary J Sullivan, Jens-Rainer Ohm, Woo-Jin Han, and Thomas Wiegand. 2012. Overview of the high efficiency video coding (HEVC) standard. *IEEE Transactions on circuits and systems for video technology* 22, 12 (2012), 1649–1668.