Compressão de Nuvem de Pontos - PCC

Franklin J. Ventura Quico[1]

fventuraq@midiacom.uff.br

[1]Instituto de Computação – Universidade Federal Fluminense (UFF)
Niterói – RJ – Brasil

Abstract. *Point cloud compression is an emerging technology that studies the compression of 3D digital content, MPEG is one of the main drivers in this area of research, which is why in early 2017 it called for researchers and companies to work on research on point cloud compression. At the end of 2019 the results of these investigations were presented Lidar Point Cloud Compression L-PCC, Superfice Point Cloud Compression S-PCC and Video Point Cloud Compression V-PCC, these results were used for a proposed ISO standard for Point Cloud Compression. Based on the results of the investigations obtained by MPEG, more investigations have been developed, many of them improve some aspects of MPEG-PCC. This document presents the MPEG-PCC base investigation and subsequent investigations.*

1. Introdução

A tecnologia emergente relacionada a imagens ou vídeos em 3D levou a pesquisas em diferentes áreas, seja no hardware usado para a projeção desse tipo de conteúdo ou no software que funciona por trás dele. A compressão de nuvem de pontos (*Point Cloud Compression* - PCC) é uma das técnicas usadas para compressão de conteúdo 3D. Atualmente, existem muitas pesquisas que abordam diferentes técnicas para compressão de nuvem de pontos. Essas investigações, por sua vez, têm abordagens diferentes, pois quando se trata de compressão de conteúdo 3D, existem diferentes cenários. As nuvens de pontos podem ser de dois tipos, estáticas ou dinâmicas. Uma nuvem de pontos dinâmica é uma sequência de nuvens de pontos estáticas, cada uma em seu próprio quadro [Tian et al. 2017].

Este trabalho apresenta as diferentes investigações realizadas até o momento em relação à compressão de nuvens de pontos PCC, tendo como referência a pesquisa de padrões MPEG PCC [Schwarz et al. 2018]. Neste trabalho, o processo de captura de conteúdo 3D não será estudado.

O restante do texto está estruturado da seguinte forma. A seção 2 apresenta referências teóricas para uma melhor compreensão da compressão de nuvem de pontos. Os modelos base propostos pelo padrão MPEG PCC são apresentados na seção 3. Na seção 4 discute estudos posteriores ao padrão proposto pelo MPEG PCC, principalmente os processos de compressão e os resultados obtidos em cada investigação. Finalmente, a seção 5 se apresentan algumas considerações finais.

1

[1]Trabalho apresentado como parte da avaliação da disciplina Sistemas Multimídia do Programa de Pós-Graduação em Computação do IC/UFF

	queen	longdress	loot	redandblack	soldier
Average number of points (in 300 frames)	1,005,000	834,000	794,000	727,000	1,076,000
Bitrates for transmitting uncompressed video (Mbytes/s)	514.47	542.22	490.61	448.21	681.96

Figura 1. Exemplo de nuvem de pontos dinâmica com média de pontos ao longo do tempo e taxas de bits correspondentes em (Mbytes/s) [Cao et al. 2019]

2. Referencial teórico

Nesta seção, os conceitos básicos de multimídia 3D serão abordados, da maneira a capturar o conteúdo 3D e representá-lo de uma perspectiva pura; isso servirá para entender a importância da compressão do conteúdo 3D, seja com base em malhas ou nuvem de pontos.

Em [Cao et al. 2019], é feita uma análise de 3D *Point Cloud Compression* de como isso mudou a maneira de compactar o conteúdo 3D dinâmico. Uma análise das características do conteúdo 3D bruto também é realizada para que seja feita uma análise do tamanho real do conteúdo 3D e os recursos necessários para o uso deste conteúdo.

2.1. Vídeo em 3D

O vídeo 3D é uma tecnologia de filmagem e projeção que simula a visão tridimensional humana real. Para obter uma visualização agradável do conteúdo 3D, é necessário o uso de nuvens de pontos de alta densidade. Esse tipo de nuvens de pontos tem uma grande desvantagem quanto ao tamanho do arquivo. Na Figura 1, é mostrada uma referência do número médio de pontos em uma sequência de 300 quadros e o peso desses em Mbytes/s.

Como mostrado na Figura 1, o número médio de pontos das diferentes sequências é 887.200 pontos e a taxa de bits é de 500 MBytes/s. Caso se deseje transmitir qualquer uma das sequências vistas anteriormente, é necessário ter uma grande largura de banda.

Para o uso de conteúdo 3D dinâmico, seja em streaming de vídeo, realidade aumentada AR, realidade mista MR, etc., é necessário usar um método de compressão eficiente que possa trabalhar com um grande volume de dados complexos e ter uma grande largura de banda para transferência eficiente de dados. Por esses motivos, é necessário investigar essa área.

2.2. Nuvem de pontos

Existem várias definições para nuvens de pontos. De acordo com [Cao et al. 2019], uma nuvem de pontos 3D é um conjunto de pontos $P_i \ n_i \ = \ 1$ que carrega informações geométricas como atributos (propriedades fotométricas). O termo nuvem referencia algo desorganizado. De acordo com [Chou et al. 2019], uma nuvem de pontos é um conjunto de pontos (também conhecidos como locais ou posições) no espaço euclidiano, no qual um vetor de atributos (como uma cor tripla) pode ser associado a cada ponto.

As nuvens de pontos podem ser estáticas ou dinâmicas; no caso das nuvens de pontos dinâmicas, o número de pontos em cada quadro pode variar.

As nuvens de pontos fornecem uma representação natural dos hologramas, porque cada ponto, com um ou mais atributos de cores, pode representar naturalmente a cor dos raios que passam por esse ponto.

Como o áudio e o vídeo, a mídia volumétrica será usada em três cenários principais de comunicação: consumo sob demanda de conteúdo pré-gravado, transmissão de conteúdo ao vivo ou pré-gravado e comunicação interativa, como conferências [Chou et al. 2019].

Os dois problemas fundamentais na compressão de nuvem de pontos são a compressão de informações geométricas e a compressão de informações de atributos.

A informação geomárica descreve a posição de um ponto em uma nuvem de pontos nas coordenadas (x, y, z) para definir sua posição no espaço.

As informações sobre atributos são as informações que acompanham cada ponto de uma nuvem de pontos. Esses atributos podem variar de acordo com os diferentes casos de uso em que são usados. Os atributos mais utilizados são vetores de cor (R, G, B) e vetores normais (nx, ny, nz). Os vetores de refletância também são considerados em alguns casos.

2.3. Casos de uso e aplicativos

Alguns casos de uso prático de vídeos 3D e nuvens de pontos são:

- Museus do patrimônio cultural.
- Visualizações 3D gratuitas.
- Telepresença imersiva em tempo real.
- Visualização de conteúdo VR com paralaxe interativo.
- Mapeamento móvel.
- Navegação automática.
- A telepresença 3D em tempo real é um dos principais aplicativos imersivos de vídeo e nuvem de pontos.
- Holoportation da Microsoft.
- Tecnologia de vídeo volumétrica 8i.
- Variações imersivas de vídeo como reprodução esportiva baseada em HMD (tela montada na cabeça) com base nas visualizações VR e 3D.

2.4. Octree

Uma *octree* é uma das bases usadas pelo MPEG para a compressão de nuvens de pontos PCC, uma vez que foi concluído que o uso de *octrees* fornece resultados muito melhores do que outras técnicas.

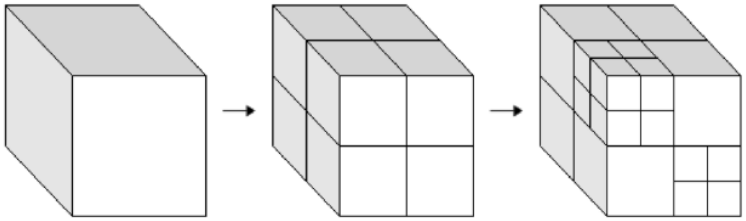

Figura 2. Gerando estrutura octree por subdivisão recursiva. [Schwarz et al. 2018]

O processo de codificação *octree* é feito da seguinte forma. Primeiro uma caixa limite B alinhada ao eixo cúbico é definida por $(0, 0, 0)$ e $(2^n, 2^n, 2^n)$ onde n é o menor número que verifica a seguinte desigualdade.

$$2^n > max(max(\hat{x}_j), max(\hat{y}_j), max(\hat{z}_j))$$

Depois, uma estrutura *octree* é construída subdividindo recursivamente B, conforme a Figura 2.

Em cada estágio, o cubo atual é subdividido em 8 subcubos. Depois é gerado um código de 8 bits chamado de código de subdivisão, associando 1 bit a cada subcubo para indicar se ele contém pontos (ou seja, está ocupado e tem um valor de 1) ou não (ou seja, está vazio e tem um valor de 0). Os subcubos ocupados que tem valor de 1 são subdivididos.

3. Compressão de nuvem de pontos MPEG-PCC

Um dos artigos de base desta pesquisa é o MPEG-PCC [Schwarz et al. 2018]. Este artigo descreve uma investigação de anos com bons resultados. O objetivo do MPEG-PCC é que ele seja entregue como um padrão ISO até 2020. Os resultados desta pesquisa datam do final de 2017 e abordam três áreas específicas: *Lidar Point Cloud Compression* (L-PCC), *Surface Point Cloud Compression* (S-PCC) e *Video Point Cloud Compression* (V-PCC), nas quais participaram muitas propostas.

Em janeiro de 2017, o MPEG lançou uma chamada para propostas de compressão de nuvem de pontos PCC, dividindo-as em 3 categorias:

- Nuvens de pontos estáticos.
- Nuvem de pontos dinâmica.
- Nuvem de pontos adquirida dinamicamente.

Os modelos são apresentados na Figura 3.

Diferentes projetos foram submetidos a esta chamada, que foram avaliados usando uma metodologia descrita brevemente a seguir.

3.1. Metodologia de avaliação

Como linha de base, um codec recente de nuvem de pontos de imagem *octree* foi usado para vídeo teleimersivo. As métricas de qualidade selecionadas foram.

- Métrica de distorção da geometria ponto a ponto.

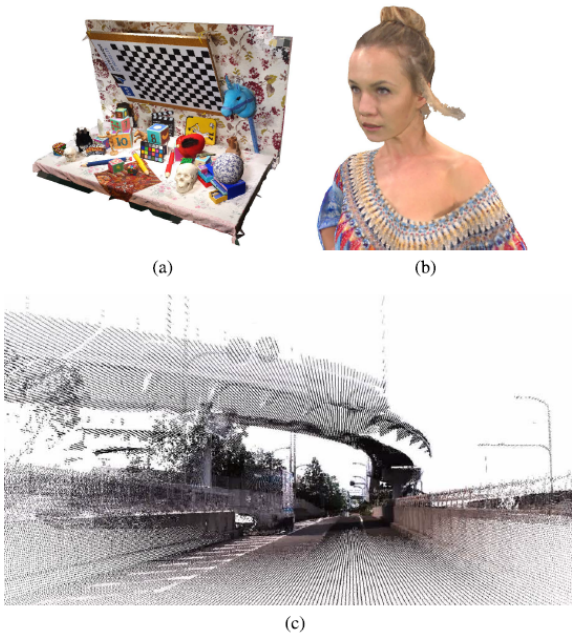

Figura 3. Exemplos de nuvens de pontos para as três categorias diferentes. (a) Estático. (b) Dinâmico (detalhe). (c) Adquirido dinamicamente (detalhe).[Schwarz et al. 2018]

- Métrica de distorção da geometria do ponto plano.

Para essas métricas, foi feito um esforço adicional para escolher taxas de bits de destino significativas. Também foi definida uma metodologia de avaliação subjetiva, onde foram utilizados 3 objetos estáticos e 3 cenas dinâmicas, enquanto 30 objetos foram utilizados para avaliação objetiva, dos quais 19 objetos estáticos, 5 objetos dinâmicos e 6 cenas adquiridas dinamicamente.

A avaliação subjetiva foi possibilitada por um renderizador de nuvem de pontos projetado pela Technicolor. A escala de medida varia de cinco graus de qualidade, com 1 (ruim) e 5 (excelente), duas ou três pessoas participaram do ambiente ao mesmo tempo, estavam sentadas a uma distância de 2 vezes a altura da televisão (55” 4K) houve um total de 22 participantes: 9 homens e 13 mulheres, com idades entre 20 e 30 anos [Schwarz et al. 2018].

Como resultado, três tecnologias diferentes foram escolhidas:

- LIDAR (L-PCC) - Dados adquiridos dinamicamente.
- Dados estáticos de superfície (S-PCC)
- Conteúdo dinâmico de vídeo (V-PCC)

3.2. Compressão de nuvem de pontos lidar L-PCC

O L-PCC foi projetado principalmente para nuvens de pontos que exibem amostras altamente irregulares. Por causa disso, o lidar primeiro comprime as informações geométricas usando uma estratégia de codificação baseada em *octree*, paralelo ele tem um módulo de transferência de tributo, que é o processo que ajuda a determinar os valores de tributo apropriados que devem ser associados às informações da geometria reconstruída. A geometria reconstruída e a informações geométrica compactada é usada para criar uma

estrutura de nível de detalhe (LoD), LoD permitirá a previsão hierárquica eficiente de atributos. Depois o módulo de previsão baseado em interpolação melhorará ainda mais a eficiência da codificação dos valores dos atributos, explorando correlações espaciais. Finalmente, a quantificação e desquantização são aplicadas aos resíduos [Tu et al. 2019] [Agarwal 2019]. A Figura 4 fornece uma visão geral dos processos de compressão e descompressão do L-PCC.

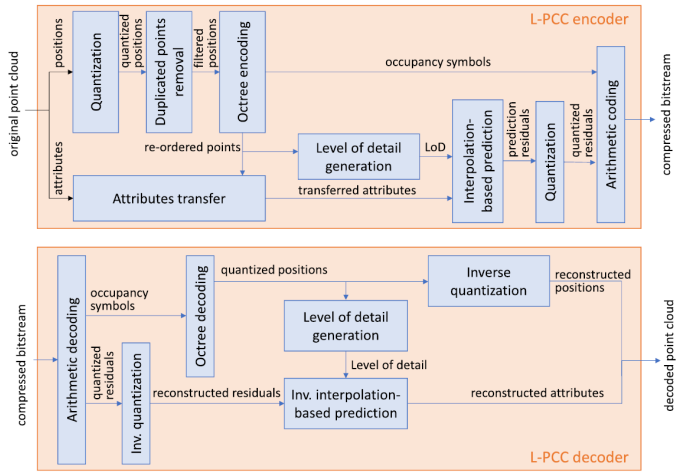

Figura 4. Visão geral do processo de compressão e descompressão L-PCC. [Schwarz et al. 2018]

3.3. Compressão de nuvem de pontos de superfície S-PCC

Esse modelo de compressão tem uma característica muito especial, diferentemente do L-PCC, a codificação geométrica é trabalhada separadamente, pois na compressão da nuvem de pontos Lidar, isso não pode ser feito, pois a geometria é adquirida dinamicamente e muda de acordo com o tempo. A arquitetura é apresentada na Figura 5.

Este modelo de compressão S-PCC possui uma característica muito especial, ao contrário do L-PCC, a geometria é trabalhada separadamente, pois ao trabalhar em uma superfície estática, a geometria não é adquirida dinamicamente e não muda com o tempo.

A arquitetura S-PCC compreende em um codificador e um decodificador, que por sua vez compreende vários módulos, como mostrado na Figura 5, a comunicação entre os módulos consiste em passar listas de localizações de pontos e atributos. Os parâmetros para o codificador S-PCC incluem o seguinte.

- *depth*: profundidade de codificação *octree*.
- *level*: nível *octree* para codificação geométrica.
- *geomstepsize*: quantização da geometria stepsize.
- *colorstepsize*: quantização da cor stepsize.
- *Mbpstarget*: objetivo da taxa de bits em Mbps.
- *fps*: quadros por segundo.
- *scale*:parâmetro de escala de quadro a mundo.
- *tradução*: tradução quadro a mundo.

Muitos desses parâmetros são passados para o decodificador no cabeçalho de fluxo de bits.

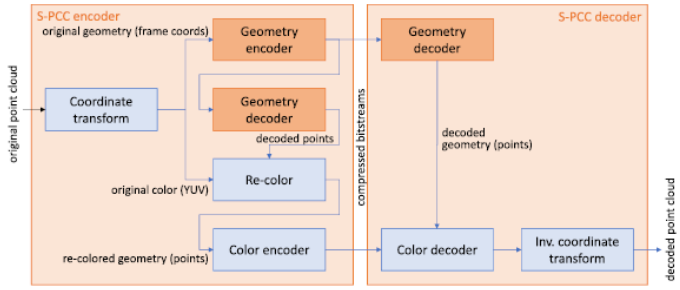

Figura 5. Diagrama de blocos do codificador e decodificador S-PCC [Schwarz et al. 2018].

O processo de codificação S-PCC consiste em receber a nuvem de pontos não compactada original; essa nuvem de pontos inclui a lista de localizações de pontos do mundo real $(x_i^{world}, y_i^{world}, z_i^{world}), i = 1, ..., N$, em que N é o número de pontos na nuvem, também recebe uma lista de atributos $(A_{1i}, A_{2i}, ..., A_{Di}), i = 1, ..., N$, em que D_i é o número de atributos para cada ponto. Por exemplo, se trabalharmos com os atributos de cores RGB, os atributos serão os três canais de cores (R_i, G_i, B_i) e, neste caso, $D_i = 3$.

As localizações dos pontos originais são transformadas de suas coordenadas originais (world) em coordenadas internas (quadro) usando a seguinte equação.

$$(x_i, y_i, z_i) = ((x_i^{world}, y_i^{world}, z_i^{world}) - (t_x, t_y, t_z))/s$$

Onde (t_x, t_y, t_z) são os parâmetros de conversão e $s = scale$ (parâmetro de escala quadro a mundo)

As cores são transformadas de (R_i, G_i, B_i) para (Y_i, U_i, V_i).

A arquitetura de compressão e descompressão da geometria tem uma arquitetura separada mostrada na figura 6.

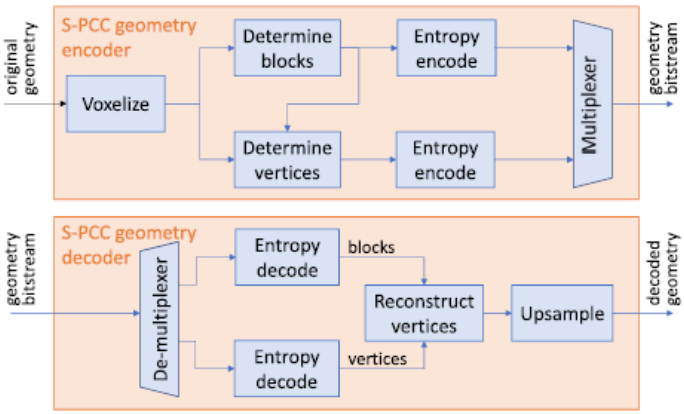

Figura 6. Diagrama de blocos do codificador e decodificador de geometria S-PCC [Schwarz et al. 2018].

Os detalhes do módulo codificador e decodificador de geometria consistem em ***voxelização***, que é o processo de agrupar pontos em voxels, voxels são conjuntos de cubos unitários. Após a ***determinação dos blocos***, consiste em dividir o cubo de voxels

em blocos de voxels $WxWxW$, onde $W = 2^{depth-l}$ e l = nível, esse parâmetro é usado no codificador e passado para o decodificador. Blocos contendo voxels ocupados serão utilizados na ***codificação da entropia*** para blocos, este processo consiste em codificar o conjunto de blocos usando *octrees*, onde cada folha de *octree* representa os blocos ocupados. O *octree* tem uma altura l = nível, então os blocos nas folhas têm voxels $W = 2^{depth-1}$ da largura do bloco em um lado, se l = level é igual à profundidade, então $W = 1$, o os blocos são $1x1x1$; o *octree* representa a coleção de voxels sem perdas; se l = level é menor que a profundidade, os blocos são $2x2x2$ ou maiores; o *octree* representa a coleção de voxels com possíveis perdas. S-PCC representa a geometria dentro de cada bloco como uma superfície que intercepta cada aresta do bloco no máximo uma vez; essas interseções são conhecidas como vértices que servirão para reconstruir um polígono não plano dentro do bloco no momento da reconstrução. Esses polígonos não planos podem ser como uma coleção de triângulos. O S-PCC usa triângulos, pois são particularmente amigáveis ao processamento gráfico padrão. Finalmente, a ***codificação da entropia do vértice***, O conjunto de vértices é codificado em duas etapas: primeiro, todas as arestas únicas dos blocos ocupados são calculadas, segundo para cada aresta que contém um vértice,a posição do vértice ao longo da aresta é uniformemente quantizado, depois o vetor de bits e as posições dos vértices são ainda mais compressãs por um codificador de entropia.

A saída do decodificador de geometria é uma lista de vértices refinados $(\hat{x}_r, \hat{y}_r, \hat{z}_r), r = 1, ...N_{ref}$

A recoloração do S-PCC é implementada colorindo cada vértice refinado $(\hat{x}_r, \hat{y}_r, \hat{z}_r)$ com a cor $(\tilde{Y}_r, \tilde{U}_r, \tilde{V}_r) = (Y_{ir}, U_{ir}, V_{ir})$ do ponto de entrada (x_{ir}, y_{ir}, z_{ir}) mais próximo de $(\hat{x}_r, \hat{y}_r, \hat{z}_r)$ na distância euclidiana, ou seja.

$$i_r\text{=arg } min_i((\hat{x}_r - x_i)^2 + (\hat{y}_r - y_i)^2 + (\hat{z}_r - z_i)^2)$$

Portanto, a saída do módulo de recoloração é a lista de cores $(\tilde{Y}_r, \tilde{U}_r, \tilde{V}_r), r = 1, ..., Nref$, correspondente aos vértices refinados $(\hat{x}_r, \hat{y}_r, \hat{z}_r), r = 1, ..., N_{ref}$.

Os detalhes do módulo de código de cores são mostrados na Figura 7.

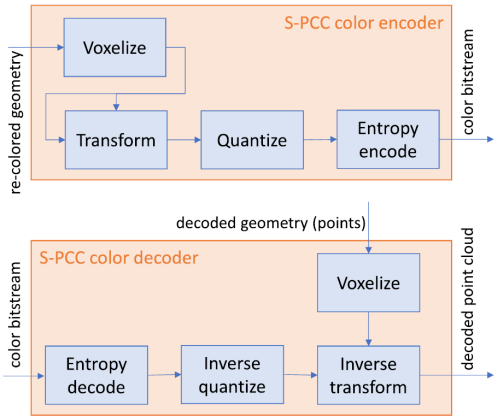

Figura 7. Diagrama de blocos do codificador e decodificador de cores S-PCC [Schwarz et al. 2018].

A saída do codificador S-PCC e a entrada para o decodificador é um fluxo de bits que compreende um fluxo de bits de geometria, um fluxo de bits de cor e um cabeçalho

de fluxo de bits. O cabeçalho do fluxo de bits contém os parâmetros necessários para decodificar os fluxos de bits de geometria e cor.

3.4. Compressão de nuvem de pontos baseada em vídeo V-PCC

A filosofia do V-PCC é aproveitar os codecs de vídeo existentes, como MPEG-4, AVC, HEVC, AV, etc. O processo envolve a compressão das informações de geometria e textura de uma nuvem de pontos dinâmica em duas seqüências de vídeo 2D, os metadados adicionais para interpretar as duas sequências de vídeo também são gerados e compressas separadamente.

As arquiteturas do processo de compressão e desccompressão podem ser vistas nas Figuras 8 e 9 respectivamente, Os processos são descritos a seguir.

Primeiro, o processo de ***geração de patches*** determina a melhor maneira de decompor a nuvem de pontos em patches, e o processo de ***empacotamento de patches*** determinará como encaixá-los eficientemente em uma grade 2D retangular. Depois, o processo de ***criação de imagens e o processo de preenchimento*** transformarão as informações de geometria e textura da nuvem de pontos em imagens 2D suaves e temporariamente correlacionadas, adequadas para codificação usando codecs de vídeo tradicionais. Em seguida, as ***informações auxiliares dos patches e blocos são compressas***, para poder reconstruir a nuvem de pontos 3D a partir das imagens de geometria e textura. As informações auxiliares também serão usadas para a ***compressão do mapa de ocupação***; esse processo consiste em a compressão de um mapa binário que indica para cada célula da grade se ela pertence ao espaço vazio ou à nuvem de pontos. Finalmente, o ***processo de suavização*** cuidará de aliviar quaisquer descontinuidades possíveis que possam surgir no limite do patch devido a os artefatos de compressão, e o ***processo de reconstrução da geometria*** da nuvem de pontos explorará as informações do mapa de ocupação para detectar pixels não vazios nas imagens/geometria/textura. As posições 3D dos pontos associados a esses pixels são calculadas usando as informações do bloco/patch auxiliar e as imagens de geométria.

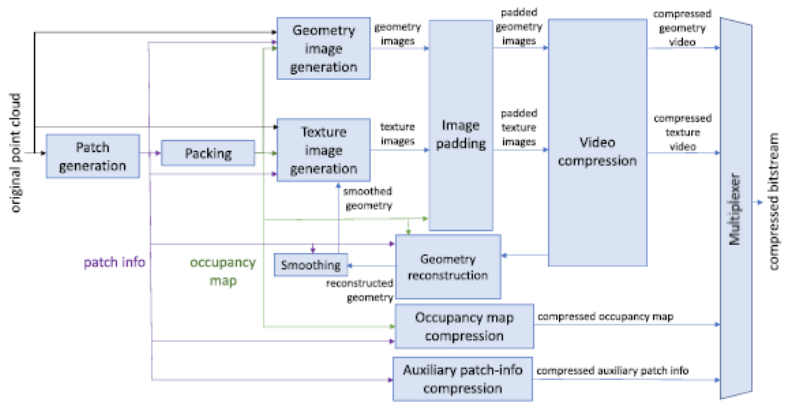

Figura 8. Visão geral do processo de codificação V-PCC [Schwarz et al. 2018].

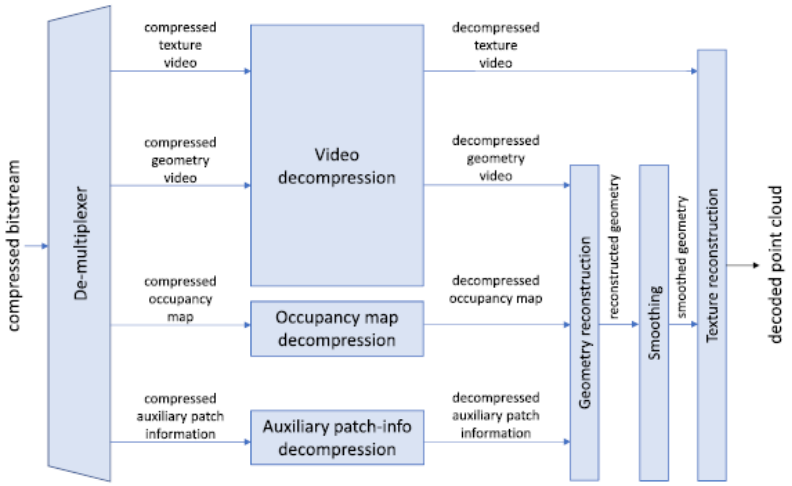

Figura 9. Visão geral do processo de decodificação V-PCC. [Schwarz et al. 2018].

4. Pesquisas recentes

4.1. Melhorias de V-PCC com base na introdução de conceitos volumétricos

Esta pesquisa apresenta uma abordagem volumétrica para a compressão de nuvens de pontos, que pode ser usada para compressão de geometria e atributos [Chou et al. 2019],[Krivokuća et al. 2019].

Para representar os atributos, usam uma função volumétrica com valor vetorial que possui a mesma dimensão que os atributos, e os atributos são reconstruídos como os valores da função volumétrica decodificada nos pontos da geometria decodificada.

4.1.1. Processo

Começa com um conjunto de locais de pontos $x_1; ...; x_n$ e um conjunto de atributos correspondentes $f_1; \ldots; f_n$. O problema da compressão de atributos é reproduzir atributos aproximados, $\hat{f}_1; \ldots; \hat{f}_n$, no decodificador, sujeito a uma restrição no número de bits comunicados, dadas as localizações dos pontos como informação adicional [Chou et al. 2019].

No codificador, um B-spline volumétrico da ordem p é definido com os valores $f_1; ...; f_n$ em locais $x_1; ...; x_n$, e seus coeficientes de wavelet são quantificados e codificados por entropia.

No decodificador, os coeficientes da onda são decodificados e entropia quantizada, e o B-spline volumétrico é reconstruído.

Finalmente, a spline B volumétrica reconstruída é amostrada nos locais $x_1; ...; x_n$ e os valores correspondentes $\hat{f}_1; ...; \hat{f}_n$ são usados como reproduções dos atributos.

Um recurso adicional é que o B-spline volumétrico reconstruído também pode ser amostrado em outro local para interpolar continuamente atributos, um recurso exclusivo de abordagem de [Chou et al. 2019].

4.1.2. Resultados

As avaliações foram realizadas nos conjuntos de dados listados na Tabela 1. Esses conjuntos de dados são aqueles usados na atividade de padronização de codificação de nuvem de pontos MPEG.

Dataset name	Original resolution	Original point count	10-bit point count	7-bit point count
loot_vox10_1200	10 bit	805285	805285	14711
queen_0200	10 bit	1000993	1000993	14683
soldier_vox10_0690	10 bit	1089091	1089091	19993
shiva_00035_vox12	12 bit	1009132	900662	30045
longdress_vox10_1300	10 bit	857966	857966	15688

Tabela 1. DATASETS USADOS PARA REPRESENTAÇÃO E COMPRESSÃO ATRIBUTOS [Chou et al. 2019]

Para experimentos, todos os conjuntos de dados originais são vocalizados primeiro na resolução de 10 ou 7 bits.

A Figura 10 mostra os resultados qualitativos da suavização dos dados de loot de 10 bits configurando os coeficientes de wavelet G' para zero para todos os níveis, onde L = 4 (esquerda) a L = 8 (direita), para p = 1 (RAHT, acima) ep = 2 (BV, abaixo).

Figura 10. Suavizando o conjunto de dados para p = 1 (RAHT, superior) e p = 2 (BV, inferior), nos níveis 4 (esquerda), 5, 6, 7 e 8 (direita) [Chou et al. 2019]

Como é observado, os artefatos de blocos são claramente visíveis para p = 1

(RAHT, acima), enquanto para BV, os artefatos de blocos são muito menos visíveis porque as cores são garantidas para serem contínuas entre os blocos.

A Figura 11 mostra os resultados da suavização quantitativamente. Cada gráfico é um gráfico de Y (luma) PSNR em função do número de coeficientes wavelet diferentes de zero p = 1 (RAHT, linha tracejada vermelha com quadrados) ep = 2 (BV, linha sólida azul com círculos). A diferença entre as linhas indica a melhoria da BV sobre RAHT no ganho de compressão de energia ou ganho de codificação de transformação, aproximadamente 3-4 dB.

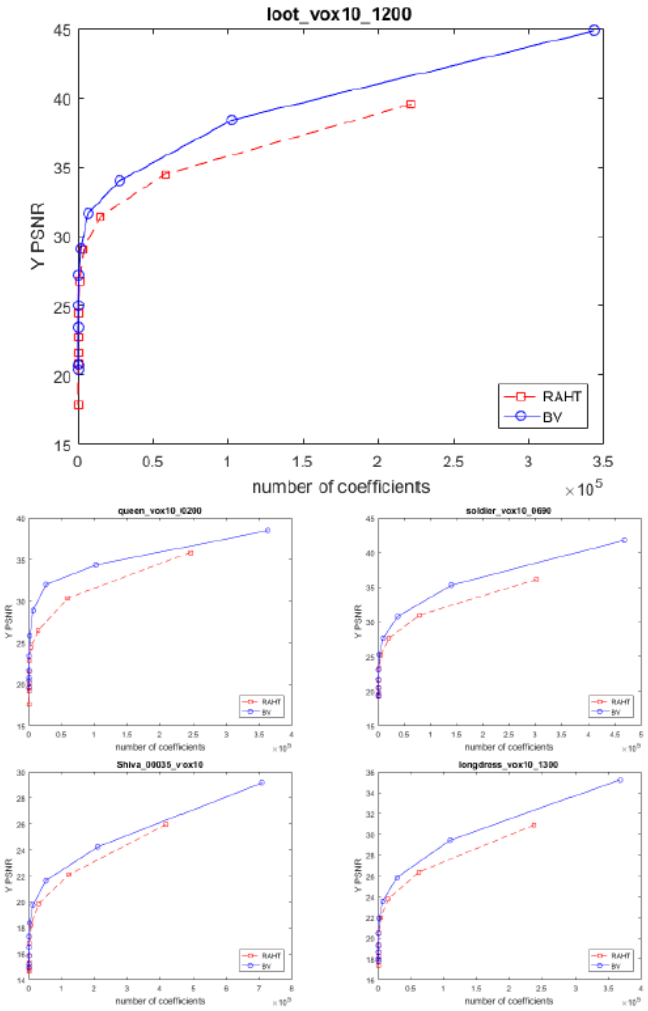

Figura 11. Compressão de energia para p = 2 (BV) vs p = 1 (RAHT) [Chou et al. 2019]

A Figura 12 mostra como a melhoria no ganho de compressão de energia se traduz em ganho de codificação real, em todos os cinco conjuntos de dados de 7 bits. Cada gráfico é um gráfico de Y PSNR com base no total de bits de cores (Y + U + V) por voxel de entrada.

Os conjuntos de dados com texturas de cores mais suaves (loot e queen) mostram ganhos maiores que os conjuntos de dados com texturas de cores médias (soldier). Enquanto em (shiva e longdress) demonstram pouco ganho de codificação BV em relação

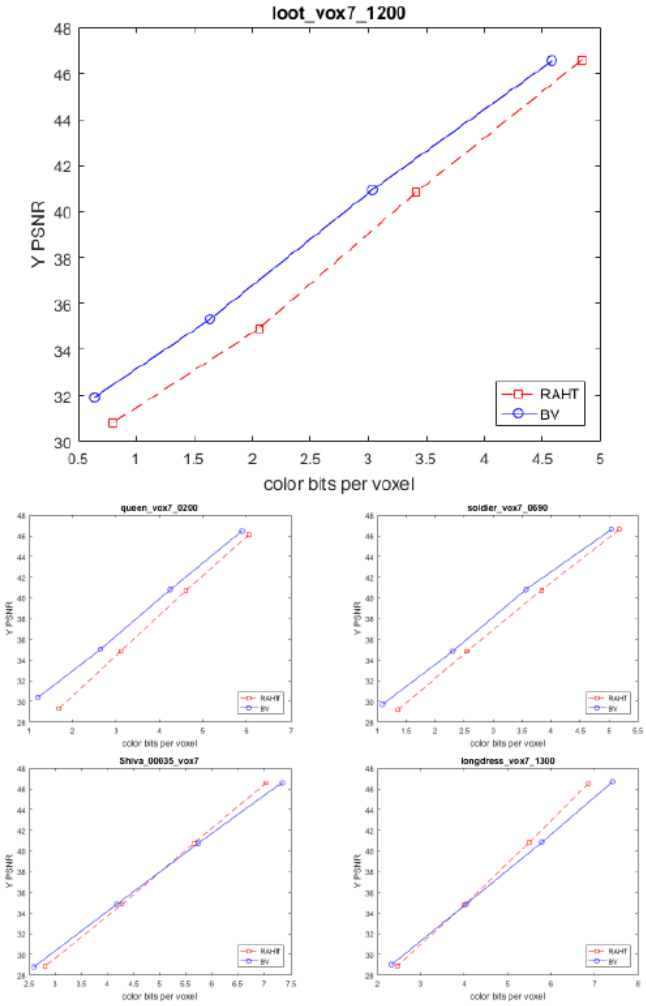

Figura 12. Desempenho da taxa de distorção para p = 2 (BV) vs p = 1 (RAHT) [Chou et al. 2019]

ao RAHT, apesar de terem um ganho significativo na compressão de energia. O motivo provavelmente tem a ver com um código de entropia incompatível.

4.2. Partição de nuvem de pontos 3D baseada em árvores binárias

O modelo de partição de nuvem de pontos 3D baseada em árvores binárias é baseado no software do grupo MPEG PCC ad-hoc, que baseia sua compressão geométrica no uso de octree e a compressão de coordenadas a reconstrói no decodificador [Schwarz et al. 2018]. Supõe-se que a codificação da geometria seja feita separadamente e os atributos possam ser codificados como sinais gráficos na geometria reconstruída.

O método de otimização realizado é incorporado principalmente no particionamento de blocos de transformação, na otimização da dispersão laplaciana para a transformação de gráficos e nas soluções Lagrangianas para a seleção do modo de quantificação [Shao et al. 2017].

O método é dividido em 3 partes, conforme as seções a seguir.

4.2.1. Partição da nuvem de pontos através da árvore k-d

Uma árvore k-d é uma árvore de partição de dados binários para organizar pontos no espaço k-dimensional [Gu et al. 2020], [Shao et al. 2017].

Como mencionado em trabalhos anteriores, o octree foi usado para obter blocos de transformação e eles construíram um gráfico para cada bloco, fazendo com que o número de pontos por bloco variasse muito, até casos foram encontrados onde um bloco não continha pontos como mostrado na Figura 13.

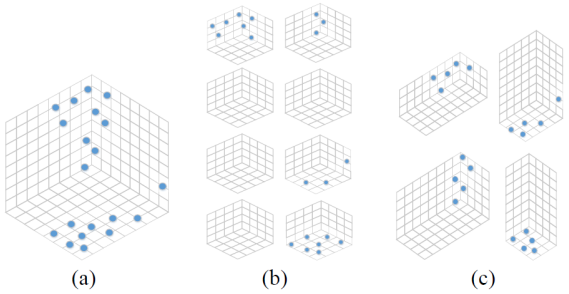

Figura 13. (a) Exemplo de uma nuvem de pontos dispersos. (b) partição com octree. (c) partição com árvore K-d [Shao et al. 2017].

A principal vantagem de uma árvore k-d sobre octree é que ela possui uma estrutura hierárquica de blocos com um número aproximado de pontos idênticos que podem ser usados na codificação de transformação subsequente [Gu et al. 2020]. A árvore k-d impede a criação de blocos e subcharts vazios, o que é significativo para uma transformação adicional do gráfico.

4.2.2. Otimização da dispersão do Laplaciano para a transformação de gráficos

Após particionar a árvore k-d na nuvem de pontos, forma-se um gráfo conectando todos os pontos com arestas em cada bloco de transformação. Um gráfo simples é construído como na Figura 14(a). Defina o gráfo como $G = (v = \{n_1, n_2, n_3, n_4, n_5\}, \epsilon)$, onde n_i são os pontos do bloco e ϵ representa os conjuntos de arestas.

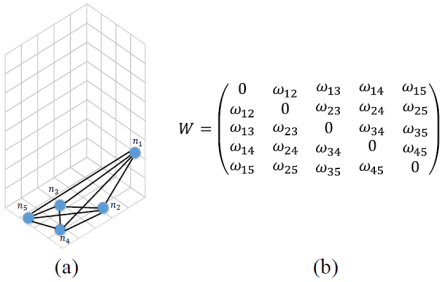

Figura 14. (a) Exemplo de gráfico em um bloco de transformação. (b) Uma matriz de adjacência para o gráfico [Shao et al. 2017].

Na Figura 14, pode-se ver a matriz W, que é composta pelos pesos das arestas dos pontos n_i a n_j, após esse cálculo, encontramos a matriz laplaciana usando a fórmula a seguir.

$$L = S - W$$

S é uma matriz diagonal $S = diag(s_1, ..., s_5)$ cujos elementos são a soma de cada linha da matriz W. Escolhemos a matriz laplaciana L do gráfico como o operador de deslocamento do gráfico e obtemos a decomposição adequada de L.

$$L = AA^1$$

4.2.3. Otimização da taxa de distorção pelo método Lagrangiano

Para resolver o problema de compensação entre taxa de bits e distorção, para obter o modo ideal de quantização do capacete convexo, é aplicado o método Lagrangiano padrão, amplamente aceito na codificação de vídeo.

4.2.4. Resultados

Como resultado, é apresentada a Figura 15, que é a (a) comparação da variação residual da transformação entre o método proposto e o DCT PCC no [Schwarz et al. 2018]. Ela envolve a comparação da compressão de entropia após a transformação. (b) Comparação da eficiência de transformação entre o método proposto e o DCT com 20 dimensões mantidas em 50 blocos de transformação do teste 1.

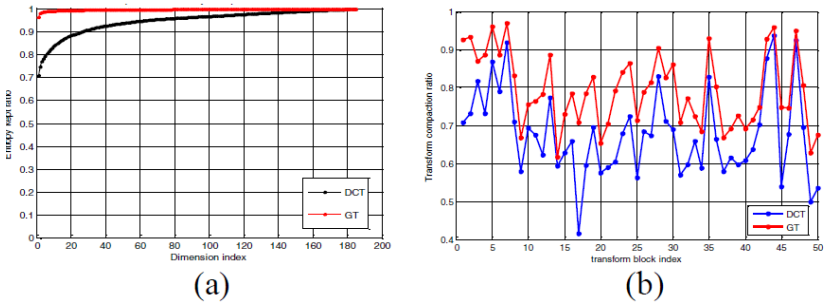

Figura 15. (a) Comparação da transformação residual (b) Comparação da eficiência da transformação [Shao et al. 2017].

Os quadros 8i a seguir foram extraídos ”longdress_vox10_1051.ply”, ”redandblack_vox10_1000.ply”, ”loot_vox10_1451.ply”e ”soldier_vox10_0537.ply”, que possuem o número $857966, 757691, 805285$ e 1089091 pontos, respectivamente. Foram usadas quatro caixas de teste, mostradas na Tabela 2. Os atributos RGB foram convertidos no espaço de cores YCbCr e usou-se a luminância Y como o atributo a ser codificado.

Por exemplo, para a métrica Y-PSNR nos testes em torno de 36dB, o método baseado em DCT usando PCC [Schwarz et al. 2018] precisa de cerca de 1,99 bits por ponto (bpp), enquanto o codificador de [Shao et al. 2017] requer menos de 0,46 bpp. O PCC(DCT-based) é mais de 4 vezes maior que o codificado proposto.

4.3. Previsão avançada de movimento 3D para compressão de nuvem de pontos dinâmica baseada em vídeo

O estudo apresentado em [Li et al. 2019] propõe um modelo geral que utiliza o movimento 3D e o mapeamento 3D-2D para calcular o vetor de movimento 2D (MV). Sob o

Testing Datasets	Proposed Method		PCC(DCT-based)	
	Y-PSNR (dB)	Bitrate (bpp)	Y-PSNR (dB)	Bitrate (bpp)
1. ***longdress_vox10_1051***	33.65	0.89	32.56	2.76
2. ***loot_vox10_1000***	35.18	0.29	37.81	1.41
3. ***redandblack_vox10_1451***	34.38	0.55	36.1	1.54
4. ***soldier_vox10_0537***	36.02	0.46	35.71	1.99

Tabela 2. Comparação do desempenho R-D (taxa de distorção) em quatro conjuntos de dados [Shao et al. 2017].

V-PCC, eles propõem um método baseado em geometria que usa geometria reconstruída em 3D que requer que o vídeo em geometria 2D estime o MV 2D no vídeo de atributo 2D [Li et al. 2019].

A arquitetura apresentada é descrita em quatro modelos, primeiro é apresentado um modelo de movimento 3D para 2D e, em seguida, o uso das informações 3D iniciais (precisas) para prever a MV 2D (movimentos) no vídeo de atributos, o segundo método usa as informações 3D auxiliar para prever o MV 2D, tanto na geometria e vídeos atributos. Finalmente, eles propõem soluções normativas e não normativas para usar movimentos 2D derivados para melhorar a eficiência da compressão.

4.3.1. Modelo de movimento geral 3D para 2D

Na Figura 16 $(x_c; y_c)$ e $(x_r; y_r)$ estão as coordenadas dos pixels centrais da unidade de predição PU atual e do bloco de referência e serão usadas para representá-los, respectivamente. $(x3_c; y3_c; z3_c)$ e $(x3_r; y3_r; z3_r)$ são as coordenadas 3D correspondentes.

$$(x_c, y_c) = f(x3_c, y3_c, z3_c)$$
$$(x_r, y_r) = g(x3_R, y3_r, z3_r)$$

Como o movimento 3D da coordenada 3D atual $(x3_c; y3_c; z3_c)$ é conhecido, pode-se derivar facilmente a coordenada 3D do pixel de referência $(x3_r; y3_r; z3_r)$. O MV atual de PU (MV_c) pode ser derivado como.

$$MV_c = g(x3_r, y3_r, z3_r) - f(x3_c, y3_c, z3_c)$$

Este modelo pode ser usado na compressão de conteúdo de jogos em 3D, pois sabemos o movimento em 3D.

4.3.2. Previsão de movimento com base na geometria

O movimento 3D real não é conhecido neste modelo, o que significa que o valor da coordenada 3D posterior $(x3_r; y3_r; z3_r)$ não pode ser determinado. No entanto, pode-se supor que o quadro atual e o quadro de referência não mudaram muito. Portanto, um vetor de movimento estimado MVE pode ser derivado para a PU atual da seguinte maneira.

$$MVE_c = g(x3_c, y3_c, z3_c) - f(x3_c, y3_c, z3_c)$$

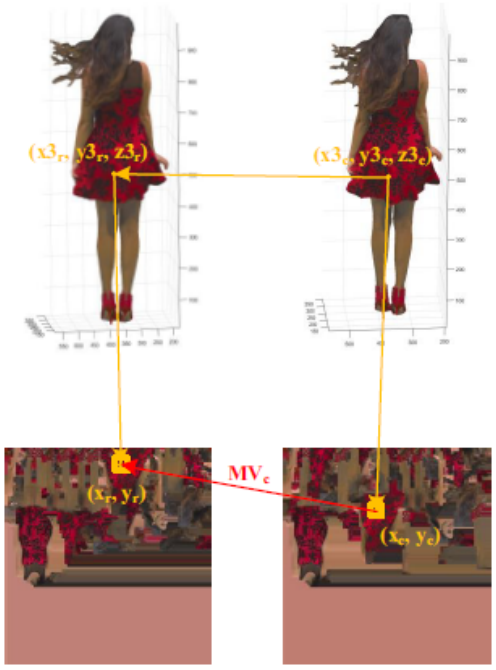

Figura 16. Modelo de movimento geral 3D para 2D [Li et al. 2019].

No entanto, $(x3_c, y3_c, z3_c)$ pode não ter um ponto 2D correspondente no quadro de referência, o que significa que a coordenada 3D assumida $g(x3_c, y3_c, z3_c)$ é inválida, o que faria com que não tivesse um ponto 2D correspondente. Para resolver essa possibilidade, propõe-se pesquisar no quadro de referência 2D para encontrar o bloco com a coordenada 3D mais próxima.

$$min(x3_c - x3_r)^2 + (y3_c - y3_r)^2 + (z3_c - z3_r)^2$$

A Figura 17 mostra uma ilustração desse processo. Para reduzir parcialmente a complexidade da pesquisa, propõe-se usar as informações auxiliares para eliminar alguns dos pixels candidatos.

4.3.3. Previsão de movimento com base em informações auxiliares

A predição de movimento baseada em geometria tem principalmente dois problemas. Primeiro, só pode melhorar o desempenho da compressão do vídeo de atributo. Segundo, a procura durante a predição de movimento aumentará significativamente a complexidade do codificador e decodificador.

Consequentemente, o trabalho [Li et al. 2019] propõe um algoritmo de predição de movimento baseado em informações auxiliares que usa geometria grossa para resolver esses dois problemas.

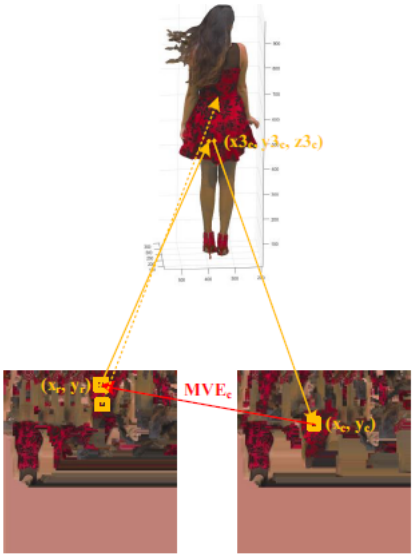

Figura 17. Predição de movimento com base na geometria [Li et al. 2019].

4.3.4. Resultados

A intenção dos algoritmos propostos é melhorar a eficiência de compressão da predição inter quadros.

As variáveis avaliadas foram determinadas, erro ponto a ponto (D1) como erro ponto a plano (D2) para distorções geométricas. Para o atributo, são fornecidas as distorções dos componentes Luma, Cb e Cr. O software V-PCC também foi alterado um pouco para determinar a complexidade relativa de cada um para o codificador e decodificador.

Test	Geom.BD-GeomRate		Attr.BD-AttrRate			Geom.BD-TotalRate		Attr.BD-TotalRate		
point cloud	D1	D2	Luma	Cb	Cr	D1	D2	Luma	Cb	Cr
Loot	0.0%	0.0%	–18.1%	–31.4%	–30.4%	–3.4%	–6.1%	–8.4%	–17.7%	–16.9%
RedAndBlack	0.0%	0.0%	–16.3%	–25.0%	–15.9%	–4.6%	–4.6%	–8.8%	–15.4%	–8.4%
Solider	0.0%	0.0%	–33.4%	–42.5%	–43.2%	–8.2%	–8.2%	–17.2%	–26.3%	–27.0%
Queen	0.0%	0.0%	–13.7%	–20.5%	–19.2%	–3.5%	–3.6%	–7.8%	–12.7%	–11.6%
LongDress	0.0%	0.0%	–9.8%	–13.5%	–12.3%	–3.7%	–3.7%	–6.4%	–9.5%	–8.4%
Avg.	0.0%	0.0%	–18.2%	–26.6%	–24.2%	–4.7%	–4.7%	–9.7%	–16.3%	–14.5%
Enc. time self	97%									
Dec. time self	98%									
Enc. time child	486%									
Dec. time child	337%									

Tabela 3. Desempenho da previsão de movimento com base em geometria comparada com V-PCC anchor [Schwarz et al. 2018] [Li et al. 2019].

A Tabela 3 mostra as complexidades de desempenho e codificação/decodificação do algoritmo de predição de movimento baseado em geometria proposto comparado à âncora V-PCC. O algoritmo de previsão de movimento baseado em geometria proposto pode fornecer uma melhoria média de desempenho de $18.2\%, 26.6\%$ e 24.2% para os componentes Y, Cb, Cr, respectivamente.

Além disso, o algoritmo de previsão de movimento baseado em geometria não levará a melhorias de desempenho para geometria. Além disso, aumentará a complexidade da codificação em 5x e aumentará a complexidade da decodificação em 3x, respectiva-

mente, em comparação com a âncora do V-PCC.

Test	Geom.BD-GeomRate		Attr.BD-AttrRate			Geom.BD-TotalRate		Attr.BD-TotalRate		
point cloud	D1	D2	Luma	Cb	Cr	D1	D2	Luma	Cb	Cr
Loot	–4.0%	–3.9%	–16.3%	–26.4%	–28.5%	–6.3%	–6.2%	–9.6%	–16.7%	–17.9%
RedAndBlack	–1.0%	–1.1%	–12.2%	–18.9%	–10.9%	–4.0%	–4.1%	–7.2%	–12.1%	–6.2%
Solider	–8.0%	–7.9%	–31.3%	–41.4%	–40.4%	–13.6%	–13.4%	–19.8%	–28.7%	–28.1%
Queen	–5.9%	–5.9%	–11.8%	–17.0%	–15.7%	–7.3%	–7.3%	–9.1%	–12.9%	–11.8%
LongDress	–1.1%	–1.1%	–8.3%	–11.2%	–10.2%	–3.8%	–3.6%	–5.7%	–8.2%	–7.3%
Avg.	–4.0%	–4.0%	–16.0%	–23.0%	–21.1%	–7.0%	–6.9%	–10.3%	–15.7%	–14.3%
Enc. time self	100%									
Dec. time self	100%									
Enc. time child	98%									
Dec. time child	99%									

Tabela 4. Desempenho da previsão de movimento com base em informações auxiliares comparadas com V-PCC Anchor[Schwarz et al. 2018] sob a solução regulatória. [Li et al. 2019]

Em relação à taxa de bits da geometria como a Tabela 4 mostra, o algoritmo proposto alcança uma economia média de 4.0% nas taxas de distorção D1 e D2. Em relação à taxa de bits do atributo, o algoritmo proposto atinge uma média de $16.0\%, 23.0\%$ e 21.1% para os componentes Luma, Cb e Cr.

5. Considerações Finais

Foi observado que, após a investigação inical do MPEG PCC, existem muitas investigações, das quais uma melhoria substancial é apreciada em muitos aspectos.

Também se pode concluir que o PCC é uma área complexa de pesquisa, uma vez que os dados com os quais trabalha são variados e dependem principalmente um do outro.

Na investigação base, foi visto que seus modelos funcionam com *octree*, agora podemos dizer que *octree*, apesar de ser uma boa técnica, pode ser superado por árvores binárias k-d.

Pode-se observar que em algumas investigações eles tinham um foco fora do já definido, um exemplo claro é que, na previsão de movimentos baseados na geometria, os autores compararam os resultados de como o método deles afeta ao comprimir e descomprimir o vídeo 3D, isso é muito importante se você deseja trabalhar com dispositivos móveis.

Referências

Agarwal, S. (2019). Lidar point cloud compression. US Patent App. 15/837,481.

Cao, C., Preda, M., and Zaharia, T. (2019). 3d point cloud compression: A survey. In *The 24th International Conference on 3D Web Technology*, pages 1–9.

Chou, P. A., Koroteev, M., and Krivokuća, M. (2019). A volumetric approach to point cloud compression, part i: Attribute compression. *IEEE Transactions on Image Processing*.

Gu, S., Hou, J., Zeng, H., and Yuan, H. (2020). 3d point cloud attribute compression via graph prediction. *IEEE Signal Processing Letters*.

Krivokuća, M., Chou, P. A., and Koroteev, M. (2019). A volumetric approach to point cloud compression, part ii: Geometry compression. *IEEE Transactions on Image Processing*.

Li, L., Li, Z., Zakharchenko, V., Chen, J., and Li, H. (2019). Advanced 3d motion prediction for video-based dynamic point cloud compression. *IEEE Transactions on Image Processing*, 29:289–302.

Schwarz, S., Preda, M., Baroncini, V., Budagavi, M., Cesar, P., Chou, P. A., Cohen, R. A., Krivokuća, M., Lasserre, S., Li, Z., et al. (2018). Emerging mpeg standards for point cloud compression. *IEEE Journal on Emerging and Selected Topics in Circuits and Systems*, 9(1):133–148.

Shao, Y., Zhang, Z., Li, Z., Fan, K., and Li, G. (2017). Attribute compression of 3d point clouds using laplacian sparsity optimized graph transform. In *2017 IEEE Visual Communications and Image Processing (VCIP)*, pages 1–4. IEEE.

Tian, D., Ochimizu, H., Feng, C., Cohen, R., and Vetro, A. (2017). Geometric distortion metrics for point cloud compression. In *2017 IEEE International Conference on Image Processing (ICIP)*, pages 3460–3464. IEEE.

Tu, C., Takeuchi, E., Carballo, A., and Takeda, K. (2019). Real-time streaming point cloud compression for 3d lidar sensor using u-net. *IEEE Access*, 7:113616–113625.