Attribute-Specific Compression Techniques for 3D Gaussian Point Cloud Sequences

上海科技大学 ShanghaiTech University

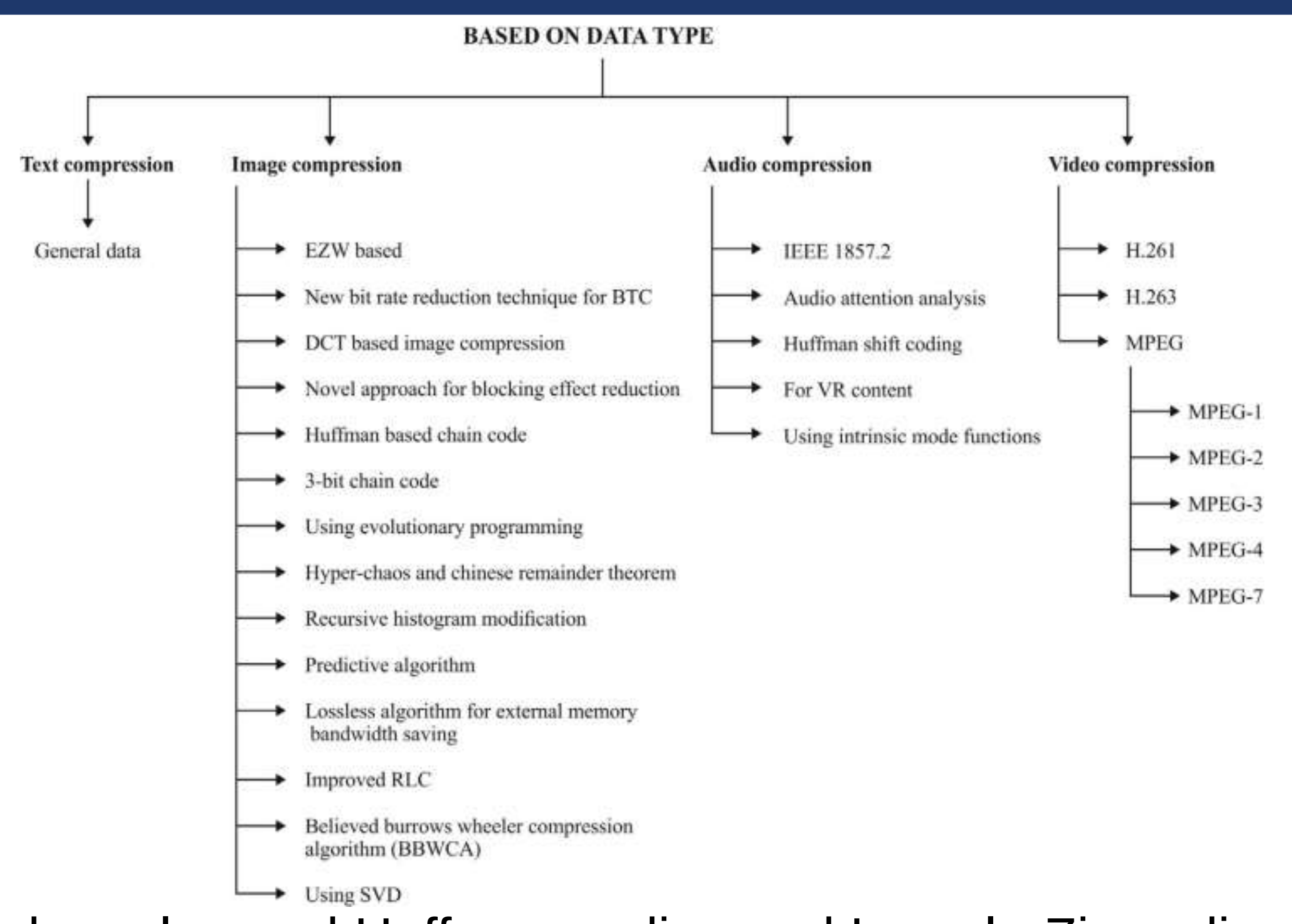
Zhehao Shen[1] Shouchen Zhou[1]

[1]ShanghaiTech University

Contributions

- We reviewed various compression methods for image, video, audio, and text formats and explored the principles behind RANS and WebP compression algorithms.
- We designed an efficient, attribute-specific compression method tailored for volumetric video representation (Gaussian point cloud sequences).
- Integrated the decompression methods into a mobile platform, developing a player that supports the rendering of long sequence volumetric videos on mobile devices.

Review


BASED ON DATA TYPE
Text compression
General data
Image compression
EZW based
New bit rate reduction technique for BTC
DCT based image compression
Novel approach for blocking effect reduction
Huffman based chain code
3-bit chain code
Using evolutionary programming
Hyper-chaos and chinese remainder theorem
Recursive histogram modification
Predictive algorithm
Lossless algorithm for external memory bandwidth saving
Improved RLC
Believed burrows wheeler compression algorithm (BBWCA)
Using SVD
Audio compression
IEEE 1857.2
Audio attention analysis
Huffman shift coding
For VR content
Using intrinsic mode functions
Video compression
H.261
H.263
MPEG
MPEG-1
MPEG-2
MPEG-3
MPEG-4
MPEG-7

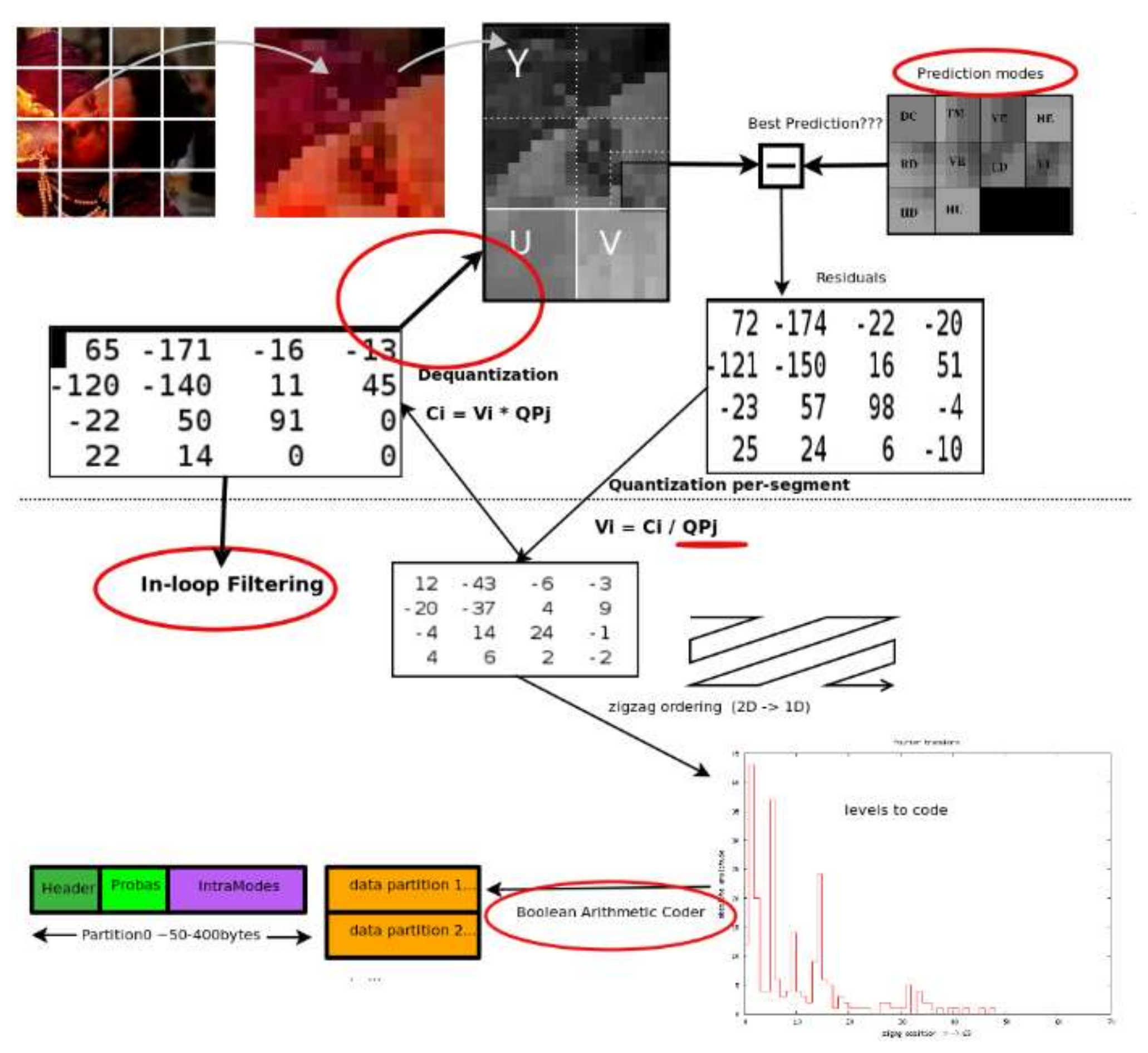
We have learned Huffman coding and Lemple-Ziv coding during the classes. To gain a comprehensive understanding of various data compression techniques and their applications, we read the review. Compression algorithms are categorized based on the type of data they process, including text, image, audio, and video data.

RANS Encoding

- Ranges Asymmetric Numeral Systems (RANS) is an entropy encoding method used in data compression developed by Jaroslaw Duda. It is an encoding method that approaches the theoretical value of entropy coding infinitely, and its essential operation is to use a decimal of [0,1) to represent the final encoding result.
- Huffman coding maps individual symbols to codewords based on their probabilities and is optimal for independent symbols under the prefix-free constraint. In contrast, RANS coding maps sequences of symbols to a single codeword, leveraging the overall probability distribution of the sequence. As a result, RANS achieves higher compression efficiency and approaches the theoretical entropy limit.
- Encoder: $C(x_i, s_{i+1}) = \left\lfloor \frac{x_i}{f_i} \right\rfloor \cdot 2^n + \sum_{k=0}^{s-1} f_k + (x_i \bmod f_s)$
- Decoder: $s_{i+1} = \min\ s : x < \sum_{k=0}^{s} f_k$

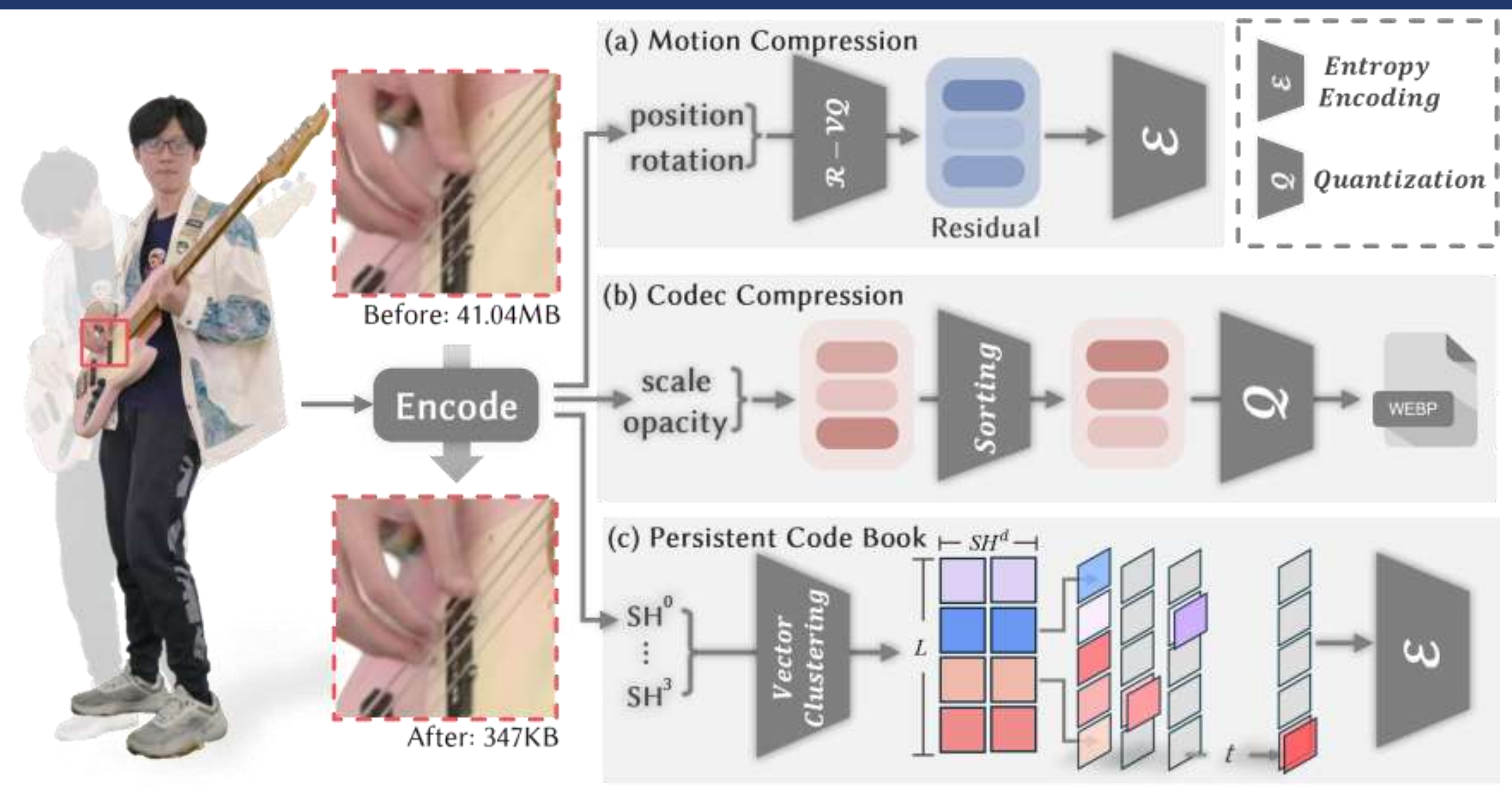
$$x_i = D(x_{x+1}) = f_s \cdot \left\lfloor \frac{x_{i+1}}{2^n} \right\rfloor - \sum_{k=0}^{s-1} f_k - (x_{i+1} \bmod 2^n)$$

WebP Compression



The steps involved in WebP lossy compression. The differentiating features compared to JPEG are circled in red. The major steps are: MacroBlocking, prediction, DCT, quantization, and arithmetic entropy encoding.

Method



Based on this, we designed an attribute-specific compression technique:

- **For motion data**, we apply residual quantization and RANS entropy encoding
- **For scale and opacity**, we construct 2D LUT table and use WebP compression
- **For Spherical harmonics**, we use persistent codebook to compress.

Component: Motion Compression

Following the entropy coding strategy of HiFi4G, we implemented a residual-based quantization compression method for position and rotation attributes. We further employ Ranged Arithmetic Numerical System(RANS) encoding for lossless compression.

Component: Codec Compression

Residual compression of opacity and scaling` parameters is limited by the large number of Gaussians, which increases storage demands. To balance data accuracy with storage efficiency, we leverage the spatiotemporal relationships between these attributes. Using a temporal regularizer, we organize the opacity and scaling data into separate Look-Up Tables (LUTs). We construct a 2D LUT where the height corresponds to the number of Gaussian point cloud sequences, and the width to the segment frame length. To enhance consistency, the rows are sorted by their average values. This transforms the data into an image-like format, enabling the use of image compression algorithms like WebP or JPEG for efficient storage.

Component: Persistent Codebook

We propose a novel compression strategy: a persistent codebook that leverages the temporal consistency of Gaussian sequences' SH parameters, achieving up to 360-fold compression. Specifically, we apply K-Means clustering to the d-order (d=0, 1, 2, 3) SH coefficients across all frames in a segment. The codebook $\mathcal{Z}_d$ is initialized with a uniform distribution and iteratively updated by randomly sampling batches of d-order coefficients. After optimization, we obtain four codebooks, each with a length L (set to 8192 in our implementation).Using these codebooks, the SH attributes are compactly encoded into SH indices as follows:

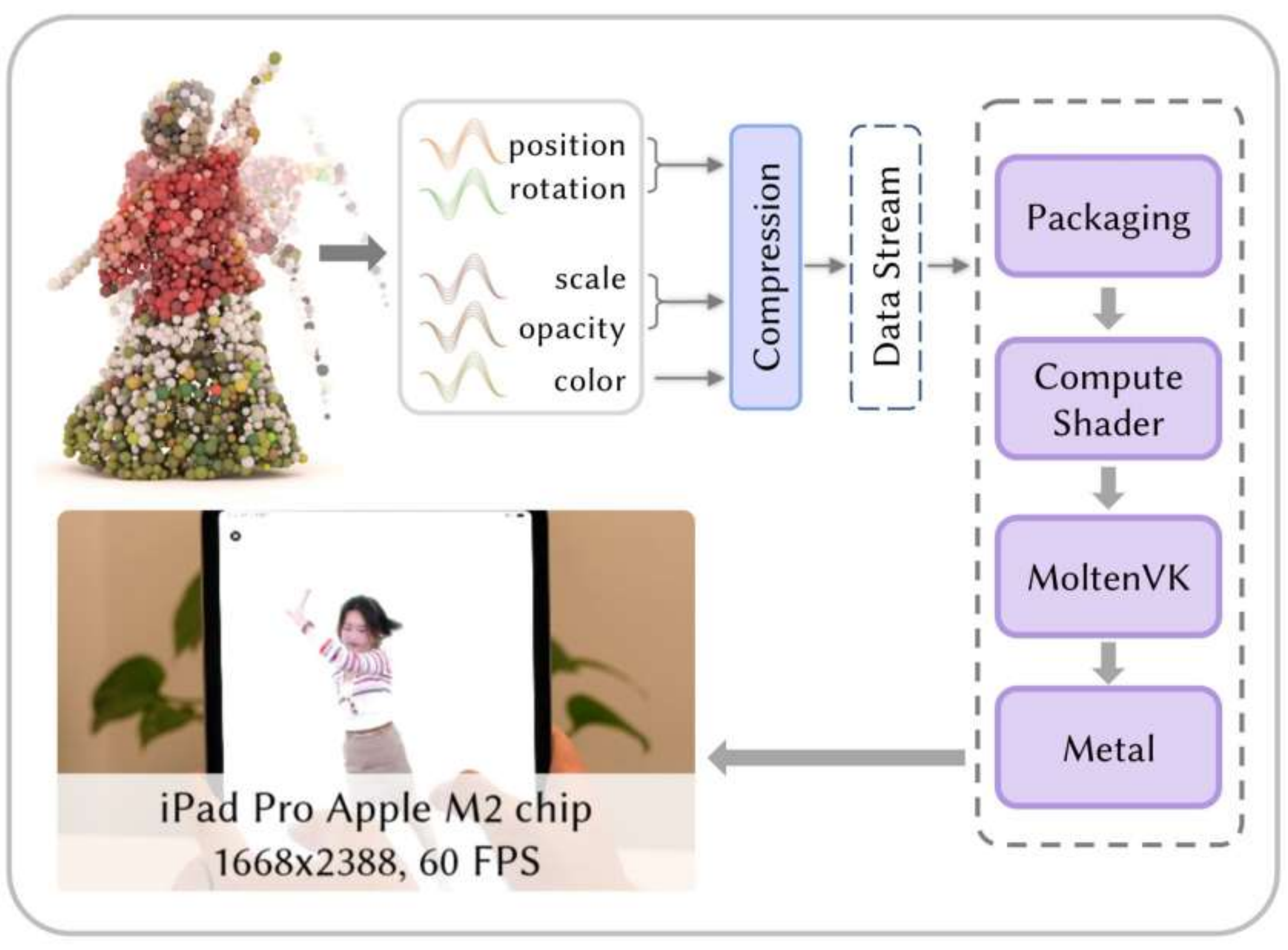
$$\tau_{i,t}^{d} = \underset{k \in \{1,\ldots,L\}}{\operatorname{argmin}} \left\| \mathcal{Z}_d[k] - \mathcal{C}_{i,t}^{d} \right\|_2^2 , 0$$

where $\tau_{i,t}^{d}$ represents the d-order SH index for Gaussian i at frame t. The compressed SH coefficients $\hat{\mathcal{C}}_{i,t}^{d}$ can then be recovered by indexing into the codebook:

$$\hat{\mathcal{C}}_{i,t}^{d} = \mathcal{Z}_d[\tau_{i,t}^{d}]$$

With this encoding, the SH attributes, originally represented as $n \times f \times 48$ float parameters (where n is the number of Gaussians and f is the number of frames), are compressed into $n \times f \times 4$ integer indices, along with four shared codebooks. Furthermore, we observe that the temporally coherent SH coefficients exhibit high consistency even after conversion to indices. On average, only 1% of the SH indices for Gaussians change between adjacent frames. To further optimize storage, instead of saving spatial-temporal SH indices for each frame, we save only the indices from the first frame and the positions where indices change in subsequent frames.

Application



The attribute-specific compression method we propose demonstrates significant potential in practical applications. Due to limitations in computational power and storage capacity, the playback of high-quality volumetric videos has primarily been confined to desktop systems. we have successfully extended this high-quality volumetric video to mobile devices such as iPads. In processing the compressed data stream, we enhanced an open-source, Vulkan-based static Gaussian renderer to support dynamic rendering, and integrated our compression algorithm for real-time decoding.