GR4DGS: SPARSE-VIEW 4D GAUSSIAN COMPRESSION VIA TEMPORAL GRID PREDICTION

Anonymous authors
Paper under double-blind review

ABSTRACT

4D Gaussian representations enable free-viewpoint video, but their frame-wise scene updates incur substantial storage and transmission costs. We investigate this compression challenge under sparse-view capture. In our three-view evaluation of a grid-based streaming codec, motion grids account for about 99% of P-frame payloads, although consecutive quantized grids remain highly correlated. Independent per-frame grid coding therefore repeatedly transmits information that is predictable from the decoder's preceding state. We propose GR4DGS to exploit this redundancy through temporal motion-grid prediction. GR4DGS transmits integer differences from the previously decoded grid, recovering the current quantized grid exactly while retaining the other rendering inputs. To further reduce payloads, GR4DGS-E2E combines residual-domain rate optimization with occupancy-aware initialization: unqueried cells inherit the preceding decoded values, while queried cells restart from a shared warm-up. Across six Neural 3D Video scenes of 300 frames each under three-view training, lossless prediction reduces P-frame payloads by 65–72% relative to the default independently coded 4DGC baseline. The separately trained GR4DGS-E2E variant reduces them by 88–93%, with per-scene mean PSNR differences within 0.03 dB. Including the shared uncompressed keyframe once, E2E reduces sequence payloads by 81–86%.

1 INTRODUCTION

Sequential free-viewpoint video systems built on 3D Gaussian Splatting (3DGS) (Kerbl et al., 2023) transmit a keyframe once, followed by motion fields and residual Gaussians that update the scene (Sun et al., 2024; Hu et al., 2025). Compressing these updates is central to reducing the storage and transmission payload of Gaussian video. A dense motion grid represents the transformation of many Gaussians, but its coefficients need not change substantially between adjacent frames. Independent grid coding repeatedly transmits this shared information. A rate penalty on each absolute grid encourages compact frames without explicitly rewarding differences that are predictable from the decoder's preceding state.

We study this redundancy in sparse-view Gaussian video, where a dense motion representation is trained from only a few cameras. In a three-view 4DGC reproduction at its default operating point, motion grids account for about 99% of P-frame bytes, while residual Gaussian attributes contribute less than 0.1% on each evaluated scene. Consecutive decoded finest-level grids correlate at 0.92–0.95, and their integer differences are predominantly zero (§3.2). These measurements identify a compression opportunity in the representation itself: predict the current grid from the previously decoded grid and transmit only its change.

We introduce GR4DGS (*Grid-Residual* 4DGS), which range-codes these integer differences. The decoder reconstructs the same quantized grid by integer addition; the other rendering inputs are retained. This provides a post-training compression step with exact grid recovery and no retraining. Temporal residual history and causal spatial neighbors additionally define context partitions that reduce the measured sub-stream payload (§4.2).

We also develop GR4DGS-E2E to train for the residual representation being transmitted. A residual-domain rate surrogate encourages compact temporal differences, while occupancy-aware initialization inherits previous values in unread cells and resets queried cells to the warm-up state. This

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

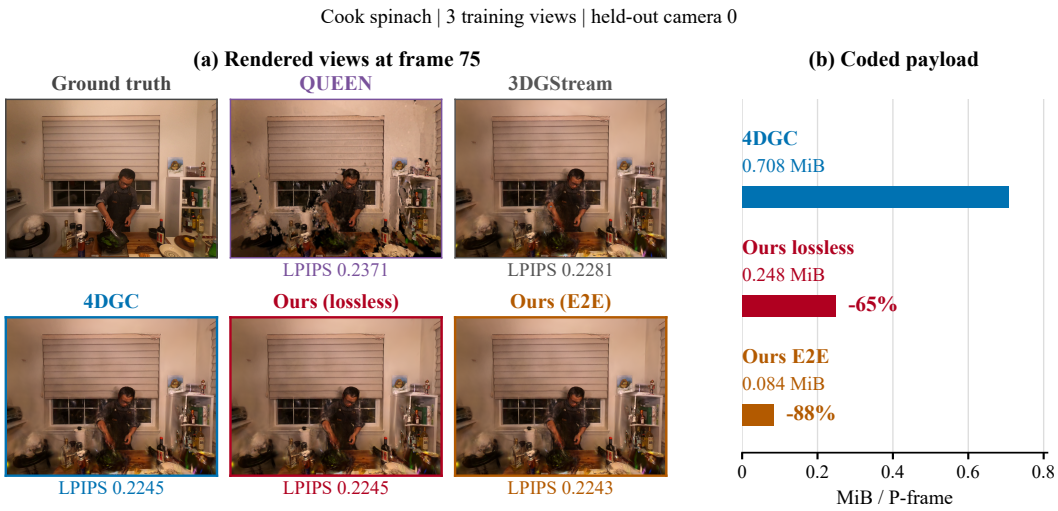


Figure 1: **Rendered reconstruction and compression.** (a) Held-out camera 0, frame 75 of `cook_spinach`, with three training views. The lossless panel reuses the 4DGC rendering: integer grids are recovered exactly and other rendering inputs are retained. (b) Coded P-frame payloads at the default rate weight ($\lambda_{\text{grid}}=10^{-5}$), excluding the keyframe. LPIPS labels and payload bars average frames 1–299; reductions are relative to the displayed 4DGC baseline. Baseline adaptations and sequence totals are in Tables 1 and 2.

combines low residual rate with adaptation of the motion field at the Gaussian reference positions. We evaluate both the lossless recoding and the separately trained variant using measured coded payloads, including sequence totals with the shared keyframe.

Our contributions are:

- **Temporal grid compression.** We formulate inter-frame prediction on quantized motion grids, obtain exact grid recovery through integer residual coding, and evaluate causal context partitions for further compression (§4.1–4.2).
- **Residual-domain training.** We introduce a residual-rate objective and occupancy-aware initialization that reduce the trained grid differences while allowing the queried cells to adapt to each frame (§4.4).
- **Measured Gaussian-video payload reductions.** On six scenes, the two variants reduce P-frame payloads by 65–72% and 88–93% against the default three-view baseline. Including the keyframe, sequence payloads fall by 58–66% and 81–86%. We report byte accounting, component studies, and rate–distortion comparisons (§5).

2 Related Work

Dynamic Gaussians and streaming. Dynamic Gaussian representations include canonical Gaussians with deformation fields (Wu et al., 2024a; Yang et al., 2024c; Bae et al., 2024), 4D primitives (Yang et al., 2024b), 3D primitives with parametric motion and temporal opacity (Li et al., 2024b), and persistent dynamic tracking (Luiten et al., 2024). Streaming approaches use neural transformation caches (Sun et al., 2024), quantized Gaussian residuals (Girish et al., 2024b), hierarchical motion (Gao et al., 2024), or Gaussian attributes arranged as 2D videos (Wang et al., 2024b). 4DGC (Hu et al., 2025) adds rate-aware coding of motion grids and residual attributes. Neural radiance fields (Mildenhall et al., 2020) have been extended to dynamic scenes (Fridovich-Keil et al., 2023; Cao & Johnson, 2023; Attal et al., 2023; Wang et al., 2023a) and streaming or video compression (Li et al., 2022a; Song et al., 2023; Wang et al., 2023b; 2024a; Wu et al., 2024b).

Gaussian compression. Static 3DGS methods reduce storage through quantization and compact attribute representations (Niedermayr et al., 2024; Lee et al., 2024; Girish et al., 2024a). Pruning

and local structure provide additional reductions (Fan et al., 2024; Shin et al., 2025), while context models exploit dependencies among Gaussian attributes (Chen et al., 2024; Wang et al., 2024c; Chen et al., 2025b;a; Zhan et al., 2025). These methods motivate compression of the Gaussian representation itself. We instead target the motion grids transmitted between frames, which can complement compression of the shared keyframe. Compact offline 4DGS (Zhang et al., 2025) stores a whole dynamic scene in one model. 4DGCPro (Zheng et al., 2025) uses hierarchical Gaussians and adaptive motion groups for progressive streaming. Temporal smoothness-aware compression (Lee & Baek, 2025) uses wavelets to encode Gaussian motion trajectories. Our coding target is the integer motion-grid state, combining direct temporal prediction with residual-rate training.

Sparse views and entropy modeling. Sparse-view methods use depth priors (Li et al., 2024a; Zhu et al., 2024; Xiong et al., 2025), co-regularization (Zhang et al., 2024), flow priors (Somraj et al., 2024), or dynamic geometry constraints (Li et al., 2025). We study the rate allocation of a particular streaming codec under sparse capture. Classical codecs use temporal prediction and causal contexts (Wiegand et al., 2003; Marpe et al., 2003; Sullivan et al., 2012); learned codecs use residual coding (Lu et al., 2019) or conditional feature coding (Li et al., 2021). Learned image priors (Ballé et al., 2018; Minnen et al., 2018) and their implementations (Bégaint et al., 2020) inform rate estimation. Our prediction operates on an integer motion grid with a fixed layout, without motion compensation of the grid (Appendix C).

3 Preliminaries and Motivation

3.1 Streamable 4DGS Codec

GR4DGS adopts the representation of 4DGC (Hu et al., 2025), a rate-aware streamable 4DGS codec with the I/P structure of 3DGStream (Sun et al., 2024). A keyframe $\mathcal{G}_0 = \{(\mu_i, \Sigma_i, \alpha_i, \mathbf{c}_i)\}$ is reconstructed from the training views and transmitted once. For frame t, a multi-resolution motion grid $G_t = \{G_t^\ell\}_{\ell=0}^2$ (32^3, 64^3, 128^3 cells) is read by trilinear interpolation at the reference positions, and a small MLP f_{ϕ_t} maps the interpolated features of each reference center μ_i to a displacement and a rotation, $(\Delta\mu_i, \Delta q_i) = f_{\phi_t}(\mathrm{interp}(G_t, \mu_i))$, giving the warped set $\tilde{\mathcal{G}}_t$; residual Gaussians $\Delta\mathcal{G}_t$ are then spawned where accumulated photometric gradients exceed a threshold. The coded payload of frame t is $(S_t, \phi_t, \Delta\mathcal{G}_t)$ with the quantized grid $S_t = \lfloor G_t \rceil$ (integer rounding, step 1); at encode time every level of S_t is range-coded (Witten et al., 1987) with an empirical frequency table transmitted as side information, so its cost is $|\mathrm{AE}(S_t)|$ bits, and during training a factorized-prior surrogate $R(\cdot)$ penalizes grid rate (Appendix D). The per-frame objective is

$$\mathcal{L}_t = \mathcal{L}_{\text{photo}} + \lambda_{\text{grid}}\, R(G_t) + \lambda_{\text{res}}\, R(\Delta\mathcal{G}_t). \tag{1}$$

The representation and its trilinear read are kept as they are; §4 changes only how the grid is *trained and encoded*. The published configuration fixes $\lambda_{\text{grid}}{=}10^{-5}$ and uses λ_{res} as its rate control; we expose λ_{grid} and sweep it. The reference for frame $t{+}1$ is $\tilde{\mathcal{G}}_t \cup \Delta\mathcal{G}_t$. Experiments use three training views unless stated otherwise; §5.1 defines the reproduced baselines.

3.2 Temporal Redundancy in Motion Grids

We examine the default three-view 4DGC reproduction on Neural 3D Video (§5.1). Motion grids contribute about 99% of the P-frame payload, with about 93% of grid bytes in the finest level. Residual Gaussian attributes occupy less than 0.1% on each of the six scenes. The motion grid is therefore the main compression target under this protocol.

Temporal predictability. Consecutive decoded finest-level grids correlate at 0.92–0.95; only 3.7–5.2% of their integer residual coefficients are non-zero (Appendix A). Independently coding each frame repeatedly transmits these correlated values. Predictive coding converts them to a sharply concentrated residual distribution while preserving the quantized grid exactly.

Occupancy guides training. The grid is queried at encoded Gaussian reference positions. A sampled diagnostic on `cook_spinach` estimates that only 41.5% of finest-level cells are queried, yet unqueried cells retain substantial non-zero coefficients (Table 8). This motivates inheriting their previous decoded values during residual-domain training, while queried cells are initialized to support new motion. Appendix E gives the sampling protocol and additional view-count evidence.

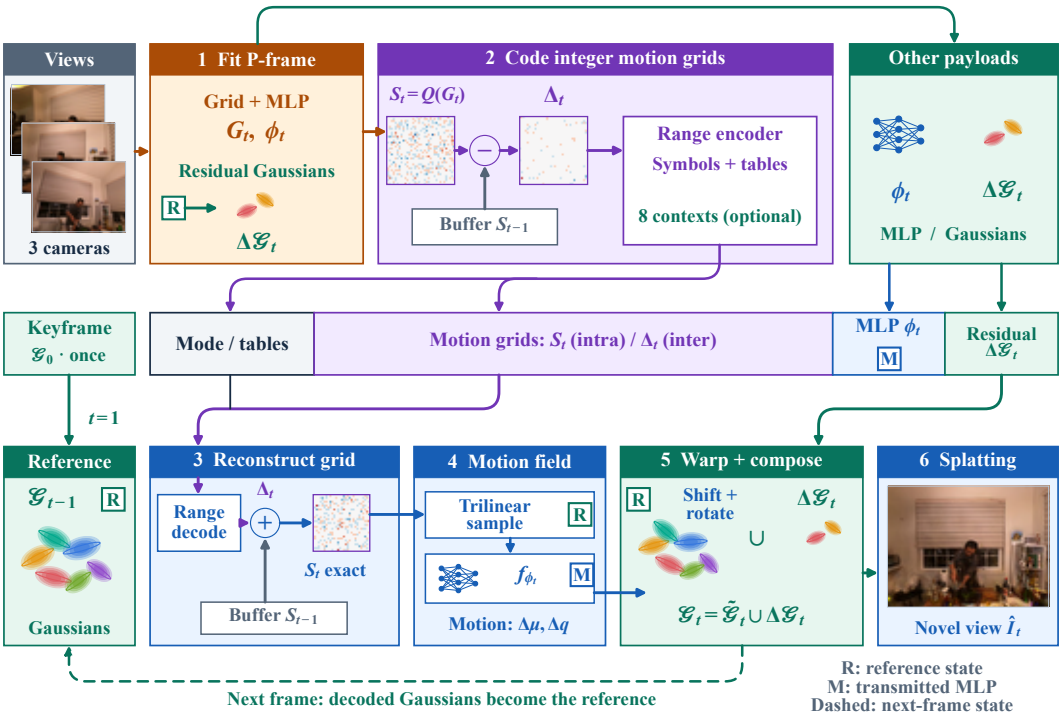


Figure 2: Gaussian codec and logical payload layout (not to byte scale). Ellipsoids denote reference, warped, and residual Gaussians; photos and grid slices are real `cook_spinach` data. For $t \geq 2$, integer residuals reconstruct S_t exactly. The transmitted MLP converts reference-position grid samples to motion. Warped and residual Gaussians form the next reference and are splatted to render a novel view. Q: quantization; matching R/M tags: reference state/transmitted MLP. Buffers hold the preceding decoded grid. The first P-frame is intra-coded; lossless prediction codes the finest level inter, E2E all levels. Context coding is optional. E2E training follows Eqs. 5–6.

4 Method

Fig. 2 shows the compression pipeline. GR4DGS retains the Gaussian representation and changes the coding of its motion grids: integer prediction gives lossless recoding (§4.1), context partitions model residual statistics (§4.2), and residual-domain training produces more compact grid differences (§4.4). §4.3 defines the decoder-visible occupancy used for intra-frame pruning and residual-domain initialization.

4.1 Lossless Inter-Frame Residual Coding

The finest level carries about 93% of grid bytes. Unqueried cells remain close to their shared warm-up values, and consecutive decoded grids correlate at 0.92–0.95. We exploit this correlation by keeping the previous decoded grid at both encoder and decoder. Let S_{t-1} be the decoded grid of the previous frame, held identically by encoder and decoder. Instead of S_t the encoder codes the integer residual

$$\Delta_t = S_t - S_{t-1}, \qquad \text{and the decoder recovers} \qquad S_t = S_{t-1} + \Delta_t, \tag{2}$$

by integer accumulation from the first, intra-coded P-frame grid (Fig. 3); an intra/inter flag distinguishes the two grid modes (container framing is excluded from the payload audit). Since the reference is the decoded integer grid, Eq. 2 is exact: the grid is bit-exact and the remaining rendering inputs are unchanged, and the added grid-prediction operations are to keep S_{t-1} (16 MiB of integers) and add Δ_t to it. The range coder is unchanged as well; Δ_t is zero on about 95–96% of the coefficients, so its empirical frequency table is far more peaked. Every stream is decoded and checked for exact recovery. Across these verified streams, the same coder spends 70–77% fewer bits on the finest level and 65–72% fewer on the P-frame payload, on all six scenes (Table 1). Only

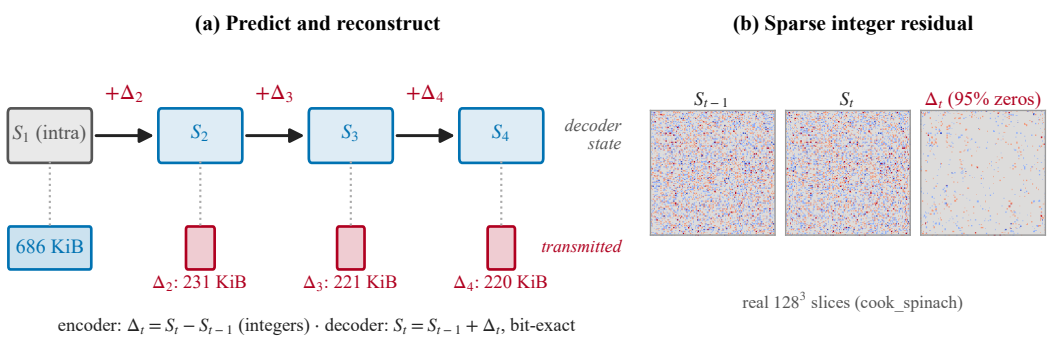


Figure 3: Inter-frame coding of the finest motion-grid level. (a) The first P-frame grid is coded intra; every later frame sends only the integer residual Δ_t against the decoder's current grid (actual frames 1–4, widths to scale, `cook_spinach`). (b) Real 128^3 slices of two consecutive decoded grids and their residual; the displayed residual slice has 94.7% zero coefficients.

the finest level is predicted: the coarser levels correlate at ~ 0.5, contribute about 6.6% of grid bytes, and show mixed rate changes under prediction (Appendix A).

An optional intra/inter switching probe is detailed in Appendix A.1.

4.2 Context-Adaptive Entropy Coding

The baseline uses one frequency table per level. We split residual symbols into groups using whether the preceding residual is non-zero and how many already decoded spatial neighbors are non-zero. These features capture persistence and clustering of grid changes. Let $\mathcal{N}(i)$ be the three causal raster neighbors of cell i ($x-1$, $y-1$, $z-1$, same channel), decoded earlier in the current frame. Each coefficient is assigned the context

$$c_t(i) = \Big(\mathbf{1}[\Delta_{t-1}(i) \neq 0],\ \textstyle\sum_{j \in \mathcal{N}(i)} \mathbf{1}[\Delta_t(j) \neq 0]\Big) \in \{0,1\} \times \{0,1,2,3\}, \tag{3}$$

which a causal decoder can compute from previously decoded symbols (missing boundary neighbors count as zero). The first residual uses one table; this temporal context starts at $t = 3$. The eight possible sub-streams are range-coded with one transmitted frequency table each (a few hundred bytes per frame, counted), with idealized symbol cost $\sum_i -\log_2 p_{c_t(i)}(\Delta_t(i))$, plus tables and coder overhead. The occupancy of the cell and the clipped previous value $|S_{t-1}(i)|$ are optional further contexts. We range-code each group and verify its decoded symbols using the encoder's group assignments. On frames where the context is defined, its measured sub-stream payload removes 25–26% of the residual bytes on every run measured (seven sequences at three, six, and nine views; `cook_spinach`: 197 $\rightarrow$ 148 KiB per frame), and four context features remove 31%; the finest level then costs 78–83% less than independent coding and, on the four full-sequence context runs, the P-frame payload 72–75% less. The contexts are causal by construction, but a standalone raster-order decoder and complete container are not implemented in this evaluation.

4.3 Occupancy and Intra-Frame Coding

The reference Gaussian positions identify which grid cells the motion decoder reads. Let $\mathcal{C}^\ell(\mu)$ be the union of the two eight-cell cubes sampled at the sine- and cosine-encoded position μ at level ℓ (Appendix D). The occupied set and its pruning operator are

$$\mathcal{O}_t^\ell = \textstyle\bigcup_i \mathcal{C}^\ell(\mu_i), \qquad \mathrm{P}(S_t^\ell)(c) = S_t^\ell(c)\,\mathbf{1}[c \in \mathcal{O}_t^\ell]. \tag{4}$$

A cell outside $\mathcal{O}_t^\ell$ contributes to no interpolated value at these reference positions. Zeroing it therefore preserves the motion decoder's grid reads, although it changes the stored grid. The mask is derived from decoder-visible positions and needs no transmitted occupancy map. In the measured intra-coded alternative, the higher zero-symbol frequency reduces the default-point stream by 32–44% (Table 3, Appendix A).

This occupancy set also determines the initialization in §4.4: unread cells can inherit the preceding decoded values with zero initial temporal residual, while queried cells restart from the shared warm-up. For post-training lossless prediction we retain the unpruned grid. The matched-grid comparison in Appendix E evaluates pruning before prediction.

4.4 Residual-Domain Training

Post-training prediction leaves Eq. 1 unchanged: training penalizes the absolute grid even though the encoder will transmit its temporal difference. Each P-frame also restarts from the same warm-up. GR4DGS-E2E aligns training with residual coding by changing the rate term and initialization. For $t \geq 2$, the objective is

$$\mathcal{L}_t = \mathcal{L}_{\text{photo}} + \lambda_{\text{grid}}\, R(G_t - S_{t-1}) + \lambda_{\text{res}}\, R(\Delta \mathcal{G}_t), \tag{5}$$

and the grid of frame t inherits the previous decoded values outside the queried-cell set $\mathcal{O}_t$ (§4.3), while queried cells restart from the warm-up W:

$$G_t^{(0)}(i) = \begin{cases} S_{t-1}(i), & i \notin \mathcal{O}_t, \\ W(i), & i \in \mathcal{O}_t. \end{cases} \tag{6}$$

Every level is then sent as an integer residual exactly as in Eq. 2 (first P-frame intra), so coding stays lossless with respect to the trained grid. Unread cells start with zero residual, and the trained differences become sparse at all levels: about 0.3–0.9% of finest-level coefficients have a non-zero residual. The P-frame payload falls by 88–93% with mean PSNR differences within 0.03 dB (§5.2). Occupancy-aware initialization avoids the two drift cases observed with plain temporal initialization; the controlled comparisons are in §5.5 and Appendix B.

At $t = 1$, no preceding grid exists: training starts from W and uses the absolute-grid rate of Eq. 1. Appendix D.1 gives the shared warm-up selection protocol. The occupancy-pruned intra alternative is evaluated separately from the main prediction variants.

5 Experiments

5.1 Experimental Setup

Datasets and Protocol. Six Neural 3D Video scenes (Li et al., 2022b), 1352×1014, 300 frames, three training cameras and camera 0 for testing (Appendix D). Our 4DGC-family keyframes use 15k iterations; P-frames use 400 motion and 100 residual iterations. Our 4DGC and GR4DGS variants share each scene's keyframe and warm-up, with offline sequence-based preprocessing; other representations follow their own reconstruction procedure from the same input views.

Baselines. *4DGC* (Hu et al., 2025) denotes our reproduction of its public implementation under this three-view protocol, with the added modules disabled and default $\lambda_{\text{grid}}{=}10^{-5}$, $\lambda_{\text{res}}{=}0.02$. *GR4DGS* recodes the same run's finest grid level losslessly, leaving other payloads unchanged. *GR4DGS-E2E* is a separate residual-domain training run. We also evaluate QUEEN (Girish et al., 2024b), TeTriRF (Wu et al., 2024b) (first 100 frames), 3DGStream (Sun et al., 2024), and GC-4DGS (Li et al., 2025), with their distinct storage conventions stated in Appendix D.

Metrics. Test-view PSNR, SSIM (Wang et al., 2004), and LPIPS (AlexNet) (Zhang et al., 2018), averaged over the 299 P-frames. PSNR is the mean of channel-wise dB for the 4DGC family, 3DGStream and GC-4DGS, and joint-RGB PSNR for QUEEN and TeTriRF. Cross-group comparisons use LPIPS (Appendix D). Sizes use MiB (2^{20} bytes) and KiB (2^{10} bytes). P-frame payload includes actual coded grids and frequency tables, residual attributes and side information, and 7196 bytes for the unchanged fp32 MLP. The shared keyframe and transport/container framing are excluded; Table 2 adds each actual keyframe once.

Rate Sweeps and Evaluation. We sweep $\lambda_{\text{grid}} \in \{1, 3, 10, 30\} \times 10^{-5}$ on `cook_spinach` and `sear_steak`. Lossless prediction is checked on the same trained run. Each E2E configuration is evaluated once, and quality differences compare matched sequence means. Appendix F details the evaluation scope.

Table 1: Six-scene comparison: three training views, frames 1–299, sizes in MiB/frame. The 4DGC family uses the default rate weight, excluding the shared keyframe. Other sizes: e, QUEEN's estimate; u, 3DGStream's uncompressed P-frame cache; m, GC-4DGS's fp32 model divided by 300. QUEEN uses joint-RGB PSNR; other rows use mean channel PSNR. Bold PSNR compares the latter; bold SSIM/LPIPS is best displayed. Bold sizes are below matched 4DGC. "–": unverified size. TeTriRF's first-100-frame results are in Table 6.

	cook_spinach				flame_steak			
Method	PSNR↑	SSIM↑	LPIPS↓	MiB↓	PSNR↑	SSIM↑	LPIPS↓	MiB↓
QUEEN^e	20.401	0.7713	0.2371	0.693	18.490	0.7154	0.3416	1.685
3DGStream^u	21.484	0.8059	0.2281	7.613	22.249	0.8198	0.2192	7.613
GC-4DGS^m	**23.377**	**0.8220**	0.2525	1.006	**23.994**	**0.8399**	0.2252	1.152
4DGC	21.365	0.8070	0.2245	0.708	22.192	0.8230	**0.2103**	0.723
GR4DGS (ours)	21.365	0.8070	0.2245	**0.248**	22.192	0.8230	**0.2103**	**0.217**
GR4DGS-E2E (ours)	21.387	0.8072	**0.2243**	**0.084**	22.195	0.8229	0.2114	**0.062**
	sear_steak				coffee_martini			
Method	PSNR↑	SSIM↑	LPIPS↓	MiB↓	PSNR↑	SSIM↑	LPIPS↓	MiB↓
QUEEN^e	20.914	0.7672	0.2273	0.657	15.597	0.5912	0.4723	1.040
3DGStream^u	21.866	0.8023	0.2135	7.607	17.285	0.6694	0.3725	7.614
GC-4DGS^m	**23.682**	**0.8159**	0.2347	1.434	**17.805**	**0.7165**	0.5100	–
4DGC	21.904	0.8028	0.2086	0.686	17.343	0.6705	**0.3693**	0.686
GR4DGS (ours)	21.904	0.8028	0.2086	**0.194**	17.343	0.6705	**0.3693**	**0.233**
GR4DGS-E2E (ours)	21.887	0.8020	**0.2084**	**0.047**	17.343	0.6708	0.3701	**0.052**
	cut_roasted_beef				flame_salmon_1			
Method	PSNR↑	SSIM↑	LPIPS↓	MiB↓	PSNR↑	SSIM↑	LPIPS↓	MiB↓
QUEEN^e	19.745	0.7483	0.2713	0.644	16.688	0.6369	0.4117	0.994
4DGC	20.212	**0.7859**	**0.2429**	0.715	17.835	0.6888	0.3547	0.686
GR4DGS (ours)	20.212	**0.7859**	**0.2429**	**0.240**	17.835	0.6888	0.3547	**0.232**
GR4DGS-E2E (ours)	**20.226**	0.7855	0.2436	**0.075**	**17.840**	**0.6897**	**0.3414**	**0.060**

5.2 Main Results

P-frame compression. Table 1 compares the full-sequence results. Lossless GR4DGS reduces the default baseline's P-frame payload from 0.686–0.723 to 0.194–0.248 MiB, a 65–72% reduction, while preserving the original quantized grids and remaining rendering inputs. GR4DGS-E2E lowers the payload further to 0.047–0.084 MiB (88–93% below the default baseline), with mean PSNR differences within 0.03 dB and SSIM within 0.001. LPIPS differs by at most 0.0012 on five scenes and improves by 0.013 on `flame_salmon_1`.

Sequence compression. Table 2 includes each actual shared uncompressed keyframe once, together with all 299 P-frames. Lossless prediction reduces sequence payloads by 58–66%, and GR4DGS-E2E by 81–86%. For example, `cook_spinach` falls from 227.90 to 90.43 MiB with exact grid recovery, or 41.30 MiB with residual-domain training. The savings therefore persist after accounting for the keyframe.

Additional view counts. On `cook_spinach`, lossless prediction also reduces the default six- and nine-view P-frame payloads by 67.1% and 68.6%, respectively (Table 4). Example held-out renderings and per-frame payload traces are provided in Appendix E and Fig. 5.

5.3 Comparison with Other Methods

Under the adapted three-view protocol, GR4DGS-E2E records lower LPIPS than QUEEN (Girish et al., 2024b) on all six scenes. QUEEN's estimated payload is 0.64–1.69 MiB per P-frame. Over the same first-100-frame evaluation interval, GR4DGS-E2E records LPIPS 0.12–0.18 lower than TeTriRF (Wu et al., 2024b). TeTriRF's video size is 0.06–0.10 MiB/frame (group MLPs excluded),

Table 2: Sequence payload in MiB: the shared uncompressed keyframe once plus 299 P-frames, with native fp32 MLPs and no transport/container framing. Bold totals are below default three-view 4DGC. Reductions list lossless / E2E.

Scene	Keyframe	4DGC	Lossless	E2E	Reduction (%)
cook_spinach	16.31	227.90	**90.43**	**41.30**	60.3 / 81.9
flame_steak	16.42	232.60	**81.38**	**34.95**	65.0 / 85.0
sear_steak	16.79	222.00	**74.69**	**30.75**	66.4 / 86.1
coffee_martini	27.87	232.94	**97.65**	**43.32**	58.1 / 81.4
cut_roasted_beef	15.93	229.66	**87.59**	**38.45**	61.9 / 83.3
flame_salmon_1	26.89	231.96	**96.30**	**44.73**	58.5 / 80.7

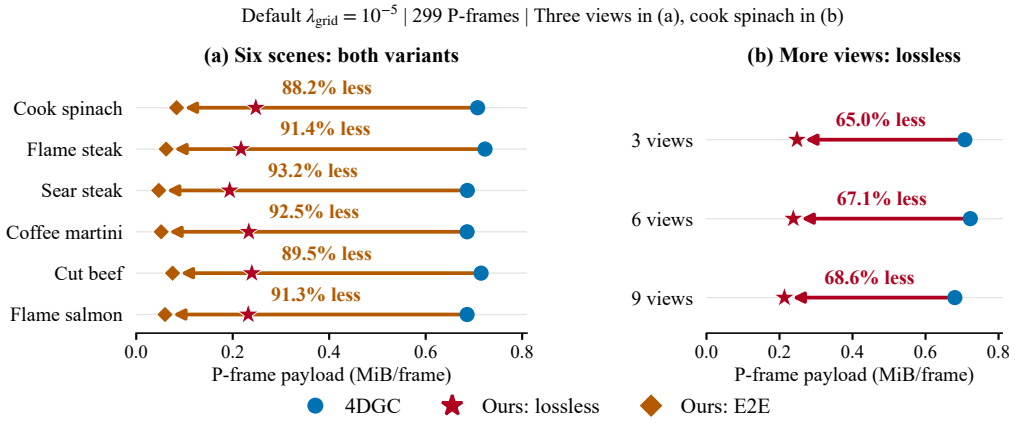


Figure 4: **P-frame compression at the default operating point.** (a) Both proposed variants reduce payload across all six three-view scenes: 65–72% for lossless prediction and 88–93% for E2E. (b) Lossless prediction also saves 67.1% and 68.6% at six and nine views. Linear axes show mean MiB/frame over 299 P-frames, excluding shared keyframes. Lossless recoding preserves grids exactly; E2E is trained separately, with mean PSNR differences within 0.03 dB in (a). Non-default rate points are in Table 9.

versus 0.05–0.09 for GR4DGS-E2E (Table 6, Appendix D). 3DGStream uses about 7.6 MiB per P-frame at similar PSNR to 4DGC. GC-4DGS achieves higher PSNR but worse LPIPS with an offline model. These storage conventions are reported separately from the matched coded-payload comparison within the 4DGC family.

5.4 Rate–Distortion Analysis

Compression at a matched operating point. Fig. 4 compares the default independent-grid payload with both proposed variants across six scenes, and verifies lossless savings at three, six, and nine views. Lossless prediction recodes the same trained representation; E2E learns compact residuals with per-scene mean PSNR differences within 0.03 dB. Table 9 reports the baseline rate-weight sweep; Appendix A compares tuned baselines and prediction variants over matched frames.

5.5 Ablation Study

Coding alternatives. Table 3 separates alternative configurations. Context coding lowers the prediction payload from 0.248 to 0.200 MiB; on the six E2E runs it removes another 13–16% of finest-level residual bytes. Appendix B reports training variants and drift cases.

Occupancy-aware initialization. Occupancy-aware initialization avoids the two observed drift cases of plain temporal initialization (1.75 and 0.65 dB losses), while inheriting unread cells keeps their initial residual zero. We use this variant for all six main-table scenes.

Table 3: Alternative coding configurations on `cook_spinach` (299 P-frames, shared keyframe). Prediction and prediction+context recode the same run; pruning and E2E use separate runs. Context size measures verified sub-streams using encoder partitions, not a complete decoder. Bold payloads are below 4DGC; quality is reported separately.

Configuration	PSNR↑	SSIM↑	LPIPS↓	MiB/frame↓	Saving↑
4DGC (independent grids)	21.365	0.8070	0.2245	0.708	–
Occupancy pruning (§4.3)	21.373	0.8075	0.2247	**0.420**	41%
Lossless prediction (§4.1)	21.365	0.8070	0.2245	**0.248**	65%
Prediction + context probe (§4.2)	21.365	0.8070	0.2245	**0.200**	72%
GR4DGS-E2E, occupancy-aware init.	21.387	0.8072	0.2243	**0.084**	88%

Training variants. Table 5 isolates residual-rate training and compares initialization strategies over matched 130-P-frame intervals. On `flame_salmon_1` and six-view `cook_spinach`, occupancy-aware initialization gives 0.069 and 0.065 MiB/frame: 90.0% and 91.0% below the matched baselines, with mean PSNR differences of $+0.01$ and -0.02 dB. Appendix B includes the rate-only, plain-initialization, and GOP-refresh controls. The six-scene main results use the full 299 P-frames.

Preserving temporal coherence. Unread cells already have sparse temporal differences. Keeping their decoded values allows prediction to reuse them if motion makes these cells queried again. GR4DGS therefore predicts unpruned grids; the matched-grid pruning controls and residual statistics are in Table 7, Appendix E.

5.6 Efficiency

Prediction adds one integer subtraction per coefficient at the encoder and one addition at the decoder. With 32-bit symbols, grid state uses 16 MiB for finest-level prediction and 20.5 MiB when all three levels are predicted; contextual coding additionally needs previous-residual indicators. The finest-level prediction probe records about 1.0–1.3 s per frame for range encoding. This excludes decoding, warping, and rendering; context-decoder timing remains unmeasured. E2E retains the baseline's 400+100 optimization iterations per frame.

6 Conclusion

We presented GR4DGS, a compression method for sparse-view 4D Gaussian representations based on temporal motion-grid prediction. Integer residual coding reconstructs the original quantized grids exactly, and residual-domain training with occupancy-aware initialization further reduces the transmitted payload. Across six scenes, the variants save 65–72% and 88–93% of default-baseline P-frame bytes, or 58–66% and 81–86% of sequence payloads including the keyframe.

AI USE STATEMENT

Claude and Codex assisted with (i) method and experiment development, (ii) implementation and debugging, (iii) result analysis and literature verification, and (iv) manuscript and figure preparation. We checked AI-assisted work against experimental records, decoder tests, and primary sources. The authors are responsible for the final work.

ETHICS STATEMENT

This work uses the public Neural 3D Video benchmark and does not collect new human-subject data. Video reconstruction can reproduce identifiable appearances; any deployment should respect data licenses and privacy.

REPRODUCIBILITY STATEMENT

The supplement provides source code, per-frame metrics, byte-counting scripts, figure inputs, and a SHA256 manifest. Appendix D specifies the data, cameras, and coded components; `GR4DGS/REPRODUCE_0925.md` gives execution commands and dependencies. Every main lossless grid stream was decoded and compared with its source grid.

REFERENCES

Benjamin Attal, Jia-Bin Huang, Christian Richardt, Michael Zollhöfer, Johannes Kopf, Matthew O'Toole, and Changil Kim. HyperReel: High-fidelity 6-DoF video with ray-conditioned sampling. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 16610–16620, 2023. doi: 10.1109/cvpr52729.2023.01594.

Jeongmin Bae, Seoha Kim, Youngsik Yun, Hahyun Lee, Gun Bang, and Youngjung Uh. Per-Gaussian embedding-based deformation for deformable 3D Gaussian splatting. In *European Conference on Computer Vision (ECCV)*, pp. 321–335, 2024. doi: 10.1007/978-3-031-72633-0_18.

Johannes Ballé, David Minnen, Saurabh Singh, Sung Jin Hwang, and Nick Johnston. Variational image compression with a scale hyperprior. In *International Conference on Learning Representations (ICLR)*, 2018. URL https://research.google/pubs/variational-image-compression-with-a-scale-hyperprior/.

Jean Bégaint, Fabien Racapé, Simon Feltman, and Akshay Pushparaja. CompressAI: a PyTorch library and evaluation platform for end-to-end compression research. *arXiv preprint arXiv:2011.03029*, 2020. doi: 10.48550/arXiv.2011.03029.

Ang Cao and Justin Johnson. HexPlane: A fast representation for dynamic scenes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 130–141, 2023. doi: 10.1109/cvpr52729.2023.00021.

Chenjie Cao, Xinlin Ren, and Yanwei Fu. MVSFormer: Multi-view stereo by learning robust image features and temperature-based depth. *Transactions on Machine Learning Research*, 2022. URL https://openreview.net/forum?id=2VWR6JfwNo.

Yihang Chen, Qianyi Wu, Weiyao Lin, Mehrtash Harandi, and Jianfei Cai. HAC: Hash-grid assisted context for 3D Gaussian splatting compression. In *European Conference on Computer Vision (ECCV)*, pp. 422–438, 2024. doi: 10.1007/978-3-031-72667-5_24.

Yihang Chen, Qianyi Wu, Mengyao Li, Weiyao Lin, Mehrtash Harandi, and Jianfei Cai. Fast feed-forward 3D Gaussian splatting compression. In *International Conference on Learning Representations (ICLR)*, 2025a. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/ba404795c58d122a4b6fc2672d84d9f3-Abstract-Conference.html.

Yihang Chen, Qianyi Wu, Weiyao Lin, Mehrtash Harandi, and Jianfei Cai. HAC++: Towards 100X compression of 3D Gaussian splatting. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 47(11):10210–10226, 2025b. doi: 10.1109/tpami.2025.3594066.

Zhiwen Fan, Kevin Wang, Kairun Wen, Zehao Zhu, Dejia Xu, and Zhangyang Wang. LightGaussian: Unbounded 3D Gaussian compression with 15x reduction and 200+ FPS. In *Advances in Neural Information Processing Systems*, volume 37, pp. 140138–140158, 2024. doi: 10.52202/079017-4447.

Sara Fridovich-Keil, Giacomo Meanti, Frederik Rahbæk Warburg, Benjamin Recht, and Angjoo Kanazawa. K-Planes: Explicit radiance fields in space, time, and appearance. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 12479–12488, 2023. doi: 10.1109/cvpr52729.2023.01201.

Qiankun Gao, Jiarui Meng, Chengxiang Wen, Jie Chen, and Jian Zhang. HiCoM: Hierarchical coherent motion for dynamic streamable scenes with 3D Gaussian splatting. In *Advances in Neural Information Processing Systems*, volume 37, pp. 80609–80633, 2024. doi: 10.52202/079017-2563.

Sharath Girish, Kamal Gupta, and Abhinav Shrivastava. EAGLES: Efficient accelerated 3D Gaussians with lightweight EncodingS. In *European Conference on Computer Vision (ECCV)*, pp. 54–71, 2024a. doi: 10.1007/978-3-031-73036-8_4.

Sharath Girish, Tianye Li, Amrita Mazumdar, Abhinav Shrivastava, David Luebke, and Shalini De Mello. QUEEN: QUantized efficient ENcoding of dynamic Gaussians for streaming freeviewpoint videos. In *Advances in Neural Information Processing Systems*, volume 37, pp. 43435–43467, 2024b. doi: 10.52202/079017-1376.

Qiang Hu, Zihan Zheng, Houqiang Zhong, Sihua Fu, Li Song, Xiaoyun Zhang, Guangtao Zhai, and Yanfeng Wang. 4DGC: Rate-aware 4D Gaussian compression for efficient streamable freeviewpoint video. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 875–885, 2025. doi: 10.1109/cvpr52734.2025.00090.

Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3D Gaussian splatting for real-time radiance field rendering. *ACM Transactions on Graphics*, 42(4):1–14, 2023. doi: 10.1145/3592433.

Hyeongmin Lee and Kyungjune Baek. Temporal smoothness-aware rate-distortion optimized 4D Gaussian splatting. In *Advances in Neural Information Processing Systems*, volume 38, Main Conference, pp. 132174–132204, 2025. doi: 10.52202/085713-4401.

Joo Chan Lee, Daniel Rho, Xiangyu Sun, Jong Hwan Ko, and Eunbyung Park. Compact 3D Gaussian representation for radiance field. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 21719–21728, 2024. doi: 10.1109/cvpr52733.2024.02052.

Jiahao Li, Bin Li, and Yan Lu. Deep contextual video compression. In *Advances in Neural Information Processing Systems*, volume 34, pp. 18114–18125, 2021. URL https://proceedings.neurips.cc/paper/2021/hash/96b250a90d3cf0868c83f8c965142d2a-Abstract.html.

Jiahe Li, Jiawei Zhang, Xiao Bai, Jin Zheng, Xin Ning, Jun Zhou, and Lin Gu. DNGaussian: Optimizing sparse-view 3D Gaussian radiance fields with global-local depth normalization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 20775–20785, 2024a. doi: 10.1109/cvpr52733.2024.01963.

Lingzhi Li, Zhen Shen, Zhongshu Wang, Li Shen, and Ping Tan. Streaming radiance fields for 3D video synthesis. In *Advances in Neural Information Processing Systems*, volume 35, pp. 13485–13498, 2022a. doi: 10.52202/068431-0980.

Tianye Li, Mira Slavcheva, Michael Zollhöfer, Simon Green, Christoph Lassner, Changil Kim, Tanner Schmidt, Steven Lovegrove, Michael Goesele, Richard Newcombe, and Zhaoyang Lv. Neural 3D video synthesis from multi-view video. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 5511–5521, 2022b. doi: 10.1109/cvpr52688.2022.00544.

Yiwei Li, Jiannong Cao, Penghui Ruan, Divya Saxena, Songye Zhu, and Yinfeng Cao. Geometry-consistent 4D Gaussian splatting for sparse-input dynamic view synthesis. In *IEEE Annual Congress on Artificial Intelligence of Things (AIoT)*, pp. 17–24, 2025. doi: 10.1109/aiot66900.2025.00015.

Zhan Li, Zhang Chen, Zhong Li, and Yi Xu. Spacetime Gaussian feature splatting for real-time dynamic view synthesis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 8508–8520, 2024b. doi: 10.1109/cvpr52733.2024.00813.

Guo Lu, Wanli Ouyang, Dong Xu, Xiaoyun Zhang, Chunlei Cai, and Zhiyong Gao. DVC: An end-to-end deep video compression framework. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 10998–11007, 2019. doi: 10.1109/cvpr.2019.01126.

Jonathon Luiten, Georgios Kopanas, Bastian Leibe, and Deva Ramanan. Dynamic 3D Gaussians: Tracking by persistent dynamic view synthesis. In *International Conference on 3D Vision (3DV)*, pp. 800–809, 2024. doi: 10.1109/3dv62453.2024.00044.

Detlev Marpe, Heiko Schwarz, and Thomas Wiegand. Context-based adaptive binary arithmetic coding in the H.264/AVC video compression standard. *IEEE Transactions on Circuits and Systems for Video Technology*, 13(7):620–636, 2003. doi: 10.1109/tcsvt.2003.815173.

Ben Mildenhall, Pratul P. Srinivasan, Matthew Tancik, Jonathan T. Barron, Ravi Ramamoorthi, and Ren Ng. NeRF: Representing scenes as neural radiance fields for view synthesis. In *European Conference on Computer Vision (ECCV)*, pp. 405–421, 2020. doi: 10.1007/978-3-030-58452-8_24.

David Minnen, Johannes Ballé, and George D. Toderici. Joint autoregressive and hierarchical priors for learned image compression. In *Advances in Neural Information Processing Systems*, volume 31, 2018. URL https://proceedings.neurips.cc/paper/2018/hash/53edebc543333dfbf7c5933af792c9c4-Abstract.html.

Simon Niedermayr, Josef Stumpfegger, and Rüdiger Westermann. Compressed 3D Gaussian splatting for accelerated novel view synthesis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 10349–10358, 2024. doi: 10.1109/cvpr52733.2024.00985.

Seungjoo Shin, Jaesik Park, and Sunghyun Cho. Locality-aware Gaussian compression for fast and high-quality rendering. In *International Conference on Learning Representations (ICLR)*, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/007ec2ca7a2079de8e7e45102d9f0feb-Abstract-Conference.html.

Nagabhushan Somraj, Kapil Choudhary, Sai Harsha Mupparaju, and Rajiv Soundararajan. Factorized motion fields for fast sparse input dynamic view synthesis. In *ACM SIGGRAPH Conference Papers (SIGGRAPH)*, pp. 1–12, 2024. doi: 10.1145/3641519.3657498.

Liangchen Song, Anpei Chen, Zhong Li, Zhang Chen, Lele Chen, Junsong Yuan, Yi Xu, and Andreas Geiger. NeRFPlayer: A streamable dynamic scene representation with decomposed neural radiance fields. *IEEE Transactions on Visualization and Computer Graphics*, 29(5):2732–2742, 2023. doi: 10.1109/tvcg.2023.3247082.

Gary J. Sullivan, Jens-Rainer Ohm, Woo-Jin Han, and Thomas Wiegand. Overview of the high efficiency video coding (HEVC) standard. *IEEE Transactions on Circuits and Systems for Video Technology*, 22(12):1649–1668, 2012. doi: 10.1109/tcsvt.2012.2221191.

Jiakai Sun, Han Jiao, Guangyuan Li, Zhanjie Zhang, Lei Zhao, and Wei Xing. 3DGStream: On-the-fly training of 3D Gaussians for efficient streaming of photo-realistic free-viewpoint videos. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 20675–20685, 2024. doi: 10.1109/cvpr52733.2024.01954.

Feng Wang, Sinan Tan, Xinghang Li, Zeyue Tian, Yafei Song, and Huaping Liu. Mixed neural voxels for fast multi-view video synthesis. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 19649–19659, 2023a. doi: 10.1109/iccv51070.2023.01805.

Liao Wang, Qiang Hu, Qihan He, Ziyu Wang, Jingyi Yu, Tinne Tuytelaars, Lan Xu, and Minye Wu. Neural residual radiance fields for streamably free-viewpoint videos. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 76–87, 2023b. doi: 10.1109/cvpr52729.2023.00016.

Liao Wang, Kaixin Yao, Chengcheng Guo, Zhirui Zhang, Qiang Hu, Jingyi Yu, Lan Xu, and Minye Wu. VideoRF: Rendering dynamic radiance fields as 2D feature video streams. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 470–481, 2024a. doi: 10.1109/cvpr52733.2024.00052.

Penghao Wang, Zhirui Zhang, Liao Wang, Kaixin Yao, Siyuan Xie, Jingyi Yu, Minye Wu, and Lan Xu. V^3: Viewing volumetric videos on mobiles via streamable 2D dynamic Gaussians. *ACM Transactions on Graphics*, 43(6):1–13, 2024b. doi: 10.1145/3687935.

Yufei Wang, Zhihao Li, Lanqing Guo, Wenhan Yang, Alex C. Kot, and Bihan Wen. ContextGS: Compact 3D Gaussian splatting with anchor level context model. In *Advances in Neural Information Processing Systems*, volume 37, pp. 51532–51551, 2024c. doi: 10.52202/079017-1631.

Zhou Wang, Alan C. Bovik, Hamid R. Sheikh, and Eero P. Simoncelli. Image quality assessment: From error visibility to structural similarity. *IEEE Transactions on Image Processing*, 13(4): 600–612, 2004. doi: 10.1109/tip.2003.819861.

Thomas Wiegand, Gary J. Sullivan, Gisle Bjøntegaard, and Ajay Luthra. Overview of the H.264/AVC video coding standard. *IEEE Transactions on Circuits and Systems for Video Technology*, 13(7):560–576, 2003. doi: 10.1109/tcsvt.2003.815165.

Ian H. Witten, Radford M. Neal, and John G. Cleary. Arithmetic coding for data compression. *Communications of the ACM*, 30(6):520–540, 1987. doi: 10.1145/214762.214771.

Guanjun Wu, Taoran Yi, Jiemin Fang, Lingxi Xie, Xiaopeng Zhang, Wei Wei, Wenyu Liu, Qi Tian, and Xinggang Wang. 4D Gaussian splatting for real-time dynamic scene rendering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 20310–20320, 2024a. doi: 10.1109/cvpr52733.2024.01920.

Minye Wu, Zehao Wang, Georgios Kouros, and Tinne Tuytelaars. TeTriRF: Temporal tri-plane radiance fields for efficient free-viewpoint video. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 6487–6496, 2024b. doi: 10.1109/cvpr52733.2024.00620.

Haolin Xiong, Sairisheek Muttukuru, Hanyuan Xiao, Rishi Upadhyay, Pradyumna Chari, Yajie Zhao, and Achuta Kadambi. SparseGS: Sparse view synthesis using 3D Gaussian splatting. In *International Conference on 3D Vision (3DV)*, pp. 1032–1041, 2025. doi: 10.1109/3dv66043.2025.00100.

Lihe Yang, Bingyi Kang, Zilong Huang, Zhen Zhao, Xiaogang Xu, Jiashi Feng, and Hengshuang Zhao. Depth Anything V2. In *Advances in Neural Information Processing Systems*, volume 37, pp. 21875–21911, 2024a. doi: 10.52202/079017-0688.

Zeyu Yang, Hongye Yang, Zijie Pan, and Li Zhang. Real-time photorealistic dynamic scene representation and rendering with 4D Gaussian splatting. In *International Conference on Learning Representations (ICLR)*, 2024b. URL https://proceedings.iclr.cc/paper_files/paper/2024/hash/26230ff5299de0929d03ed3576c3bbf9-Abstract-Conference.html.

Ziyi Yang, Xinyu Gao, Wen Zhou, Shaohui Jiao, Yuqing Zhang, and Xiaogang Jin. Deformable 3D Gaussians for high-fidelity monocular dynamic scene reconstruction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 20331–20341, 2024c. doi: 10.1109/cvpr52733.2024.01922.

Yu-Ting Zhan, Cheng-Yuan Ho, He-Bi Yang, Yi-Hsin Chen, Jui Chiu Chiang, Yu-Lun Liu, and Wen-Hsiao Peng. CAT-3DGS: A context-adaptive triplane approach to rate-distortion-optimized 3DGS compression. In *International Conference on Learning Representations (ICLR)*, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/41e638f36d9cb052669d30f559e7c0b7-Abstract-Conference.html.

Jiawei Zhang, Jiahe Li, Xiaohan Yu, Lei Huang, Lin Gu, Jin Zheng, and Xiao Bai. CoR-GS: Sparse-view 3D Gaussian splatting via co-regularization. In *European Conference on Computer Vision (ECCV)*, pp. 335–352, 2024. doi: 10.1007/978-3-031-73232-4_19.

Richard Zhang, Phillip Isola, Alexei A. Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 586–595, 2018. doi: 10.1109/cvpr.2018.00068.

Xinjie Zhang, Zhening Liu, Yifan Zhang, Xingtong Ge, Dailan He, Tongda Xu, Yan Wang, Zehong Lin, Shuicheng Yan, and Jun Zhang. MEGA: Memory-efficient 4D Gaussian splatting for dynamic scenes. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 27828–27838, 2025. doi: 10.1109/iccv51701.2025.02583.

Zihan Zheng, Zhenlong Wu, Houqiang Zhong, Yuan Tian, Ning Cao, Lan Xu, Jiangchao Yao, Xiaoyun Zhang, Qiang Hu, and Wenjun Zhang. 4DGCPro: Efficient hierarchical 4D Gaussian compression for progressive volumetric video streaming. In *Advances in Neural Information Processing Systems*, volume 38, Main Conference, pp. 100860–100885, 2025. doi: 10.52202/085713-3376.

Zehao Zhu, Zhiwen Fan, Yifan Jiang, and Zhangyang Wang. FSGS: Real-time few-shot view synthesis using Gaussian splatting. In *European Conference on Computer Vision (ECCV)*, pp. 145–163, 2024. doi: 10.1007/978-3-031-72933-1_9.

A Inter-frame prediction: full results and methodology

Measurement. The codec rounds every grid level to integers (q=1) and range-codes each level with an empirical frequency table that it transmits alongside (Appendix D); the learned entropy model shapes the grid during training but takes no part in encoding. We therefore measure prediction on the *decoded* integer grids of a finished run, with the codec's own coder: for every P-frame $t \geq 2$ the residual $\Delta_t = S_t - S_{t-1}$ of the finest level is range-coded exactly as the codec range-codes S_t, written to disk, decoded back with the codec's decoder, and compared bit for bit ($S_{t-1} + \Delta_t = S_t$ on every frame); the first P-frame is coded intra. Re-coding the decoded grid itself reproduces the run's own stream size on every frame and level, which checks that the coder is the codec's, and the run's P-frame payload in Table 1 is reduced by exactly the finest level's saving, the intra first frame included. All numbers cover the full 299 P-frames of the runs in Table 1 (`tools/temporal_delta_probe2.py`; per-frame values in `results/probe2/`, aggregates in `results/summary/temporal_prediction_probe2.csv`); Fig. 5 shows the per-frame stream sizes.

What the residual contains. Only 3.7–5.2% of the finest level's coefficients have a non-zero residual, and the frame-to-frame correlation of the decoded grids is 0.92–0.95 at 128^3 against 0.43–0.61 at 32^3 and 64^3; predicting the two coarser levels changes their bytes by between -10% and $+15\%$, a small contribution to total payload (they hold about 6.6% of grid bytes), so only the finest level is predicted. Encoding the finest-level residual takes about 1.0–1.3 s per frame in the Python range-coder probe, and the decoder's accumulation is one integer addition over 4.2M coefficients on a 16 MiB buffer. This operation count is separate from the measured range-encoding time.

The controlled pruning/prediction comparison is reported in §5.5 and Table 7.

A default-operating-point optimization. Inter-frame prediction helps at the default $\lambda_{\text{grid}}=10^{-5}$. In 40-P-frame `cook_spinach` pilots, increasing the weight to 3×10^{-5} and 10^{-4} makes the finest-level residual 14.2% and 54.5% larger than independent coding. The grid is already sparse, and its temporal difference can cost more. An encoder can select the cheaper of intra and inter coding, with a mode flag. These pilots are not inserted as full-sequence R–D points. At six and nine training views, the full-sequence default-point payload reductions remain 67.1% and 68.6%, respectively (Table 4).

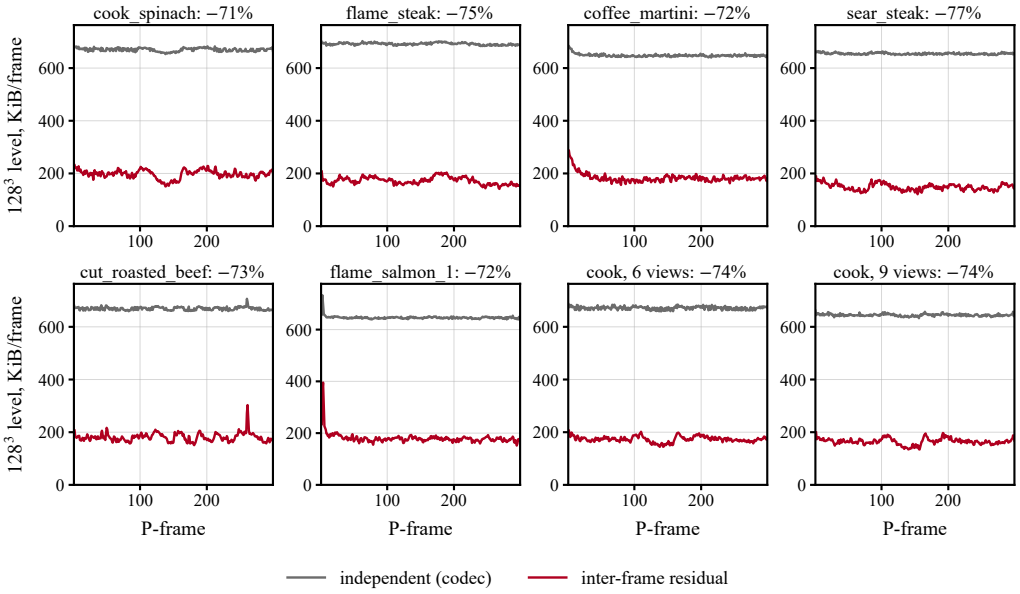


Figure 5: Per-frame size of the finest (128^3) motion-grid level over the full sequence, coded independently by the codec (grey) and as the inter-frame residual (red), for every scene and view count. Prediction consistently reduces the finest-level payload after the initial intra frame in these measured sequences; grid reconstruction remains exact.

Table 4: Full-sequence lossless prediction over 299 P-frames: all six scenes at three views, plus `cook_spinach` at six and nine views. All streams are decoded and verified bit-exact. "codec"/"Δ": mean KiB for the independently coded/residual 128^3 level; residual means use frames 2–299. "corr.": consecutive-grid correlation. P-frame payloads include the intra first frame and average all 299 P-frames (Appendix D). Bold payloads are below each matched baseline; rendering inputs are unchanged.

	128^3 level				P-frame payload (MiB)		
Scene (views)	codec	Δ	cut	corr.	base	GR4DGS	cut
`cook_spinach` (3)	669.9	197.4	70.5%	0.92	0.708	**0.248**	65.0%
`flame_steak` (3)	692.0	172.3	75.1%	0.94	0.723	**0.217**	69.9%
`coffee_martini` (3)	648.2	183.2	71.7%	0.93	0.686	**0.233**	66.0%
`sear_steak` (3)	654.8	148.6	77.3%	0.95	0.686	**0.194**	71.8%
`cut_roasted_beef` (3)	669.3	181.1	72.9%	0.93	0.715	**0.240**	66.5%
`flame_salmon_1` (3)	646.4	180.2	72.1%	0.93	0.686	**0.232**	66.2%
`cook_spinach` (6)	671.6	173.4	74.2%	0.93	0.723	**0.238**	67.1%
`cook_spinach` (9)	644.6	164.9	74.4%	0.93	0.680	**0.213**	68.6%

Non-default rate weights at more views. Full-sequence six/nine-view `cook_spinach` runs also include 4DGC at $\lambda_{\text{grid}}=10^{-4}$ (Table 9). Over all 299 P-frames, the six-view baseline uses 0.0174 MiB/frame at 29.415 dB, whereas default-point lossless prediction uses 0.2378 MiB/frame at 29.585 dB. For the *same first* 130 *P-frames* as six-view occupancy-aware E2E, the tuned baseline uses 0.0174 MiB/frame at 29.512 dB, versus 0.0653 MiB/frame at 29.627 dB for E2E: 3.75× the payload with 0.116 dB higher PSNR. These measurements trade higher PSNR for a larger payload. The six/nine-view prediction and E2E results cover the default rate weight only. A BD-rate comparison requires a shared quality interval, which one full operating point per main variant does not define; the pruning sweep is a separate alternative.

A.1 RATE-DRIVEN INTRA/INTER SWITCHING

An optional probe compares the coded bytes of a pruned intra grid with an inter residual against its actual decoded state $\bar{S}_{t-1}$:

$$m_t = \arg\min_{m \in \{\text{intra,inter}\}} B_t(m), \quad B_t(\text{intra}) = |\text{AE}(\text{P}(S_t))|, \quad B_t(\text{inter}) = |\text{AE}(T_t - \bar{S}_{t-1})|. \tag{7}$$

Here $T_t = S_t$ in the unpruned chain and $T_t = \text{P}(S_t)$ after a pruned state is selected; the decoded target becomes the next state. The probe evaluates both pruned and unpruned initial chains and retains the cheaper measured sequence. At the default weight, prediction wins after the first frame on two 130-P-frame pilots. At higher weights intra coding can be cheaper. The decision minimizes current-frame bytes; future costs depend on the selected state. Reported payloads exclude the mode flag and container framing.

B END-TO-END RESIDUAL-DOMAIN TRAINING

§4.4 describes the residual-rate objective and occupancy-aware initialization. This appendix compares training variants against each scene's 4DGC run over the same frames. The plain temporal variant inherits the previous grids, MLP, and entropy-model parameters. The occupancy-aware variant inherits only unread grid cells; queried cells, the MLP, and entropy models retain the warm-up initialization. Residual-rate-only training changes the objective while retaining the full warm-up initialization. Thus the plain-versus-occupancy comparison evaluates these complete initialization strategies.

Results. Table 5 separates plain temporal initialization from the occupancy-aware variant used in the main results. At the default rate weight, the plain variant reduces payloads substantially on five three-view scenes, but the nine-view run loses 0.08 dB and two configurations drift severely. On the full 299 P-frames of `cook_spinach`, the plain variant gives 0.041 MiB at 21.39 dB and 0.023 MiB at 21.35 dB when the rate weight is increased. For comparison, tuning 4DGC gives 0.020 MiB at 21.33 dB (§5.4). The occupancy-aware variant used throughout Table 1 costs 0.084 MiB at 21.39 dB on this scene. §5.5 discusses the 130-P-frame training comparisons; Table 5 provides the complete matched-frame measurements.

Failure mode and its fix. Plain temporal initialization loses 1.75 dB on `flame_salmon_1` and 0.65 dB on six-view `cook_spinach` over 130 P-frames. The quality losses are consistent with inherited motion errors accumulating in the reference positions. Occupancy-aware initialization restricts this inheritance to unread cells, while queried cells restart from the warm-up. In these two runs, its mean PSNR changes are $+0.01$ and -0.02 dB, at 0.069 and 0.065 MiB per P-frame (90–91% below the default baseline). Residual-rate-only training also avoids the large drift, at 0.074 and 0.076 MiB, although the salmon LPIPS is worse by 0.010. A GOP-10 grid refresh leaves a 1.08 dB loss on salmon, consistent with errors remaining in the reference positions. On three-view `cook_spinach`, occupancy-aware initialization costs 0.087 MiB versus 0.045 MiB for plain initialization over the same 130 P-frames. We use occupancy-aware initialization for every maintable scene and retain the cheaper plain variant as an ablation.

C PREDICTION OF THE MOTION GRID AS I/P CODING

Temporal prediction is central to video coding: inter-coded blocks use a residual relative to a motion-compensated reference, exploiting similarity between frames (Wiegand et al., 2003; Sullivan et al., 2012); learned codecs use motion and residual coding (Lu et al., 2019) or condition feature coding on a temporal context (Li et al., 2021). Streamable 4DGS codecs (Sun et al., 2024; Hu et al., 2025) apply the idea to the *Gaussians*: the keyframe is the I-frame, and each P-frame transmits motion that warps the previous reconstructed Gaussian set plus a residual set. In our 4DGC baseline, the motion grid is independently coded every frame: the grid of frame t is initialized from the same warm-up in the baseline, trained for a few hundred iterations, quantized, and range-coded on its own. §3.2 measures the temporal redundancy remaining in these independently coded grids. Many cells are unqueried at the sampled frames, and consecutive grids are empirically highly correlated.

Table 5: End-to-end residual-domain training against the paper's 4DGC run of each scene over the *same* frames (matched baseline metrics; plain initialization unless a row specifies another variant); the E2E payload includes the transmitted residual stream (all three levels and their tables, intra first frame, native fp32 MLP, and residual Gaussian attributes). 130-P-frame runs unless noted; E2E $\lambda_{\text{grid}}=10^{-5}$ unless noted; the matched baseline always uses 10^{-5}. Bold payloads are below baseline; quality is reported separately.

Model	PSNR	SSIM	LPIPS	MiB/frame	ΔPSNR	Saving
`cook_spinach` (3), plain, 299 P-frames						
4DGC	21.365	0.8070	0.2245	0.708	–	–
E2E	21.387	0.8069	0.2240	**0.041**	+0.02	94.2%
`cook_spinach` (3), 3×10^{-5}, plain, 299 P-frames						
4DGC	21.365	0.8070	0.2245	0.708	–	–
E2E	21.348	0.8059	0.2248	**0.023**	−0.02	96.7%
`sear_steak` (3), plain, 299 P-frames						
4DGC	21.904	0.8028	0.2086	0.686	–	–
E2E	21.912	0.8028	0.2090	**0.024**	+0.01	96.5%
`cook_spinach` (3), plain, 130 P-frames						
4DGC	21.326	0.8067	0.2219	0.709	–	–
E2E	21.341	0.8065	0.2216	**0.045**	+0.01	93.7%
`flame_steak` (3), plain, 130 P-frames						
4DGC	22.251	0.8235	0.2094	0.724	–	–
E2E	22.263	0.8230	0.2097	**0.035**	+0.01	95.2%
`coffee_martini` (3), plain, 130 P-frames						
4DGC	17.330	0.6709	0.3685	0.687	–	–
E2E	17.321	0.6712	0.3701	**0.026**	−0.01	96.3%
`sear_steak` (3), plain, 130 P-frames						
4DGC	21.948	0.8044	0.2053	0.689	–	–
E2E	21.953	0.8044	0.2056	**0.028**	+0.00	95.9%
`cut_roasted_beef` (3), plain, 130 P-frames						
4DGC	20.208	0.7880	0.2383	0.713	–	–
E2E	20.185	0.7858	0.2427	**0.084**	−0.02	88.2%
`cook_spinach` (9), plain, 130 P-frames						
4DGC	28.588	0.9306	0.1067	0.681	–	–
E2E	28.512	0.9296	0.1087	**0.031**	−0.08	95.5%
`cook_spinach` (6), plain, 130 P-frames						
4DGC	29.649	0.9182	0.1122	0.723	–	–
E2E	29.004	0.9079	0.1344	**0.141**	−0.65	80.4%
`flame_salmon_1` (3), plain, 130 P-frames						
4DGC	17.818	0.6877	0.3552	0.688	–	–
E2E	16.067	0.6384	0.4881	**0.131**	−1.75	81.0%

This motivates an additional prediction step: code the grid as the residual against the previously decoded grid.

Three properties support temporal prediction of this grid. First, the reference is the *decoded* integer grid S_{t-1}, which encoder and decoder hold identically, so prediction introduces no numerical drift in this grid: $S_t = S_{t-1} + \Delta_t$ is integer arithmetic and reproduces the codec's own quantized grid bit for bit (verified by decoding every stream, Appendix A). Second, the fixed scene-box layout permits direct subtraction of corresponding grid coefficients. This predictor uses no motion compensation of the grid; its residual is non-zero on only a few percent of coefficients in the measured sequences (Appendix A). Third, the entropy coder does not change: the codec's empirical frequency table adapts to whatever is coded, and the residual's table is simply far more peaked. The analogy also

Table 5: End-to-end residual-domain training (continued). Initialization comparisons use the same keyframe and warm-up; accounting and boldface follow the preceding panel.

Model	PSNR	SSIM	LPIPS	MiB/frame	ΔPSNR	Saving
`cook_spinach` (3), rate only, 130 P-frames						
4DGC	21.326	0.8067	0.2219	0.709	–	–
E2E	21.341	0.8067	0.2214	**0.100**	+0.02	85.8%
`flame_salmon_1` (3), occupancy-aware, 130 P-frames						
4DGC	17.818	0.6877	0.3552	0.688	–	–
E2E	17.830	0.6881	0.3532	**0.069**	+0.01	90.0%
`flame_salmon_1` (3), rate only, 130 P-frames						
4DGC	17.818	0.6877	0.3552	0.688	–	–
E2E	17.768	0.6865	0.3654	**0.074**	−0.05	89.2%
`flame_salmon_1` (3), GOP-10 refresh, 130 P-frames						
4DGC	17.818	0.6877	0.3552	0.688	–	–
E2E	16.739	0.6624	0.4349	**0.078**	−1.08	88.7%
`cook_spinach` (6), occupancy-aware, 130 P-frames						
4DGC	29.649	0.9182	0.1122	0.723	–	–
E2E	29.627	0.9178	0.1124	**0.065**	−0.02	91.0%
`cook_spinach` (6), rate only, 130 P-frames						
4DGC	29.649	0.9182	0.1122	0.723	–	–
E2E	29.643	0.9179	0.1125	**0.076**	−0.01	89.5%
`cook_spinach` (6), GOP-10 refresh, 130 P-frames						
4DGC	29.649	0.9182	0.1122	0.723	–	–
E2E	29.614	0.9178	0.1122	**0.072**	−0.03	90.1%
`cook_spinach` (3), occupancy-aware, 130 P-frames						
4DGC	21.326	0.8067	0.2219	0.709	–	–
E2E	21.327	0.8066	0.2219	**0.087**	+0.00	87.8%
`cook_spinach` (3), occupancy-aware, 299 P-frames						
4DGC	21.365	0.8070	0.2245	0.708	–	–
E2E	21.387	0.8072	0.2243	**0.084**	+0.02	88.2%
`sear_steak` (3), occupancy-aware, 299 P-frames						
4DGC	21.904	0.8028	0.2086	0.686	–	–
E2E	21.887	0.8020	0.2084	**0.047**	−0.02	93.2%

brings the usual cost of P-frames. A decoder must start from an intra grid and decode every residual in order, so seeking and error recovery require both suitable grid state and Gaussian reference state. An intra grid resets only the grid predictor. Complete random-access and recovery protocols are not evaluated; we therefore report the intra (pruning) and inter (prediction) savings separately.

D Implementation details

The transmitted motion-grid stream. Each P-frame transmits three grid levels of 4×32^3, 4×64^3, and 2×128^3 coefficients (0.13M, 1.05M, and 4.19M), the 1799-parameter MLP in its native fp32 precision (7196 bytes), and the residual Gaussians. Coefficients are quantized by rounding (step q=1), and each level is range-coded as one stream with an empirical frequency table over its integer symbols; the table (minimum symbol, cumulative counts, length) is transmitted with the stream and is counted in every coded payload we report ($\sim$100 bytes per level and frame). During training, a learned factorized prior per level (Ballé et al., 2018) provides a differentiable rate surrogate; it is not transmitted and takes no part in encoding. For the grid terms in Eqs. 1 and 5, R denotes the implementation's sum over levels of $-\sum \log_2 p/(CHW)$ plus 0.001 times the mean squared quantization error, for a level shaped (C, D, H, W). Thus R includes a quantization regularizer and

is not the actual coded byte count. The grid is read at the sinusoidally encoded contracted positions of the reference Gaussians: with $u \in [0,1]^3$ the position contracted by the fixed scene box (the 5th–95th percentile box of the keyframe, stored with the warm-up), level ℓ is sampled twice, at $\sin(2^\ell \pi u)$ and $\cos(2^\ell \pi u)$, by trilinear interpolation with zero padding outside the grid, and the 20 interpolated values feed the MLP that outputs the displacement and rotation. Gaussians outside the box receive zero motion. The occupancy in §4.3 is therefore the union, over Gaussians and over the two samples per level, of the eight-cell cubes of these reads, and it is computed by the decoder-visible positions alone.

Priors and support masks. MVSFormer (Cao et al., 2022) at 768×576 with the three training views (one reference and two source views); Depth Anything V2 (metric Hypersim ViT-L weights, 20 m maximum depth) (Yang et al., 2024a); dynamic masks from thresholded frame differences. Cross-view consistency is $c_p = \frac{1}{N_F} \sum_t \sum_{n' \neq n} \mathbf{1}[\text{in range}] \exp\big(-(\xi_p + \beta\, \xi_d)\big)$, where pixel p of view n is projected with its MVS depth into each other training view n' and back, ξ_p is the resulting pixel error scaled by W_{ref}/W (W_{ref}=1352), ξ_d the relative depth error, β=200, and the average runs over the N_F=300 frames, so this is a temporal average of per-frame cross-view agreement, not a direct temporal-consistency test (with three views $c_p \leq 2$); the support mask keeps the top 60% of c_p among pixels with MVS confidence ≥ 0.5, also requiring positive depth, when the fixed threshold τ=1.8 admits fewer than 15% of the pixels (it admits 0% on all our scenes). The fused prior point cloud ($\approx 10^5$ points) initializes our 4DGC/GR4DGS keyframes and the QUEEN reproduction; other representations follow their own initialization procedures. These sequence-averaged masks use all 300 frames, so preprocessing has future-frame access; streaming here refers to sequential representation and decoding, not causal online capture.

Keyframe training. The main 4DGC/GR4DGS comparisons share the photometrically trained keyframe; additional depth-supervision losses are disabled.

Cameras. Neural 3D Video camera 0 is the test view throughout. The three training views are cameras 1, 11, 20 (1, 9, 17 on `coffee_martini`, whose rig has 18 cameras), spaced uniformly over the rig; six views use cameras 1, 5, 9, 12, 16, 20 and nine views 1, 3, 6, 8, 11, 13, 15, 18, 20 (`cook_spinach`).

P-frame configuration. The 4DGC-family runs use the same P-frame settings: 400 motion-field and 100 residual iterations, residual spawning enabled with its default gradient threshold, opacity floor 0.01. λ_{res}=0.02; λ_{grid} as stated.

Occupancy. At each level, occupancy is the union of the $2 \times 2 \times 2$ cubes around the two encoded reference positions per Gaussian. These masks determine the initialization of Eq. 6.

PSNR aggregation. The archived 4DGC-family, 3DGStream and GC-4DGS evaluators average the three channel-wise PSNR values, $\frac{1}{3}\sum_c 10\log_{10}(1/\text{MSE}_c)$. The QUEEN and TeTriRF evaluators instead use joint-RGB PSNR, $10\log_{10}(1/(\frac{1}{3}\sum_c \text{MSE}_c))$. These conventions differ; the corresponding frozen values are labeled in the tables and are not ranked against one another. Cross-convention quality comparisons use LPIPS. SSIM and LPIPS use the same implementations across methods.

QUEEN. The public code and its DyNeRF configuration were run on the same three training views and test view (camera 0), 300 frames at half resolution, initialized from the same fused prior point cloud as our 4DGC reproduction; its MiDaS monocular-depth initialization and depth loss were disabled (no MiDaS weights are used by any method here) and the adaptive per-frame iteration schedule, which needs a precomputed frame-difference file, was left at the fixed published epochs. PSNR, SSIM, and LPIPS are computed with our metric code on its saved test renderings over the 299 P-frames; the size is QUEEN's own per-frame estimate (entropy of the gated quantized latents plus decoders), converted to MiB, with its 10–15 MiB first frame excluded.

3DGStream and GC-4DGS. Both use the same three training views, held-out camera, and frames 1–299 for quality. 3DGStream size is the measured uncompressed transformation cache plus added

Gaussians per P-frame. GC-4DGS size amortizes the recorded fp32 model parameters over 300 frames; it is an offline sequence model, not a P-frame bitstream. Its `coffee_martini` quality is available, but its stored model size was not verified and is left blank. Neither representation has a completed `cut_roasted_beef` or `flame_salmon_1` evaluation in our archive.

TeTriRF. The public code (Wu et al., 2024b) with the authors' Neural 3D Video configuration at half resolution (loader factor 2), our three training views and test camera, and the authors' schedule: frame 0 alone, then groups of 20 frames with one overlapping frame. Each scene was run on its first 100 frames (five groups); all six scenes completed and every method is evaluated on those frames in Table 6. Evaluation frames match, but temporal input access does not: our 4DGC-family and QUEEN fused initialization uses the 300-frame masks described above, whereas TeTriRF was trained on 100 frames. The stream is produced by the authors' `compression.py` at its default quality (H.265, `qp` 20, 16-bit tiled feature planes; the density canvas enlarged to 4320×7680, the size its decoder assumes), the test view is rendered from the *decoded* stream (every frame rebuilt from the decoded planes and its group MLP), and PSNR, SSIM, and LPIPS are computed with our metric code over frames 1–99. Size is the authors' accounting, bytes of the four plane and density videos divided by 100; the per-group RGB MLPs add about 0.005 MiB per frame and are not counted.

Accounting. P-frame payload is the measured range-coded grid files and their frequency tables, residual Gaussian attributes and side information, plus the 1799 unchanged fp32 MLP parameters (7196 bytes). E2E includes its own residual Gaussian attributes (up to 919 mean bytes per frame across the six main runs). The first P-frame is intra-coded. This is coded-payload accounting: PyTorch training checkpoints, transport/container framing, and shared configuration metadata are not a measured delivery format. Grid round trips are bit-exact; E2E is a separate training run and is not called lossless. Table 2 adds the actual shared uncompressed keyframe once; the 15.9–27.9 MiB range is scene dependent. Raw CSVs, artifact sizes and precision checks are archived with the reproducibility scripts.

D.1 Warm-up selection

We seed warm-up training and reject a warm-up when the mean absolute coefficient of any level exceeds 1.0 (two of 30 seeds, at 2.4–2.5× the accepted-seed mean payload; Appendix E); the main 4DGC/GR4DGS pairs use the same accepted warm-up; the seed study explicitly includes rejected warm-ups.

E Additional results

Combining pruning and prediction. We zero unqueried cells in the same decoded grids, verify unchanged trilinear reads at every reference position, and predict the pruned grids (Appendix D). On the three 130-P-frame runs with retained reference point clouds, this combination saves 60–68% of P-frame payload, versus 65–71% for prediction alone (Table 7). Two effects help interpret this result. Unqueried cells already have few non-zero residuals (1.1–1.3%, versus 7.9–10.9% in queried cells), leaving little to remove. Pruning also resets cells outside the reference footprint to zero, requiring their full value when they become queried again. GR4DGS therefore predicts unpruned grids and uses pruning for intra-coded alternatives.

View-count comparison. With six training cameras, the `cook_spinach` sequence reaches 29.6 dB instead of 21.4 dB, while the default P-frame payload stays similar (0.72 vs. 0.71 MiB). Fig. 6 illustrates this reconstruction change. Lossless prediction preserves each run's rendering inputs; Table 4 reports its byte savings at three, six, and nine views.

Table 6: First 100 frames of every scene: quality uses the same P-frames 1–99. 4DGC, GR4DGS and GR4DGS-E2E report coded P-frame payload; QUEEN reports its own estimate. TeTriRF (Wu et al., 2024b) reports H.265 feature-plane and density video bytes divided by 100, excluding group MLPs (about 0.005 MiB/frame). QUEEN and TeTriRF use joint-RGB PSNR; the 4DGC family uses mean channel PSNR. Bold PSNR compares only the 4DGC family; bold LPIPS is best displayed. Bold sizes are below matched 4DGC. 4DGC and lossless GR4DGS share quality by construction.

Method	PSNR	LPIPS	MiB/frame
cook_spinach			
4DGC	21.29	**0.2208**	0.708
GR4DGS	21.29	**0.2208**	**0.253**
GR4DGS-E2E	**21.30**	**0.2208**	**0.087**
QUEEN	20.05	0.2343	0.659
TeTriRF	18.44	0.3467	0.067
flame_steak			
4DGC	**22.25**	**0.2091**	0.726
GR4DGS	**22.25**	**0.2091**	**0.226**
GR4DGS-E2E	**22.25**	0.2099	**0.067**
QUEEN	20.57	0.2293	0.668
TeTriRF	18.37	0.3597	0.063
sear_steak			
4DGC	**21.96**	**0.2043**	0.690
GR4DGS	**21.96**	**0.2043**	**0.204**
GR4DGS-E2E	**21.96**	**0.2043**	**0.054**
QUEEN	20.71	0.2231	0.632
TeTriRF	18.77	0.3598	0.060
coffee_martini			
4DGC	**17.33**	**0.3683**	0.690
GR4DGS	**17.33**	**0.3683**	**0.245**
GR4DGS-E2E	**17.33**	0.3688	**0.061**
QUEEN	15.48	0.4697	1.010
TeTriRF	15.23	0.5388	0.100
cut_roasted_beef			
4DGC	20.18	**0.2376**	0.712
GR4DGS	20.18	**0.2376**	**0.237**
GR4DGS-E2E	**20.19**	0.2382	**0.073**
QUEEN	19.38	0.2660	0.612
TeTriRF	18.43	0.3581	0.066
flame_salmon_1			
4DGC	**17.82**	0.3552	0.689
GR4DGS	**17.82**	0.3552	**0.244**
GR4DGS-E2E	**17.82**	**0.3419**	**0.070**
QUEEN	16.09	0.4117	0.958
TeTriRF	15.59	0.5174	0.100

The 30-seed warm-up study. Across 30 seeds on `cook_spinach`, two produced a maximum over levels of the mean absolute coefficient of 1.776 and 1.788 (the other 28: 0.712–0.786). Their payloads, averaged over the first 10 P-frames and including the native fp32 MLP, were 1.747 and 1.800 MiB/frame; accepted seeds ranged from 0.680 to 0.851, with mean 0.728. The two high-magnitude seeds therefore cost 2.40 and 2.47× this mean and lose 0.17 and 0.18 dB relative to the accepted-seed mean PSNR. The threshold 1.0 separates the two groups in this sweep and is used for the paired runs reported here.

Table 7: Post-hoc pruning and prediction on the same decoded grids (130-P-frame runs; 40–42% finest-level occupancy). Prediction preserves grids bit-exactly; pruning preserves all reference-position trilinear reads. KiB prediction means exclude the initial intra frame; P-frame payloads include it. Bold payloads are below independent coding.

Scene	Scheme	128^3 (KiB)	cut	P-frame (MiB)	cut
cook_spinach	codec (independent)	670.9	–	0.708	–
	pruning	378.8	43.5%	**0.422**	40.5%
	prediction	199.9	70.2%	**0.252**	64.5%
	pruning → prediction	232.6	65.3%	**0.285**	59.7%
sear_steak	codec (independent)	655.7	–	0.688	–
	pruning	353.4	46.1%	**0.392**	43.1%
	prediction	151.7	76.9%	**0.200**	70.9%
	pruning → prediction	176.6	73.1%	**0.221**	67.9%
cut_roasted_beef	codec (independent)	670.2	–	0.715	–
	pruning	367.2	45.2%	**0.417**	41.6%
	prediction	183.4	72.6%	**0.243**	66.0%
	pruning → prediction	218.8	67.4%	**0.276**	61.4%

Table 8: Sampled grid diagnosis on cook_spinach. "q."/"unq.": queried/unqueried cells. Entries average frames 1, 150, 299; the first uses keyframe positions, the latter two use that frame's warped positions as proxies because previous-frame point clouds were not retained. Bit attribution is $-\log_2 p$ under the learned prior, not a partition of actual range-coded bytes; unqueried cells receive 52–56% of the estimated rate across all levels. Exact occupancy-preserving recoding is evaluated separately in Table 7.

Level	Queried	Non-zero (q. / unq.)	Bits/coef. (q. / unq.)	Rate in unq.
32^3	42.7%	5.9% / 0.0%	0.43 / 0.02	5.8%
64^3	66.0%	8.1% / 0.5%	0.52 / 0.12	10.5%
128^3	41.5%	29.5% / 28.3%	1.30 / 1.27	58.0%

F Evaluation scope and extensions

Evaluation setting. The tests cover six indoor Neural 3D Video scenes, up to 300 frames at half resolution, and three, six, or nine training views. Each E2E configuration is evaluated once; the reported quality differences describe sequence means without confidence intervals. Applying prediction elsewhere requires a shared grid layout; occupancy pruning requires decoder-visible interpolation positions.

State and robustness. Each P-frame needs the preceding grid and Gaussian set, so seeking requires checkpoints of both. Resetting only the grid cannot repair accumulated position errors. Plain temporal initialization drifts in two of eight 130-P-frame configurations; occupancy-aware initialization avoids those observed cases (Table 5). Longer sequences, abrupt motion, seeking and packet-loss recovery remain untested.



Figure 6: Reconstruction at three and six training views on `cook_spinach`. The six-view run shows clearer dynamic regions and bottle labels; sequence-average PSNR is 29.6 dB versus 21.4 dB at three views. Lossless GR4DGS preserves the rendering inputs of its matched 4DGC run at either view count.



Figure 7: Ground truth vs. GR4DGS-decoded held-out view (frame 75, camera 0) on three scenes. Each pair shows the same camera and frame under three-view training; Fig. 6 compares three- and six-view reconstructions. GR4DGS's coding is lossless, so its rendering inputs are unchanged from 4DGC at the sequence-average P-frame payload shown, 65–72% below it.

Table 9: Verified 299-P-frame rate configurations, including native fp32 MLP payloads. Three views unless stated. Bold payloads are below 4DGC at the same scene, view count, and rate weight. Shorter prediction pilots and matched-frame six-view E2E comparisons are in Appendix A; §5.4 discusses the tuned baseline.

Method	λ_{grid}	PSNR	LPIPS	MiB/frame
cook_spinach				
4DGC	10^{-5}	21.365	0.2245	0.708
4DGC	3×10^{-5}	21.350	0.2248	0.069
4DGC	10^{-4}	21.325	0.2279	0.020
4DGC	3×10^{-4}	21.274	0.2314	0.014
4DGC + PRUNE	10^{-5}	21.373	0.2247	**0.420**
4DGC + PRUNE	3×10^{-5}	21.363	0.2254	**0.061**
4DGC + PRUNE	10^{-4}	21.328	0.2275	**0.019**
GR4DGS (prediction, lossless)	10^{-5}	21.365	0.2245	**0.248**
sear_steak				
4DGC	10^{-5}	21.904	0.2086	0.686
4DGC	3×10^{-5}	21.895	0.2085	0.048
4DGC	10^{-4}	21.852	0.2110	0.015
4DGC	3×10^{-4}	21.837	0.2142	0.012
4DGC + PRUNE	10^{-5}	21.907	0.2083	**0.387**
4DGC + PRUNE	3×10^{-5}	21.877	0.2090	**0.041**
4DGC + PRUNE	10^{-4}	21.880	0.2109	0.015
4DGC + PRUNE	3×10^{-4}	21.854	0.2138	0.012
cut_roasted_beef				
4DGC	10^{-5}	20.212	0.2429	0.715
4DGC + PRUNE	10^{-5}	20.208	0.2433	**0.415**
cook_spinach, 6 views				
4DGC	10^{-5}	29.585	0.1155	0.723
4DGC	10^{-4}	29.415	0.1217	0.017
GR4DGS (prediction, lossless)	10^{-5}	29.585	0.1155	**0.238**
cook_spinach, 9 views				
4DGC	10^{-5}	28.572	0.1121	0.680
4DGC	10^{-4}	28.381	0.1163	0.016
GR4DGS (prediction, lossless)	10^{-5}	28.572	0.1121	**0.213**