NEURIPS 2026 · AUTHOR SLIDES

# Phase-DGS

## Phase-guided Dynamic Gaussian Splatting from Unsynchronized Multi-view Video

Hosung Jeon[1†], Jun Young Jeong[1], Sangwoon Kwak[1†], Joonsoo Kim[1],
Sangmin Kim[2], Jaesik Park[2], Won-Sik Cheong[1], Hyon-Gon Choo[1]

[1]Electronics and Telecommunications Research Institute (ETRI), Daejeon, Republic of Korea
[2]Seoul National University, Seoul, Republic of Korea

† Co-corresponding authors

# Time offsets, frame drops, and freezes make 4D Gaussian Splatting reconstruct floaters and distorted geometry

### Temporal offset

Cameras start late or drift apart, so frame f shows a different instant on each camera.

| | f1 | f2 | f3 | f4 | f5 |
|---|---|---|---|---|---|
| Cam 1 | 1 | 2 | 3 | 4 | 5 |
| Cam 2 | 3 | 4 | 5 | 6 | 7 |

### Frame drop

Packet loss or buffer overflow removes frames independently on each camera; every later frame index slips.

| | f1 | f2 | f3 | f4 | f5 |
|---|---|---|---|---|---|
| Cam 1 | 1 | 2 | 3 | 4 | 5 |
| Cam 2 | 1 | 2 | 4 | 5 | 6 |

### Camera freeze

A stalled camera repeats its last frame while the scene keeps moving.

| | f1 | f2 | f3 | f4 | f5 |
|---|---|---|---|---|---|
| Cam 1 | 1 | 2 | 3 | 4 | 5 |
| Cam 2 | 1 | 2 | 2 | 2 | 5 |

Numbers show the scene instant each frame captures. Orange marks frames that no longer match Cam 1 at the same index.

**Result:** frames that share an index no longer show the same instant, so the radiance field fits floaters, ghosting, and distorted geometry. Hardware sync (Genlock, PTP) reduces this but needs costly wiring and still drops or drifts frames over unstable networks.

# Existing sync methods and their requirements

| Approach | Examples | Requires | Breaks down when |
|---|---|---|---|
| **Hardware / protocol sync** | Genlock, Timecode, PTP, NTP | Trigger wiring or a stable network | Signals degrade, packets drop, or clocks drift; no post-hoc fix |
| **Audio alignment** | Synchformer, audio cross-correlation | A controlled acoustic environment | Capture is open; sound delay grows across large rigs |
| **Geometry / feature matching** | VisualSync, feature-based sync | Well-textured features shared across views | Texture is weak, viewpoints differ widely, or features are occluded |
| **Human / pose tracking** | PoseSync, humans as calibration | A visible person in every frame | The scene has no person in view |
| **Reconstruction-coupled** | Sync-NeRF, SyncTrack4D | Known camera poses; one offset per camera | Cameras drop or freeze frames independently |

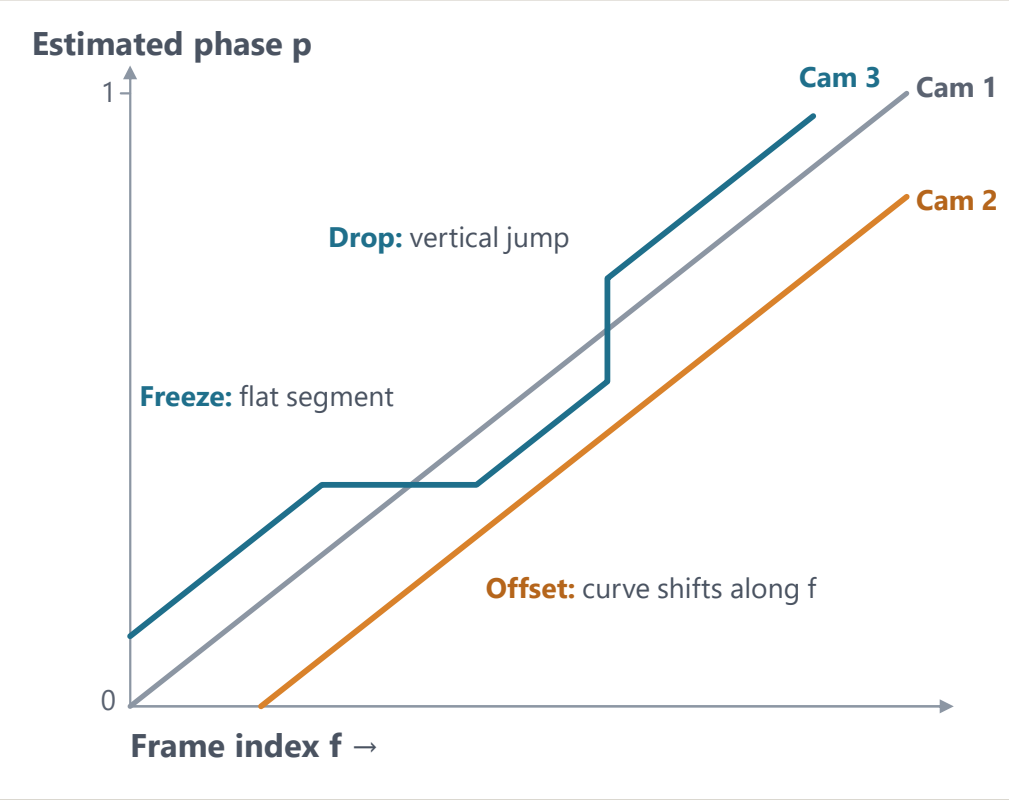
Source: Phase-DGS paper, Sections 1–2; example methods as cited there.

# Phase-DGS indexes each frame by its motion phase instead of its capture timestamp

1. **A phase is a frame's position in the motion**
   A scalar $p \in [0, 1]$ predicted from the image itself, not from the camera clock.

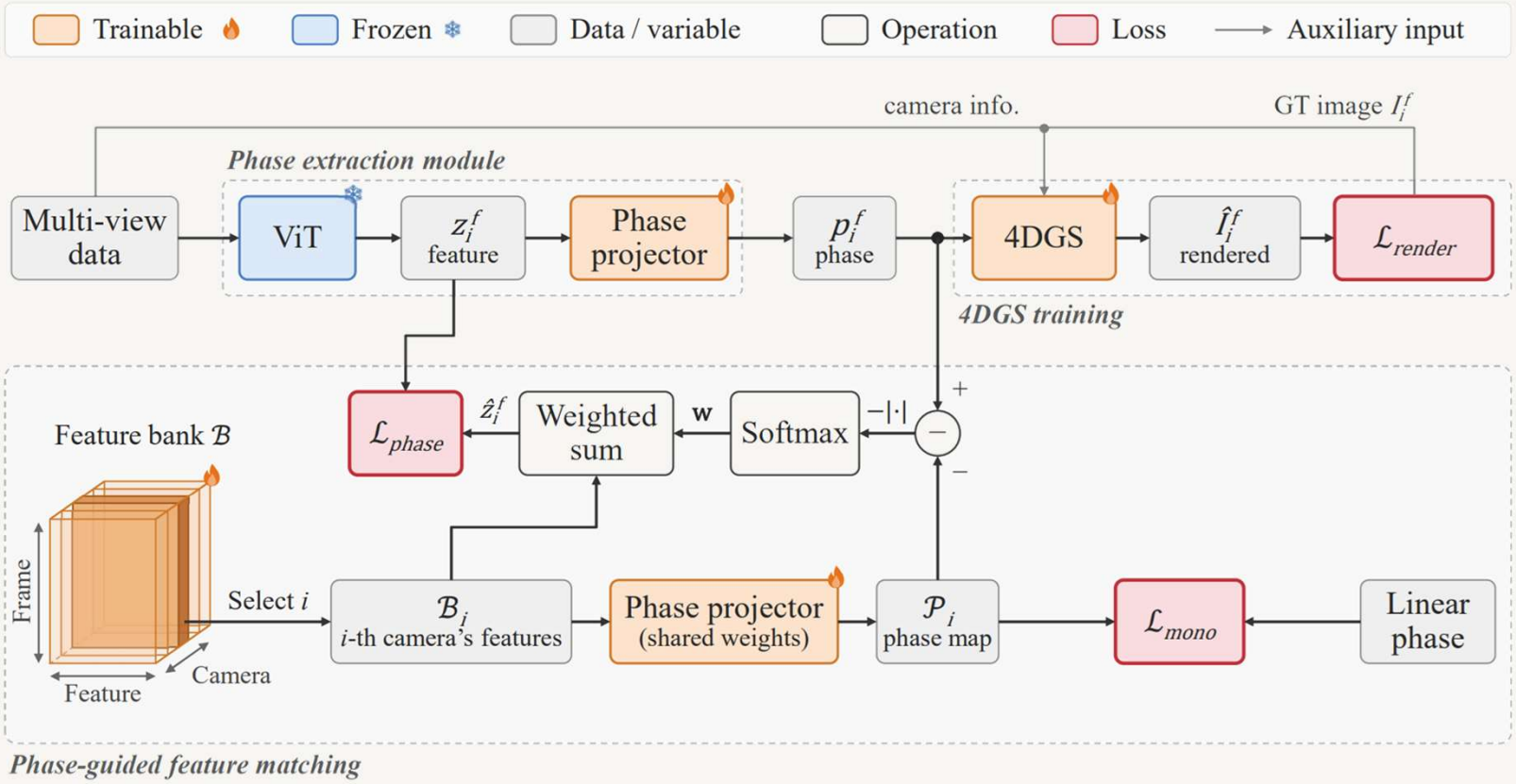
2. **Same motion state, same phase**
   A "jumping" frame gets the same p on every camera, whenever that camera recorded it.

3. **Phase replaces the timestamp**
   4DGS is trained as G(p) instead of G(t), so offsets, drops, and freezes stop corrupting supervision.




Schematic after Fig. 1(b) of the paper; not measured data.

# A frozen DINOv2 encoder and a small MLP predict each frame's phase, which replaces t in 4DGS




**At inference** only the backbone runs: it renders at a chosen phase, so rendering cost is unchanged.

# Three losses learn the phase without any phase labels

| | | |
|---|---|---|
| $L_{phase}$ **Phase consistency** | MSE($z$, $\hat{z}$)<br>$\hat{z} = \Sigma\ \mathrm{softmax}(-\lvert P_i - p\rvert) \cdot B_i$ | A frame's feature must match the bank entry at the same phase, so one motion state maps to one phase on every camera. |
| $L_{mono}$ **Monotonic** | $\Sigma\ \mathrm{relu}(1/(Fk) - \Delta)$<br>$\Delta = P_i(f+1) - P_i(f)$ over the bank | Each camera's phase map must rise by at least $1/(Fk)$ per bank step, so phase follows the order of the motion. |
| $L_{render}$ **Rendering** | Rendered $\hat{I}$ vs. ground-truth $I$<br>photometric loss of the 4DGS backbone | Trains the backbone and sends gradients back through p, so the projector learns phases that reconstruct the scene. |

**Total** $L = \lambda_{render}L_{render} + \lambda_{phase}L_{phase} + \lambda_{mono}L_{mono}$

$\lambda_render = 1.0 \cdot \lambda_phase = 0.1 \cdot \lambda_mono = 0.1$ (FreeTimeGS) or 0.01 (RealTime-4DGS)

# Phase is a scalar like t, so Phase-DGS plugs into existing 4DGS backbones unchanged

$G(t)$

**Conventional 4DGS**

t is the shared frame timestamp, which is wrong when cameras are out of sync.

↓

$G(p)$

**Phase-DGS**

p is predicted per camera and frame from the image, so no timestamp correction is needed.

**Tested with two state-of-the-art backbones**

**RealTime-4DGS**
implicit motion model

**FreeTimeGS**
explicit motion model

**+30%**

**GPU memory during training**
1,686 → 2,200 MB

**2.1×**

**training time per iteration**
148 → 70 iterations/s

**1.0×**

**rendering cost at inference**
The phase module is needed only for training; rendering uses no encoder, projector, or bank

Overhead vs. FreeTimeGS alone, measured on a 23-camera, 50-frame sequence on one RTX 5090 (paper Tab. A1).

# We test six scenes from two datasets, each under synced, shifted, and dropped capture

### Datasets

**Nvidia Dynamic Scene**
12 front-facing Full HD cameras
~100 frames per video
**Scenes:** Jumping, Playground, Umbrella

**SelfCap**
24 convergent 4K cameras
~300 frames per video
**Scenes:** Corgi, Yoga, Bike

### Capture conditions, increasing asynchrony

**Synced**
Dataset as provided
Reference for ideal capture.

**Shifted**
Each camera's start frame shifted at random
20–30 frame offsets between cameras.

**Dropped**
10% random frame drops plus 5–10 frame freezes
Varies independently per camera.

### Baselines

**4DGS w/o sync**
raw timestamps, lower bound

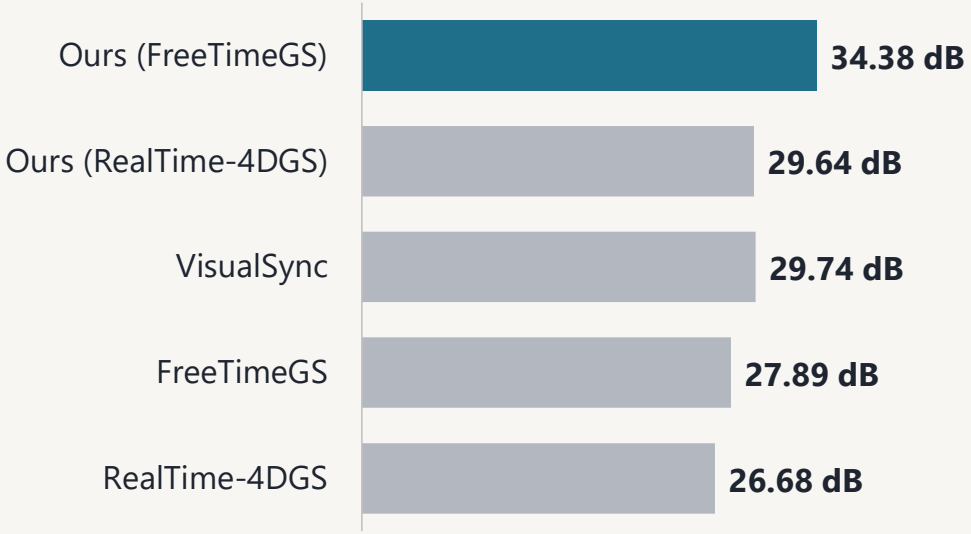
**VisualSync**
syncs videos, then trains 4DGS

**Sync-NeRF**
per-camera offsets on K-Planes

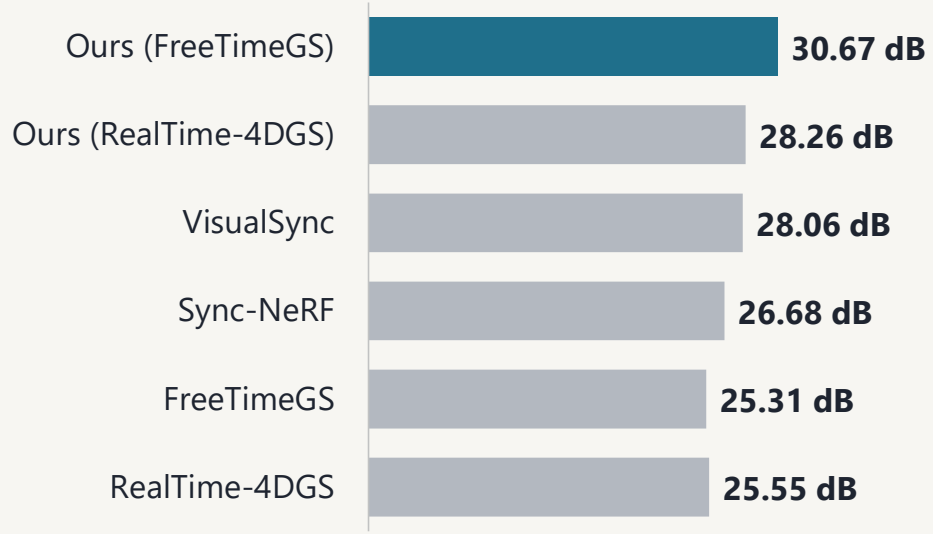
All images downsampled to 960×540. Metrics (PSNR, SSIM, LPIPS) on the held-out outermost camera, where timing errors show most.

# With frame drops and freezes, Phase-DGS beats the best baseline by 4.6 dB on SelfCap and 2.6 dB on Nvidia

**Average PSNR, SelfCap, Dropped condition**



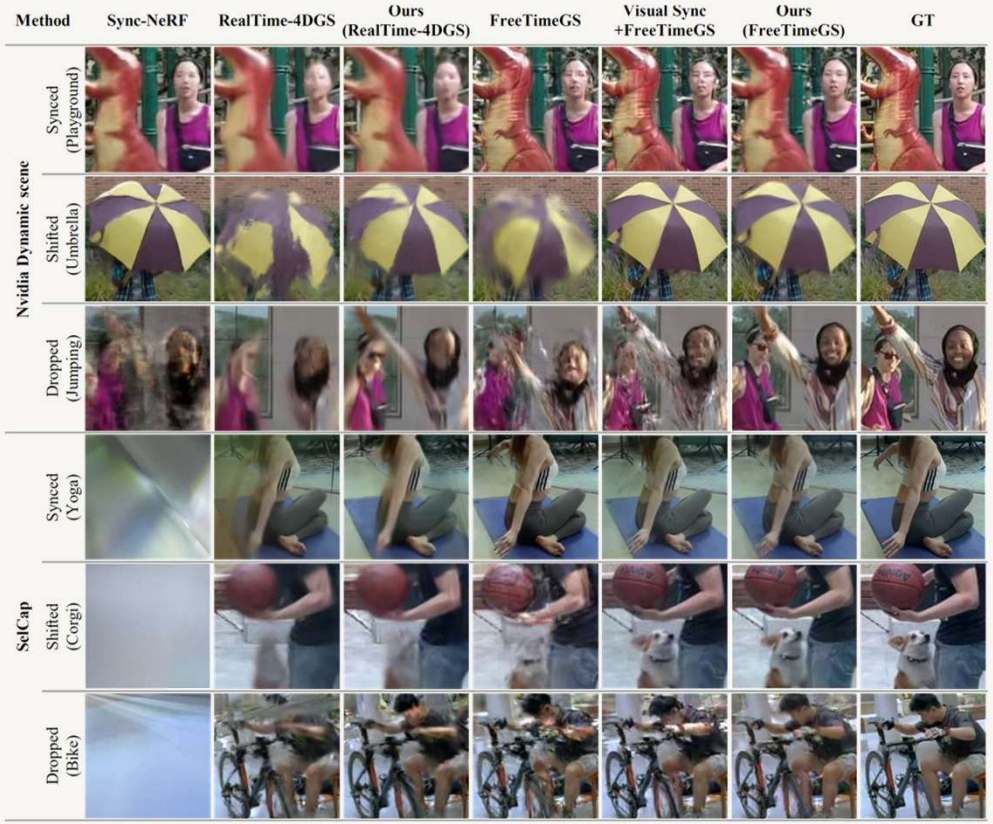

**Average PSNR, Nvidia Dynamic Scene, Dropped condition**




**Robustness:** on SelfCap, Phase-DGS stays within 0.4 dB of its synced score, while FreeTimeGS alone loses 6.3 dB.

Source: paper Tab. 1, averages over 3 scenes per dataset. Sync-NeRF is not reported on SelfCap because its default setup does not fit convergent cameras.

# Phase-DGS reduces visual artifacts such as ghosting and distortion, even on synchronized data




Nvidia: Jumping, Playground, Umbrella. SelfCap: Yoga, Corgi, Bike.

### Unsynchronized capture

Under Shifted and Dropped, the baselines duplicate or warp moving subjects; Phase-DGS with FreeTimeGS gives the cleanest renders.

### Small PSNR change, visible gain

With RealTime-4DGS the PSNR gain is small, but Gaussians in dynamic regions converge to one position and shapes look sharper.

### Even “synced” data improves

Backbones trained on data provided as synchronized still show artifacts that Phase-DGS reduces (paper Fig. 4).

# Conclusion

**Takeaways**

1. **Phase replaces timestamps**
   Per-frame phases learned from images let 4DGS train on unsynchronized video without hardware sync.

2. **Robust where baselines fail**
   Under drops and freezes, it beats the best baseline by 4.6 dB on SelfCap and 2.6 dB on Nvidia.

3. **Plug-and-play**
   Works unchanged with RealTime-4DGS and FreeTimeGS, and adds no rendering cost at inference.

4. **Limitations**
   Highly periodic motion gives different times similar phases (e.g., Balloon1), and quality follows the 4DGS backbone.

---

**Next steps:** backbone-agnostic training and inter-frame constraints for smoother phase transitions; reconstruction from archival footage.