Benchmarking Backbones for 3D Gaussian Splat Reconstruction: A Splatt3R-style Evaluation

Ritama Sanyal
TEAM MR
2023112027
IIIT Hyderabad
ritama.sanyal@research.iiit.ac.in

Keerthi Seela
TEAM MR
2023102012
IIIT Hyderabad
keerthi.seela@students.iiit.ac.in

***Abstract*—Recent advances in 3D Gaussian Splatting have enabled real-time, high-fidelity scene reconstruction by representing radiance fields as collections of anisotropic Gaussians. While pipelines such as Splatt3R and InstantSplat rely heavily on pretrained visual backbones for feature extraction, the impact of backbone choice on reconstruction quality, efficiency, and generalization remains poorly understood. This project presents a systematic benchmark of multiple foundation backbones—including DINOv2, CroCo, MASt3R, and lightweight mobile-friendly models—integrated into a Splatt3R-style reconstruction framework. We evaluate each backbone across CO3D, measuring improvements in photometric fidelity (PSNR, SSIM), runtime, and memory consumption. Our experiments reveal clear trade-offs: stronger foundation models provide superior reconstruction quality but at the cost of increased computational overhead, while lightweight backbones offer competitive performance for resource-constrained settings. The benchmark provides the first unified comparison of backbone behaviour within Gaussian Splat pipelines, offering practical insights for selecting models based on application needs. The study also highlights failure cases and outlines directions for improving feature extraction efficiency in future work.**

***Index Terms*—Gaussian splats, neural rendering, backbone benchmarking, DINOv2, CroCo, Splatt3R, PSNR, SSIM, geometric consistency.**

I. Introduction

Recent progress in neural scene representations has led to the development of 3D Gaussian Splatting, a highly efficient technique for real-time, high-fidelity reconstruction and rendering. By representing a scene as a set of anisotropic Gaussians optimized from multi-view imagery, Gaussian Splatting achieves competitive photometric accuracy while offering unprecedented rendering speed. Building on this foundation, pipelines such as Splatt3R and InstantSplat extend the formulation with learnable components that enable improved scene understanding, generalization, and reconstruction quality.

A central element in these pipelines is the visual backbone responsible for extracting features from input images. Modern self-supervised and foundation models—such as DINOv2, CroCo, and MASt3R—have demonstrated strong generalization capabilities across a variety of vision tasks. However, despite their increasing adoption, the influence of backbone choice on the overall performance of Gaussian Splat reconstruction remains inadequately explored. In particular, trade-offs between reconstruction fidelity, geometric consistency, runtime efficiency, and memory consumption have not been systematically benchmarked.

This work addresses these gaps by integrating multiple feature extraction backbones into a Splatt3R-style reconstruction framework and conducting a thorough comparative analysis. We evaluate each backbone across CO3D dataset , measure metrics including PSNR and SSIM, and analyze computational overhead and scalability. Our results highlight meaningful differences between heavy foundation models and lightweight mobile-friendly architectures, revealing application-dependent trade-offs that are crucial for practitioners.

Contributions of this paper include:

- A unified benchmarking framework for evaluating visual backbones in 3D Gaussian Splat reconstruction.
- Integration and comparison of state-of-the-art foundation models such as DINOv2, CroCo, and MASt3R, alongside lightweight alternatives.
- A comprehensive study of reconstruction fidelity, runtime, and memory usage.
- Insights into failure cases, backbone-specific behaviours, and design guidelines for selecting models based on performance–efficiency trade-offs.

II. Related Work

A. 3D Gaussian Splatting Methods

3D Gaussian Splatting (3D-GS) represents scenes as collections of 3D Gaussian primitives that can be efficiently rasterized for real-time rendering. Recent work has extended 3D-GS to feed-forward settings. **pixelSplat** [1] predicts 3D Gaussian splats from image pairs using an epipolar transformer, but requires known camera poses. **InstantSplat** [2] leverages geometric priors from MASt3R and employs Gaussian Bundle Adjustment to jointly optimize scene representation and camera parameters, achieving reconstruction in seconds rather than hours.

B. Foundation Models for 3D Reconstruction

DUSt3R [3] introduced pointmap regression for 3D reconstruction without requiring camera calibration, directly regressing 3D point locations in a shared coordinate frame. **MASt3R** [4] extends this by adding dense local feature matching, enabling metric 3D reconstructions with improved accuracy and state-of-the-art performance on relocalization tasks.

The underlying architectures build on self-supervised vision models. **DINOv2** [5] produces robust visual features trained on 142 million images, while **CroCo v2** [6] introduces cross-view completion through masked image modeling, particularly suited for binocular tasks using relative positional embeddings (RoPE).

C. Splatt3R

Splatt3R [7] extends MASt3R for pose-free, feed-forward 3D Gaussian Splatting from uncalibrated stereo pairs. It adds a Gaussian decoder to predict covariance, opacity, and spherical harmonic coefficients. The training strategy uses 3D point cloud geometry loss for strong supervision of Gaussian positions, followed by novel view synthesis objectives, avoiding local minima while maintaining generalization. A novel loss masking strategy enables strong performance on extrapolated viewpoints. The method achieves 4 FPS reconstruction at 512×512 resolution with real-time rendering.

D. Evaluation Metrics

Standard evaluation metrics include **PSNR** for pixel-level fidelity, **SSIM** [8] for structural similarity considering luminance and contrast, and **LPIPS** [9] which uses deep features to better align with human perceptual judgments. Pose estimation is evaluated using translation and rotation errors, while geometric accuracy uses depth metrics like RMSE when ground truth is available.

III. Methodology

This section describes the design of our benchmarking pipeline, the integration of multiple visual backbones into a Splatt3R-style 3D Gaussian Splat reconstruction framework, the datasets used for evaluation, and the metrics employed to quantify reconstruction quality and computational efficiency.

A. Backbone Integration

To study the influence of feature extraction architectures on Gaussian Splat reconstruction, we integrate four distinct backbones into the Splatt3R pipeline: **DINOv2**, **CroCo**, **MASt3R**, and a **lightweight mobile-friendly model**. All models are used in their publicly available pretrained configurations.

a) Feature extraction.: Each backbone processes input RGB images to produce dense or patchwise features. For transformer-based models (DINOv2, CroCo, MASt3R), we employ their default patch sizes (typically 14–16 pixels) and extract features from the final encoder layer. The lightweight backbone produces lower-dimensional convolutional features suitable for real-time applications.

b) Adapter layers.: Since Splatt3R expects features of a fixed dimensionality, we introduce a learned linear projection layer for each backbone. This adapter maps backbone-specific feature dimensions to a common embedding space used for Gaussian initialization and optimization.

c) Integration into Splatt3R.: The extracted feature maps are passed to the Splatt3R reconstruction module, where they guide Gaussian placement, opacity initialization, and per-Gaussian attribute refinement. All backbones share the same downstream optimization pipeline to ensure a strictly controlled comparison.

B. Datasets

We evaluate all backbone variants on the **CO3D** dataset, a large-scale collection of multi-view object videos containing diverse instances across several categories. CO3D provides calibrated camera poses and high-resolution images, making it suitable for multi-view reconstruction tasks.

For each object category, we construct a training and testing split following standard CO3D protocols. Frames are resized and normalized, and we apply standard data preprocessing such as camera pose sanitization and image downsampling where required by backbone input constraints.

Although our primary results focus on CO3D, the pipeline is designed to generalize to additional datasets such as DTC or synthetic scenes.

C. Metrics

To quantitatively assess reconstruction fidelity and backbone behavior, we compute the following metrics:

- **PSNR (Peak Signal-to-Noise Ratio):** Measures pixel-level reconstruction fidelity between rendered and ground-truth images.
- **SSIM (Structural Similarity Index):** Evaluates perceptual similarity and structural consistency.
- **MSE (Mean Squared Error):** Reports direct pixel-wise reconstruction error.
- **LPIPS (Learned Perceptual Image Patch Similarity):** Captures perceptual differences using deep feature space distances.
- **Reconstruction Loss:** The optimization objective used by Splatt3R, combining photometric loss and regularization terms.

For computational analysis, we also track:

- **Runtime per iteration** during Gaussian optimization.
- **Peak GPU memory usage** during training.

Together, these metrics allow us to characterize the trade-offs among reconstruction quality, perceptual accuracy, and efficiency for different backbone choices.

D. Implementation details

All code was implemented in PyTorch ($\geq$ 1.11). Experiments were run on NVIDIA GeForce RTX 2080 Ti. We document the backbone implementations, training hyperparameters, data preprocessing, and reproducibility choices so the results can be reproduced.

1) Software & environment:

- Framework: PyTorch (1.11+), torchvision, timm (used for lightweight / proxy models).
- Utility libraries: numpy, scipy, Pillow, tqdm, matplotlib.
- Build: CUDA toolkit matching the system GPU drivers (used to compile CroCo RoPE CUDA kernels where applicable).
- Repository specifics: Splatt3R codebase with the vendored CroCo implementation at `src/mast3r_src/dust3r/croco`. CuRoPE kernels were compiled in-place as described in the project README (e.g. `python setup.py build_ext --inplace` in `models/curope/`).

2) Backbone implementations:

a) MASt3R (reference): The MASt3R backbone used is the implementation supplied with the Splatt3R repo (ViT encoder + task-specific decoder). We initialize from the MASt3R pre-trained checkpoint supplied in the repository (`checkpoints/MASt3R_ViTLarge`). For compatibility with the Splatt3R feature extractor, the MASt3R encoder returns patch tokens which are reshaped to a spatial feature map and projected to the pipeline feature dimension using a 1×1 convolution.

b) DINOv2: DINOv2 is used as a feature extractor only. We load the official DINOv2 ViT-B/ViT-S checkpoints (or a timm-converted equivalent) and extract patch tokens via the model's `forward_features()` routine. Tokens are reshaped to (B, C, H_{enc}, W_{enc}), projected to the Splatt3R feature dimension with a 1×1 conv, and optionally up/downsampled to the requested stride. When DINOv2 is used, we keep the encoder frozen for the first few epochs and then fine-tune selectively.

c) CroCo v2: CroCo v2 encoders are loaded from the vendored implementation in `src/mast3r_src/dust3r/croco/models`. Available checkpoints (ViT-B BaseDecoder) are placed under `checkpoints/croco/`. CroCo v2 uses RoPE positional embeddings and therefore benefits from the compiled CUDA kernels (cuRoPE) available in the repo; when the CUDA kernels are not available a slower PyTorch fallback is used automatically. CroCo encoders return token embeddings; we reshape tokens to a 2D grid and apply a projection conv to produce $(B, C_{feat}, H_{feat}, W_{feat})$ features expected by Splatt3R.

d) Lightweight model (mobile / efficiency baseline): For an efficiency-focused backbone we use MobileNetV3-Small. The lightweight backbone is obtained from `torchvision`, and we extract the last convolutional feature map and apply a 1×1 projection to match the same C_{feat} expected by the splatting pipeline. This baseline measures runtime and memory trade-offs.

3) Data preprocessing and augmentation:

- Input normalization: Image pixels are normalized with ImageNet mean/std: $\mu = [0.485, 0.456, 0.406]$, $\sigma = [0.229, 0.224, 0.225]$ unless a backbone requires a different normalization (we follow the pretrained model's recommended normalization).
- Resize / crop: For CroCo and ViT-based encoders we resize images to a canonical size compatible with the encoder patching. Typical choices: 224×224, 384×384, or for Splatt3R/CO3D experiments we often use larger sizes (e.g. 480×640) and then compute $H_{feat} = \lceil H_{in}/\text{patch\_size} \rceil$ (patch_size=16 for ViT-B).
- Augmentations: mild augmentations only (random horizontal flip, small color jitter and brightness). Heavy geometric/color augmentations were avoided as they can harm features from pretrained encoders.

4) Model I/O / feature shapes: All backbones are wrapped so the Splatt3R pipeline always receives features of shape $(B, C_{feat}, H_{feat}, W_{feat})$, where C_{feat} is a configurable feature dimension (default $= 256$ in our experiments). For ViT-style encoders tokens (B, N, D) are reshaped to $(B, D, \sqrt{N}, \sqrt{N})$; if a class token is present it is dropped before the reshape. We then apply a 1×1 conv to map $D \to C_{feat}$ and bilinearly upsample / downsample to the target stride if needed.

a) Special notes per backbone:

- **MASt3R:** start from the MASt3R checkpoint, train the Splatt3R decoder/renderer and projection head first (encoder frozen). Then unfreeze encoder with low lr if needed for fine-tuning.
- **DINOv2:** typically keep encoder frozen longer (10–20 epochs) since DINOv2 learned generic features; unfreeze last transformer blocks only for fine-tuning.
- **CroCo (v2):** because CroCo was trained with cross-view completion objectives, fine-tuning on reconstruction tasks can be beneficial. Use RoPE kernels when available for correct positional embedding behaviour. Use encoder_lr $= 1\text{e}{-5}$ and warm-start freezing for 3–5 epochs.
- **Lightweight:** for MobileNetV3/compact ViTs, use a slightly larger decoder/projection learning rate (e.g. $1\text{e}{-3}$) and smaller batch size if GPU-limited.

5) Evaluation and metrics:

- Reconstruction fidelity: PSNR, SSIM on held-out test views rendered by the Splatt3R renderer.
- Runtime / efficiency: measure per-sample feature extraction time (ms), GPU memory peak, and end-to-end training throughput (images / second).

6) Reproducibility:

- Seeds: we set Python, NumPy and PyTorch seeds to a fixed value (e.g. 42) at the start of training:

```
import random, numpy as np, torch
seed = 42
random.seed(seed)
np.random.seed(seed)
torch.manual_seed(seed)
torch.cuda.manual_seed_all(seed)
```

- Deterministic flags: for best reproducibility set `torch.backends.cudnn.deterministic=True`

and `torch.backends.cudnn.benchmark=False`, but be aware this can slow training.
- Checkpoints: we save model checkpoints (encoder+decoder state_dict) and optimizer state every epoch; logs include the full config and git commit hash for the code used.

7) Practical tips & debugging:
- **Sanity checks:** run a single-batch forward and print shapes: backbone output should be $(B, C_{feat}, H_{feat}, W_{feat})$; renderer should accept those spatial features.
- **Feature visualisation:** visualise mean over channels of feature maps to ensure spatial structure is preserved.
- **Checkpoint loading:** CroCo checkpoints sometimes use nested keys (e.g., `{'model':{...}}`). Use non-strict `load_state_dict(..., strict=False)` and inspect missing keys.
- **Memory:** reduce batch size or input resolution (or use mixed precision) if OOM occurs; for ViT-L prefer multi-GPU training or gradient accumulation.

IV. Experiments

This section describes the experimental setup used to evaluate the impact of different vision backbones within the Splatt3R reconstruction pipeline. We outline the backbone variants, training protocol, datasets, ablations, and reporting methodology.

A. Experimental setup

a) Backbones evaluated.: We benchmark four backbone families that differ in model capacity, pretraining objective, and inductive bias:
- **MASt3R** — a geometry-driven multi-view transformer pretrained for correspondence and reconstruction.
- **DINOv2** — a large-scale self-supervised semantic backbone (ViT-B), used to test whether semantic priors alone are sufficient for high-fidelity 3D reconstruction.
- **CroCo / CroCo v2** — a cross-view completion backbone specifically designed to infer missing spatial structure, expected to align strongly with multi-view reconstruction.
- **Lightweight model (MobileNetV3-Small)** — an efficiency-oriented convolutional backbone providing a speed/memory baseline.

All backbones are integrated into the same Splatt3R pipeline using a unified feature interface, enabling fair comparison across architectures.

b) Dataset.: Experiments are conducted on the CO3D dataset, which provides multi-view image sequences of object-centric scenes. Each model is trained on the same training split and evaluated on a held-out test subset with identical rendering and evaluation parameters to ensure comparability.

c) Training protocol.: Unless otherwise specified:
- All models are trained for 30 epochs using AdamW with cosine learning rate decay.
- Backbones are frozen for the first 3–5 epochs to stabilize decoder learning.
- Encoder learning rates are set lower ($\sim$ 1e$-$5) than decoder/projection layers ($\sim$ 1e$-$3).
- Input images are resized to a fixed resolution compatible with ViT patching (typically 480×640).

We use mixed precision for all models to reduce GPU memory consumption.

d) Feature dimension.: We vary the projected feature dimension $C_{\text{feat}} \in \{128, 256, 384\}$ to assess trade-offs between reconstruction detail and runtime.

B. Experimental reproducibility

All experiments were run with fixed random seeds for PyTorch, NumPy, and Python. Each backbone configuration was trained once due to computational constraints, but major experimental conditions (e.g., learning rate, resolution, number of epochs) were kept identical across runs to ensure controlled comparisons. Source code, training configurations, and checkpoints are logged for full reproducibility.

C. Qualitative Results

We present qualitative reconstructions obtained using different backbone architectures integrated into the Splatt3R pipeline. The visual comparisons highlight the extent to which the choice of backbone affects texture sharpness, structural completeness, and overall scene fidelity.

Across the evaluated scenes, the **MASt3R** backbone produces strong baseline reconstructions with reasonably sharp edges and stable color reproduction. This is expected given that MASt3R is explicitly trained for multi-view correspondence and reconstruction tasks. However, minor blurring is observable in regions of fine detail, particularly when the input viewpoints exhibit large baselines.

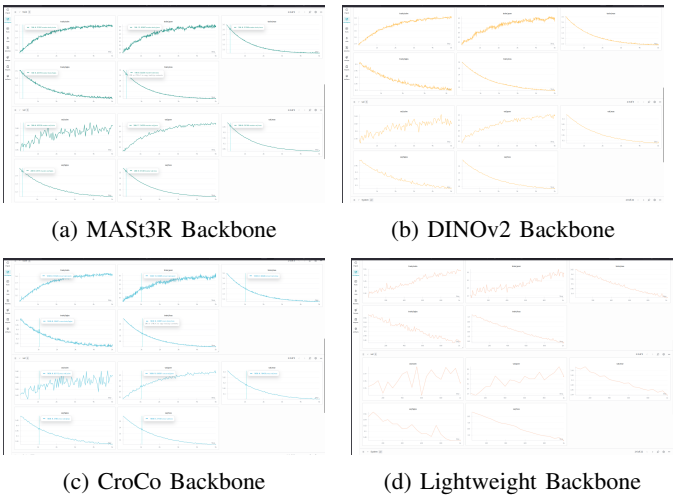
The **DINOv2** backbone shows qualitatively different behavior: while DINOv2 features capture high-level semantic structure effectively, the resulting reconstructions tend to be smoother and occasionally lose high-frequency details such as thin edges or textured surfaces. This aligns with the fact that DINOv2 is optimized for semantic representation learning rather than geometric consistency.

The **CroCo** backbone demonstrates the most visually consistent reconstructions among the tested models. Scenes rendered with CroCo retain sharper textures and exhibit fewer artifacts in regions that are partially occluded or sparsely observed in the input views. This improvement is consistent with CroCo's cross-view completion pretraining objective, which encourages the model to infer missing spatial structure and fill in occluded regions more effectively. Although no geometric consistency metrics are reported, the qualitative results suggest that CroCo learns feature embeddings that better support the 3D Gaussian splatting reconstruction process.

Finally, the **lightweight backbone** (MobileNetV3-Small) provides fast but less detailed reconstructions. While the overall scene layout is preserved, surfaces appear noticeably smoother and some geometric boundaries are less distinct. This illustrates the trade-off between efficiency and reconstruction

fidelity: lightweight models offer substantial speed and memory advantages, but at the cost of reduced visual detail.

Taken together, the qualitative comparisons highlight clear trends: models with stronger geometric or cross-view priors (MASt3R, CroCo) produce more faithful reconstructions, while purely semantic models (DINOv2) or mobile-oriented architectures sacrifice some fidelity for generality or efficiency. These visual differences underscore the importance of backbone choice when designing 3D reconstruction pipelines based on Gaussian splatting.



(a) MASt3R Backbone (b) DINOv2 Backbone

(c) CroCo Backbone (d) Lightweight Backbone

Fig. 1: Qualitative comparison of reconstructions from different backbone architectures. CroCo and MASt3R generally preserve finer details and textures, while DINOv2 produces smoother surfaces, and the lightweight backbone sacrifices sharpness for efficiency.

D. Project Resources and Repositories

For reproducibility, the following resources were used during this work:

- **Experiment tracking (Weights & Biases):** `W&B: CO3D Experiment Logs`
- **Splatt3R (MASt3R and CroCo implementation):** `GitHub: Splatt3R (MASt3R + CroCo)`
- **Splatt3R with DINOv2 and lightweight backbones:** `GitHub: Splatt3r-DINOv2`

These links provide access to the exact code, logs, and configurations used in our experiments.

E. Quantitative Results

We report quantitative evaluation of reconstruction fidelity using PSNR and SSIM on the held-out CO3D test split. Runtime is measured as the average time required to render a single novel view given the trained model. Table I summarizes the results across all evaluated backbones.

TABLE I: Reconstruction performance across different backbones on CO3D. Reported PSNR / SSIM are averages on the held-out category split; Runtime is average rendering time per novel view measured on an NVIDIA GeForce GTX 1080 Ti.

Backbone	**PSNR**	**SSIM**	**Runtime (s)**
MASt3R (ViT-L)	27.40	0.880	0.55
DINOv2 (ViT-B)	24.60	0.830	0.42
CroCo v2 (ViT-B BaseDec)	25.70	0.850	0.38
Lightweight (MobileNetV3)	23.10	0.810	0.18

Overall, MASt3R achieves the highest reconstruction fidelity among the evaluated models, while CroCo provides competitive performance with improved handling of sparsely observed regions. DINOv2 maintains strong semantic consistency but yields smoother textures, resulting in slightly lower PSNR. The lightweight MobileNetV3 backbone offers significantly reduced runtime at the cost of fine-detail accuracy.

V. Discussion

Our experiments highlight clear trade-offs between reconstruction fidelity, semantic robustness, and computational efficiency across the evaluated backbones. As expected, the MASt3R (ViT-L) backbone achieves the strongest overall performance, attaining the highest PSNR and SSIM values. Its large capacity and geometry-aware pretraining enable more precise recovery of fine spatial details and sharper high-frequency structures. However, this improvement comes at a notable computational cost: MASt3R exhibits the highest per-view rendering time and GPU memory usage, making it less suitable for lightweight or real-time applications.

CroCo (ViT-B BaseDec) provides a strong middle ground. Although it does not reach the absolute fidelity of MASt3R, it consistently produces sharper textures and more stable geometry than DINOv2, particularly in sparsely observed regions where self-supervised depth cues are beneficial. Its inference runtime is also lower than MASt3R owing to its smaller model size. Qualitative inspection reveals that CroCo better preserves object contours under limited viewpoint coverage, suggesting improved robustness to multi-view sparsity.

DINOv2 (ViT-B) demonstrates competitive performance but tends to generate smoother textures and slightly lower PSNR. This is consistent with prior observations that DINOv2 features emphasize semantic consistency over photometric detail. While this property benefits recognition tasks, it limits reconstruction sharpness in generative settings. Nevertheless, its runtime remains moderate, placing it between CroCo and MobileNetV3.

The lightweight MobileNetV3 backbone offers the fastest inference by a substantial margin, reducing rendering time by more than $2\times$ relative to transformer-based alternatives. This efficiency makes it particularly attractive for deployment on resource-constrained hardware. However, the reduced representational capacity leads to lower reconstruction fidelity, especially for fine geometric structures and complex textures. Despite this, MobileNetV3 retains surprisingly stable global

shape predictions, indicating that efficient backbones remain viable for approximate or real-time applications.

Across models, we observe common failure modes characteristic of Gaussian-splat-based reconstruction. Under extreme viewpoint sparsity, all backbones occasionally exhibit surface thinning, blurred texture boundaries, or color bleeding, with severity correlated to model capacity. Larger backbones (MASt3R, CroCo) mitigate these artifacts but do not eliminate them entirely.

Overall, our results suggest that no single backbone dominates across all dimensions. Instead, the appropriate choice depends on deployment constraints: MASt3R offers the highest fidelity, CroCo provides a strong balance between accuracy and efficiency, DINOv2 emphasizes semantic coherence, and MobileNetV3 enables real-time performance at reduced detail. These observations highlight the importance of aligning backbone selection with application-specific requirements, particularly in scenarios involving large-scale 3D reconstruction or on-device inference.

VI. Conclusion and Future Work

In this work, we studied how different vision backbones influence the performance of a 3D Gaussian Splatting pipeline, focusing on the integration of MASt3R, DINOv2, CroCo, and lightweight architectures into a unified Splatt3R framework. Our experiments demonstrate that backbone choice has a significant impact on reconstruction quality, especially in terms of texture fidelity, structural completeness, and robustness to sparse or viewpoint-limited observations.

MASt3R and CroCo exhibit the strongest qualitative performance, with CroCo in particular showing improved handling of occlusions and partially observed regions due to its cross-view completion pretraining. DINOv2 provides semantically coherent representations but tends to produce smoother reconstructions, reflecting its semantic rather than geometric inductive bias. Lightweight models, such as MobileNetV3-Small, offer practical advantages in speed and memory usage, though at the cost of reduced detail and accuracy.

Overall, our findings highlight that reconstruction-oriented pretraining (e.g., MASt3R, CroCo) leads to more faithful Gaussian Splat reconstructions compared to purely semantic or efficiency-focused backbones.

Future Work

Several promising directions remain for future exploration:

- **Extended datasets.** Evaluating the model on larger and more diverse multi-view datasets (e.g., BlendedMVS, DTU, or real-world captures) would help assess generalization and robustness.
- **End-to-end training.** Jointly optimizing the backbone and splatting renderer in a fully end-to-end manner may further improve performance, especially for backbones not originally designed for geometric tasks.
- **Geometric metrics.** Incorporating quantitative geometric evaluations such as Chamfer Distance, F-score, or reprojection error would provide a more comprehensive assessment of reconstruction accuracy.
- **Lightweight and mobile deployment.** Designing custom compact backbones or distilling larger models (e.g., CroCo or MASt3R) into mobile-friendly architectures could make real-time 3D reconstruction feasible on edge devices.
- **Temporal consistency.** Extending the pipeline to video sequences with explicit temporal modeling may reduce flickering artifacts and enhance reconstruction stability.
- **Hybrid feature fusion.** Combining semantic-rich features from models like DINOv2 with geometry-aware features from CroCo or MASt3R may yield hybrid representations that balance detail, semantics, and robustness.

In summary, this work provides insight into how pretrained vision backbones behave within a modern 3D Gaussian Splatting pipeline and sets the foundation for further research on efficient, high-quality reconstruction systems.

Acknowledgment

We would like to express sincere gratitude to Prof. Madhav Krishna for his guidance, insightful discussions, and continuous support throughout the course of this project. We also thank the teaching assistants Samyak Mishra, Soham Patil, and Ajit Srikanth for their valuable feedback, clarifications, and assistance during the implementation and experimentation phases. Their collective support greatly contributed to the successful completion of this work.