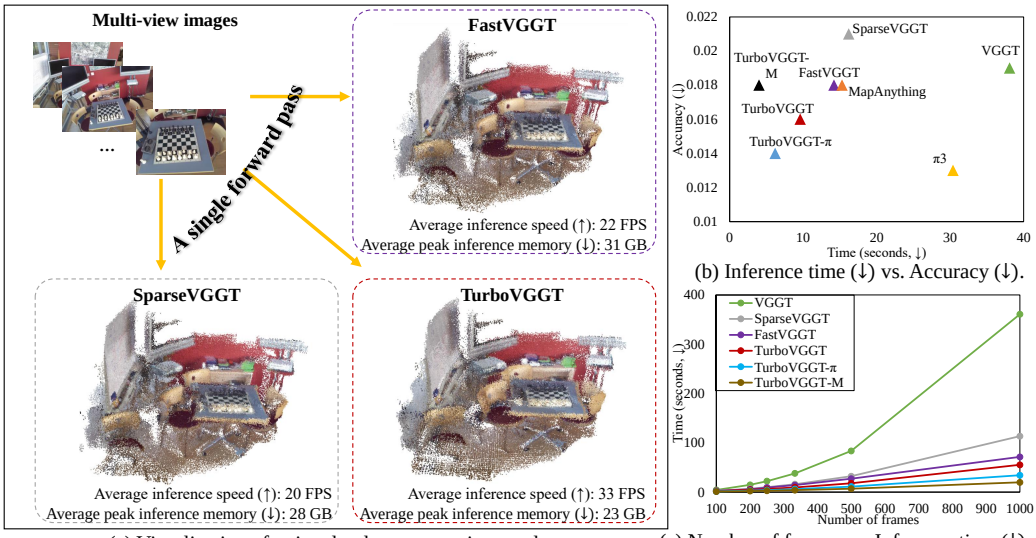
TurboVGGT: Fast Visual Geometry Reconstruction with Adaptive Alternating Attention

David Huang[1,2,*,†] **Guile Wu**[1,†] **Chengjie Huang**[1] **Bingbing Liu**[3] **Dongfeng Bai**[1]
[1]Huawei Noah's Ark Lab [2]University of Toronto [3]Foundation Model Department, Huawei
dawae.huang@mail.utoronto.ca, guile.wu@outlook.com, chengjie.huang@outlook.com,
liu.bingbing@huawei.com, baidongfeng@huawei.com
https://turbovggt.github.io/



(a) Visualization of point cloud reconstruction results.

(b) Inference time (↓) vs. Accuracy (↓).

(c) Number of frames vs. Inference time (↓).

Figure 1: An illustration of our TurboVGGT for fast multi-view 3D reconstruction. (a) Visualization of point cloud reconstruction results. TurboVGGT significantly improves computational efficiency in terms of inference time and peak GPU memory usage. (b) Comparison of inference time vs. point cloud accuracy on 7-Scenes (stride 3). TurboVGGT achieves a better trade-off between accuracy and efficiency compared to state-of-the-art methods. (c) Comparison of number of frames vs. inference time on 7-Scenes. TurboVGGT scales more efficiently with the number of input frames. TurboVGGT/TurboVGGT-π/TurboVGGT-M achieve $7\times/11\times/18\times$ speedup compared to VGGT for processing input sequences of 1000 frames.

Abstract

Recent feed-forward 3D reconstruction methods, such as visual geometry transformers, have substantially advanced the traditional per-scene optimization paradigm by enabling effective multi-view reconstruction in a single forward pass. However, most existing methods struggle to achieve a balance between reconstruction quality and computational efficiency, which limits their scalability and efficiency. Although some efficient visual geometry transformers have recently emerged, they typically use the same sparsity ratio across layers and frames and lack mechanisms

*David Huang contributed to this work during an internship at Huawei Canada.
†Equal contribution.

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

to adaptively learn representative tokens to capture global relationships, leading to suboptimal performance. In this work, we propose TurboVGGT, a novel approach that employs an efficient visual geometry transformer with adaptive alternating attention for fast multi-view 3D reconstruction. Specifically, TurboVGGT employs an end-to-end trainable framework that leverages adaptive sparse global attention guided by adaptive sparsity selection to capture global relationships across frames and uses frame attention to aggregate local details within each frame. In the adaptive sparse global attention, TurboVGGT adaptively learns representative tokens with varying sparsity levels for global geometry modeling, considering that token importance varies across frames, different attention layers operate on tokens at different levels of abstraction, and global dependencies rely on structurally informative regions. Extensive experiments on multiple 3D reconstruction benchmarks demonstrate that TurboVGGT achieves fast multi-view reconstruction while maintaining competitive reconstruction quality compared with state-of-the-art methods.

1 Introduction

Multi-view 3D reconstruction is a long-standing problem in computer vision. It aims to reconstruct the 3D geometry of a scene from a set of 2D images captured from different views. It has been widely used in various applications, such as robotics, autonomous driving, and virtual reality. Traditional methods, such as Structure-from-Motion (SfM) [33] and Multi-View Stereo (MVS) [34], typically employ a multi-stage paradigm to estimate camera poses and reconstruct 3D geometry. This paradigm is effective in producing accurate reconstructions but suffers from high computational costs.

Recently, learning-based 3D reconstruction [44, 42, 43, 22] has emerged as a promising alternative to the traditional paradigm, shifting from per-scene iterative optimization to direct inference of 3D geometry from images using end-to-end neural networks. In particular, the emergence of visual geometry transformer based feed-forward reconstruction methods, such as VGGT [42], MapAnything [22], and π^3 [46], have revolutionized multi-view 3D reconstruction, enabling efficient and robust estimation of camera poses, depth maps, and point maps in a single forward pass. Despite the significant progress, these methods still face challenges in achieving a balance between reconstruction quality and computational efficiency. This limits their scalability and hinders their application in time-sensitive scenarios. More recently, several studies [35, 47, 37, 41] have revealed that the high computational cost of visual geometry transformers primarily lies in the dense global token interactions within global attention layers, which involve quadratic complexity with respect to the number of input tokens. To resolve this problem, existing methods typically use token merging, sparse attention or token subsampling strategies to reduce the number of tokens involved in the global attention. Although these methods can accelerate visual geometry transformers for 3D reconstruction, they still have some limitations. First, they typically employ the same sparsity ratio across layers and frames. This may be suboptimal because token importance can vary across frames and attention layers operate on tokens at different levels of abstraction. Second, they lack mechanisms to adaptively learn representative tokens that can effectively capture global relationships. Since global dependencies often rely on structurally informative regions, learning representative tokens can facilitate global geometry modeling and reduce redundant computations.

To address these limitations, in this work, we propose a novel approach named TurboVGGT, which improves the efficiency of visual geometry transformers for fast multi-view 3D reconstruction. The key idea of TurboVGGT is to construct a visual geometry transformer with adaptive alternating attention blocks, which leverage adaptive sparse global attention guided by adaptive sparsity selection to capture global relationships and use frame attention to aggregate local details within each frame. Specifically, considering that token importance can vary across frames and attention layers operate on tokens at different levels of abstraction, TurboVGGT employs a gating network to adaptively assign different sparsity ratios for different frames across global attention layers. Then, in each branch corresponding to a specific sparsity level, TurboVGGT employs adaptive sparse global attention to learn compressed representative tokens for global geometry modeling. This enables TurboVGGT to effectively learn representative tokens to capture global relationships while reducing redundant computations. In addition, following existing works [42], TurboVGGT also incorporates frame attention to aggregate local details within each frame. We conduct extensive experiments on multiple 3D reconstruction benchmarks to evaluate the proposed TurboVGGT. Experimental results

demonstrate the superiority of TurboVGGT over state-of-the-art methods in achieving fast multi-view 3D reconstruction while maintaining high reconstruction quality (as illustrated in Figure 1).

In summary, our **contributions** are:

- We present a novel visual geometry transformer with adaptive alternating attention blocks for fast multi-view 3D reconstruction.
- We propose an adaptive sparsity selection mechanism for visual geometry transformers, which adaptively selects different sparsity ratios for different frames across different layers.
- We propose adaptive sparse global attention for visual geometry transformers, which learns representative tokens to model global geometry relationships.
- We conduct extensive experiments on multiple 3D reconstruction benchmarks and demonstrate the superiority of the proposed approach over state-of-the-art methods.

2 Related Work

2.1 Multi-View 3D Reconstruction

Reconstructing 3D scenes from multi-view images is a fundamental problem in computer vision. The traditional paradigm consists of multiple stages, such as feature extraction and matching, camera pose estimation, triangulation, and bundle adjustment. Typically, Structure-from-Motion [33, 30] is employed to estimate camera parameters and reconstruct sparse point clouds from multi-view images, and Multi-View Stereo [34, 14] is used to reconstruct dense geometry of 3D scenes. Although these methods can achieve high-quality reconstruction, they rely on per-scene iterative optimization, which is computationally expensive and time-consuming. Recently, learning-based methods have been proposed to directly predict 3D geometry from multi-view images in end-to-end neural networks [44, 42, 23, 22, 49, 46]. DUSt3R [44] is a pioneering work that employs transformer networks to directly predict dense 3D geometry from image pairs, but it requires costly global alignment when extending to large scenes due to the reliance on pairwise view processing. Subsequent works, such as MASt3R [23], CUT3R [43], Fast3R [49], DDUSt3R [16], and VGGT [42], have presented various strategies to resolve the limitations of DUSt3R for different problem settings, such as large scenes, long sequences, and dynamic scenes. Our work belongs to the learning-based paradigm, and we focus on improving the efficiency of visual geometry transformers for fast multi-view 3D reconstruction, which is a critical problem but has not been well explored in existing works.

2.2 Visual Geometry Transformers

Visual geometry transformers [42, 22, 46, 57] have recently emerged as powerful models for feed-forward multi-view 3D reconstruction. They can directly predict camera intrinsics, camera extrinsics, depth maps, point maps, and other geometric properties from multi-view images in a single forward pass. VGGT [42] is a pioneering work that constructs a visual geometry transformer and optimizes it with multi-task learning and large-scale datasets. π^3 [46] is a subsequent work that introduces a fully permutation-equivariant architecture to eliminate the reliance on a fixed reference view in VGGT. MapAnything [22] proposes a unified feed-forward transformer that supports various problem settings with different input and output configurations. Despite their impressive performance, they still face challenges in computational efficiency. This limits their scalability and efficiency in time-sensitive application scenarios. Although there have been some recent efforts [35, 47, 37, 41] to accelerate visual geometry transformers, they still have some limitations, such as relying on a fixed sparsity ratio for all layers and frames and lacking mechanisms to adaptively learn representative tokens for global geometry modeling. To address these limitations, our work proposes a novel visual geometry transformer with adaptive alternating attention blocks, which can adaptively select different sparsity ratios for different frames and learn representative tokens to model global geometry while reducing redundant computations, thus improving the efficiency of feed-forward 3D reconstruction.

2.3 Efficient Vision Transformers

Vision transformers [12] have achieved incredible success in computer vision. To make vision transformers more efficient, various strategies have been used, such as FlashAttention [11, 10],

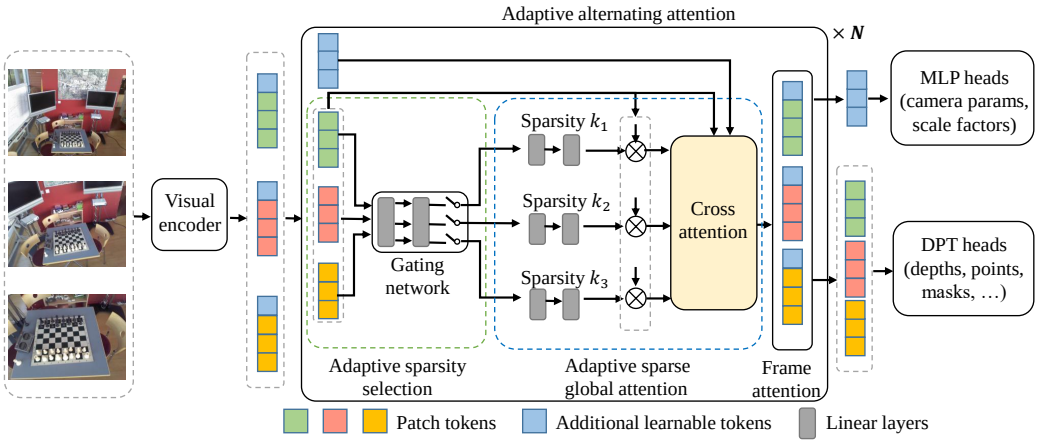


Figure 2: The overall framework of the proposed TurboVGGT.

sparse attention [48, 53], token merging [4, 52], and low-rank approximation [19, 15]. Despite their efficiency, most efficient vision transformers are designed for general 2D vision tasks. In the context of multi-view 3D reconstruction, designing an efficient visual transformer remains an open question, as it requires modeling complex global geometric relationships across views and local geometric details within views. With the success of visual geometry transformers for multi-view 3D reconstruction, there have been some recent efforts employing token merging [35], block-sparse attention [41], subsampling global attention [37], and compressed descriptors [47] to accelerate visual geometry transformers. Our work belongs to this line of research, but unlike existing works, we propose to accelerate visual geometry transformers with adaptive alternating attention blocks. In addition, DiffRate [6] learns differentiable layer-wise compression rates for conventional 2D vision transformers, while GitMerge3D [39] performs graph-based token merging for transformers operating on explicit 3D point clouds. In contrast, TurboVGGT targets multi-view image-based geometry transformers and adaptively allocates sparsity across frames and layers while learning representative tokens for cross-view geometry aggregation.

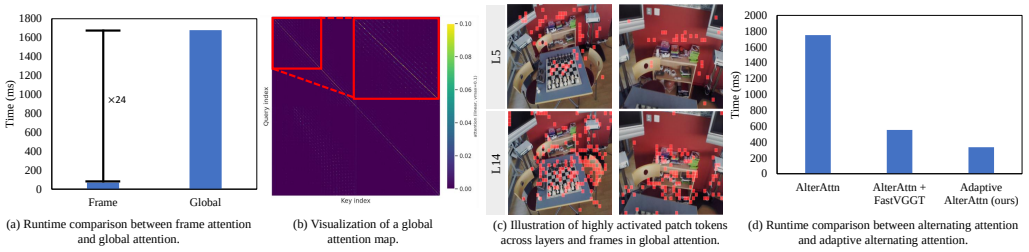
3 Methodology

3.1 Approach Overview

An overview of the proposed TurboVGGT is depicted in Figure 2. The overall architecture of TurboVGGT consists of a visual encoder, N adaptive alternating attention blocks, and multiple task-specific prediction heads.

Visual Encoder. Given a sequence of L images $\{I_i\}_{i=1}^{L}$, we first employ a visual encoder (*e.g.*, DINOv2 [28]) to extract M patch tokens from each image, denoted as $\{x_{i,j}\}_{j=1}^{M}$, where $x_{i,j} \in \mathbb{R}^D$ is a D-dimensional patch token. In addition, following [42, 22], we also append additional learnable tokens, such as camera tokens and register tokens, to the sequence of patch tokens as the input to the subsequent modules.

Adaptive Alternating Attention Blocks. Next, we use N blocks of the proposed adaptive alternating attention to process the input token sequence. As shown in Figure 2, adaptive alternating attention consists of adaptive sparsity selection, adaptive sparse global attention, and frame attention. Specifically, in each block, we first aggregate the patch tokens for each frame and use a gating network to generate the gating score for each frame. Based on gating scores, we route the patch tokens of each frame to the branch corresponding to a specific sparsity ratio k_k and generate compressed representative tokens for each frame. Then, we perform cross-attention between the compressed tokens and the dense tokens to capture global correspondences across frames. After that, we perform frame attention for tokens from each frame to capture local geometric details within each frame. This process is repeated for N blocks.



(a) Runtime comparison between frame attention and global attention.

(b) Visualization of a global attention map.

(c) Illustration of highly activated patch tokens across layers and frames in global attention.

(d) Runtime comparison between alternating attention and adaptive alternating attention.

Figure 3: Motivation for our design. (a) The runtime comparison between a frame attention layer and a global attention layer during a single forward pass on 7-Scenes. (b) A visualization of an average global attention map in the alternating attention block. (c) An illustration of highly activated patch tokens (top 5% of activations across all frames) across layers and frames in global attention. Note that visual geometry reconstruction requires not only feature matching and camera pose estimation, but also dense depth and pointmap prediction, so low-texture regions and globally informative background structures can also receive high responses. This example visualizes the top-activated tokens across different layers and frames, which may capture complementary geometric cues rather than consistently focusing on textured regions. (d) The runtime comparison between an alternating attention block and an adaptive alternating attention block during a single forward pass. Please refer to the appendix for enlarged visualizations.

Task-Specific Prediction Heads. Finally, we aggregate the output tokens from the adaptive alternating attention blocks and employ multiple task-specific prediction heads to decode the learned tokens into various 3D properties of each view. As shown in Figure 2, MLP heads [42, 22] are used to predict camera parameters and metric scaling factors, while DPT heads [31] are used to predict depth maps, point maps, confidence maps, *etc*.

3.2 Adaptive Alternating Attention

Adaptive Sparsity Selection. State-of-the-art visual geometry transformers [42, 22] typically employ alternating attention blocks consisting of frame-wise self-attention and cross-frame global full attention. This design allows transformers to learn local geometric details within each frame and to learn global correspondences across frames in an alternating manner. Although existing visual geometry transformers have achieved promising performance, they often suffer from long inference time and out-of-memory issues when processing long sequences. One of the main reasons is that global full attention involves dense global token interactions, which have a quadratic computational complexity with respect to the number of input tokens [35, 47, 37, 41]. As shown in Figure 3(a), a global attention layer consumes 24 times more runtime than a frame attention layer on the 7-Scenes dataset [36]. Consequently, as the number of input frames increases, the runtime of the global attention layer increases significantly compared to the frame attention layer. Nevertheless, as pointed out in prior work [41, 35], the attention map of each global attention layer is typically sparse (see Figure 3(b)), where only a small portion of patch tokens are highly activated for learning global relationships across frames while the majority of patch tokens have low attention activations. This suggests that it is not necessary to perform dense global full attention for all patch tokens across frames. However, unlike prior works that use the same sparsity ratio for all frames in all global attention layers, we further analyze the highly activated patch tokens across layers and frames in global attention. As shown in Figure 3(c), we find that the distribution of highly activated patch tokens can vary significantly across layers and frames. This indicates that the same sparsity ratio for all frames in all global attention layers may not be optimal. In light of this, we propose to adaptively select the sparsity level for each layer and frame rather than using a fixed sparsity ratio. Specifically, as shown in Figure 2, in each adaptive alternating attention block, we first aggregate the patch tokens for each frame. Then, we use a gating network consisting of an MLP to generate the gating score for each frame and route each frame to the branch corresponding to a specific sparsity ratio $k_k = \{k_1, k_2, \ldots, k_n\}$. This is formulated as follows:

$$k_k = F_s(F_g(F_a(\{x_{i,j}\}_{j=1}^{M}))), \tag{1}$$

where $F_a(\cdot)$ is a function for generating frame-level representations, such as average pooling, $F_g(\cdot)$ is the gating network that generates the gating score for each frame, and $F_s(\cdot)$ is the softmax function that classifies each frame into one of the branches corresponding to a specific sparsity ratio k_k.

Adaptive Sparse Global Attention. Traditional multi-view 3D reconstruction methods [33, 34] typically capture global correspondences across frames by keypoint detection and matching rather than dense pixel-wise matching. This suggests that global dependencies across frames can be achieved by focusing on structurally informative regions, such as keypoints and edges. Moreover, as shown in Figure 3(c), highly activated patch tokens in global attention are sparsely distributed across frames and different frames may have different informative regions for learning global relationships. This further indicates that learning representative tokens for each frame as structurally informative regions can not only reduce computations of redundant tokens but also facilitate global geometry modeling across frames. In light of this, we propose to learn compressed representative tokens for each frame and employ cross-attention between the compressed tokens and the dense tokens to capture global correspondences across frames. Specifically, in each branch corresponding to a specific sparsity ratio k_k, we employ an MLP $F_w(\cdot)$ to generate a weight matrix W_k for tokens of each frame, which is defined as:

$$W_i = F_w(x_i) + B_k, \tag{2}$$

where $x_i \in \mathbb{R}^{M \times D}$ are the patch tokens of frame i, $W_i \in \mathbb{R}^{M \times M_{k_k}}$ is the weight matrix, $M_{k_k} = \lfloor M(1 - k_k) \rfloor$ is the number of compressed representative tokens for the sparsity ratio k_k, and $B_k \in \mathbb{R}^{M \times M_{k_k}}$ is a learnable bias. Then, we generate the compressed representative tokens for each frame by aggregating the patch tokens of each frame using the weight matrix W_i as follows:

$$x_i^c = W_i^T x_i, \tag{3}$$

where $x_i^c \in \mathbb{R}^{M_{k_k} \times D}$ are the compressed representative tokens for frame i. Next, we concatenate the compressed representative tokens from all frames and additional learnable tokens (such as camera tokens and register tokens if used) as x^c. Here, if a reference frame is used, tokens of the reference frame are also concatenated. We then perform cross-attention between x^c and all dense tokens x' to capture global correspondences across frames, which is defined as follows:

$$x'' = \text{CrossAttn}(x' W^Q, x^c W^K, x^c W^V), \tag{4}$$

where x'' are the output dense tokens, W^Q, W^K, and W^V are the projection matrices for query, key, and value, respectively. In this way, we can capture global correspondences across frames while significantly reducing computational cost of global attention by using a small number of compressed representative tokens for each frame.

Remarks. TurboVGGT is not a straightforward combination of existing components, but a geometry-specific design for efficient visual geometry transformers. While TurboVGGT shares high-level concepts with prior token compression and routing methods, its core mechanisms are not direct instantiations of prior methods, like Perceiver [19]/Q-Former [24] or standard MoE. First, our gated router is not a standard MoE router that selects among functionally distinct experts. Instead, it dynamically allocates token budgets across both frames and layers within global geometry attention. Second, our representative tokens are input-conditioned, frame-specific weighted aggregations of dense patch tokens, rather than fixed latent queries shared across inputs as in Perceiver/Q-Former. Dense tokens remain as queries to preserve dense geometric predictions, while only cross-frame keys and values are compressed. Third, our proposed mechanisms are specifically coupled with the alternating attention architecture of VGGT, where dense frame attention preserves local geometry and adaptive compressed global attention captures cross-view correspondences. Our experiments show that the straightforward combinations of conventional methods are insufficient for VGGT acceleration.

3.3 Model Optimization

Our TurboVGGT is trained in an end-to-end manner using a combination of losses for different prediction heads and a regularization term for the adaptive sparsity selection. The overall loss is:

$$\mathcal{L} = \mathcal{L}_{\text{recon}} + \lambda \mathcal{L}_{\text{reg}}, \tag{5}$$

where $\mathcal{L}_{\text{recon}}$ is a reconstruction loss, $\mathcal{L}_{\text{reg}}$ is a sparsity regularization loss, and λ is a weight for the regularization term. Specifically, for the reconstruction loss $\mathcal{L}_{\text{recon}}$, we follow prior work [42, 46, 22]

Table 1: Evaluation of point cloud reconstruction on 7-Scenes, N-RGBD, and ScanNet. Inference timings for TurboVGGT, VGGT, FastVGGT, SparseVGGT, and Fast3R are measured on the same single data-center GPU system with approximately 140 GB of high-bandwidth memory under identical environmental settings and timing protocol. For timing, we perform multiple measured runs, with CUDA synchronization before and after each measurement, and report the average end-to-end inference time. *AVGGT results are taken from the original paper as the source code is not publicly available.

Method	7-Scenes - Stride 3							7-Scenes - Stride 10						
	Acc ↓		Comp ↓		NC ↑		Time ↓	Acc ↓		Comp ↓		NC ↑		Time ↓
	Mean	Med.	Mean	Med.	Mean	Med.		Mean	Med.	Mean	Med.	Mean	Med.	
Fast3R [49]	0.045	0.027	0.047	0.010	0.616	0.627	21.2s	0.040	0.021	0.056	0.013	0.639	0.657	2.3s
VGGT [42]	0.019	0.009	0.027	0.010	0.622	0.699	38.1s	0.019	0.008	0.027	0.010	0.628	0.710	4.5s
SparseVGGT [41]	0.021	0.010	0.029	0.010	0.621	0.693	16.2s	0.020	0.010	0.028	0.010	0.629	0.707	2.6s
AVGGT [37]*	0.054	0.031	0.131	0.078	0.528	0.541	20.6s	-	-	-	-	-	-	-
FastVGGT [35]	0.018	0.008	0.026	0.010	0.627	0.707	14.2s	0.018	0.008	0.027	0.010	0.632	0.717	2.7s
TurboVGGT (ours)	0.016	0.009	0.026	0.010	0.639	0.714	9.6s	0.016	0.009	0.025	0.010	0.650	0.730	2.0s

Method	N-RGBD - Stride 3							N-RGBD - Stride 10						
	Acc ↓		Comp ↓		NC ↑		Time ↓	Acc ↓		Comp ↓		NC ↑		Time ↓
	Mean	Med.	Mean	Med.	Mean	Med.		Mean	Med.	Mean	Med.	Mean	Med.	
Fast3R [49]	0.074	0.036	0.024	0.011	0.658	0.682	37.0s	0.061	0.028	0.031	0.013	0.669	0.712	3.9s
VGGT [42]	0.028	0.018	0.018	0.010	0.657	0.779	65.3s	0.016	0.010	0.016	0.009	0.669	0.791	7.2s
SparseVGGT [41]	0.045	0.029	0.024	0.011	0.647	0.755	26.2s	0.041	0.027	0.027	0.012	0.652	0.763	4.1s
FastVGGT [35]	0.031	0.020	0.019	0.010	0.662	0.788	30.2s	0.017	0.011	0.017	0.009	0.671	0.791	4.0s
TurboVGGT (ours)	0.025	0.016	0.022	0.011	0.657	0.773	14.7s	0.021	0.014	0.021	0.011	0.664	0.783	2.9s

Method	ScanNet - 500 Frames		ScanNet - 300 Frames		ScanNet - 100 Frames		Average	
	CD ↓	Time ↓	CD ↓	Time ↓	CD ↓	Time ↓	CD ↓	Time ↓
Fast3R [49]	0.701	62.5s	0.711	17.2s	0.723	2.5s	0.712	27.4s
VGGT [42]	0.464	90.1s	0.454	33.6s	0.438	4.9s	0.452	42.9s
SparseVGGT [41]	0.459	34.2s	0.453	14.2s	0.446	2.6s	0.453	17.0s
FastVGGT [35]	0.453	28.4s	0.447	12.3s	0.425	2.8s	0.442	14.5s
TurboVGGT (ours)	0.416	20.3s	0.413	9.5s	0.400	2.2s	0.410	10.7s

and use a sum of losses for different predictions, such as camera loss, depth loss, and pointmap loss, to supervise the learning of various 3D geometric properties. Please refer to [42, 46, 22] for the detailed definition of the reconstruction loss. For the sparsity regularization loss $\mathcal{L}_{\text{reg}}$, it is used to encourage the model to select larger sparsity levels for different layers and frames (thereby reducing the computational cost). By default, we use $\mathcal{L}_{\text{reg}} = \sum_{n=1}^{N} \sum_{i=1}^{L} (1 - k_k^{n,i})$, where $k_k^{n,i}$ is the sparsity ratio selected for frame i in the n-th adaptive alternating attention block. Additionally, we can add an entropy term to the sparsity regularization loss to push each frame in each layer to route decisively to a specific branch, which can be defined as $\sum_{n=1}^{N} (-\frac{1}{L} \sum_{i=1}^{L} p^{n,i} \log p^{n,i})$, where $p^{n,i}$ is the probability of selecting a specific branch for frame i in the nth block.

4 Experiments

4.1 Datasets and Experimental Setup

Datasets. We train our TurboVGGT on 13 high-quality datasets [22] spanning diverse indoor, outdoor, and in-the-wild scenes, including BlendedMVS [50], Mapillary Planet-Scale Depth [1], ScanNet++ v2 [51], Spring [27], TartanAirV2-WB [45, 55], UnrealStereo4K [38], Aria Synthetic Environments [2], DL3DV-10K [26], Dynamic Replica [21], MegaDepth [25], MVS-Synth [18], ParallelDomain-4D [40], and SAIL-VOS 3D [17]. We randomly resize and crop image sequences from these training datasets to various aspect ratios with a maximum resolution of 518 pixels. We evaluate our TurboVGGT on several challenging benchmark datasets, including 7-Scenes [36], N-RGBD [3], ScanNet-50 [9], RealEstate10K [56], and Sintel [5].

Evaluation Metrics. Following prior work [35], we evaluate our TurboVGGT on three tasks, including point cloud reconstruction, camera pose estimation, and depth estimation. For point cloud reconstruction, we report Accuracy distance (Acc, ↓), Completeness (Comp, ↓), and Normal Consistency (NC, ↑) on 7-Scenes and N-RGBD, and Chamfer Distance (CD, ↓) on ScanNet. For camera pose estimation, we report Relative Rotation Accuracy (RRA@30, ↑), Relative Translation Accuracy (RTA@30, ↑), and the Area Under the accuracy-threshold Curve for the minimum of RRA and RTA (AUC@30, ↑) on 7-Scenes, N-RGBD, and RealEstate10K. For depth estimation, we report

Table 2: Evaluation of camera pose estimation on 7-Scenes, N-RGBD, and RealEstate10K.

Method	7-Scenes (dense)				N-RGBD (dense)				RealEstate10K		
	RRA@30↑	RTA@30↑	AUC@30↑	Time ↓	RRA@30↑	RTA@30↑	AUC@30↑	Time ↓	RRA@30↑	RTA@30↑	AUC@30↑
Fast3R [49]	-	-	-	-	-	-	-	-	99.05	81.86	61.68
FLARE [54]	-	-	-	-	-	-	-	-	99.69	95.23	80.01
VGGT [42]	100.00	96.58	77.76	38.1s	100.00	99.47	91.59	65.3s	99.97	96.22	85.32
SparseVGGT [41]	100.00	93.40	70.07	16.2s	97.20	97.03	83.16	26.2s	99.61	91.35	76.65
AVGGT [37]	100.00	96.48	78.29	20.6s	-	-	-	-	99.93	95.62	85.45
Speed3R [32]	-	-	-	-	-	-	-	-	-	-	74.81
FlashVGGT [47]	-	-	-	-	-	-	-	-	99.92	95.61	85.30
FastVGGT [35]	99.99	96.29	76.90	14.2s	100.00	99.65	92.47	30.2s	99.92	94.76	84.37
TurboVGGT (ours)	100.00	96.83	81.87	9.6s	100.00	99.71	93.28	14.7s	99.93	94.69	84.31

Table 3: Evaluation of depth estimation on 7-Scenes, N-RGBD, and Sintel.

Method	7-Scenes (dense)			N-RGBD (dense)			Sintel	
	AbsRel↓	δ<1.25↑	Time ↓	AbsRel↓	δ<1.25↑	Time ↓	AbsRel↓	δ<1.25↑
Fast3R [49]	-	-	-	-	-	-	0.544	0.509
FLARE [54]	-	-	-	-	-	-	0.606	0.402
VGGT [42]	0.264	0.958	38.1s	0.013	0.993	65.3s	0.335	0.599
SparseVGGT [41]	0.393	0.956	16.2s	0.015	0.990	26.2s	-	-
FlashVGGT [47]	-	-	-	-	-	-	0.346	0.586
FastVGGT [35]	0.394	0.953	14.2s	0.013	0.990	30.2s	0.337	0.582
TurboVGGT (ours)	0.296	0.980	9.6s	0.013	0.994	14.7s	0.287	0.650

Absolute Relative Error (AbsRel, ↓) and δ<1.25 ratio (↑) on 7-Scenes, N-RGBD, and Sintel. In addition, we also report inference time (seconds, ↓).

Implementation Details. We implement our TurboVGGT with Python and PyTorch. By default, we use VGGT [42] as the backbone and also adapt our approach to other backbones, *i.e.*, MapAnything [22] (denoted as TurboVGGT-M) and π^3 [46] (denoted as TurboVGGT-π), to validate the generalizability of our approach. We employ DINOv2 [28] as the visual encoder and build the adaptive alternating attention blocks as described in Sec. 3.2. For the adaptive sparsity selection, we use three branches and set $k_k = \{\frac{3}{4}, \frac{8}{9}, \frac{15}{16}\}$, *i.e.*, use three sparsity ratios of 75%, 89%, and 94% for the three branches, respectively, because these sparsity levels correspond to the token budgets induced by $2\times$, $3\times$, and $4\times$ spatial compression, respectively. These levels cover conservative, moderate, and aggressive computation budgets. In addition, discrete levels produce regular token shapes and computation patterns, and a small fixed set of sparsity levels simplifies optimization and prevents the gating network from selecting extreme token budgets. The number of the adaptive alternating attention blocks is determined by the backbone [42, 46, 22]. We construct the task-specific prediction heads following [42, 46, 22]. To facilitate model convergence, we initialize our TurboVGGT with pretrained weights from [42, 46, 22]. We train our TurboVGGT on 8 GPUs for 10 epochs using AdamW optimizer with a cosine annealing learning rate and a 1-epoch warm-up schedule. The maximum learning rate is set to 5×10^{-6} and decayed to 5×10^{-8}. For FastVGGT, we use the merging ratio of 90%, and for SparseVGGT, we use the sparsity ratio of 75%.

4.2 Comparison with State-of-the-Art Methods

Point Cloud Reconstruction. Table 1 shows results of point cloud reconstruction on 7-Scenes, N-RGBD, and ScanNet. Overall, our TurboVGGT achieves faster inference speed than state-of-the-art methods, while maintaining comparable or even better performance. Specifically, on 7-Scenes, our TurboVGGT outperforms state-of-the-art methods, such as FastVGGT [35], SparseVGGT [41], and VGGT [42], in most metrics. Interestingly, TurboVGGT achieves overall better reconstruction quality than VGGT [42], while being 2-4$\times$ faster. This can be attributed to the adaptive sparse global attention under the guidance of adaptive sparsity selection, which effectively captures global context while reducing redundant computations. On N-RGBD, our TurboVGGT also achieves competitive reconstruction performance while achieving faster inference than prior methods. Although the reconstruction quality of TurboVGGT is slightly inferior to FastVGGT in some metrics, TurboVGGT is 1.4$\times$ faster than FastVGGT in the Stride 10 setting and 2.1$\times$ faster in the Stride 3 setting. On ScanNet, TurboVGGT achieves the best reconstruction quality while being faster than prior methods. Figure 4 (first row) shows qualitative comparison of point cloud reconstruction. We can see that our TurboVGGT produces more complete point clouds compared with state-of-the-art methods.

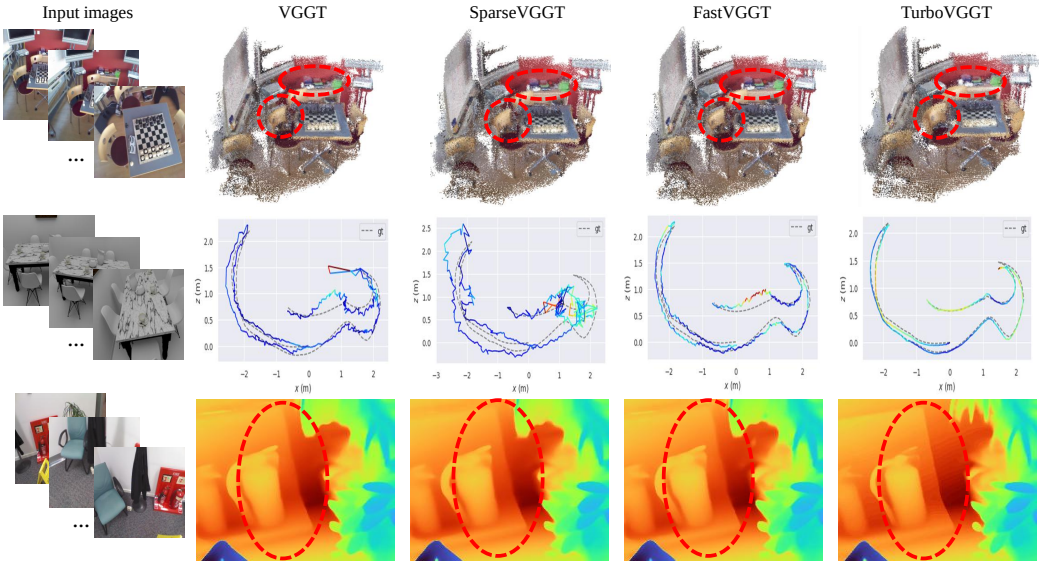


Figure 4: Qualitative comparison of point cloud reconstruction, camera pose estimation, and depth estimation (7-Scenes and N-RGBD with stride 3). See appendix for more qualitative results.

Table 4: Evaluation of our approach with various backbones on 7-Scenes (stride 3).

Method	Point Cloud Estimation		Camera Pose Estimation		Depth Estimation	Time ↓
	Acc↓	NC↑	RRA@30↑	RTA@30↑	δ<1.25↑	
VGGT [42]	0.019	0.622	100.00	96.58	0.958	38.1s
VGGT [42] (Fine-tuned)	0.016	0.638	100.00	96.57	0.960	38.1s
TurboVGGT (Ours)	0.016	0.639	100.00	96.83	0.980	**9.6s** (4.0×)
π^3 [46]	0.013	0.585	100.00	97.47	0.986	30.4s
π^3 [46] (Fine-tuned)	0.015	0.586	100.00	97.50	0.984	30.4s
TurboVGGT-π (Ours)	0.014	0.585	100.00	97.11	0.983	**6.2s** (4.9×)
MapAnything [22]	0.018	0.579	100.00	92.45	0.975	15.3s
MapAnything [22] (Fine-tuned)	0.021	0.569	99.99	90.60	0.974	15.3s
TurboVGGT-M (Ours)	0.018	0.577	100.00	92.86	0.975	**4.0s** (3.8×)

Camera Pose Estimation. We report results of camera pose estimation on 7-Scenes (dense, stride=3), N-RGBD (dense, stride=3), and RealEstate10K (sparse) in Table 2. On 7-Scenes and N-RGBD, our TurboVGGT achieves the best camera pose estimation performance and faster inference speed compared with state-of-the-art methods, such as FastVGGT [35], SparseVGGT [41], and VGGT [42]. In addition, we also evaluate TurboVGGT on RealEstate10K, which contains sparse image sequences. Our TurboVGGT still achieves competitive performance in this sparse setting. These results show the efficiency and effectiveness of our TurboVGGT in camera pose estimation across different settings. Figure 4 (second row) shows that the camera pose trajectory produced by our TurboVGGT is closer to the ground-truth trajectory than state-of-the-art competitors.

Depth Estimation. Table 3 shows results of depth estimation on 7-Scenes (dense), N-RGBD (dense), and Sintel (single). On 7-Scenes, our TurboVGGT achieves the second-best AbsRel and the best δ<1.25, while being faster than prior methods. On N-RGBD, our TurboVGGT achieves the best results in all metrics. Additionally, we evaluate TurboVGGT for monocular depth estimation on Sintel. Our TurboVGGT still achieves competitive performance in this monocular depth estimation setting, demonstrating the generalizability of our approach across settings. Figure 4 (third row) shows that our TurboVGGT produces higher-quality depth compared with state-of-the-art methods.

4.3 Adaptation to Different Backbones

To validate the generalizability of the proposed adaptive alternating attention, we adapt our TurboVGGT to more visual geometry transformer backbones, including π^3 [46] and MapAnything [22]. Results are shown in Table 4. We can see that our approach consistently improves the inference

Table 5: Ablation study of the proposed TurboVGGT on 7-Scenes (stride 3).

Method	Ada. Selection		Ada. Global Attn		Acc↓	AUC@30↑	AbsRel↓
	Adaptive	Multi-branch	Weight matrix	Cross-attn			
V1	✗	✓	✓	✓	**0.016**	81.34	0.302
V2	✗	✗	✓	✓	**0.016**	80.88	0.300
V3	✓	✓	✗	✓	0.017	79.80	0.299
V4	✓	✓	✗	✗	0.047	65.73	0.311
Full	✓	✓	✓	✓	**0.016**	**81.87**	**0.296**

Table 6: Ablation study of different sparsity ratios on 7-Scenes (stride 3).

Variants	Point Cloud Estimation		Camera Pose Estimation		Depth Estimation	Time ↓
	Acc↓	NC↑	RRA↑	RTA↑	δ<1.25↑	
Three sparsity ratios (75%, 89%, 94%) (default)	0.016	**0.639**	**100.00**	**96.83**	**0.980**	9.6s
One sparsity ratio (75%)	**0.015**	0.625	**100.00**	96.81	**0.980**	16.1s
One sparsity ratio (89%)	0.016	0.621	**100.00**	96.59	0.979	10.0s
Two sparsity ratios (89%, 94%)	0.018	0.619	**100.00**	96.32	0.979	8.9s
Four sparsity ratios (75%, 89%, 94%, 96%)	0.019	0.617	**100.00**	95.48	0.978	**8.2s**

speed of these backbones while maintaining comparable or even better performance on point cloud reconstruction, camera pose estimation, and depth estimation. This demonstrates the effectiveness and generalizability of our adaptive alternating attention in accelerating visual geometry transformers across different backbones.

4.4 Ablation Study

Component Effectiveness. We present the ablation study of our TurboVGGT on 7-Scenes in Table 5. For the adaptive sparsity selection, we report two variants of TurboVGGT, *i.e.*, V1 replaces the adaptive sparsity selection with fixed selection that evenly assigns frames to the three branches, and V2 removes the adaptive sparsity selection and uses only one branch with a fixed sparsity ratio. From Table 5, we can see that both V1 and V2 achieve worse performance than the full TurboVGGT. This shows the importance of the adaptive sparsity selection in routing frames to different branches for better capturing global context and reducing redundant computations. For the adaptive sparse global attention, we also report two variants of TurboVGGT, *i.e.*, V3 replaces weight matrix based compressed representative token learning with grid based token selection, and V4 removes the adaptive sparse global attention and uses global attention for merged tokens and upsampling to restore the original token number. From Table 5, we can see that without using weight matrix based compressed representative token learning, the reconstruction quality of V3 is worse than the full TurboVGGT. When removing the adaptive sparse global attention, V4 achieves significantly worse reconstruction quality than the full TurboVGGT. These results demonstrate the importance of the adaptive sparse global attention in learning representative tokens and performing global attention on these tokens for better capturing global relationships and reducing redundant computations.

Branch Sparsity Ratio Selection. In Table 6, we report additional ablation study on the branch sparsity ratio selection, including one, two, three, and four sparsity ratios. Using three sparsity ratios achieves the best speed-accuracy trade-off performance; When using one sparsity ratio of 75%, the reconstruction quality is comparable to the default three-sparsity-ratio setting, but the inference speed is significantly slower; When using two high sparsity ratios can reduce inference time but slightly degrade accuracy; When using four sparsity ratios, the inference time is even faster, but the reconstruction quality is significantly degraded.

5 Conclusion

In this work, we propose TurboVGGT for fast multi-view 3D reconstruction in a feed-forward manner. The core idea of TurboVGGT is to construct an efficient visual geometry transformer with adaptive alternating attention blocks, which consist of adaptive sparse global attention guided by adaptive sparsity selection for global geometry modeling and frame attention for local detail aggregation. Extensive experiments on multiple 3D reconstruction benchmarks demonstrate that TurboVGGT achieves a better balance between reconstruction quality and computational efficiency, outperforming state-of-the-art methods.

References

[1] Manuel López Antequera, Pau Gargallo, Markus Hofinger, Samuel Rota Bulo, Yubin Kuang, and Peter Kontschieder. Mapillary planet-scale depth dataset. In *European Conference on Computer Vision*, pages 589–604. Springer, 2020.

[2] Armen Avetisyan, Christopher Xie, Henry Howard-Jenkins, Tsun-Yi Yang, Samir Aroudj, Suvam Patra, Fuyang Zhang, Duncan Frost, Luke Holland, Campbell Orme, et al. Scenescript: Reconstructing scenes with an autoregressive structured language model. In *European Conference on Computer Vision*, pages 247–263. Springer, 2024.

[3] Dejan Azinović, Ricardo Martin-Brualla, Dan B Goldman, Matthias Nießner, and Justus Thies. Neural rgb-d surface reconstruction. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 6290–6301, 2022.

[4] Daniel Bolya, Cheng-Yang Fu, Xiaoliang Dai, Peizhao Zhang, Christoph Feichtenhofer, and Judy Hoffman. Token merging: Your vit but faster. In *The Eleventh International Conference on Learning Representations*, 2023.

[5] Daniel J Butler, Jonas Wulff, Garrett B Stanley, and Michael J Black. A naturalistic open source movie for optical flow evaluation. In *European conference on computer vision*, pages 611–625. Springer, 2012.

[6] Mengzhao Chen, Wenqi Shao, Peng Xu, Mingbao Lin, Kaipeng Zhang, Fei Chao, Rongrong Ji, Yu Qiao, and Ping Luo. Diffrate: Differentiable compression rate for efficient vision transformers. In *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 17118–17128. IEEE, 2023.

[7] Xingyu Chen, Yue Chen, Yuliang Xiu, Andreas Geiger, and Anpei Chen. TTT3r: 3d reconstruction as test-time training. In *The Fourteenth International Conference on Learning Representations*, 2026.

[8] Yutian Chen, Yuheng Qiu, Ruogu Li, Jay Patrikar, and Sebastian Scherer. Co-me: Confidence guided token merging for visual geometric transformers. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14590–14599, 2026.

[9] Angela Dai, Angel X Chang, Manolis Savva, Maciej Halber, Thomas Funkhouser, and Matthias Nießner. Scannet: Richly-annotated 3d reconstructions of indoor scenes. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5828–5839, 2017.

[10] Tri Dao. Flashattention-2: Faster attention with better parallelism and work partitioning. *arXiv preprint arXiv:2307.08691*, 2023.

[11] Tri Dao, Dan Fu, Stefano Ermon, Atri Rudra, and Christopher Ré. Flashattention: Fast and memory-efficient exact attention with io-awareness. *Advances in neural information processing systems*, 35: 16344–16359, 2022.

[12] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations*, 2021.

[13] Sven Elflein, Ruilong Li, Sérgio Agostinho, Zan Gojcic, Laura Leal-Taixé, Qunjie Zhou, and Aljosa Osep. Vgg-t^3: Offline feed-forward 3d reconstruction at scale. *arXiv preprint arXiv:2602.23361*, 2026.

[14] Yasutaka Furukawa and Carlos Hernández. Multi-view stereo: A tutorial. *Foundations and Trends in Computer Graphics and Vision*, 9(1-2):1–148, 2015.

[15] Dongchen Han, Tianzhu Ye, Yizeng Han, Zhuofan Xia, Siyuan Pan, Pengfei Wan, Shiji Song, and Gao Huang. Agent attention: On the integration of softmax and linear attention. In *European conference on computer vision*, pages 124–140. Springer, 2024.

[16] Jisang Han, Honggyu An, Jaewoo Jung, Takuya Narihira, Junyoung Seo, Kazumi Fukuda, Chaehyun Kim, Sunghwan Hong, Yuki Mitsufuji, and Seungryong Kim. Enhancing 3d reconstruction for dynamic scenes. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025.

[17] Yuan-Ting Hu, Jiahong Wang, Raymond A Yeh, and Alexander G Schwing. Sail-vos 3d: A synthetic dataset and baselines for object detection and 3d mesh reconstruction from video data. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1418–1428, 2021.

[18] Po-Han Huang, Kevin Matzen, Johannes Kopf, Narendra Ahuja, and Jia-Bin Huang. Deepmvs: Learning multi-view stereopsis. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2821–2830, 2018.

[19] Andrew Jaegle, Felix Gimeno, Andy Brock, Oriol Vinyals, Andrew Zisserman, and Joao Carreira. Perceiver: General perception with iterative attention. In *International conference on machine learning*, pages 4651–4664. PMLR, 2021.

[20] Haian Jin, Rundi Wu, Tianyuan Zhang, Ruiqi Gao, Jonathan T Barron, Noah Snavely, and Aleksander Holynski. Zipmap: Linear-time stateful 3d reconstruction via test-time training. *arXiv preprint arXiv:2603.04385*, 2026.

[21] Nikita Karaev, Ignacio Rocco, Benjamin Graham, Natalia Neverova, Andrea Vedaldi, and Christian Rupprecht. Dynamicstereo: Consistent dynamic depth from stereo videos. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13229–13239, 2023.

[22] Nikhil Varma Keetha, Norman Müller, Johannes Schönberger, Lorenzo Porzi, Yuchen Zhang, Tobias Fischer, Arno Knapitsch, Duncan Zauss, Ethan Weber, Nelson Antunes, Jonathon Luiten, Manuel Lopez-Antequera, Samuel Rota Bulò, Christian Richardt, Deva Ramanan, Sebastian Scherer, and Peter Kontschieder. Mapanything: Universal feed-forward metric 3d reconstruction. In *Thirteenth International Conference on 3D Vision*, 2026.

[23] Vincent Leroy, Yohann Cabon, and Jérôme Revaud. Grounding image matching in 3d with mast3r. In *European conference on computer vision*, pages 71–91. Springer, 2024.

[24] Junnan Li, Dongxu Li, Silvio Savarese, and Steven Hoi. Blip-2: Bootstrapping language-image pre-training with frozen image encoders and large language models. In *International conference on machine learning*, pages 19730–19742. PMLR, 2023.

[25] Zhengqi Li and Noah Snavely. Megadepth: Learning single-view depth prediction from internet photos. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2041–2050, 2018.

[26] Lu Ling, Yichen Sheng, Zhi Tu, Wentian Zhao, Cheng Xin, Kun Wan, Lantao Yu, Qianyu Guo, Zixun Yu, Yawen Lu, et al. Dl3dv-10k: A large-scale scene dataset for deep learning-based 3d vision. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 22160–22169, 2024.

[27] Lukas Mehl, Jenny Schmalfuss, Azin Jahedi, Yaroslava Nalivayko, and Andrés Bruhn. Spring: A high-resolution high-detail dataset and benchmark for scene flow, optical flow and stereo. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4981–4991, 2023.

[28] Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, et al. Dinov2: Learning robust visual features without supervision. *Transactions on Machine Learning Research Journal*, 2024.

[29] Emanuele Palazzolo, Jens Behley, Philipp Lottes, Philippe Giguere, and Cyrill Stachniss. Refusion: 3d reconstruction in dynamic environments for rgb-d cameras exploiting residuals. In *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 7855–7862. IEEE, 2019.

[30] Linfei Pan, Dániel Baráth, Marc Pollefeys, and Johannes L Schönberger. Global structure-from-motion revisited. In *European Conference on Computer Vision*, pages 58–77. Springer, 2024.

[31] René Ranftl, Alexey Bochkovskiy, and Vladlen Koltun. Vision transformers for dense prediction. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 12179–12188, 2021.

[32] Weining Ren, Xiao Tan, and Kai Han. Speed3r: Sparse feed-forward 3d reconstruction models. *arXiv preprint arXiv:2603.08055*, 2026.

[33] Johannes L Schonberger and Jan-Michael Frahm. Structure-from-motion revisited. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4104–4113, 2016.

[34] Johannes L Schönberger, Enliang Zheng, Jan-Michael Frahm, and Marc Pollefeys. Pixelwise view selection for unstructured multi-view stereo. In *European conference on computer vision*, pages 501–518. Springer, 2016.

[35] You Shen, Zhipeng Zhang, Yansong Qu, Xiawu Zheng, Jiayi Ji, Shengchuan Zhang, and Liujuan Cao. FastVGGT: Fast visual geometry transformer. In *The Fourteenth International Conference on Learning Representations*, 2026.

[36] Jamie Shotton, Ben Glocker, Christopher Zach, Shahram Izadi, Antonio Criminisi, and Andrew Fitzgibbon. Scene coordinate regression forests for camera relocalization in rgb-d images. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2930–2937, 2013.

[37] Xianbing Sun, Zhikai Zhu, Zhengyu Lou, Bo Yang, Jinyang Tang, Liqing Zhang, He Wang, and Jianfu Zhang. Avggt: Rethinking global attention for accelerating vggt. *arXiv preprint arXiv:2512.02541*, 2025.

[38] Fabio Tosi, Yiyi Liao, Carolin Schmitt, and Andreas Geiger. Smd-nets: Stereo mixture density networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 8942–8952, 2021.

[39] Tuan Tran Anh, Duy MH Nguyen, Hoai-Chau Tran, Michael Barz, Khoa D Doan, Roger Wattenhofer, Vien Ngo, Mathias Niepert, Daniel Sonntag, and Paul Swoboda. How many tokens do 3d point cloud transformer architectures really need? *Advances in Neural Information Processing Systems*, 38:13464–13496, 2026.

[40] Basile Van Hoorick, Rundi Wu, Ege Ozguroglu, Kyle Sargent, Ruoshi Liu, Pavel Tokmakov, Achal Dave, Changxi Zheng, and Carl Vondrick. Generative camera dolly: Extreme monocular dynamic novel view synthesis. In *European Conference on Computer Vision*, pages 313–331. Springer, 2024.

[41] Chung-Shien Brian Wang, Christian Schmidt, Jens Piekenbrinck, and Bastian Leibe. Faster vggt with block-sparse global attention. *arXiv preprint arXiv:2509.07120*, 2025.

[42] Jianyuan Wang, Minghao Chen, Nikita Karaev, Andrea Vedaldi, Christian Rupprecht, and David Novotny. Vggt: Visual geometry grounded transformer. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 5294–5306, 2025.

[43] Qianqian Wang, Yifei Zhang, Aleksander Holynski, Alexei A Efros, and Angjoo Kanazawa. Continuous 3d perception model with persistent state. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 10510–10522, 2025.

[44] Shuzhe Wang, Vincent Leroy, Yohann Cabon, Boris Chidlovskii, and Jerome Revaud. Dust3r: Geometric 3d vision made easy. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 20697–20709, 2024.

[45] Wenshan Wang, Delong Zhu, Xiangwei Wang, Yaoyu Hu, Yuheng Qiu, Chen Wang, Yafei Hu, Ashish Kapoor, and Sebastian Scherer. Tartanair: A dataset to push the limits of visual slam. In *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 4909–4916. IEEE, 2020.

[46] Yifan Wang, Jianjun Zhou, Haoyi Zhu, Wenzheng Chang, Yang Zhou, Zizun Li, Junyi Chen, Jiangmiao Pang, Chunhua Shen, and Tong He. π^3: Scalable permutation-equivariant visual geometry learning. *arXiv e-prints*, pages arXiv–2507, 2025.

[47] Zipeng Wang and Dan Xu. Flashvggt: Efficient and scalable visual geometry transformers with compressed descriptor attention. *arXiv preprint arXiv:2512.01540*, 2025.

[48] Cong Wei, Brendan Duke, Ruowei Jiang, Parham Aarabi, Graham W Taylor, and Florian Shkurti. Sparsifiner: Learning sparse instance-dependent attention for efficient vision transformers. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 22680–22689, 2023.

[49] Jianing Yang, Alexander Sax, Kevin J Liang, Mikael Henaff, Hao Tang, Ang Cao, Joyce Chai, Franziska Meier, and Matt Feiszli. Fast3r: Towards 3d reconstruction of 1000+ images in one forward pass. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 21924–21935, 2025.

[50] Yao Yao, Zixin Luo, Shiwei Li, Jingyang Zhang, Yufan Ren, Lei Zhou, Tian Fang, and Long Quan. Blendedmvs: A large-scale dataset for generalized multi-view stereo networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 1790–1799, 2020.

[51] Chandan Yeshwanth, Yueh-Cheng Liu, Matthias Nießner, and Angela Dai. Scannet++: A high-fidelity dataset of 3d indoor scenes. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 12–22, 2023.

[52] Wang Zeng, Sheng Jin, Wentao Liu, Chen Qian, Ping Luo, Wanli Ouyang, and Xiaogang Wang. Not all tokens are equal: Human-centric visual analysis via token clustering transformer. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11101–11111, 2022.

[53] Jintao Zhang, Chendong Xiang, Haofeng Huang, Jia Wei, Haocheng Xi, Jun Zhu, and Jianfei Chen. Spargeattention: Accurate and training-free sparse attention accelerating any model inference. In *International Conference on Machine Learning*, pages 76397–76413. PMLR, 2025.

[54] Shangzhan Zhang, Jianyuan Wang, Yinghao Xu, Nan Xue, Christian Rupprecht, Xiaowei Zhou, Yujun Shen, and Gordon Wetzstein. Flare: Feed-forward geometry, appearance and camera estimation from uncalibrated sparse views. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 21936–21947, 2025.

[55] Yuchen Zhang, Nikhil Keetha, Chenwei Lyu, Bhuvan Jhamb, Yutian Chen, Yuheng Qiu, Jay Karhade, Shreyas Jha, Yaoyu Hu, Deva Ramanan, et al. Ufm: A simple path towards unified dense correspondence with flow. *arXiv preprint arXiv:2506.09278*, 2025.

[56] Tinghui Zhou, Richard Tucker, John Flynn, Graham Fyffe, and Noah Snavely. Stereo magnification: learning view synthesis using multiplane images. *ACM Transactions on Graphics (TOG)*, 37(4):1–12, 2018.

[57] Dong Zhuo, Wenzhao Zheng, Jiahe Guo, Yuqi Wu, Jie Zhou, and Jiwen Lu. Streaming visual geometry transformer. In *The Fourteenth International Conference on Learning Representations*, 2026.

A Technical appendices and supplementary material

A.1 Limitation and Future Work

While the proposed TurboVGGT achieves superior performance on fast multi-view 3D reconstruction, there are still some limitations that can be addressed in future work. First, TurboVGGT does not explicitly model temporal dynamics of input sequences, which may limit its performance on highly dynamic scenes. Second, although TurboVGGT achieves a balance between reconstruction quality and computational efficiency, there is still room for improvement, especially for challenging scenes with complex geometry. Third, the variable token counts in TurboVGGT are inconvenient to batch and parallelize. For the associated training cost, TurboVGGT has lower peak-memory requirements but higher training compute cost (as shown in Table 7). Compared with the fine-tuned VGGT, TurboVGGT reduces the observed peak allocated GPU memory from 100 GB/GPU to 70 GB/GPU, enabling a lower-memory training footprint. This comes at additional optimization time cost because of the routing mechanism, adaptive sparse global attention, and gradient checkpointing. However, at inference, a single scene uses hard argmax routing and tightly packs the resulting variable-length tokens, which provides the intended efficiency gain. Thus, TurboVGGT primarily provides inference efficiency and reduced training memory, rather than faster training. In future work, we aim to explore more effective ways to model temporal dynamics for 4D reconstruction and improve the model architecture design of TurboVGGT for better performance.

Table 7: Comparison of training and inference efficiency on 7-Scenes (stride 3).

Method	Training		Inference	
	Time	Peak GPU Memory	Time	Peak GPU Memory
Fine-tuned VGGT	**16.6 h**	100 GB	14.2 s	25 GB
TurboVGGT (Ours)	22.7 h	**70 GB**	**9.6 s**	**23 GB**

A.2 Societal Impacts

This work presents an efficient visual geometry transformer for multi-view 3D reconstruction, which can potentially benefit various applications such as virtual reality, robotics, and autonomous driving. We do not discover direct negative societal impacts of our approach. However, since our approach requires training with large-scale datasets and substantial computational resources, it may raise concerns regarding data privacy and environmental impact. These societal impacts should be carefully considered when using our approach in real-world applications.

A.3 Complexity Analysis of Adaptive Sparse Global Attention

As discussed in the main paper, global full attention used in conventional visual geometry transformers [42] is computationally expensive and can lead to long inference time and out-of-memory issues when processing long sequences. The computational complexity of global full attention is $\mathcal{O}(L^2M^2)$, which is quadratic with respect to the number of input tokens, where L is the number of input frames and M is the number of patch tokens for each frame. For our adaptive sparse global attention, the computational cost mainly comes from the cross-attention between the compressed representative tokens and the dense tokens, while the other operations have relatively less computational cost (especially for long sequences) which can be ignored for simplicity. Specifically, the computational

complexity of our adaptive sparse global attention is $\mathcal{O}(L^2M^2(1-k_k))$, where k_k is the sparsity ratio. Since we encourage our model to select a large sparsity ratio k_k for each frame and have $(1-k_k)\ll 1$, *e.g.*, $(1-k_k)=\{25\%, 11\%, 6\%\}$, our adaptive sparse global attention significantly reduces the computational complexity compared to global full attention, which is consistent with the significant runtime reduction observed in our experiments (see Section 4). Although some existing works [35, 47] also have a computational complexity similar to ours, they typically use the same sparsity ratio for all frames across layers and lack mechanisms to adaptively learn representative tokens that can effectively capture global relationships; in contrast, our method adaptively selects the sparsity level across layers and frames and achieves better reconstruction quality while maintaining fast inference speed.

A.4 Memory Efficiency Analysis

In Table 8, we compare the peak GPU memory usage of our TurboVGGT, VGGT [42], SparseVGGT [41], and FastVGGT [35] on 7-Scenes (dense setting). We can see that our TurboVGGT not only achieves faster inference speed but also reduces the peak GPU memory usage compared with these state-of-the-art methods. This result further shows the superiority of our TurboVGGT that can effectively reduce the computational cost and memory usage for multi-view 3D reconstruction.

Table 8: Memory efficiency analysis on 7-Scenes.

Metrics	VGGT [42]	SparseVGGT [41]	FastVGGT [35]	TurboVGGT (Ours)
Peak inference memory↓	25.24 GB	27.84 GB	31.18 GB	**23.47 GB**
Inference speed↑	8.27 FPS	19.56 FPS	22.16 FPS	**33.01 FPS**

A.5 Evaluation of Video Depth Estimation

In Table 9, we further perform evaluation of video depth estimation on Sintel [5] and Bonn [29], following [13]. We can see that our TurboVGGT achieves the best performance on both datasets, outperforming state-of-the-art methods such as VGG-T^3 [13] and ZipMap [20]. This further demonstrates the superior performance of our TurboVGGT in video depth estimation, which is crucial for multi-view 3D reconstruction.

Table 9: Evaluation of video depth estimation on Sintel and Bonn. Results of some methods are taken from [13, 20].

Datasets	Metrics	VGGT [42]	SparseVGGT [41]	FastVGGT [35]	TTT3R [7]	ZipMap [20]	VGG-T^3 [13]	TurboVGGT (Ours)
Sintel	AbsRel↓	0.300	0.304	0.307	0.469	0.248	0.345	**0.212**
	δ<1.25↑	0.646	0.639	0.630	0.510	0.695	0.581	**0.716**
Bonn	AbsRel↓	0.059	0.057	0.058	0.061	0.059	0.063	**0.053**
	δ<1.25↑	0.967	0.968	0.969	0.969	0.973	0.963	**0.975**

A.6 Impact of Sparsity Regularization Loss $\mathcal{L}_{reg}$

In Table 10, we analyze the impact of the sparsity regularization loss $\mathcal{L}_{reg}$ on our TurboVGGT. When setting $\lambda_{reg}=0$, our TurboVGGT does not have an explicit regularization to encourage sparsity, resulting in both worse reconstruction quality and slower inference speed compared with our full model. When setting $\lambda_{reg}=0.001$, the inference speed is improved compared with $\lambda_{reg}=0$, but the reconstruction quality is still worse than our full model. When setting $\lambda_{reg}=0.01$, our TurboVGGT achieves the best performance in terms of both reconstruction quality and inference speed.

Table 10: Impact of sparsity regularization loss $\mathcal{L}_{reg}$ (on 7-Scenes).

Method	Acc↓	AUC@30↑	AbsRel↓	Time↓
TurboVGGT w/o regularization loss ($\lambda_{reg}=0$)	**0.016**	80.63	0.309	14.5s
TurboVGGT ($\lambda_{reg}=0.001$)	0.017	80.23	0.301	12.5s
TurboVGGT (default, $\lambda_{reg}=0.01$)	**0.016**	**81.87**	**0.296**	**9.6s**

A.7 Impact of Separating Additional Learnable Tokens

In our design, we separate additional learnable tokens of each frame from dense patch tokens during adaptive sparsity selection and compressed representative token learning. This allows our model to focus on capturing global relationships across frames without being distracted by additional global tokens. In Table 11, we report a variant of our TurboVGGT that does not separate the additional learnable tokens during adaptive sparsity selection and compressed representative token learning. It can be seen that this variant performs slightly worse than our full model for multi-view 3D reconstruction.

Table 11: Impact of separating additional learnable tokens (on 7-Scenes).

Method	Acc↓	Comp↓	RRA@30↑	RTA@30↑	AbsRel↓	δ<1.25↑
TurboVGGT	**0.016**	0.026	**100.00**	**96.83**	**0.296**	**0.980**
TurboVGGT w/o separating additional tokens	0.017	**0.024**	**100.00**	96.29	0.314	0.979

A.8 Difference from Q-Former and Perceiver

Q-Former [24] and Perceiver [19] employ a compact set of latent queries and cross-attention to perform attention over the input in a low-rank latent space, which shares some merits with our adaptive sparse global attention design. However, our approach significantly differs from them in motivation and design. Q-Former and Perceiver use a set of learnable latents for all inputs to transform the inputs into a compact latent space, which may not be optimal for multi-view 3D reconstruction where the importance of different frames can vary significantly; in contrast, our approach adaptively selects the sparsity level across layers and frames and learns compressed representative tokens for each frame. More importantly, our method is specifically designed for multi-view 3D reconstruction, which requires modeling global correspondences across frames while preserving local geometric details within each frame; in contrast, Q-Former and Perceiver are designed for general language and computer vision tasks and do not explicitly consider the unique challenges of multi-view 3D reconstruction. As shown Table 12, we report a variant of VGGT that uses a set of learnable query latents. We can see that this variant achieves significantly worse reconstruction quality compared with our TurboVGGT. This result further demonstrates the importance of our adaptive sparse global attention in TurboVGGT for achieving superior performance, while using existing design such as learnable query latents is not sufficient to capture representative tokens across frames for multi-view 3D reconstruction.

Table 12: Comparison between the proposed design and a variant of VGGT that uses learnable query latents (point cloud reconstruction on 7-Scenes (stride 3)).

Method	Acc↓	Comp↓	RRA@30↑	RTA@30↑	AbsRel↓	δ<1.25↑
TurboVGGT	**0.016**	**0.026**	**100.00**	**96.83**	**0.296**	**0.980**
VGGT + learnable query latents	0.109	0.041	91.05	78.24	0.362	0.975

A.9 Discrete Sparsity Ratio vs. Continuous Sparsity Ratio

In our default setting, we use three discrete sparsity ratios for the branches rather than a continuous sparsity formulation because we consider that although a continuous sparsity formulation can be more flexible, it may lead to irregular token shapes and computation patterns, which can increase optimization difficulty. In Table 13, we additionally evaluate a continuous sparsity variant. Our results show that a continuous sparsity formulation achieves slightly worse performance than the default discrete sparsity formulation.

Table 13: Ablation study of discrete and continuous sparsity formulations on 7-Scenes (stride 3).

Method	Point Cloud Estimation		Camera Pose Estimation		Depth Estimation
	Acc↓	NC↑	RRA↑	RTA↑	δ<1.25↑
Three discrete sparsity ratios	**0.016**	**0.639**	**100.00**	**96.83**	**0.980**
Continuous sparsity formulation	0.017	0.618	**100.00**	96.76	0.978

A.10 Evaluation on Camera Tracking Accuracy

In this work, we mainly report AUC, RRA, and RTA for camera pose estimation, following the common evaluation protocol of recent visual geometry transformers. These metrics support flexible, potentially sparse or unordered image inputs. In Table 14, we additionally evaluate camera tracking accuracy with ATE↓, RPE-rot↓, and RPE-trans↓ metrics on 7-Scenes, Sintel, and Bonn datasets. Our results demonstrate that TurboVGGT is faster than VGGT and achieves superior camera tracking accuracy compared with VGGT on 7-Scenes and Sintel, while maintaining competitive performance on Bonn.

Table 14: Evaluation of camera tracking accuracy on 7-Scenes, Sintel, and Bonn.

Method	7-Scenes - Stride 3			Inference Time↓
	ATE↓	RPE-rot↓	RPE-trans↓	
VGGT	0.087	0.502	**0.022**	38.1s
TurboVGGT(Ours)	**0.082**	**0.498**	**0.022**	**9.6s**
Method	Sintel			Inference Time↓
	ATE↓	RPE-rot↓	RPE-trans↓	
VGGT	0.274	0.473	0.094	1.8s
TurboVGGT(Ours)	**0.210**	**0.464**	**0.059**	**1.5s**
Method	Bonn			Inference Time↓
	ATE↓	RPE-rot↓	RPE-trans↓	
VGGT	**0.013**	**0.294**	**0.007**	7.5s
TurboVGGT(Ours)	**0.013**	0.304	0.009	**3.5s**

A.11 Comparison with FastVGGT and SparseVGGT under Different Sparsity Ratios

Table 15 presents the comparison with FastVGGT and SparseVGGT under different merging/sparsity rates on 7-Scenes (stride 3). The results show that TurboVGGT achieves better speed-accuracy trade-off than FastVGGT and SparseVGGT under all merging/sparsity ratios. It is worth noting that FastVGGT has token merging, salient token selection, reference token selection, token unmerging, and some other operations, so increasing the merging ratio from 90% to a higher value (e.g., 94%) does not necessarily lead to faster inference speed. For SparseVGGT, increasing the sparsity ratio from 75% to a higher value (e.g., 90%) lead to significantly worse reconstruction quality and accuracy though the inference speed is improved.

Table 15: Comparison with FastVGGT and SparseVGGT under different sparsity ratios on 7-Scenes (stride 3).

Method	Point Cloud Estimation		Camera Pose Estimation		Depth Estimation	Time ↓
	Acc↓	NC↑	RRA↑	RTA↑	δ<1.25↑	
TurboVGGT (Ours)	**0.016**	**0.639**	**100.00**	**96.83**	**0.980**	**9.6s**
FastVGGT (90%)	0.018	0.627	99.99	96.29	0.953	14.2s
FastVGGT (94%)	0.018	0.626	**100.00**	96.29	0.953	14.1s
SparseVGGT (75%)	0.021	0.621	**100.00**	93.40	0.956	16.2s
SparseVGGT (90%)	0.034	0.615	97.31	92.79	0.948	10.6s

A.12 Comparison with Fine-tuned FastVGGT and SparseVGGT

FastVGGT and SparseVGGT are proposed as post-hoc acceleration methods, so our main evaluation faithfully follows their official implementations by applying them directly to the vanilla VGGT. End-to-end fine-tuning of them is not defined in the original methods and would require additional method-specific training choices. However, fine-tuning them may also improve their performance. In Table 16, we further report the performance of fine-tuned FastVGGT and SparseVGGT using the same training setup as TurboVGGT. The results show that TurboVGGT achieves better speed-accuracy trade-off than FastVGGT (Fine-tuned) and SparseVGGT (Fine-tuned), demonstrating the effectiveness of our approach.

Table 16: Comparison with FastVGGT and SparseVGGT on 7-Scenes (stride 3).

Method	Point Cloud Estimation		Camera Pose Estimation		Depth Estimation	Inference Time↓	Peak GPU Memory↓	
	Acc↓	NC↑	RRA↑	RTA↑	δ<1.25↑		Inference	Training
TurboVGGT (Ours)	0.016	**0.639**	**100.00**	**96.83**	**0.980**	**9.6s**	**23GB**	**70GB**
FastVGGT (Official)	0.018	0.627	99.99	96.29	0.953	14.2s	31GB	-
FastVGGT (Fine-tuned)	0.017	0.620	**100.00**	96.80	0.979	14.2s	31GB	121GB
SparseVGGT (Official)	0.021	0.621	**100.00**	93.40	0.956	16.2s	28GB	-
SparseVGGT (Fine-tuned, reimplemented sparse attention)	**0.015**	0.623	**100.00**	96.46	0.979	170.9s	50GB	102GB

A.13 Comparison with Fine-tuned FastVGGT Under Higher Merging Rate

We further fine-tune FastVGGT with a higher merging rate of 94% and reported the results in Table 17. The results show that TurboVGGT still achieves better speed-accuracy trade-off than FastVGGT under the higher merging rate. It is worth noting that FastVGGT has token merging, salient token selection, reference token selection, token unmerging, and some other operations, so increasing the merging ratio from 90% to a higher value (e.g., 94%) does not necessarily lead to better speed-accuracy trade-off.

Table 17: Comparison with fine-tuned FastVGGT under higher merging rate on 7-Scenes (stride 3).

Method	Point Cloud Estimation		Camera Pose Estimation		Depth Estimation	Inference Time↓	Peak GPU Memory↓	
	Acc↓	NC↑	RRA↑	RTA↑	δ<1.25↑		Inference	Training
TurboVGGT (Ours)	**0.016**	**0.639**	**100.00**	**96.83**	**0.980**	**9.6s**	**23GB**	**70GB**
FastVGGT (Fine-tuned, 90%)	0.017	0.620	**100.00**	96.80	0.979	14.2s	31GB	121GB
FastVGGT (Fine-tuned, 94%)	0.017	0.619	**100.00**	96.80	0.979	14.1s	31GB	121GB

A.14 Comparison with Co-Me

Co-Me [8] uses a distilled confidence predictor to identify and merge low-confidence tokens for efficient visual geometry transformers. In contrast, TurboVGGT aims to adaptively allocate sparsity across frames and layers and learns representative tokens for cross-view geometry aggregation. Thus, Co-Me focuses on confidence-guided token merging without modifying the pretrained model, whereas TurboVGGT learns an adaptive compressed-attention mechanism during training. We further compare TurboVGGT with Co-Me in Table 18. Inference time is measured with prediction heads included. Following the official implementations provided by the authors, we run the experiments for Co-Me, including the CVPR26 version and the latest accelerated version. These results show that TurboVGGT achieves better speed-accuracy trade-off than Co-Me.

Table 18: Comparison with Co-Me on 7-Scenes (stride 3).

Method	Point Cloud Estimation		Camera Pose Estimation		Depth Estimation	Time ↓
	Acc↓	NC↑	RRA↑	RTA↑	δ<1.25↑	
TurboVGGT (Ours)	**0.016**	**0.639**	**100.00**	**96.83**	**0.980**	**9.6s**
Co-Me (CVPR26 version)	0.026	0.565	**100.00**	95.19	0.950	23.8s
Co-Me (latest accelerated version)	0.027	0.566	**100.00**	94.60	0.945	16.4s

A.15 Long Sequence Reconstruction Quality

In Table 19, we further report the reconstruction quality at long sequences of 500 frames and 1000 frames on the 7-Scenes dataset. The results show that TurboVGGT achieves competitive reconstruction quality compared to the baselines while achieving faster inference speed, demonstrating its effectiveness in long-sequence scenarios.

A.16 Supplementary Visualization

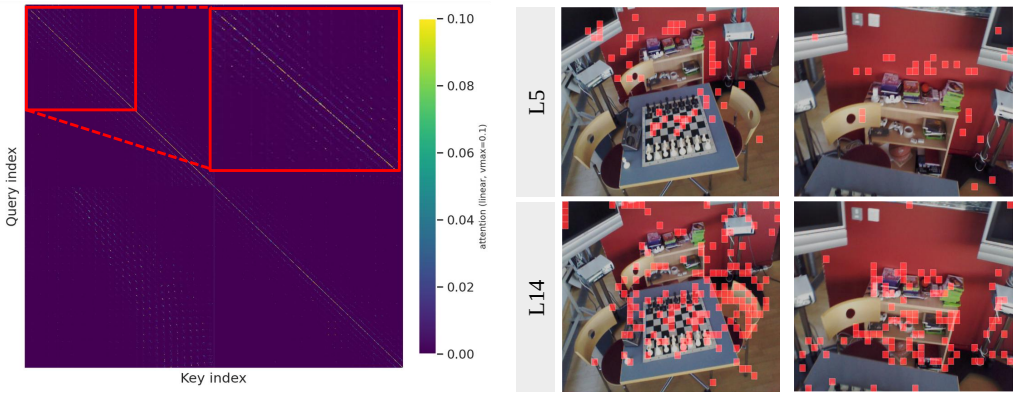
In Figure 5, we provide supplementary visualizations of the motivation for our proposed adaptive alternating attention, which are enlarged visualizations of Figure 3 in the main paper.

A.17 More Qualitative Results

In Figures 6, 7, and 8, we provide more qualitative results of point cloud reconstruction, camera pose estimation, and depth estimation. From these results, we can see that our TurboVGGT can

Table 19: Evaluation on long-sequence 7-Scenes with 1000 and 500 frames.

Method	7-Scenes - 1000 Frames					Time ↓
	Point Cloud Estimation		Camera Pose Estimation		Depth Estimation	
	Acc↓	NC↑	RRA↑	RTA↑	δ<1.25↑	
SparseVGGT	0.022	0.616	99.82	94.38	0.956	113.5s
FastVGGT	0.019	0.623	**100.00**	95.84	0.953	71.5s
TurboVGGT (Ours)	**0.017**	**0.628**	**100.00**	**96.35**	**0.979**	**55.4s**
Method	7-Scenes - 500 Frames					Time ↓
	Point Cloud Estimation		Camera Pose Estimation		Depth Estimation	
	Acc↓	NC↑	RRA↑	RTA↑	δ<1.25↑	
SparseVGGT	0.021	0.620	99.94	94.90	0.956	32.3s
FastVGGT	0.018	0.625	**100.00**	96.13	0.953	27.2s
TurboVGGT (Ours)	**0.017**	**0.627**	**100.00**	**96.66**	**0.980**	**17.8s**



(a) Visualization of a global attention map.

(b) Illustration of highly activated patch tokens across frames in global attention.

Figure 5: Supplementary visualization of the motivation for the proposed adaptive alternating attention (Enlarged visualizations). (a) A visualization of an average global attention map in the alternating attention block. (b) An illustration of highly activated patch tokens of different frames in global attention.

effectively reconstruct point clouds, estimate camera poses, and predict depth maps. These results further demonstrate the superior performance of our TurboVGGT compared with state-of-the-art methods.



Figure 6: Qualitative comparison of point cloud reconstruction.

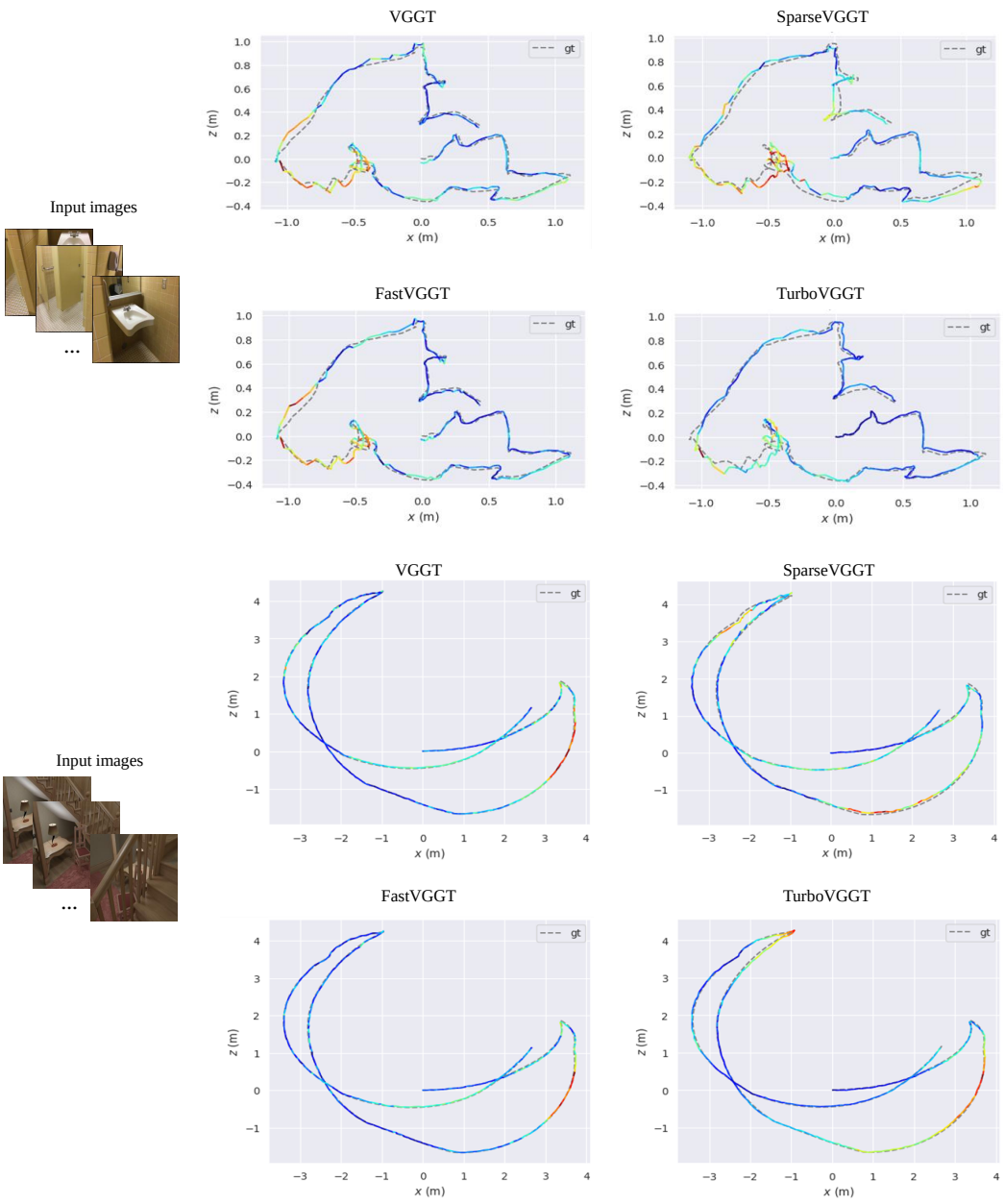


Figure 7: Qualitative comparison of camera pose estimation.

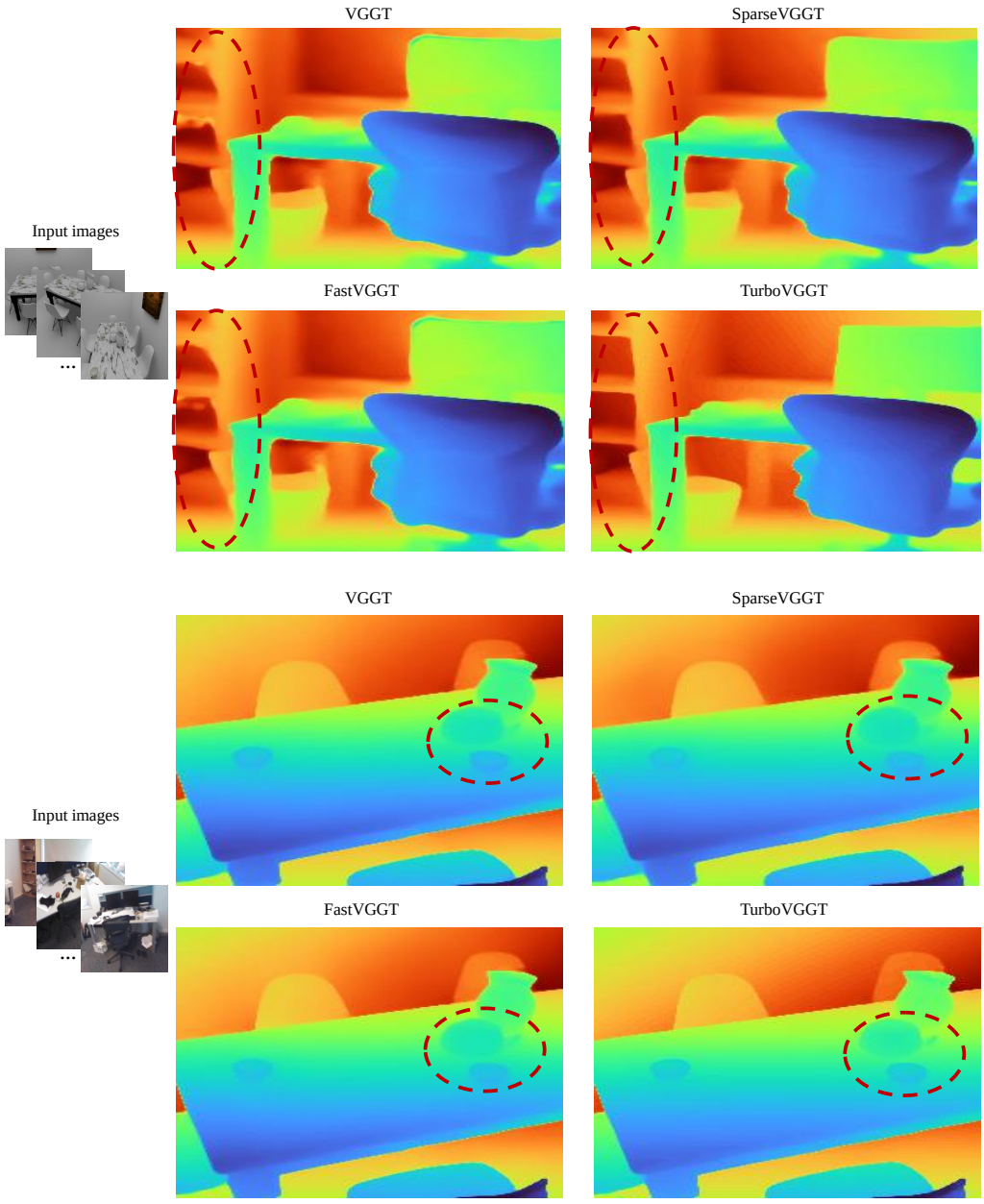


Figure 8: Qualitative comparison of depth estimation.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes].

   Justification: See the Abstract section and the Introduction section of the paper.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes].

   Justification: See the limitation discussion in the Technical Appendices and Supplementary Material section of the paper.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [N/A].

Justification: This work does not include theoretical results, so this question is not applicable.

Guidelines:

- The answer [N/A] means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes].

   Justification: We provide detailed descriptions of the techniques in the Methodology section and the Experiments section of the paper.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
     (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

Answer: [No].

Justification: This work is built on top of publicly available codebases and uses publicly available datasets, and technical details of the proposed approach are described in the Methodology section and the Experiments section of the paper.

Guidelines:

- The answer [N/A] means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
- The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
- The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
- The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
- At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes].

   Justification: See the Experiments section of the paper.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [No].

   Justification: It is not a common practice in this area to report statistical significance. Following existing work, we evaluate our approach using widely adopted benchmarks and evaluation metrics, and our results do not change much across different runs.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)

- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes].

   Justification: See the Experiments section of the paper.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes].

   Justification: This work conforms with the NeurIPS Code of Ethics.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes].

    Justification: See the potential positive and negative societal impacts discussed in the Technical Appendices and Supplementary Material section of the paper.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.

- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A].

    Justification: This work poses no such risks.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes].

    Justification: This work has properly cited the original papers and followed the licenses.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.

- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [N/A].

    Justification: This work does not release new assets.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A].

    Justification: This work does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A].

    Justification: This work does not involve crowdsourcing nor research with human subjects.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
    - We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
    - For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

Answer: [N/A].

Justification: LLMs are not an important, original, or non-standard component of the core methods in this research.

Guidelines:

- The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.