Point3R: Streaming 3D Reconstruction with Explicit Spatial Pointer Memory

Yuqi Wu[1,*] **Wenzhao Zheng**[1,*,†] **Jie Zhou**[1] **Jiwen Lu**[1,2]
[1]Department of Automation, Tsinghua University
[2]Beijing National Research Center for Information Science and Technology
https://ykiwu.github.io/Point3R/

Abstract

Dense 3D scene reconstruction from an ordered sequence or unordered image collections is a critical step when bringing research in computer vision into practical scenarios. Following the paradigm introduced by DUSt3R, which unifies an image pair densely into a shared coordinate system, subsequent methods maintain an implicit memory to achieve dense 3D reconstruction from more images. However, such implicit memory is limited in capacity and may suffer from information loss of earlier frames. We propose Point3R, an online framework targeting dense streaming 3D reconstruction. To be specific, we maintain an explicit spatial pointer memory directly associated with the 3D structure of the current scene. Each pointer in this memory is assigned a specific 3D position and aggregates scene information nearby in the global coordinate system into a changing spatial feature. Information extracted from the latest frame interacts explicitly with this pointer memory, enabling dense integration of the current observation into the global coordinate system. We design a 3D hierarchical position embedding to promote this interaction and design a simple yet effective fusion mechanism to ensure that our pointer memory is uniform and efficient. Our method achieves competitive or state-of-the-art performance on various tasks with low training costs. Code: https://github.com/YkiWu/Point3R.

1 Introduction

Dense 3D reconstruction from image collections has long been a fundamental task in computer vision, with broad applications in fields such as autonomous driving [25, 78], medical modeling, and cultural heritage preservation. Conventional approaches [1, 11, 48, 56, 67, 68] first conduct a sequence of sub-tasks, including feature extraction and matching [33, 41], triangulation, and global alignment to get sparse geometry and camera poses. Multi-view stereo [20, 22, 49] is then used to obtain dense geometry. However, this tightly coupled pipeline is inefficient and tends to be vulnerable to noise. To address these challenges, DUSt3R [64] proposes a data-driven paradigm that directly reconstructs the input image pair as point maps within a unified coordinate system.

Due to the constraint of pair-wise input, DUSt3R requires an additional global alignment step when performing dense reconstruction from multiple images, which is inefficient in multi-view settings. To improve this, subsequent works [59, 60, 63, 71] can be categorized into two main paradigms. One category of these works [60, 71] takes all input images simultaneously and employs global attention to reconstruct them into a unified coordinate system, requiring substantial computational resources and is misaligned with the incremental nature of real-world reconstruction scenarios. The second category [59, 63] introduces an external memory mechanism to retain information from past frames,

[*]Equal contribution. [†]Corresponding author.

39th Conference on Neural Information Processing Systems (NeurIPS 2025).

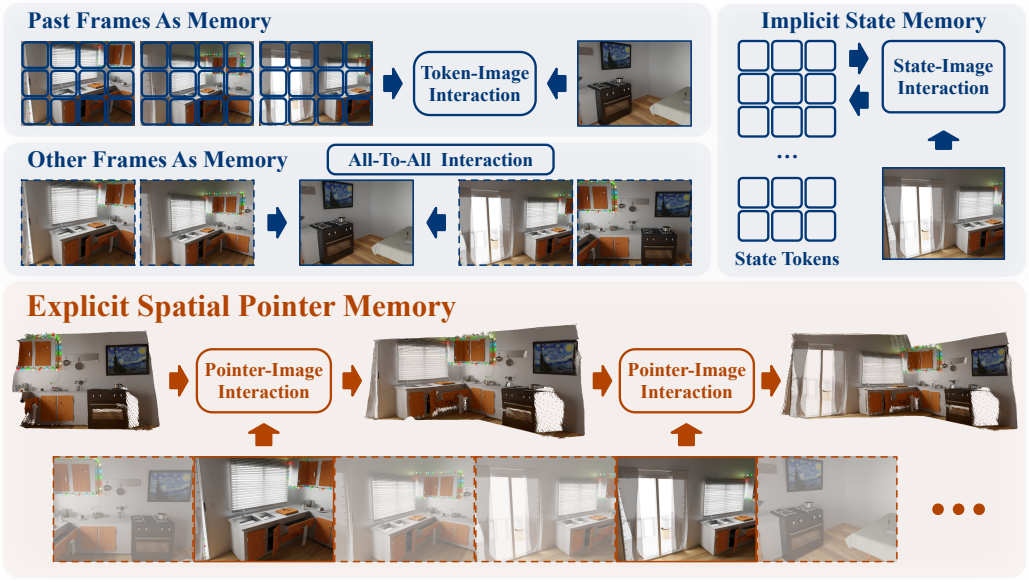

Figure 1: **Comparison between our explicit spatial pointer memory and other paradigms in dense 3D reconstruction.** Methods that conduct all-to-all interaction among all inputs simultaneously [60, 71] can be considered as using **other frames as memory** (for one of the inputs). Methods that cache encoded features of processed frames and conduct token-image interaction [59] can be considered as using **past frames as memory**. Methods maintaining a fixed-length state memory and conducting state-image interaction [63] can be considered as using **implicit state memory**. We propose an **explicit spatial pointer memory** in which each pointer is assigned a 3D position and points to a changing spatial feature. We conduct a pointer-image interaction to integrate new observations into the global coordinate system and update our spatial pointer memory accordingly.

enabling each new input to be directly integrated into the global coordinate system. For instance, Spann3R [59] maintains a memory that essentially caches implicit features of processed frames. However, this implicit memory often contains redundant information, and a discard strategy must be employed once capacity is saturated. CUT3R [63] uses a fixed-length token-based memory mechanism and directly updates it through interactions with image features. Nonetheless, as the number of processed frames increases, this memory inevitably leads to the loss of earlier information.

Inspired by the human memory mechanism, we propose **Point3R**, an online framework equipped with a spatial pointer memory. Human memory of previously encountered environments is inherently related to spatial locations. For example, when we talk about a café or our bedroom, the images we recall are distinct. Similarly, each 3D pointer in our spatial pointer memory is assigned a 3D position in the global coordinate system. Each 3D pointer is directly linked to an explored spatial location and points to a dynamically updated spatial feature. Leveraging this 3D-aware structure, we introduce a 3D hierarchical position embedding, which is integrated into the interaction module between current image features and stored spatial features, enabling more efficient and structured memory querying. Furthermore, since the spatial pointer memory expands as the scene exploration progresses, we design a simple yet effective fusion mechanism to ensure that the memory remains spatially uniform and efficient. Our spatial pointer memory evolves in sync with the current scene, allowing our model to handle both static and dynamic scenes. We use Figure 1 to compare our spatial pointer memory with other paradigms mentioned before. Our method achieves competitive or state-of-the-art performance across various tasks: dense 3D reconstruction, monocular and video depth estimation, and camera pose estimation. It is worth mentioning that although trained on a variety of datasets, the training of our method has a low cost in time and computational resources.

2 Related Work

Conventional 3D Reconstruction. Classic approaches to 3D reconstruction from image collections are typically optimization-based and tailored to specific scenes. Structure-from-motion

(SfM) [1, 11, 48, 56, 67, 68] follows a pipeline consisting of feature extraction [16, 34, 46], image matching [9, 31, 50, 68], triangulation to 3D, and bundle adjustment [2, 58] to obtain sparse geometry and estimated camera poses. Building upon this, subsequent methods such as Multi-view Stereo (MVS) [18, 19, 49, 65], Neural Radiance Fields (NeRF) [4, 8, 36, 37, 62], and 3D Gaussian Splatting (3DGS) [26] leverage known camera parameters to recover dense geometry or high-fidelity scene representation. These approaches rely on a sequential combination of multiple modules, which not only require considerable optimization time but are also vulnerable to noise. It is worth noting that Simultaneous Localization and Mapping (SLAM) [13, 17, 27, 40] can perform localization and reconstruction in an online manner. However, they often rely on specific camera motion assumptions [13, 40] (sometimes these motion assumptions may be misleading or restrictive) or require additional depth/LiDAR sensors [39] for better performance.

Learning-Based 3D Reconstruction. To enhance accuracy and efficiency, the field of 3D reconstruction is gradually shifting toward learning-based and end-to-end paradigms. Some approaches utilize learnable modules to replace hand-crafted components [14, 47] during the traditional reconstruction pipeline. Some others try to optimize the overall pipeline in an end-to-end manner [57, 61, 72]. DUSt3R [64] introduces a pointmap representation and directly learns to integrate an image pair into the same coordinate system, which unifies all sub-tasks we have mentioned above. MonST3R [75] extends this paradigm to dynamic scenes by fine-tuning it on dynamic datasets. However, this pair-wise formulation needs a global alignment to process more views, which is computationally intensive and time-consuming. Subsequent works [59, 60, 63, 71] are exploring how to further replace the global optimization step with an end-to-end learning framework. These efforts can be broadly categorized into two main branches. The former [60, 71] feeds all images simultaneously and leverages global attention to reconstruct the scene within a unified coordinate system; the latter [59, 63] proposes a streaming paradigm, in which a memory module stores information from past frames, thereby enabling online incremental reconstruction from sequential inputs. The streaming paradigm aligns more closely with real-world applications, offering improved scalability without imposing excessive computational demands.

Streaming Reconstruction and Memory Mechanism. The streaming reconstruction paradigm and the memory mechanism are inherently aligned with each other. Existing streaming reconstruction methods [10, 17, 59, 63, 76] universally incorporate a certain form of memory to store information from past frames. This memory can take on various forms, such as explicit scene representation (the most direct form of memory), recurrent neural network architectures [10], and encoded or learnable token features [59, 63]. Explicit 3D scene representation is compact and efficient but tailored to a specific scene, limiting its generalizability. Spann3R [59] stores implicit features from previous frames in the memory, which may lead to redundancy. CUT3R [63] employs a fixed-length set of learnable token features as its memory module, which is continuously updated during sequential processing. However, its limited capacity can lead to information loss. In contrast, we propose a spatial pointer memory, in which each pointer is dynamically assigned a 3D position. This ensures that the total amount of stored information scales naturally with the extent of the explored scene. Furthermore, each pointer has a spatial feature that captures aggregated scene information nearby.

3 Proposed Approach

3.1 Memory-Based Streaming 3D Reconstruction

To densely reconstruct image collections $\mathbf{I} \in \mathbb{R}^{N\times H\times W\times 3}$ into a unified global coordinate system as per-frame pointmaps $\mathbf{X} \in \mathbb{R}^{N\times H\times W\times 3}$, existing methods can be categorized into three paradigms: (1) pair-wise reconstruction [29, 64, 75] with an optimization-based global alignment, (2) one-shot global reconstruction [60, 71] with all inputs, and (3) frame-by-frame input and reconstruction [59, 63] based on a memory mechanism. Pair-wise methods take only one image pair each time as input and reconstruct it into a local coordinate system. A global alignment is then conducted to merge all local outputs into a global coordinate system. This approach suffers from inefficiency due to the need for repeated pair-wise processing and a post-processing stage. The second category feeds all images into the model simultaneously and employs a global attention to reconstruct them directly in a shared coordinate system. However, this approach is computationally intensive and inherently mismatched with the core demands of embodied agents, which typically perform streaming perception of their surroundings and respond correspondingly in practical scenarios.

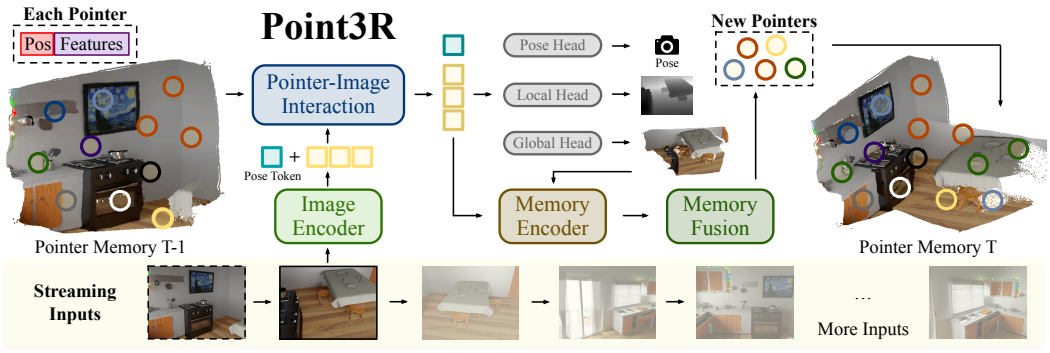

Figure 2: **Overview of Point3R.** Given streaming image inputs, our method maintains an explicit spatial pointer memory to store the observed information of the current scene. We use a ViT [15, 64] encoder to encode the current input into image tokens and use ViT-based decoders to conduct interaction between image tokens and spatial features in the memory. We use two DPT [43] heads to decode local and global pointmaps from the output image tokens. Besides, a learnable pose token is added during this stage so we can directly decode the camera parameters of the current frame. Then we use a simple memory encoder to encode the current input and its integrated output into new pointers, and use a memory fusion mechanism to enrich and update our spatial pointer memory.

In pursuit of a balance between practicality and efficiency, the key idea of memory-based methods is maintaining a memory that stores observed information and interacts with each incoming frame to enable streaming dense reconstruction. We formulate the memory-based pipeline as follows:

$$X_t = \mathcal{F}(\mathcal{M}_{t-1}, I_t), \tag{1}$$

where $I_t \in \mathbb{R}^{H\times W\times 3}$ and $X_t \in \mathbb{R}^{H\times W\times 3}$ are the current image input and pointmap output, $\mathcal{F}$ is a certain approach and $\mathcal{M}_{t-1}$ represents a memory with integrated information from the past frames.

To elaborate, in Spann3R [59], $\mathcal{M}$ is a growing set of key-value-pair features, where the features are encoded from the output of each previous frame. In CUT3R [63], $\mathcal{M}$ takes the form of a fixed-length feature sequence that is iteratively updated as new frame comes. In this work, we propose an explicit spatial pointer memory $\mathcal{M}$ that stores a set of 3D pointers corresponding to the explored regions of the current scene, along with their associated spatial features. We argue that this explicit memory enables compact and structured integration of information from past observations. Each pointer is indexed by a 3D position in the global coordinate system, rather than implicit features, making the interaction between the memory and the current frame more direct and efficient.

3.2 Pointer-Image Interaction

The core of our method is making the interaction between the current input and our spatial pointer memory more effective and efficient. We will elaborate on the overall model architecture, which is composed of an image encoder, interaction decoders, a memory encoder, and a memory fusion mechanism. We also incorporate a 3D hierarchical position embedding into the interaction decoders to promote our pointer-image interaction. Figure 2 shows the overview of our method.

Image Encoder. For each frame, we use a ViT [15] to encode the current input I_t into image tokens F_t:

$$F_t = \text{Encoder}(I_t). \tag{2}$$

Interaction Decoders. Our spatial pointer memory $\mathcal{M}$ consists of a set of 3D pointers (3D positions P and spatial features M). Before processing the first frame, the memory has not yet stored any global spatial features of the current scene. Therefore, we use a simple layer to embed the image tokens F_0 of the first frame, and use the output M_0 to initialize the features of our memory. It is worth noting that, since the precise spatial positions represented by these features are not yet known at this time, each feature has not been assigned a specific 3D position. Then we use two intertwined decoders [64, 66] to enable interaction between the current image tokens and the memory:

$$F'_0, z'_0 = \text{Decoders}((F_0, z_0), M_0), \tag{3}$$

$$F'_t, z'_t = \text{Decoders}((F_t, z_t), M_{t-1}), \tag{4}$$

where we use a learnable pose token z as a bridge between the current frame and the global coordinate system. After this interaction, we get the updated image tokens F_t' and pose token z_t'. Then we use F_t' and z_t' to predict two pointmaps ($\hat{X}_t^{self}$ in the local coordinate system of the current input and $\hat{X}_t^{global}$ in the global coordinate system) with their own confidence maps ($C_t{}^{self}$ and $C_t{}^{global}$), and a camera pose $\hat{T}_t$ representing the rigid transformation between this two coordinate system:

$$\hat{T}_t = \text{Head}_{pose}(z_t'), \tag{5}$$

$$\hat{X}_t^{self}, C_t{}^{self} = \text{Head}_{self}(F_t'), \tag{6}$$

$$\hat{X}_t^{global}, C_t{}^{global} = \text{Head}_{global}(F_t', z_t'), \tag{7}$$

where Head_{pose} is an MLP network, Head_{self} and Head_{global} are DPT [43] heads. The global coordinate system is actually the first input's own coordinate system.

Memory Encoder. After processing I_t, we use the current features $F_t, F_t'$and the predicted pointmap $\hat{X}_t^{global}$ to obtain the new pointers:

$$P_{new}(u,v) = \frac{1}{|R_{u,v}|} \sum_{(i,j)\in R_{u,v}} \hat{X}_t^{global}(i,j), \tag{8}$$

$$M_{new} = \text{Encoder}_f(F_t, F_t') + \text{Encoder}_{geo}(\hat{X}_t^{global}), \tag{9}$$

where M_{new} is the set of new spatial features and we compute the 3D location $P_{new}(u,v)$ for each feature in the global coordinate system as its 3D position by averaging all 3D coordinates within the corresponding patch $R_{u,v}$. Besides, Encoder_f is a MLP and Encoder_{geo} is a lightweight ViT.

Memory Fusion Mechanism. Apart from the first frame when we simply add all the obtained pointers into the memory, new pointers extracted from each subsequent frame I_t are integrated into the existing memory $\mathcal{M}_{t-1}$ through a fusion mechanism. To elaborate, we compute the Euclidean distance between each new pointer and all existing pointers from the memory to identify its nearest neighbor. If the distance to its nearest neighbor in the memory is below a changing threshold δ (we change δ accordingly to make the distribution of memory units more uniform), we treat the neighbors as corresponding and perform the fusion. Otherwise, the new pointer is directly added to the memory. Notably, if a memory pointer is identified as the nearest neighbor by one or a few new pointers, we update the position p and spatial feature m of this pointer as follows:

$$p' = \frac{1}{K}\sum_{i=1}^{K} p_i^{new}, m' = \frac{1}{K}\sum_{i=1}^{K} m_i^{new}, \tag{10}$$

where K is the total number of new neighbors of the target pointer. This fusion mechanism ensures that each pointer always stores the current spatial information at its location, thereby enabling the memory to deal with dynamic scenes. In this way, we obtain an enriched and updated memory $\mathcal{M}_t$.

3D Hierarchical Position Embedding. We expand the rotary position embedding (RoPE [23, 54], a method of relative position embedding usually used in transformers) to a 3D hierarchical position embedding and use this to conduct position embedding in continuous 3D space. In practical implementation, RoPE utilizes multiple frequencies θ_t using channel dimensions d_{head} of key and query as

$$\theta_t = 10000^{-t/(d_{head}/2)}, \text{where } t \in \{0, 1, ..., d_{head}/2\}. \tag{11}$$

Then a rotation matrix $\mathbf{R} \in \mathbb{C}^{N\times(d_{head}/2)}$ is defined as

$$\mathbf{R}(n,t) = e^{i\theta_t n}, \tag{12}$$

and applied to query and key with the Hadamard product $\circ$ as

$$\bar{\mathbf{q}}' = \bar{\mathbf{q}} \circ \mathbf{R},\ \bar{\mathbf{k}}' = \bar{\mathbf{k}} \circ \mathbf{R},\ \mathbf{A}' = \text{Re}[\bar{\mathbf{q}}'\bar{\mathbf{k}}'^{*}]. \tag{13}$$

Here, the attention matrix with RoPE $\mathbf{A}'$ implies relative position in rotation form $e^{i(n-m)\theta_t}$. Inspired by this, we formulate a 3D hierarchical position embedding. We need to change the 1D token index n in RoPE to a 3D token position $p_n = (p_n^x, p_n^y, p_n^z)$ where p_n^x, p_n^y and p_n^z correspond to the coordinates on three axes in a continuous 3D space. Thus, the rotation matrix $\mathbf{R} \in \mathbb{C}^{N\times(d_{head}/2)}$ in Eq. 12 is changed accordingly as

$$\mathbf{R}(n,3t) = e^{i\theta_t p_n^x},\ \mathbf{R}(n,3t+1) = e^{i\theta_t p_n^y},\ \mathbf{R}(n,3t+2) = e^{i\theta_t p_n^z}. \tag{14}$$

To accommodate spatial position inputs of varying scales, we use h different frequency bases (10000 in Eq. 11) to derive hierarchical rotation matrices and apply them to query and key as

$$\bar{\mathbf{q}}' = \frac{1}{h}\sum_{i=1}^{h}(\bar{\mathbf{q}} \circ \mathbf{R}_i),\ \bar{\mathbf{k}}' = \frac{1}{h}\sum_{i=1}^{h}(\bar{\mathbf{k}} \circ \mathbf{R}_i),\ \mathbf{A}' = \mathrm{Re}[\bar{\mathbf{q}}'\bar{\mathbf{k}}'^{*}]. \tag{15}$$

We use this hierarchical position embedding in our interaction decoders to inject relative position information into image tokens and memory features. When applying the rotation matrices in Eq. 15, the 3D token position of each memory feature is its 3D position. For image tokens from I_t, the 3D token position assigned to each of them is as follows:

$$p(u,v) = \frac{1}{|R_{u,v}|}\sum_{(i,j)\in R_{u,v}} \hat{X}_{t-1}^{global}(i,j), \tag{16}$$

where $R_{u,v}$ is the corresponding image patch and we use $\hat{X}_{t-1}^{global}$ from $t-1$ because we assume that the image tokens of the current frame are more likely to be spatially close to those of the previous frame. Of course, even in cases of significant pose shifts or unordered inputs, this assumption will not introduce any adverse effects. This is because the image tokens of the current frame interact with all memory features in our interaction decoders, and our hierarchical position embedding merely provides a potential prior. Due to the space limit, we will add more details about the design and implementation of our 3D hierarchical position embedding in the supplementary material.

3.3 Training Strategy

Training Objective. Following MASt3R [29] and CUT3R [63], we use the L2 norm loss for the poses and a confidence-aware loss for the pointmaps. In practical implementation, we parameterize the predicted pose $\hat{T}_t$ as quaternion $\hat{q}_t$ and translation $\hat{\tau}_t$. We use $\hat{\mathcal{X}} = \{\hat{\mathcal{X}}^{self}, \hat{\mathcal{X}}^{global}\}$ to denote the predicted pointmaps, where $\hat{\mathcal{X}}^{self} = \{\hat{X}_t^{self}\}_{t=1}^{N}$, $\hat{\mathcal{X}}^{global} = \{\hat{X}_t^{global}\}_{t=1}^{N}$ and N is the number of images per sequence. Besides, $\mathcal{C}$ is used to denote the set of confidence scores correspondingly. So the final expression of the loss we used is:

$$\mathcal{L}_{pose} = \sum_{t=1}^{N}\left(\|\hat{q}_t - q_t\|_2 + \left\|\frac{\hat{\tau}_t}{\hat{s}} - \frac{\tau_t}{s}\right\|_2\right), \tag{17}$$

$$\mathcal{L}_{conf} = \sum_{(\hat{x},c)\in(\hat{\mathcal{X}},\mathcal{C})}\left(c\cdot\left\|\frac{\hat{x}}{\hat{s}} - \frac{x}{s}\right\|_2 - \alpha\log c\right), \tag{18}$$

where $\hat{s}$ and s are scale normalization factors for $\hat{\mathcal{X}}$ and $\mathcal{X}$. When the groundtruth pointmaps are metric, we set $\hat{s} := s$ to force the model to learn metric-scale results.

Training Datasets. During training, we use a combination of 14 datasets, including ARKitScenes [5], ScanNet [12], ScanNet++ [74], CO3Dv2 [44], WildRGBD [70], OmniObject3D [69], HyperSim (a subset of it) [45], BlendedMVS [73], MegaDepth [30], Waymo [55], VirtualKITTI2 [7], PointOdyssey [79], Spring [35], and MVS-Synth [24]. These datasets exhibit highly diverse characteristics, encompassing both indoor and outdoor, static and dynamic, as well as real-world and synthetic scenes. See the supplementary material for more details.

Training Stages. Our model is trained in three stages. We train the model by sampling 5 frames per sequence in the first stage. The input here is 224×224 resolution. Then we use input with different aspect ratios (set the maximum side to 512) in the second stage, following CUT3R [63]. And finally, we freeze the encoder and fine-tune other parts on 8-frame sequences.

Implementation Details. We initialize our ViT-Large [15, 64] image encoder, ViT-Base interaction decoders [64, 66], and DPT [43] heads with pre-trained weights from DUSt3R [64]. Our memory encoder is composed of a light-weight ViT (6 blocks) and a 2-layer MLP. Each memory feature has a dimensionality of 768. We use the AdamW optimizer [32] and the learning rate warms up to a maximum value of 5e-5 and decreases according to a cosine schedule. We train our model on 8 A800 NVIDIA GPUs for 15 days, which is a relatively low cost.

Table 1: **Quantitative 3D reconstruction results on 7-scenes and NRGBD datasets.** We use "GA" to mark methods with global alignment, and use "Optim." and "Onl." to distinguish between optimization-based and online methods [63]. Our method achieves competitive or better performance than those optimization-based methods and current online methods.

			7-scenes						NRGBD					
			Acc↓		Comp↓		NC↑		Acc↓		Comp↓		NC↑	
Method	**Optim.**	**Onl.**	Mean	Med.	Mean	Med.	Mean	Med.	Mean	Med.	Mean	Med.	Mean	Med.
DUSt3R-GA [64]	✓		0.146	0.077	0.181	0.067	0.736	0.839	0.144	**0.019**	0.154	**0.018**	**0.870**	**0.982**
MASt3R-GA [29]	✓		0.185	0.081	0.180	0.069	0.701	0.792	0.085	0.033	**0.063**	0.028	0.794	0.928
MonST3R-GA [75]	✓		0.248	0.185	0.266	0.167	0.672	0.759	0.272	0.114	0.287	0.110	0.758	0.843
Spann3R [59]		✓	0.298	0.226	0.205	0.112	0.650	0.730	0.416	0.323	0.417	0.285	0.684	0.789
CUT3R [63]		✓	0.126	0.047	0.154	0.031	0.727	0.834	0.099	0.031	0.076	0.026	0.837	0.971
Ours		✓	**0.085**	**0.046**	**0.087**	**0.030**	**0.739**	**0.854**	**0.077**	0.030	0.069	0.027	0.835	0.971

Table 2: **Monocular Depth Evaluation** on NYU-v2 (static), Sintel, Bonn, and KITTI datasets.

	NYU-v2 (Static)		Sintel		Bonn		KITTI	
Method	Abs Rel ↓	δ<1.25 ↑	Abs Rel ↓	δ<1.25 ↑	Abs Rel ↓	δ<1.25 ↑	Abs Rel ↓	δ<1.25 ↑
DUSt3R [64]	0.080	90.7	0.424	58.7	0.141	82.5	0.112	86.3
MASt3R [29]	0.129	84.9	**0.340**	**60.4**	0.142	82.0	**0.079**	**94.7**
MonST3R [75]	0.102	88.0	0.358	54.8	0.076	93.9	0.100	89.3
Spann3R [59]	0.122	84.9	0.470	53.9	0.118	85.9	0.128	84.6
CUT3R [63]	0.086	90.9	0.428	55.4	0.063	**96.2**	0.092	91.3
Ours	**0.078**	**92.3**	0.395	55.0	**0.061**	94.5	0.083	94.6

4 Experiments

Tasks and Baselines. We use various 3D/4D tasks (dense 3D reconstruction, monocular depth estimation, video depth estimation, and camera pose estimation) to evaluate our method. We choose DUSt3R [64], MASt3R [29], MonST3R [75], Spann3R [59], and CUT3R [63] as our primary baselines. Among these methods, DUSt3R, MASt3R, and MonST3R take an image pair as input, so an optimization-based global alignment (GA) stage is conducted when dealing with streaming inputs. Both Spann3R and CUT3R have a memory module so they can process an image sequence in an online manner, similar to our method.

4.1 3D Reconstruction

We evaluate the 3D reconstruction performance on the 7-scenes [51] and NRGBD [3] datasets in Table 1, and the metrics we used include accuracy (Acc), completion (Comp), and normal consistency (NC), following previous works [3, 59, 64]. We use inputs with minimal overlap [63]: 3 to 5 frames per scene for the 7-scenes datasets and 2 to 4 frames per scene for the NRGBD dataset. Such sparsely sampled inputs can directly demonstrate the effectiveness of our proposed pointer memory, which is explicitly associated with 3D spatial locations and does not rely on similarity or continuity between input frames. As shown in Table 1, our method achieves comparable or better results than other memory-based online approaches or even DUSt3R-GA [64]. We compare the reconstruction quality of our method with other memory-based online approaches, Spann3R [59] and CUT3R [63] in Figure 3, and our method achieves state-of-the-art reconstruction performance with sparse inputs from the 7-scenes and NRGBD datasets.

4.2 Monocular and Video Depth Estimation

Monocular Depth Estimation. We evaluate zero-shot monocular depth estimation [63, 75] performance on NYU-v2 [52] (static), Sintel [6], Bonn [42], and KITTI [21] datasets. We adopt per-frame median scaling following DUSt3R [64], and the evaluation metrics we used include absolute relative error (Abs Rel) and percentage of inlier points $\delta < 1.25$. As shown in Table 2, our method achieves state-of-the-art or competitive performance in both static and dynamic, indoor and outdoor scenes.

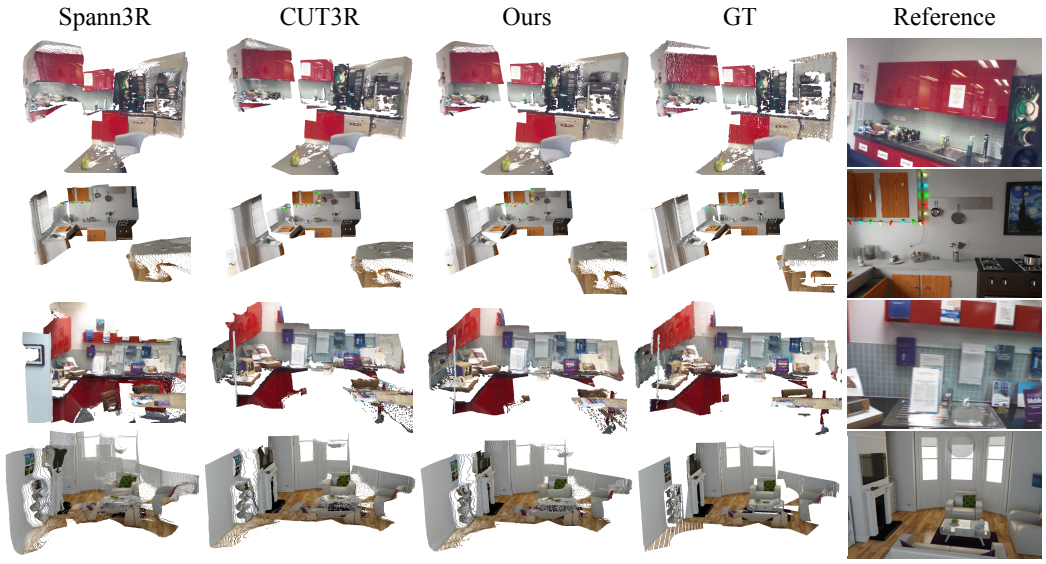

Figure 3: **Qualitative results on sparse inputs from the 7-scenes and NRGBD datasets.** Our method achieves the best qualitative results among memory-based methods.

Table 3: **Video Depth Evaluation.** We compare scale-invariant depth (per-sequence alignment) and metric depth (no alignment) results on Sintel, Bonn, and KITTI datasets.

				Sintel		BONN		KITTI	
Alignment	**Method**	**Optim.**	**Onl.**	Abs Rel ↓	δ<1.25 ↑	Abs Rel ↓	δ<1.25 ↑	Abs Rel ↓	δ <1.25 ↑
Per-sequence	DUSt3R-GA [64]	✓		0.656	45.2	0.155	83.3	0.144	81.3
	MASt3R-GA [29]	✓		0.641	43.9	0.252	70.1	0.183	74.5
	MonST3R-GA [75]	✓		**0.378**	**55.8**	0.067	**96.3**	0.168	74.4
	Spann3R [59]		✓	0.622	42.6	0.144	81.3	0.198	73.7
	CUT3R [63]		✓	0.421	47.9	0.078	93.7	0.118	88.1
	Ours		✓	0.481	44.8	**0.066**	95.8	**0.093**	**93.5**
Metric-scale	MASt3R-GA [29]	✓		**1.022**	14.3	0.272	70.6	0.467	15.2
	CUT3R [63]		✓	1.029	**23.8**	0.103	88.5	**0.122**	**85.5**
	Ours		✓	1.208	13.8	**0.081**	**95.8**	0.169	80.5

Video Depth Estimation. We align predicted depth maps to ground truth using a per-sequence scale (Per-sequence alignment) to evaluate per-frame quality and inter-frame consistency. We also compare results without alignment with other metric pointmap methods like MASt3R [29] and CUT3R [63] (Metric-scale alignment). As shown in Table 3, with the per-sequence scale alignment, our method outperforms DUSt3R [64], MASt3R [29], and Spann3R [59] by a large margin. These methods have a static scene assumption and are trained only on static datasets. Our spatial pointer memory directly associates spatial features with their real-world positions and imposes no assumptions or dependencies on whether the scene is static or dynamic. Our method performs comparably, or even better than MonST3R-GA [75] and CUT3R [63] (methods trained on dynamic datasets). Besides, in the metric-scale setting, our method outperforms MASt3R-GA [29] and performs comparably with CUT3R [63], leading on Bonn.

4.3 Camera Pose Estimation

We evaluate the camera pose estimation performance on ScanNet [12] (static), Sintel [6], and TUM-dynamics [53] datasets following MonST3R [75] and CUT3R [63]. We report Absolute Translation Error (ATE), Relative Translation Error (RPE trans), and Relative Rotation Error (RPE rot) after applying a Sim(3) Umeyama alignment with the ground truth [75]. It is worth noting that prior approaches conduct additional optimization while our method does not require any post-processing. Results in Table 4 show that our method performs comparably with other online methods, but there persists a performance gap between those optimization-based baselines and our method.

Table 4: **Camera Pose Estimation Evaluation** on ScanNet, Sintel, and TUM-dynamics datasets.

Method	Optim.	Onl.	ScanNet (Static)			Sintel			TUM-dynamics		
			ATE ↓	RPE trans ↓	RPE rot ↓	ATE ↓	RPE trans ↓	RPE rot ↓	ATE ↓	RPE trans ↓	RPE rot ↓
Robust-CVD [28]	✓		0.227	0.064	7.374	0.360	0.154	3.443	0.153	0.026	3.528
CasualSAM [77]	✓		0.158	0.034	1.618	0.141	**0.035**	**0.615**	0.071	**0.010**	1.712
DUSt3R-GA [64]	✓		0.081	0.028	0.784	0.417	0.250	5.796	0.083	0.017	3.567
MASt3R-GA [29]	✓		0.078	0.020	**0.475**	0.185	0.060	1.496	**0.038**	0.012	**0.448**
MonST3R-GA [75]	✓		**0.077**	**0.018**	0.529	**0.111**	0.044	0.869	0.098	0.019	0.935
Spann3R [59]		✓	0.096	0.023	0.661	0.329	0.110	4.471	0.056	0.021	0.591
CUT3R [63]		✓	0.099	0.022	0.600	0.213	0.066	0.621	0.046	0.015	0.473
Ours		✓	0.097	0.035	2.791	0.442	0.154	1.897	0.058	0.031	0.758

Table 5: **Quantitative 3D reconstruction results on long sequences.**

Method	7-scenes						NRGBD					
	Acc↓		Comp↓		NC↑		Acc↓		Comp↓		NC↑	
	Mean	Med.	Mean	Med.	Mean	Med.	Mean	Med.	Mean	Med.	Mean	Med.
CUT3R [63]	0.238	0.172	0.105	0.025	0.527	0.537	0.372	0.270	0.211	0.090	0.556	0.582
Ours	**0.071**	**0.033**	**0.031**	**0.015**	**0.558**	**0.587**	**0.110**	**0.050**	**0.025**	**0.009**	**0.641**	**0.729**

4.4 Analysis and Discussion

Comparison on Long Sequences. We compare our model with CUT3R [63] on long sequences to show the advantage of our explicit spatial pointer memory to effectively store more information from past frames. We resampled the testing sequences from 7-scenes at the interval of 1 (each contains 500-1000 frames) and NRGBD at the interval of 2 (400-900 frames). As shown in the Table 5, our method outperforms CUT3R by a large margin, showing the advantage of our model when handling long sequences in practical applications.

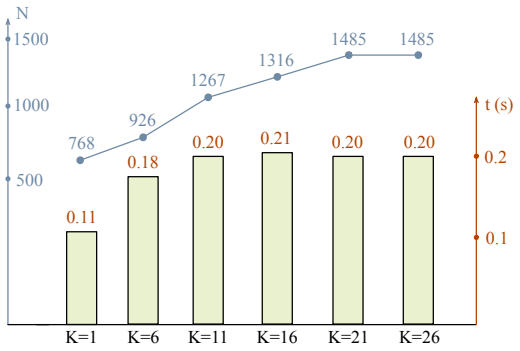

Figure 4: Changes on **the total number of pointers** and **per-frame runtime** with memory fusion.

Robustness to the Input Orders. We evaluate the robustness of our model to the order of camera registration. To be specific, for each scene in 7-scenes, we sample a sequence with a frame interval of 20 (each sequence contains 25-50 frames). For each scene in NRGBD, we sample a sequence with a frame interval of 40 (each sequence contains 20-40 frames). Then we disrupt the input sequence after the sampling to evaluate the reconstruction results. From Table 6, we can observe that our method still achieves good reconstruction performance after the input sequence is shuffled. This verifies the robustness of our explicit pointer memory to handle the discontinuity of the inputs.

Effect of the Memory Fusion Mechanism and Runtime Analysis. We design the memory fusion mechanism to get a balance between efficiency and performance. Figure 4 shows the changes in the number of pointers N in the memory (line graph) and per-frame runtime t (histogram) with the increasing number of frames K when processing Scene *WhiteRoom* (from NRGBD dataset). We can see that this memory fusion mechanism can control the total number of pointers and per-frame runtime within a reasonable range. We also report the 3D reconstruction results without the memory fusion mechanism on the 7-scenes and NRGBD datasets in Table 7. Although this fusion mechanism may result in a slight decrease in some metrics, we believe that the sacrifice made for efficiency improvement is worthwhile.

Effect of the 3D Hierarchical Position Embedding. To demonstrate the effectiveness of our 3D hierarchical position embedding, we conduct ablation studies for this module in Table 7. We report the 3D reconstruction results without the 3D hierarchical position embedding ("w/o 3DHPE"), which shows the effectiveness of the elaborate position embedding we proposed.

Comparison with SLAM Systems and More Discussions. Traditional SLAM systems and the recent feed-forward 3D reconstruction methods (e.g., DUSt3R [64], CUT3R [63], and ours) are similar

Table 6: **Robustness to the input orders.**

Sample	7-scenes						NRGBD					
	Acc↓		Comp↓		NC↑		Acc↓		Comp↓		NC↑	
	Mean	Med.	Mean	Med.	Mean	Med.	Mean	Med.	Mean	Med.	Mean	Med.
Ordered	0.032	0.016	0.024	0.008	0.665	0.758	0.064	0.031	0.029	0.011	0.801	0.949
Shuffled	0.033	0.014	0.019	0.010	0.669	0.764	0.063	0.027	0.029	0.009	0.800	0.953

Table 7: **Effects of the 3D hierarchical position embedding and the memory fusion mechanism.**

	7-scenes			NRGBD		
	Acc↓	Comp↓	NC↑	Acc↓	Comp↓	NC↑
Ours (w/o 3DHPE)	0.142	**0.132**	0.698	0.083	0.079	0.808
Ours (w/o Mem-Fusion)	**0.118**	0.148	0.721	**0.079**	0.074	**0.824**
Ours	0.124	0.139	**0.725**	**0.079**	**0.073**	**0.824**

in terms of output format (reconstruction results and predicted poses), but their core objectives are different. The main objective of SLAM lies in accurately registering the camera poses from RGB-D inputs (where the depth could be obtained from prediction models). Differently, feed-forward 3D reconstruction methods aim to develop a unified model to predict 3D reconstructed points for each frame in a shared coordinate system (and some recent methods simultaneously predict the camera pose with an additional head). To better evaluate these two paradigms, we compare our method with MASt3R-SLAM [38] on its own established benchmarks.

We conduct the dense geometry evaluation on 7-Scenes seq-01 and also report the RMSE of the absolute trajectory error (ATE) in meters. We also compare these two methods in terms of peak memory usage and per-frame runtime. As shown in Table 8, our method achieves better dense geometry reconstruction results with lower memory consumption. However, there is still room for further improvement in tracking accuracy (as discussed in Limitations) and runtime efficiency. We hold the belief that these two paradigms are compatible and essentially beneficial to each other. Visual SLAM systems could use a feed-forward 3D reconstruction model to produce reconstructed points (just like MASt3R-SLAM [38]). Feed-forward 3D reconstruction methods can further employ SLAM techniques such as bundle adjustment to obtain more accurate poses.

Table 8: **Comparison with SLAM in mapping quality, tracking accuracy and efficiency.**

	Acc↓	Comp↓	ATE↓	Peak Memory Usage↓	Per-frame Runtime↓
MASt3R-SLAM [38]	0.068	0.045	**0.066**	7.18 GB	0.11 s
Ours	**0.061**	**0.022**	0.084	5.46 GB	0.20 s

5 Conclusion and Discussions

In this paper, we have presented an online streaming 3D reconstruction framework, Point3R, with a spatial pointer memory. When processing streaming inputs, our method maintains a growing spatial pointer memory in which each pointer is assigned a specific 3D position and aggregates scene information nearby with a changing spatial feature. Equipped with a 3D hierarchical position embedding and a simple yet effective memory fusion mechanism, our method imposes minimal constraints on the input, handling both static and dynamic scenes as well as ordered or unordered image collections. With a low training cost, our method achieves competitive or state-of-the-art performance on various 3D/4D tasks, which verifies the effectiveness of our method.

Limitations. As the explored area expands, the positions where pointers are stored also grow progressively, which may introduce additional interference to camera pose estimation in subsequent frames. One of our future works is improving the pointer-image interaction to mitigate this issue.

Broader Impacts. Our method facilitates scalable and efficient dense streaming 3D scene reconstruction, benefiting a wide range of applications. The explicit and interpretable design of our pointer memory makes our method more transparent and adaptable to different real-world scenarios.

Acknowledgements

This work was supported in part by the National Key Research and Development Program of China under Grant 2023YFB280693, in part by the National Natural Science Foundation of China under Grant 62125603, Grant 62336004, Grant 62576188, and Grant 62321005, in part by the Beijing Natural Science Foundation under Grant L247009, and in part by the Beijing National Research Center for Information Science and Technology.

References

[1] Sameer Agarwal, Yasutaka Furukawa, Noah Snavely, Ian Simon, Brian Curless, Steven M Seitz, and Richard Szeliski. Building rome in a day. *Communications of the ACM*, 54(10):105–112, 2011. 1, 3

[2] Sameer Agarwal, Noah Snavely, Steven M Seitz, and Richard Szeliski. Bundle adjustment in the large. In *CVPR*, pages 29–42. Springer, 2010. 3

[3] Dejan Azinović, Ricardo Martin-Brualla, Dan B Goldman, Matthias Nießner, and Justus Thies. Neural rgb-d surface reconstruction. In *CVPR*, pages 6290–6301, 2022. 7

[4] Jonathan T Barron, Ben Mildenhall, Dor Verbin, Pratul P Srinivasan, and Peter Hedman. Mip-nerf 360: Unbounded anti-aliased neural radiance fields. In *CVPR*, pages 5470–5479, 2022. 3

[5] Gilad Baruch, Zhuoyuan Chen, Afshin Dehghan, Tal Dimry, Yuri Feigin, Peter Fu, Thomas Gebauer, Brandon Joffe, Daniel Kurz, Arik Schwartz, et al. Arkitscenes: A diverse real-world dataset for 3d indoor scene understanding using mobile rgb-d data. *arXiv preprint arXiv:2111.08897*, 2021. 6, 17

[6] Daniel J Butler, Jonas Wulff, Garrett B Stanley, and Michael J Black. A naturalistic open source movie for optical flow evaluation. In *ECCV*, pages 611–625. Springer, 2012. 7, 8

[7] Yohann Cabon, Naila Murray, and Martin Humenberger. Virtual kitti 2. *arXiv preprint arXiv:2001.10773*, 2020. 6, 17

[8] Anpei Chen, Zexiang Xu, Andreas Geiger, Jingyi Yu, and Hao Su. Tensorf: Tensorial radiance fields. In *ECCV*, pages 333–350. Springer, 2022. 3

[9] Hongkai Chen, Zixin Luo, Jiahui Zhang, Lei Zhou, Xuyang Bai, Zeyu Hu, Chiew-Lan Tai, and Long Quan. Learning to match features with seeded graph matching network. In *ICCV*, pages 6301–6310, 2021. 3

[10] Christopher B Choy, Danfei Xu, JunYoung Gwak, Kevin Chen, and Silvio Savarese. 3d-r2n2: A unified approach for single and multi-view 3d object reconstruction. In *ECCV*, pages 628–644. Springer, 2016. 3

[11] David J Crandall, Andrew Owens, Noah Snavely, and Daniel P Huttenlocher. Sfm with mrfs: Discrete-continuous optimization for large-scale structure from motion. *TPAMI*, 35(12):2841–2853, 2012. 1, 3

[12] Angela Dai, Angel X Chang, Manolis Savva, Maciej Halber, Thomas Funkhouser, and Matthias Nießner. Scannet: Richly-annotated 3d reconstructions of indoor scenes. In *CVPR*, pages 5828–5839, 2017. 6, 8, 17

[13] Andrew J Davison, Ian D Reid, Nicholas D Molton, and Olivier Stasse. Monoslam: Real-time single camera slam. *TPAMI*, 29(6):1052–1067, 2007. 3

[14] Daniel DeTone, Tomasz Malisiewicz, and Andrew Rabinovich. Superpoint: Self-supervised interest point detection and description. In *CVPRW*, pages 224–236, 2018. 3

[15] 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 *ICLR*, 2021. 4, 6

[16] Mihai Dusmanu, Ignacio Rocco, Tomas Pajdla, Marc Pollefeys, Josef Sivic, Akihiko Torii, and Torsten Sattler. D2-net: A trainable cnn for joint description and detection of local features. In *CVPR*, pages 8092–8101, 2019. 3

[17] Jakob Engel, Thomas Schöps, and Daniel Cremers. Lsd-slam: Large-scale direct monocular slam. In *ECCV*, pages 834–849. Springer, 2014. 3

[18] Qiancheng Fu, Qingshan Xu, Yew Soon Ong, and Wenbing Tao. Geo-neus: Geometry-consistent neural implicit surfaces learning for multi-view reconstruction. *NeurIPS*, 35:3403–3416, 2022. 3

[19] Yasutaka Furukawa, Carlos Hernández, et al. Multi-view stereo: A tutorial. *Foundations and Trends® in Computer Graphics and Vision*, 9(1-2):1–148, 2015. 3

[20] Yasutaka Furukawa and Jean Ponce. Accurate, dense, and robust multiview stereopsis. *TPAMI*, 32(8):1362–1376, 2009. 1

[21] Andreas Geiger, Philip Lenz, Christoph Stiller, and Raquel Urtasun. Vision meets robotics: The kitti dataset. *The international journal of robotics research*, 32(11):1231–1237, 2013. 7

[22] Xiaodong Gu, Zhiwen Fan, Siyu Zhu, Zuozhuo Dai, Feitong Tan, and Ping Tan. Cascade cost volume for high-resolution multi-view stereo and stereo matching. In *CVPR*, pages 2495–2504, 2020. 1

[23] Byeongho Heo, Song Park, Dongyoon Han, and Sangdoo Yun. Rotary position embedding for vision transformer. In *ECCV*, pages 289–305. Springer, 2024. 5

[24] Po-Han Huang, Kevin Matzen, Johannes Kopf, Narendra Ahuja, and Jia-Bin Huang. Deepmvs: Learning multi-view stereopsis. In *CVPR*, 2018. 6, 17

[25] Yuanhui Huang, Wenzhao Zheng, Yunpeng Zhang, Jie Zhou, and Jiwen Lu. Tri-perspective view for vision-based 3d semantic occupancy prediction. In *CVPR*, pages 9223–9232, 2023. 1

[26] Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3d gaussian splatting for real-time radiance field rendering. *ACM Transactions on Graphics*, 42(4), July 2023. 3

[27] Georg Klein and David Murray. Parallel tracking and mapping for small ar workspaces. In *2007 6th IEEE and ACM international symposium on mixed and augmented reality*, pages 225–234. IEEE, 2007. 3

[28] Johannes Kopf, Xuejian Rong, and Jia-Bin Huang. Robust consistent video depth estimation. In *CVPR*, pages 1611–1621, 2021. 9

[29] Vincent Leroy, Yohann Cabon, and Jérôme Revaud. Grounding image matching in 3d with mast3r. *arXiv preprint arXiv:2406.09756*, 2024. 3, 6, 7, 8, 9

[30] Zhengqi Li and Noah Snavely. Megadepth: Learning single-view depth prediction from internet photos. In *CVPR*, pages 2041–2050, 2018. 6, 17

[31] Philipp Lindenberger, Paul-Edouard Sarlin, and Marc Pollefeys. Lightglue: Local feature matching at light speed. *arXiv preprint arXiv:2306.13643*, 2023. 3

[32] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017. 6

[33] David G Lowe. Object recognition from local scale-invariant features. In *ICCV*, volume 2, pages 1150–1157. Ieee, 1999. 1

[34] David G Lowe. Distinctive image features from scale-invariant keypoints. *IJCV*, 60:91–110, 2004. 3

[35] 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 *CVPR*, pages 4981–4991, 2023. 6, 17

[36] Ben Mildenhall, Pratul P Srinivasan, Matthew Tancik, Jonathan T Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. *Communications of the ACM*, 65(1):99–106, 2021. 3

[37] Thomas Müller, Alex Evans, Christoph Schied, and Alexander Keller. Instant neural graphics primitives with a multiresolution hash encoding. *ACM transactions on graphics (TOG)*, 41(4):1–15, 2022. 3

[38] Riku Murai, Eric Dexheimer, and Andrew J Davison. Mast3r-slam: Real-time dense slam with 3d reconstruction priors. In *CVPR*, pages 16695–16705, 2025. 10

[39] Richard A Newcombe, Shahram Izadi, Otmar Hilliges, David Molyneaux, David Kim, Andrew J Davison, Pushmeet Kohi, Jamie Shotton, Steve Hodges, and Andrew Fitzgibbon. Kinectfusion: Real-time dense surface mapping and tracking. In *2011 10th IEEE international symposium on mixed and augmented reality*, pages 127–136. Ieee, 2011. 3

[40] Richard A Newcombe, Steven J Lovegrove, and Andrew J Davison. Dtam: Dense tracking and mapping in real-time. In *ICCV*, pages 2320–2327. IEEE, 2011. 3

[41] John Oliensis. A critique of structure-from-motion algorithms. *Computer Vision and Image Understanding*, 80(2):172–214, 2000. 1

[42] Emanuele Palazzolo, Jens Behley, Philipp Lottes, Philippe Giguere, and Cyrill Stachniss. Refusion: 3d reconstruction in dynamic environments for rgb-d cameras exploiting residuals. In *IROS*, pages 7855–7862. IEEE, 2019. 7

[43] René Ranftl, Alexey Bochkovskiy, and Vladlen Koltun. Vision transformers for dense prediction. In *ICCV*, pages 12179–12188, 2021. 4, 5, 6

[44] Jeremy Reizenstein, Roman Shapovalov, Philipp Henzler, Luca Sbordone, Patrick Labatut, and David Novotny. Common objects in 3d: Large-scale learning and evaluation of real-life 3d category reconstruction. In *ICCV*, pages 10901–10911, 2021. 6, 17

[45] Mike Roberts, Jason Ramapuram, Anurag Ranjan, Atulit Kumar, Miguel Angel Bautista, Nathan Paczan, Russ Webb, and Joshua M. Susskind. Hypersim: A photorealistic synthetic dataset for holistic indoor scene understanding. In *ICCV*, 2021. 6, 17

[46] Ethan Rublee, Vincent Rabaud, Kurt Konolige, and Gary Bradski. Orb: An efficient alternative to sift or surf. In *ICCV*, pages 2564–2571. Ieee, 2011. 3

[47] Paul-Edouard Sarlin, Daniel DeTone, Tomasz Malisiewicz, and Andrew Rabinovich. Superglue: Learning feature matching with graph neural networks. In *CVPR*, pages 4938–4947, 2020. 3

[48] Johannes Lutz Schönberger and Jan-Michael Frahm. Structure-from-motion revisited. In *CVPR*, 2016. 1, 3

[49] Johannes Lutz Schönberger, Enliang Zheng, Marc Pollefeys, and Jan-Michael Frahm. Pixelwise view selection for unstructured multi-view stereo. In *ECCV*, 2016. 1, 3

[50] Yan Shi, Jun-Xiong Cai, Yoli Shavit, Tai-Jiang Mu, Wensen Feng, and Kai Zhang. Clustergnn: Cluster-based coarse-to-fine graph neural network for efficient feature matching. In *CVPR*, pages 12517–12526, 2022. 3

[51] 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 *CVPR*, pages 2930–2937, 2013. 7

[52] Nathan Silberman, Derek Hoiem, Pushmeet Kohli, and Rob Fergus. Indoor segmentation and support inference from rgbd images. In *ECCV*, pages 746–760. Springer, 2012. 7

[53] Jürgen Sturm, Nikolas Engelhard, Felix Endres, Wolfram Burgard, and Daniel Cremers. A benchmark for the evaluation of rgb-d slam systems. In *IROS*, pages 573–580. IEEE, 2012. 8

[54] Jianlin Su, Murtadha Ahmed, Yu Lu, Shengfeng Pan, Wen Bo, and Yunfeng Liu. Roformer: Enhanced transformer with rotary position embedding. *Neurocomputing*, 568:127063, 2024. 5

[55] Pei Sun, Henrik Kretzschmar, Xerxes Dotiwalla, Aurelien Chouard, Vijaysai Patnaik, Paul Tsui, James Guo, Yin Zhou, Yuning Chai, Benjamin Caine, et al. Scalability in perception for autonomous driving: Waymo open dataset. In *CVPR*, pages 2446–2454, 2020. 6, 17

[56] Chris Sweeney, Torsten Sattler, Tobias Hollerer, Matthew Turk, and Marc Pollefeys. Optimizing the viewing graph for structure-from-motion. In *ICCV*, pages 801–809, 2015. 1, 3

[57] Chengzhou Tang and Ping Tan. Ba-net: Dense bundle adjustment network. *arXiv preprint arXiv:1806.04807*, 2018. 3

[58] Bill Triggs, Philip F McLauchlan, Richard I Hartley, and Andrew W Fitzgibbon. Bundle adjustment—a modern synthesis. In *International workshop on vision algorithms*, pages 298–372. Springer, 1999. 3

[59] Hengyi Wang and Lourdes Agapito. 3d reconstruction with spatial memory. *arXiv preprint arXiv:2408.16061*, 2024. 1, 2, 3, 4, 7, 8, 9

[60] Jianyuan Wang, Minghao Chen, Nikita Karaev, Andrea Vedaldi, Christian Rupprecht, and David Novotny. Vggt: Visual geometry grounded transformer, 2025. 1, 2, 3

[61] Jianyuan Wang, Nikita Karaev, Christian Rupprecht, and David Novotny. Vggsfm: Visual geometry grounded deep structure from motion. In *CVPR*, pages 21686–21697, 2024. 3

[62] Peng Wang, Lingjie Liu, Yuan Liu, Christian Theobalt, Taku Komura, and Wenping Wang. Neus: Learning neural implicit surfaces by volume rendering for multi-view reconstruction. *arXiv preprint arXiv:2106.10689*, 2021. 3

[63] Qianqian Wang, Yifei Zhang, Aleksander Holynski, Alexei A. Efros, and Angjoo Kanazawa. Continuous 3d perception model with persistent state, 2025. 1, 2, 3, 4, 6, 7, 8, 9, 16, 18

[64] Shuzhe Wang, Vincent Leroy, Yohann Cabon, Boris Chidlovskii, and Jerome Revaud. Dust3r: Geometric 3d vision made easy. In *CVPR*, pages 20697–20709, 2024. 1, 3, 4, 6, 7, 8, 9

[65] Yi Wei, Shaohui Liu, Yongming Rao, Wang Zhao, Jiwen Lu, and Jie Zhou. Nerfingmvs: Guided optimization of neural radiance fields for indoor multi-view stereo. In *ICCV*, pages 5610–5619, October 2021. 3

[66] Philippe Weinzaepfel, Vincent Leroy, Thomas Lucas, Romain Brégier, Yohann Cabon, Vaibhav Arora, Leonid Antsfeld, Boris Chidlovskii, Gabriela Csurka, and Jérôme Revaud. Croco: Self-supervised pre-training for 3d vision tasks by cross-view completion. *NeurIPS*, 35:3502–3516, 2022. 4, 6

[67] Kyle Wilson and Noah Snavely. Robust global translations with 1dsfm. In *ECCV*, pages 61–75. Springer, 2014. 1, 3

[68] Changchang Wu. Towards linear-time incremental structure from motion. In *2013 International Conference on 3D Vision-3DV 2013*, pages 127–134. IEEE, 2013. 1, 3

[69] Tong Wu, Jiarui Zhang, Xiao Fu, Yuxin Wang, Jiawei Ren, Liang Pan, Wayne Wu, Lei Yang, Jiaqi Wang, Chen Qian, et al. Omniobject3d: Large-vocabulary 3d object dataset for realistic perception, reconstruction and generation. In *CVPR*, pages 803–814, 2023. 6, 17

[70] Hongchi Xia, Yang Fu, Sifei Liu, and Xiaolong Wang. Rgbd objects in the wild: scaling real-world 3d object learning from rgb-d videos. In *CVPR*, pages 22378–22389, 2024. 6, 17

[71] 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. *arXiv preprint arXiv:2501.13928*, 2025. 1, 2, 3

[72] Yao Yao, Zixin Luo, Shiwei Li, Tian Fang, and Long Quan. Mvsnet: Depth inference for unstructured multi-view stereo. In *ECCV*, pages 767–783, 2018. 3

[73] 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 *CVPR*, pages 1790–1799, 2020. 6, 17

[74] Chandan Yeshwanth, Yueh-Cheng Liu, Matthias Nießner, and Angela Dai. Scannet++: A high-fidelity dataset of 3d indoor scenes. In *ICCV*, pages 12–22, 2023. 6, 17

[75] Junyi Zhang, Charles Herrmann, Junhwa Hur, Varun Jampani, Trevor Darrell, Forrester Cole, Deqing Sun, and Ming-Hsuan Yang. Monst3r: A simple approach for estimating geometry in the presence of motion. *arXiv preprint arXiv:2410.03825*, 2024. 3, 7, 8, 9

[76] Xiaoshuai Zhang, Sai Bi, Kalyan Sunkavalli, Hao Su, and Zexiang Xu. Nerfusion: Fusing radiance fields for large-scale scene reconstruction. In *CVPR*, pages 5449–5458, 2022. 3

[77] Zhoutong Zhang, Forrester Cole, Zhengqi Li, Michael Rubinstein, Noah Snavely, and William T Freeman. Structure and motion from casual videos. In *ECCV*, pages 20–37. Springer, 2022. 9

[78] Wenzhao Zheng, Weiliang Chen, Yuanhui Huang, Borui Zhang, Yueqi Duan, and Jiwen Lu. Occworld: Learning a 3d occupancy world model for autonomous driving. In *ECCV*, pages 55–72. Springer, 2024. 1

[79] Yang Zheng, Adam W Harley, Bokui Shen, Gordon Wetzstein, and Leonidas J Guibas. Pointodyssey: A large-scale synthetic dataset for long-term point tracking. In *ICCV*, pages 19855–19865, 2023. 6, 17

A More Method Details

A.1 Memory Fusion Mechanism

We use a changing threshold δ to determine whether a new pointer and its nearest neighbor are sufficiently close. At time t, we have:

$$\delta = \sqrt{(\frac{\max P_{t-1}^x - \min P_{t-1}^x}{l_x})^2 + (\frac{\max P_{t-1}^y - \min P_{t-1}^y}{l_y})^2 + (\frac{\max P_{t-1}^z - \min P_{t-1}^z}{l_z})^2}, \tag{19}$$

where $P_{t-1}^x, P_{t-1}^y, P_{t-1}^z$ are the set of X, Y, Z components of the coordinates of all memory pointers from $\mathcal{M}_{t-1}$, l_x, l_y, l_z are constants we use to control the distribution of memory pointers. In practical implementation, we set l_x, l_y, l_z to 20.

A.2 3D Hierarchical Position Embedding

For n, m-th query and key $\mathbf{q}_n, \mathbf{k}_m \in \mathbb{R}^{1\times d_{head}}$, RoPE converts them to complex vector $\bar{\mathbf{q}}_n, \bar{\mathbf{k}}_m \in \mathbb{R}^{1\times(d_{head}/2)}$ by considering $(2t)$-th dim as real part and $(2t+1)$-th dim as imaginary part. We follow this so θ_t in Eq. 14 in the main text is

$$\theta_t = b^{-t/(d_{head}/6)}, \text{where } t \in \{0, 1, ..., d_{head}/6\}, b \in \{10, 100, 1000, 10000\}. \tag{20}$$

We use different frequency bases b to accommodate spatial inputs of varying scales, and thus derive four (h in Eq. 15 in the main text) different rotation matrices $\mathbf{R}_i$ $(i = 0, 1, 2, 3)$. Then we can obtain the embedded query and key, and the corresponding attention matrix as follows:

$$\bar{\mathbf{q}}' = \frac{1}{4}\sum_{i=1}^{4}(\bar{\mathbf{q}} \circ \mathbf{R}_i),\ \bar{\mathbf{k}}' = \frac{1}{4}\sum_{i=1}^{4}(\bar{\mathbf{k}} \circ \mathbf{R}_i),\ \mathbf{A}' = \mathrm{Re}[\bar{\mathbf{q}}'\bar{\mathbf{k}}'^{*}]. \tag{21}$$

The rotation matrix (with our 3D hierarchical position embedding) $\mathbf{A}'$ implies relative position in rotation form, and thus boosts the performance.

A.3 Pose Retrieval Mechanism

For each input sequence, we introduce a learnable token that is initialized within the model and serves as the pose token of the first frame. This token acts as the latent representation of the camera pose at the beginning of the sequence and provides a reference for subsequent frames. In addition to this initialization, we define a compact set of learnable tokens that function as a lightweight pose memory module [63]. This memory is specifically designed for pose retrieval, enabling the model to accumulate and reuse pose-related information across frames in a sequence without incurring excessive computational cost.

After processing each frame, the pose token that has interacted with the current image features is used to update this memory set, thereby maintaining a condensed yet informative representation of previously observed poses. When the next frame arrives, its corresponding pose token is initialized not randomly, but by retrieving an initial value from this pose memory. This retrieval process leverages the image tokens of the current frame to query the memory and select the most relevant pose context. In this way, the proposed memory mechanism allows the model to efficiently capture temporal dependencies between frames while avoiding redundancy and instability.

In the forward pass of Head_{global}, we first generate the pose-modulated tokens using an additional modulation function in CUT3R [63], and then feed them to the DPT architecture to generate the output $\hat{X}_t^{global}$. The modulation function uses two self-attention blocks and modulates the input tokens within the Layer Normalization layers using the pose token.

B More Training Details

Table 9 provides detailed information about the datasets used to train our model. To ensure both diversity and robustness, we adopt a multi-stage training strategy that progressively enhances the model's generalization capability across a wide range of real-world and synthetic environments.

Dataset	Scene Type	Dynamic	Real	Metric
ARKitScenes [5]	Indoor	Static	Real	Yes
BlendedMVS [73]	Mixed	Static	Synthetic	No
CO3Dv2 [44]	Object-Centric	Static	Real	No
HyperSim [45]	Indoor	Static	Synthetic	Yes
MegaDepth [30]	Outdoor	Static	Real	No
OmniObject3D [69]	Object-Centric	Static	Synthetic	Yes
ScanNet [12]	Indoor	Static	Real	Yes
ScanNet++ [74]	Indoor	Static	Real	Yes
WildRGBD [70]	Object-Centric	Static	Real	Yes
MVS-Synth [24]	Outdoor	Dynamic	Synthetic	Yes
PointOdyssey [79]	Mixed	Dynamic	Synthetic	Yes
Spring [35]	Mixed	Dynamic	Synthetic	Yes
VirtualKITTI2 [7]	Outdoor	Dynamic	Synthetic	Yes
Waymo [55]	Outdoor	Dynamic	Real	Yes

Table 9: **Training Datasets.**

In the first stage, we train our model on a collection of large-scale datasets, including ARKitScenes, BlendedMVS, CO3Dv2, HyperSim, MegaDepth, ScanNet, ScanNet++, WildRGBD, VirtualKITTI2, and Waymo. Such diversity allows the model to learn robust geometric priors and generalizable visual representations that are independent of specific dataset characteristics.

In the second and third stage, we introduce additional datasets and increase the input resolution. This progressive training paradigm enables the model to gradually adapt from coarse-level scene reasoning to high-resolution, detail-oriented prediction, effectively improving its performance in both accuracy and stability.

To further stabilize training in the early phase, we disable the memory fusion mechanism during the first stage. This prevents unstable memory updates that may arise when both pose and feature representations are still under training. Once the model achieves stable convergence, the memory fusion mechanism is introduced to facilitate temporal consistency learning across frames and improve the model's efficiency.

Moreover, we employ a dynamic dataset sampling strategy, where the sampling ratio of each dataset is adaptively adjusted throughout training. This adaptive reweighting ensures that the model receives balanced supervision from different datasets, mitigates potential domain bias, and leads to improved generalization across diverse scene distributions.

C More Experiments

Robustness to Poor Initialization Frames. The use of the initial frame as the reference leading to possible sensitivity to poor initialization frames is a common issue faced by the "-3R" series of work. We therefore evaluate the robustness of our method on such scenarios. We select Scene Kitchen in the NRGBD dataset and obtain 37 input images by sampling at the interval of 40 frames. To simulate a poor initialization, we set the 23rd sample (corresponding to frame 920 in the original dataset, which mainly contains large areas of the low-texture wall and floor with no other objects) as the first frame and randomly shuffled the order of the remaining frames. The comparison of results between our method and CUT3R under these two input settings is shown in Table 10.

We observe that our method is more robust to the effect of the initialization. Although the performance slightly degrades when the initial frame has significantly degraded content, the drop is acceptable considering the difficulty of such a scenario. We think this is because our explicit pointer memory is aligned with 3D structures of the current scene, which can minimize the impact of initialization as much as possible.

Table 10: **Robustness to poor initialization frames.**

Method	Normal Initialization						Poor Initialization					
	Acc↓		Comp↓		NC↑		Acc↓		Comp↓		NC↑	
	Mean	Med.	Mean	Med.	Mean	Med.	Mean	Med.	Mean	Med.	Mean	Med.
CUT3R [63]	0.166	0.040	0.081	0.016	0.980	0.976	0.290	0.154	0.156	0.051	0.733	0.942
Ours	0.096	0.037	0.058	0.014	0.977	0.977	0.120	0.070	0.068	0.019	0.853	0.972

D More Visualizations

We show qualitative results on sparse inputs in Figure 3 in the main text. In this section, we show more qualitative results on dense inputs from static (Figure 5) and dynamic (Figure 6) scenes.

Figure 5: **Qualitative results on dense inputs from static scenes.**

Figure 6: **Qualitative results on dense inputs from dynamic scenes.**

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: We claim a streaming dense 3d reconstruction framework with a explicit spatial pointer memory. These claims have been justified by the experimental results in Section 4.

   Guidelines:

   - The answer NA 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 NA 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: Limitations have been discussed in the Section 5.

   Guidelines:

   - The answer NA 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: [NA]

Justification: No theoretical results.

Guidelines:

- The answer NA 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 include details about our experiments in Section 3. We also include code with instructions to reproduce our results on public datasets in the supplementary materials.

   Guidelines:

   - The answer NA 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: [Yes]

Justification: Code and instructions to reproduce our results on public datasets are provided in the supplementary materials.

Guidelines:

- The answer NA means that paper does not include experiments requiring code.
- Please see the NeurIPS code and data submission guidelines (https://nips.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://nips.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, etc.) necessary to understand the results?

Answer: [Yes]

Justification: We have provided all details in Section 3, in Section 4 ,and in the code.

Guidelines:

- The answer NA 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 conventional procedure to report error bars in this field. However, the results do not fluctuate much through differen runs from our experiments.

Guidelines:

- The answer NA 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: All details have been provided in Section 3.

   Guidelines:

   - The answer NA 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 paper conforms with the NeurIPS Code of Ethics because authors have read and followed it.

   Guidelines:

   - The answer NA 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: Broader impact is discussed in Section 5.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA 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., pretrained language models, image generators, or scraped datasets)?

    Answer: [NA]

    Justification: Only existing datasets and models have been used in this work.

    Guidelines:

    - The answer NA 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: Authors of used code libraries, models and data have been cited and version details have been provided in our code package.

    Guidelines:

    - The answer NA 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: [Yes]

    Justification: Our code contains instructions how to run experiments and text comments.

    Guidelines:

    - The answer NA 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: [NA]

    Justification: Crowdsourcing has not been used.

    Guidelines:

    - The answer NA 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: [NA]

    Justification: Crowdsourcing has not been used.

    Guidelines:

    - The answer NA 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 rigorousness, or originality of the research, declaration is not required.

    Answer: [NA]

    Justification: Authors did not use LLMs for core method development or any components.

    Guidelines:

    - The answer NA 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 (https://neurips.cc/Conferences/2025/LLM) for what should or should not be described.