MITSUBISHI ELECTRIC RESEARCH LABORATORIES
https://www.merl.com

Point4Cast: Streaming Dynamic Scene Reconstruction and Forecasting

Liu, Xinhang; Miraldo, Pedro; Lohit, Suhas; Jiang, Huaizu; Sawada, Naoko; Tai, Yu-Wing; Tang, Chi-Keung; Chatterjee, Moitreya

TR2026-077 June 04, 2026

Abstract

Understanding how the 3D world evolves over time is a fundamental task in computer vision, essential for embodied settings, autonomous driving, etc. It requires not only the reconstruction of the observed scene but also the anticipation of how the scene dynamics will unfold in the future. While the area of 3D reconstruction has progressed rapidly with the advent of recent feed-forward neural networks, forecasting future dynamics in 3D, given the 2D frames of a video remains unexplored. We present Point4Cast, a unified framework that processes streaming 2D frame se- quences of a video to estimate the past, present, and future of the underlying dynamic scene, in 3D. At the core of our approach lies a persistently evolving latent space- time representation that models the environment's evolution across time. Upon receiving a new 2D frame, an up- date operation integrates the incoming evidence to refine the latent spacetime representation. When queried for any time instant, whether before, at, or beyond the timestamp of the last update, a readout procedure predicts temporally conditioned point maps and camera parameters describing the scene geometry at the queried time. Unlike prior approaches for online dynamic scene reconstruction that estimate each frame's point map solely at the timestamp of the last observed frame, Point4Cast achieves coherent reconstruction across any queried time. Empirical evaluations show that Point4Cast achieves state-of-the-art performance on streaming dynamic scene reconstruction and forecasting benchmarks, across multiple challenging datasets, while providing scene flow estimation and forecasting without the need for any additional inference or training.

IEEE Conference on Computer Vision and Pattern Recognition (CVPR) 2026

Mitsubishi Electric Research Laboratories, Inc.
201 Broadway, Cambridge, Massachusetts 02139

Point4Cast: Streaming Dynamic Scene Reconstruction and Forecasting

Xinhang Liu[1*†] Pedro Miraldo[2] Suhas Lohit[2]
Huaizu Jiang[3‡] Naoko Sawada[4] Yu-Wing Tai[5] Chi-Keung Tang[1†] Moitreya Chatterjee[2]
[1]HKUST [2]Mitsubishi Electric Research Laboratories (MERL) [3]Northeastern University
[4]Mitsubishi Electric [5]Dartmouth College
{xliufe, cktang}@connect.ust.hk, yu-wing.tai@dartmouth.edu, h.jiang@northeastern.edu,
sawada.naoko@df.mitsubishielectric.co.jp, {miraldo, slohit, chatterjee}@merl.com

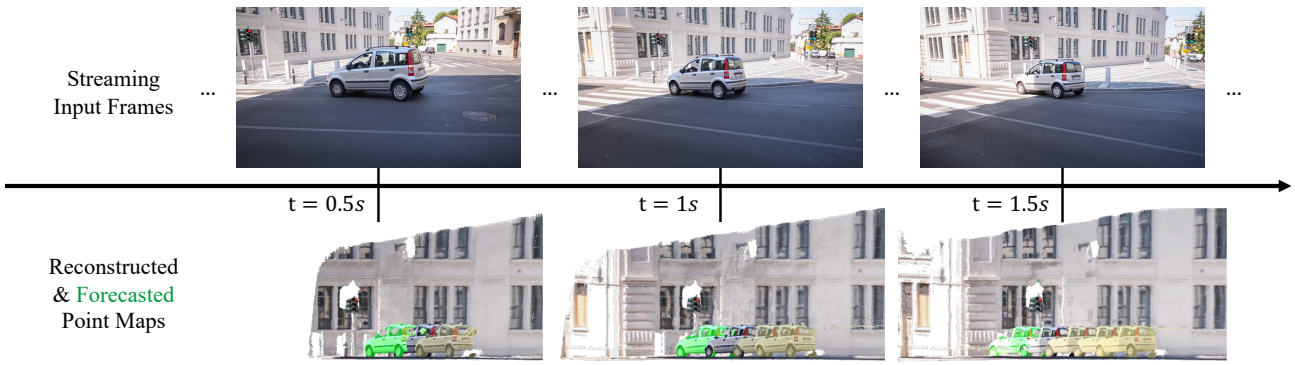

Figure 1. **Overview of *Point4Cast*.** Given a stream of input frames (top), our approach reconstructs and forecasts corresponding point maps over time (bottom). Overlaid point maps are shown for different queried time instants from past, **present**, and future.

Abstract

Understanding how the 3D world evolves over time is a fundamental task in computer vision, essential for embodied settings, autonomous driving, etc. It requires not only the reconstruction *of the observed scene but also the* anticipation *of how the scene dynamics will unfold in the future. While the area of 3D reconstruction has progressed rapidly with the advent of recent feed-forward neural networks, forecasting future dynamics in 3D, given the 2D frames of a video remains unexplored. We present* ***Point4Cast****, a unified framework that processes streaming 2D frame sequences of a video to estimate the past, present, and future of the underlying dynamic scene, in 3D. At the core of our approach lies a persistently evolving latent* spacetime representation *that models the environment's evolution across time. Upon receiving a new 2D frame, an* update *operation integrates the incoming evidence to refine the latent spacetime representation. When queried for any time instant, whether before, at, or beyond the timestamp of the last update, a* readout *procedure predicts temporally conditioned point maps and camera parameters describing the scene geometry at the queried time. Unlike prior approaches for online dynamic scene reconstruction that estimate each frame's point map solely at the timestamp of the last observed frame,* Point4Cast *achieves coherent reconstruction across any queried time. Empirical evaluations show that* Point4Cast *achieves state-of-the-art performance on streaming dynamic scene reconstruction and forecasting benchmarks, across multiple challenging datasets, while providing scene flow estimation and forecasting without the need for any additional inference or training. Project page: https://merl.com/research/highlights/point4cast.*

1. Introduction

The ability to model the evolution of the 3D world over time is a fundamental capability for several tasks in the realm of robotics, embodied intelligence, augmented/virtual reality (AR/VR), *etc*. Importantly, the ability to forecast the dynamics of this 3D world into the future is critical in order to avoid untoward outcomes. For instance, an agent (such as an autonomous vehicle) operating in a dynamic environment might not only need to *reconstruct* the scene's geometry and dynamics from a given set of image observations but might also need to *anticipate* how its environment will change in the near future and adapt accordingly (such as whether a pedestrian might come in the way of the vehicle).

Classical geometry-based methods for 3D reconstruction

*Work mainly done when XL was an intern at MERL.

such as Structure-from-Motion (SfM) [32, 63] and Simultaneous Localization and Mapping (SLAM) [15, 19, 58, 68] rebuild scenes independently for each video but often struggle in dynamic settings where both the camera and the objects in the scene move simultaneously. Neural approaches for 3D scene representations, including NeRFs [56] and Gaussian Splatting [37], have advanced the fidelity of reconstruction, yet they typically rely on known camera parameters and process entire scenes offline. Recent offline feed-forward methods for point map reconstruction [73, 75, 85], including those tailored for dynamic scenes [44, 46, 93] have enabled per-frame 3D reconstruction directly from images, leveraging data-driven priors for the task. Streaming variants to these feed-forward approaches [72, 74, 96] mark another significant advancement in the field. However, the aforementioned approaches remain limited to reconstructing instantaneous scene geometry, as point maps, without the capability to *forecast* how the scene evolves.

To bridge this gap, we introduce Point4Cast, a unified framework for streaming 3D scene reconstruction and the novel task of point map forecasting (see Fig. 1). At the core of Point4Cast, lies a persistently evolving *spacetime representation* that is trained to model the environment's structure and dynamics across the past, present, and anticipated future. As new frames arrive, an *update* operation integrates incoming observations into this latent representation, progressively constructing a consistent representation of the scene over time. When queried with an image and any time instant, Point4Cast performs a *readout* operation, yielding the scene geometry and camera parameters at the queried time. This design enables temporally coherent reconstruction across the observed time span and plausible forecasting of future scene geometry, unifying the tasks of 3D reconstruction and forecasting into a single framework. Moreover, Point4Cast's estimates of the reconstructed point maps over different time steps are aligned to the same coordinate system, allowing for establishing motion tracks of specific points without the need for any additional inference or training.

We evaluate Point4Cast on multiple, challenging benchmarks including PointOdyssey [94] and TAPVid-3D [38], demonstrating superior performance on reconstruction and the newly introduced 3D point map forecasting task, over both offline feed-forward and streaming baselines. The model generalizes across architectures and datasets, marking a prominent step toward continuous 3D perception for dynamic environments. Additionally, we obtain scene flow across the 3D point maps, over time steps, without any additional training or inference.

The main contributions of our work are as follows:

- We study the novel task of 3D point map *forecasting* from a sequence of streaming video frames, unlike existing approaches which exclusively focus on the task of 3D point map reconstruction from images.
- We introduce ***Point4Cast***, a unified architecture that integrates reconstruction and forecasting through a persistently evolving spacetime representation and temporally conditioned decoding.
- Furthermore, our approach provides scene flow estimates between the point maps over different time steps, without any additional training or inference.
- We achieve state-of-the-art performance on challenging, dynamic scene benchmarks, demonstrating coherent reconstruction while also achieving plausible forecasting, and scene flow estimates.

2. Related Work

Visual Forecasting. Within computer vision, significant progress has been made towards forecasting 2D image frames of a video. This task, formally called *video frame prediction*, entails forecasting the frames of a video given an initial set of frames. Video frame prediction has matured into two broad groups of approaches. (i) The first set of approaches takes a sequence prediction perspective to this task and uses network architectures designed to capture temporal dependencies, such as Recurrent Neural Networks (RNNs) [5–7, 65, 76, 77] or Transformers [27, 30, 33, 67, 90] as a key component of the prediction network. Some of these methods adopt a deterministic approach and generate one prediction for every input video [65, 76], while others explicitly model the stochasticity of the process and generate the output by sampling from a learned distribution [8, 16]. (ii) The second set of methods adopt a motion forecasting lens and seek to avoid synthesizing/forecasting the whole frame, thereby circumventing the need to generate the redundant/static regions of the frames. While our approach also takes a given set of 2D frames as input, different from video prediction approaches, we forecast the future time steps as 3D attributes rather than as 2D frames.

Also related to our work is the task of *scene-flow forecasting*, in which the goal is to forecast the trajectory of 3D point clouds [70, 88]. These approaches take a set of 3D points in a point cloud and extrapolate them over time. While promising, these approaches, usually start with an input point cloud which is often sparse, often corresponding only to distinct keypoints in the scene. On the other hand, our approach can estimate flows between a dense set of 3D point clouds or track a particular 3D point of choice, without any additional training or inference, starting with a set of 2D frames.

Per-scene 3D Reconstruction. Classical approaches for 3D scene reconstruction usually operate on a per-scene basis, starting from scratch for each new scene. These include approaches based on Structure from Motion (SfM) [32, 63] and Simultaneous Localization and Mapping (SLAM) [15, 19, 58, 68]. These methods, however, do not usually

deal with dynamic scenes which is the key focus of our work. *Neural Radiance Fields* (NeRF) [56] present an implicit approach for 3D scene representation by using multilayer perceptrons (MLPs) for novel view synthesis [1–3, 29, 48, 49, 71, 80, 82]. Subsequent works replace the deep MLPs in NeRFs with a feature voxel grid in order to improve training and inference speeds [9, 22, 57, 66]. Recently explicit scene representation methods, such as *3D Gaussian Splatting* (3DGS), have gained traction. These approaches use a large set of Gaussians to represent a 3D scene in order to drive gains primarily in the training and inference times [36, 79].

Extensions of the aforementioned approaches to dynamic scenes is a more recent topic of interest, including those based on SLAM [14, 78], NeRFs [13, 26, 34, 47, 50, 52, 62, 84, 91], or 3DGS [55, 79, 86, 87]. In the context of dynamic scenes, SLAM-based approaches often filter out the dynamic objects in the scene and use the static regions to establish correspondence across frames [95]. NeRF-based approaches either directly condition the radiance field on time [24, 41, 42, 83], learn a deformation field to map coordinates from different time stamps to a common canonical space [18, 20, 61, 92], or represent the scene using a space-time grid [4, 23, 50, 64]. Similarly, Gaussian Splatting-based approaches too either adopt a time-conditioning based approach [89], learn a motion-field on 3D Gaussians [43], or learn representations for 4D Gaussians [79]. Different from these techniques, ours is a feed-forward model that leverages data-driven priors learned across several scenes to enable dense 3D reconstruction directly from the frames of a video, without any knowledge of camera parameters.

Feed-forward 3D Reconstruction. Feed-forward 3D reconstruction models represent a paradigm shift in the area of 3D reconstruction. These approaches seek to harness data-driven priors learned across a wide-variety of scenes for 3D reconstruction without the knowledge of camera poses. The pioneering work of DUSt3R [75] takes two input images (two views of a scene) and predicts two point maps (per-pixel 3D point clouds) which are in the same coordinate frame. Extensions to DUSt3R, such as MASt3R [40] have sought to make these approaches more robust. However, these approaches are capable of handling only two input images at a time, which presents a serious limitation for dynamic scenes where a set of video frames need to be processed. FASt3R [85] extends this framework to the P-view setting ($P > 2$), capable of processing a set of variable number of input views, equipped with Transformers powered by Flash-Attention and parallel view fusion. VGGT [73] trains their model on an even larger set of data, including on dynamic scenes, achieving relatively encouraging results across a wide-variety of scenes. Despite their progress, these methods do not explicitly model scene motion which prevents them from effectively modeling complex dynamic scenes.

More recently, some approaches [11, 31, 51, 54, 93] have extended learned 3D reconstruction approaches to deal specifically with dynamic scenes, using cues such as monocular depth or optical flow, to drive a separation between the static and dynamic aspects of the scene, for improved modeling. Other approaches [21, 35, 44, 46] hinge on dense video correspondence for supervision in order to achieve robustness across a wide variety of dynamic scenes. However, these methods are not capable of forecasting how the 3D scene evolves in the future, in contrast to our method.

Streaming Feed-forward 3D Reconstruction. In several real-world application settings, such as driving, augmented/virtual reality (AR/VR), *etc*. approaches that can deal with a set of streaming frames are of crucial import. Recently, the community has focused its attention towards this end, with (persistent) memory-centric approaches, such as Cut3R, Point3R, Spann3R, StreamingVGGT [72, 74, 81, 96]. Some others have proposed causal-transformer designs [39, 45, 96] to better tackle streaming frames. In concurrent work, TTT3R [12] proposes to leverage test-time adaptation towards addressing this task. Unlike these approaches, our method can also predict 3D pointmaps of the scene for unseen future timesteps.

3. Proposed Approach

Our method operates on a continuous stream of images from a monocular video, each image denoted by $I_k \in \mathbb{R}^{H\times W\times 3}$; I_k represents a 2D perspective of the dynamic 3D scene at time step k. Assuming k frames are observed, given an arbitrary query frame $I_q; 1 \leq q \leq k$, and a query time t, our goal is to estimate the 3D point map $\mathbf{X}_q^{(t)} \in \mathbb{R}^{H\times W\times 3}$, corresponding to the pixels in frame I_q at the query time t, and the camera parameters $\mathbf{g}_q \in \mathbb{R}^9$ (parameterizing the intrinsics and extrinsics) corresponding to the frame I_q. The query frame I_q can correspond to any of the observed inputs, *i.e.*, $q \in \{1, \ldots, k\}$. The query time t is unconstrained and may represent a moment in the past, present, or future, *i.e.*, $1 \leq t \lesseqgtr q$.

3.1. *Point4Cast* Overview

In order to handle streaming inputs and capture the temporal evolution of the scene, our system maintains a latent, persistently evolving *spacetime representation* $\mathbf{w}_k \in \mathbb{R}^{N\times C}$, consisting of N learnable tokens with C-dimensional channels, where $0 \leq k \leq T$ denotes the number of observed frames. This latent representation captures the 3D scene's evolution across the past, present, and anticipated future time steps, having observed k streaming input images.

At the outset, before any image observations appear, we randomly initialize this latent spacetime representation to be $\mathbf{w}_0$. When the system receives the k-th input image I_k,

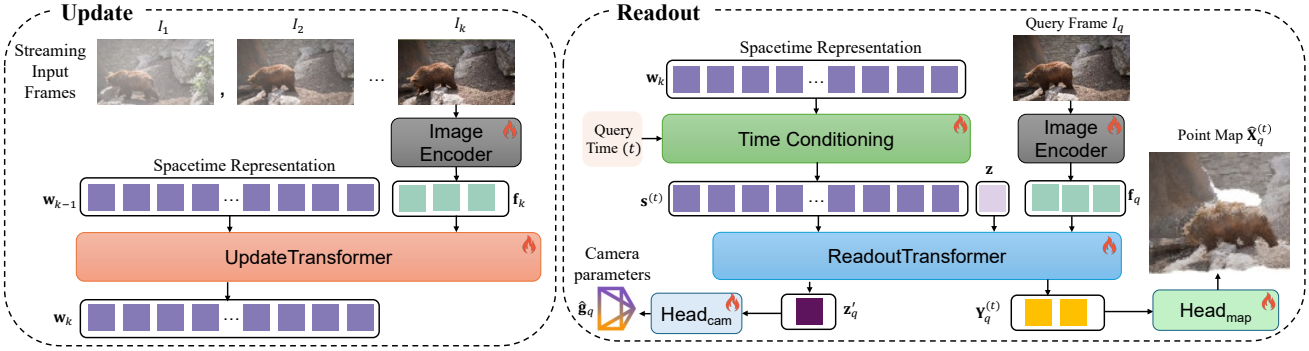

Figure 2. An overview of ***Point4Cast***, showing the details of the Update and Readout operations alongwith the trainable modules.

the representation is updated from its previous state $\mathbf{w}_{k-1}$ to $\mathbf{w}_k$ by integrating information from the newly observed frame. This can be represented as:

$$\mathbf{w}_k = \text{Update}(\mathbf{w}_{k-1}, I_k). \tag{1}$$

Given a latent spacetime representation, $\mathbf{w}_k$, based on observing the sequence of k images, one can specify an arbitrary frame $I_q, q \leq k$ and a time instant $t, 1 \leq t \leq T$, as a query to the model, in order to estimate the corresponding point map on the fly. This process can be formally represented as:

$$\hat{\mathbf{X}}_q^{(t)} = \text{Readout}(\mathbf{w}_k, I_q, t), \tag{2}$$

where $\hat{\mathbf{X}}_q^{(t)}$ denotes the predicted point map representing the geometry of the dynamic 3D scene at the query frame I_q and at the queried time t.

3.2. Spacetime Representation Update and Readout

To effectively integrate new observations and infer scene geometry over time, both the *Update* and *Readout* operations are implemented using transformers with interleaved self-attention and cross-attention layers, enabling bidirectional information exchange between the latent representation and image features.

Update. The update process incorporates each new observation into the latent representation, as represented in Equation 1. This operation is implemented as follows. First, an image encoder [17] extracts visual features from the incoming frame I_k:

$$\mathbf{f}_k = \text{Encoder}(I_k) \in \mathbb{R}^{M \times C}, \tag{3}$$

where $\mathbf{f}_k$ consists of M image tokens. These features are then fused with the previous state of the *spacetime representation* $\mathbf{w}_{k-1}$ using a transformer, as follows:

$$\mathbf{w}_k = \text{UpdateTransformer}(\mathbf{w}_{k-1}, \mathbf{f}_k) \in \mathbb{R}^{N \times C}. \tag{4}$$

The *UpdateTransformer* employs interleaved self-attention and cross-attention between $\mathbf{f}_k$ and $\mathbf{w}_{k-1}$, enabling bidirectional information exchange between the new visual evidence and the existing latent state. This iterative process refines the latent representation to reflect the updated understanding of the scene after each incoming frame, I_k.

Readout. The readout process generates the point map corresponding to a queried image $I_q, q \leq k$ and a time t, either from the past, present, or anticipated future, as shown in Equation 2. This operation proceeds as follows. First, the latent representation $\mathbf{w}_k$ is modulated by the query time through a time-conditioning operation:

$$\mathbf{s}^{(t)} = \text{TimeCondition}(\mathbf{w}_k, t) \in \mathbb{R}^{N \times C}. \tag{5}$$

The *TimeCondition* operation modulates the latent representation with a learned embedding of the query time t through conditional normalization:

$$e_t = \text{Embed}(t), \quad \gamma = W_\gamma e_t, \quad \beta = W_\beta e_t, \quad \gamma, \beta \in \mathbb{R}^C,$$

$$\mathbf{s}^{(t)}[i,:] = \gamma \odot \frac{\mathbf{w}_k[i,:] - \mu_i}{\sigma_i} + \beta, \qquad \forall i \in \{1, \ldots, N\} \tag{6}$$

where $Embed(\cdot)$ captures a learned $D-$dimensional embedding for the query time t, $W_\gamma, W_\beta \in \mathbb{R}^{C \times D}$, μ_i (batch mean), σ_i (batch standard deviation) $\in \mathbb{R}^C$. This FiLM-style modulation [60] enables the latent representation to adapt its internal state to the temporal context specified by the query time t, allowing it to represent the scene configuration either at observed or future time steps. Next, the image encoder extracts visual features from the query frame, I_q:

$$\mathbf{f}_q = \text{Encoder}(I_q) \in \mathbb{R}^{N \times C}. \tag{7}$$

A learnable *pose token* $\mathbf{z} \in \mathbb{R}^C$ is appended to the image tokens of I_q, and a transformer is used to fuse the appended tokens $\mathbf{f}_q$, $\mathbf{z}$, with the latent representation at the query time t, $\mathbf{s}^{(t)}$, as follows:

$$\mathbf{Y}_q^{(t)}, \mathbf{z}'_q = \text{ReadoutTransformer}(\mathbf{f}_q, \mathbf{z}, \mathbf{s}^{(t)}), \tag{8}$$

where $\mathbf{Y}_q^{(t)}$ denotes the temporally conditioned tokens to be used for 3D point map estimation, and $\mathbf{z}'_q$ aggregates image-level information from the query image, I_q to be used to

decode the camera parameters. A prediction head then generates the 3D point map corresponding to the query frame at the specified time:

$$\hat{\mathbf{X}}_q^{(t)} = \text{Head}_{\text{map}}(\mathbf{Y}_q^{(t)}), \tag{9}$$

representing the estimated 3D scene geometry corresponding to I_q at time t.

Camera Parameters. A *camera head* estimates the camera parameters (pose and intrinsics) from the aggregated token:

$$\hat{\mathbf{g}}_q = \text{Head}_{\text{cam}}(\mathbf{z}'_q). \tag{10}$$

Scene Flow. Since our approach predicts temporally conditioned 3D point maps $\hat{\mathbf{X}}_q^{(t)}$ across time, scene flow between consecutive frames can be directly derived from its outputs. Given two successive time instants t and $t+1$, the 3D motion of each point is computed as:

$$\mathbf{F}_q^{(t\to t+1)} = \hat{\mathbf{X}}_q^{(t+1)} - \hat{\mathbf{X}}_q^{(t)}, \tag{11}$$

representing the per-point displacement field between adjacent frames. This formulation yields dense, geometry-consistent motion estimation as a natural byproduct of the model's inference scheme, without requiring any explicit scene-flow head or additional supervision.

3.3. Online Training Paradigm

We train *Point4Cast* in an online streaming fashion that mirrors its inference behavior. Given a video sequence $V = \{I_k\}_{k=1}^T$, with a total of T frames, at each step, the incoming frame I_k updates the latent *spacetime representation* via the *Update* module, yielding an updated state $\mathbf{w}_k$. After each update, the model is queried at each frame $q \le k$ and all time instants t within the range $1 \le t \le T$, to predict 3D point maps: $\hat{\mathbf{X}}_q^{(t)} = \text{Readout}(\mathbf{w}_k, I_q, t)$ and camera parameters $\hat{\mathbf{g}}_q = \text{Head}_{\text{cam}}(\mathbf{z}'_q)$. The predictions are supervised using ground-truth point maps $\mathbf{X}_q^{(t)}$ and ground-truth camera parameters $\mathbf{g}_q$ with an ℓ_1 loss:

$$\mathcal{L}_q^{(t)} = \|\hat{\mathbf{X}}_q^{(t)} - \mathbf{X}_q^{(t)}\|_1 + \lambda_{\text{cam}}\|\hat{\mathbf{g}}_q - \mathbf{g}_q\|_1, \lambda_{\text{cam}} > 0.$$

This online training scheme encourages the model to develop a temporally coherent, continuously evolving scene representation that generalizes naturally to a set of streaming input frames. The overall training procedure, for each video in the training set, is summarized in Algorithm 1.

Algorithm 1 Training pseudocode for a single video in *Point4Cast*

Input: A video $V = \{I_k\}_{k=1}^T$, ground truth 3D point maps $\mathbf{X} = \{\mathbf{X}_k\}_{k=1}^T$, camera parameters $\mathbf{g} = \{\mathbf{g}_k\}_{k=1}^T$
Output: Loss for the video sample $\mathcal{L}$.

Initialize latent state $\mathbf{w}_k \leftarrow \mathbf{w}_0$
$\mathcal{L} \leftarrow 0, \quad n \leftarrow 0$
for $k = 1$ **to** T **do** ▷ streaming frames
  $\mathbf{w}_k \leftarrow \text{Update}(\mathbf{w}_{k-1}, I_k)$
  for $q = 1$ **to** k **do** ▷ query images
    for $t = 1$ **to** T **do** ▷ query times
      $\mathbf{Y}_q^{(t)}, \mathbf{z}'_q \leftarrow \text{Readout}(\mathbf{w}_k, I_q, t)$
      $\hat{\mathbf{X}}_q^{(t)} \leftarrow \text{Head}_{\text{map}}(\mathbf{Y}_q^{(t)})$
      $\hat{\mathbf{g}}_q \leftarrow \text{Head}_{\text{cam}}(\mathbf{z}'_q)$
      $\mathcal{L}_q^{(t)} \leftarrow \|\hat{\mathbf{X}}_q^{(t)} - \mathbf{X}_q^{(t)}\|_1 + \lambda_{\text{cam}}\|\hat{\mathbf{g}}_q - \mathbf{g}_q\|_1$
      $\mathcal{L} \leftarrow \mathcal{L} + \mathcal{L}_q^{(t)}; \quad n \leftarrow n+1$
    end for
  end for
end for
$\mathcal{L} \leftarrow \mathcal{L}/n$ ▷ average over all frames, queries, and times

4. Experiments

We evaluate *Point4Cast* on 3D point map reconstruction, camera pose estimation, 3D point map forecasting, and scene-flow estimation tasks on synthetic and real-world benchmarks, and compare against recent open-source, state-of-the-art baselines. Ablations, additional details, qualitative results, and a code stub are provided in the supplementary material.

Implementation Details. We initialize *Point4Cast* using strong pretrained components rather than training all modules from scratch. The image encoder is initialized from the ViT backbone of VGGT [73], and the attention blocks of both the *UpdateTransformer* and *ReadoutTransformer* are initialized from VGGT's Dense Prediction Transformer (DPT) modules. The prediction heads, $Head_{map}$ and $Head_{cam}$, are similarly initialized from VGGT's camera head. The overall design remains modular, enabling alternative streaming backbones e.g., continuous 3D perception models [74], which can be substituted without architectural changes. $N = 4096$, while $C = 1024$. Both the *Update* and *Readout* modules use transformer decoders of moderate depth, while the map and camera heads use lighter-weight decoders. Our implementation is based on PyTorch [59] and trained with AdamW [53] using a fixed learning-rate schedule and mini-batches distributed across eight NVIDIA A100 (80GB) GPUs.

Training Datasets. To equip *Point4Cast* with strong priors for 3D reconstruction and forecasting, we train on a diverse mixture of synthetic and real-world dynamic-scene datasets. Starting from VGGT pre-trained weights [73], we finetune all trainable modules of *Point4Cast* on a curated set consisting of *Kubric* [28], *PointOdyssey* [94], *Stereo4D* [35], and an additional synthetic dataset that we create by rendering in Blender using Mixamo[1] motion assets and BlenderKit[2] scene elements. For datasets lacking ground-truth 3D point clouds, we generate pseudo-depth supervision via off-the-

[1] https://www.mixamo.com/
[2] https://www.blenderkit.com/

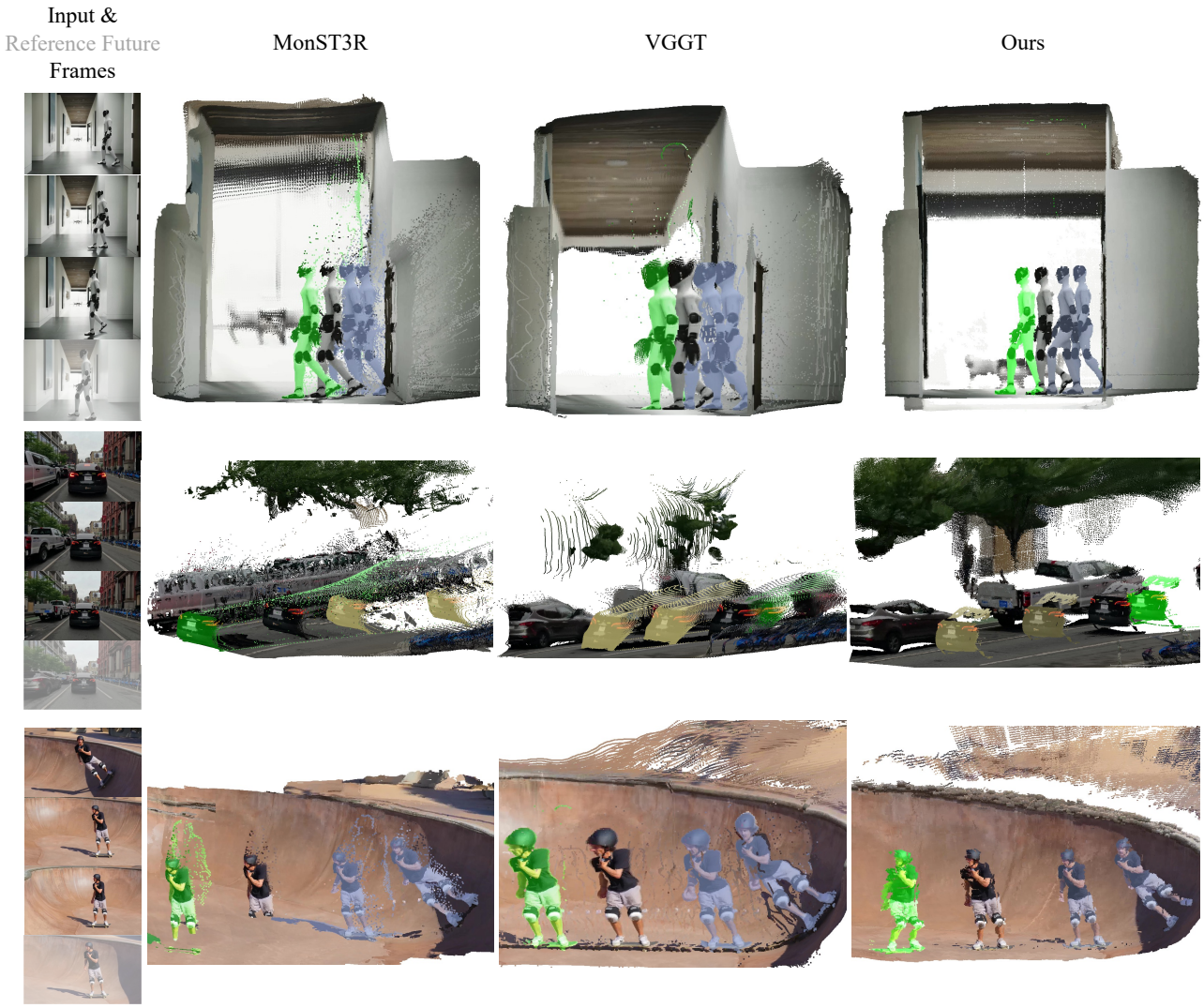

Figure 3. **Qualitative comparison of dynamic-scene reconstruction (shown in yellow/blue) and forecasting (shown in green).** We compare *Point4Cast* with MonST3R and VGGT on challenging human-type and driving scenes. *Point4Cast* produces more complete and temporally consistent 3D point maps, with sharper geometry, fewer artifacts, and more reasonable future predictions.

shelf monocular video depth estimators [10].

Despite being substantially smaller than the large-scale corpora used by recent feed-forward 3D reconstruction models (e.g., VGGT), this mixture provides sufficient variability for strong generalization to unseen dynamic scenes. We follow a simple curriculum: training first on controlled, synthetic human and object motion data (Kubric, PointOdyssey, Blender-rendered data), and then introducing more complex, real-world scenarios (Stereo4D). Full dataset, training details, and statistics are provided in the supplementary.

4.1. Experimental Setup

Datasets. For both reconstruction and forecasting, we evaluate *Point4Cast* on two challenging benchmarks: PointOdyssey [94], which consists of synthetic dynamic scenes, and TapVid-3D [38], a real-world counterpart focusing on long-term point tracking in cluttered scenes. Unless otherwise stated, we follow the official data splits and protocols for each benchmark. Importantly, TapVid-3D is not included in the training set of *Point4Cast*, so results on this dataset correspond to a *zero-shot evaluation* setting. For reconstruction, we estimate per-frame 3D point maps and camera parameters over all evaluation time steps of each video. For forecasting, we consider two horizons: one-step-ahead prediction ("next frame") and multi-step prediction ("next 10 frames"), where metrics are computed on the predicted 3D point maps and averaged across time steps.

Evaluation Metrics. Following prior work on feed-forward 3D reconstruction [73, 74], we report (i) *Accuracy* (Acc.) and (ii) *Completion* (Comp.), the two standard components of the symmetric Chamfer Distance. Accuracy measures how close each predicted point is to the nearest ground-truth point, while Completion measures how close each ground

truth point is to the predicted points. For camera pose estimation, we report (i) Relative Translation Error (RTE) and (ii) Relative Rotation Error (RRE), after performing Sim(3) alignment with ground truth [74]. For scene-flow estimation and forecasting, we follow Liang *et al*. [46], and report (i) End-Point Error (EPE) and (ii) Accuracy (Acc.), computed between consecutive 3D point maps.

Baselines. We compare *Point4Cast* with the latest open-source, state-of-the-art methods for 3D point map reconstruction. Among offline approaches, we include MonST3R [93], designed for dynamic scenes, and VGGT [73], a large-scale feed-forward baseline. Among streaming approaches, we evaluate our approach against CUT3R [74] and StreamingVGGT [96], which represent the state-of-the-art in online 3D reconstruction.

For the task of 3D point map forecasting from 2D frames, we build two forecasting variants of each baseline. In the first variant, we forecast future RGB frames using a recent open-source video generation model [25] and then reconstruct each predicted frame using the baseline model. We denote this setting as *Frame generation*. In the second variant, we reconstruct the first forecasted frame in 3D using the baseline, and obtain subsequent predictions by propagating the 3D point map forward using a scene-flow continuation mechanism computed from the latest two predicted point maps. We dub this setting as *Scene-flow cont.*. In contrast, *Point4Cast* performs forecasting intrinsically through its time-conditioned readout mechanism, requiring no external video generator, scene-flow propagation module, or additional supervision.

4.2. Results

Reconstruction. Tables 1 and 2 show that *Point4Cast* achieves consistent performance improvement over both offline (MonST3R, VGGT) and online (CUT3R, StreamingVGGT) baselines on PointOdyssey and TapVid-3D. With either backbone, *Point4Cast* yields better point-map quality and lower camera pose estimation errors, on both PointOdyssey and TAPVid-3D datasets. The gains are on TapVid-3D, which presents a fully zero-shot setting, highlight the robustness of our proposed approach.

Forecasting. As reported in Tables 3 and 4, *Point4Cast* generally outperforms all forecasting variants of MonST3R, VGGT, CUT3R, and StreamingVGGT for both next-frame and 10-step prediction. Unlike frame-generation or scene-flow–continuation pipelines, whose errors compound rapidly over time, *Point4Cast* maintains substantially more stable forecasts due to its unified spacetime representation.

Figure 4 further illustrates this trend on PointOdyssey: both accuracy and completeness degrade smoothly as the prediction horizon increases, reflecting the inherent difficulty of long-range 3D forecasting, while highlighting the relative stability of our approach across future time steps.

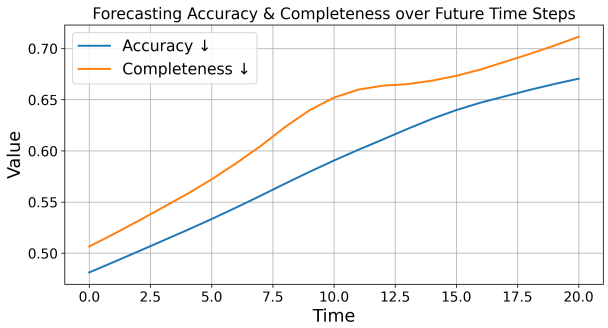

Figure 4. **Forecasting performance over future time steps on the PointOdyssey dataset.** Accuracy and completeness (lower the better) gradually decline as the prediction horizon increases.

Figure 5. **3D point tracks over past, present, and future time steps.** *Point4Cast* yields 3D point tracks that are semantically consistent across different time steps, as shown by the colored tracks.

Qualitatively too (see Figure 3), we see that our proposed approach exhibits denser and more coherent 3D point maps with smoother motion, across both the reconstructed (yellow/blue) and forecasted (green) point maps, compared to competing methods, such as MonST3R [93] or VGGT [73]. Importantly, our proposed approach is able to forecast motions across both rigid (such as cars) and non-rigid (such as humans) objects, attesting to its effectiveness.

Scene Flow. Table 5 shows that *Point4Cast* also achieves the strongest scene-flow estimation and forecasting, despite not being trained with any flow supervision. As shown in the qualitative visualization in Figure 5, we notice that *Point4Cast*is able to accurately track semantically consistent locations in the scene over past, present, and future time steps. Accurate flows emerge directly from the temporally conditioned point maps, indicating that a coherent latent spacetime representation implicitly models 3D motion.

Runtime. *Point4Cast* has inference times comparable to Cut3R of $\sim$ 20fps on both datasets.

4.3. Ablation Study

We explore different time-conditioning strategies: (i) a sinusoidal $C-$dimensional Positional Encoding (PE) of time, $e_t = [\sin(\omega_1 t), \cos(\omega_1 t), \ldots, \sin(\omega_{C/2} t), \cos(\omega_{C/2} t)] \in \mathbb{R}^C$; (ii) a learned embedding which directly maps the scalar time to a learned $C-$dimensional embedding vec-

Table 1. **Reconstruction results on the PointOdyssey dataset.** The **best** and second best results are highlighted.

Method	Online	Backbone	Acc.↓	Comp.↓	Acc-d.↓	Comp-d.↓	RTE↓	RRE↓
MonST3R [93]	×	DUSt3R	0.481	0.502	0.536	0.561	0.021	0.482
VGGT [73]	×	VGGT	0.464	0.491	0.517	0.534	**0.016**	0.441
CUT3R [74]	✓	CUT3R	0.530	0.557	0.572	0.594	0.026	0.480
StreamVGGT [96]	✓	VGGT	0.525	0.569	0.570	0.612	0.031	0.571
Ours	✓	CUT3R	**0.410**	0.484	0.458	0.516	0.020	0.479
Ours	✓	VGGT	0.428	**0.472**	**0.474**	**0.488**	**0.016**	**0.437**

Table 2. **Reconstruction results on the TAPVid-3D dataset.** The **best** and second best results are highlighted.

Method	Online	Backbone	Acc.↓	Comp.↓	Acc-d.↓	Comp-d.↓	RTE↓	RRE↓
MonST3R [93]	×	DUSt3R	0.775	0.502	0.830	0.575	0.032	0.541
VGGT [73]	×	VGGT	0.757	0.491	0.897	0.551	0.029	0.511
CUT3R [74]	✓	CUT3R	0.869	0.657	0.972	0.581	0.045	0.575
StreamVGGT [96]	✓	VGGT	0.817	0.569	0.910	0.612	0.039	0.598
Ours	✓	CUT3R	0.768	0.540	0.818	0.537	0.037	0.530
Ours	✓	VGGT	**0.711**	**0.476**	**0.784**	**0.513**	**0.028**	**0.508**

Table 3. **Forecasting results on the PointOdyssey dataset.** The **best** and second best results are highlighted.

Method	Backbone	Forecasting Mechanism	Next frame		Next 10 frames	
			Acc. ↓	Comp. ↓	Acc. ↓	Comp. ↓
MonST3R [93]	DUSt3R	Frame generation	0.509	0.569	0.617	0.732
MonST3R [93]	DUSt3R	Scene-flow cont.	0.566	0.603	0.781	0.870
VGGT [73]	VGGT	Frame generation	0.561	0.595	0.871	0.955
VGGT [73]	VGGT	Scene-flow cont.	0.678	0.746	0.785	0.892
CUT3R [74]	CUT3R	Frame generation	0.716	0.794	0.945	1.039
CUT3R [74]	CUT3R	Scene-flow cont.	0.669	0.749	0.716	1.011
StreamVGGT [96]	VGGT	Frame generation	0.558	0.603	0.603	0.671
StreamVGGT [96]	VGGT	Scene-flow cont.	0.580	0.694	0.776	0.817
Ours	CUT3R	Inherent	0.498	0.542	0.561	0.604
Ours	VGGT	Inherent	**0.481**	**0.506**	**0.533**	**0.571**

tor, $e_t = \text{Embed}(t) \in \mathbb{R}^C$, followed by a cross attention scheme:

$$\hat{s} = \text{MHA}(Q = \mathbf{w}_k, K = e_t, V = e_t), \quad (12)$$

$$\mathbf{s}^{(t)} = FFN(LN(\hat{s})) + \hat{s}, \quad (13)$$

where MHA denotes Multi-head Attention, Q, K, V denote the query, key, and value of the attention module, FFN denotes a feed-forward network while LN denotes Layer Norm, as is common in transformer modules [69]. Table 6 summarizes ablations on the time-encoding and conditioning mechanisms. Learned time embeddings outperform sinusoidal ones, while our FiLM [60]-style conditioning (Eq. 6) provides the largest improvements. These results confirm that flexible temporal conditioning is crucial for accurate reconstruction and forecasting.

Moreover, we also assess the sensitivity of our proposed model, *Point4Cast*, to the choice of different backbones. As we see from the model performances in Tables 3 and 4, our proposed approach remains relatively robust across different backbone choices (viz. Cut3R [74] or VGGT [73]). Additional ablations are provided in the supplementary.

Table 4. **Forecasting results on the TAPVid-3D dataset.** The **best** and second best results are highlighted.

Method	Backbone	Forecasting Mechanism	Next frame		Next 10 frames	
			Acc. ↓	Comp. ↓	Acc. ↓	Comp. ↓
MonST3R [93]	DUSt3R	Frame generation	0.912	0.969	1.271	**1.326**
MonST3R [93]	DUSt3R	Scene-flow cont.	0.936	1.058	1.516	1.630
VGGT [73]	VGGT	Frame generation	0.881	0.959	1.382	1.428
VGGT [73]	VGGT	Scene-flow cont.	0.977	1.189	1.541	1.696
CUT3R [74]	CUT3R	Frame generation	0.948	1.010	1.481	1.539
CUT3R [74]	CUT3R	Scene-flow cont.	0.972	1.328	1.678	1.911
StreamVGGT [96]	VGGT	Frame generation	0.956	1.259	1.693	1.671
StreamVGGT [96]	VGGT	Scene-flow cont.	0.921	1.310	1.896	1.817
Ours	CUT3R	Inherent	0.831	0.890	1.371	1.471
Ours	VGGT	Inherent	**0.810**	**0.878**	**1.259**	1.456

Table 5. **Scene flow estimation and forecasting results on the PointOdyssey dataset.** The **best** and second best results are highlighted.

Method	Scene Flow Estimation		Scene Flow Forecasting	
	EPE ↓	Acc ↑	EPE ↓	Acc ↑
MonST3R [93]	2.058	0.741	3.158	0.725
VGGT [73]	3.170	0.707	3.601	0.661
CUT3R [74]	3.838	0.661	4.101	0.623
StreamVGGT [96]	2.441	0.718	3.717	0.537
Ours (VGGT backbone)	**1.355**	**0.848**	**1.619**	**0.766**

Table 6. **Ablation study on the choice of Time Conditioning technique on the PointOdyssey dataset.** The **best** results are highlighted.

Query Time Embedding	Conditioning	Acc. ↓	Comp. ↓
Sinusoidal	Cross-Attention	0.470	0.502
Learned	Cross-Attention	0.437	0.492
Learned	FiLM	**0.428**	**0.472**

5. Conclusions and Future Work

In this work, we introduce the novel task of 3D point map forecasting, given a sequence of streaming frames of a video. Towards this end, we propose Point4Cast, a unified architecture that integrates reconstruction and forecasting through a persistently evolving latent *spacetime* representation and temporally conditioned decoding. The flexible design of our approach permits the use of multiple backbone networks and different temporal conditioning strategies. Moreover, our approach provides scene flow estimates between the point maps over different time steps, without any additional training or inference. Empirical evaluations across challenging benchmarks, show that our proposed approach can outperform latest open-source, state-of-the-art methods at both 3D point map reconstruction and camera parameter estimation as well as for the novel task of 3D point map forecasting. Going forward, we intend to incorporate uncertainty modeling into our prediction framework.

References

[1] Jonathan T Barron, Ben Mildenhall, Matthew Tancik, Peter Hedman, Ricardo Martin-Brualla, and Pratul P Srinivasan. Mip-nerf: A multiscale representation for anti-aliasing neural radiance fields. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 5855–5864, 2021. 3

[2] Jonathan T Barron, Ben Mildenhall, Dor Verbin, Pratul P Srinivasan, and Peter Hedman. Mip-nerf 360: Unbounded anti-aliased neural radiance fields. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5470–5479, 2022.

[3] Jonathan T Barron, Ben Mildenhall, Dor Verbin, Pratul P Srinivasan, and Peter Hedman. Zip-nerf: Anti-aliased grid-based neural radiance fields. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 19697–19705, 2023. 3

[4] Ang Cao and Justin Johnson. Hexplane: A fast representation for dynamic scenes. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 130–141, 2023. 3

[5] Lluis Castrejon, Nicolas Ballas, and Aaron Courville. Improved conditional vrnns for video prediction. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 7608–7617, 2019. 2

[6] Zheng Chang, Xinfeng Zhang, Shanshe Wang, Siwei Ma, and Wen Gao. Stip: A spatiotemporal information-preserving and perception-augmented model for high-resolution video prediction. *arXiv preprint arXiv:2206.04381*, 2022.

[7] Zheng Chang, Xinfeng Zhang, Shanshe Wang, Siwei Ma, and Wen Gao. Strpm: A spatiotemporal residual predictive model for high-resolution video prediction. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 13946–13955, 2022. 2

[8] Moitreya Chatterjee, Narendra Ahuja, and Anoop Cherian. A hierarchical variational neural uncertainty model for stochastic video prediction. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 9751–9761, 2021. 2

[9] Anpei Chen, Zexiang Xu, Andreas Geiger, Jingyi Yu, and Hao Su. Tensorf: Tensorial radiance fields. In *European Conference on Computer Vision (ECCV)*, pages 333–350. Springer, 2022. 3

[10] Sili Chen, Hengkai Guo, Shengnan Zhu, Feihu Zhang, Zilong Huang, Jiashi Feng, and Bingyi Kang. Video depth anything: Consistent depth estimation for super-long videos. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 22831–22840, 2025. 6

[11] Xingyu Chen, Yue Chen, Yuliang Xiu, Andreas Geiger, and Anpei Chen. Easi3r: Estimating disentangled motion from dust3r without training. *arXiv preprint arXiv:2503.24391*, 2025. 3

[12] Xingyu Chen, Yue Chen, Yuliang Xiu, Andreas Geiger, and Anpei Chen. Ttt3r: 3d reconstruction as test-time training. *arXiv preprint arXiv:2509.26645*, 2025. 3

[13] Yue Chen, Xuan Wang, Xingyu Chen, Qi Zhang, Xiaoyu Li, Yu Guo, Jue Wang, and Fei Wang. Uv volumes for real-time rendering of editable free-view human performance. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 16621–16631, 2023. 3

[14] Shuhong Cheng, Changhe Sun, Shijun Zhang, and Dianfan Zhang. Sg-slam: A real-time rgb-d visual slam toward dynamic scenes with semantic and geometric information. *IEEE Transactions on Instrumentation and Measurement*, 72:1–12, 2022. 3

[15] Andrew J Davison, Ian D Reid, Nicholas D Molton, and Olivier Stasse. Monoslam: Real-time single camera slam. *IEEE transactions on pattern analysis and machine intelligence*, 29(6):1052–1067, 2007. 2

[16] Emily Denton and Rob Fergus. Stochastic video generation with a learned prior. In *International conference on machine learning*, pages 1174–1183. PMLR, 2018. 2

[17] Alexey Dosovitskiy. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020. 4

[18] Yilun Du, Yinan Zhang, Hong-Xing Yu, Joshua B Tenenbaum, and Jiajun Wu. Neural radiance flow for 4d view synthesis and video processing. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 14304–14314. IEEE Computer Society, 2021. 3

[19] Jakob Engel, Thomas Schöps, and Daniel Cremers. Lsd-slam: Large-scale direct monocular slam. In *European conference on computer vision*, pages 834–849. Springer, 2014. 2

[20] Jiemin Fang, Taoran Yi, Xinggang Wang, Lingxi Xie, Xiaopeng Zhang, Wenyu Liu, Matthias Nießner, and Qi Tian. Fast dynamic radiance fields with time-aware neural voxels. In *ACM Transactions on Graphics (SIGGRAPH Asia)*, pages 1–9, 2022. 3

[21] Haiwen Feng, Junyi Zhang, Qianqian Wang, Yufei Ye, Pengcheng Yu, Michael J Black, Trevor Darrell, and Angjoo Kanazawa. St4rtrack: Simultaneous 4d reconstruction and tracking in the world. *arXiv preprint arXiv:2504.13152*, 2025. 3

[22] Sara Fridovich-Keil, Alex Yu, Matthew Tancik, Qinhong Chen, Benjamin Recht, and Angjoo Kanazawa. Plenoxels: Radiance fields without neural networks. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5501–5510, 2022. 3

[23] Sara Fridovich-Keil, Giacomo Meanti, Frederik Rahbæk Warburg, Benjamin Recht, and Angjoo Kanazawa. K-planes: Explicit radiance fields in space, time, and appearance. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 12479–12488, 2023. 3

[24] Chen Gao, Ayush Saraf, Johannes Kopf, and Jia-Bin Huang. Dynamic view synthesis from dynamic monocular video. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 5712–5721, 2021. 3

[25] Yu Gao, Haoyuan Guo, Tuyen Hoang, Weilin Huang, Lu Jiang, Fangyuan Kong, Huixia Li, Jiashi Li, Liang Li, Xiaojie Li, et al. Seedance 1.0: Exploring the boundaries of video generation models. *arXiv preprint arXiv:2506.09113*, 2025. 7

[26] Chen Geng, Sida Peng, Zhen Xu, Hujun Bao, and Xiaowei Zhou. Learning neural volumetric representations of dynamic humans in minutes. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 8759–8770, 2023. 3

[27] Rohit Girdhar and Kristen Grauman. Anticipative video transformer. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 13505–13515, 2021. 2

[28] Klaus Greff, Francois Belletti, Lucas Beyer, Carl Doersch, Yilun Du, Daniel Duckworth, David J Fleet, Dan Gnanapragasam, Florian Golemo, Charles Herrmann, Thomas Kipf, Abhijit Kundu, Dmitry Lagun, Issam Laradji, Hsueh-Ti (Derek) Liu, Henning Meyer, Yishu Miao, Derek Nowrouzezahrai, Cengiz Oztireli, Etienne Pot, Noha Radwan, Daniel Rebain, Sara Sabour, Mehdi S. M. Sajjadi, Matan Sela, Vincent Sitzmann, Austin Stone, Deqing Sun, Suhani Vora, Ziyu Wang, Tianhao Wu, Kwang Moo Yi, Fangcheng Zhong, and Andrea Tagliasacchi. Kubric: a scalable dataset generator. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022. 5

[29] Yuan-Chen Guo, Di Kang, Linchao Bao, Yu He, and Song-Hai Zhang. Nerfren: Neural radiance fields with reflections. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 18409–18418, 2022. 3

[30] Agrim Gupta, Stephen Tian, Yunzhi Zhang, Jiajun Wu, Roberto Martín-Martín, and Li Fei-Fei. Maskvit: Masked visual pre-training for video prediction. *arXiv preprint arXiv:2206.11894*, 2022. 2

[31] Jisang Han, Honggyu An, Jaewoo Jung, Takuya Narihira, Junyoung Seo, Kazumi Fukuda, Chaehyun Kim, Sunghwan Hong, Yuki Mitsufuji, and Seungryong Kim. Dˆ 2ust3r: Enhancing 3d reconstruction with 4d pointmaps for dynamic scenes. *arXiv preprint arXiv:2504.06264*, 2025. 3

[32] Richard Hartley. *Multiple view geometry in computer vision*. Cambridge university press, 2003. 2

[33] Tobias Höppe, Arash Mehrjou, Stefan Bauer, Didrik Nielsen, and Andrea Dittadi. Diffusion models for video prediction and infilling. *Transactions on Machine Learning Research*, 2022, 2022. 2

[34] Mustafa Işık, Martin Rünz, Markos Georgopoulos, Taras Khakhulin, Jonathan Starck, Lourdes Agapito, and Matthias Nießner. Humanrf: High-fidelity neural radiance fields for humans in motion. *ACM Transactions on Graphics (SIGGRAPH)*, 2023. 3

[35] Linyi Jin, Richard Tucker, Zhengqi Li, David Fouhey, Noah Snavely, and Aleksander Holynski. Stereo4d: Learning how things move in 3d from internet stereo videos. *arXiv preprint arXiv:2412.09621*, 2024. 3, 5

[36] Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3d gaussian splatting for real-time radiance field rendering. *ACM Transactions on Graphics (TOG)*, 42(4):1–14, 2023. 3

[37] Bernhard Kerbl, Georgios Kopanas, Thomas Leimkuhler, and George Drettakis. 3d gaussian splatting for real-time radiance field rendering. *ACM Transactions on Graphics (TOG)*, 42(4):1–14, 2023. 2

[38] Skanda Koppula, Ignacio Rocco, Yi Yang, Joe Heyward, Joao Carreira, Andrew Zisserman, Gabriel Brostow, and Carl Doersch. Tapvid-3d: A benchmark for tracking any point in 3d. *Advances in Neural Information Processing Systems*, 37: 82149–82165, 2024. 2, 6

[39] Yushi Lan, Yihang Luo, Fangzhou Hong, Shangchen Zhou, Honghua Chen, Zhaoyang Lyu, Shuai Yang, Bo Dai, Chen Change Loy, and Xingang Pan. Stream3r: Scalable sequential 3d reconstruction with causal transformer. *arXiv preprint arXiv:2508.10893*, 2025. 3

[40] 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. 3

[41] Tianye Li, Mira Slavcheva, Michael Zollhoefer, Simon Green, Christoph Lassner, Changil Kim, Tanner Schmidt, Steven Lovegrove, Michael Goesele, Richard Newcombe, et al. Neural 3d video synthesis from multi-view video. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5521–5531, 2022. 3

[42] Zhengqi Li, Simon Niklaus, Noah Snavely, and Oliver Wang. Neural scene flow fields for space-time view synthesis of dynamic scenes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6498–6508, 2021. 3

[43] Zhan Li, Zhang Chen, Zhong Li, and Yi Xu. Spacetime gaussian feature splatting for real-time dynamic view synthesis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8508–8520, 2024. 3

[44] Zhengqi Li, Richard Tucker, Forrester Cole, Qianqian Wang, Linyi Jin, Vickie Ye, Angjoo Kanazawa, Aleksander Holynski, and Noah Snavely. Megasam: Accurate, fast and robust structure and motion from casual dynamic videos. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 10486–10496, 2025. 2, 3

[45] Zizun Li, Jianjun Zhou, Yifan Wang, Haoyu Guo, Wenzheng Chang, Yang Zhou, Haoyi Zhu, Junyi Chen, Chunhua Shen, and Tong He. Wint3r: Window-based streaming reconstruction with camera token pool. *arXiv preprint arXiv:2509.05296*, 2025. 3

[46] Yiqing Liang, Abhishek Badki, Hang Su, James Tompkin, and Orazio Gallo. Zero-shot monocular scene flow estimation in the wild. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 21031–21044, 2025. 2, 3, 7

[47] Haotong Lin, Sida Peng, Zhen Xu, Yunzhi Yan, Qing Shuai, Hujun Bao, and Xiaowei Zhou. Efficient neural radiance fields for interactive free-viewpoint video. In *ACM Transactions on Graphics (SIGGRAPH Asia)*, pages 1–9, 2022. 3

[48] Xinhang Liu, Shiu-hong Kao, Jiaben Chen, Yu-Wing Tai, and Chi-Keung Tang. Deceptive-nerf: Enhancing nerf reconstruction using pseudo-observations from diffusion models. *arXiv preprint arXiv:2305.15171*, 2023. 3

[49] Xinhang Liu, Yu-Wing Tai, and Chi-Keung Tang. Clean-nerf: Reformulating nerf to account for view-dependent observations. *arXiv preprint arXiv:2303.14707*, 2023. 3

[50] Xinhang Liu, Yu-Wing Tai, Chi-Keung Tang, Pedro Miraldo, Suhas Lohit, and Moitreya Chatterjee. Gear-nerf: free-viewpoint rendering and tracking with motion-aware spatio-temporal sampling. In *Proceedings of the IEEE/CVF Con-*

ference on Computer Vision and Pattern Recognition, pages 19667–19679, 2024. 3

[51] Xinhang Liu, Yuxi Xiao, Donny Y. Chen, Jiashi Feng, Yu-Wing Tai, Chi-Keung Tang, and Bingyi Kang. Trace anything: Representing any video in 4d via trajectory fields, 2025. 3

[52] Yu-Lun Liu, Chen Gao, Andreas Meuleman, Hung-Yu Tseng, Ayush Saraf, Changil Kim, Yung-Yu Chuang, Johannes Kopf, and Jia-Bin Huang. Robust dynamic radiance fields. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 13–23, 2023. 3

[53] Ilya Loshchilov, Frank Hutter, et al. Fixing weight decay regularization in adam. *arXiv preprint arXiv:1711.05101*, 5 (5):5, 2017. 5

[54] Jiahao Lu, Tianyu Huang, Peng Li, Zhiyang Dou, Cheng Lin, Zhiming Cui, Zhen Dong, Sai-Kit Yeung, Wenping Wang, and Yuan Liu. Align3r: Aligned monocular depth estimation for dynamic videos. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 22820–22830, 2025. 3

[55] Jonathon Luiten, Georgios Kopanas, Bastian Leibe, and Deva Ramanan. Dynamic 3d gaussians: Tracking by persistent dynamic view synthesis. *arXiv preprint arXiv:2308.09713*, 2023. 3

[56] Ben Mildenhall, Pratul P. Srinivasan, Matthew Tancik, Jonathan T. Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. In *European Conference on Computer Vision (ECCV)*, pages 405–421, Glasgow, UK, 2020. Springer. 2, 3

[57] 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

[58] Raul Mur-Artal and Juan D Tardós. Orb-slam2: An open-source slam system for monocular, stereo, and rgb-d cameras. *IEEE transactions on robotics*, 33(5):1255–1262, 2017. 2

[59] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32, 2019. 5

[60] Ethan Perez, Florian Strub, Harm De Vries, Vincent Dumoulin, and Aaron Courville. Film: Visual reasoning with a general conditioning layer. In *Proceedings of the AAAI conference on artificial intelligence*, 2018. 4, 8

[61] Albert Pumarola, Enric Corona, Gerard Pons-Moll, and Francesc Moreno-Noguer. D-nerf: Neural radiance fields for dynamic scenes. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 10318–10327, 2021. 3

[62] Yi-Ling Qiao, Alexander Gao, Yiran Xu, Yue Feng, Jia-Bin Huang, and Ming C Lin. Dynamic mesh-aware radiance fields. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 385–396, 2023. 3

[63] Johannes L. Schönberger and Jan-Michael Frahm. Structure-from-motion revisited. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4104–4113, Las Vegas, NV, USA, 2016. IEEE. 2

[64] Ruizhi Shao, Zerong Zheng, Hanzhang Tu, Boning Liu, Hongwen Zhang, and Yebin Liu. Tensor4d: Efficient neural 4d decomposition for high-fidelity dynamic reconstruction and rendering. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 16632–16642, 2023. 3

[65] Xingjian Shi, Zhourong Chen, Hao Wang, Dit-Yan Yeung, Wai-Kin Wong, and Wang-chun Woo. Convolutional lstm network: A machine learning approach for precipitation nowcasting. *Advances in neural information processing systems*, 28, 2015. 2

[66] Cheng Sun, Min Sun, and Hwann-Tzong Chen. Direct voxel grid optimization: Super-fast convergence for radiance fields reconstruction. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5459–5469, 2022. 3

[67] Mingzhen Sun, Weining Wang, Xinxin Zhu, and Jing Liu. Moso: Decomposing motion, scene and object for video prediction. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 18727–18737, 2023. 2

[68] Zachary Teed and Jia Deng. Droid-slam: Deep visual slam for monocular, stereo, and rgb-d cameras. *Advances in neural information processing systems*, 34:16558–16569, 2021. 2

[69] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017. 8

[70] Kyle Vedder, Neehar Peri, Ishan Khatri, Siyi Li, Eric Eaton, Mehmet Kocamaz, Yue Wang, Zhiding Yu, Deva Ramanan, and Joachim Pehserl. Neural eulerian scene flow fields. *arXiv preprint arXiv:2410.02031*, 2024. 2

[71] Dor Verbin, Peter Hedman, Ben Mildenhall, Todd Zickler, Jonathan T Barron, and Pratul P Srinivasan. Ref-nerf: Structured view-dependent appearance for neural radiance fields. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5481–5490. IEEE, 2022. 3

[72] Hengyi Wang and Lourdes Agapito. 3d reconstruction with spatial memory. *arXiv preprint arXiv:2408.16061*, 2024. 2, 3

[73] Jianyuan Wang, Minghao Chen, Nikita Karaev, Andrea Vedaldi, Christian Rupprecht, and David Novotny. Vggt: Visual geometry grounded transformer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Vancouver, BC, Canada, 2025. Computer Vision Foundation / IEEE. Open Access CVF version. 2, 3, 5, 6, 7, 8

[74] 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. 2, 3, 5, 6, 7, 8

[75] 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. 2, 3

[76] Yunbo Wang, Mingsheng Long, Jianmin Wang, Zhifeng Gao, and Philip S Yu. Predrnn: Recurrent neural networks for predictive learning using spatiotemporal lstms. *Advances in neural information processing systems*, 30, 2017. 2

[77] Yunbo Wang, Zhifeng Gao, Mingsheng Long, Jianmin Wang, and Philip S Yu. Predrnn++: Towards a resolution of the deep-in-time dilemma in spatiotemporal predictive learning. In *International conference on machine learning*, pages 5123–5132. PMLR, 2018. 2

[78] Shuhuan Wen, Xiongfei Li, Xin Liu, Jiaqi Li, Sheng Tao, Yidan Long, and Tony Qiu. Dynamic slam: A visual slam in outdoor dynamic scenes. *IEEE Transactions on Instrumentation and Measurement*, 72:1–11, 2023. 3

[79] Guanjun Wu, Taoran Yi, Jiemin Fang, Lingxi Xie, Xiaopeng Zhang, Wei Wei, Wenyu Liu, Qi Tian, and Xinggang Wang. 4d gaussian splatting for real-time dynamic scene rendering. *arXiv preprint arXiv:2310.08528*, 2023. 3

[80] Rundi Wu, Ben Mildenhall, Philipp Henzler, Keunhong Park, Ruiqi Gao, Daniel Watson, Pratul P Srinivasan, Dor Verbin, Jonathan T Barron, Ben Poole, et al. Reconfusion: 3d reconstruction with diffusion priors. *arXiv preprint arXiv:2312.02981*, 2023. 3

[81] Yuqi Wu, Wenzhao Zheng, Jie Zhou, and Jiwen Lu. Point3r: Streaming 3d reconstruction with explicit spatial pointer memory. *arXiv preprint arXiv:2507.02863*, 2025. 3

[82] Jamie Wynn and Daniyar Turmukhambetov. Diffusionerf: Regularizing neural radiance fields with denoising diffusion models. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4180–4189, 2023. 3

[83] Wenqi Xian, Jia-Bin Huang, Johannes Kopf, and Changil Kim. Space-time neural irradiance fields for free-viewpoint video. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 9421–9431, 2021. 3

[84] Zhiwen Yan, Chen Li, and Gim Hee Lee. Nerf-ds: Neural radiance fields for dynamic specular objects. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 8285–8295, 2023. 3

[85] 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. 2, 3

[86] Ziyi Yang, Xinyu Gao, Wen Zhou, Shaohui Jiao, Yuqing Zhang, and Xiaogang Jin. Deformable 3d gaussians for high-fidelity monocular dynamic scene reconstruction. *arXiv preprint arXiv:2309.13101*, 2023. 3

[87] Zeyu Yang, Hongye Yang, Zijie Pan, Xiatian Zhu, and Li Zhang. Real-time photorealistic dynamic scene representation and rendering with 4d gaussian splatting. *arXiv preprint arXiv:2310.10642*, 2023. 3

[88] Zetong Yang, Li Chen, Yanan Sun, and Hongyang Li. Visual point cloud forecasting enables scalable autonomous driving. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14673–14684, 2024. 2

[89] Zeyu Yang, Hongye Yang, Zijie Pan, and Li Zhang. Real-time photorealistic dynamic scene representation and rendering with 4d gaussian splatting. In *The Twelfth International Conference on Learning Representations*, 2024. 3

[90] Xi Ye and Guillaume-Alexandre Bilodeau. Vptr: Efficient transformers for video prediction. In *2022 26th International conference on pattern recognition (ICPR)*, pages 3492–3499. IEEE, 2022. 2

[91] Zhengming Yu, Wei Cheng, Xian Liu, Wayne Wu, and Kwan-Yee Lin. Monohuman: Animatable human neural field from monocular video. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 16943–16953, 2023. 3

[92] Jiakai Zhang, Xinhang Liu, Xinyi Ye, Fuqiang Zhao, Yanshun Zhang, Minye Wu, Yingliang Zhang, Lan Xu, and Jingyi Yu. Editable free-viewpoint video using a layered neural representation. *ACM Transactions on Graphics (TOG)*, 40(4):1–18, 2021. 3

[93] 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. In *The Thirteenth International Conference on Learning Representations*, 2025. 2, 3, 7, 8

[94] 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 *IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 19855–19865, 2023. 2, 5, 6

[95] Zihan Zhu, Songyou Peng, Viktor Larsson, Weiwei Xu, Hujun Bao, Zhaopeng Cui, Martin R Oswald, and Marc Pollefeys. Nice-slam: Neural implicit scalable encoding for slam. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 12786–12796, 2022. 3

[96] Dong Zhuo, Wenzhao Zheng, Jiahe Guo, Yuqi Wu, Jie Zhou, and Jiwen Lu. Streaming 4d visual geometry transformer. *arXiv preprint arXiv:2507.11539*, 2025. 2, 3, 7, 8

Point4Cast: Streaming Dynamic Scene Reconstruction and Forecasting

Supplementary Material

We begin this supplementary document by providing details of the datasets used for training in Section **A**. In Section **B**, we present additional architectural specifications of our proposed model. Next, in Section **C**, we report inference speed comparisons between our approach and competing baselines for 3D point map reconstruction. We further include, in Section **D**, additional ablation studies that examine the influence of our time-conditioning strategy on overall performance. Finally, in Section **E** and Section **F**, we provide qualitative results for 3D point map reconstruction and forecasting, camera pose forecasting, and 3D point tracks across a diverse set of scenes. For the challenging task of 3D point map reconstruction and forecasting, we also present qualitative comparisons against state-of-the-art baselines.

The following summarizes the supplementary materials we present:

1. Training details, including those of the datasets used for training our model as well as loss plots.
2. Additional details about the network architecture of *Point4Cast* including discussion of adaptations needed to incorporate popular backbones into our framework.
3. Runtime comparison of different competing methods for 3D point map reconstruction.
4. Ablation studies comparing the different time conditioning mechanisms.
5. Qualitative results comparing *Point4Cast* with different competing methods on samples from an in-domain dataset as well as from unseen, online videos for the task of 3D point map reconstruction/forecasting.
6. Qualitative results of camera pose estimates and 3D point tracks obtained by our method.

In addition, in the supplementary bundle, we provide: a video showing examples of reconstruction/forecasting of 3D point maps and recovered camera poses on the chosen dataset as well as on videos captured in the wild, by *Point4Cast* and other competing methods, compiled together in *supplementary_video.mp4*.

A. Details of Training Datasets

Table A. **Description of training datasets used for *Point4Cast*.**

Dataset Name	# Videos	Total # Frames	Frame Resolution
Kubric [1]	1,000	∼24,000	512 × 512
PointOdyssey [9]	100	∼150,000	960 × 540
Stereo4D [2]	2,000	∼1,600,000	512 × 512
Custom synthetic dataset	2,000	∼240,000	960 × 540

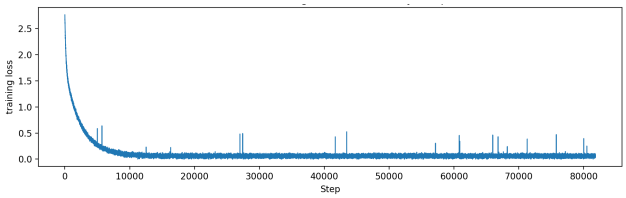

Figure A. Training Loss on the PointOdyssey dataset.

In order to equip, *Point4Cast* with strong inductive priors for online 3D point map reconstruction and forecasting of *dynamic scenes*, beginning with pre-trained modules of popular feed-forward 3D point map reconstruction frameworks [6, 7], we train our model on four diverse datasets of dynamic scenes: (i) Kubric [1], (ii) PointOdyssey [9], (iii) Stereo4D [2], and (iv) an additional synthetic dataset which we create by using Blender - coupling chosen Mixamo[1] motion assets with scene elements from BlenderKit[2].

The reuse of modules from popular backbones, such as CUT3R [7] or VGGT [6], provides *Point4Cast* with important priors about the geometry of the 3D world across several different types of scenes. Nonetheless, to further finetune our model with 3D priors from complex *dynamic scenes* and to also train the new, untrained modules introduced in our architecture, we leverage the datasets listed in Table A. In particular, we use: (i) The Kubric dataset [1], consisting of ∼1,000 indoor, synthetic videos which have metric depth provided with each video. Every video in the dataset is 24 frames long and has frames of 512 x 512 resolution. (ii) The PointOdyssey dataset [9], which comprises of ∼100 synthetic videos which also include metric depth per frame. Each video in the dataset is ∼1500 frames long and has frames of 960 x 540 resolution. (iii) The Stereo4D dataset [2], containing 2,000 real-world clips, each approximately ∼800 frames long, with frames of 512 x 512 resolution. (iv) Finally, we also construct an in-house, synthetic 3D-scene dataset which we construct by using Blender to render chosen Mixamo motion assets (such as human characters and animals) with scene elements (those that constitute the background) obtained from BlenderKit. Following this procedure, we obtain a dataset of 2k videos, where each video is ∼120 frames long and has frames of 960 x 540 resolution. This process yields perfect ground-truth metric depth for all frames.

Point4Cast is trained end-to-end with gradients through successive updates of the spacetime representation, similar to Backpropagation Through Time (BPTT), but super-

[1] https://www.mixamo.com/
[2] https://www.blenderkit.com/

vision after every update shortens gradient paths and avoids long-horizon error accumulation. Moreover, update/readout use attention-based transformers (not sequential RNNs), helping avoid vanishing gradients. Pretrained geometric backbones further stabilize optimization. *Point4Cast* is trained in a single-stage on 8×A100 (80GB), takes ∼107h, and converges smoothly (See Fig. A for loss curve on the PointOdyssey dataset).

B. Additional Architectural Details

Several components of *Point4Cast* are initialized using pretrained modules from established backbones for 3D point–map reconstruction from images, such as CUT3R [7] and VGGT [6]. In particular, the image encoder is initialized from the ViT backbone of VGGT, producing 14×14 patches that are embedded into 1024-dimensional tokens. The attention blocks of both the *UpdateTransformer* and the *ReadoutTransformer* are initialized from the Dense Prediction Transformer (DPT) modules of VGGT. The *UpdateTransformer* and *ReadoutTransformer* each consist of 12 attention layers configured as transformer decoders, with 16 multi-head attention (MHA) heads per layer. The prediction heads, Head_{map} and Head_{cam}, follow the architecture of VGGT's camera head and employ simple 1×1 convolutional layers to map the final 1024-dimensional tokens to the predicted 3D point map and camera parameters.

Incorporating different backbones: The flexible design of *Point4Cast*'s architecture allows for the integration of various backbone networks, enabling us to leverage their inductive biases for 3D point map reconstruction and forecasting. We experiment with two popular backbones: VGGT [6] and CUT3R [7]. When using VGGT as the backbone, we initialize the image encoder with the DINO encoder from VGGT. The *UpdateTransformer* and *ReadoutTransformer* modules are initialized using VGGT's transformer blocks. In addition, both the $\textit{Head}_{map}$ and $\textit{Head}_{cam}$ modules are initialized with the weights of VGGT's DPT heads. When using CUT3R [7] as the backbone, we initialize the image encoder with the pre-trained ViT encoder from CUT3R. The *UpdateTransformer* and *ReadoutTransformer* modules are both initialized using the transformer decoder from CUT3R.

C. Runtime Comparisons

In Table B, we compare the inference speed (FPS) of our method against several state-of-the-art baselines on the PointOdyssey dataset. Our approach achieves competitive runtime performance, demonstrating that the additional architectural components introduced in our framework do not impose any significant computational overhead.

Table B. **Runtime comparison of competing methods on the PointOdyssey dataset for 3D point map reconstruction.** Higher FPS indicates faster inference. The **best** result is highlighted.

Method	Inference speed (FPS) ↑
MonST3R [8]	4.29
VGGT [6]	1.13
CUT3R [7]	**23.74**
StreamVGGT [10]	17.18
Ours (VGGT backbone)	<u>20.83</u>

Table C. **Ablation study on the choice of Time Conditioning technique on the PointOdyssey dataset.** The design choices for our approach are highlighted.

Query Time Embedding	Conditioning	Acc. ↓	Comp. ↓
Sinusoidal	Cross-Attention	0.470	0.502
Learned	Cross-Attention	0.437	0.492
Learned	FiLM	**0.428**	**0.472**

D. Additional Ablation Results

In this section, we present some additional ablation results of our proposed approach.

D.1. Comparisons of Time-Conditioning Strategies

Table C summarizes the performances of plausible time-encoding and conditioning mechanisms on the PointOdyssey dataset, with the design choice of our model shown in the final row (highlighted in gray). Starting with the input time stamp, a sinusoidal embedding denotes a $C-$dimensional Positional Encoding (PE) of time. This maybe represented as:

$$e_t = [\sin(\omega_1 t), \cos(\omega_1 t), \ldots, \sin(\omega_{C/2} t), \cos(\omega_{C/2} t)] \in \mathbb{R}^C,$$

while a learned embedding directly maps the scalar time to a learned $C-$dimensional embedding vector.

$$e_t = \text{Embed}(t) \in \mathbb{R}^C,$$

With the time embeddings e_t in place, we explore two approaches towards deriving the time conditioning vector. The first leverages the popular FiLM [3] conditioning technique which may be represented by the equations shown below:

$$\gamma = W_\gamma e_t, \quad \beta = W_\beta e_t, \quad \gamma, \beta \in \mathbb{R}^C,$$

$$\mathbf{s}^{(t)}[i,:] = \gamma \odot \frac{\mathbf{w}_k[i,:] - \mu_i}{\sigma_i} + \beta, \qquad \forall i \in \{1, \ldots, N\} \tag{1}$$

Alternatively, we also explore the efficacy of a cross attention scheme as shown below:

$$\hat{s} = \text{MHA}(Q = \mathbf{w}_k, K = e_t, V = e_t), \tag{2}$$

$$\mathbf{s}^{(t)} = FFN(LN(\hat{s})) + \hat{s}, \tag{3}$$

where MHA denotes Multi-head Attention, Q, K, V denote the query, key, and value of the attention module, FFN denotes a feed-forward network while LN denotes Layer Norm, as is common in transformer modules [5].

As we see from the results in Table C learned time embeddings outperform sinusoidal ones, and FiLM [3] conditioning provides the largest improvements.

D.2. Frame-Rate and Spacetime Representation Sensitivity

Rate	Acc.↓	Comp.↓
60 FPS	0.415	0.461
30 FPS	0.427	0.469
10 FPS	0.474	0.493
Uneven	0.433	0.473

Table D. Time-spacing sensitivity.

Config		Acc.↓	Comp.↓	FPS↑
Backbone	Stream			
None	✓	0.719	0.897	20.83
CUT3R	×	0.539	0.662	5.75
CUT3R	✓	<u>0.484</u>	<u>0.516</u>	**22.05**
VGGT	×	0.671	0.838	1.47
VGGT	✓	**0.438**	**0.465**	<u>20.91</u>

Table E. Ablations on streaming versus full video setting on the PointOdyssey dataset.

In order to study the impact of the video frame rate on the reconstruction, we synthesize a 3 s, 60 FPS clip and create variants by down-sampling the clip to 30 and 10 FPS, as well as an unevenly spaced video setting. Table D shows that down-sampling the frae rate leads to performance degradation due to larger inter-frame motions, albeit gracefully. *Point4Cast* remains largely robust to uneven sampling.

We also ablate the performance of *Point4Cast* on the type of backbone it is initialized with and the use of the proposed streaming update versus a per-frame inference without maintaining a persistent spacetime state. The results in Table E show that enabling the spacetime representation consistently improves reconstruction quality across both backbones, while also yielding a substantial efficiency gain in terms of FPS. The VGGT backbone combined with the streaming update achieves the best reconstruction accuracy with high throughput.

E. Qualitative 3D Reconstruction and Forecasting Results

Qualitative 3D point map reconstruction and forecasting results obtained by our method against competing, state-of-the-art baselines (MonST3R [8] and VGGT [6]) are shown in Figure E on the Point Odyssey dataset as well as on unseen, real-world, outdoor videos, as shown in Figure B and Figure C. Since, the competing baselines are not equipped for the novel task of 3D point map forecasting, we use optical flow [4] to first forecast the 2d image frames and then use the baseline approaches to predict the 3D point maps. We see that our proposed approach exhibits denser and more coherent 3D point maps with smoother motion, across both the reconstructed (yellow/blue) and forecasted (green) point maps, compared to the competing methods. Importantly, our proposed approach is able to forecast motions across both rigid (such as cars) and non-rigid (such as humans) objects, attesting to the effectiveness of our method. Additionally, Figure D shows that the estimated camera poses, obtained by our method, align well with human intuition - attesting to the effectiveness of our approach.

F. Qualitative 3D Point Track Results

One key feature of the design of our approach is that we obtain correspondences between point clouds across time steps, without any additional inference or training. We are thus able to compute tracks of 3D points over time. In Figure F, we show qualitative results for the same, across two different unseen sequences for videos captured in the wild. In both cases, we see that the point tracks are consistent with the motion of the vehicles (the car on the side of the street and the bus respectively.)

References

[1] Klaus Greff, Francois Belletti, Lucas Beyer, Carl Doersch, Yilun Du, Daniel Duckworth, David J Fleet, Dan Gnanapragasam, Florian Golemo, Charles Herrmann, Thomas Kipf, Abhijit Kundu, Dmitry Lagun, Issam Laradji, Hsueh-Ti (Derek) Liu, Henning Meyer, Yishu Miao, Derek Nowrouzezahrai, Cengiz Oztireli, Etienne Pot, Noha Radwan, Daniel Rebain, Sara Sabour, Mehdi S. M. Sajjadi, Matan Sela, Vincent Sitzmann, Austin Stone, Deqing Sun, Suhani Vora, Ziyu Wang, Tianhao Wu, Kwang Moo Yi, Fangcheng Zhong, and Andrea Tagliasacchi. Kubric: a scalable dataset generator. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022. 1

[2] Linyi Jin, Richard Tucker, Zhengqi Li, David Fouhey, Noah Snavely, and Aleksander Holynski. Stereo4d: Learning how things move in 3d from internet stereo videos. *arXiv preprint arXiv:2412.09621*, 2024. 1

[3] Ethan Perez, Florian Strub, Harm De Vries, Vincent Dumoulin, and Aaron Courville. Film: Visual reasoning with a general conditioning layer. In *Proceedings of the AAAI conference on artificial intelligence*, 2018. 2, 3

[4] Zachary Teed and Jia Deng. Raft: Recurrent all-pairs field transforms for optical flow. In *European conference on computer vision*, pages 402–419. Springer, 2020. 3

[5] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017. 3

[6] Jianyuan Wang, Minghao Chen, Nikita Karaev, Andrea Vedaldi, Christian Rupprecht, and David Novotny. Vggt: Visual geometry grounded transformer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Vancouver, BC, Canada, 2025. Computer Vision Foundation / IEEE. Open Access CVF version. 1, 2, 3

[7] 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. 1, 2

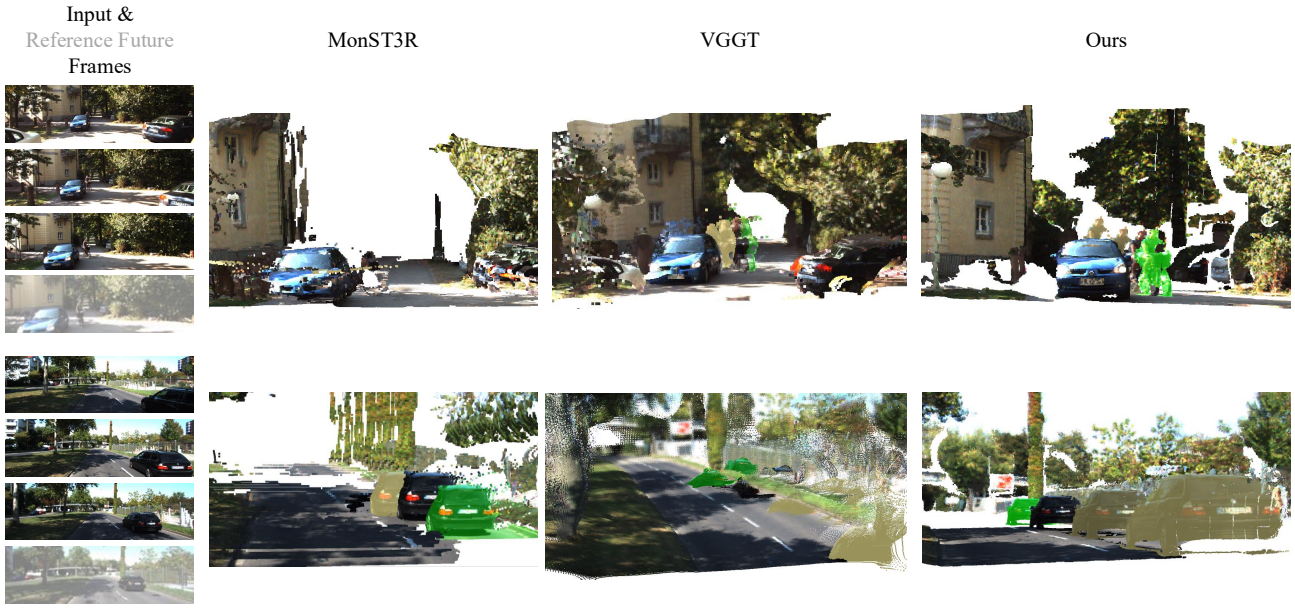

Figure B. **Qualitative comparison of dynamic-scene reconstruction (shown in blue) and forecasting (shown in green) on unseen videos, captured in the wild.** We compare *Point4Cast* (our proposed approach) with MonST3R and VGGT (adapted to use scene flow for forecasting) on challenging outdoor scenes. *Point4Cast* produces more complete and temporally consistent 3D point maps, with sharper geometry, fewer artifacts, and more reasonable future predictions.

Figure C. **Qualitative comparisons on challenging, dynamic-scene reconstruction (shown in blue) and forecasting (shown in green) on unseen videos of outdoor scenes,** between *Point4Cast* and VGGT.

[8] 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. In *The Thirteenth International Conference on Learning Representations*, 2025. 2, 3

[9] 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 *IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 19855–19865, 2023. 1, 5

[10] Dong Zhuo, Wenzhao Zheng, Jiahe Guo, Yuqi Wu, Jie Zhou, and Jiwen Lu. Streaming 4d visual geometry transformer. *arXiv preprint arXiv:2507.11539*, 2025. 2

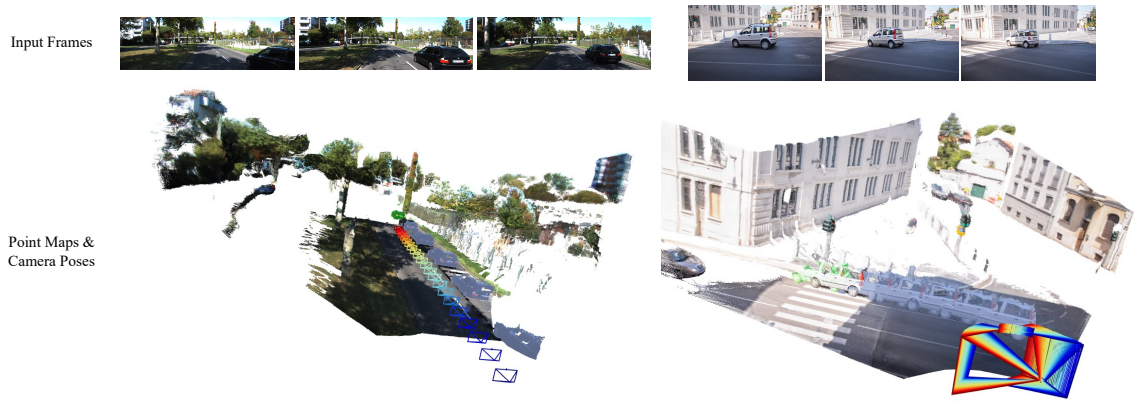

Figure D. **Qualitative visualization of camera trajectories, obtained by our method, across two unseen videos, captured in-the-wild.**

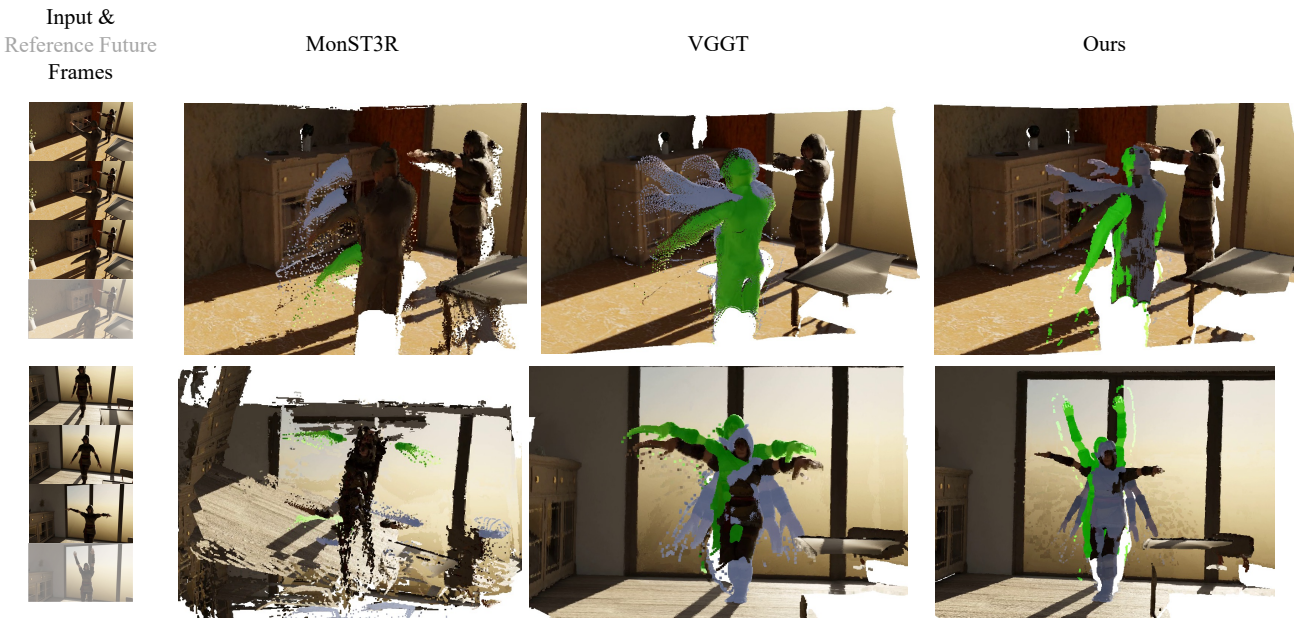

Figure E. **Qualitative comparison of 3D point map reconstruction (shown in blue) and forecasting (shown in green) on the Point Odyssey dataset [9].** We compare *Point4Cast* (our proposed approach) with MonST3R and VGGT (adapted to use scene flow for forecasting) on challenging scenes featuring non-rigid human motion. *Point4Cast* produces more complete and temporally consistent 3D point maps, with sharper geometry, fewer artifacts, and more reasonable future predictions.

Figure F. **Qualitative visualization of 3D point tracks, obtained by our method, over time steps.**