Tracktention: Leveraging Point Tracking to Attend Videos Faster and Better

Zihang Lai Andrea Vedaldi
Visual Geometry Group (VGG), University of Oxford
{zlai,vedaldi}@robots.ox.ac.uk

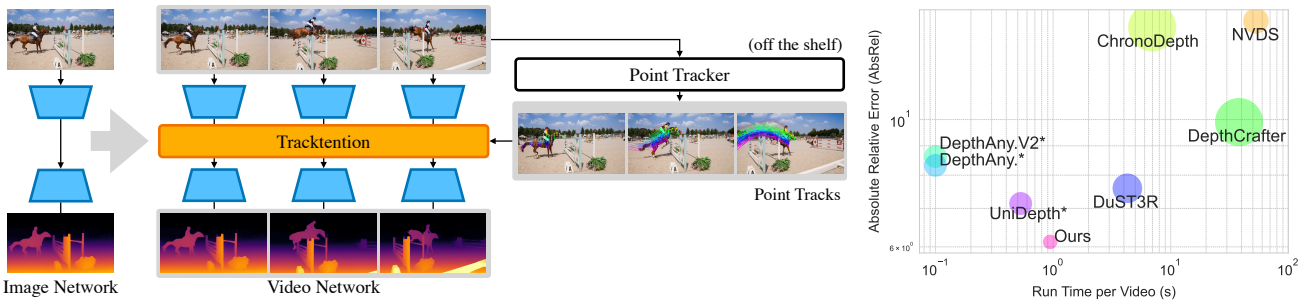

Figure 1. *Left*: The **Tracktention Layer** is a plug-and-play module that can convert an image-based network (*e.g.*, for monocular depth prediction) into a state-of-the-art video network (*e.g.*, for video depth prediction). It does so by integrating the output of any off-the-shelf, modern, and powerful point trackers via track cross-attention. *Right*: For example, Tracktention achieves **state-of-the-art and efficient video depth** prediction by transforming Depth Anything into a video depth model. See Tab. 2 for detailed results. *Single-image models.

Abstract

Temporal consistency is critical in video prediction to ensure that outputs are coherent and free of artifacts. Traditional methods, such as temporal attention and 3D convolution, may struggle with significant object motion and may not capture long-range temporal dependencies in dynamic scenes. To address this gap, we propose the Tracktention Layer, a novel architectural component that explicitly integrates motion information using point tracks, i.e.*, sequences of corresponding points across frames. By incorporating these motion cues, the Tracktention Layer enhances temporal alignment and effectively handles complex object motions, maintaining consistent feature representations over time. Our approach is computationally efficient and can be seamlessly integrated into existing models, such as Vision Transformers, with minimal modification. It can be used to upgrade image-only models to state-of-the-art video ones, sometimes outperforming models natively designed for video prediction. We demonstrate this on video depth prediction and video colorization, where models augmented with the Tracktention Layer exhibit significantly improved temporal consistency compared to baselines. Project website:* zlai0.github.io/TrackTention.

1. Introduction

Compared to image analysis tasks, video analysis tasks, such as video segmentation, video depth estimation, and video colorization, pose additional challenges due to the temporal dimension inherent in video data. A particularly important challenge is ensuring that outputs are temporally consistent across frames, which is necessary to produce coherent and artifact-free results. Temporal inconsistencies, such as flickering or abrupt changes in predicted object attributes, are especially noticeable in applications like colorization, where the output is a new video.

An aspect of video analysis that has seen significant progress in recent years is *point tracking*. New trackers such as PIPs [19], TAPIR [12], BootsTAP [13], and CoTracker [26] can now track quasi-dense collections of points across long video sequences with high reliability and efficiency. So far, these trackers have been primarily used in selected applications like 3D reconstruction and object tracking through ad-hoc algorithms. In this paper, we ask whether progress in point tracking can benefit a much wider range of video analysis tasks.

To answer this question, we propose the *Tracktention Layer*, a novel architectural component designed to enhance temporal consistency in video transformers. Tracktention integrates seamlessly into vision transformers as a new at-

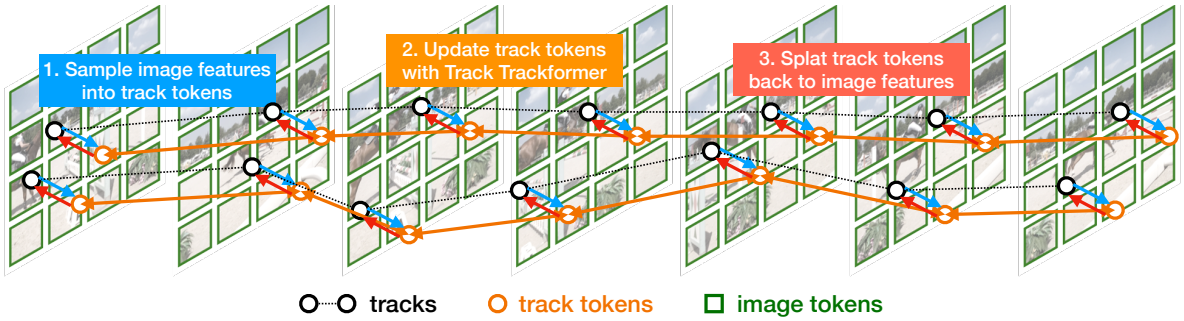

Figure 2. **Overview of Tracktention**. We begin by using an off-the-shelf point tracker to extract a number of video tracks. Given these, we first **sample** image tokens at the track locations, obtaining corresponding track tokens (Sec. 3.1). Next, we use a Track Transformer to **update** these tokens, propagating information temporally at consistent spatial locations (Sec. 3.2). Finally, we **splat** the information back to the image tokens (Sec. 3.3). By explicitly incorporating motion information through point tracks, Tracktention improves temporal alignment, effectively captures complex object movements, and ensures stable feature representations over time.

tention layer, making it compatible with a wide range of existing models. The key innovation of Tracktention lies in leveraging pre-extracted point tracks to inform the model about temporal correspondences between image tokens in different frames (Fig. 2). This is achieved by sampling the image tokens at the track points. The information is then propagated *along* the tracks using a transformer layer, which justifies the name "Tracktention". Finally, the information is *splatted* back to the image tokens, allowing the computation to progress as normal.

By explicitly incorporating tracking information, the model can establish direct correspondences, enabling precise temporal alignment regardless of how much objects move. This explicit temporal alignment allows the model to maintain consistent feature representations over time, smoothing features, and reducing temporal artifacts.

The problem of propagating information in video neural networks has been extensively studied in the literature. Many approaches are *implicit*, in the sense that they do not explicitly estimate or account for the motion observed in the scene. Examples include 3D convolution [6, 53] and space-time attention [1, 3, 14, 40, 58]. However, to maintain manageable computational costs, these methods often reduce the spatial resolution of features and limit the temporal range they can model—either by using small convolution kernels or by decoupling spatial and temporal attention. The resulting networks may struggle to represent motion precisely and to keep up with large object displacements.

In contrast, Tracktention explicitly accounts for motion at the resolution of point tracks, which is much finer than that of typical image features. By pooling and splatting information at image locations that are *already* put in correspondence by the tracker, it only needs to propagate information along the tracks, obviating the need to implicitly establish such correspondences via space-time attention.

There are also approaches to video analysis that, like Tracktention, incorporate motion information *explicitly*. They often work by feeding optical flow to the network, capturing pixel-level motion between frames [42]. While optical flow can be effective in certain contexts, it may struggle with occlusions or large displacements. Building on the power of point trackers, Tracktention can handle occlusions, large displacements, and long temporal dependencies more effectively.

In summary, Tracktention offers several advantages over traditional video network architectures: (1) It leverages the power of modern point tracker models, which are excellent *motion experts*, and provides a general mechanism to inject this knowledge into any video transformer. In particular, it can handle complex and large object motions, limited only by the capabilities of the point trackers. (2) It directly establishes long-term space-time correspondences, unlike 3D convolutions that are local and unlike separated space-time attention, which can only do so indirectly. (3) It avoids the need to repeatedly calculate such correspondences explicitly, as full spatio-temporal attention does, achieving much greater efficiency and operating at a much finer resolution.

We deliver Tracktention as a **plugin layer** (Fig. 1) that can be added to single-image neural network architectures to extend them to powerful, temporally-aware video models. For example, we demonstrate that a pre-trained single-image depth predictor can be extended to a corresponding *state-of-the-art video depth predictor* by integrating our Tracktention Layer. We achieve similar results for *automatic video colorization*, outperforming models natively designed to process videos while also being more efficient.

2. Related Work

Temporal Modeling for Videos. Temporal modeling is a cornerstone of video understanding tasks. Over the years, various techniques have been proposed to capture temporal dependencies in video data. 3D CNNs [6, 16, 53] extend 2D convolutions into the temporal dimension, allowing for direct spatiotemporal feature extraction from video data. Temporal attention mechanisms provide a new approach to

effectively capture long-range dependencies across frames. Video transformers [1, 3, 15, 37, 42] apply self-attention mechanisms over temporal and spatial dimensions, typically separately or hierarchically, to reduce computational complexity while modeling temporal dependencies effectively. Recent video processing models [4, 5, 50] also use a hybrid of these approaches. However, these methods have notable limitations: temporal attention struggles with rapid object movements, while 3D convolutions assume local spatiotemporal correlations, failing with unpredictable motion. In this work, we explicitly leverage tracking information to precisely align temporal features regardless of object movement.

Consistent Video Prediction. Addressing temporal inconsistency is critical for ensuring smooth and coherent video outputs. Optical flow-based methods [69] enforce temporal coherence by aligning features or predictions across frames, propagating features along motion paths estimated by optical flow. Liu et al. [35] utilize convolutional LSTMs to capture temporal information for consistent video frame synthesis, modeling temporal dependencies through recurrent connections. Lai et al. [30] propose a post-processing model with a temporal consistency objective for video prediction. Similarly, Luo et al. [38] and Kopf et al. [29] present test-time training post-processing methods specifically designed to stabilize video depth predictions using geometric consistency cues. In contrast, our work aims to *stabilize features* by allowing them to attend to corresponding areas across time based on point tracks. This approach inherently accounts for object movement and embeds temporal alignment efficiently within the model.

Point Tracking. Point tracking is the task of identifying and continuously tracking specific points across frames in a video. PIPs [18] pioneered a neural network-based approach that updates tracked points iteratively by extracting correlation maps between frames, inspired by the RAFT model for optical flow [52]. TAP-Vid [10] proposed a comprehensive benchmark and TAP-Net, a new model for point tracking. TAPIR [11] further improved performance by combining global matching capabilities with PIPs, enhancing the accuracy of tracked points even in complex motion scenarios. OmniMotion [55] tracks points by leveraging a quasi-3D canonical volume but is approximately 10^3 times slower than CoTracker due to its test-time optimization. CoTracker [25] leveraged a transformer-based architecture to track multiple points jointly, yielding improvements particularly for occluded points. Recent methods such as LocoTrack [7], which introduced 4D correlation features, and TAPTR [32] further improved precision.

In this work, we do not propose new models for point tracking. Instead, we use existing trackers to establish direct token correspondences, ensuring precise temporal alignment despite object motion.

3. Method

In this section, we introduce Tracktention, a novel transformer layer designed to aggregate and redistribute feature information in video transformers in a motion-aware manner (Fig. 3). By leveraging motion information captured by the output of a point tracker, this layer improves the temporal consistency and motion awareness of video features.

Tracktention comprises an Attentional Sampling block (Sec. 3.1) that pools information from the video features into the tracks, a Track Transformer (Sec. 3.2) that updates the track tokens, and an Attentional Splatting block (Sec. 3.3) that pushes the updated track tokens back to the video features. We describe these components next.

3.1. Attentional Sampling

The *Attentional Sampling* module is the first stage in Tracktention. It uses cross-attention to pool information from the video features into the track tokens.

Let $F \in \mathbb{R}^{T\times HW\times D_f}$ denote the input feature map, where T is the number of time steps, H and W are the spatial dimensions, and D_f is the feature dimension. Let $\mathbf{P} \in \mathbb{R}^{T\times M\times 2}$ represent a set of M tracks, each of which is a sequence of 2D points. We first apply a positional embedding to these 2D points, obtaining one token per track and per frame, and arrange these tokens into a tensor $\mathcal{T} \in \mathbb{R}^{T\times M\times D_f}$. We then project the track tokens into queries $Q \in \mathbb{R}^{T\times M\times D_k}$ and the feature map into keys $K \in \mathbb{R}^{T\times HW\times D_k}$ and values $V \in \mathbb{R}^{T\times HW\times D_f}$ as:

$$Q_t = \mathcal{T}_t W_Q, \quad K_t = F_t W_K, \quad V_t = F_t,$$

where W_Q and W_K are learnable weight matrices, and the subscript $t \in \{1,\dots,T\}$ indexes the first dimensions of the tensors, extracting matrices. Note that we avoid projecting the values to preserve the original features during sampling.

For each time t, the attention weights $A_t \in \mathbb{R}^{M\times HW}$ are computed using the scaled dot-product attention mechanism:

$$A_t = \mathrm{softmax}\left(\frac{Q_t K_t^\top}{\sqrt{D_k}} + B_t\right), \tag{1}$$

where $B_t \in \mathbb{R}^{M\times HW}$ is a bias term that encourages attention to align with the tracks. It is defined as:

$$B_{tij} = -\frac{\|P_{ti} - \mathrm{pos}_{F_t}(j)\|^2}{2\sigma^2},$$

where $i \in \{1,\dots,M\}$, $j \in \{1,\dots,HW\}$, $P_{ti} \in \mathbb{R}^2$ is the spatial location of track i at time t, and $\mathrm{pos}_{F_t}(j) \in \mathbb{R}^2$ is the spatial location of the j-th feature map token at time t.

In the softmax operator of Eq. (1), this bias is analogous to multiplying the attention map by a Gaussian window that decays exponentially as the feature position deviates from the track position. The parameter σ, which we set to $1/2$, determines how fast the window decays.

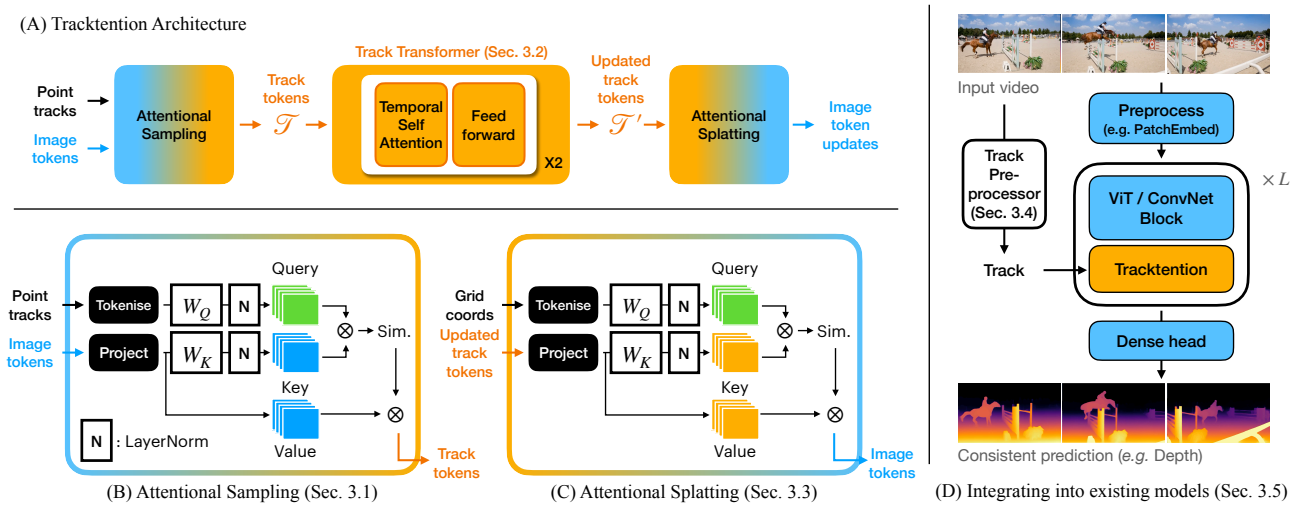

Figure 3. Left: the **Tracktention architecture** comprises Attentional Sampling, pooling information from images to track, Track Transformer, processing this information temporally, and Attentional Splatting, moving the processed information back to the images. Right: Tracktention is easily **integrated** in ViTs and ConvNets to make video networks out image ones.

We use the attention to "sample" track features $S \in \mathbb{R}^{T\times M\times D_f}$ by setting $S_t = A_t V_t = A_t F_t \in \mathbb{R}^{M\times D_f}$ for each time t. Alternatively, we could have obtained S by *sampling* the features F at the given track locations, for instance, by using bilinear sampling. This can also be expressed as $A_t F_t$ for a specific matrix A_t, so sampling is a special case of our Attentional Sampling block. The advantage of Attentional Sampling is that it allows the model to learn a better sampling strategy.

Relative Spatial Information. To capture the relative spatial relationships between track tokens and feature map locations, we incorporate Rotational Position Encodings (RoPE) [51] into the keys and values derived from the feature map. RoPE is a form of positional encoding that allows attention mechanisms to be aware of the relative positions of points defined in a continuous space.

To compute the RoPE embedding $\text{RoPE}(\mathbf{f})$ of a feature vector $\mathbf{f}$ at spatial location (x, y), we use the first $d/2$ channels to encode the x position. For $i = 1, \ldots, d/4$, we define:

$$\text{RoPE}(\mathbf{f})[2i] = \mathbf{f}[2i]\cdot\cos(\theta_i x) - \mathbf{f}[2i+1]\cdot\sin(\theta_i x),$$
$$\text{RoPE}(\mathbf{f})[2i+1] = \mathbf{f}[2i]\cdot\sin(\theta_i x) + \mathbf{f}[2i+1]\cdot\cos(\theta_i x),$$

where $\theta_i = 100^{\frac{-2(i-1)}{d/2}}$. The second $d/2$ channels encode the y position analogously by replacing x with y in the above equations.

Implementation Details. To stabilize the training process, especially due to the difference in scale between the track token embeddings and the feature map embeddings (which tend to have larger norms), we apply QK-normalization [9] to the queries and keys before computing the attention weights. This normalization ensures that the scale of the embeddings is consistent, preventing issues related to gradient instability. We also use H attention heads, allowing the model to capture diverse contextual relationships across different subspaces.

3.2. Track Transformer

Once the track tokens S are obtained, the *Track Transformer* processes them to propagate information along the tracks, making them more temporally consistent and smoothing out any irregularities. It is designed as a transformer that operates along the time dimension of the point tracks and outputs updated track tokens $\mathcal{T}'$.

In more detail, recall that the track tokens S are of shape $T \times M \times D_f$. We swap the first two dimensions to obtain a tensor of shape $M \times T \times D_f$. Then, we apply a transformer block, interpreting the first dimension M as the batch dimension. The effect is to apply attention along the temporal dimension, allowing the model to learn temporal dependencies along each track.

This approach ensures that information is not exchanged across tracks, meaning each track's temporal sequence is processed independently. We considered a variant of the Track Transformer that included attention across tracks as well, but found that this approach was slightly worse and more computationally expensive. Our interpretation is that information is already exchanged spatially by the Vision Transformer paired with Tracktention, making additional exchange of information across locations redundant.

Architecture. For each track, we utilize a 2-layer transformer encoder along the temporal dimension, structured as a standard transformer model. Each layer consists of Multi-Head Self-Attention, which allows each time step within a track to attend to all other time steps within that same track,

and Position-Wise Feed-Forward Networks, which apply nonlinear transformations independently to each time step, enhancing the model's representational capacity. We also incorporate sinusoidal positional encodings to provide the model with a sense of temporal order. These encodings are added to the input features prior to the transformer layers, ensuring the model can distinguish between different time steps and understand the temporal structure of the sequence.

3.3. Attentional Splatting

The *Attentional Splatting* module maps the updated track tokens $\mathcal{T}'$ back onto the feature maps F. This is achieved by reversing the roles of the track tokens and the feature tokens in the various expressions in Sec. 3.1. This means the queries are derived from the grid coordinates of the output feature map, while the keys and values are generated from the track tokens. The same bias term B_t is used as before (but now the matrix is transposed), as well as the same QK-normalization and RoPE encodings, this time obtaining a matrix $A'_t \in \mathbb{R}^{HW \times M}$. The final output $\mathrm{Tracktention}(F)$ is then computed as:

$$[\mathrm{Tracktention}(F)]_t = W_{\text{out}} A'_t [\mathcal{T}']_t,$$

where W_{out} is a final output projection. By adopting a symmetric design for Attentional Sampling and Splatting, we ensure that information is handled in a consistent manner.

3.4. Point Tracker Pre-Processor

To apply the Tracktention layer to a video, we first need to extract the point tracks $\mathbf{P}$. We use CoTracker3 [24] due to its robustness in handling complex motions and occlusions in videos. Trackers require a seed point for each track, called a query. We sample 576 points uniformly at random from the spatio-temporal video volume and track each forward and backward in time. Compared to initializing tracks on a grid in the first frame of the video, this simple initialization scheme encourages tracks to cover the video well, despite camera and object motion.

3.5. Integration into Image-Based Models

A key application of Tracktention is to build video neural networks from image-based ones. Integrating Tracktention into existing image-based models, such as a ViT or ConvNet, is straightforward. For example, monocular depth estimation models like Depth Anything, which are designed for single images, may lack temporal consistency when applied to video frame by frame. Adding Tracktention layers after each transformer block and fine-tuning the model on videos can increase temporal stability and significantly improve performance in video-based applications.

To incorporate Tracktention into an existing network backbone, we insert it immediately after all or some transformer or convolutional blocks. We also include a residual connection to preserve the original information flow, setting $F' = F + \mathrm{Tracktention}(F)$, where F represents the output feature map from the preceding block, and F' is the updated feature map after the Tracktention module.

Furthermore, the Tracktention output projection, W_{out}, is initialized to zero, preserving the original network output at training onset. This strategy retains benefits of pre-trained weights while allowing the model to gradually adapt to temporal updates. Consequently, Tracktention modules capture fine spatial details and ensure consistent temporal modeling across layers.

4. Experiments

We show that Tracktention can turn image-based models into state-of-the-art ones for video depth estimation and colorization (Secs. 4.1 and 4.2) and ablate its design (Sec. 4.3).

4.1. Video Depth Estimation

To apply Tracktention to video depth estimation, we start from an image-based model, Depth Anything [61], and upgrade it to a video-based model using the Tracktention module. The Depth Anything model has a DINO ViT backbone consisting of 12 transformer blocks. We insert our Tracktention module after each of the last 6 blocks.

We fine-tune the model on a large dataset of videos, using a combination of synthetic and real data. This data comes with ground-truth depth, which we fit using a scale- and shift-invariant loss, as Depth Anything predicts depth up to an affine transformation. However, we share the *same* calibration parameters for *all* frames in a video, as calibrating frames independently would mask temporal inconsistencies, which we want the model to learn to correct.

To reduce the number of new parameters added to the model, we share them between all Tracktention modules. During training, the original model is kept frozen, and only the Tracktention modules are updated. We use the AdamW optimizer with an initial learning rate of 1.6×10^{-5} and cosine learning rate decay. Training is conducted for 4 epochs with a batch size of 4 videos, each containing a randomly sampled number of frames between 8 and 16.

Data and evaluation metrics. For training, we use a combination of datasets containing both synthetic and real videos. We use standard depth estimation metrics [62]. Please refer to Appendix A.3.1 for details.

4.1.1 Quantitative Results

In Tab. 1, we evaluate our model on four video collections in the DepthCrafter benchmark [20] against several alternatives. Tracktention improves the image-based model it augments, *i.e.*, Depth Anything [62], substantially: Sintel AbsRel improves by 9.2% and δ_1 by 13.5%, KITTI by 26.8% and 12.5%, Bonn by 15.4% and 3.4%, and on average by

Method	Type	#Params	Sintel (∼50 frames)		Scannet (90 frames)		KITTI (110 frames)		Bonn (110 frames)		Average	
			AbsRel ↓	$\delta_{1.25}$ ↑	AbsRel ↓	$\delta_{1.25}$ ↑	AbsRel ↓	$\delta_{1.25}$ ↑	AbsRel ↓	$\delta_{1.25}$ ↑	AbsRel ↓	$\delta_{1.25}$ ↑
DUSt3R [56]	Vid.	578M	0.628	0.393	0.194	0.694	0.292	0.456	0.250	0.588	0.341	0.533
NVDS [59]	Vid.	430M	0.408	0.483	0.187	0.677	0.253	0.588	0.167	0.766	0.254	0.629
ChronoDepth [49]	Vid.	1521M	0.587	0.486	0.159	0.783	0.167	0.759	0.100	0.911	0.253	0.735
DepthCrafter [20]	Vid.	1521M	0.343†	**0.673**†	0.125	0.848	0.110	0.881	0.075	**0.971**	0.163	0.843
Marigold [27]	Im.	865M	0.532	0.515	0.166	0.769	0.149	0.796	0.091	0.931	0.235	0.753
DepthAny. [62] (∗)	Im.	343M	0.325	0.564	0.130	0.838	0.142	0.803	0.078	0.939	0.169	0.786
DepthAny.-V2 [63]	Im.	343M	0.367	0.554	0.135	0.822	0.140	0.804	0.106	0.921	0.187	0.775
Ours	Vid.	140M	**0.295**	0.640	**0.087**	**0.933**	**0.104**	**0.903**	**0.066**	**0.971**	**0.138**	**0.862**

Table 1. **DepthCrafter [20] video depth benchmark.** Our model upgrades DepthAnything (∗) to a video depth predictor, outperforming all baselines (image- or video-based) while having the smallest parameter count (140M). We use DepthAnything-Base (97M) as the base model and still outperform the DepthAnything-Large (343M) model. †Reproduced results confirmed by original authors.

Methods	GT Pose	GT Intr.	Align	KITTI		ScanNet		ETH3D		DTU		T&T		Average		Runtime (s)
				rel ↓	$\delta_{1.03}$ ↑	rel ↓	$\delta_{1.03}$ ↑	rel ↓	$\delta_{1.03}$ ↑	rel ↓	$\delta_{1.03}$ ↑	rel ↓	$\delta_{1.03}$ ↑	rel ↓	$\delta_{1.03}$ ↑	
COLMAP [45, 46]♯	✓	✓	×	**12.0**	**58.2**	**14.6**	**34.2**	**16.4**	**55.1**	**0.7**	**96.5**	**2.7**	**95.0**	**9.3**	**67.8**	≈ 3 min
COLMAP Dense [45, 46]	✓	✓	×	26.9	52.7	38.0	22.5	89.8	23.2	20.8	69.3	25.7	76.4	40.2	48.8	≈ 3 min
NVDS [54]	×	×	LS_{vid}	17.4	13.9	13.0	17.1	21.4	11.9	7.0	31.6	15.5	14.4	14.9	17.8	53.0
ChronoDepth [49]	×	×	LS_{vid}	15.3	14.8	14.7	14.5	21.1	11.3	7.0	31.0	14.3	16.8	14.5	17.7	7.0
DepthCrafter [20]	×	×	LS_{vid}	10.2	22.2	6.6	34.6	15.4	14.8	4.5	46.9	12.9	18.4	9.9	27.4	38.1
DepthAny. [62] (∗)	×	×	LS_{vid}	8.4	28.1	5.2	44.3	13.9	18.6	3.7	55.1	10.4	25.3	8.3	34.3	**0.1**
DepthAny.V2 [63]	×	×	LS_{vid}	9.1	26.3	4.6	50.0	16.1	16.0	**3.2**	61.6	10.1	26.5	8.6	36.1	**0.1**
DUSt3R [56]	×	×	LS_{vid}	7.2	31.7	4.6	50.0	13.4	19.0	3.4	59.2	9.3	28.4	7.6	37.7	4.3
UniDepth [44]	×	×	LS_{vid}	(3.9)	(58.2)	(1.5)	(90.1)	12.7	18.9	7.0	30.7	10.6	23.2	7.1	44.2	0.5
Ours	×	×	LS_{vid}	**6.9**	**32.4**	**4.5**	**50.8**	**9.2**	**27.1**	**3.2**	**62.4**	**7.3**	**32.9**	**6.2**	**41.1**	0.9‡

Table 2. **RobustMVD [48] multi-view depth benchmark**, analogous to Tab. 1. Parentheses indicate that a model was trained on the same benchmark used for evaluation, which provides an advantage compared to training on different data. ‡Includes approximately 0.4 seconds for tracking. ♯Metrics computed only for pixels with a valid prediction, which is an easier task.

18.3% and 9.7%. Second, Tracktention performs better than all other baselines, including DepthCrafter [20], which is a video depth model, by 15.3% and 2.3% on AbsRel and $\delta_{1.25}$, respectively. This demonstrates that Tracktention can turn an image-based model into a video-based one, outperforming models designed for video data from the start.

Table 2 reports results on the RobustMVD benchmark [48], which features shorter video clips with potentially larger camera motions. Our model achieved the lowest average relative error of 6.2 and the highest average inlier ratio of 41.1%, surpassing all baselines. In particular, it improves the base model Depth Anything by 33.8% on ETH3D and by 29.8% on Tanks and Temples.

Our model is also efficient, utilizing only 140 million parameters — only 17.1 million for the Tracktention module. It operates with a runtime of only 0.9 seconds per sequence, of which 0.4 seconds are attributed to the tracker. Despite its smaller size, our model outperforms DepthCrafter [20], which relies on video diffusion models with over 1.5 billion parameters. Tracktention will further benefit from future accuracy and speed improvements in point trackers.

4.1.2 Qualitative Results

Figure 4 shows a qualitative comparison of depth estimation by our model, DepthCrafter, and DUSt3R in dynamic scenes, complex textures, and cluttered environments. Our model produces stable, coherent depth maps across all frames, demonstrating strong temporal consistency. In contrast, DepthCrafter shows significant depth estimation errors in certain regions (blue box), especially in scenes with complex structures. DUSt3R, relying on implicit triangulation, struggles with dynamic content, resulting in inconsistent depth maps across time.

Figure 6 illustrates the Attentional Sampling block, which pools information along the track by attending to image features in correspondence with the track locations.

4.2. Automatic Video Colorization

The goal of video colorization is to add colors to an input grayscale video. Colors should be realistic, vibrant, and consistent across frames to avoid flickering.

We enhance four models — CIC [66], IDC [67], Colorformer [21], and DDColor [22] — by incorporating our Tracktention module into layers of each model's architecture. For instance, for DDColor, which has a ConvNet-based architecture with a ConvNeXt backbone [36] and a U-Net decoder, we insert our Tracktention module before each decoder block. We fine-tune the model using AdamW on the YouTube-VIS training set [60]. For testing, we use the validation sets of both DAVIS [43] and Videvo [30].

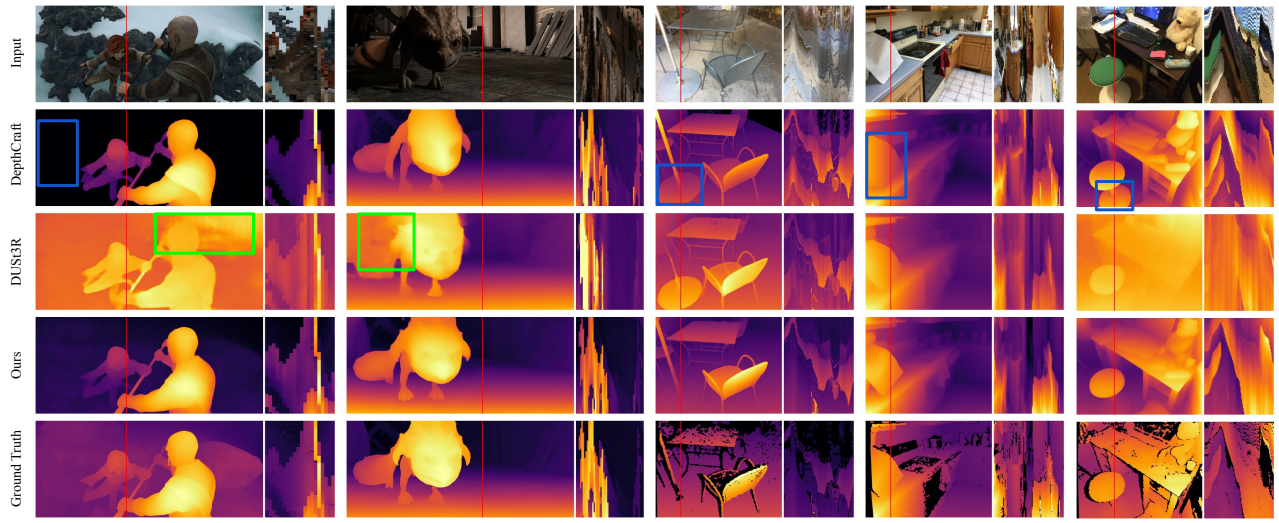

Figure 4. **Video depth prediction**, comparing Tracktention (+DepthAnything), DepthCrafter [20], and DUSt3R [56]. We visualize a column of pixels (highlighted in red) over time to illustrate temporal variation. Our model shows stable, coherent depth estimation over time, while DepthCrafter exhibits significant errors in certain regions (blue box). DUSt3R struggles with dynamic content (green box).

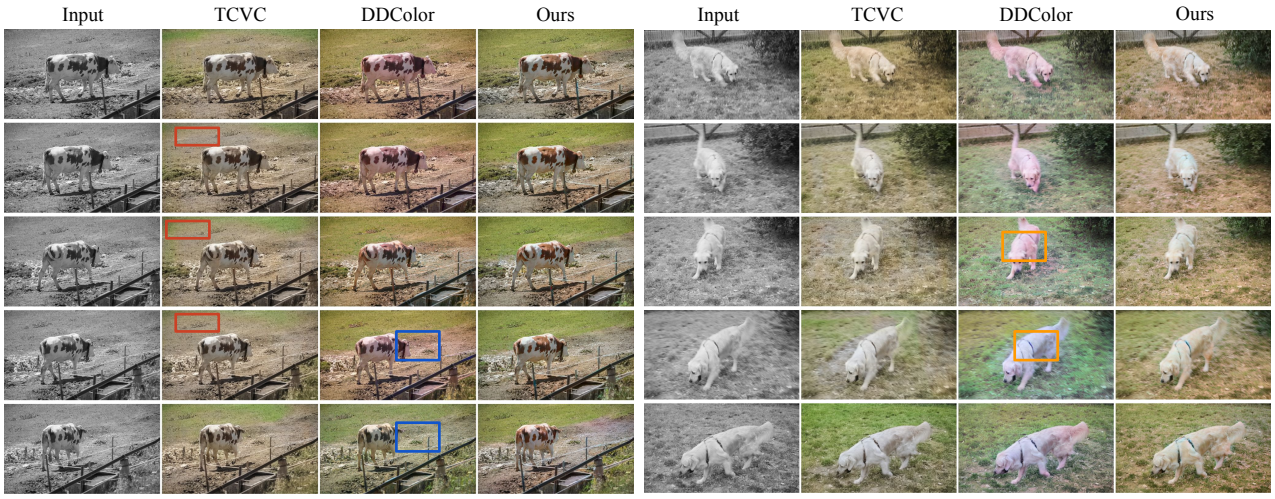

Figure 5. **Qualitative comparison of video colorization methods.** Tracktention (+DDColor) yields vibrant, realistic, and consistent colors. In contrast, TCVC appears less vibrant (21.7 vs. 29.5 Colorfulness), while DDColor lacks temporal consistency.

Evaluation metrics. We evaluate colorization results using standard metrics. Please refer to Appendix A.3.2 for details.

Quantitative Results. Figure 5 reports video colorization results on DAVIS and Videvo. We apply Tracktention to several image colorization models, consistently improving their performance when applied to videos, particularly in color consistency. Specifically, CDC improves by 22.8% on DAVIS and 28.3% on Videvo for CIC [66], 41.3% and 28.8% for Colorformer [21], and 46.5% and 45.7% for DDColor [22]. Tracktention also improves FID and Colorfulness, indicating overall improvements in visual quality. We also compare to native video colorization networks (top rows), often outperforming these as well.

Qualitative Results. In Fig. 5, our method delivers more vibrant, realistic, and consistent colors compared to TCVC and DDColor. TCVC's colors are less vibrant, with a colorfulness score of 21.7 compared to our 29.5, and DDColor exhibits inconsistencies across frames, with colors sometimes changing dramatically over time.

4.3. Ablation Studies

Here, we measure the impact of various design decisions in Tracktention using the AbsRel metric on depth prediction in Sintel, ScanNet, and Tanks and Temples.

Table 4a assesses the impact of the *number of tracks* used in *training*: even a single track per video can lead to improvements, and the optimal number is around 24^2 tracks. Table 4b assesses the *number of tracks* used at *testing* time, with similar results. Table 4c compares different tracking strategies: eschewing the tracker and simply setting tracks

Method	DAVIS (medium frame length)				Videvo (long frame length)			
	FID↓	CF↑	CDC↓	PSNR↑	FID↓	CF↑	CDC↓	PSNR↑
Specialized video colorization models:								
FAVC [31]	—	18.55	4.22	24.38	—	16.28	1.88	24.81
TCVC [34]	46.51	21.70	3.73	25.50	39.58	19.07	1.64	25.43
CIC [66]	44.23	30.34	6.18	**23.19**	39.78	29.19	3.59	22.51
[66]+Ours	**43.13**	**34.22**	**4.77** (-22.8%)	21.58	**37.00**	**32.73**	**2.57** (-28.3%)	**22.64**
IDC [67]	42.99	21.70	5.01	**25.42**	37.25	19.07	2.57	**25.35**
[67]+Ours	**33.47**	**28.10**	**4.45** (-11.1%)	23.51	**33.13**	**25.22**	**2.42** (-6.0%)	24.04
ColorFormer [21]	40.71	29.61	8.29	23.03	40.12	**29.76**	4.96	23.08
[21]+Ours	**29.73**	**32.24**	**4.86** (-41.3%)	**23.18**	**27.25**	29.65	**3.53** (-28.8%)	**23.31**
DDColor [22]	26.81	**30.61**	8.64	23.81	**20.27**	**30.38**	5.60	24.34
[22]+Ours	**24.61**	29.53	**4.62** (-46.5%)	**23.85**	22.78	26.06	**3.04** (-45.7%)	**24.39**

Table 3. **Quantitative comparison of video colorization methods on the DAVIS and Videvo datasets.** Our method, when augmented onto four different baseline models, consistently improves the Color Distribution Consistency (CDC) metric across both datasets.

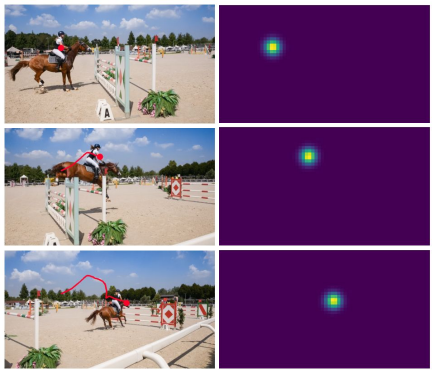

Figure 6. **Attentional Sampling module.** The module attends to image features in correspondence with each track, as shown by the attention maps next to each frame.

#tracks	Sintel↓	ScanNet↓	T&T↓
0 (base)	0.325	0.130	0.104
1	0.347	0.094	0.087
12^2	0.319	**0.086**	0.080
24^2	**0.295**	0.087	**0.073**

(a) **Number of training tracks**: Increasing training tracks consistently improved depth estimation accuracy, with optimal results at 24^2 tracks.

#tracks	Sintel↓	ScanNet↓	T&T↓
0 (base)	0.325	0.130	0.104
1	0.318	0.118	0.108
12^2	0.321	**0.087**	0.075
24^2	0.295	**0.087**	**0.073**

(b) **Number of testing tracks**: Increasing testing tracks significantly boosted performance, indicating better temporal capture with more tracks.

Type	Sintel↓	ScanNet↓	T&T↓
Constant	0.352	0.099	0.085
Grid, T_0	0.325	0.096	**0.073**
Rand., T_0	0.320	0.095	**0.073**
Rand., all	**0.295**	**0.087**	**0.073**

(c) **Type of tracks**: Using a point tracker performs better than using static pseudo-tracks. Furthermore, querying tracks uniformly is better.

Method	Sintel↓	ScanNet↓	T&T↓
3D Conv.	0.364	0.112	0.096
Time Attn.	0.363	0.097	0.081
Tracktention	**0.295**	**0.087**	**0.073**

(d) **Temporal modeling**: Tracktention outperformed 3D conv. and time attention by leveraging tracking data for better temporal alignment.

#Layers	Sintel↓	ScanNet↓	T&T↓
1	0.316	0.089	0.074
2	**0.295**	0.087	**0.073**
3	0.317	**0.083**	**0.073**

(e) **Number of Time Attention Layers**: Two layers balanced performance and complexity, while additional layers offered no consistent benefits.

#Layers	Sintel↓	ScanNet↓	T&T↓
1	0.344	0.090	0.087
6	**0.295**	**0.087**	**0.073**
12	0.333	0.105	0.075

(f) **Number of Tracktention layers**: 6 layers improved temporal representation, but further layers led to diminishing returns due to overfitting risks.

Table 4. **Ablations on video depth estimation results.** Each table examines the impact of different factors, such as the number and type of tracks used during training and testing. Results are reported using the AbsRel error metric, where ↓ indicates that lower values are better.

to be constant across all frames, and initializing the tracks on a grid in the first frame, at random in the first frame, or at random throughout the video. The latter results in much better performance, showing the importance of tracking and ensuring that tracks cover the video well. Table 4d tests replacing Tracktention Layers with 3D convolutions and time attention, which, by comparison, barely improve results. Table 4e shows the impact of increasing the *number of transformer layers* in the Track Transformer, where two layers are optimal in most cases. Finally, Tab. 4f shows the impact of adding a different *number of Tracktention layers*, concluding that six is about optimal.

4.4. Limitations

Tracktention's success depends on the quality of the extracted tracks. While modern point trackers are robust and efficient, they may still fail, particularly when tracks are off-camera for extended periods. Furthermore, running a point tracker in addition to the primary neural network adds complexity and computational cost (although this is often offset by avoiding spatio-temporal attention).

5. Conclusion

We have introduced Tracktention, a plug-and-play layer that upgrades image predictor networks into video predictor ones. It leverages the power of modern point trackers, transferring their understanding of motion to new applications such as depth prediction and colorization. The module is lightweight, simple to integrate, and compatible with a variety of architectures, including Vision Transformers and ConvNets. Thanks to the efficiency of recent point trackers like CoTracker3, it incurs only a small computational overhead. Remarkably, the resulting upgraded image models are *overall* faster and more accurate than some state-of-the-art methods specifically designed for video prediction.

Acknowledgments. This research was supported by ERC 101001212-UNION.

References

[1] Anurag Arnab, Mostafa Dehghani, Georg Heigold, Chen Sun, Mario Lučić, and Cordelia Schmid. Vivit: A video vision transformer. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 6836–6846, 2021. 2, 3

[2] 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. 14

[3] Gedas Bertasius, Heng Wang, and Lorenzo Torresani. Is space-time attention all you need for video understanding? In *ICML*, page 4, 2021. 2, 3

[4] Andreas Blattmann, Tim Dockhorn, Sumith Kulal, Daniel Mendelevitch, Maciej Kilian, Dominik Lorenz, Yam Levi, Zion English, Vikram Voleti, Adam Letts, et al. Stable video diffusion: Scaling latent video diffusion models to large datasets. *arXiv preprint arXiv:2311.15127*, 2023. 3

[5] Andreas Blattmann, Robin Rombach, Huan Ling, Tim Dockhorn, Seung Wook Kim, Sanja Fidler, and Karsten Kreis. Align your latents: High-resolution video synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 22563–22575, 2023. 3

[6] Joao Carreira and Andrew Zisserman. Quo vadis, action recognition? a new model and the kinetics dataset. In *proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6299–6308, 2017. 2

[7] Seokju Cho, Jiahui Huang, Jisu Nam, Honggyu An, Seungryong Kim, and Joon-Young Lee. Local all-pair correspondence for point tracking. *arXiv preprint arXiv:2407.15420*, 2024. 3

[8] Angela Dai, Angel X Chang, Manolis Savva, Maciej Halber, Thomas Funkhouser, and Matthias Nießner. Scannet: Richly-annotated 3d reconstructions of indoor scenes. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5828–5839, 2017. 14

[9] Mostafa Dehghani, Josip Djolonga, Basil Mustafa, Piotr Padlewski, Jonathan Heek, Justin Gilmer, Andreas Peter Steiner, Mathilde Caron, Robert Geirhos, Ibrahim Alabdulmohsin, et al. Scaling vision transformers to 22 billion parameters. In *International Conference on Machine Learning*, pages 7480–7512. PMLR, 2023. 4

[10] Carl Doersch, Ankush Gupta, Larisa Markeeva, Adria Recasens, Lucas Smaira, Yusuf Aytar, Joao Carreira, Andrew Zisserman, and Yi Yang. Tap-vid: A benchmark for tracking any point in a video. *Advances in Neural Information Processing Systems*, 35:13610–13626, 2022. 3

[11] Carl Doersch, Yi Yang, Mel Vecerik, Dilara Gokay, Ankush Gupta, Yusuf Aytar, Joao Carreira, and Andrew Zisserman. Tapir: Tracking any point with per-frame initialization and temporal refinement. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 10061–10072, 2023. 3

[12] Carl Doersch, Yi Yang, Mel Vecerik, Dilara Gokay, Ankush Gupta, Yusuf Aytar, Joao Carreira, and Andrew Zisserman. TAPIR: tracking any point with per-frame initialization and temporal refinement. In *Proc. CVPR*, 2023. 1

[13] Carl Doersch, Yi Yang, Dilara Gokay, Pauline Luc, Skanda Koppula, Ankush Gupta, Joseph Heyward, Ross Goroshin, João Carreira, and Andrew Zisserman. BootsTAP: Bootstrapped training for tracking-any-point. *arXiv*, 2402.00847, 2024. 1

[14] H Fan, B Xiong, K Mangalam, Y Li, Z Yan, and J Malik. . . . Multiscale vision transformers. In *Proc. CVPR*, 2021. 2

[15] Haoqi Fan, Bo Xiong, Karttikeya Mangalam, Yanghao Li, Zhicheng Yan, Jitendra Malik, and Christoph Feichtenhofer. Multiscale vision transformers. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 6824–6835, 2021. 3

[16] Christoph Feichtenhofer, Haoqi Fan, Jitendra Malik, and Kaiming He. Slowfast networks for video recognition. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 6202–6211, 2019. 2

[17] 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. 14

[18] Adam W Harley, Zhaoyuan Fang, and Katerina Fragkiadaki. Particle video revisited: Tracking through occlusions using point trajectories. In *European Conference on Computer Vision*, pages 59–75. Springer, 2022. 3

[19] Adam W. Harley, Zhaoyuan Fang, and Katerina Fragkiadaki. Particle videos revisited: Tracking through occlusions using point trajectories. In *Proc. ECCV*, 2022. 1

[20] Wenbo Hu, Xiangjun Gao, Xiaoyu Li, Sijie Zhao, Xiaodong Cun, Yong Zhang, Long Quan, and Ying Shan. Depthcrafter: Generating consistent long depth sequences for open-world videos. *arXiv preprint arXiv:2409.02095*, 2024. 5, 6, 7, 14

[21] Xiaozhong Ji, Boyuan Jiang, Donghao Luo, Guangpin Tao, Wenqing Chu, Zhifeng Xie, Chengjie Wang, and Ying Tai. Colorformer: Image colorization via color memory assisted hybrid-attention transformer. In *European Conference on Computer Vision*, pages 20–36. Springer, 2022. 6, 7, 8, 15

[22] Xiaoyang Kang, Tao Yang, Wenqi Ouyang, Peiran Ren, Lingzhi Li, and Xuansong Xie. Ddcolor: Towards photo-realistic image colorization via dual decoders. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 328–338, 2023. 6, 7, 8, 12, 15

[23] Nikita Karaev, Ignacio Rocco, Benjamin Graham, Natalia Neverova, Andrea Vedaldi, and Christian Rupprecht. Dynamicstereo: Consistent dynamic depth from stereo videos. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13229–13239, 2023. 14

[24] Nikita Karaev, Iurii Makarov, Jianyuan Wang, Natalia Neverova, Andrea Vedaldi, and Christian Rupprecht. CoTracker3: Simpler and better point tracking by pseudo-labelling real videos. *arxiv*, 2024. 5, 13

[25] Nikita Karaev, Ignacio Rocco, Benjamin Graham, Natalia Neverova, Andrea Vedaldi, and Christian Rupprecht. Co-

tracker: It is better to track together. In *Proc. ECCV*, 2024. 3

[26] Nikita Karaev, Ignacio Rocco, Ben Graham, Natalia Neverova, Andrea Vedaldi, and Christian Rupprecht. CoTracker: It is better to track together. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2024. 1

[27] Bingxin Ke, Anton Obukhov, Shengyu Huang, Nando Metzger, Rodrigo Caye Daudt, and Konrad Schindler. Repurposing diffusion-based image generators for monocular depth estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9492–9502, 2024. 6

[28] Arno Knapitsch, Jaesik Park, Qian-Yi Zhou, and Vladlen Koltun. Tanks and temples: Benchmarking large-scale scene reconstruction. *ACM Transactions on Graphics (ToG)*, 36 (4):1–13, 2017. 14

[29] Johannes Kopf, Xuejian Rong, and Jia-Bin Huang. Robust consistent video depth estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1611–1621, 2021. 3, 14

[30] Wei-Sheng Lai, Jia-Bin Huang, Oliver Wang, Eli Shechtman, Ersin Yumer, and Ming-Hsuan Yang. Learning blind video temporal consistency. In *Proceedings of the European conference on computer vision (ECCV)*, pages 170–185, 2018. 3, 6

[31] Chenyang Lei and Qifeng Chen. Fully automatic video colorization with self-regularization and diversity. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 3753–3761, 2019. 8, 15

[32] Hongyang Li, Hao Zhang, Shilong Liu, Zhaoyang Zeng, Tianhe Ren, Feng Li, and Lei Zhang. Taptr: Tracking any point with transformers as detection. In *European Conference on Computer Vision*, pages 57–75. Springer, 2025. 3

[33] Lu Ling, Yichen Sheng, Zhi Tu, Wentian Zhao, Cheng Xin, Kun Wan, Lantao Yu, Qianyu Guo, Zixun Yu, Yawen Lu, et al. Dl3dv-10k: A large-scale scene dataset for deep learning-based 3d vision. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 22160–22169, 2024. 14

[34] Yihao Liu, Hengyuan Zhao, Kelvin CK Chan, Xintao Wang, Chen Change Loy, Yu Qiao, and Chao Dong. Temporally consistent video colorization with deep feature propagation and self-regularization learning. *arXiv preprint arXiv:2110.04562*, 2021. 8, 15

[35] Ziwei Liu, Raymond A Yeh, Xiaoou Tang, Yiming Liu, and Aseem Agarwala. Video frame synthesis using deep voxel flow. In *Proceedings of the IEEE international conference on computer vision*, pages 4463–4471, 2017. 3

[36] Zhuang Liu, Hanzi Mao, Chao-Yuan Wu, Christoph Feichtenhofer, Trevor Darrell, and Saining Xie. A convnet for the 2020s. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11976–11986, 2022. 6, 15

[37] Ze Liu, Jia Ning, Yue Cao, Yixuan Wei, Zheng Zhang, Stephen Lin, and Han Hu. Video swin transformer. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 3202–3211, 2022. 3

[38] Xuan Luo, Jia-Bin Huang, Richard Szeliski, Kevin Matzen, and Johannes Kopf. Consistent video depth estimation. *ACM Transactions on Graphics (ToG)*, 39(4):71–1, 2020. 3, 14

[39] Nikolaus Mayer, Eddy Ilg, Philip Hausser, Philipp Fischer, Daniel Cremers, Alexey Dosovitskiy, and Thomas Brox. A large dataset to train convolutional networks for disparity, optical flow, and scene flow estimation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4040–4048, 2016. 14

[40] Daniel Neimark, Omri Bar, Maya Zohar, and Dotan Asselmann. Video transformer network. In *Proc. ICCV Workshops*, 2021. 2

[41] E. Palazzolo, J. Behley, P. Lottes, P. Giguère, and C. Stachniss. ReFusion: 3D Reconstruction in Dynamic Environments for RGB-D Cameras Exploiting Residuals. In *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2019. 14

[42] Mandela Patrick, Dylan Campbell, Yuki Markus Asano, Ishan Misra, Florian Metze, Christoph Feichtenhofer, Andrea Vedaldi, and João F. Henriques. Keeping your eye on the ball: Trajectory attention in video transformers. In *Proceedings of Advances in Neural Information Processing Systems (NeurIPS)*, 2021. 2, 3

[43] Federico Perazzi, Jordi Pont-Tuset, Brian McWilliams, Luc Van Gool, Markus Gross, and Alexander Sorkine-Hornung. A benchmark dataset and evaluation methodology for video object segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 724–732, 2016. 6

[44] Luigi Piccinelli, Yung-Hsu Yang, Christos Sakaridis, Mattia Segu, Siyuan Li, Luc Van Gool, and Fisher Yu. UniDepth: universal monocular metric depth estimation. In *Proc. CVPR*, 2024. 6

[45] Johannes Lutz Schönberger and Jan-Michael Frahm. Structure-from-motion revisited. In *Proc. CVPR*, 2016. 6

[46] Johannes Lutz Schönberger, Enliang Zheng, Marc Pollefeys, and Jan-Michael Frahm. Pixelwise view selection for unstructured multi-view stereo. In *Proc. ECCV*, 2016. 6

[47] Thomas Schops, Johannes L. Schonberger, Silvano Galliani, Torsten Sattler, Konrad Schindler, Marc Pollefeys, and Andreas Geiger. A multi-view stereo benchmark with high-resolution images and multi-camera videos. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 14

[48] Philipp Schröppel, Jan Bechtold, Artemij Amiranashvili, and Thomas Brox. A benchmark and a baseline for robust multi-view depth estimation. In *2022 International Conference on 3D Vision (3DV)*, pages 637–645. IEEE, 2022. 6, 14

[49] Jiahao Shao, Yuanbo Yang, Hongyu Zhou, Youmin Zhang, Yujun Shen, Matteo Poggi, and Yiyi Liao. Learning temporally consistent video depth from video diffusion priors. *arXiv*, 2406.01493, 2024. 6, 14

[50] Uriel Singer, Adam Polyak, Thomas Hayes, Xi Yin, Jie An, Songyang Zhang, Qiyuan Hu, Harry Yang, Oron Ashual, Oran Gafni, et al. Make-a-video: Text-to-video generation without text-video data. *arXiv preprint arXiv:2209.14792*, 2022. 3

[51] Jianlin Su, Yu Lu, Shengfeng Pan, Bo Wen, and Yunfeng Liu. Roformer: Enhanced transformer with rotary position embedding, 2021. 4

[52] Zachary Teed and Jia Deng. Raft: Recurrent all-pairs field transforms for optical flow. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part II 16*, pages 402–419. Springer, 2020. 3

[53] Du Tran, Lubomir Bourdev, Rob Fergus, Lorenzo Torresani, and Manohar Paluri. Learning spatiotemporal features with 3d convolutional networks. In *Proceedings of the IEEE international conference on computer vision*, pages 4489–4497, 2015. 2

[54] Liao Wang, Qiang Hu, Qihan He, Ziyu Wang, Jingyi Yu, Tinne Tuytelaars, Lan Xu, and Minye Wu. Neural residual radiance fields for streamably free-viewpoint videos. In *Proc. CVPR*, 2023. 6

[55] Qianqian Wang, Yen-Yu Chang, Ruojin Cai, Zhengqi Li, Bharath Hariharan, Aleksander Holynski, and Noah Snavely. Tracking everything everywhere all at once. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 19795–19806, 2023. 3

[56] Shuzhe Wang, Vincent Leroy, Yohann Cabon, Boris Chidlovskii, and Jerome Revaud. DUSt3R: Geometric 3D vision made easy. In *Proc. CVPR*, 2024. 6, 7

[57] Wenshan Wang, Delong Zhu, Xiangwei Wang, Yaoyu Hu, Yuheng Qiu, Chen Wang, Yafei Hu, Ashish Kapoor, and Sebastian Scherer. Tartanair: A dataset to push the limits of visual slam. In *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 4909–4916. IEEE, 2020. 14

[58] Xiaolong Wang, Ross B. Girshick, Abhinav Gupta, and Kaiming He. Non-local neural networks. In *Proc. CVPR*, 2018. 2

[59] Yiran Wang, Min Shi, Jiaqi Li, Zihao Huang, Zhiguo Cao, Jianming Zhang, Ke Xian, and Guosheng Lin. Neural video depth stabilizer. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9466–9476, 2023. 6, 14

[60] Linjie Yang, Yuchen Fan, and Ning Xu. Video instance segmentation. In *ICCV*, 2019. 6, 15

[61] Lihe Yang, Bingyi Kang, Zilong Huang, Xiaogang Xu, Jiashi Feng, and Hengshuang Zhao. Depth anything: Unleashing the power of large-scale unlabeled data. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10371–10381, 2024. 5

[62] Lihe Yang, Bingyi Kang, Zilong Huang, Xiaogang Xu, Jiashi Feng, and Hengshuang Zhao. Depth anything: Unleashing the power of large-scale unlabeled data. In *Proc. CVPR*, 2024. 5, 6, 12, 14

[63] Lihe Yang, Bingyi Kang, Zilong Huang, Zhen Zhao, Xiaogang Xu, Jiashi Feng, and Hengshuang Zhao. Depth anything V2. *arXiv*, 2406.09414, 2024. 6

[64] Yao Yao, Zixin Luo, Shiwei Li, Tian Fang, and Long Quan. Mvsnet: Depth inference for unstructured multi-view stereo. In *Proceedings of the European conference on computer vision (ECCV)*, pages 767–783, 2018. 14

[65] Chandan Yeshwanth, Yueh-Cheng Liu, Matthias Nießner, and Angela Dai. Scannet++: A high-fidelity dataset of 3d indoor scenes. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 12–22, 2023. 14

[66] Richard Zhang, Phillip Isola, and Alexei A Efros. Colorful image colorization. In *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part III 14*, pages 649–666. Springer, 2016. 6, 7, 8, 15

[67] Richard Zhang, Jun-Yan Zhu, Phillip Isola, Xinyang Geng, Angela S Lin, Tianhe Yu, and Alexei A Efros. Real-time user-guided image colorization with learned deep priors. *arXiv preprint arXiv:1705.02999*, 2017. 6, 8, 15

[68] 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 *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 19855–19865, 2023. 14

[69] Xizhou Zhu, Yuwen Xiong, Jifeng Dai, Lu Yuan, and Yichen Wei. Deep feature flow for video recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2349–2358, 2017. 3

Tracktention: Leveraging Point Tracking to Attend Videos Faster and Better

Supplementary Material

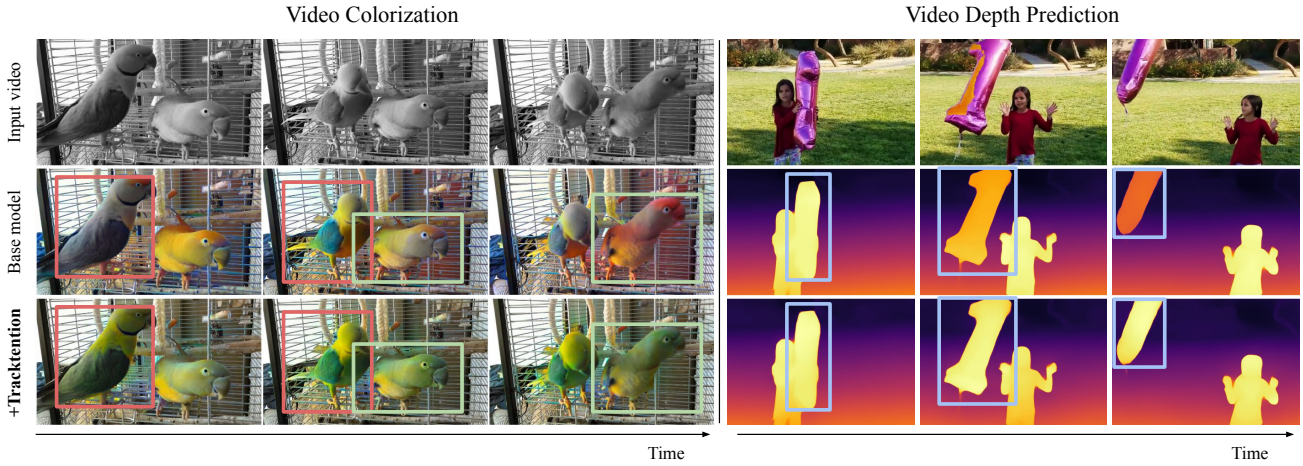

Figure 7. **Stabilization effect of our model on video prediction tasks:** *Left (Video Colorization)*: The first row shows input grayscale video frames. The second row demonstrates the output of a frame-by-frame base model [22], producing inconsistent colors across frames (*e.g.*, varying hues in pink and green boxes). The third row highlights our proposed Tracktention (+base model [22]), achieving consistent and stable colorization across frames. *Right (Video Depth Prediction)*: The first row displays input video frames. The second row shows the depth predictions from a base model [62], which suffers from temporal instability. The third row presents Tracktention's (+base model [62]) depth predictions, offering consistent and stable outputs over time.

A. Design and Implementation Details

A.1. Query Initialization and Point Tracking

In Tracktention, we use randomly initialized queries for the point tracking model. In Figure 8, we compare the results of two different query initialization strategies for object tracking in video: grid-initialized queries and random-initialized queries that we use. Queries (represented as larger dots with white edges) are seed coordinates in the video's spatio-temporal space that are used to start the tracking process. Effective query initialization is crucial for maintaining complete and consistent coverage of objects throughout the video frames.

The top row illustrates the tracks obtained using a grid sampling strategy, where a uniform grid of queries is placed over the spatial dimensions of the first frame. While this method provides good initial coverage, it results in significant gaps in later frames due to motion and occlusion, leading to many areas being left untracked as the video progresses.

In contrast, the bottom row demonstrates tracks produced by our proposed random sampling method, where queries are initialized randomly in the spatio-temporal space. This approach results in more robust and complete tracking, as seen in the wider spatial distribution of tracks in the later frames.

This enhanced tracking coverage is advantageous for our tracking-based attention model, Tracktention, which depends on the completeness and density of query tracks to deliver comprehensive attention across the scene. By ensuring consistent and uniform tracking throughout the video, our random query initialization method enables Tracktention to more effectively focus on and process critical regions of interest, even in complex and dynamic scenarios. This leads to improved performance and robustness, as demonstrated in the Ablation Study presented in the main paper.

A.2. Tracktention compared with Standard Attention Mechanisms

In Figure 9, we provide a conceptual overview comparing Tracktention to standard attention mechanisms for video processing. **Spatial-temporal attention** attends to all tokens across space and time, capturing comprehensive relationships but at a prohibitive computational cost. **Spatial attention** focuses only on spatial tokens within individual frames, ignoring temporal dependencies, while **temporal attention** processes temporal evolution at fixed spatial locations and fails when objects move across patches. While some methods combine spatial and temporal attention, attending to a different location in another frame requires traversing intermediate tokens implicitly, leading to indirect and inefficient attention pathways. These limitations

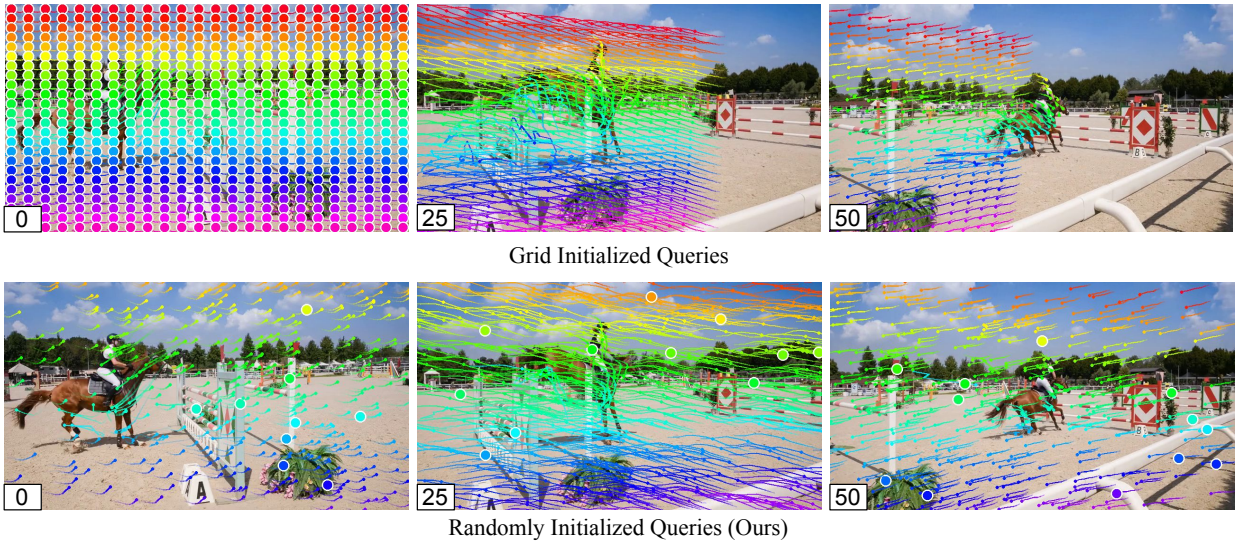

Figure 8. **Comparison of query initialization strategies for point tracking.** The top row shows the point tracks obtained with grid-initialized queries, which suffer from significant coverage loss in later frames. In contrast, the bottom row illustrates tracks obtained with our random initialization method, which maintains comprehensive coverage across the scene over time. Larger dots with white edges represent queries, which are seed coordinates used to initiate tracking in the video. Numbers at the bottom left of each frame indicate the frame index, highlighting the improved completeness of tracks produced by our approach, particularly in later frames.

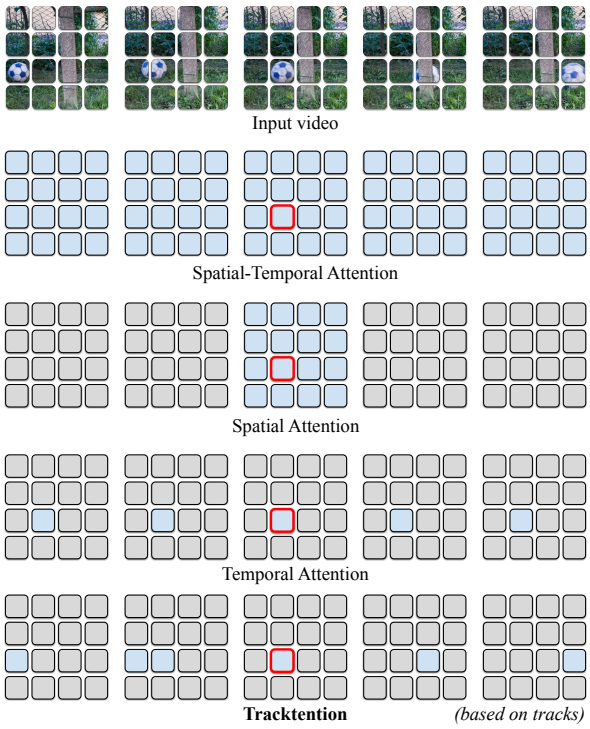

Figure 9. **Comparison of different attention mechanisms for video processing.** The red-bordered block represents the query token, while the blue-highlighted blocks indicate the range of tokens the mechanism attends to. See text for details.

hinder their ability to efficiently and effectively model dynamic video content.

Our proposed Tracktention addresses these challenges by leveraging point tracks to attend selectively to spatial-temporal tokens relevant to a query's trajectory. Through our Attentional Sampling module, Tracktention handles cases where objects span multiple patches (*e.g.*, the second frame), ensuring fragmented representations do not disrupt attention. Additionally, it is robust to occlusion (*e.g.*, the fourth frame) by using a tracker [24] designed to handle occlusions. By combining computational efficiency, adaptability to motion, and robustness to occlusion, Tracktention significantly improves video processing in complex scenarios.

A.2.1 Complexity Analysis

We analyze the computational complexity of our proposed Tracktention layer in comparison to standard attention mechanisms used in video processing. Let H and W denote the height and width of the frame features, T the number of frames (temporal dimension), and N the number of point tracks used in Tracktention, where $N < HW$ and $N \ll HWT$.

Spatio-temporal Attention: Spatio-temporal attention attends to all tokens across both space and time, capturing comprehensive relationships but at a prohibitive computational cost: $O((HWT)^2) = O(HWT \cdot HWT)$.

This quadratic complexity over the total number of tokens HWT makes it computationally infeasible for larger

videos or higher resolutions.

Spatial Attention: Spatial attention operates on each frame independently, attending to all spatial tokens within a frame. The computational complexity for processing all frames is: $O(T \cdot (HW)^2) = O(HWT \cdot HW)$.

This is because, for each of the T frames, attention computations involve all pairs of spatial tokens, resulting in a quadratic complexity with respect to the number of spatial tokens HW.

Temporal Attention: Temporal attention focuses on the temporal evolution at fixed spatial locations. Each spatial position attends over the temporal dimension. The computational complexity is: $O(HW \cdot T^2) = O(HWT \cdot T)$

Here, each of the HW spatial positions computes attention over all T temporal tokens at that position, leading to quadratic complexity with respect to the temporal length T.

Tracktention Complexity: Our proposed Tracktention leverages point tracks to attend selectively to relevant spatial-temporal tokens along a query's trajectory. The computational complexity of Tracktention is: $O(HWT \cdot N + T^2 \cdot N)$

The term $O(HWT \cdot N)$ corresponds to the *Attentional Sampling* and *Attentional Splatting* process, where features are sampled along the point tracks from the video tokens.

The term $O(T^2 \cdot N)$ arises from the *Temporal Transformer* operating over the point tracks, computing attention across time for each track.

Complexity Comparison: Since N is significantly smaller than the total number of tokens ($N \ll HWT$). The complexity of Tracktention is substantially lower than that of spatio-temporal attention mechanisms.

Factorized spatial and temporal attention achieves better efficiency by separating spatial ($O\left(T \times (HW)^2\right)$) and temporal ($O\left(HW \times T^2\right)$) attention, resulting in a combined complexity of $O(HWT \cdot (HW + T))$. As $N < (HW + T)$, Tracktention achieves lower complexity by attending selectively to N point tracks, yielding an more efficient tool for video processing.

Furthermore, we note that the attentional sampling and splatting processes realistically focus on a local patch around each query. This allows us to further reduce the complexity of these processes from $O\left(HWT \times N\right)$ to $O\left(P^2 \times T \times N\right)$, where P is a local patch size ($P^2 \ll HW$). We leave this sparsity optimization for future work.

A.3. Other Implementation Details

A.3.1 Video Depth Estimation Evaluation

Data and evaluation metric. For training, we use a combination of datasets containing both synthetic and real videos: ARKitScenes [2], ScanNet++ [65], TartanAir [57], PointOdyssey [68], DynamicReplica [23], and DL3DV [33], totaling 12,947 videos. All videos are resized to have a short side of 336 pixels for computational efficiency. For evaluation, we use two benchmarks: RobustMVD [48], containing short clips of 8 frames with large motion, and the longer benchmark videos from DepthCrafter [20]. These evaluation datasets include a wide variety of scenes from KITTI [17], ScanNet [8], DTU [64], Tanks and Temples [28], ETH3D [47], Sintel [39], and Bonn RGB-D [41].

We use standard depth estimation metrics [62]: *Absolute Relative Difference* (AbsRel) — calculated as $|\hat{d} - d|/d$, where $\hat{d}$ is the estimated depth and d is the true depth — and *Threshold Accuracy* (δ_τ), the percentage of pixels satisfying $\max(\frac{d}{\hat{d}}, \frac{\hat{d}}{d}) < \tau$, with τ set per the original benchmark. As in training, we calibrate each prediction to the ground truth using a scale and shift factor, shared across all frames in a video, before assessing a metric.

Scale and shift ambiguity Evaluating depth estimation in video sequences presents unique challenges due to the scale ambiguity inherent in monocular depth prediction, where predicted depths may differ from ground truth by a global scale and shift. Traditional methods [29, 38, 49, 59] often employ frame-wise evaluation, fitting a separate scale and shift for each frame independently:

$$\min_{s_i, t_i} \sum_{(x,y)} \left(D_{2,i}(x,y) - (s_i \cdot D_{1,i}(x,y) + t_i)\right)^2,$$

where $D_{1,i}$ is the predicted depth, $D_{2,i}$ is the ground-truth depth for frame i, and s_i, t_i are the scale and shift for that frame. While this approach minimizes per-frame error, it can mask significant temporal inconsistencies, as it allows scale and shift to vary freely between frames, leading to flickering or unstable depth predictions in videos.

To address this issue, we adopt a video-wise evaluation method similar to DepthCrafter [20]. This approach enforces a single global scale and shift across the entire video sequence:

$$\min_{s,t} \sum_{i} \sum_{(x,y)} \left(D_{2,i}(x,y) - (s \cdot D_{1,i}(x,y) + t)\right)^2.$$

By using consistent scaling factors s and t for all frames, the video-wise evaluation penalizes variations in predicted depth over time, providing a more accurate assessment of temporal consistency. This method highlights the model's ability to maintain stable and coherent depth predictions across frames, which is crucial for applications requiring consistent video outputs.

A.3.2 Implementation Details for Video Colorization

Evaluation metrics. We assess the quality of the colorization using standard metrics: the *Fréchet Inception Distance (FID)*, which evaluates how well the predicted colorization

matches the ground truth statistically in feature space; the *Colorfulness Score (CF)*, which quantifies color vibrancy; and the *Color Distribution Consistency (CDC)*, which measures the temporal consistency of the colorization. We also include the Peak Signal-to-Noise Ratio (PSNR), though it is generally acknowledged that this is a poor metric for evaluating colorization accurately [22].

Integration implementation We evaluate the effectiveness of the Tracktention layer in enhancing temporal consistency by integrating it into four image colorization models: CIC [66], IDC [67], ColorFormer [21], and DDColor [22].

Integration with CIC and IDC For the ConvNet-based architectures CIC and IDC, which feature encoder-decoder structures with downsampling, standard, and upsampling convolutional layers, we insert the Tracktention layer after the standard convolutional layers 5, 6, and 7. This integration allows temporal alignment at multiple stages of feature extraction, enhancing consistency across video frames.

Integration with ColorFormer and DDColor In ColorFormer, which utilizes a transformer backbone followed by a 4-layer U-Net decoder with residual connections, we integrate the Tracktention layer after the first three layers of the transformer backbone. Similarly, for DDColor, which employs a ConvNeXt [36] backbone with a U-Net decoder, we add the Tracktention layer after the first three layers of the backbone.

Training Details All models are trained using the loss functions from DDColor with the AdamW optimizer (initial learning rate 1.6×10^{-5}). The learning rate is decayed using a MultiStepLR scheduler every 4k iterations starting from the 8k-th iteration, over a total of 40k iterations. Training is conducted on the YouTube-VIS [60] dataset using raw, unlabeled video data, with frames resized to 256×256 pixels.

We do not employ temporal consistency losses such as flow warping losses used in prior works [31, 34]. By avoiding the computation of optical flow and associated warping operations, our training process remains efficient while achieving temporal consistency through the Tracktention layer integration. This shows that the Tracktention layer can effectively enhance temporal consistency in video colorization without relying on additional temporal loss functions, highlighting its flexibility and practicality.

B. Additional Experimental Results

Here, we present additional experimental results and visual comparisons showing how the Tracktention layer enhances video consistency, stability, and accuracy, along with an analysis of input tracks' influence.

B.1. Influence of Input Tracks on Video Consistency

We analyzed how the selective use of specific tracks affects the temporal consistency of video colorization. By selectively activating tracks corresponding to specific objects or regions, Tracktention guides its attention mechanism to maintain stable and consistent colorization for those targeted areas across video frames, while areas without active tracks may exhibit color inconsistencies. Figure 10 illustrates this behavior. In each case, activating tracks for a particular object (e.g., one bird, the front of the train, or the dog) results in consistent coloring for that object, while other objects without active tracks (e.g., the other bird, the back of the train, or the ball) show inconsistent coloring. This demonstrates that Tracktention can leverage selective tracks to ensure localized stability in the colorization process, even when other regions of the frame are affected.

B.2. Additional Visual Examples

Video Depth Estimation Figures 11 and 12 present additional visual comparisons for video depth estimation. Similar to the results in the main paper, our model produces stable and accurate depth maps across all frames. The incorporation of the Tracktention layer enables precise temporal alignment of features, resulting in consistent and accurate depth estimation over time.

The state-of-the-art method, DepthCrafter exhibits significant errors in certain regions, particularly where complex motion or occlusions occur. DUSt3R, which relies on implicit triangulation, struggles with dynamic content, leading to inaccuracies and temporal inconsistencies in the estimated depth maps.

Our base model, Depth Anything, is an image-based depth estimation model that processes frames independently. As a result, it shows inconsistent depth estimation across frames, with noticeable instability in the depth maps. By integrating the Tracktention layer into Depth Anything, we enhance temporal consistency, achieving results that are both accurate and stable throughout the video sequence.

Automatic Video Colorization Figures 13 and 14 provide more results on automatic video colorization. Consistent with observations in the main paper, the baseline method TCVC is unable to produce vibrant colors, resulting in desaturated and less realistic outputs.

Our base model, DDColor, performs frame-by-frame colorization, which leads to unstable and inconsistent color results. The absence of temporal coherence causes color flickering and discrepancies between frames, detracting from the overall visual quality of the video.

When we augment DDColor with our Tracktention layer, we observe a clear improvement in temporal consistency while retaining the original color vibrancy and realism. The Tracktention layer allows the model to attend to corresponding areas across time based on point tracks, ensuring smooth and coherent colorization throughout the video.

B.3. Video Demonstrations

We also provide a supplementary video that explains our method and showcases the resulting video outputs. The video includes side-by-side comparisons of the base models before and after augmented with our Tracktention layer for both video depth estimation and automatic video colorization tasks. It highlights the temporal consistency and accuracy achieved by integrating the Tracktention layer into existing models. Please find the enclosed `supplementary_video.mp4` file for a visual demonstration of our method's performance.

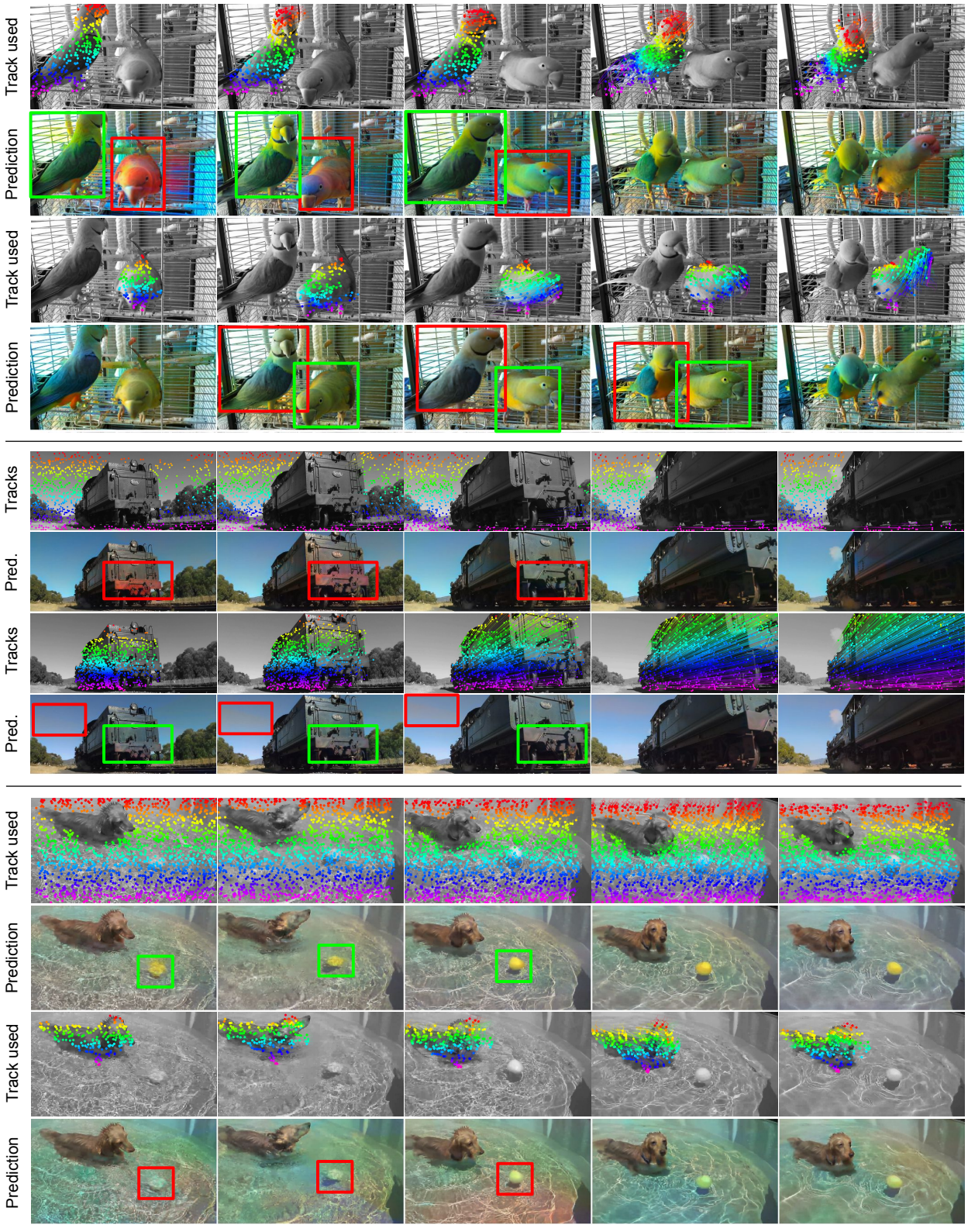

Figure 10. **The impact of selective tracks used on video colorization consistency.** In the top example, using tracks corresponding to the left bird (Row 1) results in consistent colorization of the left bird (Row 2, green box), while the right bird exhibits inconsistent colorization (red box). Conversely, using tracks for the right bird (Row 3) ensures stable colorization for the right bird (Row 4, green box), while the left bird becomes inconsistent (red box). Similar patterns are observed for the train and dog examples. Green boxes highlight regions with stable and consistent colors, while red boxes indicate inconsistent colorization.

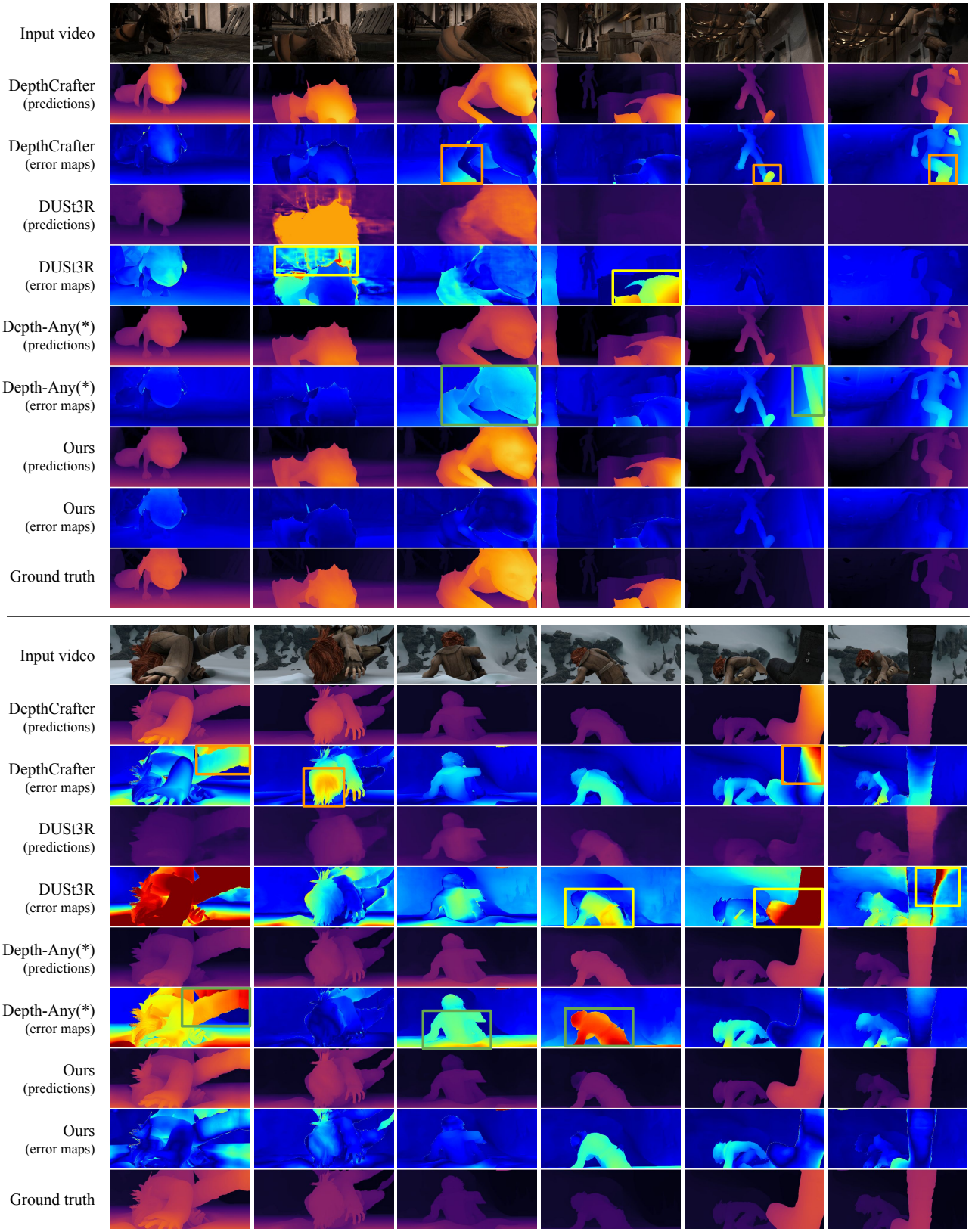

Figure 11. **Comparison of depth prediction results across different models:** DepthCrafter, DUSt3R, Depth-Anything (marked with * as the base model), and Ours (+Depth-Anything). Each row shows depth predictions, corresponding error maps, and the ground truth for the input video frames. Highlighted rectangles emphasize key issues in baseline methods: DepthCrafter exhibits significant errors in certain areas, DUSt3R tends to fail in dynamic regions, and Depth-Anything produces flickering results, evident from inconsistent error patterns.

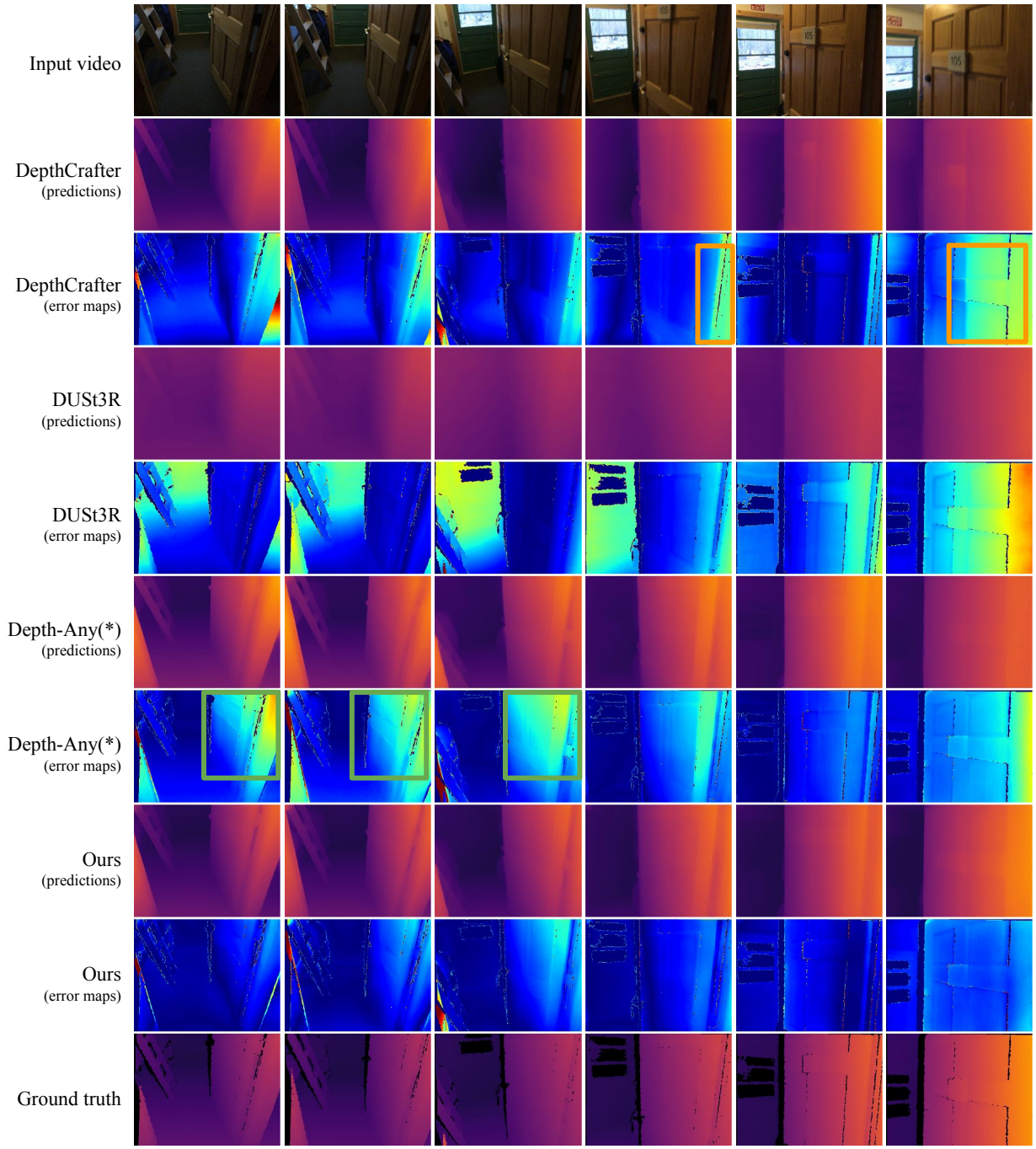

Figure 12. **Additional comparison of depth prediction results** across DepthCrafter, DUSt3R, Depth-Anything (*denotes the base model), and Ours (+Depth-Anything). The rows present depth predictions, their error maps, and ground truth for a different set of input video frames, illustrating consistency across varying scenes.

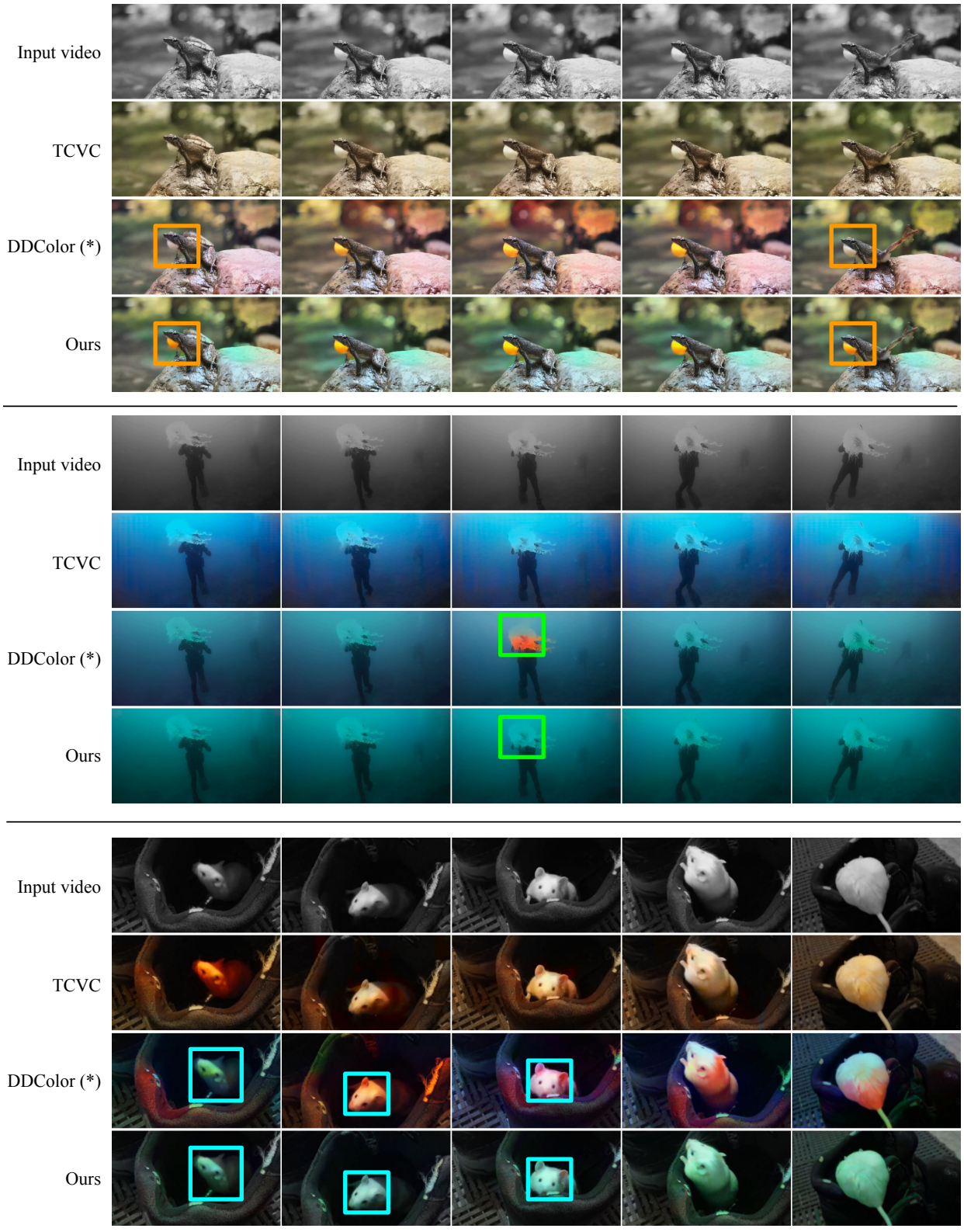

Figure 13. **Video colorization results** comparing TCVC, DDColor (*denoting the base model), and Ours (+DDColor). The rows show the input grayscale video frames, followed by the colorized outputs from each method. Highlighted areas indicate inconsistencies in the base model (DDColor), which are resolved by our model, demonstrating its ability to produce consistent and accurate colorization.

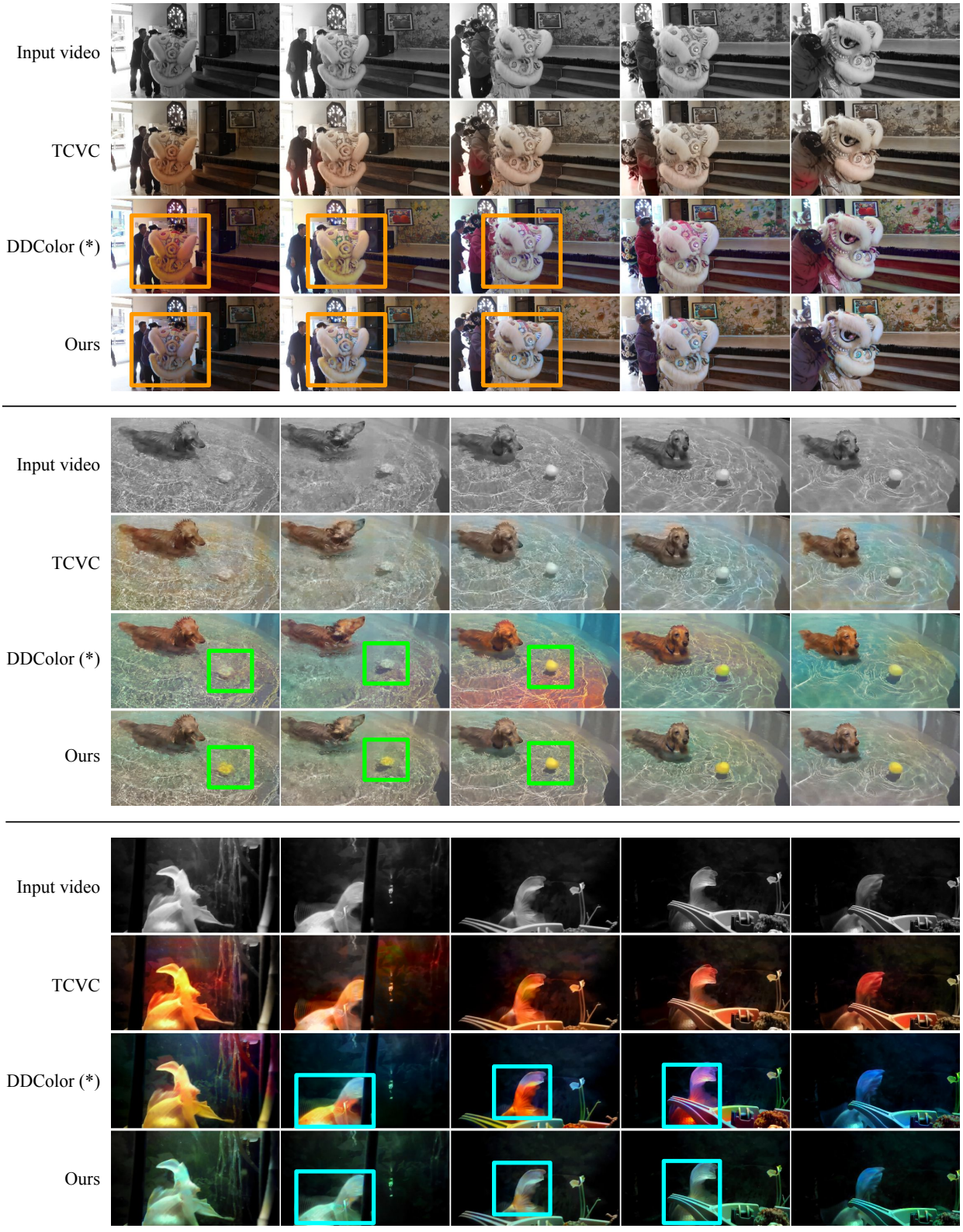

Figure 14. **Additional video colorization results** comparing TCVC, DDColor (*denoting the base model), and Ours (+DDColor). The rows display the input grayscale video frames alongside the colorized outputs from each method. Highlighted areas pinpoint inconsistencies in the base model (DDColor), which are effectively resolved by our model, showcasing its improved consistency and color accuracy.