

SUPPLEMENTARY MATERIALS

Anonymous authors

Paper under double-blind review

A APPENDIX

A.1 LIMITATIONS

As shown in Fig. 11, our method has several limitations that require further exploration. Currently, it depends on external techniques such as Karaev et al. (2023); Yang et al. (2023) for trajectory extraction. Investigating how to generate trajectories from more flexible inputs, such as text-based conditions, is a promising direction for future research. Moreover, our method depends on the generative capabilities of the underlying foundation models. If these models face difficulties handling rapid motions, the effectiveness of trajectory-based control diminishes. Additionally, our method encounters challenges with 3D cycle consistency. For instance, performing 360-degree rotations would require further design adaptations. Furthermore, the control precision diminishes when the trajectories become too sparse since the auxiliary branch does not provide enough control information.

A.2 DETAILS OF FULL-ATTENTION IMPLEMENTATION

We implement trajectory attention in Open-Sora-Plan Lab & etc. (2024), a DiT (Peebles & Xie, 2023) model with 3D attention. The trajectory attention is constructed and trained following the same procedure outlined in the main paper, with the key difference being that it is appended to the 3D full attention block instead of the temporal attention block (Fig. 1). For other training details, we follow the setting in Lab & etc. (2024).

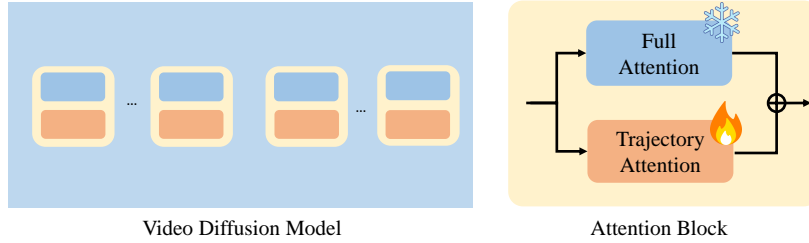


Figure 1: **Pipe for video diffusion models with 3D Attention.** The key distinction with the pipeline in the main paper lies in applying trajectory attention to the 3D attention module, rather than to the temporal attention mechanism.

A.3 DETAILS OF TASK PROCESS

Camera Parameters. The intrinsic parameters describe the internal characteristics of the camera. These parameters define how the camera transforms 3D points in its coordinate system to 2D points on the image plane. The matrix $\mathbf{K} \in \mathbb{R}^{3 \times 3}$ typically has the following structure:

$$\mathbf{K} = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \quad (1)$$

- f_x and f_y : Focal lengths in the x and y directions, often in pixel units.

- c_x and c_y : Coordinates of the principal point (optical center) in the image plane.

The intrinsic matrix \mathbf{K} encapsulates how pixel coordinates relate to normalized image coordinates. The extrinsic parameters define the camera’s position and orientation in the world coordinate system. This involves:

- **Rotation matrix $\mathbf{R} \in \mathbb{R}^{3 \times 3}$:** Represents the camera’s orientation by rotating the world coordinate system to align with the camera coordinate system.
- **Translation vector $\mathbf{t} \in \mathbb{R}^{3 \times 1}$:** Represents the position of the camera in the world coordinate system.

These two components are combined to form the extrinsic matrix, which can be represented as: $\mathbf{E} = [\mathbf{R} \mid \mathbf{t}]$. Here, \mathbf{R} defines how the 3D space is rotated relative to the camera, while \mathbf{t} indicates the displacement of the camera’s origin from the world coordinate origin. For a random input image, we predefine the intrinsic parameters and use the given camera trajectory to generate the extrinsic parameters.

Effects of intrinsic parameters. Since we cannot precisely estimate the intrinsic and extrinsic parameters from a single image, we use predefined intrinsic parameters and some hyperparameters for extrinsic parameters. From our observations, these predefined parameters with statistics can effectively generate reasonable results. We can also adjust them accordingly. Specifically, we set c_x and c_y in the center of the image plane, and f_x and f_y to 260 (relative). As shown in Fig. 12, we illustrate cases with different focal lengths.

Depth estimation. We use DepthAnythingV2Yang et al. (2024) to estimate the depth maps from frames. Examples of estimated results are shown in Fig. 2

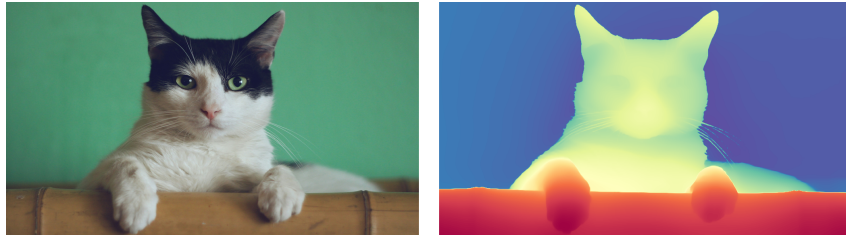


Figure 2: Depth estimation results.

Translation computing. Based on the depth map, camera parameters of two views, we can get the translation of pixels shown in Alg. 1.

Algorithm 1: Compute Pixel Translation

Input: $\mathbf{D} \in \mathbb{R}^{H_p \times W_p}$: Depth map of the first view
 $\mathbf{E}_1 \in \mathbb{R}^{4 \times 4}$: Extrinsic matrix of the first view
 $\mathbf{E}_2 \in \mathbb{R}^{4 \times 4}$: Extrinsic matrix of the second view
 $\mathbf{K} \in \mathbb{R}^{3 \times 3}$: Intrinsic matrix for both views
Output: $\mathbf{T}_{12} \in \mathbb{R}^{H_p \times W_p \times 2}$: Transformed pixel coordinates between views

```

1  $\mathbf{T} \leftarrow \mathbf{E}_2 \cdot \mathbf{E}_1^{-1}$ ; // Compute relative transformation
2  $\mathbf{y} \leftarrow [0, \dots, H_p - 1], \mathbf{x} \leftarrow [0, \dots, W_p - 1]$ 
3  $\mathbf{X}, \mathbf{Y} \leftarrow \text{meshgrid}(\mathbf{x}, \mathbf{y})$ 
4  $\mathbf{P}_{\text{homo}} \leftarrow \text{stack}([\mathbf{X}, \mathbf{Y}, \mathbf{1}_{H_p \times W_p}])$ ; // Homogeneous pixel positions
5  $\tilde{\mathbf{P}} \leftarrow \mathbf{K}^{-1} \cdot \mathbf{P}_{\text{homo}}$ ; // Normalized positions in camera space
6  $\mathbf{D}_{\text{reshaped}} \leftarrow \mathbf{D}$  reshaped to  $(H_p, W_p, 1, 1)$ 
7  $\mathbf{P}_{\text{world}} \leftarrow \mathbf{D}_{\text{reshaped}} \cdot \tilde{\mathbf{P}}$ 
8  $\mathbf{P}_{\text{world, homo}} \leftarrow \text{concatenate}([\mathbf{P}_{\text{world}}, \mathbf{1}_{H_p \times W_p}])$ 
9  $\mathbf{P}'_{\text{world, homo}} \leftarrow \mathbf{T} \cdot \mathbf{P}_{\text{world, homo}}$ 
10  $\mathbf{P}'_{\text{world}} \leftarrow \mathbf{P}'_{\text{world, homo}}[:, :, : 3]$ 
11  $\mathbf{T}_{12} \leftarrow \mathbf{K} \cdot \mathbf{P}'_{\text{world}}$ 
12 return  $\mathbf{T}_{12}$ 

```

Point trajectory extraction from videos. We use CoTracker Karaev et al. (2023) to extract point trajectories. An example is shown in Fig. 3.

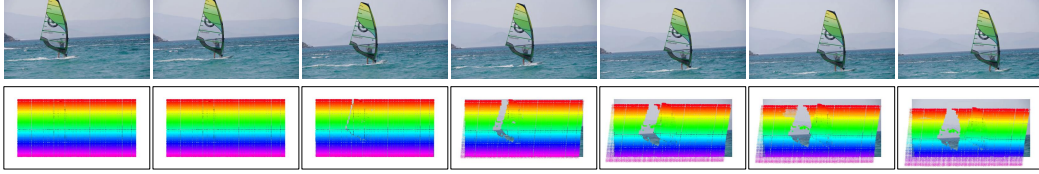


Figure 3: Point trajectory estimation results.

A.4 INPUT PROCESS.

In Alg. 4 of the main paper, we present the process of combining the video point trajectories and camera motion trajectories. In Sec 4.3, we present the input process of video editing. To clarify this process, we further provide the visualization in Fig. 4

A.5 ABLATIONS ON TRAINING DATASETS.

We initially selected 10k short real-world videos from Miradata Ju et al. (2024). To evaluate the impact of training domain diversity and dataset size, we conducted ablation experiments on various training datasets. As shown in Table 1, we did not observe substantial improvements when varying the training domains or increasing the dataset size. The evaluation setup matches the 25-frame version described in Table 1.

A.6 USING SYNTHETIC OPTICAL FLOW AS GUIDANCE

Our method directly leverages optical flow to guide the generation process. Unlike previous approaches Geng & Owens (2024) requiring inference-time optimization, our method seamlessly integrates this guidance into the attention mechanism. In addition to generating intermediate frames by interpolating the optical flow, our approach achieves improved consistency, as demonstrated in Fig. 5.

Table 1: Ablations on training datasets.

Dataset Setting	Training steps	ATE (m, ↓)	RPE trans (m, ↓)	RPE Rot (deg, ↓)	FID (↓)
10k real-world	40k	0.0396	0.0232	0.1939	103.5
10k games	40k	0.0421	0.0211	0.2139	105.3
10k real-world +10k games	40k	0.0372	0.0233	0.1899	102.2

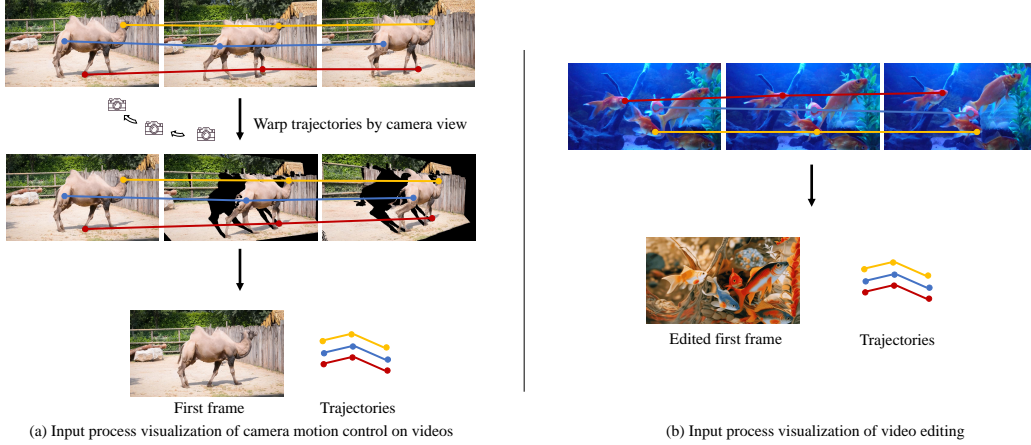


Figure 4: **Input process visualization.** For all tasks, the inputs to the network are the first frame and the extracted trajectories. The usage of the first frame and the trajectories are identical to Fig. 3 in the main paper. The wrapped frames and the reference frames will not be used as inputs to the generation network.

A.7 MORE CHALLENGING CASES.

As shown in Fig. 6, our method can also cover many challenging situations, like (a) video editing with multiple objects, (b) video editing with occlusions, and (c) diverse and rapid camera motions (i.e. zoom-in, zoom-out, and clockwise rotation.).

A.8 EXPERIMENTS ON THE SPARSITY OF TRAJECTORY ATTENTION

To assess the generalization capability of trajectory attention under varying levels of sparsity, we conduct experiments in two scenarios: using trajectories with reduced density and applying a small region mask to the trajectories. As shown in Fig. 8, our method performs effectively in both settings. However, when the trajectory density is extremely sparse (below 1/32 resolution), the results become unstable.

Moreover, we also find trajectory attention can be generalized to sparse hand-crafter trajectories, as shown in Fig. 9.

A.9 COMPARISON ON CAMERA TRAJECTORIES

For clearer understanding, we visualize the predicted trajectories in Fig. 7, illustrating results from five scenes with a single camera trajectory. The figure shows that, thanks to the explicit modeling of camera motion, our method’s estimated trajectories closely align with the ground truth. In contrast, methods like He et al. (2024) exhibit inconsistencies in some cases.

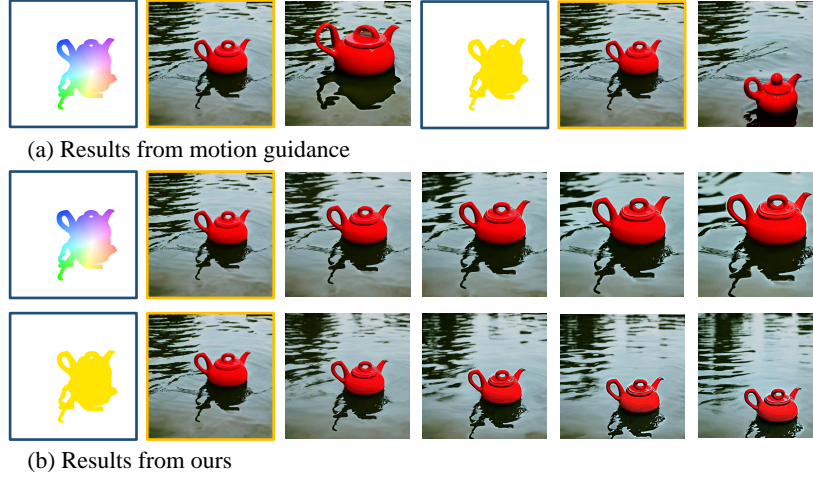


Figure 5: **Using synthetic optical flow as guidance.** Our method supports directly using optical flow to guide generation. Blue boxes indicate the optical flow. Yellow boxes indicate the reference image.

A.10 TRAINING DATA CONSTRUCTION

The training data consists of natural scenes with inherent camera and object movements. We estimate the optical flow and occlusion masks using the method proposed by (Yang et al., 2023). Examples are shown in Fig. 10.

A.11 MORE QUALITATIVE RESULT

We strongly recommend viewing the attached webpage for a more intuitive visualization.

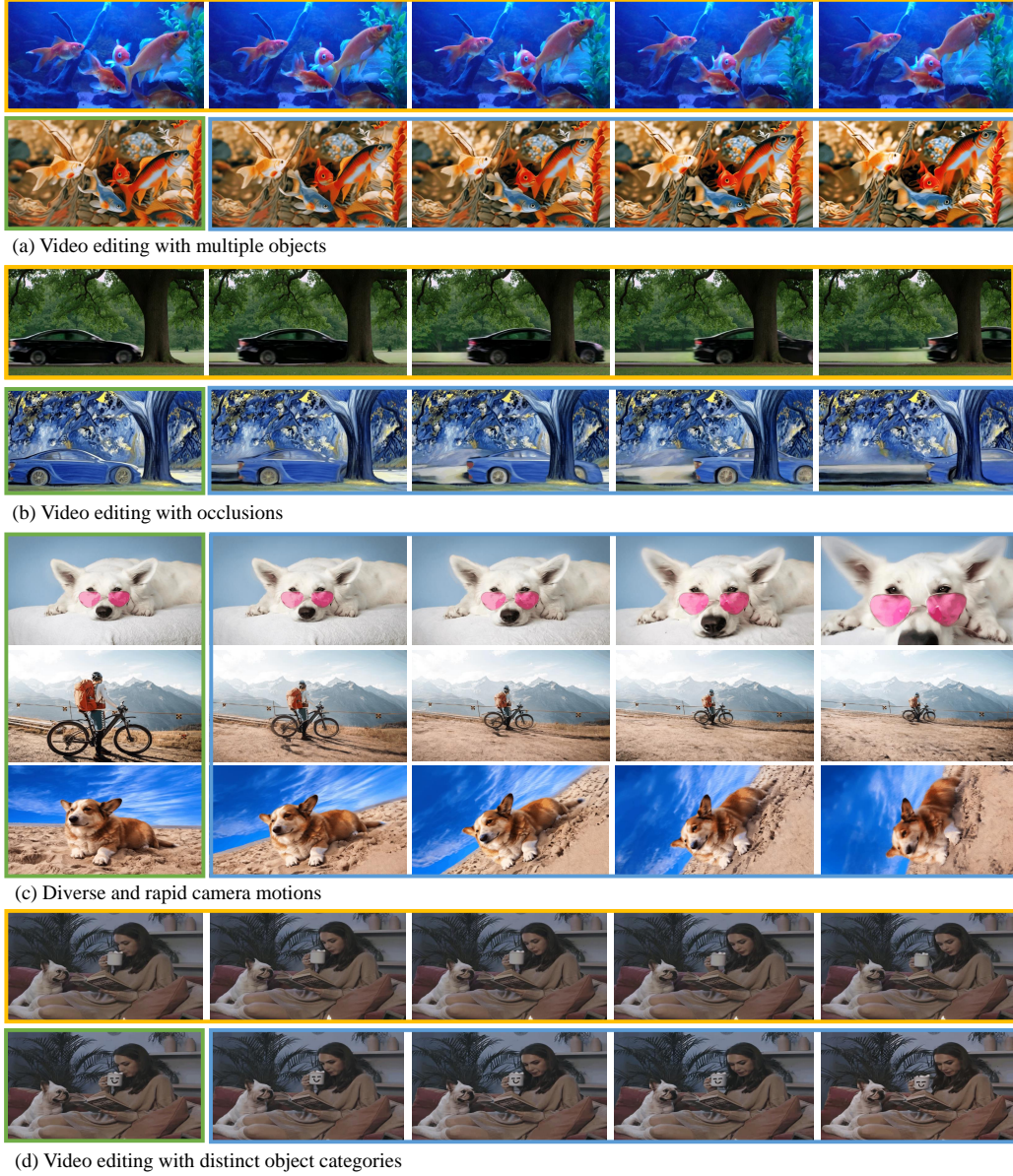


Figure 6: **Examples of challenging situations.** Our method effectively addresses complex scenarios, including (a) video editing involving multiple objects, (b) video editing in the presence of occlusions, (c) diverse and rapid camera movements, such as zooming in and out, as well as clock-wise rotations, and (d) video editing with distinct object categories. Please note that yellow boxes indicate reference videos, green boxes indicate input frames and blue boxes indicate output results.

REFERENCES

- Daniel Geng and Andrew Owens. Motion guidance: Diffusion-based image editing with differentiable motion estimators. [arXiv preprint arXiv:2401.18085](#), 2024.
- Hao He, Yinghao Xu, Yuwei Guo, Gordon Wetzstein, Bo Dai, Hongsheng Li, and Ceyuan Yang. Cameractrl: Enabling camera control for text-to-video generation. [arXiv preprint arXiv:2404.02101](#), 2024.

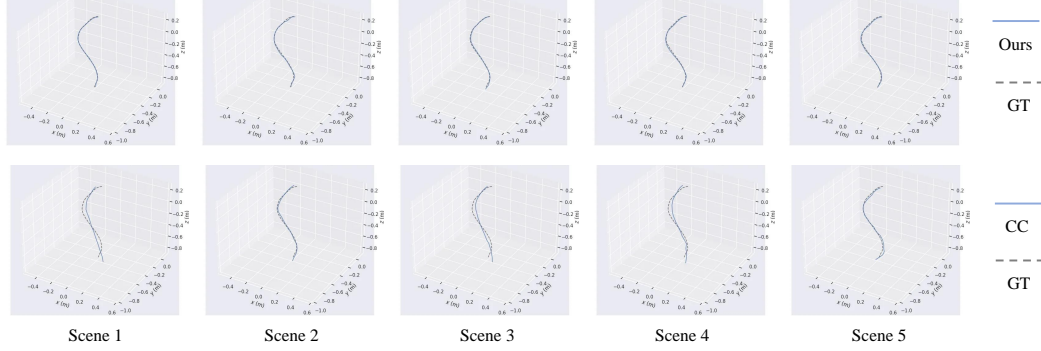


Figure 7: **Visualization of Camera Trajectories.** The first row displays the estimated trajectories from our generation alongside the ground truth trajectories. The second row presents the estimated trajectories from CameraCtrl (denoted as “CC”) compared to the ground truth. The results indicate that our method aligns significantly better with the ground truth camera motion trajectories.

Xuan Ju, Yiming Gao, Zhaoyang Zhang, Ziyang Yuan, Xintao Wang, Ailing Zeng, Yu Xiong, Qiang Xu, and Ying Shan. Miradata: A large-scale video dataset with long durations and structured captions. [arXiv preprint arXiv:2407.06358](https://arxiv.org/abs/2407.06358), 2024.

Nikita Karaev, Ignacio Rocco, Benjamin Graham, Natalia Neverova, Andrea Vedaldi, and Christian Rupprecht. Cotracker: It is better to track together. [arXiv preprint arXiv:2307.07635](https://arxiv.org/abs/2307.07635), 2023.

PKU-Yuan Lab and Tuzhan AI etc. Open-sora-plan, April 2024. URL <https://doi.org/10.5281/zenodo.10948109>.

William Peebles and Saining Xie. Scalable diffusion models with transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 4195–4205, 2023.

Lihe Yang, Lei Qi, Litong Feng, Wayne Zhang, and Yinghuan Shi. Revisiting weak-to-strong consistency in semi-supervised semantic segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 7236–7246, 2023.

Lihe Yang, Bingyi Kang, Zilong Huang, Zhen Zhao, Xiaogang Xu, Jiashi Feng, and Hengshuang Zhao. Depth anything v2. [arXiv preprint arXiv:2406.09414](https://arxiv.org/abs/2406.09414), 2024.

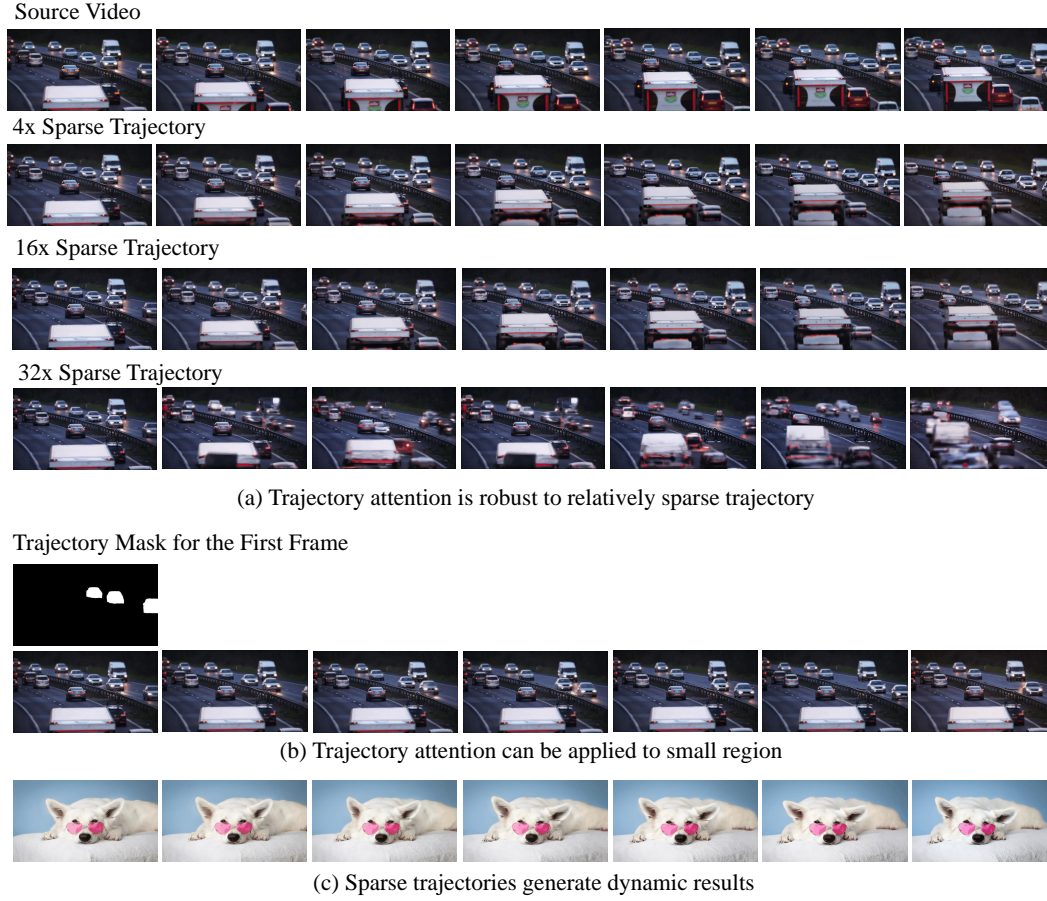


Figure 8: **Results on Sparse Trajectories.** In (a), we show that trajectory attention remains robust even with relatively sparse trajectories. Even when the trajectory density is reduced to 1/16 of the original video resolution, it still performs well in motion control. In (b), we apply the trajectory mask to selectively use only a portion of the trajectories, keeping the regions outside the mask static. The model accurately follows the motion within the small masked area. (c) If we apply sparse trajectories control to a specific region (*i.e.*, the dog region), the output results are more dynamic. Best viewed on the attached HTML file.

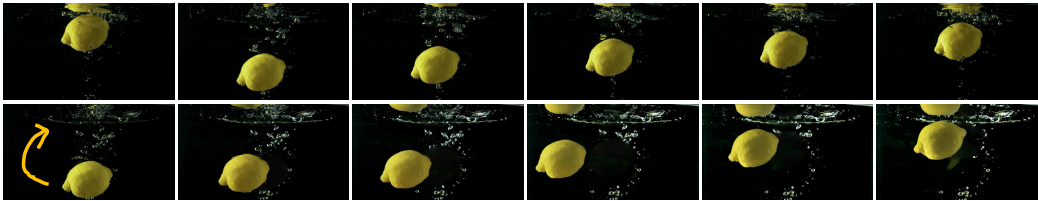


Figure 9: **Applications on drag signals.** Trajectory attention supports hand-crafted dragging trajectory. Row 1: origin videos. Row 2: dragged results.

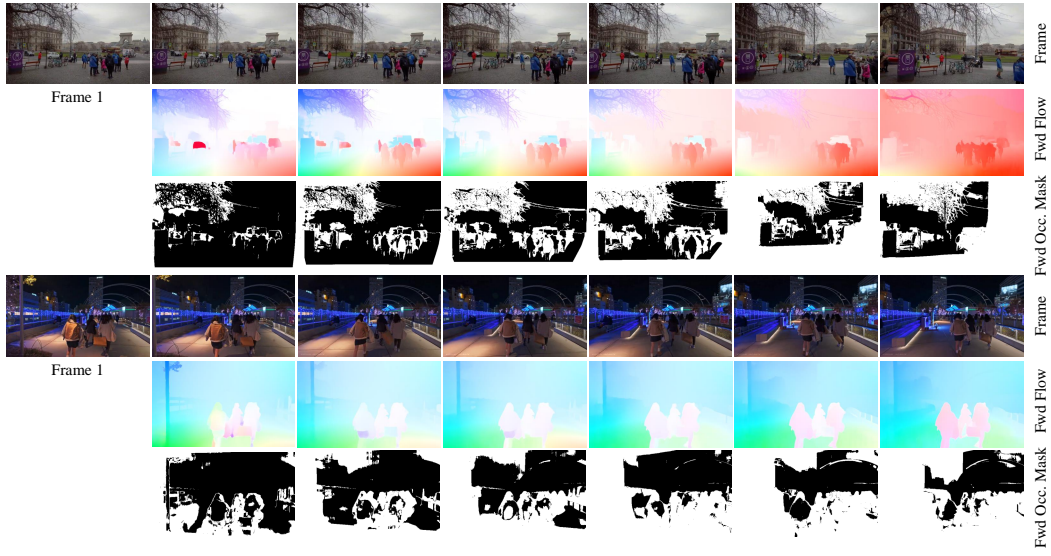


Figure 10: **Visualization on the training data.** The training data includes origin frames, predicted optical flow, and occlusion masks.



Figure 11: **Visualization on failure cases.** Our method encounters challenges when dealing with extremely fast motions as well as complex and difficult-to-estimate motion patterns.

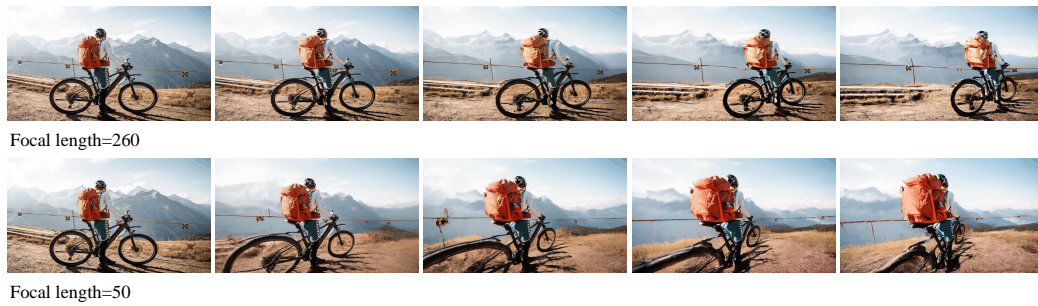


Figure 12: **Visualization on different intrinsic parameters.**