Beyond the Frame: Generating 360° Panoramic Videos from Perspective Videos

Supplementary Material

1. Supplementary Material Overview

In this supplementary material, we provide additional dataset and implementation details. Accompanying this supplementary file is our project page. Following the submission guideline, the project page is also accessible by opening the 'index.html' file in the supplementary material.

2. Dataset Collection and Statistics

In this section, we introduce a scalable data curation strategy for training a video-to-360° diffusion model. Then we show examples from our dataset and introduce its statistics to provide a rough understanding of our dataset.

2.1. Data Processing

We begin with the 360-1M dataset [13], which includes approximately 1 million 360° videos of varying quality. To establish a quality baseline, we retain only videos with more than 50 likes. Despite this initial filtering, the dataset still contains mislabeled 180° videos, standard perspective videos, static posters, static scenes, and unrealistic animations. To address this, we developed a scalable data processing pipeline:

1. **Format Filtering.** We sample frames from each video and detect horizontal lines in the center or vertical lines at the boundaries to verify the equirectangular format. Horizontal line detection removes up-down formatted 360° videos, while vertical line detection filters out perspective videos and posters.
2. **Intra-frame Filtering.** We compute LPIPS between the left and right halves to filter 180° videos and between the top and bottom halves to filter improperly formatted 360° videos.
3. **Inter-frame Filtering.** To ensure scene dynamics, we sample frames at random intervals and calculate the pixel variance. Static videos with minimal inter-frame variation are removed.

After coarse filtering, the videos are split into 10-second clips. We then apply fine-grained filtering using optical flow [5] to detect low-motion clips, TransNetv2 [11] to identify cuts, and DPText-DETR [16] to detect texts from unwrapped perspective views. Clips with excessive black pixels or low pixel variance are also excluded, as they indicate low visual complexity.

2.2. Dataset Statistics

The final dataset consists of 283,863 ten-second clips, distributed across 14 subject categories. The most prominent category, "Travel and Events," accounts for 63,935 clips. From this dataset, we also build a high-quality selected after manual inspection of the video frames. This subset was used for high-quality fine-tuning. The distribution of categories in the dataset is shown in Figure 1, with examples of filtered and included clips in Figures 2 and 3.

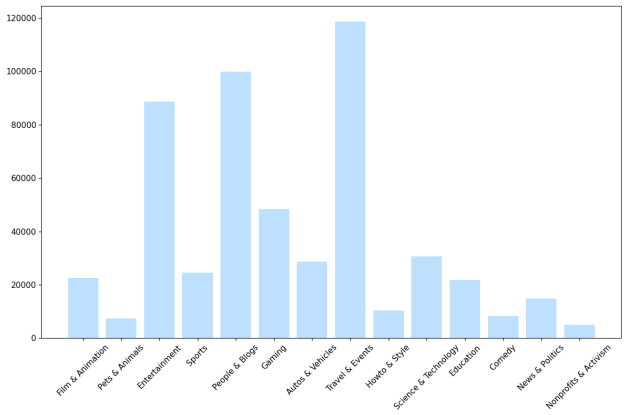

Figure 1. **Clip category distribution in our dataset.**

3. Implementation Details and Analyses

3.1. Perspective to Equirectangular Projection

We detail the mathematical process of mapping perspective video pixels to equirectangular maps. This includes equations for coordinate normalization, rotation, and spherical mapping.

To map a pixel coordinate (u, v) from an image with a given field of view, roll, pitch, and yaw to an equirectangular map, we first normalize the pixel coordinates to the normalized device coordinates (NDC). Assuming an image resolution of (W, H), the NDC coordinates (x_{ndc}, y_{ndc}) are given by

$$x_{ndc} = \frac{2u}{W} - 1, \quad y_{ndc} = \frac{2v}{H} - 1. \tag{1}$$

Given horizontal and vertical FOVs α and β, we compute a 3D direction vector (X, Y, Z) for the pixel in the camera's coordinate frame as follows:

$$X = x_{ndc} \cdot \tan\left(\frac{\alpha}{2}\right), \quad Y = y_{ndc} \cdot \tan\left(\frac{\beta}{2}\right), \quad Z = -1. \tag{2}$$

To reorient this vector from the camera frame to the equirectangular frame, we apply a series of rotations defined by the roll r, pitch p, and yaw y angles. Each angle defines a

rotation matrix: R_r for roll,

$$R_r = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(r) & -\sin(r) \\ 0 & \sin(r) & \cos(r) \end{bmatrix}, \tag{3}$$

R_p for pitch,

$$R_p = \begin{bmatrix} \cos(p) & 0 & \sin(p) \\ 0 & 1 & 0 \\ -\sin(p) & 0 & \cos(p) \end{bmatrix}, \tag{4}$$

and R_y for yaw,

$$R_y = \begin{bmatrix} \cos(y) & -\sin(y) & 0 \\ \sin(y) & \cos(y) & 0 \\ 0 & 0 & 1 \end{bmatrix}. \tag{5}$$

The rotated vector (X', Y', Z') is obtained by applying these transformations in the order $R_y \cdot R_p \cdot R_r$:

$$\begin{bmatrix} X' \\ Y' \\ Z' \end{bmatrix} = R_y \cdot R_p \cdot R_r \cdot \begin{bmatrix} X \\ Y \\ Z \end{bmatrix}. \tag{6}$$

We then convert (X', Y', Z') to spherical coordinates, where $\theta = \arctan 2(Y', X')$ and $\phi = \arcsin\left(\frac{Z'}{\sqrt{X'^2+Y'^2+Z'^2}}\right)$. Finally, the spherical coordinates are mapped to equirectangular pixel coordinates (u_{eq}, v_{eq}) for an equirectangular map of dimensions (W_{eq}, H_{eq}) by

$$u_{eq} = \frac{W_{eq}}{2\pi} \cdot (\theta + \pi), \quad v_{eq} = \frac{H_{eq}}{\pi} \cdot \left(\frac{\pi}{2} - \phi\right). \tag{7}$$

This yields the pixel location on the equirectangular map corresponding to the input pixel in the original image.

3.2. Training Details

Our model is initialized from the Stable Video Diffusion-I2V-XL model [1]. We implement a two-phase training strategy: initially at 384×768 resolution for 100K iterations, where we sample the noise scheduler parameter σ from a log-Gaussian distribution ($\log \sigma \sim \mathcal{N}(P_{\text{mean}}, P_{\text{std}}^2)$) and progressively increase the noise schedule from $(P_{\text{mean}}, P_{\text{std}}^2) = (-1, 1)$ to $(0, 1)$. In the second phase, we finetune the model at higher 512×1024 resolution on a high-quality subset for 20K iterations, employing context-aware training with a stronger noise schedule of $(P_{\text{mean}}, P_{\text{std}}) = (1, 1)$ as recommended by [3]. We set the sequence length $T = 25$ and context length $S = 5$. For both phases, we use the AdamW optimizer with a learning rate of 10^{-5} and a batch size of 16. The training required approximately six days on 8 A6000 GPUs for the first phase and four days on 8 A100 GPUs for the second phase.

Figure 2. **Examples of videos discarded during data the data filtering pipeline.** We discard 180° videos, standard perspective videos, static posters, static scenes, and unrealistic animations from the initial noisy dataset.

3.3. Inference Details on In-the-Wild Videos

For in-the-wild input videos, we first employ MegaSaM [8] to estimate the camera intrinsics and poses, followed by generating the corresponding masked equirectangular video used to condition the network. After generation, we apply video super-resolution model [6] enhanced by our proposed blended decoding to increase the spatial resolution of the generated video by a factor of 2. Note that we do not apply super resolution modules in ablation studies and comparison with baseline methods.

3.4. Metrics

We evaluate our results based on three key criteria: image quality, temporal coherency, and geometric consistency. For image quality, we use PSNR, LPIPS [17], Imaging Quality, and Aesthetic Quality metrics from VBench [7]. For temporal coherency, we employ FVD [12] and the Motion Smoothness [7]. For geometric consistency, we introduce a *line consistency* metric to evaluate whether straight lines remain straight within extrapolated views. This metric is particularly important for assessing whether our model preserves fundamental geometric properties when generating novel views. To quantitatively measure this consistency, we follow [9] and use EA-score [18] to evaluate the angular and Euclidean distances between line pairs.

Specifically, FVD is calculated on the full 360° scene to evaluate overall distribution, while VBench metrics are

Figure 3. **Video frames sampled from our dataset.** We arrange the video frames to from a 360° image.

applied to four square 2D projections (front, back, left, right) extracted from the 360° video, as VBench is designed for perspective videos. PSNR and LPIPS are computed only within masked regions of visible directions and aggregated across frames, since other directions are extrapolated. Though this visible region remains under-constrained (visible areas at timestamp 0 may not appear at timestamp T), this approach provides more accurate evaluation than existing video outpainting methods [2, 4, 14] that calculate scores over the entire generated video.

Line Consistency. We introduce a line consistency metric to evaluate geometric fidelity across extrapolated viewpoints. This metric assesses whether straight lines in the original perspective remain consistent in neighboring views. Our approach uses real-world perspective videos that contain prominent linear structures, such as lanes and sidewalks.

Specifically, we first annotate lines in input views, then detect corresponding lines in neighboring views unwrapped from generated 360° videos using the Hough transform. Then, we compute the analytical solution of ground truth lines in neighboring views using homography and employ bipartite matching to pair these with detected lines. Finally, we follow [9] and report the EA-score [18], a score in $[0, 1]$ to measure the angle and euclidean distance between two lines, between the matched ground truth and detected lines. An example of our dataset and the line detection result in shown in Fig. 4.

Figure 4. **Illustration of our line detection metric.** Given input view with annotated linear structures, we detect their extension in the neighboring views and measure their consistency.

3.5. Baseline Implementation Details

PanoDiffusion [15]. We reproduced this model due to the unavailability of their training code. We finetuned the image inpainting model [10] on the video frames of our dataset, omitting the depth branch due to the lack of depth information in the dataset. The model was trained for 50K iterations using the AdamW optimizer with a learning rate of 10^{-5} and a batch size of 128, running on 8 NVIDIA A6000 GPUs.

Be-Your-Outpainter [14] and Follow-Your-Canvas [2]. Video outpainting methods support only rectangular inputs, so we centered square videos on the canvas and expanded the vertical field of view to 180° and horizontal field of view 360°. For evaluation, we extracted three perspective videos from each 360° test video with FoVs of 60°, 90°, and 120°. Because these models require per-video optimization for

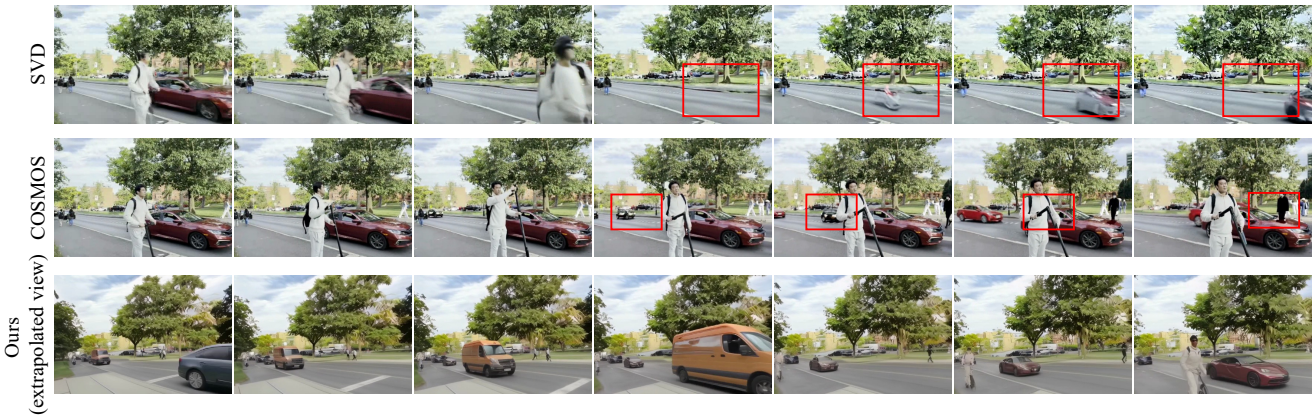

Figure 5. **Comparison with perspective video generation models.** Preserving shape consistency and dynamic plausibility remains an open challenge for video generation models. Specifically, our base model, SVD, exhibits noticeable appearance changes in the generated video (first row), while even state-of-the-art video models such as COSMOS demonstrate physical artifacts, where the black car on the back disappears (middle row).

each generation, they are very compute expensive, taking about 14 and 11 minutes, respectively, on a single NVIDIA A6000 GPU for each generation. In contrast, our method does not introduce additional compute overhead upon SVD, taking around 90 seconds for each generation while achieving significantly better quality.

Limitations. Due to computational resource constraints, our current output resolution (512×1024) is lower than that of typical 4K real-world panoramas. The resolution further decreases when unwrapping back to perspective views. Additionally, while our model substantially improves upon the base SVD model in terms of object dynamics and temporal consistency, it still exhibits shape inconsistencies and physics artifacts, similar to SVD and other SoTA video models such as COSMOS, as shown in Figure 5.

4. Additional Qualitative Results

Additional comparison, application, and in-the-wild video generation results are available in our project page.

References

[1] 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:2311.15127*, 2023. 2

[2] Qihua Chen, Yue Ma, Hongfa Wang, Junkun Yuan, Wenzhe Zhao, Qi Tian, Hongmei Wang, Shaobo Min, Qifeng Chen, and Wei Liu. Follow-your-canvas: Higher-resolution video outpainting with extensive content generation. *arXiv:2409.01055*, 2024. 3

[3] Ting Chen. On the importance of noise scheduling for diffusion models. *arXiv:2301.10972*, 2023. 2

[4] Loïc Dehan, Wiebe Van Ranst, Patrick Vandewalle, and Toon Goedemé. Complete and temporally consistent video outpainting. In *CVPR*, 2022. 3

[5] Gunnar Farnebäck. Two-frame motion estimation based on polynomial expansion. In *Image Analysis*, 2003. 1

[6] Jingwen He, Tianfan Xue, Dongyang Liu, Xinqi Lin, Peng Gao, Dahua Lin, Yu Qiao, Wanli Ouyang, and Ziwei Liu. Venhancer: Generative space-time enhancement for video generation. *arXiv:2407.07667*, 2024. 2

[7] Ziqi Huang, Yinan He, Jiashuo Yu, Fan Zhang, Chenyang Si, Yuming Jiang, Yuanhan Zhang, Tianxing Wu, Qingyang Jin, Nattapol Chanpaisit, et al. Vbench: Comprehensive benchmark suite for video generative models. In *CVPR*, 2024. 2

[8] 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. *arXiv:2412.04463*, 2024. 2

[9] Shengyi Qian and David F Fouhey. Understanding 3d object interaction from a single image. In *CVPR*, 2023. 2, 3

[10] Runwayml. Stable-diffusion-inpainting, 2022. 3

[11] Tomás Soucek and Jakub Lokoc. Transnet v2: An effective deep network architecture for fast shot transition detection. In *ACM MM*, 2024. 1

[12] Thomas Unterthiner, Sjoerd van Steenkiste, Karol Kurach, Raphaël Marinier, Marcin Michalski, and Sylvain Gelly. Fvd: A new metric for video generation. In *ICLR*, 2019. 2

[13] Matthew Wallingford, Anand Bhattad, Aditya Kusupati, Vivek Ramanujan, Matt Deitke, Aniruddha Kembhavi, Roozbeh Mottaghi, Wei-Chiu Ma, and Ali Farhadi. From an image to a scene: Learning to imagine the world from a million 360° videos. In *NeurIPS*, 2024. 1

[14] Fu-Yun Wang, Xiaoshi Wu, Zhaoyang Huang, Xiaoyu Shi, Dazhong Shen, Guanglu Song, Yu Liu, and Hongsheng Li. Be-your-outpainter: Mastering video outpainting through input-specific adaptation. In *ECCV*, 2024. 3

[15] Tianhao Wu, Chuanxia Zheng, and Tat-Jen Cham. Panodiffusion: 360-degree panorama outpainting via diffusion. In *ICLR*, 2023. 3

[16] Maoyuan Ye, Jing Zhang, Shanshan Zhao, Juhua Liu, Bo Du, and Dacheng Tao. Dptext-detr: Towards better scene text detection with dynamic points in transformer. In *AAAI*, 2023. 1

[17] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *CVPR*, 2018. 2

[18] Kai Zhao, Qi Han, Chang-Bin Zhang, Jun Xu, and Ming-Ming Cheng. Deep hough transform for semantic line detection. *TPAMI*, 2021. 2, 3