Pantheon360: Taming Digital Twin Generation via 3D-Aware 360° Video Diffusion

Supplementary Material

A. Overview

This supplementary material provides additional details to support the main paper. Section B provides comprehensive implementation details, including training and inference configurations, 3D Cache reconstruction settings, and dual-anchor latent fusion mechanism. Section C describes the detailed data curation pipeline for our training dataset, including quality filtering criteria and automatic trajectory annotation. Section D presents additional experimental results including a 3D Cache ablation study, runtime and memory analysis, 3D consistency validation via point cloud reconstruction, robustness analysis, and closed-loop trajectory validation.

In addition to this document, we provide an interactive HTML interface with supplementary videos demonstrating our method's capabilities on various tasks, including Google Street View synthesis, extended trajectory generation, and video stabilization. We also include qualitative comparisons with other state-of-the-art methods.

B. Implementation Details

B.1. Training and Inference Details

Our model is initialized from Argus [5], which is initialized from Stable Video Diffusion-I2V-XL model [1]. We train two separate models with identical configurations: (1) a single-anchor model conditioned on the first frame, and (2) a dual-anchor model conditioned on both start and end frames for interpolation tasks.

Both models are trained at 512×1024 resolution (height $\times$ width) in equirectangular format for 50,000 iterations, requiring approximately 5 days on 4 A100 GPUs. We set the sequence length $T = 25$ frames and train with a stronger noise schedule of $(P_{\text{mean}}, P_{\text{std}}) = (1, 1)$. We use the AdamW optimizer with a learning rate of 1×10^{-5} and a batch size of 16. The training employs mixed precision (FP16) and gradient checkpointing for memory efficiency. At inference time, we use 25 denoising steps with a guidance scale of 5.0. For extended trajectory synthesis, we chain multiple generation segments by using the final frame of one segment as the anchor for the next.

B.2. 3D Cache Reconstruction

We use π^3 (Pi3) [10] as our primary 3D reconstruction foundation model. Since Pi3 is designed for perspective images, we first convert each 360° equirectangular input frame into multiple perspective views before feeding them into the model.

Perspective View Extraction. For each 360° input frame, we use the Equi2Pers converter to extract perspective views with 90° horizontal field of view and output resolution of 768×512 (width $\times$ height). We sample at two pitch angles: 0° (horizontal view) and 60° (downward view toward the floor), excluding ceiling views to avoid sky regions. For yaw sampling, we use 50% overlap between adjacent views, sampling at 45° intervals, resulting in 8 views per pitch level for complete 360° coverage. In total, we extract 16 perspective views per 360° frame (8 horizontal + 8 floor views).

These 16 perspective views are fed into PI3 to produce dense point cloud predictions with associated confidence scores. The predictions from all views are merged in a common coordinate frame defined by the camera poses estimated by Pi3.

Point Cloud Filtering. We apply multiple filtering strategies to ensure high-quality 3D Cache. For confidence filtering, we apply a confidence threshold of 0.25, converting the raw confidence scores to probabilities via sigmoid function before thresholding. For edge filtering, we detect depth discontinuities using a relative tolerance of 0.03 and set their confidence to zero to remove unreliable edge points. Finally, we implement sky masking to remove unreliable sky points, which typically lack geometric structure and can introduce artifacts.

The filtered point clouds from all 16 views are merged to form the complete 3D Cache. Our framework is compatible with other 3D reconstruction methods such as VGGT [8], DUSt3R [9], and MASt3R [4].

B.3. Dual-Anchor Latent Fusion

For our dual-anchor interpolation model, we employ the latent fusion technique from Time Reversal Fusion [2] to blend information from both start and end anchor frames. This approach is particularly effective when the 3D Cache quality is suboptimal due to sparse input views, where direct geometric conditioning alone may lead to discontinuities between the interpolated video and the target end frame.

Bidirectional Geometric Conditioning. Given start and end frames, we reconstruct a 3D Cache from both anchor frames and render it along the target trajectory in two directions. The forward rendering produces a geometry video $V_{\text{geo}}^{\text{fwd}}$ by rendering from the start frame's viewpoint toward the end frame. The backward rendering produces $V_{\text{geo}}^{\text{bwd}}$ by rendering from the end frame's viewpoint toward the start frame (i.e., the trajectory is reversed). Both geometric videos are encoded into latent space as $v_{\text{fwd}} = \mathcal{E}(V_{\text{geo}}^{\text{fwd}})$ and $v_{\text{bwd}} = \mathcal{E}(V_{\text{geo}}^{\text{bwd}})$, where $\mathcal{E}$ denotes the VAE encoder.

Latent Fusion Process. The fusion process operates in the latent space during the denoising procedure. At each denoising timestep t, we perform two separate denoising passes with different geometric and semantic conditioning. The forward pass is conditioned on the start frame features c_s and forward geometric scaffold v_{fwd}, computing $\boldsymbol{x}_{t-1,s} = \Phi(\boldsymbol{x}_t, c_s, v_{\text{fwd}}, t)$. The backward pass is conditioned on the end frame features c_e and backward geometric scaffold v_{bwd}, computing $\boldsymbol{x}_{t-1,e} = \Phi(\boldsymbol{x}_t, c_e, v_{\text{bwd}}, t)$, where Φ represents our denoising U-Net.

These two predictions are then fused using a simple averaging strategy to produce the final denoised latent: $\boldsymbol{x}_{t-1} = \frac{1}{2}(\boldsymbol{x}_{t-1,s} + \boldsymbol{x}_{t-1,e})$. This fusion effectively combines the geometric information from both directions, helping to resolve inconsistencies and ensuring smooth convergence to the end frame while maintaining temporal coherence throughout the interpolated sequence.

Note that we do not employ the noise injection refinement proposed in [2] (setting $t_0 = 0$ and $M = 0$) to maintain faster inference speed while still achieving effective interpolation results as demonstrated in our ablation study (Table 3 in the main paper). The bidirectional geometric conditioning alone provides sufficient guidance for high-quality interpolation.

C. Data Curation and Preparation

C.1. Quality Filtering Pipeline

We build our training dataset starting from a curated subset prepared by [7], which selected approximately 100,000 high-quality video clips from the 360-1M dataset. We apply additional comprehensive filtering to further ensure training data quality suitable for our trajectory-controlled generation task.

Format Validation. We first filter out mislabeled videos using two methods. For dual fisheye detection, we use Hough Circle detection to identify dual-fisheye format videos (two circular regions side-by-side), which are not true equirectangular panoramas. We sample 4 frames per video and reject videos where more than 90% of frames contain dual circles. For perspective detection, we analyze boundary smoothness to detect perspective videos mislabeled as 360°. Videos with boundary smoothness greater than 0.25 are rejected.

Motion Quality Assessment. Since our method requires meaningful camera motion, we filter videos with insufficient or problematic motion. For optical flow analysis, we compute optical flow using RAFT [6] at 1 FPS sampling rate with input resolution of 512×256. We apply equirectangular-aware weighting (latitude-based cosine weighting) to account for polar distortion. Videos with 75th percentile flow magnitude less than 3.0 pixels are rejected as static. For cut detection, we detect abrupt scene cuts using PySceneDetect to avoid training on concatenated clips. Videos with single-frame cut ratio greater than 0.3 or overall cut ratio greater than 0.2 are rejected.

Content Quality Filtering. We remove low-quality or inappropriate content through two mechanisms. For image set detection, we detect slideshows by computing frame-to-frame MSE at 1 FPS. Videos with minimum MSE less than 1.0 (indicating identical consecutive frames) are rejected. For static region detection, we analyze top and bottom regions at multiple height ratios (from 1% to 80% of frame height) to detect static overlays or borders. For the 20% height region, videos with MSE less than 1.0 in either top or bottom regions are rejected as they likely contain static UI elements or watermarks.

Trajectory Annotation Filtering. After applying ViPE [3] for automatic trajectory annotation, we additionally filter videos where ViPE fails to produce reliable results. This includes videos with insufficient camera baseline (too little camera motion for robust pose estimation), videos with too few SLAM feature points (indicating textureless or highly repetitive scenes), and videos where ViPE's optimization fails to converge.

Filtering Statistics. Starting from the 100K curated subset, our additional filtering pipeline retains approximately 55,000 high-quality videos, corresponding to a retention rate of 55%. This ensures that our final training dataset contains only videos with clear 360° format, sufficient motion, no static artifacts, and successful trajectory annotations. Each retained video is a 5-second clip, providing diverse camera trajectories and scene content for training.

D. Additional Experimental Results

D.1. 3D Cache Ablation Study

We conduct an ablation study by progressively dropping points from the 3D Cache to quantify the importance of geometric conditioning V_{geo}. We report results on both Web360 (Table 1) and Habitat (Table 2) benchmarks.

Drop Ratio	FVD↓	SSIM↑	PSNR↑	LPIPS↓	MET3R↓
0% (Ours)	**356.2**	**0.746**	**22.84**	**0.065**	**0.284**
25%	382.6	0.708	22.10	0.087	0.318
50%	427.9	0.643	20.76	0.124	0.372
75%	496.4	0.539	18.85	0.178	0.446
100% (w/o V_{geo})	553.3	0.421	16.93	0.251	0.523

Table 1. **3D Cache ablation on Web360.** Performance degrades consistently as more points are dropped from the 3D Cache.

Without V_{geo}, the model reduces to standard image-to-video generation. While it can still produce visually plausible results, geometric correctness cannot be fully guaranteed. Moreover, since we explicitly rely on the rendered point cloud as a condition for camera control, removing V_{geo} makes precise trajectory control infeasible.

Drop Ratio	FVD↓	SSIM↑	PSNR↑	LPIPS↓	MET3R↓
0% (Ours)	**450.7**	**0.756**	**20.39**	**0.091**	**0.303**
25%	489.2	0.712	19.55	0.118	0.349
50%	551.8	0.638	18.12	0.162	0.414
75%	637.5	0.524	16.28	0.231	0.502
100% (w/o V_{geo})	724.2	0.387	14.22	0.319	0.597

Table 2. **3D Cache ablation on Habitat.** Consistent with Web360, removing V_{geo} leads to significant performance degradation across all metrics.

D.2. Runtime and Memory Analysis

We provide detailed runtime and memory analysis on Google Map data using a single A100 GPU at 1024×512 resolution. As shown in Table 3, diffusion denoising is the main bottleneck, accounting for approximately 80% of total inference time. Our method runs entirely on a single GPU, and the modular framework is compatible with faster diffusion models, which can reduce inference time in future work.

Setting	Input Views	Output Frames	Time (s) Recon.	Render	Diff.	Total	Mem.
Single-view	1	25	34s	2s	163s	199s	30GB
Interp. (w/ latent fusion)	2	25	50s	5s	320s	375s	41GB
Long traj. (w/ latent fusion)	5	100	74s	7s	1284s	1365s	41GB

Table 3. **Runtime and memory analysis** across different settings on a single A100 GPU. Recon., Render, and Diff. denote the time for 3D point cloud reconstruction, geometry rendering, and diffusion denoising, respectively.

D.3. 3D Consistency Validation via Point Cloud Reconstruction

To validate that our generated videos maintain 3D geometric consistency while successfully hallucinating unseen regions, we reconstruct 3D point clouds from both the reference image and our generated video using Pi3 [10]. As shown in Figure 1, the point cloud reconstructed from the reference image (Before) contains only the visible geometry from the input viewpoint. In contrast, the point cloud reconstructed from our generated video (After) is significantly more complete, successfully hallucinating previously occluded regions while maintaining consistency with the original scene structure. This demonstrates that our method not only preserves 3D geometric consistency but also generates plausible geometry for unseen areas, resulting in a more complete 3D reconstruction. This capability is essential for applications requiring comprehensive scene understanding from limited input views.

D.4. Robustness Analysis and Failure Cases

Our method is robust to moderate reconstruction errors due to the video diffusion prior. Even when the rendered 3D

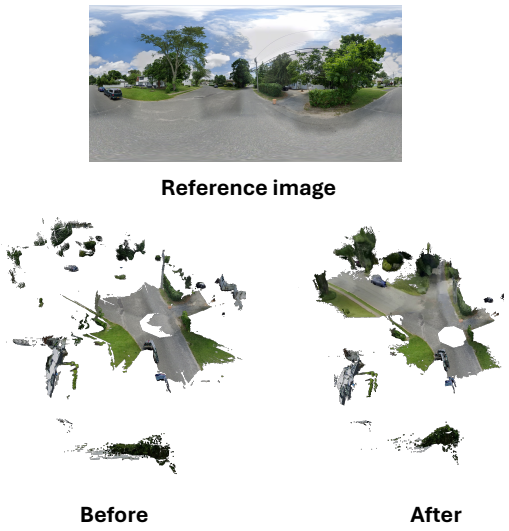

Figure 1. **3D Point Cloud Reconstruction Quality.** We reconstruct 3D point clouds using Pi3 [10] from the reference image (Before) and our generated video (After). Our generated video produces a more complete 3D reconstruction by successfully hallucinating occluded regions while maintaining geometric consistency with the original scene.

Cache contains holes or inaccuracies (e.g., from dynamic objects or low-light conditions), our model can inpaint and refine these regions, as shown in Figure 2.

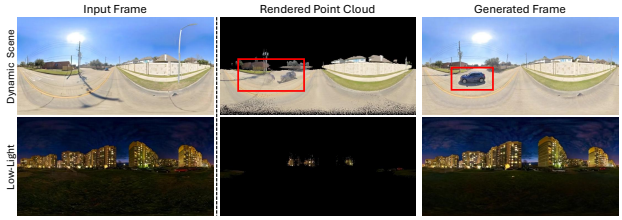

Figure 2. **Robustness to 3D Cache imperfections.** Our video diffusion prior can inpaint and refine regions where the 3D Cache contains holes or inaccuracies caused by dynamic objects or low-light conditions.

However, we identify two failure cases illustrated in Figure 3: (1) crowded dynamic scenes with many moving objects, resulting in motion blur artifacts; (2) input 360° images with stitching artifacts that propagate through the pipeline. Addressing these limitations requires future advances in 4D reconstruction or higher-quality input capture.

D.5. Closed-Loop Trajectory Validation

We validate trajectory robustness on a closed-loop trajectory from Google Map data. Using sparse 360° panoramas as input, our method generates temporally consistent video when revisiting the starting region, as shown in Figure 4.

Figure 3. **Failure cases.** Our method struggles with (1) crowded dynamic scenes with many moving objects, and (2) input 360° images with stitching artifacts.

This is enabled by our 3D Cache, which provides persistent geometric grounding throughout the trajectory.

Figure 4. **Closed-loop trajectory validation.** Our method generates consistent video along a closed-loop trajectory, successfully revisiting the starting region without temporal inconsistencies.

References

[1] Andreas Blattmann et al. Stable video diffusion: Scaling latent video diffusion models to large datasets. *arXiv preprint arXiv:2311.15127*, 2023. 1

[2] Haiwen Feng, Zheng Ding, Zhihao Xia, Simon Niklaus, Victoria Abrevaya, Michael J Black, and Xuaner Zhang. Explorative inbetweening of time and space. In *European Conference on Computer Vision*, pages 378–395. Springer, 2024. 1, 2

[3] Jiahui Huang, Qunjie Zhou, Hesam Rabeti, Aleksandr Korovko, Huan Ling, Xuanchi Ren, Tianchang Shen, Jun Gao, Dmitry Slepichev, Chen-Hsuan Lin, Jiawei Ren, Kevin Xie, Joydeep Biswas, Laura Leal-Taixe, and Sanja Fidler. Vipe: Video pose engine for 3d geometric perception. In *NVIDIA Research Whitepapers arXiv:2508.10934*, 2025. 2

[4] Vincent Leroy, Yohann Cabon, and Jérôme Revaud. Grounding image matching in 3d with mast3r. In *European Conference on Computer Vision (ECCV)*, pages 71–91. Springer, 2024. 1

[5] Zidong Tan, Chenhao Ji, Fuchen Tan, and Yiyang Shen. Beyond the frame: Generating 360° panoramic videos from perspective videos. *arXiv preprint arXiv:2504.07940*, 2025. 1

[6] Zachary Teed and Jia Deng. Raft: Recurrent all-pairs field transforms for optical flow. In *European conference on computer vision*, pages 402–419. Springer, 2020. 2

[7] 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. *Advances in Neural Information Processing Systems*, 37:17743–17760, 2024. 2

[8] Jianyuan Wang, Minghao Chen, Nikita Karaev, Andrea Vedaldi, Christian Rupprecht, and David Novotny. Vggt: Visual geometry grounded transformer. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 5294–5306, 2025. 1

[9] Shuzhe Wang, Vincent Leroy, Yohann Cabon, Boris Chidlovskii, and Jerome Revaud. Dust3r: Geometric 3d vision made easy. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 20697–20709, 2024. 1

[10] Yifan Wang, Jianjun Zhou, Haoyi Zhu, Wenzheng Chang, Yang Zhou, Zizun Li, Junyi Chen, Jiangmiao Pang, Chunhua Shen, and Tong He. π^3: Scalable permutation-equivariant visual geometry learning. *arXiv preprint arXiv:2507.13347*, 2025. 1, 3