Vivid4D: Improving 4D Reconstruction from Monocular Video by Video Inpainting

Supplementary Material

6. Implemenentation Details

6.1. Training Data Curation

Based on our proposed pipeline that utilizes pre-trained 2D tracking model to generate masked videos, we curated our training dataset from OpenVid-1M [55]. To ensure the quality and effectiveness of the data, we applied two filtering criteria. First, we removed videos with fixed viewpoints using the camera motion annotations provided by OpenVid-1M, as they do not offer the viewpoint variation needed for robust view synthesis and 3D understanding. Second, we filtered out sequences with insufficient dynamic content, despite having camera motion. Note that for process efficiency, we tracked dense grids on downsampled videos (128×128) with a grid size of 120, then resized the masks to 512×512. We computed the ratio of masked regions to the total video area:

$$R_v = \frac{1}{T} \sum_{t=1}^{T} \frac{|\mathcal{M}_t|}{|I_t|}$$

where $|\mathcal{M}_t|$ represents the number of pixels in the mask region of frame t, and $|I_t|$ is the total number of pixels in frame t. Videos with a ratio R_v below a threshold τ ($\tau = 0.98$) were excluded from our training set, as they indicate limited scene dynamics and thus provide insufficient novel content for learning inpainting patterns.

Through this filtering process, our final dataset comprises $150K$ video clips totaling $7.5M$ frames, each containing rich viewpoint variations and dynamic scene content. The simplicity and automation of our curation pipeline makes it particularly scalable, enabling continuous dataset expansion as new video content becomes available online.

6.2. Training Details of Diffusion Model

We fine-tuned our video inpainting diffusion model based on a 1.7B text-to-video diffusion model with 3D U-Net from ali-vilab [7, 51, 76]. The training data is selected from OpenVid-1M [55], which provides detailed text captions for each video. We center-crop the input training frames to a resolution of 512×512, with a frame length of 16, and randomly select an anchor video as a 16-frame slice from the video. We expand the input channels of the original U-Net from 4 to 13, initializing the additional convolutional layer parameters to zero. The entire U-Net was trained for 450K iterations with a learning rate of 10^{-5} and a batch size of 2.

During inference, we process input videos exceeding 16 frames using multiple overlapping slices. To maintain temporal consistency, each subsequent slice (after the first) incorporates the last 4 frames from the previous slice, concatenated with the next 12 frames as input. This overlapping strategy ensures smooth transitions between slices and mitigates temporal discontinuities in the reconstructed video.

6.3. Depth Scale Alignment & View Warping Details

Depth Scale Alignment: To align the monocular depth predicted by the model to the metric scale, we utilize the sparse point cloud $\{X_i\}_{i=1}^{M}$ from COLMAP [65, 66] and apply least squares algorithm combined with the RANSAC algorithm [22] to achieve a more robust and accurate depth map, effectively reducing the impact of outliers and improving alignment reliability.

Specifically, given a video $\mathcal{V} = \{I_t\}_{t=1}^{T}$ with T frames and its camera intrinsics $\mathbf{K}$, poses $\{\mathbf{T}_t\}_{t=1}^{T}$, we first project the sparse 3D points onto the image plane for each frame t:

$$\mathbf{P}_t^i = \mathbf{K}[\mathbf{T}_t \tilde{\mathbf{X}}_i]_{1:3}, \quad d_t^i = [\mathbf{P}_t^i]_3, \quad p_t^i = [\mathbf{P}_t^i / d_t^i]_{1:2}$$

where $\tilde{X}_i$ is the homogeneous coordinate of each point in world frame, $[\cdot]$ extracts the corresponding component of the coordinate. We determine valid points located in the image plane based on pixel coordinates, and then compute an optimal scale factor α_t and a shift factor β_t using RANSAC combined with least squares optimization:

For each frame t, we perform RANSAC for $K = 100$ iterations. In each iteration:

1. Randomly sample 10 valid projected points $\{p_t^j\}_{j=1}^{10}$ with their corresponding metric depths $\{d_t^j\}_{j=1}^{10}$.
2. Compute candidate scale $\alpha_t^{(k)}$ and shift $\beta_t^{(k)}$ by minimizing the least square error:
$$\alpha_t^{(k)}, \beta_t^{(k)} = \underset{\alpha,\beta}{\operatorname{argmin}} \sum_{j=1}^{10} \left(\alpha \hat{D}_t(p_t^j) + \beta - d_t^j \right)^2$$
3. Apply the computed scale and shift to align the predicted depth map: $D_t^{(k)} = \alpha_t^{(k)} \hat{D}_t + \beta_t^{(k)}$.
4. Back-project all valid pixels using the aligned depth to obtain a point cloud $\mathcal{P}_t^{(k)}$.
5. Compute the Chamfer distance between this point cloud and the corresponding COLMAP sparse points:
$$\mathrm{CD}_t^{(k)} = \frac{1}{|\mathcal{P}_t^{(k)}|} \sum_{p \in \mathcal{P}_t^{(k)}} \min_{q \in \mathcal{X}_t} \|p - q\|_2^2 + \frac{1}{|\mathcal{X}_t|} \sum_{q \in \mathcal{X}_t} \min_{p \in \mathcal{P}_t^{(k)}} \|q - p\|_2^2$$

where $\mathcal{X}_t$ represents the COLMAP sparse points visible

Dataset / Scene	Start Frame	End Frame	Num Frames
iPhone Dataset			
apple	290	321	32
block	240	303	64
paper windmill	56	136	81
spin	10	90	81
teddy	185	265	81
HyperNeRF Dataset			
3dprinter	40	103	64
broom	106	169	64
chicken	40	103	64

Table 7. **Details of the datasets** we test for 4D Reconstrcution.

in frame t.

Finally, we select the optimal scale and shift parameters that yield the minimum Chamfer distance:

$$\alpha_t, \beta_t = \alpha_t^{(k^*)}, \beta_t^{(k^*)} \quad \text{where} \quad k^* = \underset{k}{\operatorname{argmin}} \ \mathrm{CD}_t^{(k)}$$

The final metric-aligned depth maps are obtained as $D_t = \alpha_t \hat{D}_t + \beta_t$.

View Warping: The view warping stage transforms each frame to a desired novel viewpoint using the aligned depth maps and camera poses independently. For a target camera pose $\mathbf{T}'_t$, we perform a two-step projection process. First, we back-project each pixel p_t^m from frame I_t into 3D space using its depth value:

$$\tilde{X}(p_t^m) = \mathbf{T}_t^{-1} \tilde{p}_{cam,t}^m, \quad p_{cam,t}^m = D_t(p_t^m)\mathbf{K}^{-1}\tilde{p}_t^m$$

where $\tilde{\cdot}$ represents the homogeneous coordinate. These 3D points are then projected onto the target view by perspective projection $\pi(\cdot)$:

$$p_t^{m\prime} = \pi(\tilde{X}(p_t^m), \mathbf{T}'_t, \mathbf{K})$$

The warping process inevitably creates regions with invalid projections due to occlusions. This results in a masked video sequence $\mathcal{V}' = \{I'_t\}_{t=1}^T$, where each warped frame $I'_t \in \mathbb{R}^{H\times W\times 3}$ is accompanied by a binary mask $\mathcal{M}'_t \in \{0,1\}^{H\times W}$, indicating valid pixels (1) and masked regions (0).

6.4. Iterative View Augmentation

Following recent advances in 3D reconstruction [16, 26], we pre-compute a set of target camera viewpoints in 4D space. Specifically, we start with a data buffer $\mathcal{D}_0$ with input monocular video $\mathcal{V}^0$ along with its aligned depths D^0 and poses $\mathbf{T}^0$, namely $\mathcal{D}_0 = (\mathcal{V}^0, D^0, \mathbf{T}^0)$. We define H novel camera poses $\{\mathbf{T}^i\}_{i=1}^H$ distributed around the scene and set the total number of iteration to N, resulting in the synthesis of $h = H/N$ novel videos in each iteration.

During iteration $j (1 \le j \le N)$, we select frames with minimal warping angles from data buffer $\mathcal{D}_{j-1}$ at each timestamp t and warp them to target poses $\mathbf{T}_t^{j,l}$ by our depth-guided warping pipeline, where $l \in \{1, 2, ..., h\}$. Specifically, we first identify the two most peripheral frames from the data buffer $\mathcal{D}_{j-1}$ at each timestamp t, denoted as $I_{t,1}$ and $I_{t,2}$ with poses $\mathbf{T}_{t,1}$ and $\mathbf{T}_{t,2}$. We then select a total of h closest unvisited target poses from the set of remaining target poses, in which each target pose is closest to either of the peripheral frames:

$$\{\mathbf{T}_t^{j,1}, ..., \mathbf{T}_t^{j,h}\} = \underset{\mathbf{T}^i \in \mathcal{U}_t}{\operatorname{argmin}} \ \min(d(\mathbf{T}^i, \mathbf{T}_{t,1}), d(\mathbf{T}^i, \mathbf{T}_{t,2}))$$

where $\mathcal{U}_t$ is the set of unvisited poses for frame t, and $d(\cdot,\cdot)$ measures the pose distance. For each selected target pose $\mathbf{T}_t^{j,l}$ where $l \in \{1, 2, ..., h\}$, we find the closer peripheral frame and warp it to the target pose using our depth-guided warping pipeline. Next, we organize the masked frames temporally based on warp distance rank, meaning that the l-th least distant masked frames from each timestamp are grouped together into a video. This results in h novel video sequences $\mathcal{V}^j = \{\mathcal{V}^{j,1}, ..., \mathcal{V}^{j,h}\}$ where $\mathcal{V}^{j,l} = \{I_t^{j,l}\}_{t=1}^T$, along with corresponding masks $\mathcal{M}^j = \{\mathcal{M}^{j,1}, ..., \mathcal{M}^{j,h}\}$ with consistent occluded regions.

To complete the novel view synthesis for each of the h videos, we employ our trained video inpainting diffusion model $\mathcal{F}$, which utilizes the input monocular video $\mathcal{V}^0$ as an anchor along with a text description of the scene:

$$\hat{\mathcal{V}}^{j,l} = \mathcal{F}(\mathcal{V}^{j,l}, \mathcal{M}^{j,l}, \mathcal{V}^0)$$

where $\hat{\mathcal{V}}^{j,l} = \{\hat{I}_t^{j,l}\}_{t=1}^T$ represents the final inpainted video sequence for the l-th novel video in iteration j.

After each successful inpainting iteration, we estimate the metric depth $D^{j,l}$ for each of the h novel synthesized videos and update the data buffer $\mathcal{D}_j = \mathcal{D}_{j-1} \cup \{(\hat{\mathcal{V}}^{j,l}, D^{j,l}, \mathbf{T}^{j,l})\}_{l=1}^h$, where $\mathbf{T}^{j,l} = \{\mathbf{T}_t^{j,l}\}_{t=1}^T$. We repeat this process until all pre-defined poses are traversed by each frame. Note that to prevent error accumulation, views with cumulative warp angles exceeding a predefined threshold are excluded from supervising the reconstruction.

Above all, we progressively build a set of multi-view observations of the scene. Since our pre-defined target poses cover the entire observation space with smooth transitions between adjacent poses, this iterative approach allows us to gradually expand viewpoint coverage, hence maintaining geometric accuracy. This strategy also helps mitigate floaters that typically arise from inaccurate depth estimates

Method	mPSNR ↑	mSSIM ↑	mLPIPS ↓
StereoCrafter	16.35	0.5052	0.5491
ViewCrafter	16.42	0.5027	0.5441
ProPainter	16.40	0.4869	0.4993
NVS-Solver	16.37	0.4979	0.5536
TrajAttention	16.40	0.4883	0.5786
Ours	**16.80**	**0.5170**	**0.4750**

Table 8. **Quantitative comparison of dynamic scene reconstruction** on iPhone dataset and HyperNeRF dataset.

when warping at large angles. By the final iteration, we obtain data buffer $\mathcal{D}_N$, which provides rich supervision signals for robust 4D scene reconstruction.

6.5. Details for Obtaining Supervision Masks

During iterative view augmentation, we introduce a supervision mask to avoid the inconsistency of 4D reconstruction supervision signals caused by repeated inpainting across different iterations.

To maintain this mask, we utilize a global point cloud $\mathcal{P} = \{P_i\}_{i=1}^{K}$ merged from all input monocular frames to track the inpainting history. For each inpainted frame, we first render the point cloud to the corresponding camera pose, generating a visibility mask $V_t^{j,l}$ where $V_t^{j,l}(p) = 1$ indicates visible regions and $V_t^{j,l}(p) = 0$ indicates invisible regions. When a masked frame is inpainted, we check the visibility of each filled pixel $p \in \mathcal{M}_t^{j,l}$, where $\mathcal{M}_t^{j,l}$ is the binary mask corresponding to the masked frame. If the pixel is not yet visible ($V_t^{j,l}(p) = 0$), we back-project it into the global point cloud as $P_p \in \mathbb{R}^3$ (making this region visible in future renderings) and set its supervision mask value $S_t^{j,l}(p) = 1$. If the pixel is already visible ($V_t^{j,l}(p) = 1$), we set $S_t^{j,l}(p) = 0$. Note that the supervision mask for the initially visible areas of the scene is always set to 1. This mechanism ensures that only the first inpainted result for each region is used as supervision signal, preventing conflicting supervision from repeated inpainting of the same region.

7. Additional Experimental Results

7.1. Monocular 4D Reconstruction

Datasets: We select a continuous clip containing a few dozen frames from each test scene in the iPhone [25] and HyperNeRF [58] datasets, rather than using all available observations. The perspectives used for reconstruction exhibit significant parallax compared to the test viewpoints in these cases. The specific scenes, along with their start and end frames, are detailed in Tab. 7.

Comparison Baselines: We presented several of our 4D reconstruction results in the main paper (Sec. 4.1). Here, we further provide comparisons with StereoCrafter [108], ViewCrafter [101], ProPainter [111], NVS-Solver [98] and TrajAttention [87] to illustrate our superiority in reconstruction quality. Among these, StereoCrafter and ProPainter are primarily used for video inpainting, while ViewCrafter targets 3D-aware inpainting. NVS-Solver and TrajAttention aim to generate novel views given a single image. We leverage these models to fill in or synthesize novel view images within our pipeline, which are then used to support 4D reconstruction.

Results: The results are shown in Tab. 8 and Fig. 10. It can be observed that StereoCrafter produces results with noticeable color differences, which is primarily due to limitations in its video inpainting model when handling large missing areas. ViewCrafter, while avoiding color discrepancies, also suffers from blurry outputs, as the input masked videos contain dynamic objects. In contrast, our method generates significantly clearer results with minimal deviation from the ground truth, demonstrating the superior capability of our video inpainting model in generating warped multi-view videos.

Method	PSNR ↑	SSIM ↑	LPIPS ↓	FVD ↓
CoCoCo	24.98	0.8213	0.0987	**12.58**
StereoCrafter	23.44	0.8006	0.1498	52.63
ViewCrafter	16.95	0.5822	0.2560	52.07
Ours	**27.22**	**0.8223**	**0.0801**	14.30

Table 9. **Quantitative comparison of novel view-aware video in-painting** on our processed $5K$ videos from OpenVid-1M.

Backbone	mPSNR ↑	mSSIM ↑	mLPIPS ↓
4D GS	16.16	0.4676	0.5164
Motion Field (ours)	**16.80**	**0.5170**	**0.4750**

Table 10. **Quantitative ablation study on reconstruction backbones** on iPhone dataset and HyperNeRF dataset.

Quantitatively, our method outperforms all other approaches, including transformer-based ProPainter and the NVS models NVS-Solver and TrajAttention. ProPainter is typically trained on random or object masks, resulting in a domain gap when addressing view-change-induced occlusions in 4D reconstruction. The generative NVS models fail to synthesize spatial-temporal consistent images required for 4D reconstruction, as illustrated in Fig. 11. On the contrary, we recast view augmentation as video inpainting with tailored training data and model, leading to more robust augmented views and higher-quality reconstructions.

Time Cost Analysis: Our video model takes about 20 seconds to generate a 16-frame clip on a single GPU. In our experiments, for a scene with up to 81 frames, one iteration takes around 2 minutes. With $N = 6$ augmentations, the total time is 12 minutes, which can be reduced by multi-GPU parallelism. This time consumption is much shorter than reconstruction, which typically takes at least 30 minutes.

7.2. Novel View-Aware Video inpainting

Dataset and Metrics: We compare the novel view-aware video inpainting task on our processed non-training $5K$ masked videos from OpenVid [55], where an anchor video is paired with each masked video. We employ PSNR, SSIM [82], LPIPS [106] and FVD [72] as the evaluation metrics for assessing inpainting quality.

Comparison Baselines: We choose CoCoCo [114], Stere-

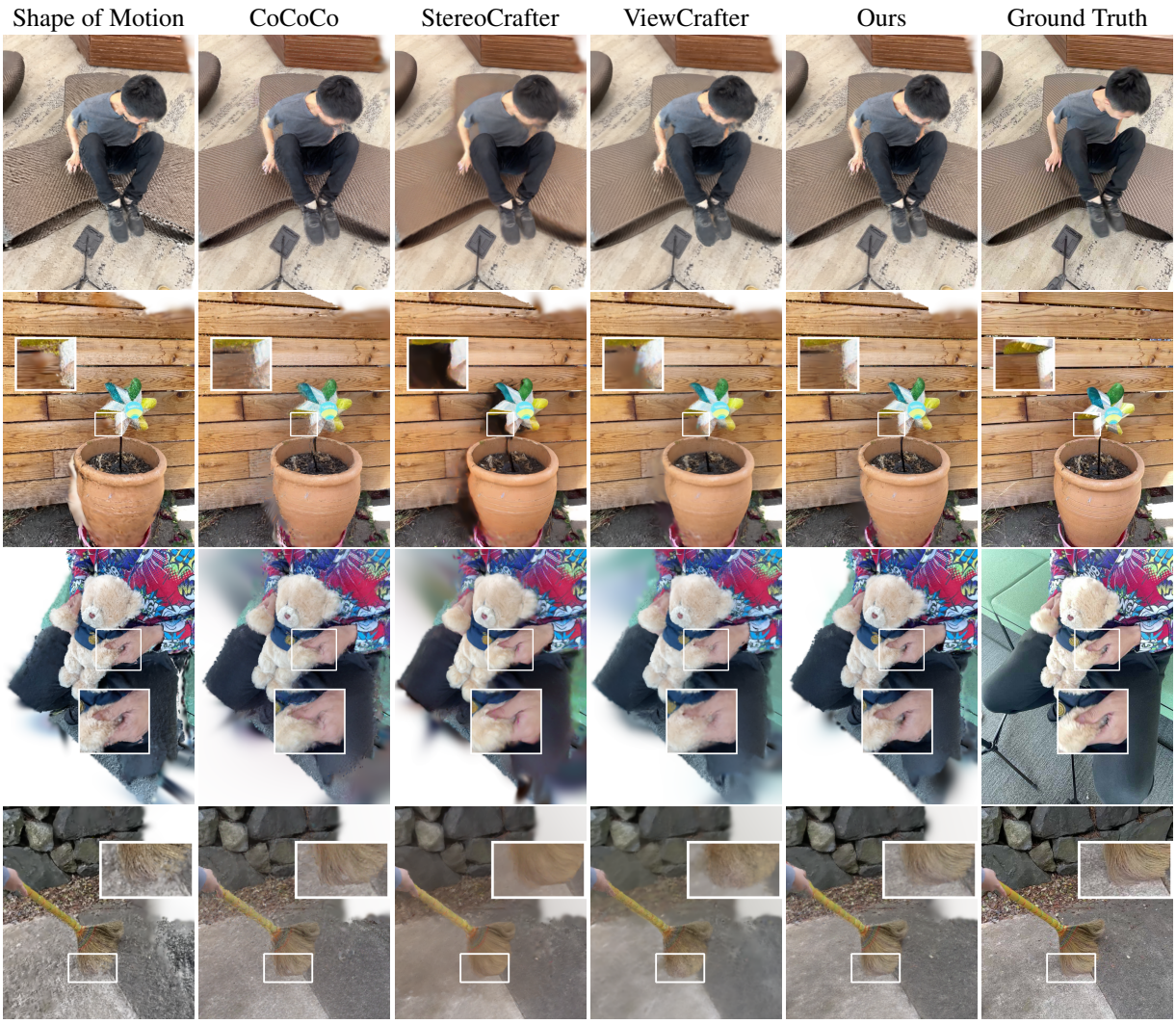

Figure 10. **Qualitative comparison of dynamic scene reconstruction** on iPhone dataset and HyperNeRF dataset.

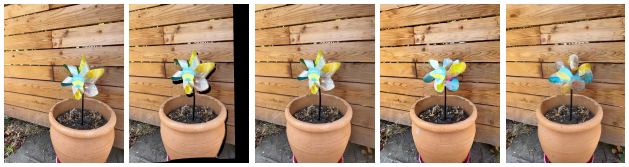

Figure 11. **Generative outcomes** without reconstruction.

oCrafter [108] and ViewCrafter [101] as our baselines. CoCoCo is a text-guided video inpainting diffusion model with strong consistency and controllability. StereoCrafter trains a video diffusion model for stereo video generation. Although ViewCrafter is a 3D-aware video inpainting method, we include it as a strong baseline due to its state-of-the-art performance in geometry-aware scene completion. For StereoCrafter, we provide the masked video and corresponding mask as input. For CoCoCo, we additionally supply a text prompt to guide the inpainting process, following its original setup. For ViewCrafter, we use the first frame from our anchor video as the image condition.

Results: The results are presented in Tab. 9 and Fig. 12. Among the baseline methods, CoCoCo demonstrates superior inpainting performance due to its integration with base image generation models. However, this approach tend to generates content that deviates from the scene context. In regions with weak contextual constraints, the results exhibit significant discrepancies from the ground truth, suggesting a tendency toward generation rather than faithful reconstruction. This observation is further supported by our method's higher PSNR/SSIM/LPIPS scores in quantitative evaluations. StereoCrafter exhibits limitations in handling large-scale holes due to its task orientation and training data constraints. Viewcrafter, originally designed for 3D inpainting, shows notable distortions and color inconsistencies when handling videos with dynamic objects.

In contrast, our method produces results that are most consistent with the ground truth, demonstrating that our

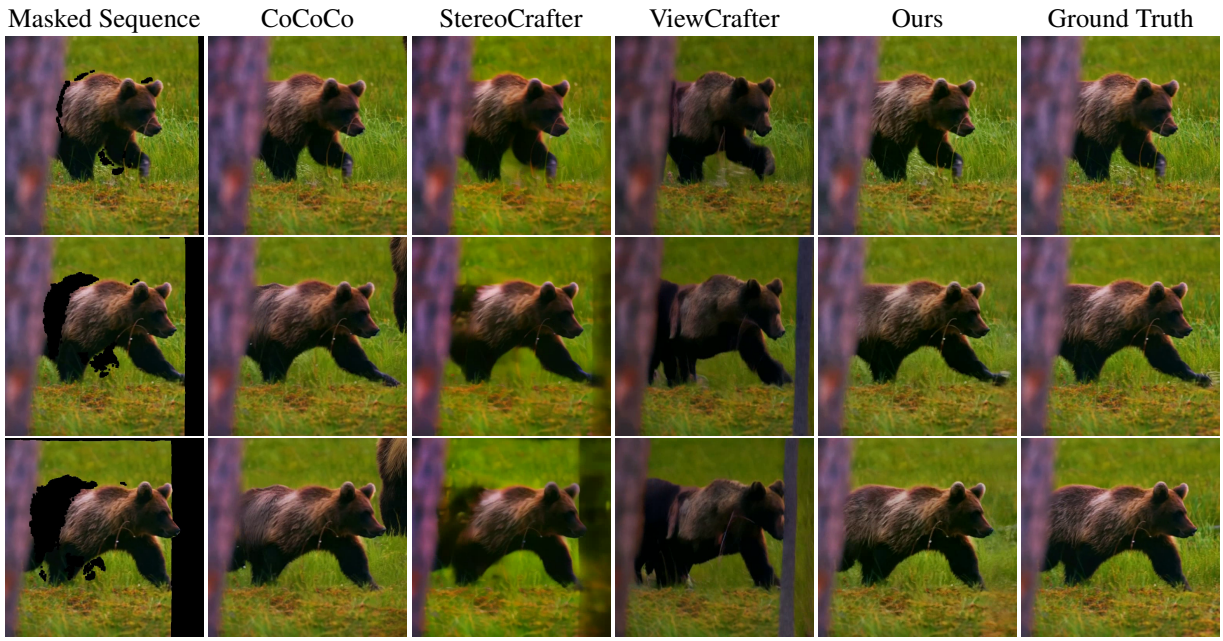

Figure 12. **Qualitative comparison of novel view-aware video in-painting** on our processed $5K$ videos from OpenVid-1M.

Figure 13. **Qualitative ablation study on reconstruction backbones** on iPhone dataset.

model effectively learns the spatiotemporal priors embedded in the anchor video. Additionally, leveraging an anchor video as a constraint provides crucial spatiotemporal priors and partial observations of occluded regions (due to dynamic foreground objects). This additional source of information enables more accurate inpainting and *reconstruction*, making our approach particularly well-suited for 4D pseudo-ground-truth generation. By incorporating natural geometric and temporal constraints, our method ensures that the inpainted results closely match the original video and maintain consistency across different views. Furthermore, subsequent 4D reconstruction experiments validate the necessity of our model design.

7.3. Ablation Study

Effect of 4D Reconstruction Backbones: We also evaluate the reconstruction backbone of our method. The results are illustrated in Tab. 10 and Fig. 13. With the supervision of multiview synchronized video, the results of 4D GS [96] have improved significantly but still do not reach the same level as motion field [78]. This discrepancy may be attributed to differences in their representations. The combination of the canonical GS with a motion field may offer greater robustness than extending 3D GS with an additional time dimension when using generated videos as supervision.

8. Limitations

We present some failure cases in Fig. 14. The main limitation of our proposed method is its dependency on the accuracy of poses and metric depth estimation. Although we utilize robust depth scale alignment to improve depth accu-

Figure 14. **Failure Cases.** Our method degenerates when the depth of the input video is not correctly obtained.

racy, as well as iterative view augmentation and IV loss to mitigate the impact of depth noise on warping and reconstruction, our results may still degrade in scenes where the monocular depth prediction is highly inaccurate.

In future work, we could combine our method with some recent dense reconstruction methods [43, 79, 104] to obtain more accurate scene geometry, as well as improving our diffusion models by improving model architecture to better synthesize fine details during the inpainting process. Additionally, extending our approach to handle more extreme camera motions and complex scene dynamics would further enhance its applicability to a wider range of scenarios.