Geo4D: Leveraging Video Generators for Geometric 4D Scene Reconstruction

Supplementary Material

In this **supplementary material**, we provide additional information to supplement our main submission. The **code** is available here for research purposes: `github.com/jzr99/Geo4D`

6. Implementation Details

6.1. Training Dataset

As shown in Tab. 5, we use five synthetic datasets for training: Spring [50], BEDLAM [2], PointOdyssey [119], TarTanAir [93], and VirtualKitti [6]. Although all datasets are synthetic, we found that some depth pixels are missing in PointOdyssey [119]. To address this, we apply max pooling to inpaint the missing pixels. During training, we sample each dataset according to the ratios in Tab. 5. For each sample, we select 16 frames from the sequence, with the sampling stride randomly chosen from $\{1, 2, 3\}$ to allow our diffusion model to adapt to input videos with various frame rates.

6.2. Optimization Details

The overall optimization process is outlined in Algorithm 1. We first predict all three modality maps using our diffusion model for each video clip g. The predicted point maps are then roughly aligned based on the overlapping frames using the Umeyama algorithm [84]. The camera intrinsic $\mathbf{K}^k$ is initialized by minimizing the projection error of the point map $\boldsymbol{X}^{k,g^k}$ in its reference (first) frame k within each window group g^k. The camera extrinsics are then initialized using the RANSAC PnP algorithm. In the first stage of optimization, the point maps are roughly disentangled into camera pose and depth map. The disparity map is then aligned with the global depth inferred from point maps by solving Eq. (5) from the main paper to obtain the scale and shift parameters. The camera parameters extracted from the predicted ray map are aligned with the global camera trajectory based on the reference (first) frame of each video clip g via Eq. (8) from the main paper. After initializing all the alignment learnable parameters, including rotation $\mathbf{R}^g_*$, scale λ^g_*, and shift $\boldsymbol{\beta}^g_*$ across different modalities, where $* \in \{\mathrm{p}, \mathrm{d}, \mathrm{c}\}$, we jointly optimize all the learnable parameters by Eq. (10). Specifically, we set the weights for each loss term in Eq. (10) as $\alpha_1 = 1, \alpha_2 = 2, \alpha_3 = 0.005, \alpha_4 = 0.015$ to roughly equalize the scale of the different losses.

Algorithm 1 Multi-Modal Alignment Optimization

```
1:  $\boldsymbol{X}^{i,g}, \boldsymbol{D}^{i,g}, \boldsymbol{r}^{i,g} \leftarrow$ Predicted by our diffusion model
2:  $\boldsymbol{D}^i_\mathrm{p}, \lambda^g_\mathrm{p}, \boldsymbol{R}^g_\mathrm{p}, \boldsymbol{\beta}^g_\mathrm{p} \leftarrow$ Initialized by Umeyama algorithm
3:  $\mathbf{K}^k_\mathrm{p} \leftarrow$ Optimized from $\mathbf{X}^{k,g^k}$
4:  $\mathbf{R}^i_\mathrm{p}, \mathbf{o}^i_\mathrm{p} \leftarrow$ Initialized by Ransac PnP from pointmaps $\boldsymbol{X}^i$
5:  $\mathbf{R}^{i,g}_\mathrm{c}, \boldsymbol{o}^{i,g}_\mathrm{c} \leftarrow$ Initialized by Eqs. (6) and (7) from raymaps $\boldsymbol{r}^{i,g}$
6:  repeat
7:      if Iteration = Align start iteration then
8:          $\lambda^g_\mathrm{d}, \beta^g_\mathrm{d} \leftarrow \arg\min \mathcal{L}_\mathrm{d}$ ( Eq. (5))
9:          $\boldsymbol{R}^g_\mathrm{c}, \lambda^g_\mathrm{c}, \boldsymbol{\beta}^g_\mathrm{c} \leftarrow \arg\min \mathcal{L}_\mathrm{c}$ ( Eq. (8))
10:     else if Iteration $<$ Align start iteration then
11:         $\boldsymbol{D}^i_\mathrm{p}, \mathbf{K}^i_\mathrm{p}, \mathbf{R}^i_\mathrm{p}, \boldsymbol{o}^i_\mathrm{p}, \lambda^g_\mathrm{p}, \boldsymbol{R}^g_\mathrm{p}, \boldsymbol{\beta}^g_\mathrm{p}, \leftarrow \arg\min \mathcal{L}_\mathrm{p} + \mathcal{L}_\mathrm{s}$
12:     else
13:         $\boldsymbol{D}^i_\mathrm{p}, \mathbf{K}^i_\mathrm{p}, \mathbf{R}^i_\mathrm{p}, \boldsymbol{o}^i_\mathrm{p}, \lambda^g_*, \boldsymbol{R}^g_*, \boldsymbol{\beta}^g_* \leftarrow \arg\min \mathcal{L}_\mathrm{all}$
14:     end if
15: until max loop reached
```

Dataset	Scene type	#Frames	#Sequences	Ratio
PointOdyssey [119]	Indoors/Outdoors	200K	131	16.7%
TartanAir [93]	Indoors/Outdoors	1000K	163	16.7%
Spring [50]	Outdoors	6K	37	16.7%
VirtualKITTI [6]	Driving	43K	320	16.7%
BEDLAM [2]	Indoors/Outdoors	380K	10K	33.3%

Table 5. **Details of training datasets.** Our method only uses synthetic datasets for training.

Steps	Video Depth		Camera Pose		
	Abs Rel ↓	$\delta < 1.25$ ↑	ATE ↓	RPE trans ↓	RPE rot ↓
1	0.221	70.7	0.234	0.072	0.753
5	**0.205**	**73.5**	**0.185**	**0.063**	0.547
10	0.207	73.2	0.212	0.071	**0.508**
25	0.220	72.2	0.211	0.074	0.564

Table 6. **Ablation study for the DDIM sampling steps.** on the Sintel [5] dataset.

7. Additional Analysis

7.1. Ablating the Number of Denoising Steps

We study the influence of the number of denoising steps during inference. As shown in Tab. 6, the model achieves optimal performance after around 5 steps. Compared to the video generation task, where a larger number of denoising steps usually produces a more detailed generated video, 4D reconstruction is a more deterministic task, which requires fewer steps. Similar phenomena are also observed in [22], which uses a video generator for video depth estimation.

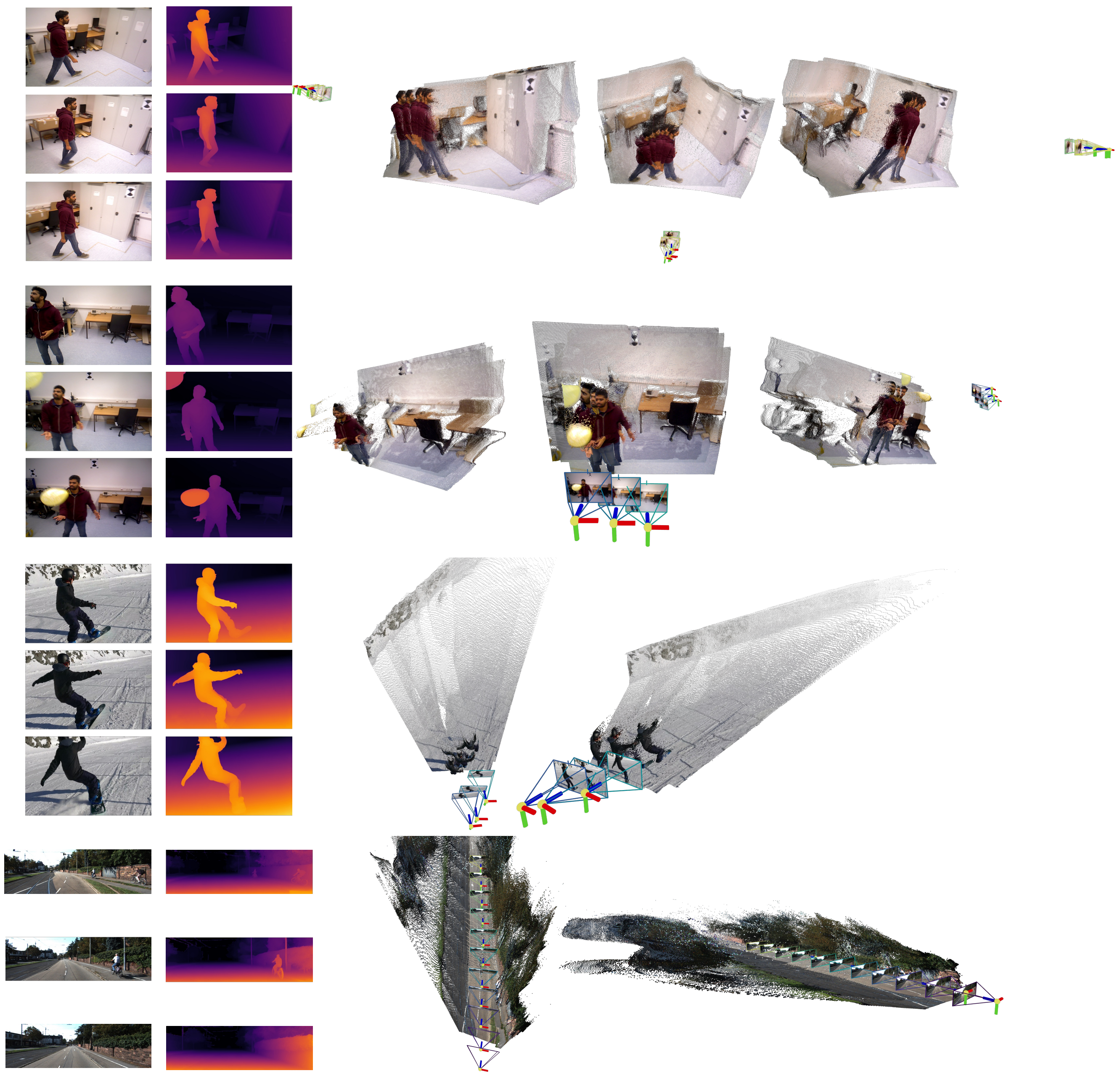

Figure 5. **Additional qualitative results.** Our method generalizes well to various scenes with different 4D objects and performs robustly against different camera and object motions.

Method	Video Depth		Camera Pose		
	Abs Rel ↓	$\delta < 1.25$ ↑	ATE ↓	RPE trans ↓	RPE rot ↓
w/o fine-tuned	0.212	72.1	0.192	**0.061**	0.577
w fine-tuned	**0.205**	**73.5**	**0.185**	0.063	**0.547**

Table 7. **Ablation study for the fine-tuned point map VAE** on the Sintel [5] dataset. The fine-tuned point map VAE performs better than the original one.

7.2. Ablation Study for Fine-Tuned Point Map VAE

As stated in the main paper, we added an additional branch to predict the uncertainty for our point map VAE and fine-tuned it based on Eq. 3. We perform an ablation study on our fine-tuning strategy. As shown in Tab. 7, our fine-tuned point map VAE achieves consistently better performance on both video depth estimation and camera pose estimation tasks compared with the original pre-trained image VAE,

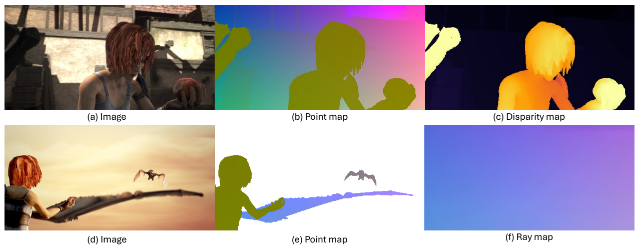

Figure 6. **Visualization of different geometric modality maps.**

demonstrating the necessity and effectiveness of our fine-tuning strategy.

7.3. Analysis of Multi-Modal Representation

Point maps (PMs) and disparity maps (DMs) are complementary. DMs better represent near objects, while PMs are more depth-agnostic (*e.g.*, human vs house in Fig. 6 (b,c)). As in prior work, DMs are affine invariant (which here makes them range-compatible with the pretrained RGB VAE); their scale and shift, needed to recover undistorted geometry, are inferred by matching them to the predicted PMs. Ray maps (RMs) help infer the camera pose when PMs fail to represent points at infinity (such as the sky in Fig. 6 (e)). We observed that PMs tend to be noisier than DMs, so we prioritized modeling the PMs' uncertainty. Per-pixel uncertainty for ray maps are less meaningful given the high degree of correlation between individual rays. During multi-modal alignment, we align global point clouds with DMs in disparity space and with PMs in linear space. This naturally gives more weight to near points, which tend to be estimated well by DMs, and weighs points based on uncertainty with PMs, thus taking advantage of both modalities.

8. Visualization

Figure 5 shows additional visualizations for indoor, outdoor, and driving scenes. Although our model is only trained on synthetic datasets, it generalizes to real-world data with diverse objects and motions.

9. Limitations

Although our method performs well and generalizes to a wide range of in-the-wild videos, it can struggle in cases involving significant changes in focal length or extreme camera motion throughout a sequence. This limitation likely stems from the lack of focal length variation in our training data. Incorporating more sequences with diverse camera movements and zooming effects could help mitigate this issue. Moreover, due to the inherent temporal attention mechanism in our network architecture, our approach currently supports only monocular video input. Extending the method to handle multi-view images or videos is a promising direction for future work.