V-DPM: 4D Video Reconstruction with Dynamic Point Maps

Supplementary Material

Figure 8. **Point map fusion:** the point maps of two images with significant motion are fused into the same time frame.

6. Training details

Each training batch contains windows of frames randomly sampled from our dataset mixture. We choose the central frame in the sampled snippet as the reference view that defines the coordinate system for multi-view reconstruction with the VGGT backbone. As in VGGT, we randomise the length of the video snippet during training, which we found helps reconstruct longer and more complex motions. Specifically, for each batch we sample a 5-, 9-, 13-, or 19-frame window. To utilise the hardware more efficiently, we dynamically select the batch size depending on the snippet length: a window of length 5 allows for a batch size of 4, whereas a 19-frame snippet can fit in VRAM only with a batch size of 1.

We train our final model on 16 GH200 GPUs for 60 epochs. During each epoch, we sample the following number of examples from each dataset: 5000 from Kubric-G, 5000 from Kubric-F, 15000 from PointOdyssey, 2500 from Waymo, 2500 from ScanNet++, and 2500 from BlendedMVS. We use the AdamW optimiser with a base learning rate of 1.5×10^{-4} and a cosine decay schedule.

Our dynamic point map reconstruction loss is defined for each pixel in each frame of every video snippet in the batch. Naively averaging the loss across all valid pixels (i.e., those for which we have annotations) can lead to problems. In particular, datasets with 4D annotations such as PointOdyssey often contain only sparse ground-truth 3D point tracks. When averaging the loss across all points in the batch, the numerous annotated points from static 3D datasets can easily dominate the sparse dynamic 3D annotations from the synthetic training set. As a result, the parts of the neural network responsible for dynamic reconstruction receive relatively small gradient updates. To mitigate this, we propose the following normalisation scheme: we first average the loss within each example and then compute the average across the batch dimension. This ensures that the magnitude of the loss is comparable across training samples. We found this improves the accuracy of dynamic reconstruction.

7. Network design ablation

We train a smaller run of 35 epochs to test different design choices for the network architecture. We compare four variants of the network design: (i) *Original*, (ii) *Decoder depth 2*, (iii) *Addition conditioning*, and (iv) *DPT decoder*. The *Original* is our complete model with four transformer blocks for decoding time-invariant point maps. In *Decoder depth 2*, we reduce the number of transformer blocks to two. In *Addition conditioning*, instead of using adaLN for time conditioning, we add the time token to the input tokens. In *DPT decoder*, we use no extra transformer layers for time-invariant decoding; instead, we make a copy of the DPT head and condition it directly through adaLN.

We evaluate the dynamic point map reconstruction on two views with a margin of 8 on the Kubric-G dataset; see Sec. 4.1. The results verify the importance of each design element for the full performance of the model.

	$P_0(t_1)$	$P_1(t_0)$
V-DPM: Original	**0.0500**	**0.0472**
V-DPM: Decoder depth 2	0.0518	0.0476
V-DPM: Addition conditioning	0.0524	0.0484
V-DPM: DPT decoder	0.0538	0.0502

8. Fusion

We show in Fig. 8 the point maps of two images fused into the same time frame, obtained via a forward pass of our network. This corresponds to the time-invariant point maps illustrated as a column in Fig. 3.