Spatia: Video Generation with Updatable Spatial Memory

Supplementary Material

1. More Implementation Details

Reference Frame Retrieval. In Section 3.1.2 of the main paper, we describe how to select up to K spatially relevant frames (with $K = 7$ by default) from the candidate-frame set. The complete procedure is provided in Algorithm 1.

Augmentation of Preceding-Frame Latents. Spatia conditions on preceding frames to generate future frames, enabling long-horizon video generation. However, while training uses ground-truth preceding frames as conditions, inference relies on model-generated frames, creating a distribution gap between training and inference. To alleviate this mismatch, we introduce a simple augmentation strategy for preceding-frame latents during training. Specifically, we sample a timestep $t_{\text{aug}} \in [0, 50]$ from a low-noise interval using the same noise scheduler as in Flow Matching training, and add the corresponding noise to the clean preceding-frame latents. The resulting augmented latents are then used as the conditioning inputs in place of the clean latents.

Match Accuracy. In Tables 3 and 5 of the main paper, we include *Match Accuracy* as an additional metric to assess the effectiveness of the memory mechanism in closed-loop video generation, where the last frame is expected to reproduce spatially similar content to the initial frame. *Match Accuracy* quantifies the structural and spatial correspondence between two frames. In our implementation, we use RoMa [1], a robust dense feature-matching algorithm, to estimate correspondences between the first frame $\boldsymbol{I}_{\text{first}}$ and the last frame $\boldsymbol{I}_{\text{last}}$. After obtaining the correspondence map, we discard low-confidence matches and count the remaining high-confidence correspondences as the number of valid matches. To ensure comparability across scenes, the final match accuracy is normalized by the number of high-confidence self-matches obtained by matching $\boldsymbol{I}_{\text{first}}$ with itself.

Dynamic-Static Disentanglement in the Inference Stage. Our model supports generating videos that contain dynamic entities while maintaining a spatial memory representing the static scene. During inference, to strictly enforce dynamic–static disentanglement, we first apply SAM2 [3] to track and segment dynamic entities in the initial conditioning image or previously generated video clips, and record their segmentation masks. These masks are then used to exclude dynamic regions when updating the spatial memory (i.e., the scene point cloud) with MapAnything [2].

Algorithm 1 Reference Frame Retrieval

Input: Target frames $\{\boldsymbol{T}\}^N$, candidate frames $\{\boldsymbol{C}\}^O$, view-specific scene point clouds $\{\mathcal{S}_{\boldsymbol{T}}\}^N$ and $\{\mathcal{S}_{\boldsymbol{C}}\}^O$, threshold ϵ, maximum number of reference frames K
Output: Retrieved reference-frame set $\{\boldsymbol{R}\}$

Initialize $\{\boldsymbol{R}\} \leftarrow \emptyset$
for each target frame $\boldsymbol{T}_i \in \{\boldsymbol{T}\}^N$ **do**
  if $i \bmod K \neq 0$ **then**
    break ▷ Operate every K frames.
  end if
  Initialize $s \leftarrow 0$ ▷ Maximal spatial overlap score.
  Initialize $\hat{\boldsymbol{R}} \leftarrow \emptyset$ ▷ Empty reference frame.
  Identify the scene map $\mathcal{S}_{\boldsymbol{T}_i} \in \{\mathcal{S}_{\boldsymbol{T}}\}^N$
  for each candidate frame $\boldsymbol{C}_j \in \{\boldsymbol{C}\}^O$ **do**
    Identify the scene map $\mathcal{S}_{\boldsymbol{C}_j} \in \{\mathcal{S}_{\boldsymbol{C}}\}^O$
    $s(\boldsymbol{T}_i, \boldsymbol{C}_j) \leftarrow \textsc{SpatialOverlap}(\mathcal{S}_{\boldsymbol{T}_i}, \mathcal{S}_{\boldsymbol{C}_j})$
    if $s(\boldsymbol{T}_i, \boldsymbol{C}_j) > s$ **then**
      $s \leftarrow s(\boldsymbol{T}_i, \boldsymbol{C}_j)$
      $\hat{\boldsymbol{R}} \leftarrow \boldsymbol{C}_j$
    end if
  end for
  if $s > \epsilon$ **then**
    $\{\boldsymbol{R}\} \leftarrow \{\boldsymbol{R}\} \cup \hat{\boldsymbol{R}}$
  end if
end for
return $\{\boldsymbol{R}\}$
function SpatialOverlap(x, y)
  $y' \leftarrow \text{Register}(y, x)$ ▷ Register y to x space.
  $s \leftarrow \text{3DIoU}(x, y')$
  return s
end function

2. Visualization

Qualitative Study on Spatial Memory in Long-Horizon Generation. In Table 4 of the main paper, we quantitatively study two key factors for enabling spatial memory and achieving spatially consistent long-horizon generation: (1) the use of reference frames and (2) the use of scene videos. Figure 1 presents a qualitative comparison among three variants: (1) our default model incorporating both components, (2) a model that uses only scene videos without reference frames, and (3) a model that uses reference frames but excludes scene videos. As shown, our full model substantially outperforms both ablated variants, successfully preserving global scene consistency and structural integrity over long temporal sequences, while the baselines exhibit pronounced geometric drift.

Closed-Loop Generation Figure 2 shows visualizations of closed-loop video generation. In these examples, the camera follows a trajectory that returns to the initial viewpoint

at the end of the sequence. This setup enables direct evaluation of both visual and geometric consistency by examining whether the final frame spatially aligns with the first frame, thereby validating the effectiveness of our spatial memory in preserving global scene structure.

Generation of Dynamic Entities while Maintaining Static Scenes. Our model supports dynamic–static disentanglement by representing only the static scene in the spatial memory. This is accomplished by removing dynamic entities from the estimated scene point cloud, which is used as the spatial memory, while the original videos containing dynamic entities are used as training targets. Figure 3 illustrates several examples showing the static-only spatial memory alongside the corresponding generated videos, where dynamic entities perform actions within the same scenes.

3D-Aware Interactive Editing Maintaining a scene point cloud as spatial memory and conditioning on it during video generation also enables 3D-aware interactive editing. As shown in Figure 4, manipulating the estimated scene point cloud—such as removing objects, adding new ones, or modifying object colors—leads to corresponding and accurate changes in the generated videos.

References

[1] Johan Edstedt, Qiyu Sun, Georg Bökman, Mårten Wadenbäck, and Michael Felsberg. RoMa: Robust Dense Feature Matching. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2024. 1

[2] Nikhil Keetha, Norman Müller, Johannes Schönberger, Lorenzo Porzi, Yuchen Zhang, Tobias Fischer, Arno Knapitsch, Duncan Zauss, Ethan Weber, Nelson Antunes, Jonathon Luiten, Manuel Lopez-Antequera, Samuel Rota Bulò, Christian Richardt, Deva Ramanan, Sebastian Scherer, and Peter Kontschieder. MapAnything: Universal feed-forward metric 3D reconstruction, 2025. arXiv preprint arXiv:2509.13414. 1

[3] Nikhila Ravi, Valentin Gabeur, Yuan-Ting Hu, Ronghang Hu, Chaitanya Ryali, Tengyu Ma, Haitham Khedr, Roman Rädle, Chloe Rolland, Laura Gustafson, Eric Mintun, Junting Pan, Kalyan Vasudev Alwala, Nicolas Carion, Chao-Yuan Wu, Ross Girshick, Piotr Dollár, and Christoph Feichtenhofer. Sam 2: Segment anything in images and videos. *arXiv preprint arXiv:2408.00714*, 2024. 1

Figure 1. Qualitative comparison of three variants for long-horizon video generation: (1) our default model Spatia, (2) a variant using only scene videos without reference frames, and (3) a variant using reference frames but no scene videos. The spatial memories shown in the figure are generated by Spatia.

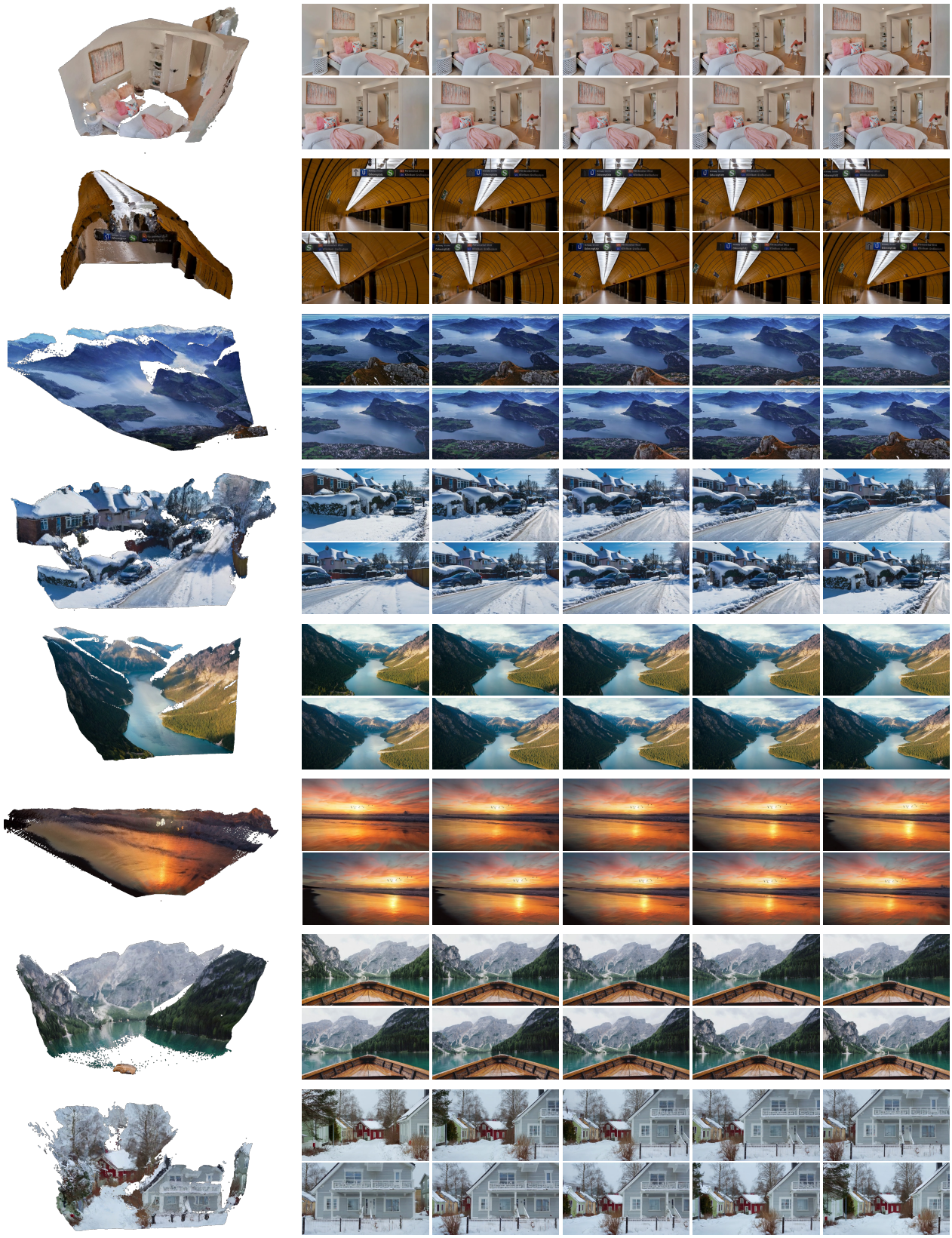

Figure 2. Visualization of closed-loop video generation. The camera follows a trajectory that returns to its initial viewpoint, enabling direct comparison between the first and final frames to evaluate the effectiveness of the spatial memory mechanism.

Figure 3. Visualizations of dynamic–static disentanglement. Our model maintains a spatial memory containing only the static scene point cloud while generating videos that include dynamic entities acting within the same scenes.

Figure 4. Demonstration of 3D-aware interactive editing. By directly modifying the spatial memory (i.e., the scene point cloud), users can achieve geometrically precise edits in the generated videos, such as removing an object (2nd row), adding a new object (3rd row), or altering object attributes (4th row).