




PolyOculus: Simultaneous Multi-view Image-based Novel View Synthesis (Supplementary Materials)

Jason J. Yu^{1,2}, Tristan Aumentado-Armstrong^{1,2}, Fereshteh Forghani¹,
Konstantinos G. Derpanis^{1,2,3}, and Marcus A. Brubaker^{1,2}

¹ York University

² Vector Institute for AI

³ Samsung AI Centre Toronto

This document provides additional material that is supplemental to the main submission. Section 1 outlines details of the ray encoding used to provide pose information to our model. Section 2 discusses the generalization of our method to numbers of views beyond those seen during training. Section 3 describes the method used to obtain novel views from the DFM baseline. Section 4 describes additional qualitative results for ground-truth trajectories that can be found in the supplemental webpage. Section 5 describes improvements for generations over cyclical motions. Section 6 describes improvements for generations over stereo pairs, where there is no natural ordering between the views of the pairs. Section 7 describes a heuristic for generating large sets of unordered views.

1 Camera Ray Encoding

To provide our model with access to the camera geometry, we adopt the ray representation used in previous works [1, 4]. Given the intrinsic matrix, \mathbf{K} , and the extrinsic matrix, $[\mathbf{R}|\mathbf{t}]$, of a camera, the projection matrix is defined as $\mathbf{P} = \mathbf{K}[\mathbf{R}|\mathbf{t}]$. The ray $\mathbf{r}_{u,v} = (\boldsymbol{\tau}, \mathbf{d}_{u,v})$ at pixel coordinates (u, v) is composed of the camera center $\boldsymbol{\tau} = -\mathbf{R}^{-1}\mathbf{t}$, and normalized direction $\mathbf{d}_{u,v}$. The unnormalized ray direction is given by:

$$\bar{\mathbf{d}}_{u,v} = \mathbf{R}^{-1}\mathbf{K}^{-1} [u \ v \ 1]^\top. \quad (1)$$

The rays, \mathbf{r} , are frequency encoded [3] into their final representation, \mathcal{R} , which is then used to condition the model:

$$\mathcal{R} = [\sin(f_1\pi\mathbf{r}), \cos(f_1\pi\mathbf{r}), \dots, \sin(f_K\pi\mathbf{r}), \cos(f_K\pi\mathbf{r})], \quad (2)$$

where K frequencies are used, which increment in frequency by powers of two.

2 Generalization to Number of Views

Computational constraints limit the number of views that can be used during training. In this section, we provide additional evaluations that investigate the

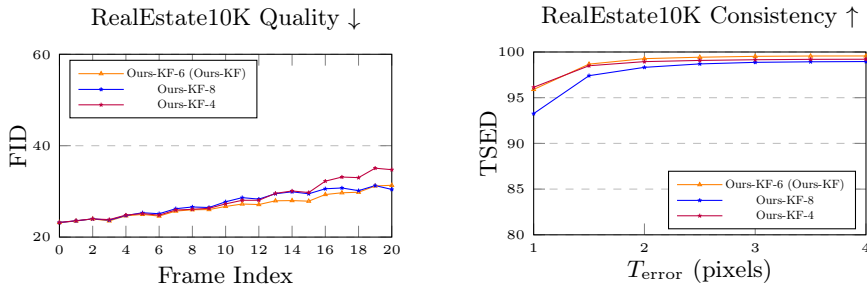


Fig. 1: Quality and consistency of our generations on RealEstate10k while varying the number total number of conditioning and generating views. Ours-KF- i denotes generation using keyframes where the maximum number of total views used for simultaneous generation is at most i . In all cases, the maximum number of conditioning views is $\frac{i}{2}$. Ours-KF-6 is the model presented in the main paper as Ours-KF. Note that performance is best using Ours-KF-6, which does not use a significantly larger number of views as seen during training, which would lower generated image quality as seen with Ours-KF-8, while having a lower *sampling depth* than Ours-KF-4, which reduces error accumulation from autoregressive generation.

impact of sampling using more views than those seen by our model during training.

First, we investigate the impact of the maximum *total* number of views used for our keyframing sampling approach. The total number of views is the sum of the number of conditioning and generated views. The keyframing results presented in the main paper use a maximum total of six (combined). In this section, we investigate higher and lower maximum totals of eight and four. In all cases, the maximum number of conditioning views is half that of the total number of views. The results shown in Fig. 1 show that increasing the maximum total views reduces both generated image quality (see, e.g., the FID of frames 12-16) and TSED consistency (especially for $T_{\text{error}} \leq 2$), while lowering the limit also reduces quality (see FID on frame 12 and higher), while slightly lowering consistency.

Second, we more directly analyse the impact of the number of generated frames to the quality of frames for a fixed frame index in a sequence. Given the ground-truth trajectories, we construct custom trajectories that vary the number of generated views, while conditioning on the first view in the sequence, which has a frame index of zero. We consider a minimum number of generated views of two, where we always use the ground-truth poses at frame indices three and six. When evaluating image quality as FID, we always measure the FID over the views at index six over all scenes. To increase the number of generated views beyond two, additional views are given poses that are the same as the pose at index six, but with an uniform random translation with a magnitude of 10% of the distance between the view at index six and three. The results in Fig. 2 show that the FID remains relatively unchanged for the number of generated views

up to those seen during training (i.e., four), but begins to increase noticeably beyond four generated views.

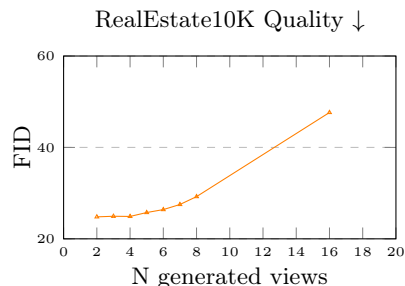


Fig. 2: Quality of images generated at framed index six, with respect to the number of simultaneously generated views. Additional views beyond two generated views are selected randomly in proximity of view index six. Beyond the number of views used during training, the quality begins to decrease as FID rises.

3 Rendering Novel Views from DFM

For evaluations comparing our method with DFM [2], we obtain novel views from DFM using the publicly available pretrained weights and code. Sampling with one target view (DFM-1) is readily supported using the available code base; however, multiple target views should be considered to ensure coverage over larger scenes. DFM supports autoregressive generation for a variable number of target views; however, few details are available on the best methodology for rendering the intermediate non-target views in this setting. In this section, we provide details on the generation and rendering method we use in our experiments for DFM-2 on RealEstate10K.

Given an observed image and the pose of a target view, DFM performs a generative diffusion process to generate the target view. For multiple targets, autoregressive generation is used, where each target is diffused sequentially while conditioning on past observed or generated target views. In our experiments with DFM-2, with two target views, we first run two iterations of autoregressive generation until all target views have been generated. DFM creates an intermediate NeRF representation for every target view generated. Our intermediate novel views (non-target views) are rendered from the intermediate NeRF after diffusing the final target view.

4 Additional Results: Ground-truth Trajectories

Additional qualitative results for ground-truth trajectories are provided with an interactive viewer on our [Offline Supplementary Webpage](#), for RealEstate10K

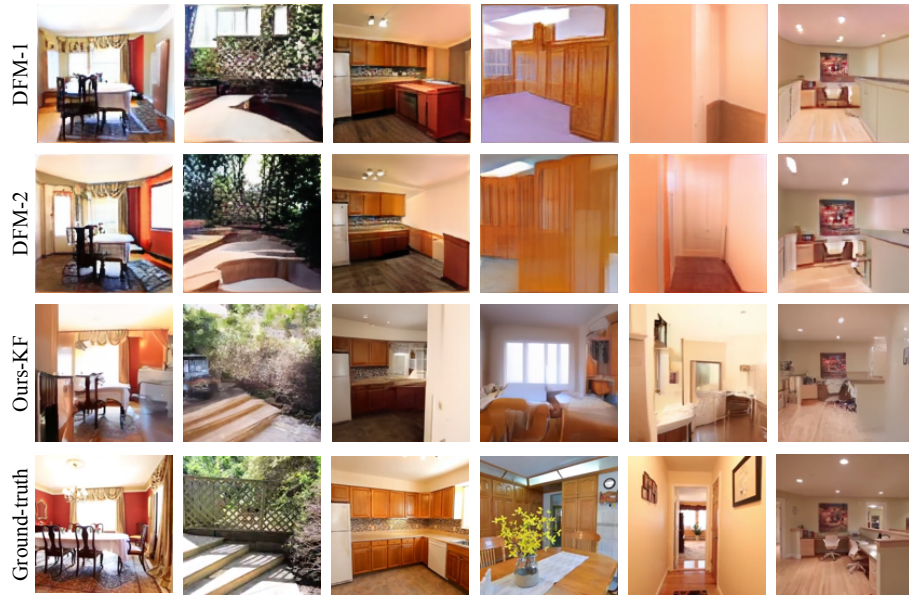


Fig. 3: Visualization of final frames of generated sequences on ground-truth trajectories comparing our method with DFM [2]. All images here are at the native resolution of DFM at 128×128 . The quality of images from our keyframed method is higher than all other methods.

and Matterport3D, under the **RealEstate10K Qualitative Results: Ground-truth Trajectories** and **Matterport3D Qualitative Results: Ground-truth Trajectories** headings. Additional results comparing with DFM [2] at a lower resolution are also provided on our **Offline Supplementary Webpage** under the **DFM Qualitative Results: Ground-truth Trajectories** heading. The viewer allows views along the trajectory to be inspected manually or with automated playback. Different scenes along with three different samples are available. Notice that the quality of the frames later in our keyframed generations tend to be of higher quality.

Last frame visualization for RealEstate10K. Additional qualitative visualizations of the last generated frames comparing our method with DFM [2] at 128×128 are provided in Fig. 3. This visualization allows the quality of the final frames to be inspected easily.

Last frame visualization for Matterport3D. Additional qualitative visualizations of the last generated frames on Matterport3D are provided in Fig. 4. Similar to the qualitative results on RealEstate10k in the main paper, the quality of the final frames on Matterport3D generated using our keyframing method are superior to the baselines.

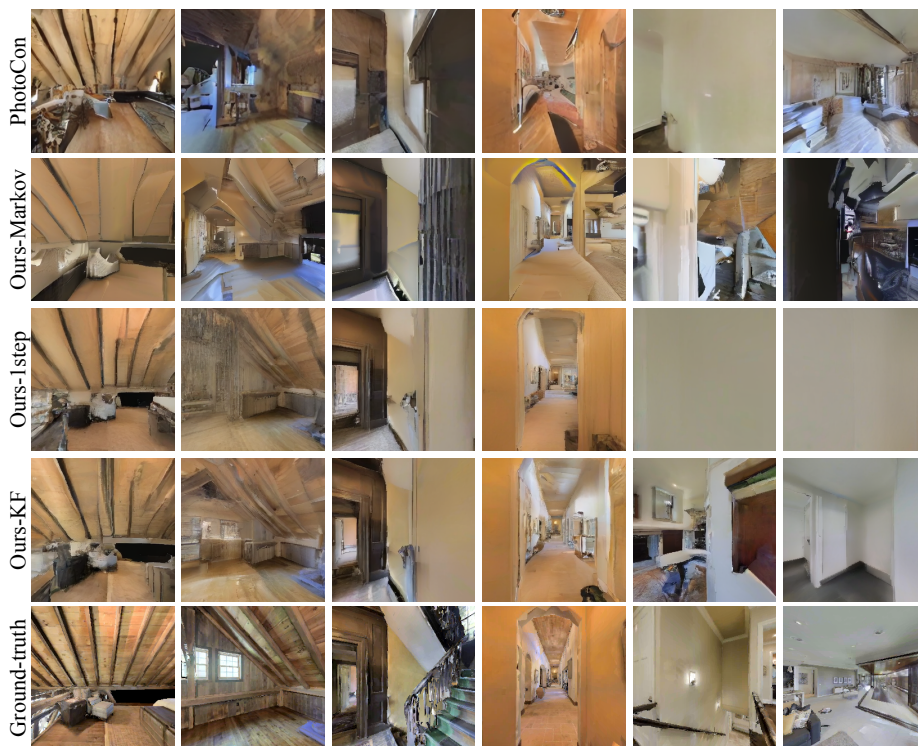


Fig. 4: Visualization of final frames of generated sequences on ground-truth trajectories on Matterport3D. The quality of images from our keyframed method is higher than all other methods.

5 Additional Results: Cyclical Trajectories - Spin

Additional qualitative results for the *spin* trajectory are provided with an interactive viewer on our **Offline Supplementary Webpage**, under the **Real-Estate10K Qualitative Results: Cyclical Trajectory - Spin** heading. This trajectory is of particular interest due to its cyclical structure; the views eventually travel back to the original position. Generating views with this trajectory using previous autoregressive Markov methods is challenging due to the limited conditioning window, which degrades the consistency between the last and first views, as expected.

The viewer provided on our **Offline Supplementary Webpage** loops the trajectory to make the inconsistency easier to see. There is also a **Loop First and Last** playback option to directly compare the first and last frame. Note the scene content changes significantly in the other methods, while our keyframed method is far more consistent.

6 Additional Results: Stereo Grouped View Generation

We provide an example stereo generation under **RealEstate10K Qualitative Results: Stereo Grouped View Generation** on our **Offline Supplementary Webpage**. These views contain stereo pairs along a trajectory, which do not have any natural ordering within the pairs. A naive baseline samples these views using a standard autoregressive method, where the right then the left view is sampled before repeating for the next stereo pair in the trajectory. Our set-based model is able to group the pairs and generate them simultaneously without imposing any ordering within the pairs. This allows the generated stereo pairs to maintain a stable disparity along the trajectory, which is challenging using previous methods.

7 Generating Large Sets of Unordered Views

In general, views within a set do not necessarily have a single natural ordering or any at all. In cases where the ordering may be completely arbitrary, a heuristic based on the proximity of camera poses can be used to order the views for sampling. Next, we describe one such simple heuristic. This is an interesting design space for future work.

Given a set of N camera poses, we iteratively grow a set of keyframes, Ω , as a small subset of all the frames. Starting with the pose of the given view, k_1 , we choose the next keyframe in Ω as a view that is not already a keyframe, and is furthest from any of the existing keyframes:

$$k_i = \arg \max_{\mathbf{c}_r \notin \{k_1, \dots, k_{i-1}\}} \min_{\mathbf{c}_s \in \{k_1, \dots, k_{i-1}\}} d(\mathbf{c}_r, \mathbf{c}_s), \quad (3)$$

where k_i is the i^{th} selected keyframe, and $d(\mathbf{c}_r, \mathbf{c}_s)$ quantifies a distance between the cameras. This heuristic is chosen to spread the keyframes out while having high coverage of the space occupied by the views. The remaining views are used as in-between frames with a generation order defined in a similar manner to the keyframe selection method in Eq. 3. The in-between frames are conditioned on a subset of the closest views that have already been generated (these may not necessarily be keyframes).

We provide an interactive viewer in our **Offline Supplementary Webpage** for a set of novel views generated along a fixed radius, under **RealEstate10K Qualitative Results: Alternative Sampling**. Specifically, the distance, $d(\mathbf{c}_r, \mathbf{c}_s)$, is set to the Euclidean distance between the camera origins. The exact algorithm used for keyframe selection is specified in Algorithm 1. The generation and conditioning scheme for keyframes and in-between frames is specified in Algorithm 2 and Algorithm 3, respectively. The interactive viewer organizes the views in a grid, where the horizontal and vertical position of the view selection squares correspond to the view azimuth and elevation. The viewer also includes a visualization to illustrate the order of generations and conditioning views used to sample the set of views.

Algorithm 1: Keyframe selection. \mathbf{C} is the set of all camera poses. N_k is the maximum number of keyframes. $d(\cdot, \cdot)$ measures the distance between camera poses. Ω is the set of keyframe camera poses, initialized with the observed view camera poses. The alternative sampling result in the **Offline Supplementary Webpage** uses $N_k = 37$ and $d(\cdot, \cdot)$ is the the Euclidean distance between camera origins.

Given $\mathbf{C}, N_k, d(\cdot, \cdot)$:

Let $\Omega \subset \mathbf{C}$ contain the camera poses of the observed views

Function FindClosest(A, B):

 | Let $a \in A$, such that $d(a, b)$ is minimized, over all $b \in B$

 | **return** a

end

Function FindFurthest(A, B):

 | Let $a \in A$, such that $d(a, \text{FindClosest}(B, \{a\}))$ is maximized

 | **return** a

end

while $|\Omega| < N_k$ **do**

 | Let $\mathbf{C}_{\text{remaining}} = \mathbf{C} / \Omega$

 | Set $\Omega \leftarrow \Omega \cup \{ \text{FindClosest}(\mathbf{C}_{\text{remaining}}, \Omega) \}$

end

Algorithm 2: Keyframe sampling and conditioning selection. Initially one keyframe is selected to be generated with two conditioning frames based on the proximity to available views (observed or already generated views). Additional keyframes may be selected to be simultaneously generated with this initial keyframe, which may add up to two conditioning views. When the total number of views ($|\Omega_{\text{gen}}| + |\Omega_{\text{cond}}|$) exceeds N_{softlim} , no additional keyframes are considered for generation. The alternative sampling result in the **Offline Supplementary Webpage** uses $N_{\text{softlim}} = 6$.

```

Given  $\Omega, N_{\text{softlim}}$ :
Let  $\Omega_{\text{avail}} \subset \Omega$  contain the observed views
Function FindClosest( $A, B$ ):
  | Let  $a \in A$ , such that  $d(a, b)$  is minimized, over all  $b \in B$ 
  | return  $a$ 
end

while  $\Omega \neq \Omega_{\text{avail}}$  do
  | Set  $\Omega_{\text{remain}} \leftarrow \Omega / \Omega_{\text{avail}}$ 
  | Set  $k \leftarrow \text{FindClosest}(\Omega_{\text{remain}}, \Omega_{\text{avail}})$ 
  | Set  $c_1 \leftarrow \text{FindClosest}(\Omega_{\text{avail}}, \{k\})$ 
  | Set  $c_2 \leftarrow \text{FindClosest}(\Omega_{\text{avail}} / \{c_1\}, \{k\})$ 
  | Set  $\Omega_{\text{gen}} \leftarrow \{k\}$ 
  | Set  $\Omega_{\text{cond}} \leftarrow \{c_1, c_2\}$ 
  | while  $|\Omega_{\text{gen}}| + |\Omega_{\text{cond}}| < N_{\text{softlim}}$  and  $\Omega \neq \Omega_{\text{avail}} \cup \Omega_{\text{gen}}$  do
  | | Set  $k_{\text{extra}} \leftarrow \text{FindClosest}(\Omega_{\text{remain}} / \Omega_{\text{gen}}, \Omega_{\text{cond}})$ 
  | | Set  $c_{\text{extra}1} \leftarrow \text{FindClosest}(\Omega_{\text{avail}}, \{k_{\text{extra}}\})$ 
  | | Set  $c_{\text{extra}2} \leftarrow \text{FindClosest}(\Omega_{\text{avail}} / \{c_{\text{extra}1}\}, \{k_{\text{extra}}\})$ 
  | | Set  $\Omega_{\text{gen}} \leftarrow \Omega_{\text{gen}} \cup \{k_{\text{extra}}\}$ 
  | | Set  $\Omega_{\text{cond}} \leftarrow \Omega_{\text{cond}} \cup \{c_{\text{extra}1}, c_{\text{extra}2}\}$ 
  | end
  | Sample views in  $\Omega_{\text{gen}}$  conditioned on  $\Omega_{\text{cond}}$ 
  | Set  $\Omega_{\text{avail}} \leftarrow \Omega_{\text{avail}} \cup \Omega_{\text{gen}}$ 
end

```

Algorithm 3: In-between frame sampling and conditioning selection. In-between frames are selected in a similar manner to keyframe selection, where in-between frames that are further from the generated frames are sampled first. Frames are generated individually while conditioning on N_{cond} views. The alternative sampling result in the **Offline Supplementary Webpage** uses $N_{\text{cond}} = 3$.

```

Given  $\mathbf{C}, \Omega, N_{\text{cond}}$ :
Let  $\mathbf{C}_{\text{between}} = \mathbf{C}/\Omega$ 
Let  $\mathbf{C}_{\text{avail}} = \Omega$ 
Function FindClosest( $A, B$ ):
  | Let  $a \in A$ , such that  $d(a, b)$  is minimized, over all  $b \in B$ 
  | return  $a$ 
end

Function FindFurthest( $A, B$ ):
  | Let  $a \in A$ , such that  $d(a, \text{FindClosest}(B, \{a\}))$  is maximized
  | return  $a$ 
end

while  $\mathbf{C}_{\text{avail}} \neq \mathbf{C}$  do
  | Let  $c_{\text{cur}} = \text{FindFurthest}(\mathbf{C}_{\text{between}}, \mathbf{C}_{\text{avail}})$ 
  | Set  $\Omega_{\text{cond}} \leftarrow \{\}$ 
  | for  $i \leftarrow 1$  to  $N_{\text{cond}}$  do
  |   | Set  $\Omega_{\text{cond}} \leftarrow \Omega_{\text{cond}} \cup \{ \text{FindClosest}(\mathbf{C}_{\text{avail}}/\Omega_{\text{cond}}, \{c_{\text{cur}}\}) \}$ 
  |   end
  | Sample view  $c_{\text{cur}}$  conditioned on  $\Omega_{\text{cond}}$ 
  | Set  $\mathbf{C}_{\text{avail}} \leftarrow \mathbf{C}_{\text{avail}} \cup \{c_{\text{cur}}\}$ 
  | Set  $\mathbf{C}_{\text{between}} \leftarrow \mathbf{C}_{\text{between}}/\{c_{\text{cur}}\}$ 
end

```

References

1. Sajjadi, M.S.M., Meyer, H., Pot, E., Bergmann, U., Greff, K., Radwan, N., Vora, S., Lucic, M., Duckworth, D., Dosovitskiy, A., Uszkoreit, J., Funkhouser, T., Tagliasacchi, A.: Scene representation transformer: Geometry-free novel view synthesis through set-latent scene representations. In: *CVPR (2022)*
2. Tewari, A., Yin, T., Cazenavette, G., Rezhikov, S., Tenenbaum, J.B., Durand, F., Freeman, W.T., Sitzmann, V.: Diffusion with forward models: Solving stochastic inverse problems without direct supervision. *NeurIPS (2023)*
3. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, Ł., Polosukhin, I.: Attention is all you need. *NeurIPS (2017)*
4. Yu, J.J., Forghani, F., Derpanis, K.G., Brubaker, M.A.: Long-term photometric consistent novel view synthesis with diffusion models. In: *ICCV (2023)*