EditableNeRF: Editing Topologically Varying Neural Radiance Fields by Key Points — Supplementary Materials —

1. Additional Details

Local coordinate systems for key points. The key point positions are transferred into local coordinate systems for normalization. For each key point, this local coordinate system takes the average key point position as the origin and scales the coordinates to ensure that the largest range of the three dimensions is 1, both based on the initialized positions. Without this normalization, the positional encoding functions [2] for key points will almost become linear functions when the positions vary in a small range. Besides, for some objects with complex geometries, our method may select duplicate key points on the same object. We can remove duplicate key points if the initialized local coordinates of two key points are always very similar to each other in the full sequence.

Network architecture and sampling. We present the detailed architecture of the warp field network in Fig. 6, the weight estimating network in Fig. 7, and the template NeRF network in Fig. 8. For the template NeRF network, we also use a coarse network and a fine network as in NeRF [2]. For each ray in volumetric rendering, we sample 128 points for the coarse network and 128 points for the fine network.

2. Intermediate Results

In this section, we show some additional intermediate results of our method.

Key point positions optimization. First, we show key point positions after training and those before training but after initialization. In Fig. 1, we can see that there are accumulative errors in the key point initialization, and these errors are successfully eliminated in our training stage. In addition, directly using the initialized key point positions without optimizing the positions in our training stage will cause artifacts in the results, as shown in Fig. 2.

Key point weights. Then we show the key point weights of two scenes in Fig. 5, one with two key points and one with three key points. These weights are obtained by using surface points corresponding to the pixels as query points. Our method can get correct weights automatically even when two parts are close to each other.

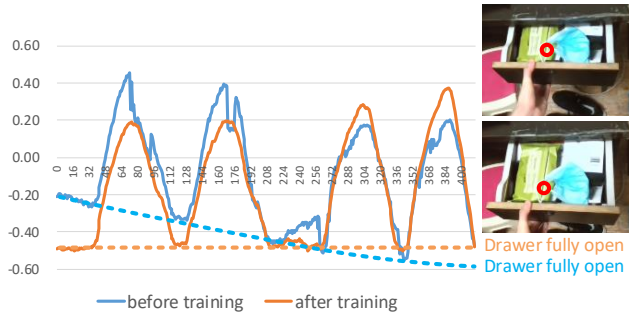

Figure 1. This data sequence contains pushing and pulling a drawer four times. The vertical coordinates here show the key points' coordinates along the direction of the drawer movements, and the horizontal coordinates are the frame indices. Dot lines indicate the key point coordinates when the drawer is fully open. We can see that after training, different frames in which the drawer is fully open have the same coordinate, and the accumulative errors in the initialization are eliminated. The two images on the right also show the key point positions after training.

Figure 2. Directly using the initialized key point positions without optimization will lead to artifacts.

Key point detection. We show more results on key point detection in a 2D version in Fig. 3. These 2D version results are obtained by rendering input frames in a fixed camera view, recording ambient coordinates of surface points corresponding to pixels, computing the variance in the whole sequence for each pixel, and selecting the pixels with local maximum variances after a 2D Gaussian filter.

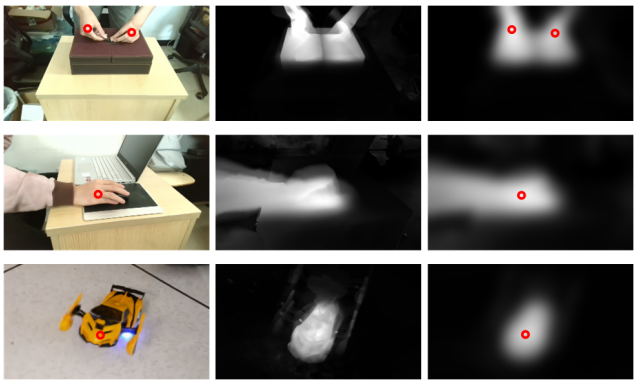

Figure 3. Key point detection results in 2D version. Left: input frames and corresponding key points; middle: variances of ambient coordinates; right: variances after a 2D Gaussian filter and selected key points.

Figure 4. Replacing our weighted key points with concatenated key points will lead to artifacts, such as in hands, cups, and shadows.

3. Additional Discussions and Experiments

Weighted key points vector. In our method, we use key point weights to integrate key point positions by linear combination and get a weighted key points vector, then we use this 3D vector to model topologically varying dynamics. A naive alternative method is to directly concatenate all the key point positions to form a vector with $3 \times N$ dimensions, where N is the number of key points, and replace our weighted key points with this concatenated vector for all query points. However, this concatenated vector will lead to artifacts. This is because our weighted key points vectors perform as ambient coordinates in hyperspace. In our method, the number of ambient dimensions is always three, while the concatenated vector has a higher number of dimensions, resulting in a high-dimensional hyperspace. Many areas in this high-dimensional hyperspace are far away from the training space, which leads to artifacts when the key points are edited into these areas, as shown in Fig. 4. Our method avoids this problem by using a fixed number of ambient dimensions and allowing different query points to have different key point weights.

Reconstruction quality. As our method slightly outperforms HyperNeRF on reconstruction quality, we also show ablation studies on reconstruction in Table 1. We evaluate the two losses (motion loss and geometry loss) and the initialization stage. These results indicate that this improvement is mainly due to the key point initialization. In our initialization, the frames with similar motions are initialized with similar key point positions, which leads to better results than the random initialization in HyperNeRF. The two losses have little effect on reconstruction quality, but note that the modeled scene can not be edited without these losses, and our method focuses on editing rather than improving the reconstruction quality.

Method	PSNR ↑	MS-SSIM ↑	LPIPS ↓
HyperNeRF [3]	42.67	0.9964	0.0823
Ours w/o init	42.36	0.9959	0.0828
Ours w/o 2 losses	44.51	0.9973	0.0803
Ours	44.35	0.9973	0.0808

Table 1. Ablation studies on reconstruction quality. We evaluate the two losses (motion loss and geometry loss) and the initialization stage.

Warp field and appearance latent code. Our warp field is used to compensate for the errors in the input camera parameters and distortions in the input images. Without this warp field, our method will try to compensate for these errors using the weighted key points model, leading to jitters in the key-point-based video editing results. And our appearance latent code is used to model the changes in exposure as in [3]. However, both the warp field and the appearance latent code are unnecessary for synthetic data, because it has ground truth camera models and parameters, and the exposure will never change.

4. Potential Social Impact

Our method is mainly designed for entertainment and creativity, like enabling end-users to generate a scene for playing a piece of music. While all the methods that enable editing may be potentially abused and have the risk of causing harm, such as deep fakes. However, it has been shown possible to detect deep fakes effectively using recent deep fake detection methods [1, 4–6] and reduce the risk of harm. And we believe our method can bring meaningful social benefits rather than potentially harmful applications.

References

[1] Liang Chen, Yong Zhang, Yibing Song, Lingqiao Liu, and Jue Wang. Self-supervised learning of adversarial example: Towards good generalizations for deepfake detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 18710–18719, 2022. 2

[2] B Mildenhall. Nerf: Representing scenes as neural radiance fields for view synthesis. In *European Conference on Computer Vision*, pages 405–421. Springer, 2020. 1

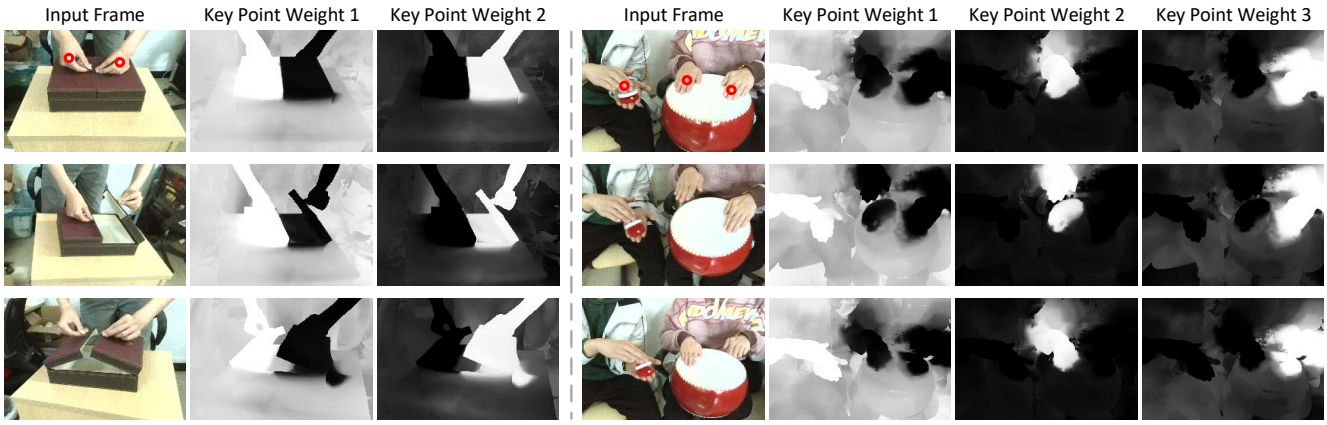

Figure 5. Key point weights of two scenes. The first input frame also shows the key points circled in red. Our method can get correct weights even when two parts are close to each other, as shown in the left scene.

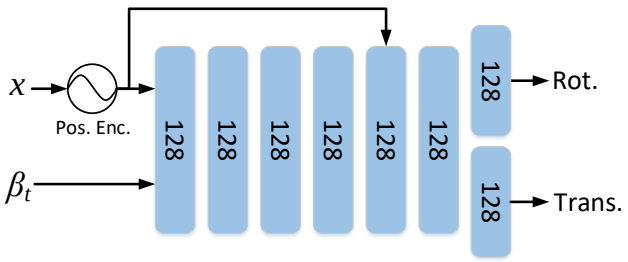

Figure 6. Warp filed network. Input with a query point x and a warp latent code β_t, the network calculates the rotation and the translation of this point, which will be used to deform x to its canonical coordinate x'. (Pos. Enc. means positional encoding.)

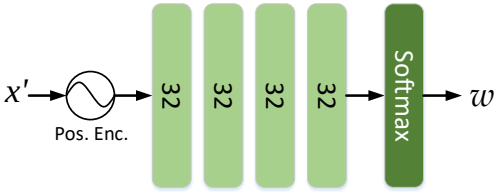

Figure 7. Weight estimating network. This network takes a canonical point x' as input and computes the key point weight vector w of this point.

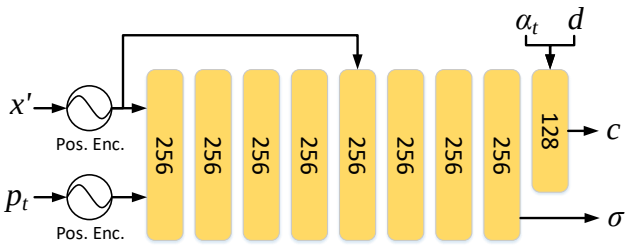

Figure 8. Template NeRF network. The input of this network is a canonical position x' and its corresponding weighted key points vector p_t, as well as an appearance latent code α_t and a view direction d. Then this network outputs the density σ and the view-dependent color c that are further used for volumetric rendering.

[3] Keunhong Park, Utkarsh Sinha, Peter Hedman, Jonathan T Barron, Sofien Bouaziz, Dan B Goldman, Ricardo Martin-Brualla, and Steven M Seitz. Hypernerf: a higher-dimensional representation for topologically varying neural radiance fields. *ACM Transactions on Graphics (TOG)*, 40(6):1–12, 2021. 2

[4] Zekun Sun, Yujie Han, Zeyu Hua, Na Ruan, and Weijia Jia. Improving the efficiency and robustness of deepfakes detection through precise geometric features. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3609–3618, 2021. 2

[5] Hanqing Zhao, Wenbo Zhou, Dongdong Chen, Tianyi Wei, Weiming Zhang, and Nenghai Yu. Multi-attentional deepfake detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2185–2194, 2021. 2

[6] Tianchen Zhao, Xiang Xu, Mingze Xu, Hui Ding, Yuanjun Xiong, and Wei Xia. Learning self-consistency for deepfake detection. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 15023–15033, 2021. 2