PARIS: Part-level Reconstruction and Motion Analysis for Articulated Objects Supplemental Materials

Jiayi Liu, Ali Mahdavi-Amiri, Manolis Savva
Simon Fraser University

In this supplement to the main paper, we provide implementation details (Appendix A) and additional quantitative and qualitative results (Appendix B).

A. Implementation Details

A.1. PARIS Network Architecture

Our part-level implicit representations are implemented using Instant-NGP [7] and NerfAcc [3] to enhance the approximation quality of NeRF [5] while minimizing the training time for each scene. Recall that we represent static and movable parts in separate neural fields. The static field and mobile field share the same network architecture. For each field $\mathcal{F}(\mathbf{x}, \mathbf{d})$, we leverage the multi-resolution hash encoding proposed by Instant-NGP to encode the point position $\mathbf{x}$ into a 16-dimensional feature, while the viewing direction $\mathbf{d}$ is encoded with spherical harmonics of order 4. We configure the hash grid to be in 16 levels with 2^{19} as the maximum entries per level (hash table size), and other hyper-parameters are using the default configurations recommended in the original paper [7]. Following the encoding layer, a two-layer fully-fused MLP of the *tiny-cuda-nn* framework [6] is integrated into the geometry network, while a one-layer fully-fused MLP is integrated into the texture network. All the activation functions used are ReLU. To further speed up the ray marching and volumetric rendering, we leverage NerfAcc to parallelize the procedure.

A.2. Training Details

All the results reported in our paper are trained within 300,000 iterations. This takes less than 30 minutes of training on a single NVIDIA GeForce RTX 3060. We optimize the motion parameters and neural radiance fields with two separate Adam [2] optimizers. We set the learning rate for the field optimizer at 0.002. For motion parameters, we set the learning rate at 0.01 for revolute joints and 0.005 for prismatic joints. Empirically, we observe a two-stage optimizing pattern during training. Within the first 5,000 iterations, the motion parameters converge to a point very close to ground truth while low-frequency details of the fields are jointly learned. As the motion stabilized, the high-frequency details of the fields are refined during later iterations. For this feature, we set different MultiStepLR schedulers for the two optimizers, where the learning rate for motion drops around 4 times faster after 5,000 iterations.

B. Additional Results and Analysis

B.1. More Results on Synthetic Data

Part segmentation and motion prediction. Figure 1 shows additional qualitative visualizations for the results reported in the paper. All these four examples are categories that were not used to train Ditto [1]. Ditto can separate reasonable part geometry for *Washer* and *Fridge*. However, as we demonstrated, the performance of Ditto significantly drops on unseen categories, especially for motion prediction. In contrast, our PARIS approach can predict more accurate segmentation and motion parameters regardless of the categories, and produces more geometric details even on thin structures (e.g. *Scissor*, *Blade*).

Novel view synthesis and articulation generation. Table 1 reports quantitative results for novel view synthesis (NVS) at two given states. We observe that our PARIS approach can produce comparable high-fidelity renderings with our baseline method within the same number of training iterations. *Ours-ICP* can produce higher quality renderings on average. This is not surprising since the baseline method learns one static object at a time without involving any transformation. Therefore, the baseline can converge faster and better within the same number of iterations compared to our composite NeRFs where joint motion optimization is involved We also show the rest of the qualitative results for articulation generation with ground truth states as the comparison in Figure 2.

B.2. Failure Cases

We discovered two failure modes in our approach which both occur on objects with a revolute joint. We show qualitative and quantitative results that illustrate these failure cases in Figure 3 and table 2. We also compare our approach with Ditto to demonstrate that these failure cases also impact prior work.

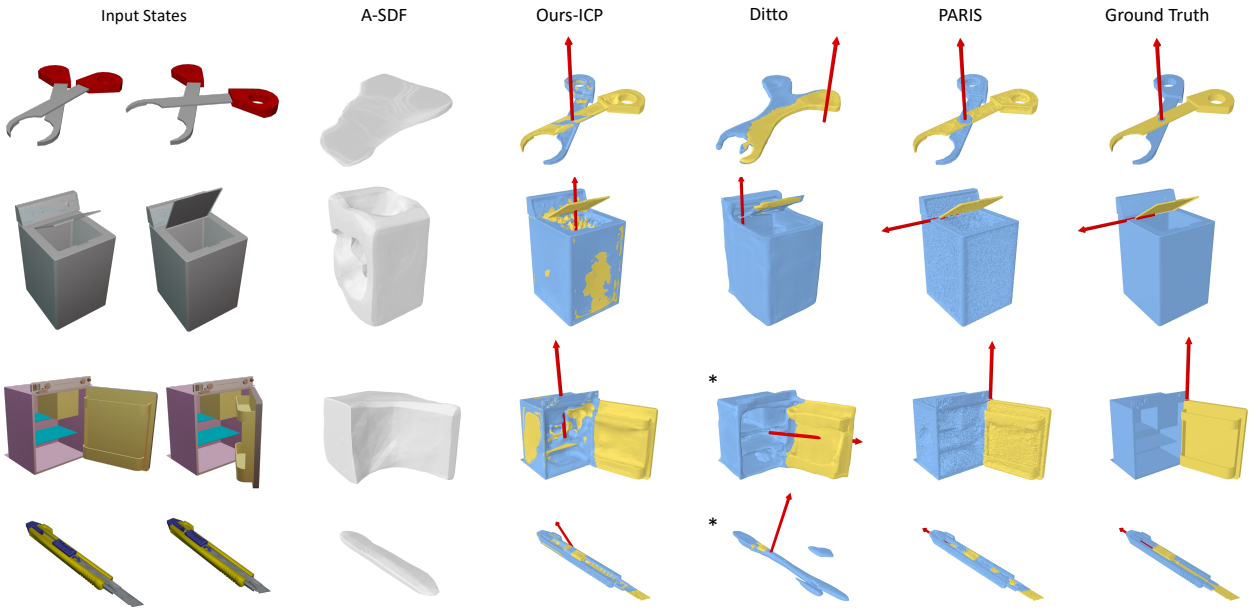

Figure 1. Additional qualitative results showing object and part reconstruction, as well as the predicted motion joint axis. Static parts are cblue while movable parts are yellow. All the categories above are 'unseen' for Ditto. There are significant axis prediction and segmentation errors. Moreover, $*$ denotes a wrong motion type prediction. In contrast, PARIS produces better segmentation with more geometric details while predicting an accurate motion axis across all the categories.

Metrics	Methods	Stapler	USB	Scissor	Fridge	FoldChair	Washer	Blade	Laptop	Oven	Storage	**Mean**
PSNR↑	Ours-ICP	**40.18**	**39.90**	**41.05**	37.68	**34.81**	40.12	**41.84**	**40.23**	33.52	36.72	**38.61**
	PARIS	38.59	38.24	38.70	**38.78**	33.11	**40.18**	38.56	38.66	**35.41**	**37.08**	37.62
SSIM↑	Ours-ICP	**0.997**	**0.995**	**0.997**	0.992	**0.990**	**0.992**	**0.998**	**0.993**	0.979	0.994	**0.993**
	PARIS	0.995	0.992	0.996	**0.994**	0.986	0.991	0.996	0.990	**0.981**	0.994	0.992

Table 1. Quantitative results of the rendering quality from novel views. Our PARIS trains two NeRFs compositely while jointly estimating motion, and our rendering quality is comparable to *Ours-ICP* which is trained on a completely static object for each state.

		Geometry			Motion		
Example	Method	CD-w↓	CD-s↓	CD-m↓	Ang Err↓	Pos Err↓	Geo Dist↓
ClosedFridge	Ditto	**3.17**	4.29	**1.75**	**1.43**	**0.10**	**17.26**
	PARIS	4.05	**3.59**	11.28	2.26	0.43	66.92
SymmetricChair	Ditto	1.65	41.52	107.03	86.30	0.06	**111.27**
	PARIS	**0.26**	**2.83**	**0.16**	**0.04**	**0.03**	180.01

Table 2. Quantitative results for failure cases on synthetic data. Each example illustrates one failure mode. *ClosedFridge* shows the results when one of the given states has severe occlusion. Our methods predict an incorrect joint axis and rotation angle, which also leads to an unsatisfactory segmentation for the movable part. Ditto is less sensitive to occlusion and produces better part surfaces, but the rotation angle is also inaccurate. *SymmetricChair* shows the results when the movable part is geometrically symmetric. We can produce high-quality output but the rotation angle is flipped due to the symmetry (see angle error of $\approx$ 180). Our method cannot resolve such symmetric ambiguities. Ditto also fails in this example and has quite high angle error.

Severe occlusion. If the movable part has severe occlusion across all views (e.g., the door of a refrigerator is fully closed in one of the given states), there is a high chance to produce a wrong motion estimation and unsatisfactory part segmentation. We demonstrate this failure mode with the *ClosedFridge* example. We observe that Ditto can predict better movable part reconstruction and joint axis in this situation, while PARIS can produce a cleaner static part. Our undesirable prediction of the movable part and motion is mainly due to the lack of correspondence points being observed from the input multi-view images. This leaves the neural fields with "blind views", which allows the closed door to be hidden inside the static part. This behavior can be better illustrated in the articulation generation process in Figure 4. Specifically, the predicted joint axis is outside of the object region, so that the door can be rotated into the body of the refrigerator to geometrically satisfy the multi-view observation. To address this issue, we can po-

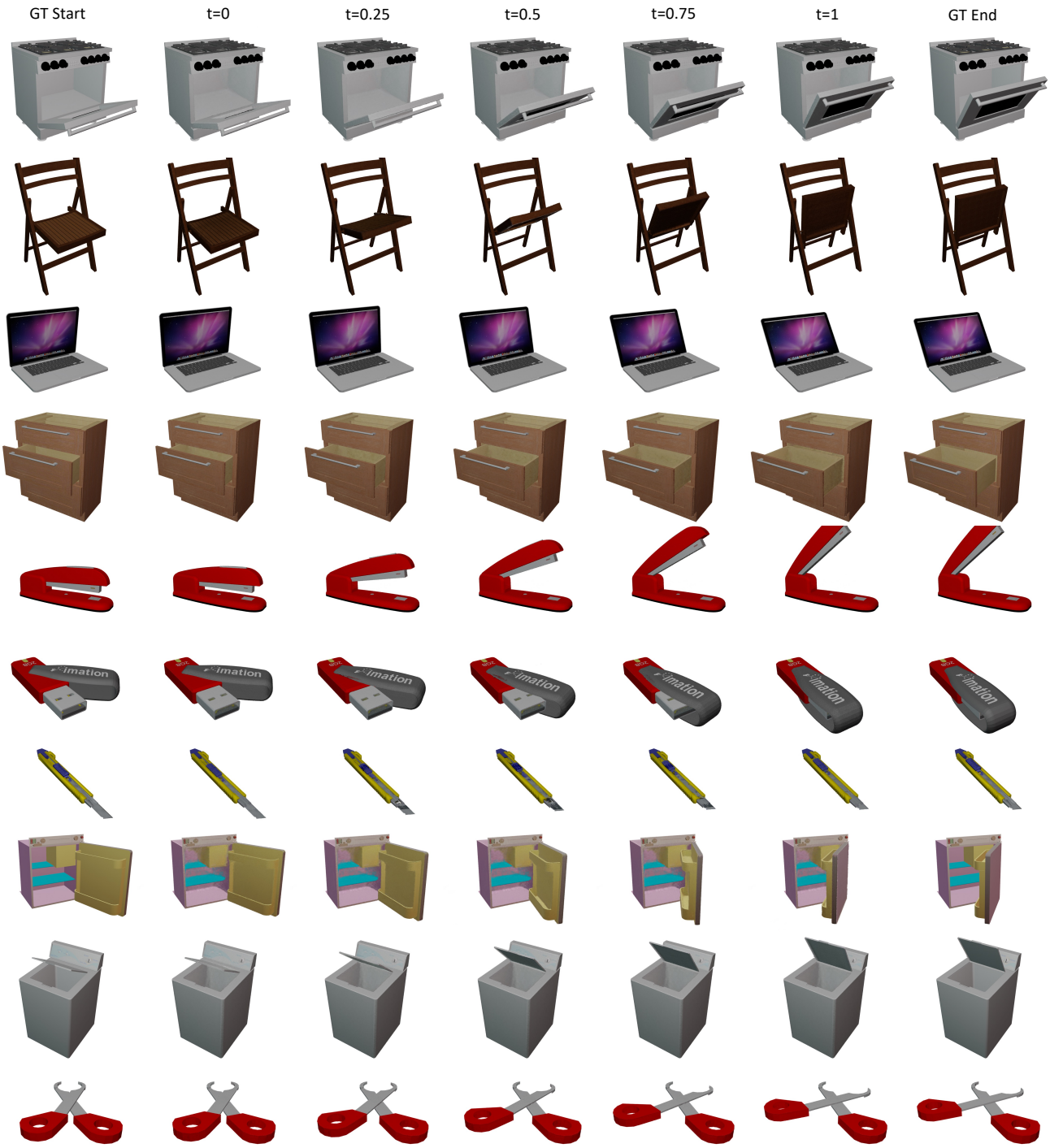

Figure 2. Qualitative results demonstrating articulation generation. The ground truth image inputs at given states are provided for comparison in the leftmost and rightmost columns. All the images are rendered from a novel view that is unseen during training. Our high-quality rendering results demonstrate that our predicted motion parameters and decoupling of the part appearance allow for effective generation of arbitrary unseen states.

tentially impose additional priors to the joint axis (e.g., the joint should be attached to the object instead of being allowed to float away from the geometry).

Ambiguities due to geometric symmetries. We might produce a wrong rotation angle if the movable part is geometrically symmetric even when the joint axis is predicted cor-

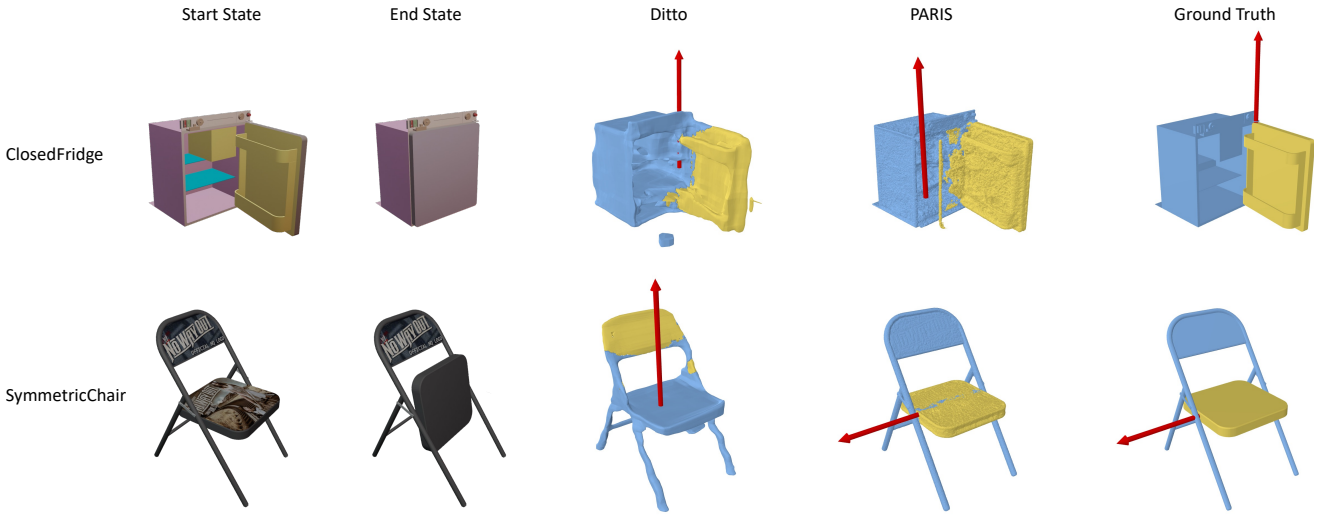

Figure 3. Qualitative results showing failure cases for geometry reconstruction and joint axis prediction. The *ClosedFridge* example in the first row illustrates that we fail to predict an accurate joint axis when one of the given states has severe occlusion (completely closed fridge, leaving interior unobserved). We also produce unsatisfactory part segmentation with wrong motion estimation. The *SymmetricChair* example in the second row shows the results for the situation when the movable part is geometrically symmetric. We can still predict an accurate joint axis with correct part surfaces in this case, but the rotation angle estimate is wrong (see Figure 4).

Figure 4. Articulation generation failure cases using our predicted motion with ground truth image at given states for comparison. The images are rendered from a novel view that is unseen during training. This figure illustrates how our predicted motion manipulates the movable part to satisfy the supervision at two given states in a wrong way.

rectly. We demonstrate this failure case with the *SymmetricChair* example. From Table 2, we can observe that we produce high-quality part segmentation and joint axis (low Ang Err and Pos Err). However, the geodesic angle distance reported is near 180 degrees (bottom right of table). This 180-degree difference happens because our prediction of the rotation angle from $t = 0$ to $t = 1$ is 80 degrees, while the ground truth is -100 degrees. This means we predicted an opposite rotation ending up with the same start and end states. Figure 4 illustrates how this opposite rotation manipulates the articulation in the wrong direction. This ambiguity is not surprising since we only have the object states in two end states as supervision, and the two solutions are both reasonable in terms of geometry. The only information that disambiguates the correct rotation direction is the texture information on the movable part (i.e. seat of the chair). The texture should stick to the corresponding face while rotating. However, the color MLP "hacks" the appearance to look correct. Therefore, we can potentially resolve this ambiguity by tightening the bonds between the geometry and appearance in the neural radiance field representation.

B.3. More Real Examples

For experiments on real data, apart from reconstructing the object directly from real RGB-D scan videos, we also collect multi-view images by rendering a real scan from the MultiScan [4] dataset as shown in the main paper. The ren-

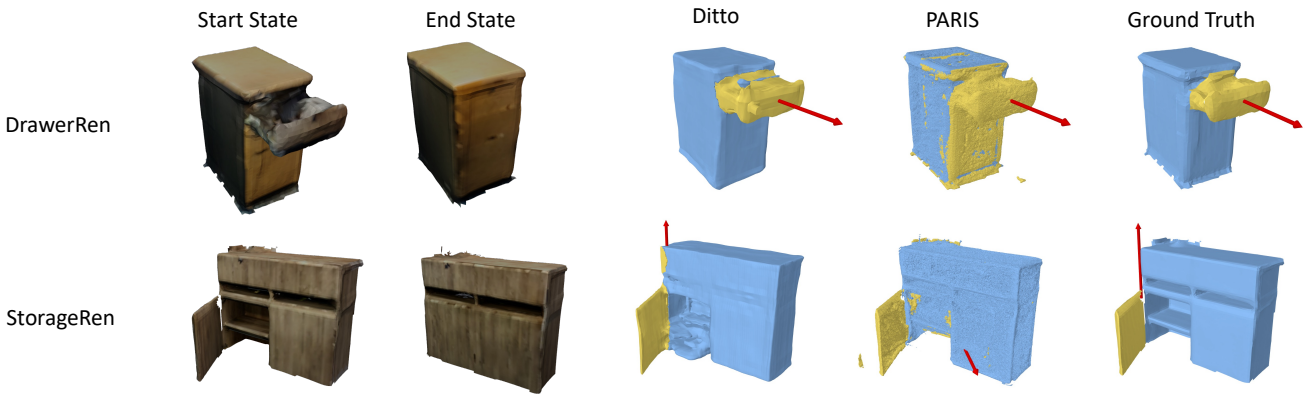

Figure 5. Qualitative results for the real examples. Overall, our methods can be applied to real cases, but perform less effectively than on synthetic data since our methods are sensitive to the error introduced in camera poses, reconstruction, and alignment. We fail motion estimation of the *StorageRen* case mainly due to the movable part having severe occlusion which is a limitation we cannot handle for now. Ditto performs well on these real examples.

		Geometry			Motion			
Example	Method	CD-w↓	CD-s↓	CD-m↓	Ang↓	Pos↓	Geo↓	Trans↓
DrawerRen	Ditto	**14.08**	16.09	**20.35**	5.88	-	-	0.38
	PARIS	15.05	**15.86**	179.52	**1.47**	-	-	**0.26**
StorageRen	Ditto	12.68	**15.06**	**12.18**	**1.12**	**0.95**	**10.61**	-
	PARIS	**12.58**	18.02	101.20	70.82	6.34	47.88	-

Table 3. Quantitative results for real cases. We produce comparable surfaces for the static part and object as a whole with Ditto, but less accurate surfaces for movable parts. We predict better motion estimation for the first two examples, but fail in the last example due to severe occlusion.

dering process is consistent with the configuration for synthetic data.

The evaluation metrics follow the description in the main paper: angular error (Ang), position error (Pos), geodesic distance (Geo) and translational error (Trans). We denote the two examples trained with rendering images as *DrawerRen* and *StorageRen*.

We show the qualitative results in Figure 5 and quantitative results in Table 3. The ground truth segmentation and motion information is annotated in the dataset or manually annotated using the tool provided by MultiScan.

For geometric reconstruction, we observe that our PARIS reconstructs comparable surface quality with Ditto for the whole object and static part, but a less accurate surface for the movable part. The large statistical errors come from "noise" hidden inside the static part. For motion estimation, we outperform Ditto in the translation case *DrawerRen*. However, PARIS fails in the other rotation case (*StorageRen*) because we cannot handle the situation where one input state has severe occlusion. Ditto performs better in this case since they do not suffer from this issue (see Appendix B.2). Overall, PARIS can be applied to real data but performs less effectively than on synthetic data due to sensitivity to errors from estimated camera poses, reconstruction artifacts, and end state alignment. A promising direction for future work is to increase robustness to such errors by jointly optimizing over estimated camera poses, and input end state alignment.

References

[1] Zhenyu Jiang, Cheng-Chun Hsu, and Yuke Zhu. Ditto: Building Digital Twins of Articulated Objects from Interaction. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5606–5616, 2022. 1

[2] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *Proceedings of the International Conference on Learning Representations (ICLR)*, pages 1–15, 2015. 1

[3] Ruilong Li, Matthew Tancik, and Angjoo Kanazawa. NerfAcc: A general NeRF acceleration toolbox. *CoRR*, abs/2210.04847, 2022. 1

[4] Yongsen Mao, Yiming Zhang, Hanxiao Jiang, Angel X Chang, and Manolis Savva. MultiScan: Scalable RGBD scanning for 3D environments with articulated objects. In *Advances in neural information processing systems*, 2022. 4

[5] Ben Mildenhall, Pratul P. Srinivasan, Matthew Tancik, Jonathan T. Barron, Ravi Ramamoorthi, and Ren Ng. NeRF: Representing scenes as neural radiance fields for view synthesis. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 405–421, 2020. 1

[6] Thomas Müller. tiny-cuda-nn, 4 2021. URL https://github.com/NVlabs/tiny-cuda-nn. 1

[7] Thomas Müller, Alex Evans, Christoph Schied, and Alexander Keller. Instant neural graphics primitives with a multiresolution hash encoding. *ACM Transactions on Graphics*, 41(4): 102:1–102:15, 2022. 1