Supplementary Material for NU-MCC: Multiview Compressive Coding with Neighborhood Decoder and Repulsive UDF

In this supplementary document, we first give implementation details on our network architectures, loss function, and color metrics in Section 1. We then provide some demonstration on NU-MCC's flexibility for high-resolution reconstruction and smoothing in Section 2. Finally, we present the scene reconstruction experiments in Section 3.

1 Implementation Details

1.1 Network Architectures

Encoder. The encoder consists of two ViTs, one for image and the other for point coordinates. Each ViT uses a 12-layer 768-dimensional "ViT-Base" architecture [1, 2]. The input image is resized to 224×224 and input points to 112×112.

Anchor predictor. The anchor predictor uses an 8-layer Transformer [1] with hidden and output dimensions of 512.

Feature aggregation. The MLP architecture and feature conditioning follow the decoder in [3] with five ResNet blocks and a hidden dimension of 512. We map the query point from $\mathbb{R}^3$ to $\mathbb{R}^{60}$ using frequency encoding [4] before feeding it into the MLP.

1.2 Loss Function

Anchor loss. We supervise the 3D locations of coarse anchor features $\mathbf{X}_c$ using L_1-chamfer distance loss to the ground truth points sampled using furthest point sampling (FPS), detailed as follows:

$$\mathcal{L}_{\text{anch}} = \frac{1}{|\mathbf{X}_c|} \sum_{\mathbf{x} \in \mathbf{X}_c} \min_{\mathbf{g} \in \mathcal{P}_{\text{GT,FPS}}} \|\mathbf{x} - \mathbf{g}\|_1 + \frac{1}{|\mathcal{P}_{\text{GT,FPS}}|} \sum_{\mathbf{g} \in \mathcal{P}_{\text{GT,FPS}}} \min_{\mathbf{x} \in \mathbf{X}_c} \|\mathbf{g} - \mathbf{x}\|_1 . \tag{1}$$

The number of sampled ground truth points are the same as the number of coarse anchor features.

UDF loss. Given a batch of query points $\mathcal{Q}$, the UDF is supervised using the following loss function:

$$\mathcal{L}_{\text{UDF}} = \frac{1}{|\mathcal{Q}|} \sum_{\mathbf{q} \in \mathcal{Q}} \|\min(f(\mathbf{q}), \delta) - \min(\text{UDF}(\mathbf{q}), \delta)\|_1 , \tag{2}$$

where $f(\mathbf{q})$ is the UDF prediction and $\text{UDF}(\mathbf{q})$ the distance from a query point to the nearest ground truth point. Following [5], a distance clamping of δ is applied in calculating the loss. We empirically set $\delta = 0.5$.

RGB loss. Following MCC [6], the color is modelled as 256-way classification for each color channel and supervised using a cross-entropy loss.

Total loss. The total loss for one batch of sample is therefore given as:

$$\mathcal{L} = \mathcal{L}_{\text{UDF}} + 0.01 \times \mathcal{L}_{\text{RGB}} + 0.03 \times \mathcal{L}_{\text{anch}}. \tag{3}$$

1.3 Color Metrics

To evaluate colors, we calculate the L_1-norm of RGB value differences between prediction points and the nearest ground truth within 0.1 radius, and vice versa. The 0.1 radius is applied so that the color metrics is not influenced by geometry as much as possible.

Here, we define $\mathcal{P}$ as 3D locations of points, and $\mathcal{C}$ as their corresponding RGB values. To calculate the first component of the L_1-RGB metrics, we first define the set of predicted points that are within 0.1 radius to the nearest ground truths:

$$\mathcal{P}_{\mathrm{A}} = \{\mathbf{p} \in \mathcal{P}_{\mathrm{pred}} | \min_{\mathbf{g} \in \mathcal{P}_{\mathrm{GT}}} \|\mathbf{p} - \mathbf{g}\|_2 < 0.1\}. \tag{4}$$

The L_1-RGB metrics' first component is then obtained from the mean L_1-norms of the RGB differences between points in $\mathcal{P}_{\mathrm{A}}$ and the nearest ground truths:

$$L_1\text{-}RGB_{\mathrm{A}} = \frac{1}{|\mathcal{P}_{\mathrm{A}}|} \sum_i \left\| \mathcal{C}^i_{\mathrm{A}} - \mathcal{C}^k_{\mathrm{GT}} \right\|_1, \tag{5}$$

where $k = \mathrm{argmin}_j \left\| \mathcal{P}^i_{\mathrm{A}} - \mathcal{P}^j_{\mathrm{GT}} \right\|_2$. Likewise, we define the set of ground truth points within 0.1 radius to the nearest predicted points:

$$\mathcal{P}_{\mathrm{B}} = \{\mathbf{g} \in \mathcal{P}_{\mathrm{GT}} | \min_{\mathbf{p} \in \mathcal{P}_{\mathrm{pred}}} \|\mathbf{g} - \mathbf{p}\|_2 < 0.1\}, \tag{6}$$

and the corresponding L_1-RGB metrics from this set to the nearest predictions:

$$L_1\text{-}RGB_{\mathrm{B}} = \frac{1}{|\mathcal{P}_{\mathrm{B}}|} \sum_i \left\| \mathcal{C}^i_{\mathrm{B}} - \mathcal{C}^k_{\mathrm{pred}} \right\|_1, \tag{7}$$

where $k = \mathrm{argmin}_j \left\| \mathcal{P}^i_{\mathrm{B}} - \mathcal{P}^j_{\mathrm{pred}} \right\|_2$. Finally, the L_1-RGB metrics is given as:

$$L_1\text{-}RGB = \frac{1}{2}(L_1\text{-}RGB_{\mathrm{A}} + L_1\text{-}RGB_{\mathrm{B}}). \tag{8}$$

2 Model Flexibility

2.1 High-Resolution Reconstruction

We demonstrate NU-MCC's flexibility in adopting higher resolution fine features to generate reconstruction with enhanced details on the seen part. As shown in Figure 1, we use a resolution of 224×224 for the fine feature at test time while the model was trained using 112×112 resolution. The intricate details on the vase can be better reconstructed without retraining.

Figure 1: **High-resolution reconstruction.** Finer details can be reconstructed by adopting higher resolution fine features at test time. The variable s indicates resolution.

2.2 Smoothing

The feature aggregation process in our Neighborhood decoder also allows versatility for the number of features to be aggregated at test time. Increasing the number of features results in a similar effect as smoothing. As shown in Figure 2, while we set $k = 4$ to aggregate the nearest 4 coarse anchors and 4 fine features during training, we can increase the number of features to be aggregated to handle noisy inputs.

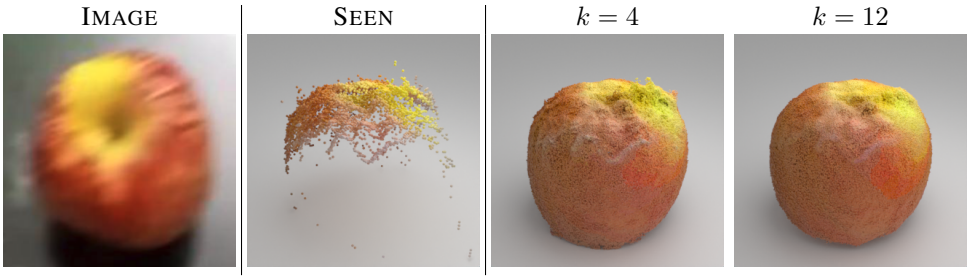

Figure 2: **Smoothing effect.** Increasing the number of aggregated features (e.g., 12 coarse and fine features) at test time gives smoothing effect.

3 Scene Reconstruction Experiments

In this section, we discuss the scene reconstruction experiments with the same setup as established in MCC [6]. Given one RGB-D frame of a scene, the task aims to reconstruct the scene including the parts outside camera frustum.

3.1 Hypersim Dataset

The scene experiment uses Hypersim [7], a photorealistic synthetic scene dataset for training. We train NU-MCC with 550 coarse anchor features for 50 epochs following the data split and training hyperparameters in MCC [6]. Different from MCC which uses the scene 3D meshes for evaluation, we use the Hypersim ground truth data that is sparse in quality since the meshes are not freely available. The quantitative results on the validation set are presented in Table 1. Our model outperforms MCC in all metrics, where the F1 score improves by 59%, and the L_1-RGB error reduces by 25%.

The qualitative comparison is shown in Figure 3. While it is inherently difficult to appropriately hallucinate the parts outside camera frustum, NU-MCC notably reconstructs the seen part with higher details and propagates colors more accurately (e.g., the floor checkerboard pattern) compared to MCC. Our repulsive UDF representation also results in more accurate geometry, including the floor, wall, and ceiling thicknesses.

Table 1: **Quantitative results on Hypersim [7] validation set.** Results are from ten different views.

ARCHITECTURE	L_1-CD↓	F1↑	L_1-RGB↓
MCC [6]	1.118 ± 0.227	31.25 ± 1.1	0.807 ± 0.020
Ours	$\mathbf{1.024 \pm 0.199}$	$\mathbf{49.74 \pm 1.7}$	$\mathbf{0.605 \pm 0.020}$

3.2 Zero-Shot Generalization to Taskonomy Dataset

Finally, we show the generalization of our model trained on Hypersim to Taskonomy [8], a dataset of real scene in zero-shot setting. Despite the distribution shift, our model can reasonably reconstruct the parts outside camera frustum and accurately reconstruct the seen part, as shown in Figure 4.

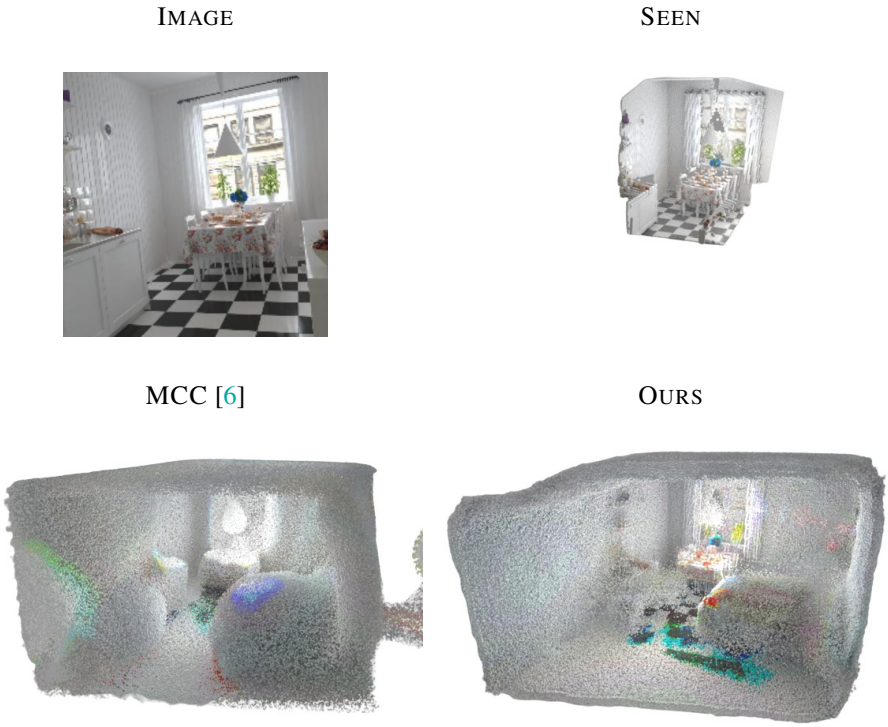

Figure 3: **Qualitative comparison on Hypersim [7] validation set.** Our method achieves better reconstruction compared to the state-of-the-art [6].

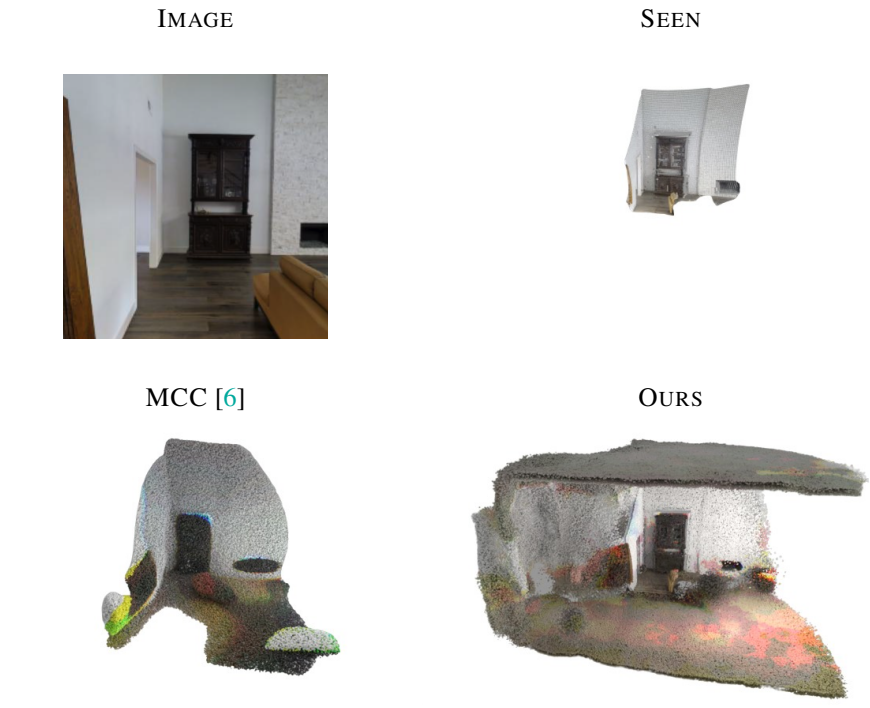

Figure 4: **Scene reconstruction on Taskonomy [8] dataset.** Our method is able to generalize in a challenging zero-shot synthetic-to-real setting.

References

[1] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," *Advances in neural information processing systems*, vol. 30, 2017.

[2] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly *et al.*, "An image is worth 16x16 words: Transformers for image recognition at scale," *arXiv preprint arXiv:2010.11929*, 2020.

[3] S. Peng, M. Niemeyer, L. Mescheder, M. Pollefeys, and A. Geiger, "Convolutional occupancy networks," in *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part III 16*. Springer, 2020, pp. 523–540.

[4] B. Mildenhall, P. P. Srinivasan, M. Tancik, J. T. Barron, R. Ramamoorthi, and R. Ng, "Nerf: Representing scenes as neural radiance fields for view synthesis," *Communications of the ACM*, vol. 65, no. 1, pp. 99–106, 2021.

[5] J. Chibane, G. Pons-Moll *et al.*, "Neural unsigned distance fields for implicit function learning," *Advances in Neural Information Processing Systems*, vol. 33, pp. 21 638–21 652, 2020.

[6] C.-Y. Wu, J. Johnson, J. Malik, C. Feichtenhofer, and G. Gkioxari, "Multiview compressive coding for 3d reconstruction," *arXiv preprint arXiv:2301.08247*, 2023.

[7] M. Roberts, J. Ramapuram, A. Ranjan, A. Kumar, M. A. Bautista, N. Paczan, R. Webb, and J. M. Susskind, "Hypersim: A photorealistic synthetic dataset for holistic indoor scene understanding," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2021, pp. 10 912–10 922.

[8] A. R. Zamir, A. Sax, W. Shen, L. J. Guibas, J. Malik, and S. Savarese, "Taskonomy: Disentangling task transfer learning," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 3712–3722.