Supplementary Material:
Multiview Compressive Coding for 3D Reconstruction

Chao-Yuan Wu Justin Johnson Jitendra Malik Christoph Feichtenhofer Georgia Gkioxari
FAIR, Meta AI

1. Animations and Interactive Visualizations

We provide 360-view animations and interactive 3D visualizations for all qualitative results, in Figures 4, 7 and 9, and more in our project page. Our video animations are shown in the main window and interactive 3D visualizations are available by clicking on the *3D icon*, per the instructions in the webpage.

2. Architecture Specifications

Table 1 describes in detail the MCC architecture for the E^{RGB} and E^{XYZ} encoders and the decoder.

The E^{RGB} and E^{XYZ} encoders follow the "ViT-Base" transformer architecture by Dosovitskiy *et al*. [4, 10]. The transformer architecture is composed of 12 layers of a 768-dimensional self-attention operator with 12 heads, referred to as multi-head attention (MHA), followed by a 3072-dimensional 2-layer MLP. The input image size is 224×224. The RGB image I, input to the E^{RGB} encoder, is embedded via a single convolutional layer, of a 16×16-sized kernel and a 16×16 stride, to produce $N^{enc} = 196$ tokens. The (seen) points P, input to the E^{XYZ} encoder, are first linearly projected to a 768-dimensional representation and then embedded via a single transformer layer which operates on 16×16 non-overlapping patches as explained in Section 3.4 of the main paper and further described in Table 1, resulting also in $N^{enc} = 196$ tokens. The single transformer layer used for the patch embeddings defines a `[cls]` token whose output is the embedding for each patch, as is commonly used in [3, 4] and referred to as a readout token.

Our decoder follows the decoder design from MAE [6]. It is composed of 8 layers of a 512-dimensional self-attention operator with 16 heads followed by a 2048-dimensional 2-layer MLP. The input to the decoder is: (a) $N^q = 550$ 3D point queries which are linearly projected to a 768-dimensional vector, and (b) input R which concatenates the N^{enc} output tokens from E^{RGB} and E^{XYZ} in the channel dimension and then linearly projects each to a 768-dimensional vector. This results in a $768 \times (N^q + N^{enc}) = 768 \times 746$ input to the decoder. Our decoder additional defines a global `[cls]` token whose role is to "summarize" all inputs in the transformer and can be attended freely by other tokens.

LayerNorm [1] is used in all self-attention and MLP layers following standard practice [4, 6, 10].

3. Held-Out CO3D Categories

In our experiments, we hold out 10 randomly selected categories which we use as our test set. The 10 randomly selected held-out categories are: {*apple, ball, baseball-glove, book, bowl, carrot, cup, handbag, suitcase, toy-plane*}. They have 8,453 example videos in total. Please see the original CO3D paper for more information about CO3D [8].

4. Additional Implementation Details for Scene Reconstruction Experiments

Similar to the object reconstruction experiments, we train MCC on Hypersim [9] with Adam [7] for 100k iterations with an effective batch size of 512 using 32 GPUs, a base learning rate of 5×10^{-5} with a cosine schedule and a linear warm-up for the first 10% of iterations. Training takes ~1.6 days. We normalize each scene to have zero-mean and unit-variance. At inference time, we predict points up to 6.0 units (*i.e.*, 6× standard deviation) away from the camera origin. Since we aim at predicting the scene in front of the camera, we use the camera view coordinate system (X-axis points top/down, Y-axis points left/right, and Z-axis points out from the image plane). We randomly scale augment images by $s \in [0.8, 1.2]$, as in the object reconstruction model, but do not perform rotation augmentation. Other implementation details follow the CO3D experiments.

Stage	Operators	Output sizes
Input I	-	3×224×224
Patch embed	Conv 16×16, 768 (stride 16×16)	768×196
Transformer layers	$\begin{bmatrix} \text{MHA(768)} \\ \text{MLP(3072)} \end{bmatrix} \times 12$	768×196

(a) **Encoder** E^{RGB}

Stage	Operators	Output sizes
Input P	-	3×224×224
Embed	Linear, 768	768×224×224
Patch embed	$\begin{bmatrix} \text{MHA(768)} \\ \text{MLP(1536)} \\ \texttt{[cls]}\text{ readout} \end{bmatrix} \times 1$ (on each 16×16 patch)	768×196
Transformer layers	$\begin{bmatrix} \text{MHA(768)} \\ \text{MLP(3072)} \end{bmatrix} \times 12$	768×196

(b) **Encoder** E^{XYZ}

Stage	Operators	Output sizes
Input encodings	-	768×196 768×196
Concat	Concat	1536×196
Linear	Linear, 768	768×196

(c) **Fusion Module** f

Stage	Operators	Output sizes
Input queries	-	3×550
Embed	Linear, 768	768×550
Concat with R	Concat	768×746
Transformer layers	$\begin{bmatrix} \text{MHA(512)} \\ \text{MLP(2048)} \end{bmatrix} \times 8$	768×746

(d) **Decoder** Dec

Table 1. **Architecture specification** for each part of the MCC model. MHA(c): Multi-Head Attention [10] with c channels. MLP(c'): MultiLayer Perceptron with c' channels. `[cls]` readout: feature readout with the `[cls]` token [3,4]. Here, we use the default choice of $N^q = 550$ queries. We omit the optional `[cls]` token in the outputs of the transformers for clarity.

5. Additional Experiments

Comparison to Prior Works on Generalization. Fig. 2 compares MCC with PoinTr [11], trained on CO3D, and Mesh R-CNN [5], trained on ShapeNet [2] on a challenging DALL·E 2 image. Both baselines struggle possibly due to the large domain gap with their respective training sets, while Mesh R-CNN seems to do a bit better than PoinTr. MCC, trained on the same dataset as PoinTr, performs much better than both.

Qualitative Results of 'Detailed' vs. 'Global Pooling'. Table 1(d) in the main paper shows that the default 'detailed' feature conditioning design outperforms 'global' by

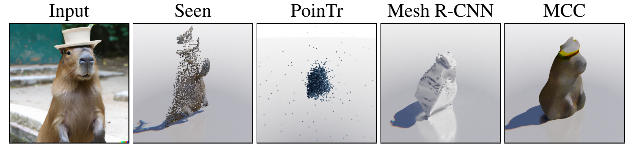

Table 2. **Comparison to Prior Works on Generalization.** MCC performs much better than PionTr [11] and Mesh R-CNN [5] on the challenging DALL·E 2 image.

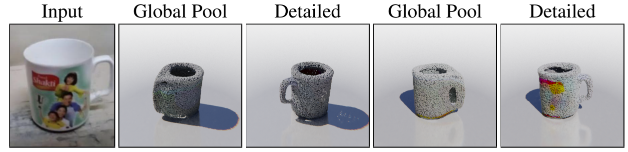

Figure 1. **'Detailed' vs. 'Global Pooling' for Feature Conditioning.** The default 'detailed' design shows better geometry and texture details.

N^q	Acc	Cmp	F1
250	46.6	75.7	55.3
550 (default)	47.5	76.0	**56.7**
1000	47.2	76.2	56.3

Figure 2. **Number of training queries N^q.** Increasing N^q beyond the default choice of 550 does not perform better.

2.2% in F1. Fig. 1 presents a qualitative example. We can see that the 'detailed' design shows better geometry and texture details.

Impact of the Number of Training Queries N^q. Fig. 2 presents the ablation results. We observe that overall MCC is not very sensitive to the choice of N^q. Also, further increasing N^q beyond the default choice of 550 does not perform better.

References

[1] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016. 1

[2] Angel X Chang, Thomas Funkhouser, Leonidas Guibas, Pat Hanrahan, Qixing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, et al. ShapeNet: An information-rich 3D model repository. *arXiv preprint arXiv:1512.03012*, 2015. 2

[3] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In *NAACL*, 2019. 1, 2

[4] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Trans-

formers for image recognition at scale. In *ICLR*, 2021. 1, 2

[5] Georgia Gkioxari, Jitendra Malik, and Justin Johnson. Mesh R-CNN. In *ICCV*, 2019. 2

[6] Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked autoencoders are scalable vision learners. In *CVPR*, 2022. 1

[7] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *ICLR*, 2015. 1

[8] Jeremy Reizenstein, Roman Shapovalov, Philipp Henzler, Luca Sbordone, Patrick Labatut, and David Novotny. Common objects in 3D: Large-scale learning and evaluation of real-life 3D category reconstruction. In *ICCV*, 2021. 1

[9] Mike Roberts, Jason Ramapuram, Anurag Ranjan, Atulit Kumar, Miguel Angel Bautista, Nathan Paczan, Russ Webb, and Joshua M Susskind. Hypersim: A photorealistic synthetic dataset for holistic indoor scene understanding. In *ICCV*, 2021. 1

[10] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *NeurIPS*, 2017. 1, 2

[11] Xumin Yu, Yongming Rao, Ziyi Wang, Zuyan Liu, Jiwen Lu, and Jie Zhou. PoinTr: Diverse point cloud completion with geometry-aware transformers. In *ICCV*, 2021. 2