# POEM: Reconstructing Hand in a Point Embedded Multi-view Stereo
## **Supplementary Material**

Lixin Yang[1,2] Jian Xu[3] Licheng Zhong[1] Xinyu Zhan[1] Zhicheng Wang[3] Kejian Wu[3] Cewu Lu[1,2†]
[1]Shanghai Jiao Tong University [2]Shanghai Qi Zhi Institute [3]Nreal
{siriusyang, zlicheng, kelvin34501, lucewu}@sjtu.edu.cn
{jianxu, kejian}@nreal.ai chgggo@gmail.com

## 1. Implementation Details

All the experiments are developed using PyTorch library and are conducted on a machine with 4 NVIDIA A10 GPUs (24GB RAM). These experiments use the batch size of 16 and total 100 epochs of training. The learning rate is set to $1 \times 10^{-4}$ and decayed by a factor of 0.1 at the 70 epochs. All the experiments related to the multi-view settings use a same CNN backbone: ResNet34 [1]. The vision Transformer is initialized with *xavier* uniform distribution and the CNN backbone is initialized with ImageNet [2] pre-trained weights.

Regarding to the experiments in POEM, the radius of ball query is set to be $0.2m$ around the center of hand and total $S = 2048$ points (number of points in $\bar{\mathbf{P}}$) are sampled within this range. The number $k$ of nearest neighbors is set to be 16 in the cross-set point Transformer.

We apply standard image augmentation techniques to train our model, including random center offset, scaling, and color jittering. Additionally, we apply random rotation. However, rotational augmentation in the multi-view setting differs from that in the single-view setting. In single-view training, a rotation on the image corresponds to the same rotation on the 3D hand model. However, in multi-view training, a rotation on the image is represented as left-multiplication of the rotation on the camera extrinsic matrix.

## 2. More evaluations

**Number of Decoder Layers.** We examine the performance of POEM on varying the numbers of decoder layers in its point Transformer. The results on HO3D-MV are shown in Tab. 1 rows 1–4, where the "d1" indicate only use one decoder layer. We find that using 6 decoder layers achieves the best performance.

**Numver of Camera Views.** We evaluate the performance of POEM on varying the numbers of cameras in the multi-view setting. The results on HO3D-MV are shown in Tab. 1 rows 5–8, where the "c2" indicates only use two cameras. The results show POEM can effectively fuse the features from different camera frustums and thus boost the performance when the number of cameras increases.

**Number of *k* Nearest Neighbors.** We evaluate the performance of POEM on varying the number of $k$ from 4, 8, 16, to 32. A value of $k = 32$ with a batch size of 16 almost exhausts the memory of the A10 GPU. Tab. 1 rows 9–12 show that larger $k$ could lead to better results. But the computation cost also increases with it. We use $k$=16 for the trade-off between the cost and performance.

| | Exp | Joints | | | Vertices | | |
|---|---|---|---|---|---|---|---|
| | | MPJPE | RR-J | PA-J | MPVPE | RR-V | PA-V |
| HO3D | d1 | 19.29 | 24.49 | 11.19 | 19.17 | 23.86 | 11.99 |
| | d2 | 19.00 | 24.34 | 11.22 | 18.72 | 23.65 | 11.81 |
| | d4 | 18.81 | 24.33 | 10.79 | 18.52 | 23.58 | 11.27 |
| | d6 | **17.55** | **22.59** | **9.83** | **17.56** | **23.07** | **9.40** |
| | c2 | 28.82 | 38.53 | 20.85 | 28.86 | 38.27 | 22.43 |
| | c3 | 26.29 | 40.55 | 17.78 | 27.72 | 39.78 | 21.65 |
| | c4 | 20.82 | 26.77 | 12.18 | 20.55 | 25.84 | 12.69 |
| | c5 | **19.22** | **24.97** | **11.17** | **18.93** | **24.22** | **11.59** |
| OakInk | k4 | 6.34 | 8.08 | 4.40 | 8.33 | 9.75 | 6.87 |
| | k8 | 6.39 | 8.07 | 4.42 | 8.16 | 9.57 | 6.64 |
| | k16 | **6.34** | **8.02** | **4.37** | 8.08 | 9.53 | 6.57 |
| | k32 | 6.36 | 8.02 | 4.39 | **7.93** | **9.39** | **6.36** |

Table 1. Performance of POEM on varying the number of decoder layers, cameras, and $k$ nearest neighbors in the multi-view setting. The best results are highlighted in **bold**.

**Inference Time.** Tab. 2 compares the inference time and model parameters of four methods: POEM, MVP, PE-Mesh-TR, and the multi-view mesh fitting. The inference time is calculated as an average feed-forward time of one multi-view sample on one GPU.

| Model | POEM | MVP | PE-Mesh-TR | Fit. |
|---|---|---|---|---|
| time (s) | 0.067 | 0.055 | 0.035 | 9.89 |
| params(M) | 117 | 144 | 124 | - |

Table 2. Inference time and model parameters.

## 3. Qualitative results

We demonstrate more qualitative results of POEM on the three datasets in Figs. 1 to 3. From top to bottom we plot the results on DexYCB-MV, HO3D-MV and OakInk-MV dataset. For each multi-view frame, we draw its result from 5 different views. One of the views is in normal size and the other four views are half size.

Figure 1. Qualitative results on DexYCB-MV testing set.

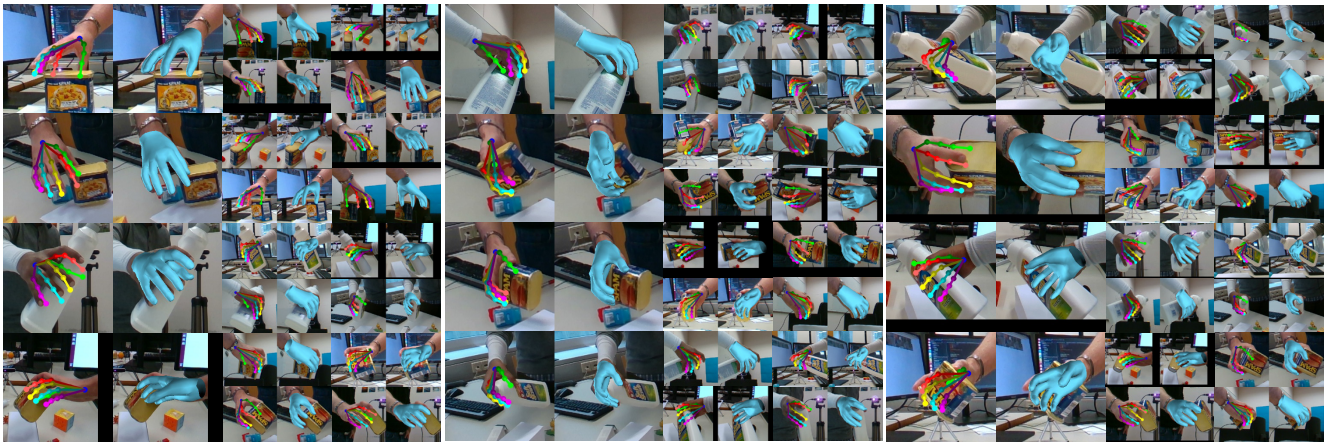

Figure 2. Qualitative results on HO3D-MV testing set.

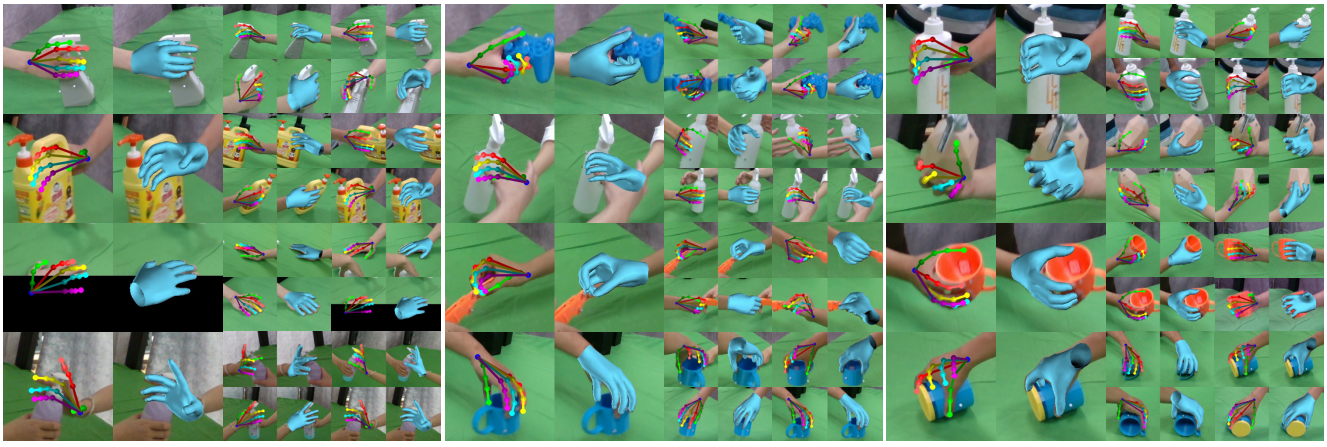

Figure 3. Qualitative results on OakInk-MV testing set.

## References


[1] Kaiming He, X. Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Computer Vision and Pattern Recognition (CVPR)*, 2016. 1

[2] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael S. Bernstein, Alexander C. Berg, and Li Fei-Fei. ImageNet large scale visual recognition challenge. *International Journal of Computer Vision*, 115:211–252, 2015. 1