OnePose++: Keypoint-Free One-Shot Object Pose Estimation without CAD Models

Anonymous Author(s)
Affiliation
Address
email

1 Method Details

1.1 Keypoint-Free SfM

Reference nodes selection strategy. The proposed keypoint-free SfM establishes 3D structures in a coarse-to-fine manner. It first reconstructs a complete 3D model leveraging the semi-dense matches of the coarse-level LoFTR, then refines the initial 3D model to higher accuracy. We refine the initial point cloud by refining "keypoints" in the coarse feature tracks to sub-pixel accuracy and optimize the point cloud based on the refined feature tracks. The fine-level LoFTR is used for coarse feature track refinement, which refines all points in a feature track with Transformers with reference to a fixed reference point.

We find that the fine-level LoFTR is robust and insensitive to the reference point selection strategy. This is reasonable since the fine-level LoFTR is exactly trained to find a sub-pixel correspondence on a local feature patch for an arbitrarily given feature point. Consequently, we design the reference node selection strategy mainly for the ease of implementation and lower memory consumption. To refine all feature tracks with the fine-level LoFTR, we need to extract fine-level CNN feature maps for all images and store them for further use. This would take a ton of storage, varied according to the total number of frames, which can hardly fit into the RAM of consumer-grade GPUs.

To make our reconstruction pipeline broadly usable, we treat an image, instead of a feature track, as the minimum processing unit. More specifically, we recursively select the frame containing the maximum number of "keypoints" (i.e., involved in the most feature tracks), and select all "keypoints" in this frame as the fixed reference nodes of their belonging feature tracks. All feature tracks in the selected frame are then refined, and marked as processed. We repeat this process untill all feature tracks are refined. This strategy avoids the need to store dense feature maps of all frames and minimizes the number of frames whose feature maps are repeatedly extracted.

1.2 Object Pose Estimation

Positional encoding. We apply positional encoding modules on top of the coarse 3D features $\tilde{\mathbf{F}}_{3D} \in \mathbb{R}^{N \times D_c}$ and 2D feature maps $\tilde{\mathbf{F}}_{2D}$ to make them position-dependent, which is proved to boost the matching performance [7, 10]. Because the 2D-3D matching involves two modalities, we use the standard sinusoidal encoding for the 2D feature maps and leverage a learned positional encoding for the 3D features. More specifically, for the 3D features, we embed the normalized 3D coordinates $\mathbf{x}_{3D} \in \mathbb{R}^{N \times 3}$ into a high-dimensional vector with an MLP:

$$\tilde{\mathbf{F}}'_{3D} = \tilde{\mathbf{F}}_{3D} + \mathrm{MLP}_{\mathrm{pe}}(\mathbf{x}_{3D}). \tag{1}$$

For the 2D feature maps, we use a 2D extension of the standard sinusoidal positional encoding proposed in Transformers following DETR [1]:

$$\mathcal{PE}_{x,y}^{i} = f(x,y)^{i} := \begin{cases} \sin\left(\omega_k \cdot x\right), & i = 4k \\ \cos\left(\omega_k \cdot x\right), & i = 4k+1 \\ \sin\left(\omega_k \cdot y\right), & i = 4k+2 \\ \cos\left(\omega_k \cdot y\right), & i = 4k+3, \end{cases} \tag{2}$$

where $\omega_k = \frac{1}{10000^{2k/d}}$, d is the number of feature channels on which positional encoding is applied, i is the index of the feature channel.

The positional encoding modules enable the later attention modules to jointly reason about visual appearances and positions, benefiting 2D-3D matching. Note that the positional encodings are only applied once before the first attention module.

Attention module. Directly using the vanilla Transformer [12] to our model is not applicable because its computation cost grows quadratically with the length of input features. Following [10], we use the Linear Transformer [2] to efficiently transform 2D and 3D features. It reduces the computational complexity of the Transformer [12] from $O(N^2)$ to $O(N)$ by substituting the exponential kernel with an alternative kernel function $\mathrm{sim}(Q,K) = \phi(Q)\cdot\phi(K)^{\mathrm{T}}$, where $\phi(\cdot) = \mathrm{elu}(\cdot)+1$. Please refer to the original paper [2] for more details.

We denote a set of self- and cross-attention layers as an attention block:

$$\begin{cases} \mathbf{F'}_{2D}^{(l+1)} = \mathrm{SelfAtten}(\mathbf{F}_{2D}^{(l)}, \mathbf{F}_{2D}^{(l)}), \\ \mathbf{F'}_{3D}^{(l+1)} = \mathrm{SelfAtten}(\mathbf{F}_{3D}^{(l)}, \mathbf{F}_{3D}^{(l)}), \\ \mathbf{F}_{2D}^{(l+1)}, \mathbf{F}_{3D}^{(l+1)} = \mathrm{CrossAtten}(\mathbf{F'}_{2D}^{(l+1)}, \mathbf{F'}_{3D}^{(l+1)}). \end{cases} \tag{3}$$

The indices of intermediate features are indicated by $\cdot^{(l)}$. $\mathbf{F'}$ represents an intermediate feature processed by a self-attention layer. Our attention module sequentially performs the attention block $N_c = 3$ times to transform the 3D and 2D features.

Supervision. We jointly train the coarse and fine modules in our 2D-3D matching framework with different supervisions. We project the observable 3D points to the 2D frame to build the ground-truth 2D-3D correspondences $\mathcal{M}_{gt}^{f}$ for our fine matching module. For the coarse matching module, we round the projected 2D points to their nearest grid points to obtain the ground-truth coarse 2D-3D correspondences $\mathcal{M}_{gt}^{c}$. We optimize the coarse module by minimizing the focal loss [4] between the predicted matching probability matrix $\mathcal{P}^c$ and the ground truth $\mathcal{P}_{gt}^{c}$ constructed with $\mathcal{M}_{gt}^{c}$ similar to [7, 10]:

$$\mathcal{L}_c = \frac{1}{|\mathcal{P}_{gt}^{c}|}\sum_{j,q}\mathrm{FL}(\mathcal{P}^c(j,q)) \tag{4}$$

$$\mathrm{FL}(\mathcal{P}^c(j,q)) = \begin{cases} -\alpha(1-\mathcal{P}^c(j,q))^{\gamma}\log(\mathcal{P}^c(j,q)), & \text{if } \mathcal{P}_{gt}^{c}(j,q) = 1 \\ -(1-\alpha)\mathcal{P}^c(j,q)^{\gamma}\log(1-\mathcal{P}^c(j,q)), & \text{if } \mathcal{P}_{gt}^{c}(j,q) \neq 1. \end{cases}$$

For the fine module, we use a ℓ_2 loss to minimize the distances between the predicted 2D coordinates $\hat{\mathbf{u}}^q$ and the ground truth $\hat{\mathbf{u}}_{gt}^{q}$. Following [13, 10], we make our loss uncertainty-weighted with a variance term $\sigma^2(q)$:

$$\mathcal{L}_f = \frac{1}{|\mathcal{M}_f|}\sum_{q\in\mathcal{M}_f}\frac{1}{\sigma^2(q)}\left\|\hat{\mathbf{u}}^q - \hat{\mathbf{u}}_{gt}^{q}\right\|_2, \tag{5}$$

Notably, we detach $\sigma^2(q)$ during training to prevent the network from decreasing the loss by increasing the variance. The total loss is the weighted sum of the coarse and fine losses $\mathcal{L} = \omega_c\mathcal{L}_c + \omega_f\mathcal{L}_f$. In the experiment, α is 0.5, γ is 2.0, ω_c is 1.0 and ω_f is 1.0.

2 OnePose-HARD Dataset

In this section, we provide more details of the proposed OnePose-HARD evaluation dataset. This dataset contains 80 sequences of 40 household low-textured objects. For each object, two video sequences with object poses and annotated object 3D bounding boxes are provided. The video

Figure 1: **CAD models from the proposed OnePose-HARD dataset.** We capture CAD models for a subset of ten objects from the OnePose-HARD dataset. These CAD models can be used to train instance-level methods such as PVNet [6] and CDPN [3] and enable further comparisons between CAD-model-free methods and CAD-model-based methods.

sequences of each object are captured with different backgrounds, simulating the real-world using scenario. Each video is recorded at 30 fps for about 30 seconds in 1920×1440 resolution.

The data capture and annotation pipeline follow the setup of OnePose [11]. The camera poses provided by ARKit can be transformed into the object-centric coordinate system induced from the user-annotated object 3D bounding boxes. Following [11], we align multiple captured sequences of an object with the annotated object 3D bounding boxes. Then, we perform a bundle adjustment with COLMAP to reduce the pose drift of ARKit and inconsistency between 3D bounding box annotations in multiple sequences. This offline optimization process leads to more consistent 3D bounding box annotations across sequences and more accurate object poses.

To compare with instance-level methods such as PVNet [6] and CDPN [3], we additionally capture high-quality 3D CAD models for a selected subset of ten objects from the OnePose-HARD dataset. We use the SHINING$^{(R)}$ scanner for the CAD model capturing. Fig. 1 illustrates all captured CAD models.

3 Experiment Details

3.1 Training Details

Our model is trained on the OnePose [11] training set, which contains 49 objects. We first reconstruct the semi-dense object point cloud with our keypoint-free SfM for each object using all training sequences with 5x downsampled video frames. Then we leverage the 2D-3D correspondences computed from the annotated poses and the reconstructed 3D model to train our sparse-to-dense 2D-3D matching module. Note that we compute the rough 2D object bounding boxes from the annotated 3D bounding boxes in the dataset and use the cropped images for reconstruction and training, following OnePose [11].

3.2 Metrics

We use the commonly used *cm-degree* pose error, the *ADD(s)* and the *Proj2D* metrics to evaluate the estimated object poses. We follow PixSfM [5] to evaluate the reconstructed object point cloud.

cm-degree metric. For a predicted pose, the rotation error and translation error are computed separately. A predicted pose is considered correct if both its rotation error and translation error are less than a threshold.

Table 1: **More ablation results.**

	OnePose dataset			OnePose-HARD			Time
	1cm-1deg	3cm-3deg	5cm-5deg	1cm-1deg	3cm-3deg	5cm-5deg	
Full($N_c = 3, N_f = 1$, use all 3D points)	**50.7**	**80.0**	**87.0**	**16.3**	**55.4**	**70.3**	88ms
w/o Position Encoding	49.6	79.4	86.4	15.6	53.4	68.7	87ms
Large model($N_c = 6, N_f = 2$)	50.6	79.9	87.0	**16.3**	53.8	68.4	133ms
Sample 7000 3D points	49.1	79.3	86.3	15.5	52.8	68.1	57ms
Sample 3000 3D points	47.7	78.4	85.8	14.7	50.7	65.6	42ms

Proj2D metric. The *Proj2D* metric computes the mean distance between the projection of 3D model points with given predicted and ground truth object poses. The estimated pose is considered correct if the mean projection distance is less than 5 pixels.

ADD metric. We first transform the 3D model points with the ground truth and the predicted poses. Then we compute the *ADD* metric using the mean distance between two transformed point sets. The pose is regarded as correct if the mean distance is less than 10% of the object diameter. Note that for symmetric objects, we use the *ADD(S)* [14] metric for evaluation.

Point cloud accuracy. We evaluate the point cloud accuracy in the ablation studies, following the metric in [5, 9]. The accuracy is defined as the percentage of reconstructed points which are within a distance threshold(e.g., *3mm*) with reference to the ground truth point cloud. We use vertices of the scanned object meshes as the ground truth point clouds.

3.3 Runtime Analyses of Keypoint-Free SfM

We evaluate the runtime of each part in the proposed keypoint-free SfM. The experiment is conducted on a server with two Intel$^{(R)}$ Xeon Gold 6146 CPU and an NVIDIA-V100-32GB GPU. We illustrate the runtime analyses with only one object instance below. The overall runtime varies according to several factors, such as image resolutions, the number of images used for reconstruction, and the number of successfully built coarse matches. For a video sequence with 193 images in 512×512 resolution, it takes 135s to perform sequential coarse matching on 1436 image pairs, and 40s to load all coarse matches and perform the triangulation [8]. Then, we perform fine matching between 1122 image pairs from the scene graph of coarse reconstructions to refine the feature tracks, which takes 171s. Finally, we optimize the object point cloud, which only consumes 1.03s.

3.4 More Ablation Results

We further conduct additional ablation studies on variants of the 2D-3D matching network architectures and different numbers of 3D points used for pose estimation.

Ablation on 2D-3D matching network architectures. The results of a large model with more attention layers and a model without positional encoding are shown in Tab 3.3. Increasing the number of attention layers by twice barely changes the results.

Different numbers of 3D points. We evaluate the effect of using different numbers of 3D points for object pose estimation. The results are illustrated in Tab 3.3. Our full model uses all reconstructed 3D object points for pose estimation, obtaining the highest accuracy. Decreasing the number of points with subsampling leads to minorly degraded pose estimation accuracy and faster inference speed.

References

[1] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. *ArXiv*, abs/2005.12872, 2020. 2

[2] Angelos Katharopoulos, Apoorv Vyas, Nikolaos Pappas, and Franccois Fleuret. Transformers are rnns: Fast autoregressive transformers with linear attention. *ArXiv*, abs/2006.16236, 2020. 2

[3] Zhigang Li, Gu Wang, and Xiangyang Ji. Cdpn: Coordinates-based disentangled pose network for real-time rgb-based 6-dof object pose estimation. *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 7677–7686, 2019. 3

[4] Tsung-Yi Lin, Priya Goyal, Ross B. Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *ICCV*, 2017. 2

[5] Philipp Lindenberger, Paul-Edouard Sarlin, Viktor Larsson, and Marc Pollefeys. Pixel-perfect structure-from-motion with featuremetric refinement. *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 5967–5977, 2021. 3, 4

[6] Sida Peng, Xiaowei Zhou, Yuan Liu, Haotong Lin, Qixing Huang, and Hujun Bao. PVNet: pixel-wise voting network for 6dof object pose estimation. *T-PAMI*, 2020. 3

[7] Paul-Edouard Sarlin, Daniel DeTone, Tomasz Malisiewicz, and Andrew Rabinovich. SuperGlue: Learning feature matching with graph neural networks. In *ICCV*, 2020. 1, 2

[8] Johannes L. Schönberger and Jan-Michael Frahm. Structure-from-motion revisited. In *CVPR*, 2016. 4

[9] Thomas Schöps, Johannes L. Schönberger, S. Galliani, Torsten Sattler, Konrad Schindler, Marc Pollefeys, and Andreas Geiger. A multi-view stereo benchmark with high-resolution images and multi-camera videos. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2538–2547, 2017. 4

[10] Jiaming Sun, Zehong Shen, Yuang Wang, Hujun Bao, and Xiaowei Zhou. LoFTR: Detector-free local feature matching with transformers. *CVPR*, 2021. 1, 2

[11] Jiaming Sun, Zihao Wang, Siyu Zhang, Xingyi He, Hongcheng Zhao, Guofeng Zhang, and Xiaowei Zhou. OnePose: One-shot object pose estimation without CAD models. *CVPR*, 2022. 3

[12] Ashish Vaswani, Noam M. Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. *ArXiv*, abs/1706.03762, 2017. 2

[13] Qianqian Wang, Xiaowei Zhou, Bharath Hariharan, and Noah Snavely. Learning feature descriptors using camera pose supervision. In *ECCV*, 2020. 2

[14] Yu Xiang, Tanner Schmidt, Venkatraman Narayanan, and Dieter Fox. PoseCNN: A convolutional neural network for 6d object pose estimation in cluttered scenes. *RSS*, 2018. 4