CLUSTER-BASED 3D KEYPOINT DETECTION FOR CATEGORY-AGNOSTIC 6D POSE TRACKING

Long Tian, Andrea Cavallaro, Changjae Oh

Centre for Intelligent Sensing, Queen Mary University of London, United Kingdom

ABSTRACT

We present a model for category-agnostic 6D pose tracking. We tackle object pose tracking as a 3D keypoint detection and matching task that does not require ground-truth annotation of the keypoints. Using RGB-D data and the target object mask as inputs, we spatially segment the point cloud of the object into clusters. Each 3D point in the cluster is characterised by features encoding appearance and geometric information. We use these features to detect a keypoint for each cluster and, with the detected keypoint sets from two time instants, we recover the pose change through least-squares optimisation. The loss functions are designed to ensure that the detected keypoints are consistent over time and suitable for pose tracking.

***Index Terms*—** 6D pose tracking, keypoint detection, category-agnostic

1. INTRODUCTION

6D pose tracking, the task of estimating the pose of an object over time using an initial 6D pose [1, 2], is important for several computer vision tasks, such as 3D scene understanding, robotic manipulation and augmented reality. Traditional 6D pose tracking methods [3–5] use Bayesian networks [6] to estimate a posterior distribution over the current object pose. These methods require the design of handcrafted features. Recently, machine learning techniques have been employed for 6D pose tracking. Assuming that 3D object models are available, SE(3)-TrackNet [7] uses a deep neural network to detect the relative pose at the current frame and the frame rendered by the previous pose detection. PoseRBPF [8] formulates 6D pose tracking in a Rao-Blackwellized particle filtering framework [9]. PoseRBPF [8] first uses the 3D translation to determine the target object bounding box in the image, and then uses an autoencoder to estimate object feature embeddings to update the 3D rotation distribution. These models only achieve instance-level 6D pose tracking, and thus require accurate 3D shapes of the tracking instances, such as those given by CAD models.

6D pose tracking can be performed for each object category to track unseen (object) instances within a specific category without the need of 3D shape information about the target object (e.g. 6-PACK [1]). 6-PACK selects a detected anchor closest to the object centroid and, from the selected anchor, generates ordered 3D keypoints for tracking. Such an approach requires multiple trained models on single object categories to handle different object categories. BundleTrack [2], a category-agnostic model, uses pose graph optimisation for 6D pose tracking irrespective of object categories. BundleTrack uses an off-the-shelf keypoint detector, LF-Net [10], with fixed pre-trained parameters.

In this paper, we propose a cluster-based keypoint detection module to be used in a generic, category-agnostic 6D pose tracking pipeline. Given an RGB-D input and the corresponding object mask, we convert the depth image into a 3D point cloud of the target object. We then use the farthest point sampling (FPS) algorithm [11] to sample initial points from the point cloud. We group points closest to each initial point into a cluster and get the feature map of each cluster. We then detect a keypoint from the feature map of each cluster and use the resulting keypoint set to represent the target object. Using keypoint sets from two time instants, we estimate the pose change by least-squares optimisation. Experimental results on the NOCS-REAL275 dataset [12] show that our category-agnostic model with detection of matched 3D keypoint sets is competitive to category-specific state-of-the-art models for 6D pose tracking[1].

2. PROPOSED MODEL

The 6D pose change, Δp^t, of the target object at time t with respect to time $t-1$ consists of a 3D rotation, $\Delta R^t \in SO(3)$, and a 3D translation, $\Delta T^t \in \mathbb{R}^3$:

$$\Delta p^t = [\Delta R^t | \Delta T^t]. \quad (1)$$

To estimate Δp^t, we use the 3D keypoints at t corresponding to those at $t-1$ determined after solving a point set alignment problem with least-squares optimisation [1, 16]. After obtaining features for each 3D point using RGB, depth and the target object segmentation mask, the 3D points are grouped into clusters and each cluster is represented by a 3D keypoint (see Fig. 1).

[1]Project page: https://eecs.qmul.ac.uk/~coh/projects/6D_CAPT.html

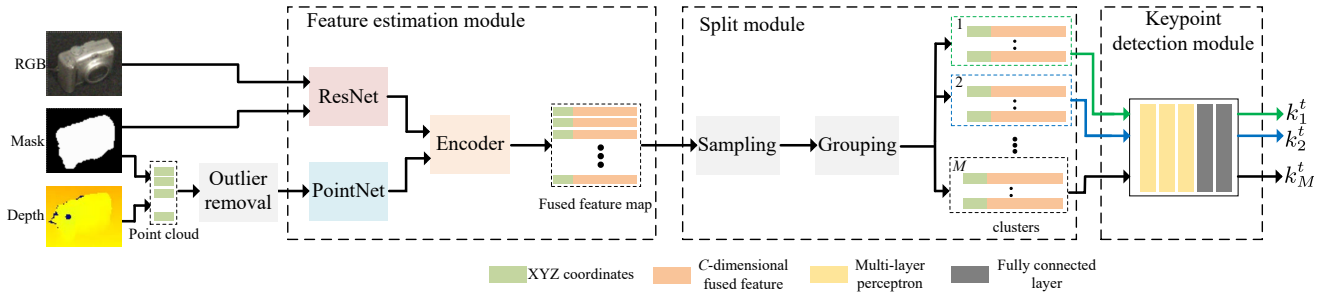

Fig. 1. The overall pipeline of 3D keypoint detection, consisting of three modules: a feature estimation module, a split module, and a 3D keypoint detection module. Given RGB and depth images, and a segmentation mask of the target object, the network generates a 3D keypoint set, $K^t = \{k_1^t, k_2^t, ..., k_M^t\}$. We use ResNet [13] to extract the appearance feature of the object using the RGB image and the segmentation mask. We also extract the geometric feature using the point cloud, after outlier removal, as input to PointNet [14]. The appearance and geometric features are fused by an encoder, following the architecture from DenseFusion [15]. The split module uses the farthest point sampling algorithm to sample initial points, finds N_c nearest 3D points for each initial point, and groups the initial point and its neighbours into a cluster. For each of the M clusters a keypoint is defined by the keypoint detection module.

2.1. 3D Point clustering

We use a deep neural network to fuse colour and geometric information into a feature map for pose estimation [15]. We first crop the RGB image by using the target object 2D bounding box, and use the cropped image as input to ResNet [13] to obtain the appearance feature of each pixel. We then use the segmentation mask and the depth image to obtain the 3D points of the target object. Since the obtained point cloud may contain outliers due to sensor noise, we use a radius outlier removal algorithm from Open3D [17] to remove points that have few neighbours in a certain sphere around them. We use this point cloud as input for PointNet [14] to obtain the geometric feature from each point. Note that the 2D bounding box and the segmentation mask, generated using Mask R-CNN [18], are given in the NOCS-REAL275 dataset [12]. Using the model in DenseFusion [15], we fuse the appearance and geometric features to obtain a pixel-wise fused feature map $v_i = [x_i,\ f_i]$, $i = 1, 2, ..., N$, where N denotes the number of points, $x_i \in \mathbb{R}^3$ represents the point with XYZ coordinates, and $f_i \in \mathbb{R}^C$ represents C-dimensional feature vector, which encodes appearance and geometric information.

Directly detecting a set of 3D keypoints for pose tracking is challenging due to the large output space to explore [1]. We are inspired by PointNet++ [19] that splits the point cloud into different clusters, and then capture each cluster features for point cloud classification and semantic scene labelling. We evenly split the target object into different clusters. Each cluster includes the same number of points and the cluster feature map contains appearance and geometric information. We then detect a 3D keypoint from each cluster instead of directly detecting keypoints from the whole point cloud. Such design not only helps us improve the efficiency of keypoint detection, but also makes us easily to find matched keypoints since we can easily find corresponding clusters over times. Another benefit is that our split module does not require knowledge of a specific category, allowing us to achieve category-agnostic keypoint detection. We sample M initial points and then group the nearest points for each initial point as a cluster. As a result, we split the 3D point cloud of the target object into M clusters. Specifically, we use the FPS algorithm to sample M points from the point cloud, that is, sample M points from $\{v_i\}_{i=1}^N$. The FPS algorithm is used to select points that are farthest from each other in the 3D space, which is widely used in point cloud related models [8, 19–21]. For each sampled point, we calculate the Euclidean distance with its neighbours in the 3D space, and group its N_c nearest neighbours into a cluster. Note that some points can be included in two or more clusters as our goal is to find the same number (N_c) of nearest points to each cluster to be used as input for the keypoint detection module.

2.2. Keypoint detection

The module to detect a 3D keypoint for each cluster is trained without using ground-truth annotation of keypoints. We train the module, consisting of a three-layer perceptron and two fully connected layers, using as loss functions, a centre loss, a multi-view consistency loss, a pose loss and a silhouette consistency loss.

The *centre loss*, $\mathcal{L}_{cen}$, ensures that the detected keypoints are evenly distributed on the target object. This loss measures the distance between the detected keypoint, k_i^t, from the i^{th} cluster at t, and the cluster centre, c_i^t:

$$\mathcal{L}_{cen} = \frac{1}{M}\sum_{i=1}^{M} \Big| ||k_i^t - c_i^t||^2 - \delta \Big|, \tag{2}$$

where M is the number of clusters and δ can be considered as

a margin for k_i^t to be around the cluster centre.

The *multi-view consistency loss*, $\mathcal{L}_{mvc}$, measures the temporal consistency between the keypoints:

$$\mathcal{L}_{mvc} = \frac{1}{M}\sum_{i=1}^{M}||k_i^t - (\Delta R_{gt}^t \cdot k_i^{t-1} + \Delta T_{gt}^t)||^2, \quad (3)$$

where $\Delta p_{gt}^t = [\Delta R_{gt}^t|\Delta T_{gt}^t]$ is the ground-truth 6D pose change between $t-1$ and t.

Eq. 3 encourages keypoints consistency across two time instants, but does not guarantee that these keypoints are optimal for pose tracking. To this end, we add a *pose loss* [1], which measures the error between the 3D rotation and translation estimated from the keypoints and their corresponding ground-truths, ΔR_{gt}^t and ΔT_{gt}^t, respectively. This loss includes the rotation loss, $\mathcal{L}_{rot}$, which measures the error between ΔR^t and ΔR_{gt}^t:

$$\mathcal{L}_{rot} = 2\text{arcsin}\left(\frac{1}{2\sqrt{2}}||\Delta R^t - \Delta R_{gt}^t||^2\right), \quad (4)$$

and the translation loss, $\mathcal{L}_{tra}$, which measures the error between ΔT_{gt}^t and the translation vector computed by the centroid of the detected keypoints, $\bar{k}^{t-1}$ and $\bar{k}^t$, at $t-1$ and t:

$$\mathcal{L}_{tra} = ||(\bar{k}^t - \bar{k}^{t-1}) - \Delta T_{gt}^t||^2. \quad (5)$$

The clusters centres may not lie on the surface of a target object (silhouette) due to the outliers in the point cloud. We add a *silhouette consistency loss* [1,22], $\mathcal{L}_{sil}$, which alleviates the impact of outliers on the model. This loss encourages the detected keypoints to be close to the object surface:

$$\mathcal{L}_{sil} = \frac{1}{M}\sum_{i=1}^{M}||k_i^t - u_i||^2, \quad (6)$$

where $u_i \in \mathbb{R}^3$ is the nearest point to k_i^t in the unit sphere[2], whose origin coincides with the centroid of the target object point cloud.

We train our model combining the loss functions as:

$$\mathcal{L} = \lambda_1\mathcal{L}_{cen} + \lambda_2\mathcal{L}_{mvc} + \lambda_3(\mathcal{L}_{rot} + \mathcal{L}_{tra}) + \lambda_4\mathcal{L}_{sil}, \quad (7)$$

where λ_1, λ_2, λ_3 and λ_4 are the hyperparameters that control the effect of each loss.

To ensure the consistency of the keypoints from $t-1$ and t, we initialise the points for the FPS algorithm [11] at t using the detected keypoints at $t-1$. Using the detected 3D keypoint set at $t-1$:

$$K^{t-1} = \{k_1^{t-1}, k_2^{t-1}, ..., k_M^{t-1}\}, \quad (8)$$

and the 3D keypoint set at t:

$$K^t = \{k_1^t, k_2^t, ..., k_M^t\}, \quad (9)$$

we estimate the change of 6D pose, Δp^t, by least-squares optimisation [23]:

$$\Delta p^t = \underset{\Delta R^t, \Delta T^t}{\text{argmin}} \sum_{i=1}^{M}||k_i^t - (\Delta R^t \cdot k_i^{t-1} + \Delta T^t)||^2. \quad (10)$$

3. VALIDATION

We evaluate the models on the NOCS-REAL275 dataset [12], which contains 3 categories of rotational symmetric objects (*bottle*, *bowl* and *can*) and 3 categories of asymmetric object (*camera*, *laptop* and *mug*). The training set has 7 real videos with 3 instances of each category in total, annotated with ground-truth poses. The testing set has 6 real videos with 3 different unseen instances within each category, resulting in 18 different object instances and 3,200 frames in total.

We compare our model (Ours-Agn) with other six models, which are listed below[3].

- NOCS [12], a category-level 6D pose estimation model which employs a shared representation for all instances within a category.
- ICP (iterative closest point) [24], a point-set alignment model that can be used to obtain the 6D pose change between two point clouds.
- KeypointNet [22], a category-level 3D keypoint detector that detects 3D keypoint from a point cloud, which can be used to estimate the pose change between the object in two different views.
- TEASER++ [25], a deep learning based model for point cloud registration, which redesigns the least-squares optimisation to mitigate the effect of outliers on point cloud registration.
- MaskFusion [26], an object recognition, tracking and reconstruction model.
- 6-PACK [1], a category-level 6D pose tracker. We consider two versions of this tracker, namely 6-PACK-Spe, a category-level model that trains one model for each category; and 6-PACK-ASY, trained on the objects in 3 asymmetric categories, *camera*, *laptop* and *mug*. We further train a single 6-PACK model on the objects of multiple categories for fair comparison with Ours-Agn. Noting that 6-PACK uses a different coordinate systems for training models on the symmetric objects for symmetry invariance, we train the single 6-PACK model.

[2]The unit sphere is from a resized 3D bounding box at initialisation and is provided by 6-PACK [1].

[3]6D pose tracking results of NOCS [12], ICP [24] and KeypointNet [22] are provided by [1], while 6D pose tracking results of TEASER++ [25] and MaskFusion [26] are provided by [2].

We use four evaluation measures:

- the percentage of results with orientation error smaller than 5° and translation error smaller than 5cm (5°5cm);
- the percentage of volume overlap between the detection and ground-truth 3D bounding box that is larger than 25% (IoU25, or Intersection over Union);
- the orientation error in degrees (R_{err});
- the translation error in centimetres (T_{err}).

All modules in Ours-Agn are trained from scratch on NOCS-REAL275. During training, we randomly select pairs of consecutive images as input. The proposed model detects matched keypoints from input images, and changes the detected keypoint location during backpropagation to minimise the loss without requiring ground-truth annotation of keypoints. We use the Adam [27] optimiser with an initial learning rate of 0.00001 and 500 epochs, using PyTorch and NVIDIA 1080Ti. During testing, we followed the 6-PACK strategy, that is, given an initial 6D pose $p^0 = [R^0|T^0]$ of the target object, the 6D pose at t can be obtained by $R^t = R^0 \cdot \Delta R^{t,0}$ and $T^t = T^0 + \Delta T^{t,0}$, where $\Delta R^{t,0}$ and $\Delta T^{t,0}$ are estimated by matching keypoint set at 0 and t.

We follow the evaluation protocol defined in 6-PACK [1] and used a perturbed ground-truth object pose for initialisation. The perturbation adds a uniformly sampled random translation within a 4cm range to evaluate robustness against a noisy initial pose. We sample 500 points for the target object in each frame of the RGB-D image. We set $\delta = 0.15$, $M = 8$, $N_c = 64$, $\lambda_1 = 0.2$, $\lambda_2 = 0.3$, $\lambda_3 = 0.3$ and $\lambda_4 = 0.2$. Our model includes 20.51M parameters and can run at 35 frames per second for 640×480 images.

To compare with 6-PACK-ASY, we train our pipeline on the objects in 3 asymmetric categories (*camera*, *laptop* and *mug*), and term the resulting model Ours-ASY. The parameter settings are the same with Ours-Agn.

Table 1 shows the 6D pose estimation results of the models under analysis. Ours-Agn outperforms in most cases ICP, KeypointNet, NOCS, MaskFusion, and TEASER++. TEASER++ achieves good tracking performance. 6-PACK-Spe has a marginally better performance than Ours-Agn. However, 6-PACK-Spe is a category-level model that trains one model for each category, whereas Ours-Agn trains one model to track the objects regardless of their categories. Furthermore, 6-PACK-Spe requires target object centroid as an anchor for keypoint detection, whereas Ours-Agn does not require any ground-truth of keypoints or object centroid. Ours-ASY outperforms 6-PACK-ASY, and 6-PACK-ASY is worse than 6-PACK-Spe.

Table 1. Evaluation of 6D pose estimation on the NOCS-REAL275 dataset.

Category	Measure	NOCS	ICP	KN	TE	MF	6-PACK		**Ours**	
							Spe	ASY	Agn	ASY
bottle	5°5cm ↑	5.5	10.1	5.9	13.9	15.5	24.5	-	22.3	-
	IoU25 ↑	48.7	29.9	23.1	100.0	51.4	91.1	-	88.7	-
	R_{err} ↓	25.6	48.0	28.5	17.0	36.7	15.6	-	16.2	-
	T_{err} ↓	14.4	15.7	9.5	2.7	11.3	4.0	-	5.2	-
bowl	5°5cm	62.2	40.3	16.8	35.5	32.3	55.0	-	54.1	-
	IoU25	99.6	79.7	74.7	99.9	71.4	100.0	-	97.5	-
	R_{err}	4.7	19.0	9.8	10.6	12.3	5.2	-	6.2	-
	T_{err}	1.2	4.7	8.2	1.8	5.3	1.7	-	2.1	
camera	5°5cm	0.6	12.6	1.8	10.7	11.7	10.1	9.2	11.2	11.9
	IoU25	90.6	53.1	30.9	99.9	60.8	87.6	87.1	88.6	90.1
	R_{err}	33.8	80.5	45.2	18.8	43.0	35.7	36.2	40.8	38.1
	T_{err}	3.1	12.2	8.5	2.8	11.1	5.6	6.1	6.2	5.7
can	5°5cm	7.1	17.2	4.3	11.7	8.8	22.6	-	23.4	-
	IoU25	77.0	40.5	42.6	100.0	49.7	92.6	-	91.1	-
	R_{err}	16.9	47.1	28.8	20.4	34.9	13.9	-	14.2	-
	T_{err}	4.0	9.4	13.1	2.7	9.3	4.8	-	4.7	-
laptop	5°5cm	25.5	14.8	49.2	40.9	73.9	63.5	61.7	62.5	64.2
	IoU25	94.7	50.9	94.6	99.9	99.9	98.1	97.2	96.2	98.8
	R_{err}	8.6	37.7	6.5	7.2	3.4	4.7	5.5	5.2	4.8
	T_{err}	2.4	9.2	4.4	2.6	3.5	2.5	2.8	2.7	2.2
mug	5°5cm	0.9	6.2	3.1	7.5	16.4	24.1	23.9	25.3	26.0
	IoU25	82.8	27.7	52.0	99.9	56.2	95.2	94.1	97.2	97.5
	R_{err}	31.5	56.3	61.2	23.0	40.6	21.3	22.4	22.3	21.5
	T_{err}	4.0	9.2	6.7	2.4	9.2	2.3	2.5	2.1	2.0

6-PACK-Spe is trained on each specific category and tested on the data of the same category. Ours-Agn is trained by using all categories and tested on each category. KEY – MF: MaskFusion [26]; TE: TEASER++ [25]; KN: KeypointNet [22].

4. CONCLUSION

We presented a model for category-agnostic 6D pose tracking. The key idea is to separate the target object into parts and to train a cluster-based keypoint detection network without the need of ground-truth annotation of the keypoints. These keypoints are evenly distributed on the object and therefore suitable for pose estimation. The results of the proposed category-agnostic model are competitive to those of category-level 6D pose tracking. Future work includes addressing pose tracking under heavy occlusions and using, as post-processing, pose graph optimisation [2].

5. REFERENCES

[1] C. Wang, R. Martín-Martín, D. Xu, J. Lv, C. Lu, L. Fei-Fei, S. Savarese, and Y. Zhu, "6-PACK: Category-level 6D pose tracker with anchor-based keypoints," in *IEEE Int. Conf. Robotics Autom.*, 2020.

[2] B. Wen and K. Bekris, "BundleTrack: 6D pose tracking for novel objects without instance or category-level 3D models," in *IEEE Int. Conf. Intell. Robot Syst.*, 2021.

[3] M. Wüthrich, P. Pastor, M. Kalakrishnan, J. Bohg, and S. Schaal, "Probabilistic object tracking using a range camera," in *IEEE Int. Conf. Intell. Robot Syst.*, 2013.

[4] C. Choi and H. I. Christensen, "RGB-D object tracking: A particle filter approach on GPU," in *IEEE Int. Conf. Intell. Robot Syst.*, 2013.

[5] J. Issac, M. Wüthrich, C. G. Cifuentes, J. Bohg, S. Trimpe, and S. Schaal, "Depth-based object tracking using a robust gaussian filter," in *IEEE Int. Conf. Robotics Autom.*, 2016.

[6] J. Deutscher, A. Blake, and I. Reid, "Articulated body motion capture by annealed particle filtering," in *Conf. Comput. Vis. Pattern Recognit.*, 2000.

[7] B. Wen, C. Mitash, B. Ren, and K. E. Bekris, "SE(3)-TrackNet: Data-driven 6D pose tracking by calibrating image residuals in synthetic domains," in *IEEE Int. Conf. Intell. Robot Syst.*, 2020.

[8] X. Deng, A. Mousavian, Y. Xiang, F. Xia, T. Bretl, and D. Fox, "PoseRBPF: A Rao–Blackwellized particle filter for 6D object pose tracking," *IEEE Trans. Robotics*, vol. 37, no. 5, 2021.

[9] K. Murphy and S. Russell, "Rao-Blackwellised particle filtering for dynamic bayesian networks," in *Sequential Monte Carlo Methods in Practice*, pp. 499–515. Springer, 2001.

[10] Y. Ono, E. Trulls, P. Fua, and K. M. Yi, "LF-Net: Learning local features from images," in *Adv. Neural Inf. Process. Syst.*, 2018.

[11] C. Moenning and N. A. Dodgson, "A new point cloud simplification algorithm," in *Proc. Int. Conf. Vis. Imaging and Image Proces.*, 2003.

[12] H. Wang, S. Sridhar, J. Huang, J. Valentin, S. Song, and L. J. Guibas, "Normalized object coordinate space for category-level 6D object pose and size estimation," in *Conf. Comput. Vis. Pattern Recognit.*, 2019.

[13] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Conf. Comput. Vis. Pattern Recognit.*, 2016.

[14] C. R. Qi, H. Su, K. Mo, and L. J. Guibas, "PointNet: Deep learning on point sets for 3D classification and segmentation," in *Conf. Comput. Vis. Pattern Recognit.*, 2017.

[15] C. Wang, D. Xu, Y. Zhu, R. Martín-Martín, C. Lu, L. Fei-Fei, and S. Savarese, "DenseFusion: 6D object pose estimation by iterative dense fusion," in *Conf. Comput. Vis. Pattern Recognit.*, 2019.

[16] Y. He, W. Sun, H. Huang, J. Liu, H. Fan, and J. Sun, "PVN3D: A deep point-wise 3D keypoints voting network for 6DoF pose estimation," in *Conf. Comput. Vis. Pattern Recognit.*, 2020.

[17] R. Mehra, P. Tripathi, A. Sheffer, and N. J. Mitra, "Visibility of noisy point cloud data," *Computers & Graphics*, vol. 34, no. 3, pp. 219–230, 2010.

[18] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask R-CNN," in *Int. Conf. Comput. Vis.*, 2017.

[19] C. R. Qi, L. Yi, H. Su, and L. J. Guibas, "PointNet++: Deep hierarchical feature learning on point sets in a metric space," in *Adv. Neural Inf. Process. Syst.*, 2017.

[20] C. R. Qi, K. He, and L. J. Guibas, "Deep hough voting for 3D object detection in point clouds," in *Int. Conf. Comput. Vis.*, 2019.

[21] S. Peng, Y. Liu, Q. Huang, X. Zhou, and H. Bao, "PVNet: Pixel-wise voting network for 6DoF pose estimation," in *Conf. Comput. Vis. Pattern Recognit.*, 2019.

[22] S. Suwajanakorn, N. Snavely, J. Tompson, and M. Norouzi, "Discovery of latent 3D keypoints via end-to-end geometric reasoning," *Adv. Neural Inf. Process. Syst.*, 2018.

[23] K. S. Arun, T. S. Huang, and S. D. Blostein, "Least-squares fitting of two 3D point sets," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 9, no. 5, pp. 698–700, 1987.

[24] Q. Zhou, J. Park, and V. Koltun, "Open3D: A modern library for 3D data processing," arXiv:1801.09847 [cs.CV], 2018, `http://www.open3d.org/`.

[25] H. Yang, J. Shi, and L. Carlone, "TEASER: Fast and certifiable point cloud registration," *IEEE Trans. Robotics*, vol. 37, no. 2, pp. 314–333, 2020.

[26] M. Runz, M. Buffier, and L. Agapito, "MaskFusion: Real-time recognition, tracking and reconstruction of multiple moving objects," in *Proc. IEEE/ACM Int. Symp. Mixed and Augmented Reality*, 2018.

[27] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.