Learning Better Keypoints for Multi-Object 6DoF Pose Estimation

Yangzheng Wu , Michael Greenspan
RCV Lab, Dept. of Electrical and Computer Engineering, Ingenuity Labs,
Queen's University, Kingston, Ontario, Canada
{y.wu, michael.greenspan}@queensu.ca

Abstract

We address the problem of keypoint selection, and find that the performance of 6DoF pose estimation methods can be improved when pre-defined keypoint locations are learned, rather than being heuristically selected as has been the standard approach. We found that accuracy and efficiency can be improved by training a graph network to select a set of disperse keypoints with similarly distributed votes. These votes, learned by a regression network to accumulate evidence for the keypoint locations, can be regressed more accurately compared to previous heuristic keypoint algorithms. The proposed KeyGNet, supervised by a combined loss measuring both Wasserstein distance and dispersion, learns the color and geometry features of the target objects to estimate optimal keypoint locations. Experiments demonstrate the keypoints selected by KeyGNet improved the accuracy for all evaluation metrics of all seven datasets tested, for three keypoint voting methods. The challenging Occlusion LINEMOD dataset notably improved ADD(S) by +16.4% on PVN3D, and all core BOP datasets showed an AR improvement for all objects, of between +1% and +21.5%. There was also a notable increase in performance when transitioning from single object to multiple object training using KeyGNet keypoints, essentially eliminating the SISO-MIMO gap for Occlusion LINEMOD.

1. Introduction

Estimating the pose of objects in a scene is a fundamental problem in computer vision [24, 34, 54], which enables a number of important applications such as robot grasping operations [29] and augmented reality [24]. The most basic formulation, called Six Degree-of-Freedom Pose Estimation (6DoF PE), recovers the 3DoF translation and 3DoF rotation parameters of an object that has undergone a rigid transformation. The research community has expanded its efforts to address more general variations, such as allowing deformable transformations [30, 31] and recent one shot training scenarios [7, 42, 45] which estimate poses without
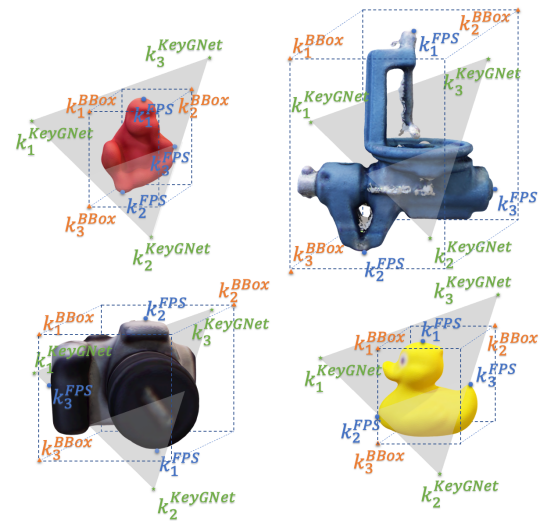

Figure 1. Keypoints sampled by FPS (blue dot), BBox (orange triangle), and KeyGNet (green star). FPS and BBox keypoints are based on the geometry of single objects, while KeyGNet keypoints are learned from the color and geometry of all dataset objects.

groundtruth values on novel (never-seen and unknown) objects, albeit with a performance reduction. Nevertheless, 6DoF PE remains an active area of investigation, with the efficient pose estimation of multiple objects being a particular focus recently [46].

There are two main approaches to solve 6DoF PE. In the first, the pose is directly regressed by the network [9, 10, 13, 28, 41, 54], with the network's output represented as either rotation angles and translation offsets [28], a transformation matrix [56], or quaternions [54]. Regression methods are relatively efficient and often implemented as end-to-end trainable architectures [28, 41, 54]. While early versions of the direct regression approach tended to lack accuracy [54], this has been improved upon recently [56].

The second approach are keypoint-based methods [21, 22, 40, 52, 53], which are the main alternative to direct regression methods. For the 6DoF PE problem, keypoints are defined as a set of 3D coordinates, expressed within an
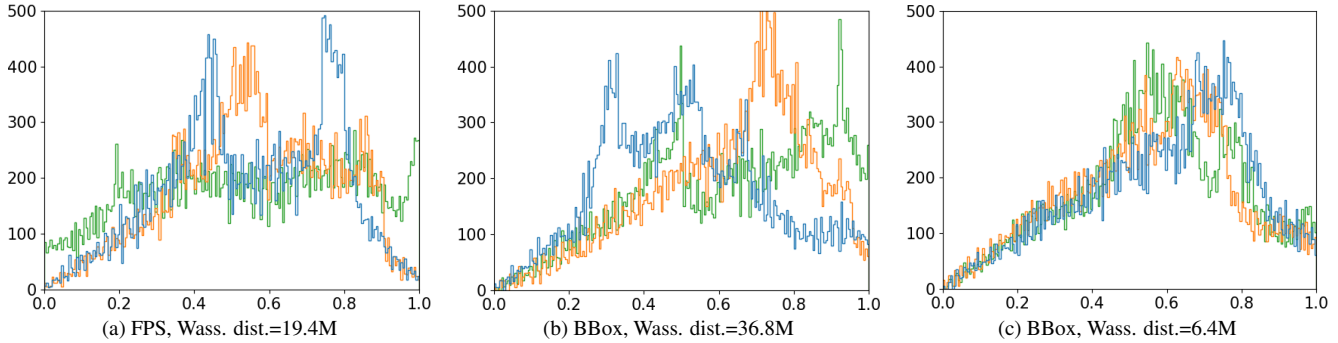
Figure 2. Histograms of Wasserstein distances to LINEMOD driller model surface points, for 3 sets of 3 keypoints: a) FPS keypoints, b) BBox keypoints with maximum Wasserstein distance, c) BBox keypoints with minimum Wasserstein distance.

object-centric coordinate reference frame. Keypoint-based methods are typically two-stage processes, starting with a keypoint regression network that first estimates keypoint locations within an image. The second stage then uses model-to-image keypoint correspondences to estimate pose, often making use of classical methods such as RANSAC or voting-based approaches to increase robustness [4, 16, 26]. Keypoint methods have been shown to be among the most accurate solutions to 6DoF PE, with recent works continuing to evolve and advance this approach [8, 23, 55, 57].

Despite the success of keypoint methods, the selection of the pre-defined object-centric keypoint locations has been overwhelmingly based on just two approaches, the first of which is Farthest Point Sampling (FPS) [22, 40]. FPS samples points on an object surface based on their relative proximities, and was originally developed for progressive image sampling [15] and subsequently repurposed for keypoint selection [40]. The second approach selects a subset of the corners of an object bounding box (BBox) [52, 53]. Both of these approaches generate keypoints purely based on the 3D surface geometry of the set of objects, and ignore other appearance information, such as color. Both approaches are also heuristic, with their main objective being to produce keypoints that are geometrically dispersed, and which fall on [22, 40], or close to [52, 53], the objects' surfaces. Their main constraint is that the keypoints are sufficient in number (i.e. ≥ 3) and distributed in a way (i.e. non-collinearly) so that object transformations can be recovered from the downstream model-to-image point correspondences, e.g. using PnP [16] or a suitable alternative [26].

In this paper, we cast attention to the generation of the pre-defined keypoints themselves. We show that a data-driven generation of the initial object-centric keypoint set can serve to improve the accuracy and the efficiency of existing keypoint-based 6DoF PE methods. A graph network is trained to optimize a disperse set of keypoints with similarly distributed votes for keypoint voting 6DoF PE methods [22, 40, 52, 53]. The network that encodes the geometry

and color information is supervised in a manner that regularizes the learning process, by considering both the distribution of votes to each keypoint, as well as the dispersion (i.e. geometrical sparseness) among them. Specifically, one loss term considers the Wasserstein similarity of the histograms of voters for each keypoint, and a separate loss term enforces keypoint dispersion. Some examples of our learned keypoints, along with heuristically generated FPS and BBox keypoints, are shown in Fig. 1. Our main contributions are:

- We introduce a novel loss function and a graph convolutional network to learn to generate a set of keypoints for a given set of objects.

- We experimentally demonstrate that our generated KeyGNet keypoint sets improve performance of existing keypoint-based 6DoF PE methods. The method not only increases accuracy on networks trained on single objects, but also reduces the performance gap between single and mutliple object scenarios. When using the learned keypoint locations, the training time for the 6DoF PE methods is also reduced.

To our knowledge, this is the first work that learns the location of pre-defined keypoints for 6DoF PE, rather than generating them heuristically. The main innovation of this work is the concept of learning keypoint locations, and the specific approach to achieve this. The performance improvement is in some cases significant, increasing accuracy by between 1% and 21.5%, and reducing training convergence time by between $6h$ and $11h$. Our code is available at: https://github.com/aaronWool/keygnet.

1.1. Motivation

Existing keypoint methods [22, 40, 52, 53] use regression networks to estimate a quantity that geometrically relates each image pixel (and/or point) to each keypoint. A variety of such quantities have been explored in the literature,

including the offset [22], direction vector [40], and radial distance [52, 53] between points. Once estimated, these quantities are used to cast votes in an accumulator space, the collection of which allows for the robust estimate of the keypoint locations in the scene.

Fig. 2 illustrates three histograms of the radial distance quantity [52, 53] from each point on the surface of an object's CAD model (the LINEMOD driller), to each of three sets of three keypoints. Each keypoint set was chosen using a different keypoint selection method. Each histogram bin represents the number of votes for that bin value that results using a voting scheme. Fig. 2a shows the histogram of this distribution with keypoints selected using Farthest Point Sampling (FPS). Fig. 2b shows the histogram when the three keypoints are chosen from the eight corners of the bounding box which maximize the Wasserstein distance. Fig. 2c shows the histogram of those three bounding box keypoints with minimal Wasserstein distance.

It can be seen that both FPS and maximum Wasserstein BBox keypoints have distributions with a greater variance among the three keypoints, compared to those selected by minimum Wasserstein distance wherein the distributions of votes are more similar. As the majority of methods [22, 40, 52] train a single network for all keypoints, a larger variance between the quantities regressed for each keypoint results in a scenario similar to class imbalance [19] in a classification network. In contrast, a reduced variance of the regression quantities, as in Fig. 2c, similar to regularization techniques, can allow for better convergence of the network resulting in both more accurate estimates and faster training, as demonstrated in Section 4.

We repeated this test, and found that the lower variation for the minimum Wasserstein keypoints occurred consistently for all LINEMOD objects. This preliminary test motivated us to further explore keypoint selection, and ultimately develop a network structure to learn keypoints that result in similarly distributed votes.

2. Related Work

Keypoint Extraction Methods identify keypoint locations in a scene, and have been applied to a variety of computer vision-related problems such as Simultaneous Localization And Mapping (SLAM) [17, 48], Neural Radiance Field (NeRF) [12], and Non-Rigid Structure from Motion (NrSFM) [30, 31]. The main objective of these methods is to improve the effective detection of keypoints for better overall performance. Some of these methods [8, 55] employ a trainable network to estimate keypoints with confidence scores in order to provide redundancies for the consecutive downstream processes. Others [6, 53] apply logical or geometric constraints to the keypoints during training. More recent methods [20, 47] embed keypoints into the latent space of a network so that these latent keypoints can also be train-

able making the overall structure end-to-end.

Whereas in the above-described methods, the keypoints are not fixed with respect to some world coordinate reference frame, keypoints have also been pre-defined at fixed locations for problems such as facial recognition [1, 11] and human pose estimation [6, 43], for which skeleton joints appear as an obvious choice for keypoint locations to facilitate modeling human motion. Keypoint selection for some other problems, however, such as 6DoF PE, is not as apparent.

Most keypoint optimization methods [3, 14] focus on fine-tuning the post-processing ignoring the impact of the keypoint selection process. Very few methods [5, 44] address such issues, and find that the overall performance can be improved by altering pre-defined keypoints selection.

Keypoint-based 6DoF PE Methods [22, 40, 52, 53] exhibit relatively good accuracy compared to viewpoint-based methods [32, 38, 49] or direct regression [28, 36, 54] methods. Some keypoint-based methods [36, 37, 39] generate hypotheses of the pre-defined keypoint locations by training a network. The hypotheses are in the form of probability heat maps, and are often filtered by a mask, estimated by a detection network in order to remove background pixels. These masks are typically noisy due to the occlusion of scenes in 6DoF PE, and performance can heavily rely on the robustness of the final least square fitting algorithms.

Keypoint voting-based methods [21, 22, 40, 52, 53], however, can better accommodate noise. The networks cast votes, typically one per pixel, to accumulate evidence for each keypoint. Voting adds redundancy to the detection mask, and due to the highly redundant and independent nature of voting, the resulting keypoints can be more precise, leading to better overall pose estimation.

Distribution Similarity Measures are widely used in ML to quantify the uncertainty of distributions, inspired by information theory. There are a variety such metrics including KL Divergence (KL Div), JS Divergence (JS Div), and Wasserstein Distance [27]. Cross-entropy [35] is widely used for classification tasks. It simplifies the similarity measure by removing the relative entropy of groundtruth in KL Div, which is consistent during training.

WGAN [2] justified that Wasserstein distance can optimize GAN training by transforming classification into a regression problem, so that a linear gradient can be created compared to a traditional GAN. Unlike KL Div and JS Div, Wasserstein distance is a symmetric metric. They also demonstrate that Wasserstein distance can still be measured when two distributions do not overlap, which most other metrics cannot. This often happens in GAN training, and can occur in keypoint voting-based 6DoF PE when a regression network estimates votes for multiple keypoints.
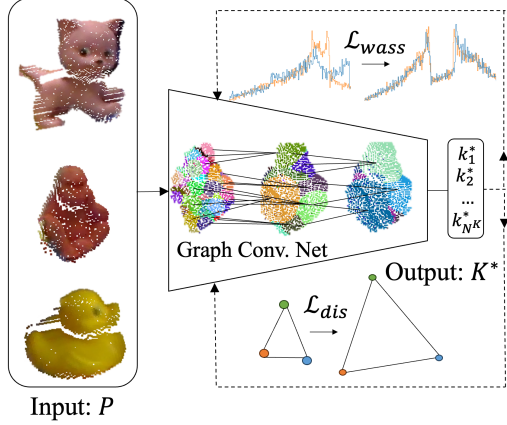
Figure 3. Overview of Keypoint Graph Network (KeyGNet). The network is trained to estimate optimized dispersed keypoint locations with similarly distributed votes for all objects in a dataset.

3. Method: Keypoint Graph Network

The preliminary distribution similarity test of keypoint votes motivated us to design a network that learns keypoint locations for a set of objects. The network, as illustrated in Fig. 3, is trained to maximize the similarity of the regression quantities between surface points and keypoints. It is based on a Graph Convolutional Network whose nodes are all points on the visible (non-occluded) surface of the object of interest in each scene of the training data.

Our loss function comprises two terms, the first of which enforces the keypoint votes to distribute similarly for each keypoint. Wasserstein distance was chosen as the similarty measure, as it has been shown to outperform other popular similarity measures in learned networks [2,18]. The second keypoint dispersion term increases the separation of keypoints, which serves to improve the accuracy of final transformation estimation [53]. Our network is trained using all objects in a dataset, resulting in a single set of keypoints for all objects, which makes the method appropriate for Multiple Instance Multiple Object (MIMO) tasks [46].

Let $K = \{k_j\}_1^{N_K}$ be a set of keypoints within an object-centric frame, and let $S = \{p_i\}_1^{N_S}$ be a point cloud comprising the non-occluded, visible surface of an object of interest in a scene. For each point p_i in S, a *vote* v_i^j is defined as the regression quantity between p_i and keypoint k_j. For example, if radial voting is used [53], then vote v_i^j will be a scalar distance, whereas for offset voting [22], v_i^j will be a 3D displacement, etc.

The task is to estimate a keypoint set $K^* = \{k_j^*\}_1^{N_K}$ that satisfies two conditions. The first condition is that votes $V^j = \{v_i^j\} \ \forall \ p_i \in S$ all share a similar distribution:

$$V^j \sim V^i \quad \forall \ i, j \in [1 \ldots N_K]. \tag{1}$$

The second condition is that the keypoints k_j^* are geometrically dispersed, so that they lie not too close to each other:

$$\arg\max \sum_{i=1}^{N_K} \sum_{j=i+1}^{N_K} \|k_i^* - k_j^*\|. \tag{2}$$

Our keypoint graph network (*KeyGNet*) builds on edge-conv [51], in which graphs are dynamically computed in a neighborhood determined using k-nearest neighbors within a hierarchy of Voronoi regions of increasing radii. The convolution kernel operates on the edges between points by creating a KNN graph along with a linear operation and a pooling operation. The edgeconv architecture is unmodified, with the exception of a change to the network classification by reshaping the output from $3 \times C$ classes into N_K normalized keypoints. At training, the network input is the superset P of all surfaces S in a training dataset, i.e. $P = \{S_1, S_2 \ldots S_{N_P}\}$. The output is the set of optimized keypoints $K^* = \{k_j^*\}_1^{N_K}$, which can then be used for both training and inference by any keypoint-based 6DoF PE method.

To supervise KeyGNet training, we utilize a combined loss with two terms. The first term is a Wasserstein loss $\mathcal{L}_{wass}$, which is inspired by the work of WassGAN-GP [18] involving both a critic loss and gradient penalty. The $\mathcal{L}_{wass}$ between two sets of keypoint votes V^i and V^j is:

$$\begin{aligned} \mathcal{L}_{wass} = \ &\mathbb{E}_{V^i}[D(V^i)] - \mathbb{E}_{V^j}[D(V^j)] \\ &+ \lambda \, \mathbb{E}_V[(\|\nabla_V D(V)\| - 1)^2] \end{aligned} \tag{3}$$

where $D(V^i)$ and $D(V^j)$ are histogram distributions of votes V^i and V^j for respective keypoints k_i^* and k_j^*, $D(V)$ is the joint distribution of V^i and V^j, ∇_V is a gradient calculated from V, and λ is a gradient penalty hyperparameter. Instead of applying the gradient penalty only when the generator learns to imitate groundtruth (as in WassGAN-GP [18]), we apply it to all votes. In this way, the distributions of votes are trained to be similar to each other, rather than similar to the groundtruth.

The second term is a dispersion loss $\mathcal{L}_{dis}$, which is inspired by the FPS algorithm:

$$\mathcal{L}_{dis} = e^{-\gamma\|k_i - k_j\|} \tag{4}$$

where $\|k_i - k_j\|$ is the distance separating keypoints k_i and k_j. This term reduces the loss when keypoints stay farther separated, with the value of hyperparameter γ chosen so that $\mathcal{L}_{dis} \in (0, 1]$. The combined loss will then be:

$$\mathcal{L} = \alpha\mathcal{L}_{wass} + \beta\mathcal{L}_{dis} \tag{5}$$

which is calculated for all pairs of keypoints. The relative values of α and β can evolve as training proceeds, to initially confer a greater emphasis on the $\mathcal{L}_{wass}$ term.

In summary, we train KeyGNet to learn a set of dispersed keypoints with similarly distributed votes, by adapting the

Dataset	PVNet				PVN3D				RCVPose			
	SISO		MIMO		SISO		MIMO		SISO		MIMO	
	FPS	KGN	FPS	KGN	FPS	KGN	FPS	KGN	BBox	KGN	BBox	KGN
LMO	61.3	64.8 (+3.5)	52.7	64.8 (+12.1)	66.4	69.9 (+3.5)	54.8	68.7 (**+13.9**)	73.6	76.7 (+3.1)	64.8	76.4 (+11.6)
YCB-V	77.4	79.8 (+2.4)	68.2	78.9 (+10.7)	77.8	82.6 (+4.8)	69.3	81.9 (**+12.6**)	85.2	88.7 (+3.5)	79.8	88.2 (+8.4)
TLESS	67.7	70.1 (+2.4)	62.5	69.7 (+7.2)	67.3	70.3 (+3)	63.2	70.3 (+7.1)	71.5	75.4 (+3.9)	64.3	74.9 (**+10.6**)
TUDL	91.6	93.1 (+1.5)	85.7	93.1 (+7.4)	90.1	93.4 (+3.3)	87.2	92.9 (+5.7)	97.8	98.8 (+1)	90.2	98.2 (**+8**)
IC-BIN	70.6	73.2 (+2.6)	65.7	73.2 (+7.5)	70.4	76.6 (+6.2)	67.2	76.1 (**+8.9**)	74.0	76.6 (+2.6)	69.7	76.2 (+6.5)
ITODO	48.0	49.4 (+1.4)	27.8	48.0 (+20.2)	49.5	53.9 (+9.8)	32.4	53.9 (**+21.5**)	54.7	58.1 (+3.4)	46.5	58.1 (+11.6)
HB	82.5	85.0 (+2.5)	70.9	84.7 (**+13.8**)	82.8	87.6 (+4.8)	73.5	87.2 (+13.7)	87.3	89.7 (+2.4)	76.2	89.7 (+13.5)
Average	71.3	73.6 (+2.3)	61.9	73.2 (+11.3)	72.0	76.3 (+4.3)	63.9	75.9 (**+12**)	77	80.6 (+3.6)	70.2	80.2 (+10)

Table 1. Comparison of heuristic (FPS, BBox) vs. KeyGNet (KGN) performance evaluated by BOP AR. Keypoints selected by KeyGNet improve the evaluation metrics for all seven BOP core datasets.

keypoint locations to the objects' surface geometries. The result is a single set of keypoints defined for all objects within a dataset, which can then be used within keypoint-based 6DoF PE methods.

4. Experiments

4.1. Implementation Details

The proposed KeyGNet is trained to generate a set of optimized keypoints, which we tested on a variety of state-of-the-art PE methods [22, 40, 53]. The input segments of KeyGNet are the non-occluded visible surfaces of the objects of interest within a scene. These segments are colored point clouds, derived from training images by applying the camera intrinsics and groundtruth (GT) object pose values. Layer normalization instead of batch normalization is used in order to apply the gradient penalty in $\mathcal{L}_{wass}$ [18]. We use the default value of $\lambda = 10$ in Eq. 3 suggested by WGAN-GP. The value of $\gamma = 0.5$ in Eq. 4 is based on the diameter of objects in those datasets. We train KeyGNet with a batch size of 32 fully paralleled on six RTX6000 GPUs. The optimization used is SGD, with the initial learning rate set to $lr = 1e-3$, decaying by a scale of 0.1 every 50 epochs.

We test the optimized keypoints on three keypoint-based 6DoF PE voting methods, RCVPose [53], PVNet [40] and PVN3D [22]. These methods use three distinct types of voting, i.e. radial [53], vector [40], and offset [22] respectively. As each PE method uses a different voting scheme, the $D()$ quantities in the Wasserstein loss of Eq. 3 will be different for each, leading to distinct keypoint sets for each method, even when using the same dataset. These keypoint sets are trained for each PE method, for all objects in each dataset.

We use publicly available implementations of the above three PE networks, as provided by the authors of the original works. These networks are mostly unaltered, the only change being to the output shape to accommodate the shift from SISO (Single Instance Single Object) to MIMO. The SISO structures are converted to MIMO by simply training

PE Method	Mode	Keypoint Method	LMO ADD(S)	YCB-V ADD(S) AUC	Train Time
PVNet	SISO	FPS	40.8	73.4	16h
		KGN	48.0 (+7.2)	79.1 (+5.7)	10h (-6h)
	MIMO	FPS	31.7	65.1	22h
		KGN	47.9 (+16.2)	78.1 (**+13**)	12h (-10h)
PVN3D	SISO	FPS	63.2	92.3	17h
		KGN	70.5 (+7.3)	95.7 (+3.4)	10h (-7h)
	MIMO	FPS	53.9	86.5	23h
		KGN	70.3 (**+16.4**)	94.6 (+8.1)	12h (-10h)
RCVPose	SISO	BBox	71.1	95.9	16h
		KGN	75.5 (+4.4)	97.6 (+1.7)	09h (-7h)
	MIMO	BBox	62	92.5	22h
		KGN	75.6 (+13.6)	96.6 (+4.1)	11h (**-11h**)

Table 2. Comparison of heuristic (FPS, BBox) vs. KeyGNet (KGN) performance evaluated by ADD(S). KeyGNet improves ADD(S) for both LMO and YCB-V datasets.

a single regression network for all keypoints of all dataset objects. For each training run of each PE method, all factors remained the same including the loss function, hyperparameters, network depth, and number of keypoints.

4.2. Datasets and Evaluation Metrics

We test our optimized keypoints generated by KeyGNet on all Benchmark for 6D Object Pose Estimation (BOP) [25] core datasets. BOP is a benchmark of 6DoF PE providing uniform structures and evaluation metrics for twelve 6DoF PE datasets in various application fields, among which seven are considered to be core. The BOP metrics evaluate the performance on various aspects including visibility, symmetry, and projection when an estimated pose and a GT pose are applied. The average recall (AR) of Visible Surface Discrepancy (VSD), Maximum Symmetry-Aware Surface Distance (MSSD), and Maximum Symmetry-Aware Projection Distance (MSPD) were tested, as well as the overall AR.

We also tested on the original metrics of LINEMOD (LM), Occlusion LINEMOD (LMO) [24], and YCB Video (YCB-V) in order to compare against other recent lead-
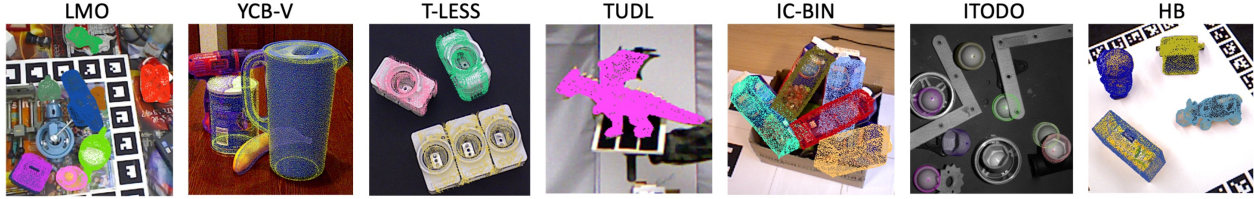
Figure 4. Visualization of poses estimated by KeyGNet keypoints. The overlapped dots are projected point clouds sampled from the object CAD models, applied with the estimated poses.

ing non-keypoint-based methods. ADD(S) was introduced with the LM dataset, and evaluates the average distance (for asymmetric objects) and the minimum distance (for symmetric objects) between estimated and GT poses. The estimate is considered correct when the average distance between the CAD model transformed by the estimated and GT poses is within a certain threshold (e.g. 10% of the object diameter). PoseCNN [54] proposed ADD(S) AUC, which was designed to be robust by evaluating ADD(S) distance over a series of thresholds. The accuracy ratio is then plotted in 2D space and connected by a curve. The overall score is the ratio of the area under the curve versus total area.

4.3. Results and Time Performance

The impact of KeyGNet keypoints for the three 6DoF PE methods is shown in Table 1 and Table 2, with some examples shown in Fig. 4. The heuristic keypoint selection method that led to the best performance was applied to each technique, i.e. FPS for PVNet and PVN3D, and BBox for RCVPose. In all cases, KeyGNet-selected keypoints boost performance, whether in SISO mode or MIMO mode, for all three methods and on all seven BOP core datasets. Specifically, in MIMO mode, the ADD(S) of PVN3D improves by 16.4% on LMO, the ADD(S) AUC of PVNet improves by 13% on YCB-Video, and the BOP AR of PVN3D improves by 12% on average on all seven BOP core datasets. These results indicate that 6DoF PE performance can be impacted and improved, just by altering the location of the pre-defined keypoints. Use of KeyGNet keypoints not only improved the metrics on average over all objects in each dataset, but they also improved the metrics for each individual object (see Supplementary Material Table.S.2, Table.S.3, and Table.S.4). The observed improvement is likely due primarily to similarly distributed votes being easier to learn by the regression network, compared to other distributions, as indicated in Fig. 2. A secondary contributing factor is keypoint dispersion, as supported by Sec. 5.4.

The use of KeyGNet keypoints has no impact on inference speed in SISO mode. In MIMO mode, inference speed is improved with a negligible accuracy reduction, as multiple objects share the same KeyGNet keypoints.

LMO	PVNet		PVN3D		RCVPose	
object	FPS	KGN	FPS	KGN	BBox	KGN
ape	-7.6	**-0.3**	-8.2	**-0.3**	-4.2	**-0.4**
can	-12.2	**0**	-11.2	**0**	-2.9	**0**
cat	-7.1	**0**	-7.6	**-0.3**	-5.7	**-0.2**
driller	-8.2	**0**	-10.3	**0**	-1.3	**+0.4**
duck	-11.3	**-0.3**	-9.7	**-0.4**	-7.2	**+0.4**
eggbox	-12.2	**0**	-10.2	**0**	-2.1	**+0.2**
glue	-6.7	**0**	-9.8	**-0.4**	-6.3	**+0.2**
holepuncher	-7.3	**0**	-7.2	**0**	-2.1	**0**
average	-9.1	**-0.1**	-9.3	**-0.2**	-6.3	**+0.1**

Table 3. SISO-MIMO ADD(S) performance gap on LMO. The change when converting from SISO to MIMO, for heuristic and KeyGNet (KGN) keypoint selection. The ADD(S) change is small for MIMO training using KGN keypoints.

5. Ablation and Tuning Experiments

5.1. SISO vs MIMO

BOP introduced the two terms SISO (Single Instance of a Single Object) and MIMO (Multiple Instances of Multiple different Objects) to distinguish between varying levels of challenge in solving 6DoF PE for different types of scenes. The majority of previous keypoint-based methods [22, 40, 52, 53] were designed to address the less challenging SISO case by only processing a single object at a time, which allows the training of unique network parameters for each distinct object. Motivated in part by the most recent edition of the BOP Challenge competition [25], several recent methods [21, 33] address the MIMO case, with multiple object categories trained within a single network.

Networks trained for MIMO are typically less accurate than those trained for SISO, likely due to a degradation of accuracy of the regression network. In contrast, our method can effectively handle the MIMO case, with little or no degradation in accuracy. One single set of optimized keypoints returned by KeyGNet for all dataset objects serves to simplify the learning process of the subsequent 6DoF PE, reducing the SISO-MIMO performance gap.

To illustrate this, we conduct an experiment comparing the performance change when a single network is first

PVNet		PVN3D		RCVPose	
FPS	KGN	FPS	KGN	BBox	KGN
-6.3	**-1.0**	-6.7	**-1.2**	-3.4	**-1.0**

Table 4. SISO-MIMO ADD(S) AUC performance gap on YCB-V. The average change for all objects, converting from SISO to MIMO, for heuristic and KeyGNet (KGN) keypoints. The change is small when trained simultaneously on all objects using KGN.

trained individually for each dataset object (SISO), and is then trained simultaneously for all objects (MIMO). In both scenarios, the network parameters were initialized randomly, from a standard normal distribution (z-distribution).

The results are shown for each LMO object in Table 3, and averaged for all YCB-V objects in Table 4. (For all YCB-V results, see Supplementary Material Table S.3.) For the three PE methods, the change in ADD(S) when training for single vs. multiple objects is listed, when using either heuristic (FPS or BBox) or KeyGNet (KGN) keypoints. The performance gap is significantly reduced when the network is converted from SISO to MIMO. While degradation existed for all objects and all three PE methods using heuristic keypoints, no degradation resulted for most LMO objects using KeyGNet keypoints, with a much smaller average degradation for YCB-V. Interestingly, RCVPose even demonstrates a small improvement of +0.1 on average in LMO. This experiment again shows that the distribution of votes has a vital impact on the performance of the network, closely correlated to overall 6DoF PE performance.

5.2. Distribution Similarity Losses of KeyGNet

There are a variety of distribution similarity metrics such as KL Divergence, JS Divergence, and cross entropy, which have been discussed in Sec. 2. The distribution of the comparison of votes in KeyGNet is similar to the generator supervision in WGAN, which used Wasserstein distance. We conduct an experiment to compare different distribution similarity losses against $\mathcal{L}_{wass}$ defined in Eq. 3.

We train KeyGNet by applying losses based on different metrics (KL Div Loss $\mathcal{L}_{kl}$, JS Div Loss $\mathcal{L}_{js}$, Cross Entropy Loss $\mathcal{L}_{ce}$) on LMO, on all three 6DoF PE methods. The results are shown in Table 5. $\mathcal{L}_{wass}$ improves the performance the most compared to the other losses, scoring highest for each individual object for each PE method. The second best performance was $\mathcal{L}_{js}$, followed by the $\mathcal{L}_{kl}$ and $\mathcal{L}_{ce}$ which had similar improvements. The result of this experiment confirmed the WGAN conclusion, and our assumption made in Sec. 2 that, even when the distribution of keypoint votes has no overlap, $\mathcal{L}_{wass}$ can still measure the differences and calculate gradients for the network compared to other similarity measures.
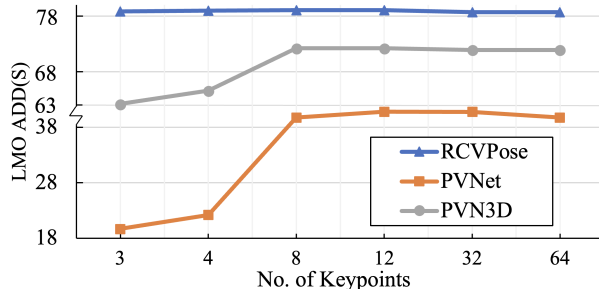

Figure 5. ADD(S) vs. number of KeyGNet keypoints, LMO dataset. PVNet and PVN3D improve for more than 3, and saturate at 8 keypoints. RCVPose saturates at 3 keypoints.

5.3. Number of Keypoints

PE methods use various numbers of keypoints, to balance time and accuracy. Some [22, 40, 50] argue that more keypoints provide redundancy to the least square fitting algorithm ultimately used in the final transformation estimation, whereas others [52, 53] use as few as three keypoints to ease the estimation task of the backbone network.

We conduct an experiment testing the impact of the number of KeyGNet keypoints on network accuracy. Fig. 5 shows the ADD(S) trend of three PE methods with optimized keypoints training with the optimized keypoints in MIMO mode, using all LMO objects. PVNet [40] and PVN3D [22] exhibited a slight overall improvement in ADD(s) as the number of keypoints increased, which saturated when there are more than eight keypoints. The ADD(S) of RCVPose [53] stayed fairly constant, independent of an increase in the number of keypoints beyond three. In subsequent experiments, we chose the number of keypoints that optimized performance for each PE method, i.e. 8 keypoints for PVNet and PVN3D, and 3 for RCVPose.

5.4. Loss Component Weights

The KGNet loss function (Eq. 5) comprises two terms. $\mathcal{L}_{wass}$ supervises the distribution of votes associated with each keypoint whereas $\mathcal{L}_{dis}$ keeps the keypoints dispersed. These two components can compete with each other simply because those keypoints with a similar distribution of votes tend to cluster within a local neighborhood. Training with an equal weight of both components can therefore cause the network to converge to a local minimum, with closely clustered keypoints, which therefore are less effective at the ultimate PE goal of transformation estimation [53].

To address this, we experiment with different combinations of α and β, which are the weighting factors of the respective loss terms. The smoothed loss curve for training RCVPose on LMO for 3 keypoints is plotted in Fig. 6. This plot shows that KeyGNet converges faster when $\alpha = 0.7$ and

LMO	PVNet				PVN3D				RCVPose			
object	$\mathcal{L}_{kl}$	$\mathcal{L}_{js}$	$\mathcal{L}_{ce}$	$\mathcal{L}_{wass}$	$\mathcal{L}_{kl}$	$\mathcal{L}_{js}$	$\mathcal{L}_{ce}$	$\mathcal{L}_{wass}$	$\mathcal{L}_{kl}$	$\mathcal{L}_{js}$	$\mathcal{L}_{ce}$	$\mathcal{L}_{wass}$
ape	+20.2	+22.0	+20.3	**+22.6**	+20.3	+22.5	+21.4	**+22.9**	+9.6	+9.6	+8.6	**+11.7**
can	+11.8	+12.8	+12.3	**+13.8**	+11.5	+12.6	+12.4	**+14.1**	+13.4	+15.4	+13.0	**+15.6**
cat	+14.9	+15.8	+15.1	**+16.9**	+15.2	+14.5	+14.8	**+17.1**	+12.6	+13.2	+13.1	**+14.1**
driller	+13.1	+14.2	+13.2	**+15.0**	+13.8	+13.2	+10.2	**+15.4**	+9.0	+10.2	+8.0	**+11.1**
duck	+15.8	+17.2	+15.9	**+18.0**	+16.1	+16.8	+16.7	**+18.0**	+21.1	+22.6	+21.8	**+23.1**
eggbox	+9.9	+10.9	+10.0	**+11.9**	+9.1	+10.0	+10.4	**+12.2**	+9.6	+10.6	+8.6	**+11.9**
glue	+16.1	+17.3	+16.8	**+18.2**	+16.6	+16.3	+15.9	**+18.2**	+6.6	+8.2	+5.4	**+8.7**
holepuncher	+10.7	+12.5	+11.2	**+13.3**	+11.2	+12.6	+12.0	**+13.7**	+9.7	+11.7	+9.1	**+12.7**
average	+14.1	+15.3	+14.3	**+16.2**	+14.2	+14.8	+14.2	**+16.4**	+11.5	+12.7	+11.0	**+13.6**

Table 5. ADD(S) Improvements of KeyGNet Distribution Similarity Losses tested on LMO in MIMO mode. keypoints selected by $\mathcal{L}_{wass}$ lead to more improvements on all three PE methods compared to the other losses. $\mathcal{L}_{js}$ improves $+1.2\%$ more than $\mathcal{L}_{kl}$ and $\mathcal{L}_{ce}$ on average.
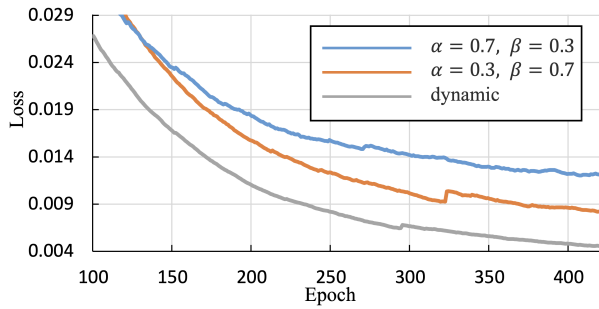

Figure 6. Impact of weighting loss components of Eq. 5. KeyGNet converges either slowly ($\alpha = 0.7, \beta = 0.3$) or less accurately ($\alpha = 0.3, \beta = 0.7$) with fixed weights. Dynamic weight scheduling improves training by swapping weight values at epoch 50.

		SISO	MIMO
LMO object	FPS-KGN	BBox-KGN	KGN
ape	**65**	64.9	**65**
can	96	96	**96.4**
cat	57.5	57.9	**58**
driller	79.9	80.2	**82.1**
duck	64.7	65.2	**65.6**
eggbox	80.7	80.7	**82.2**
glue	73.9	74.2	**75.1**
holepuncher	79.8	80.7	**81.2**
average	74.7	75	**75.6**

Table 6. ADD(S) for different initial KeyGNet inputs, tested on LMO using RCVPose. Segment initialization (KGN) is slightly more accurate than heuristic keypoint initialization.

$\beta = 0.3$, whereas it is more accurate but converges slower when $\alpha = 0.3$ and $\beta = 0.7$. We therefore train the network using the plotted dynamic schedule, which weights $\mathcal{L}_{wass}$ more heavily ($\alpha = 0.7$) initially, and shifts to weigh $\mathcal{L}_{dis}$ more heavily ($\beta = 0.7$) at epoch 50. This schedule has the benefit of converging both efficiently and accurately.

5.5. Initial Input of KeyGNet

At training, KeyGNet accepts as input all points on the visible surface of objects within a scene (segments). An alternative could input fewer initial keypoints, sampled by FPS or BBox.

To explore the impact of initialization, we conduct an experiment to compare performance with keypoints optimized on different KeyGNet inputs. One KeyGNet is trained and evaluated in SISO mode with the input of heuristically selected keypoints, with KGN optimization (FPS-KGN and BBox-KGN). An alternate KeyGNet is trained on segments, and estimates keypoints for all object categories in MIMO mode (KGN). As shown in Table 6, KeyGNet trained with object segment input data is slightly more accurate than those with heuristically defined keypoints as input. In our subsequent experiments, we trained on segments, with one set of keypoints for all objects.

6. Conclusion

In summary, we proposed KeyGNet to select a set of pre-defined dispersed keypoints with optimal similarly distributed votes for MIMO keypoint voting-based 6DoF PE. The keypoints were selected by training a graph network to make the histograms of votes share similar distributions, so that the regression network can estimate votes more accurately. Our keypoints improved the performance and training time on all seven BOP core datasets among all three SOTA methods tested, and reduced the SISO-MIMO degradation of these methods.

References

[1] Brandon Amos, Bartosz Ludwiczuk, and Mahadev Satyanarayanan. Openface: A general-purpose face recognition library with mobile applications. Technical report, CMU-CS-16-118, CMU School of Computer Science, 2016. 3

[2] Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein generative adversarial networks. In *International conference on machine learning*, pages 214–223. PMLR, 2017. 3, 4

[3] Axel Barroso-Laguna, Edgar Riba, Daniel Ponsa, and Krystian Mikolajczyk. Key. net: Keypoint detection by hand-crafted and learned cnn filters. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 5836–5844, 2019. 3

[4] Paul J Besl and Neil D McKay. Method for registration of 3-d shapes. In *Sensor fusion IV: control paradigms and data structures*, volume 1611, pages 586–606. International Society for Optics and Photonics, 1992. 2

[5] Michael Bosse and Robert Zlot. Keypoint design and evaluation for place recognition in 2d lidar maps. *Robotics and Autonomous Systems*, 57(12):1211–1224, 2009. 3

[6] Z. Cao, G. Hidalgo Martinez, T. Simon, S. Wei, and Y. A. Sheikh. Openpose: Realtime multi-person 2d pose estimation using part affinity fields. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2019. 3

[7] Hanzhi Chen, Fabian Manhardt, Nassir Navab, and Benjamin Busam. Texpose: Neural texture learning for self-supervised 6d object pose estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4841–4852, 2023. 1

[8] Hansheng Chen, Pichao Wang, Fan Wang, Wei Tian, Lu Xiong, and Hao Li. Epro-pnp: Generalized end-to-end probabilistic perspective-n-points for monocular object pose estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2781–2790, 2022. 2, 3

[9] Wei Chen, Xi Jia, Hyung Jin Chang, Jinming Duan, and Ales Leonardis. G2l-net: Global to local network for real-time 6d pose estimation with embedding vector features. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4233–4242, 2020. 1

[10] Wei Chen, Xi Jia, Hyung Jin Chang, Jinming Duan, Linlin Shen, and Ales Leonardis. Fs-net: Fast shape-based network for category-level 6d object pose estimation with decoupled rotation mechanism. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1581–1590, 2021. 1

[11] Savina Colaco and Dong Seog Han. Facial keypoint detection with convolutional neural networks. In *2020 International Conference on Artificial Intelligence in Information and Communication (ICAIIC)*, pages 671–674. IEEE, 2020. 3

[12] Kangle Deng, Andrew Liu, Jun-Yan Zhu, and Deva Ramanan. Depth-supervised nerf: Fewer views and faster training for free. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12882–12891, 2022. 3

[13] Yan Di, Ruida Zhang, Zhiqiang Lou, Fabian Manhardt, Xiangyang Ji, Nassir Navab, and Federico Tombari. Gpv-pose: Category-level object pose estimation via geometry-guided point-wise voting. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6781–6791, 2022. 1

[14] Kaiwen Duan, Song Bai, Lingxi Xie, Honggang Qi, Qingming Huang, and Qi Tian. Centernet: Keypoint triplets for object detection. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 6569–6578, 2019. 3

[15] Yuval Eldar, Michael Lindenbaum, Moshe Porat, and Yehoshua Zeevi. The farthest point strategy for progressive image sampling. *IEEE transactions on image processing : a publication of the IEEE Signal Processing Society*, 6:1305–15, 02 1997. 2

[16] Martin A Fischler and Robert C Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6):381–395, 1981. 2

[17] Ruben Gomez-Ojeda, Francisco-Angel Moreno, David Zuniga-Noël, Davide Scaramuzza, and Javier Gonzalez-Jimenez. Pl-slam: A stereo slam system through the combination of points and line segments. *IEEE Transactions on Robotics*, 35(3):734–746, 2019. 3

[18] Ishaan Gulrajani, Faruk Ahmed, Martin Arjovsky, Vincent Dumoulin, and Aaron C Courville. Improved training of wasserstein gans. *Advances in neural information processing systems*, 30, 2017. 4, 5

[19] Haibo He and Edwardo A Garcia. Learning from imbalanced data. *IEEE Transactions on knowledge and data engineering*, 21(9):1263–1284, 2009. 3

[20] Xingzhe He, Bastian Wandt, and Helge Rhodin. Latentkeypointgan: Controlling gans via latent keypoints. *arXiv preprint arXiv:2103.15812*, 2021. 3

[21] Yisheng He, Haibin Huang, Haoqiang Fan, Qifeng Chen, and Jian Sun. Ffb6d: A full flow bidirectional fusion network for 6d pose estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3003–3013, 2021. 1, 3, 6

[22] Yisheng He, Wei Sun, Haibin Huang, Jianran Liu, Haoqiang Fan, and Jian Sun. Pvn3d: A deep point-wise 3d keypoints voting network for 6dof pose estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020. 1, 2, 3, 4, 5, 6, 7

[23] Yisheng He, Yao Wang, Haoqiang Fan, Jian Sun, and Qifeng Chen. Fs6d: Few-shot 6d pose estimation of novel objects. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6814–6824, 2022. 2

[24] Stefan Hinterstoisser, Vincent Lepetit, Slobodan Ilic, Stefan Holzer, Gary Bradski, Kurt Konolige, and Nassir Navab. Model based training, detection and pose estimation of texture-less 3d objects in heavily cluttered scenes. In *Asian conference on computer vision*, pages 548–562. Springer, 2012. 1, 5

[25] Tomáš Hodaň, Martin Sundermeyer, Bertram Drost, Yann Labbé, Eric Brachmann, Frank Michel, Carsten Rother, and

Jiří Matas. Bop challenge 2020 on 6d object localization. In *European Conference on Computer Vision*, pages 577–594. Springer, 2020. 5, 6

[26] Berthold KP Horn, Hugh M Hilden, and Shahriar Negahdaripour. Closed-form solution of absolute orientation using orthonormal matrices. *JOSA A*, 5(7):1127–1135, 1988. 2

[27] Leonid V Kantorovich. Mathematical methods of organizing and planning production. *Management science*, 6(4):366–422, 1960. 3

[28] Wadim Kehl, Fabian Manhardt, Federico Tombari, Slobodan Ilic, and Nassir Navab. Ssd-6d: Making rgb-based 3d detection and 6d pose estimation great again. In *Proceedings of the IEEE international conference on computer vision*, pages 1521–1529, 2017. 1, 3

[29] Kilian Kleeberger, Christian Landgraf, and Marco F Huber. Large-scale 6d object pose estimation dataset for industrial bin-picking. In *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2573–2578. IEEE, 2019. 1

[30] Suryansh Kumar, Yuchao Dai, and Hongdong Li. Multibody non-rigid structure-from-motion. In *2016 Fourth International Conference on 3D Vision (3DV)*, pages 148–156. IEEE, 2016. 1, 3

[31] Suryansh Kumar and Luc Van Gool. Organic priors in nonrigid structure from motion. In *Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part II*, pages 71–88. Springer, 2022. 1, 3

[32] Yann Labbé, Justin Carpentier, Mathieu Aubry, and Josef Sivic. Cosypose: Consistent multi-view multi-object 6d pose estimation. In *European Conference on Computer Vision*, pages 574–591. Springer, 2020. 3

[33] Alan Li and Angela P Schoellig. Multi-view keypoints for reliable 6d object pose estimation. *arXiv preprint arXiv:2303.16833*, 2023. 6

[34] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part V 13*, pages 740–755. Springer, 2014. 1

[35] Shie Mannor, Dori Peleg, and Reuven Rubinstein. The cross entropy method for classification. In *Proceedings of the 22nd international conference on Machine learning*, pages 561–568, 2005. 3

[36] Markus Oberweger, Mahdi Rad, and Vincent Lepetit. Making deep heatmaps robust to partial occlusions for 3d object pose estimation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 119–134, 2018. 3

[37] Kiru Park, Timothy Patten, and Markus Vincze. Pix2pose: Pixel-wise coordinate regression of objects for 6d pose estimation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 7668–7677, 2019. 3

[38] Kiru Park, Timothy Patten, and Markus Vincze. Neural object learning for 6d pose estimation using a few cluttered images. In Andrea Vedaldi, Horst Bischof, Thomas Brox, and Jan-Michael Frahm, editors, *Computer Vision – ECCV 2020*,

pages 656–673, Cham, 2020. Springer International Publishing. 3

[39] Georgios Pavlakos, Xiaowei Zhou, Aaron Chan, Konstantinos G Derpanis, and Kostas Daniilidis. 6-dof object pose from semantic keypoints. In *2017 IEEE international conference on robotics and automation (ICRA)*, pages 2011–2018. IEEE, 2017. 3

[40] Sida Peng, Yuan Liu, Qixing Huang, Xiaowei Zhou, and Hujun Bao. Pvnet: Pixel-wise voting network for 6dof pose estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4561–4570, 2019. 1, 2, 3, 5, 6, 7

[41] Mahdi Rad and Vincent Lepetit. Bb8: A scalable, accurate, robust to partial occlusion method for predicting the 3d poses of challenging objects without using depth. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 3828–3836, 2017. 1

[42] Ivan Shugurov, Fu Li, Benjamin Busam, and Slobodan Ilic. Osop: a multi-stage one shot object pose estimation framework. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6835–6844, 2022. 1

[43] Iqbal Singh, Xiaodan Zhu, and Michael Greenspan. Multimodal fusion with observation points for skeleton action recognition. In *2020 IEEE International Conference on Image Processing (ICIP)*, pages 1781–1785. IEEE, 2020. 3

[44] Haowen Sun, Taiyong Wang, and Enlin Yu. A dynamic keypoint selection network for 6dof pose estimation. *Image and Vision Computing*, 118:104372, 2022. 3

[45] Jiaming Sun, Zihao Wang, Siyu Zhang, Xingyi He, Hongcheng Zhao, Guofeng Zhang, and Xiaowei Zhou. Onepose: One-shot object pose estimation without cad models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6825–6834, 2022. 1

[46] Martin Sundermeyer, Tomáš Hodaň, Yann Labbe, Gu Wang, Eric Brachmann, Bertram Drost, Carsten Rother, and Jiří Matas. Bop challenge 2022 on detection, segmentation and pose estimation of specific rigid objects. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2784–2793, 2023. 1, 4

[47] Supasorn Suwajanakorn, Noah Snavely, Jonathan J Tompson, and Mohammad Norouzi. Discovery of latent 3d keypoints via end-to-end geometric reasoning. *Advances in neural information processing systems*, 31, 2018. 3

[48] Jiexiong Tang, Ludvig Ericson, John Folkesson, and Patric Jensfelt. Gcnv2: Efficient correspondence prediction for real-time slam. *IEEE Robotics and Automation Letters*, 4(4):3505–3512, 2019. 3

[49] Gu Wang, Fabian Manhardt, Jianzhun Shao, Xiangyang Ji, Nassir Navab, and Federico Tombari. Self6d: Selfsupervised monocular 6d object pose estimation. In *European Conference on Computer Vision*, pages 108–125. Springer, 2020. 3

[50] Gu Wang, Fabian Manhardt, Federico Tombari, and Xiangyang Ji. Gdr-net: Geometry-guided direct regression network for monocular 6d object pose estimation. In *Proceed-*

ings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 16611–16621, 2021. 7

[51] Yue Wang, Yongbin Sun, Ziwei Liu, Sanjay E Sarma, Michael M Bronstein, and Justin M Solomon. Dynamic graph cnn for learning on point clouds. *Acm Transactions On Graphics (tog)*, 38(5):1–12, 2019. 4

[52] Y. Wu, A. Javaheri, M. Zand, and M. Greenspan. Keypoint cascade voting for point cloud based 6dof pose estimation. In *2022 International Conference on 3D Vision (3DV)*, pages 176–186, Los Alamitos, CA, USA, sep 2022. IEEE Computer Society. 1, 2, 3, 6, 7

[53] Yangzheng Wu, Mohsen Zand, Ali Etemad, and Michael Greenspan. Vote from the center: 6 dof pose estimation in rgb-d images by radial keypoint voting. In *Computer Vision– ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part X*, pages 335–352. Springer, 2022. 1, 2, 3, 4, 5, 6, 7

[54] Yu Xiang, Tanner Schmidt, Venkatraman Narayanan, and Dieter Fox. Posecnn: A convolutional neural network for 6d object pose estimation in cluttered scenes. 2018. 1, 3, 6

[55] Heng Yang and Marco Pavone. Object pose estimation with statistical guarantees: Conformal keypoint detection and geometric uncertainty propagation. *arXiv preprint arXiv:2303.12246*, 2023. 2, 3

[56] Linfang Zheng, Chen Wang, Yinghan Sun, Esha Dasgupta, Hua Chen, Ales Leonardis, Wei Zhang, and Hyung Jin Chang. Hs-pose: Hybrid scope feature extraction for category-level object pose estimation. *arXiv preprint arXiv:2303.15743*, 2023. 1

[57] Jun Zhou, Kai Chen, Linlin Xu, Qi Dou, and Jing Qin. Deep fusion transformer network with weighted vector-wise keypoints voting for robust 6d object pose estimation, 2023. 2