6DoF Object Pose Estimation via Differentiable Proxy Voting Regularizer

Xin Yu[1]
xin.yu@uts.edu.au

Zheyu Zhuang[2]
zheyu.zhuang@anu.edu.au

Piotr Koniusz[2,3]
piotr.koniusz@data61.csiro.au

Hongdong Li[2]
hongdong.li@anu.edu.au

[1] University of Technology Sydney

[2] Australian National University

[3] Data61, CSIRO, Australia

Abstract

Estimating a 6DOF object pose from a single image is very challenging due to occlusions or textureless appearances. Vector-field based keypoint voting has demonstrated its effectiveness and superiority on tackling those issues. However, direct regression of vector-fields neglects that the distances between pixels and keypoints also affect the deviations of hypotheses dramatically. In other words, small errors in direction vectors may generate severely deviated hypotheses when pixels are far away from a keypoint. In this paper, we aim to reduce such errors by incorporating the distances between pixels and keypoints into our objective. To this end, we develop a simple yet effective differentiable proxy voting regularizer (DPVR) which mimics the hypothesis selection in the voting procedure. By exploiting our voting regularizer, we are able to train our network in an end-to-end manner. Experiments on widely used datasets, *i.e.*, LINEMOD and Occlusion LINEMOD, manifest that our DPVR improves pose estimation performance significantly and speeds up the training convergence.

1 Introduction

Object pose estimation aims at obtaining objects' orientations and translations relative to a camera, and is widely used in many applications, such as robotic picking and virtual reality. Considering an object may undergo severe occlusions, different lighting conditions or cluttered backgrounds, estimating its pose, including 3D orientations and translations, from an RGB image is quite difficult but has attracted great attention.

Conventional pose estimation methods exploit hand-crafted features [19] or learned deep local features [28, 32] to establish the correspondences between an input image and a 3D model, and then estimate an object pose. However, those hand-crafted and learned image local features may not be robust to variations, such as occlusions and drastic view point changes. As a result, they may fail to estimate poses accurately or even localize objects in cluttered backgrounds. Thus, discriminative feature representations are highly desirable to account for those variations.
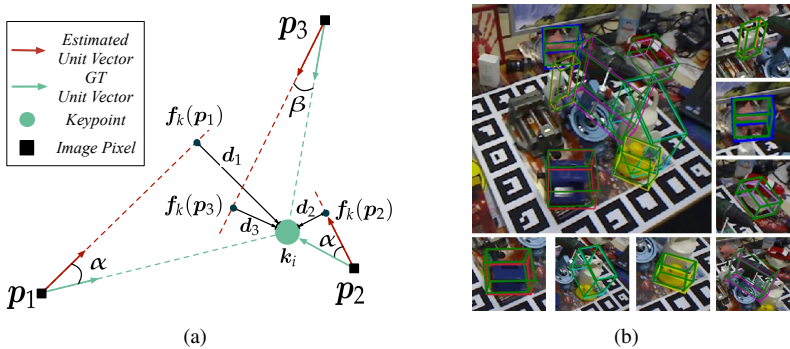
Figure 1: Motivation of our differentiable proxy voting regularizer (a) and our results on the *Occlusion LINEMOD* dataset (b). (a) when the estimation errors of direction vectors are the same (*e.g.*, α), the distance between a pixel and a keypoint affects the closeness between a hypothesis and the keypoint. (b) all the poses of the visible objects (except "Bench vise") are estimated. Green 3D bounding boxes indicate the ground-truth poses and those with other colors represent our predictions.

Recently, deep learning based object pose estimation methods have been proposed [13, 21, 51]. Several methods leverage large training data to regress object poses or coordinates directly from input images [13, 14, 27, 33, 34]. Due to severe occlusions, the predicted poses or object coordinates may be inaccurate. To increase the robustness to occlusions, voting strategies are employed to localize feature keypoints or coordinates [12, 21, 51]. Specifically, a keypoint is localized by a hypothesis with the highest vote from a vector field. In this manner, even invisible keypoints can be inferred from visible pixels. Considering that some pixels are far away from a keypoint and their direction vectors exhibit small errors, previous methods may generate hypotheses (*i.e.*, an intersection of the directional vectors of two pixels) with large deviations from the keypoint. As illustrated in Fig. 1(a), a smaller error β leads to a larger deviation d_3 compared to d_2 resulted from a larger error α.

In this paper, we propose a differentiable proxy voting regularizer (DPVR) to achieve accurate vector-field representations. DPVR takes the distances between pixels and keypoints into account to reduce hypothesis deviations caused by inaccurate direction vectors. Since sampling and inlier searching are involved in voting procedures, such as RANSAC [4], they are often indifferentiable. Therefore, it is difficult to exploit the voted keypoints and their ground-truth counterparts as supervision to train networks in an end-to-end fashion.

Instead of using voted keypoints in training, we develop a proxy hypothesis for each pixel to approximate the deviated distance between a voted keypoint (*i.e.*, a hypothesis with the highest vote) and its ground-truth. To be specific, we employ the foot of the perpendicular through a ground-truth keypoint with respect to a line generated by a pixel and its estimated direction vector as our proxy hypothesis. In this way, each pixel produces an approximated hypothesis with respect to a ground-truth keypoint. Note that, calculating the foot of the perpendicular for a keypoint is differentiable and the distance between a keypoint and a proxy hypothesis takes into account the estimated direction vector of a pixel as well as its distance to the keypoint. We therefore enforce a proxy hypothesis to be close to its corresponding keypoint in an end to end training manner. By doing so, we reduce the deviations of hypotheses and thus estimate keypoints accurately. Experiments on popular widely-used

standard datasets also demonstrate that our DPVR not only improves the pose estimation performance significantly but also accelerates the training convergence.

2 Related Work

In this section, we mainly review single RGB image based 6DoF pose estimation approaches.

Traditional methods: Conventional object pose estimation methods mainly leverage on the local feature/keypoint matching. The extracted local feature descriptors from 2D images, such as SIFT [19], need to be robust to viewpoint, illumination, rotation and scale variations. After associating the extracted local features with points on a 3D model, the object poses are solved via a Perspective-n-Point (PnP) problem [15]. However, these methods can only tackle textured objects, where local features are detectable. Other than local features, image templates [6, 9, 24, 35], and image edges [8, 17] are also exploited for pose estimation.

Deep model based methods: Due to the powerful feature representation ability of deep neural networks, deep learning based methods have demonstrated impressive results on object pose estimation [13, 21, 31].

Some methods, such as Viewpoints and Keypoints [29] and Render for CNN [26], formulate the 3D pose estimation as a classification task by discretizing the pose space and then assigning objects with discrete pose labels. Motivated by the state-of-the-art image detection methods [18, 23], pose estimation approaches are designed to localize objects while predicting their viewpoints based on the estimation of 3D bounding-boxes [13, 22, 27], features of interest [21, 31] or coordinate maps [14, 30, 33]. SSD6D [13] extends the ideas of 2D object detection and classifies localized objects with discrete poses while YOLO6D [27] regresses 3D bounding-boxes of objects. BB8 [22] firstly generates 2D segmentation masks for objects and predicts 3D bounding-boxes from the 2D masks. CPDN [34], DPOD [33] and Pix2Pose [14] regress the 2D/3D coordinates of 3D object models from images. Object poses are estimated by PnP after obtaining 2D-3D correspondences.

Since directly regressing the 2D projections of 3D points is not reliable, PoseCNN [31] firstly estimates a vector-field pointing to an object center from object pixels and then employs Hough voting to determine the center. The translations and rotations of the object are regressed subsequently by a subnetwork. Rather than only estimating a centroid, PVNet [21] votes several features of interest, while the work [12] votes the corners of a 3D bounding-box from each segmentation grid. Due to the voting strategy, the locations of estimated 2D projections are more robust to occlusions. However, small errors in a direction vector field may lead to large deviations of hypotheses.

Pose refinement: Depth images are used to refine estimated poses. For instance, PoseCNN employs the depth information and Iterative Closest Point (ICP) [1] to refine the estimated poses. Meanwhile, DeepIM [16] and DPOD [33] exploit optical flow to modify initial pose estimation by minimizing the differences between the object appearance and the 2D projection of the 3D model. Moreover, Hu *et al.* [11] develop a standalone pose refinement network by taking initial pose estimation from other methods. Hybrid-pose [25] improves PVNet by incorporating different pose representations and an additional refinement sub-network. Note that, those methods are effective when initial poses are close to their ground-truths. Therefore, generating accurate initial poses also plays a key role in the pose refinement pipeline.
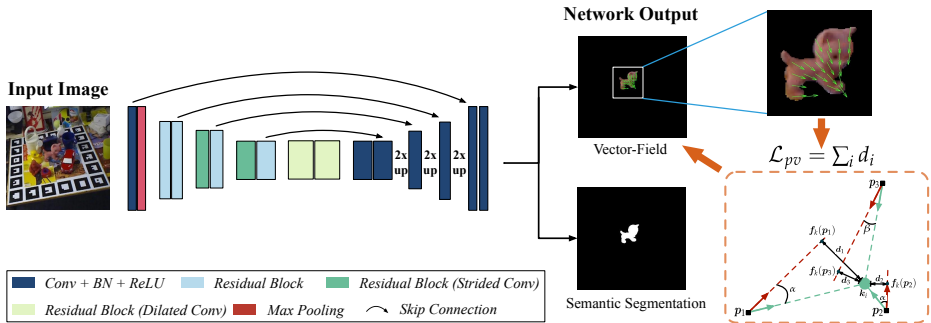
Figure 2: Illustration of our pipeline. Here, we only highlight our proposed differentiable proxy voting regularizer (DPVR). For simplicity, the vector-field regression loss and segmentation loss are omitted.

3 Proposed Method

In this paper, we focus on obtaining accurate initial pose estimation. In particular, our method is designed to localize and estimate the orientations and translations of an object accurately without any refinement. The object pose is represented by a rigid transformation $(\mathcal{R}, \boldsymbol{t})$ from the object coordinate system to the camera coordinate system, where $\mathcal{R}$ and $\boldsymbol{t}$ indicate 3D rotations and translations, respectively.

Since voting based methods [21, 51] have demonstrated their robustness to occlusions and view changes, we therefore follow the voting based pose estimation pipeline. Specifically, our method firstly votes 2D positions of the object keypoints from the vector-fields (as illustrated by the green arrows in Fig. 2) and then estimates the 6DOF pose by solving a PnP problem. Prior works regress pixel-wise vector-fields by an ℓ_1 loss. However, small errors in the vector-fields may lead to large deviation errors of hypotheses because the loss does not take the distance between a pixel and a keypoint into account. Therefore, we present a differentiable proxy voting regularizer (DPVR) to reduce such errors by mimicking the hypothesis selection in the voting procedure. Furthermore, benefiting from our DPVR, our network is able to converge much faster.

3.1 Vector-fields of keypoints

In order to estimate object poses, we firstly localize 2D keypoints of a given object. Since the sizes of objects vary significantly, we opt to estimate a vector-field for each keypoint instead of predicting their keypoint positions from images. The vector-field is composed of unit direction vectors pointing from a pixel to a certain keypoint. Different from feature keypoints extracted from image local patches, our direction vectors are estimated by a deep neural network with a larger receptive field covering a large portion of objects. In this manner, even though a keypoint is not visible, it is able to be deduced from the visible parts of an object.

Motivated by the works [21, 51], our network simultaneously outputs a segmentation mask for an object and keypoint vector-fields, as shown in Fig. 2. Specifically, if a pixel $\boldsymbol{p}$ belongs to an object, its segmentation mask will be assigned to 1. Otherwise, its segmentation mask is assigned to 0. A unit direction vector $\boldsymbol{u}_k(\boldsymbol{p})$ pointing from a pixel $\boldsymbol{p}$ to a keypoint $\boldsymbol{k}$ is expressed as $\boldsymbol{u}_k(\boldsymbol{p}) = (\boldsymbol{k} - \boldsymbol{p})/\|\boldsymbol{k} - \boldsymbol{p}\|_2$.

3.2 Differentiable proxy voting regularizer

Given an image $\mathcal{I}$ and the keypoint locations of an object $\boldsymbol{K} = \{\boldsymbol{k}_i\}, i = 1, \cdots, N,$[1] where N represents the number of the chosen keypoints, the ground-truth keypoint vector-fields are easily derived while the estimated vector-fields are generated from our network. Then, the smooth ℓ_1 loss [5] is employed to regress the ground-truth direction vectors as:

$$\mathcal{L}_{vf} = \sum_{\boldsymbol{k} \in \boldsymbol{K}} \sum_{\boldsymbol{p} \in \mathcal{M}} \ell_1(\|\boldsymbol{u}_k(\boldsymbol{p}) - \boldsymbol{v}_k(\boldsymbol{p})\|_1),$$

$$\ell_1(a) = 0.5a^2 \mathbb{I}(|a| < 1) + (|a| - 0.5)\mathbb{I}(|a| \geq 1),$$

(1)

where $\boldsymbol{v}_k(\boldsymbol{p})$ represents an estimated direction vector, $\mathcal{M}$ indicates an object mask, $\mathbb{I}(\cdot)$ is an indicator function and a represents a scalar variable.

As aforementioned, small errors in the estimated unit direction vectors may lead to large deviations of hypotheses (as illustrated by d_2 and d_3 in Fig. 1(a)), and diverse hypotheses will result in inaccurate keypoints, thus decreasing the performance of pose estimation. Unlike previous works, we take the distributions of hypotheses into account and enforce all the hypotheses to be close to the ground-truth keypoints. Assuming an object contains M pixels, there will be $M(M-1)$ hypotheses. Although there is a closed-form solution to obtaining a hypothesis given two direction vectors from two pixels, calculating all the hypotheses leads to inefficient training of deep networks especially when the resolution of an object is very large.

Since the distance between a keypoint and a point on a line is not bounded from above but from below, we opt to use the foot of the perpendicular through a keypoint with respect to a line specified from a direction vector of a pixel $\boldsymbol{p}$ to approximate a hypothesis. Note that, the distance d (in Fig. 1(a)) is the lower bound distance between a keypoint and a line. By doing so, we only need to compute M perpendiculars rather than $M(M-1)$ hypotheses, significantly reducing computations. More importantly, the distance between a keypoint $\boldsymbol{k}$ and its corresponding foot of the perpendicular $\boldsymbol{f}_k(\boldsymbol{p})$ with respect to the direction vector $\boldsymbol{v}_k(\boldsymbol{p})$ of pixel $\boldsymbol{p}$ has a close-form solution and is also differentiable. Therefore, we minimize the distance, as our differentiable proxy voting regularizer (DPVR) $\mathcal{L}_{pv}$, to force the proxy hypotheses to be close to the keypoint as follows:

$$\mathcal{L}_{pv} = \sum_{\boldsymbol{k} \in \boldsymbol{K}} \sum_{\boldsymbol{p} \in \mathcal{M}} \ell_1(\|\boldsymbol{k} - \boldsymbol{f}_k(\boldsymbol{p})\|_1) = \sum_{\boldsymbol{k} \in \boldsymbol{K}} \sum_{\boldsymbol{p} \in \mathcal{M}} \ell_1 \left(\frac{|v_k^y k^x - v_k^x k^y + v_k^x p^y - v_k^y p^x|}{\sqrt{(v_k^x)^2 + (v_k^y)^2)}} \right), \quad (2)$$

where $\boldsymbol{v}_k(\boldsymbol{p}) = (v_k^x, v_k^y)$, $\boldsymbol{p} = (p^x, p^y)$, $\boldsymbol{k} = (k^x, k^y)$ and the superscripts x, y represent horizontal and vertical coordinates respectively. Since $\boldsymbol{v}_k(\boldsymbol{p})$ is directly estimated from our network and it might not be a unit vector, there is a normalization operation in Eqn. (2) (The detailed derivations of Eqn. (2) are provided in the supplementary material). As indicated by Eqn. (2), our developed regularizer $\mathcal{L}_{pv}$ not only enforces the direction vector $\boldsymbol{v}_k(p)$ to point to keypoints but also is sensitive to the pixel locations. In other words, small errors in the direction vectors of pixels away from a keypoint will be penalized more severely in order to produce more concentrated hypotheses.

---

[1] For notation convenience, we will drop the subscript i in $\boldsymbol{k}_i$.

3.3   Network architecture and training strategy

To demonstrate the effectiveness of our proposed regularizer, we adopt the same architecture as PVNet [21], as illustrated in Fig. 2. The network is built on a pre-trained ResNet18 backbone [7] with three subsequent upsampling layers, and outputs segmentation predictions $s(\boldsymbol{p}) \in [0, 1]$ and the vector-fields of keypoints. Our overall objective is written as:

$$\mathcal{L} = \mathcal{L}_{seg} + \mathcal{L}_{vf} + \lambda \mathcal{L}_{pv}, \tag{3}$$

where $\mathcal{L}_{seg} = -\sum_{\boldsymbol{p} \in \mathcal{M}} \log(s(\boldsymbol{p}))$ is the segmentation loss, and λ is a trade-off weight. Following the training protocols of PVNet, we set the same weights 1 to $\mathcal{L}_{seg}$ and $\mathcal{L}_{pv}$. In our experiments, we set λ to $1e^{-3}$. Since the objective is a multi-task problem, the gradients of these two tasks are balanced similar to gradient normalization [3]. Note that, although we only use one more term (*i.e.*, DPVR) in Eqn. (3) compared to PVNet while using the same network architecture, it makes the training efficiency and final performance significantly different. In particular, it speeds up the convergence of training and improves estimation accuracy of vector fields without introducing any other network parameters.

3.4   DPVR as a regularizer

Since our proposed DPVR is designed to obtain generated hypothesis with small deviations, it does not directly force the estimated vector-fields to be close to the ground-truth ones. Therefore, we remove the regression loss $\mathcal{L}_{vf}$ in Eqn. 3 and examine whether DPVR can be used as a standalone loss. We found that by only using DPVR to estimate vector-fields, our network fail to converge. Note that, a direction vector and its reversed one will result in the same loss. Without employing the vector-field regression, our network will suffer this ambiguity and fail to produce consistent vector-fields when minimizing the DPVR. In other words, DPVR cannot be used as a standalone loss and it requires the vector-field regression loss to reduce the ambiguity. Therefore, our proposed DPVR is employed as a regularization term for accurate vector-field estimation.

3.5   Implementation details

For a fair comparison with our baseline method PVNet, we select 8 keypoints for each object by running the farthest point sampling algorithm on its 3D model. Following [21], we render 10,000 images and synthesize 10,000 images by "Cut and Paste" for each object. Data augmentation, including random cropping, resizing, rotation and color jittering, is employed to prevent overfitting.

In training, we set the batch size to 16 and employ Adam optimizer with an initial learning rate $1e^{-3}$. The learning rate is decayed to $1e^{-5}$ by a factor of 0.85 every 5 epochs. Our maximum training epoch is set to 100, whereas PVNet requires 200 epochs. In testing, Hough voting is employed to localize keypoints and then EPnP [15] is used to solve 6DOF poses. Similar to PVNet, our method also runs in real-time on a test image of 480×640 pixels on a GTX 2080 Ti GPU.

4   Experiments

We conduct experiments on two popular widely used datasets that are designed to evaluate 6DOF pose estimation methods.

Table 1: Pose estimation performance on the *LINEMOD* dataset w.r.t. ADD(-S) scores. Glue and egg box are considered as symmetric.

Methods	BB8	SSD6D	YOLO6D	DPOD	Pix2Pose	CDPN	PoseCNN	PVNet	Ours
Ape	40.4	65	21.62	53.28	58.1	64.38	27.8	43.62	**69.05**
Bench vise	91.8	80	81.80	95.34	91.0	97.77	68.9	99.90	**100.0**
Cam	55.7	78	36.57	90.36	60.9	91.67	47.5	86.86	**94.12**
Can	64.1	86	68.80	94.10	84.4	95.87	71.4	95.47	**98.52**
Cat	62.6	70	41.82	60.38	65.0	**83.83**	56.7	79.34	83.13
Driller	74.7	73	63.51	97.72	73.6	96.23	65.4	96.43	**99.01**
Duck	44.3	66	27.23	66.01	43.8	**66.76**	42.8	52.58	63.47
Egg box	57.8	**100**	69.58	99.72	96.8	99.72	98.3	99.15	**100.0**
Glue	41.2	**100**	80.02	93.83	79.4	99.61	95.6	95.66	97.97
Hole puncher	67.2	49	42.63	65.83	74.8	85.82	50.9	81.92	**88.20**
Iron	84.7	78	74.97	99.80	83.4	97.85	65.6	98.88	**99.90**
Lamp	76.5	73	71.11	88.11	82.0	97.86	70.3	99.33	**99.81**
Phone	54.0	79	47.74	74.24	45.0	90.75	54.6	92.41	**96.35**
Mean	62.7	79	55.95	82.98	72.4	89.86	62.7	86.27	**91.50**

Figure 3: Visualization of our qualitative results on the *LINEMOD* dataset. Green 3D bounding-boxes indicate the ground-truth poses and the blue ones represent our prediction.

4.1 Datasets

LINEMOD [10] is a *de facto* 6DOF pose estimation benchmark consisting of 15783 images for 13 objects (around 1,200 instances for each object). CAD models of those objects are also provided. LINEMOD exhibits many challenges including cluttered scenes, textureless objects, and lighting variations.

Occlusion LINEMOD [2] provides additional annotations on a subset of LINEMOD images. Each image contains multiple annotated objects and those objects are severely occluded due to the scene clutter, posing extreme challenges for pose estimation.

4.2 Evaluation metrics

We evaluate the performance of 6DOF estimation by using two commonly used metrics: **2D projection error** and **ADD score** [10]. 2D projection error measures the mean distance between the projections of a 3D model using the ground-truth and estimated poses. If the mean distance is less than 5 pixels, an estimated pose is regarded as correct. Thus, 2D projection error indicates the closeness between an object in the image and the projection of its 3D model using the estimated pose. ADD score measures average 3D distance between 3D model points transformed by the ground-truth pose and the estimated one. When the distance is less than 10 percent of the diameter of the 3D model, the estimated pose is considered as correct. For a symmetric object, ADD(-S) is employed, where the 3D distance is calculated based on the closest point distance.

Table 2: Pose estimation performance on the *LINEMOD* dataset w.r.t. 2D projection errors.

Methods	BB8	YOLO6D	CDPN	PoseCNN	PVNet	Ours
Ape	96.6	92.10	96.86	83.0	99.23	**99.04**
Bench vise	90.1	95.06	98.35	50.0	**99.81**	99.71
Cam	86.0	93.24	98.73	71.9	99.21	**99.41**
Can	91.2	97.44	99.41	69.8	99.90	**100.0**
Cat	98.8	97.41	**99.80**	92.0	99.30	99.70
Driller	80.9	79.41	95.34	43.6	96.92	**98.12**
Duck	92.2	94.65	98.59	91.8	98.02	**99.06**
Egg box	91.0	90.33	98.97	91.1	99.34	**99.43**
Glue	92.3	96.53	99.23	88.0	98.45	**99.51**
Holepuncher	95.3	92.86	99.71	82.1	**100.0**	**100.0**
Iron	84.8	82.94	97.24	41.8	99.18	**99.69**
Lamp	75.8	76.87	95.49	48.4	98.27	**99.14**
Phone	85.3	86.07	97.64	58.8	**99.42**	**99.42**
Mean	89.3	90.37	98.10	70.2	99.00	**99.40**

Table 3: Impact of different λ on the *LINEMOD* dataset.

Mean Value	0	$1e^{-4}$	$1e^{-3}$	$1e^{-2}$
ADD(-S)	86.27	87.63	91.50	90.92
Proj. err	99.00	99.24	99.40	99.36

Table 4: Pose estimation performance on the *Occlusion LINEMOD* dataset w.r.t. ADD(-S) scores.

Methods	YOLO6D	PoseCNN	Oberweger	Pix2Pose	DPOD	PVNet	Ours
Ape	2.48	9.6	17.6	**22.0**	-	15.81	19.23
Can	17.48	45.2	53.9	44.7	-	63.30	**69.76**
Cat	0.67	0.93	3.31	**22.7**	-	16.68	21.06
Driller	7.66	41.4	62.4	44.7	-	65.65	**71.58**
Duck	1.14	19.6	19.2	15.0	-	25.24	**34.27**
Egg box	-	22	25.9	25.2	-	**50.17**	47.32
Glue	10.08	38.5	39.6	32.4	-	**49.62**	39.65
Hole puncher	5.45	22.1	21.3	**49.5**	-	39.67	45.27
Mean	6.42	24.9	30.4	32.0	32.79	40.77	**43.52**

4.3   Comparisons with the state-of-the-art

Since our method focuses on providing an accurate initial pose estimation from an RGB image, we do not adopt any pose refinement methods to further improve the estimation performance. Furthermore, as mentioned in the work [16], the performance of pose refinement also highly relies on good initial pose estimation. Therefore, in this section, we compare with the state-of-the-art RGB image based pose estimation methods without using any pose refinements.

Comparisons on LINEMOD: We compare our algorithm with BB8 [22], SSD6D [13], YOLO6D [27], DPOD [33], Pix2Pose [14], CDPN [34], PoseCNN [31], and PVNet [21] on ADD(-S) scores (in Table 1) and 2D projection errors (in Table 2). BB8, SSD6D and YOLO6D localize keypoints by regression, while DPOD, Pix2Pose and CDPN regress 3D object coordinates. PoseCNN firstly localizes object centers via Hough voting and then estimates poses by regression. PVNet and our method employ voting mechanisms to localize all the keypoints for pose estimation.

Note that, BB8 and SS6D require necessary refinements to obtain satisfying results. Thus, we report their results with pose refinements. Since some methods do not report their 2D projection errors, we do not include those in Table 2. As seen in Table 1 and Table 2, our method outperforms the state-of-the-art methods on both of the metrics. Figure 3 illustrates our qualitative results (more qualitative results are shown in the supplementary material).

More importantly, our method achieves better performance on all the objects in terms of ADD(-S) scores compared to our baseline method PVNet, and improves the performance by a large margin of 5.23%. Our method also outperforms PVNet in terms of 2D projection errors except one class. As indicated in Table 1, it is difficult to estimate accurate vector-

Figure 4: Our visual results on the *Occlusion LINEMOD* dataset. Green 3D bounding-boxes indicate the ground-truth poses and blue ones are our predictions.
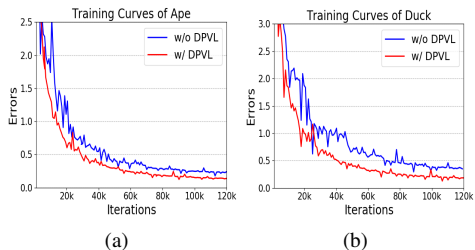

(a)

(b)

Figure 5: Comparisons of the training curves of two objects with and without using our DPVR. The errors are measured by $\mathcal{L}_{pv}$.

fields from textureless objects, such as "ape" and "duck", by PVNet. On the contrary, our method is able to estimate vector-fields more accurately and thus obtains better pose estimation performance than PVNet. In particular, our method improves the accuracy by **25.43%** and **10.89%** on "ape" and "duck" respectively in terms of ADD(-S) scores.

Comparisons on Occlusion LINEMOD: Following previous methods (*i.e.*, YOLO6D, PoseCNN, Oberweger[20], Pix2Pose, DPOD and PVNet), we directly apply the models trained on the LINEMOD dataset to the Occlusion LINEMOD dataset. Note that, we do not re-train our network on occluded data. Thus, for fair comparisons, we do not compare with Hybrid-pose [25] since it is trained on occluded data and uses pose refinements. ADD(-S) scores are often reported on this dataset and the comparisons of the state-of-the-art approaches are shown in Table 4. Our method achieves the best overall performance compared to the state-of-the-art. In Fig 1(b), we demonstrate the estimate results of all the objects (except "Bench vise") from one image. Figure 4 demonstrates our qualitative results on the Occlusion LINEMOD dataset (more visual results are shown in the supplementary material). As expected, our method outperforms PVNet, the second best, thus demonstrating the effectiveness of our proposed DPVR. Note that, due to the heavy occlusion, inaccurate segmentation (*i.e.*, object localization) is an important reason accounting for erroneous pose estimation. Improving the segmentation performance is thus left for our future work.

4.4 Ablation study

By incorporating our proposed DPVR, our method is able to attain more accurate pose estimation and improve the training convergence. Since our network architecture and training data (the synthetic and rendered images are generated by the code provided by the authors of PVNet) are as the same as PVNet, the significant performance improvements mainly benefit from our proposed regularizer, as indicated in Table 1, Table 2 and Table 4.

We also conduct an ablation study on the weight λ. Here, we set the weight λ to $0, 1e^{-2}, 1e^{-3}$ and $1e^{-4}$ to demonstrate how DPVR affects the network performance in Table 3. Note that, when λ is 0, the training objective of the network is identical to PVNet. When λ is large, *i.e.*, $1e^{-2}$, our DPVR $\mathcal{L}_{pv}$ may overwhelm the vector field regression term $\mathcal{L}_{vf}$ and lead to inferior vector field estimation. However, using too small λ, *i.e.*, $1e^{-4}$, the impact of our DPVR is still marginal. Therefore, we set λ to $1e^{-3}$.

In Fig. 5, we demonstrate that by using our DPVR, our method obtains lower distance errors between keypoints and proxy hypotheses. This indicates that our method converges

faster with the help of DPVR compared to the original PVNet. Therefore, our network achieves convergence within 100 while PVNet requires 200 epochs. Figure. 5 also implies that the distribution of hypotheses produced by our method is more concentrated since the mean distance between proxy hypotheses and keypoints is smaller. This enables our method to reach consensus more easily when voting keypoints.

5   Limitations

Since DPVR focuses on addressing the keypoint estimation errors of PVNet, we adopt the same network architecture as PVNet. Resnet-18 is employed as the backbone and thus rather weak for semantic segmentation. As mentioned in Sec. 4.3, we observed that failure cases mainly come from incorrect segmentation masks. For example, our method may fail to segment the Glue when only its sides (a very thin object) are visible or partially occluded. Moreover, the network is trained on non-occluded objects. This also imposes difficulty on segmenting occluded ones. We noticed that using a deeper backbone can significantly mitigate this problem.

6   Conclusion

In this paper, we proposed a novel differentiable proxy voting regularizer (DPVR) to achieve accurate vector-field estimation by mimicking the hypothesis voting procedure in the testing phase. Our DPVR takes the distances between pixels and the keypoints into account and enforces pixels away from a keypoint to be intolerant to inaccurate direction vector estimation. In this manner, our method achieves more accurate vector fields, thus leading to better pose estimation. Moreover, DPVR is able to speed up the convergence of our network in training. Thus, our method requires fewer iterations in training but achieves better testing performance compared to our baseline method PVNet. Extensive experiments on two standard pose estimation datasets demonstrate the superiority of our proposed method.

7   Acknowledgment

References

[1] 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] Eric Brachmann, Alexander Krull, Frank Michel, Stefan Gumhold, Jamie Shotton, and Carsten Rother. Learning 6d object pose estimation using 3d object coordinates. In *European Conference on Computer Vision (ECCV)*, pages 536–551. Springer, 2014.

[3] Zhao Chen, Vijay Badrinarayanan, Chen-Yu Lee, and Andrew Rabinovich. Gradnorm: Gradient normalization for adaptive loss balancing in deep multitask networks. *arXiv preprint arXiv:1711.02257*, 2017.

[4] Martin A. Fischler and Robert C. Bolles. Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. *Commun. ACM*, 1981.

[5] Ross Girshick. Fast r-cnn. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 1440–1448, 2015.

[6] Chunhui Gu and Xiaofeng Ren. Discriminative mixture-of-templates for viewpoint classification. In *European Conference on Computer Vision (ECCV)*, pages 408–421. Springer, 2010.

[7] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016.

[8] Stefan Hinterstoisser, Cedric Cagniart, Slobodan Ilic, Peter Sturm, Nassir Navab, Pascal Fua, and Vincent Lepetit. Gradient response maps for real-time detection of textureless objects. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34 (5):876–888, 2011.

[9] Stefan Hinterstoisser, Stefan Holzer, Cedric Cagniart, Slobodan Ilic, Kurt Konolige, Nassir Navab, and Vincent Lepetit. Multimodal templates for real-time detection of texture-less objects in heavily cluttered scenes. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 858–865. IEEE, 2011.

[10] 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 (ACCV)*, pages 548–562. Springer, 2012.

[11] Yinlin Hu, Pascal Fua, Wei Wang, and Mathieu Salzmann. Single-stage 6d object pose estimation. *arXiv preprint arXiv:1911.08324*, 2019.

[12] Yinlin Hu, Joachim Hugonot, Pascal Fua, and Mathieu Salzmann. Segmentation-driven 6d object pose estimation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3385–3394, 2019.

[13] 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 (ICCV)*, pages 1521–1529, 2017.

[14] Park Kiru, Patten Timothy, and Vincze Markus. Pix2pose: Pixel-wise coordinate regression of objects for 6d pose estimation. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2019.

[15] Vincent Lepetit, Francesc Moreno-Noguer, and Pascal Fua. Epnp: An accurate o (n) solution to the pnp problem. *International Journal of Computer Vision*, 81(2):155, 2009.

[16] Yi Li, Gu Wang, Xiangyang Ji, Yu Xiang, and Dieter Fox. Deepim: Deep iterative matching for 6d pose estimation. In *European Conference on Computer Vision (ECCV)*, pages 683–698, 2018.

[17] Ming-Yu Liu, Oncel Tuzel, Ashok Veeraraghavan, and Rama Chellappa. Fast directional chamfer matching. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1696–1703. IEEE, 2010.

[18] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C Berg. Ssd: Single shot multibox detector. In *European conference on computer vision (ECCV)*, pages 21–37. Springer, 2016.

[19] David G Lowe. Distinctive image features from scale-invariant keypoints. *International journal of computer vision*, 60(2):91–110, 2004.

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

[21] 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 Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4561–4570, 2019.

[22] 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 (CVPR)*, pages 3828–3836, 2017.

[23] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in Neural Information Processing Systems*, pages 91–99, 2015.

[24] Reyes Rios-Cabrera and Tinne Tuytelaars. Discriminatively trained templates for 3d object detection: A real time scalable approach. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 2048–2055, 2013.

[25] Chen Song, Jiaru Song, and Qixing Huang. Hybridpose: 6d object pose estimation under hybrid representations. *arXiv preprint arXiv:2001.01869*, 2020.

[26] Hao Su, Charles R Qi, Yangyan Li, and Leonidas J Guibas. Render for cnn: Viewpoint estimation in images using cnns trained with rendered 3d model views. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 2686–2694, 2015.

[27] Bugra Tekin, Sudipta N Sinha, and Pascal Fua. Real-time seamless single shot 6d object pose prediction. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 292–301, 2018.

[28] Yurun Tian, Xin Yu, Bin Fan, Fuchao Wu, Huub Heijnen, and Vassileios Balntas. Sosnet: Second order similarity regularization for local descriptor learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 11016–11025, 2019.

[29] Shubham Tulsiani and Jitendra Malik. Viewpoints and keypoints. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1510–1519, 2015.

[30] He Wang, Srinath Sridhar, Jingwei Huang, Julien Valentin, Shuran Song, and Leonidas J Guibas. Normalized object coordinate space for category-level 6d object pose and size estimation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2642–2651, 2019.

[31] Yu Xiang, Tanner Schmidt, Venkatraman Narayanan, and Dieter Fox. Posecnn: A convolutional neural network for 6d object pose estimation in cluttered scenes. *Robotics: Science and Systems*, 2017.

[32] Xin Yu, Yurun Tian, Fatih Porikli, Richard Hartley, Hongdong Li, Huub Heijnen, and Vassileios Balntas. Unsupervised extraction of local image descriptors via relative distance ranking loss. In *Proceedings of the IEEE International Conference on Computer Vision Workshops*, pages 0–0, 2019.

[33] Sergey Zakharov, Ivan Shugurov, and Slobodan Ilic. Dpod: 6d pose object detector and refiner. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2019.

[34] Li Zhigang, Wang Gu, and Ji Xiangyang. Cdpn: Coordinates-based disentangled pose network for real-time rgb-based 6-dof object pose estimation. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2019.

[35] Menglong Zhu, Konstantinos G Derpanis, Yinfei Yang, Samarth Brahmbhatt, Mabel Zhang, Cody Phillips, Matthieu Lecce, and Kostas Daniilidis. Single image 3d object detection and pose estimation for grasping. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 3936–3943. IEEE, 2014.