PrimA6D: Rotational Primitive Reconstruction for Enhanced and Robust 6D Pose Estimation

Myung-Hwan Jeon[1] and Ayoung Kim[2*]

Abstract— In this paper, we introduce a rotational primitive prediction based 6D object pose estimation using a single image as an input. We solve for the 6D object pose of a known object relative to the camera using a single image with occlusion. Many recent state-of-the-art (SOTA) two-step approaches have exploited image keypoints extraction followed by PnP regression for pose estimation. Instead of relying on bounding box or keypoints on the object, we propose to learn orientation-induced primitive so as to achieve the pose estimation accuracy regardless of the object size. We leverage a Variational AutoEncoder (VAE) to learn this underlying primitive and its associated keypoints. The keypoints inferred from the reconstructed primitive image are then used to regress the rotation using PnP. Lastly, we compute the translation in a separate localization module to complete the entire 6D pose estimation. When evaluated over public datasets, the proposed method yields a notable improvement over the LINEMOD, the Occlusion LINEMOD, and the YCB-Video dataset. We further provide a synthetic-only trained case presenting comparable performance to the existing methods which require real images in the training phase.

I. INTRODUCTION

The 6D object pose estimation intends to recover the orientation and translation of the object relative to the camera. When solving 6D pose estimation, reaching accurate, fast and robust solutions has been a key element of augmented reality [1], robotic tasks [2, 3, 4], and autonomous vehicles [5]. Despite the widespread use of RGB-D cameras, single image-based pose estimation is still essential when an RGB-D camera is not applicable.

Recent advances in deep learning-based object pose estimation has been groundbreaking. Direct end-to-end 6D pose regression suffered from the non-linearity of the rotation, and studies tackled this issue through orientation representation [6] or combining Perspective-n-Point (PnP) and random sample consensus (RANSAC). The latter approaches reported current state-of-the-art (SOTA) performance by leveraging a two-step approach for pose estimation [1, 7, 8]. These approaches extract keypoints from the RGB image using deep learning and compute the 6D pose of the object using 2D-3D correspondence matching, such as the PnP algorithm.

We were also partially motivated by the finding in [9] that an Augmented Auto Encoder (AAE) could implicitly represent a rotation when training an AAE from a 3D CAD. The main advantage of this approach is that there is no need for manual annotation to prepare ground pose labeled

[1]M. Jeon is with Department of the Robotics Program KAIST, Daejeon, S. Korea myunghwan.jeon@kaist.ac.kr
[2]A. Kim is with the Department of Civil and Environmental Engineering, KAIST, Daejeon, S. Korea ayoungk@kaist.ac.kr
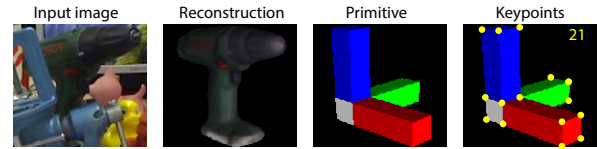
Fig. 1: We train VAE to reconstruct objects and the associated primitive followed by the keypoints extraction over the inferred primitive. Keypoints are extracted from each corner plus the center, the same as the object center (i.e., 21).

training images. While [9] only solved for the orientation retrieval given a discretized orientation set, we reconstruct a primitive from which keypoints are extracted under occlusion and complete a full orientation regression.

This paper solves for the direct 6D pose regression by introducing a novel primitive descriptor in Fig. 1. The proposed solution, primitive associated 6D (PrimA6D) pose estimation, combines a direct and holistic understanding of an image with keypoint based PnP for the final orientation regression. More specifically, we newly introduce the primitive decoder in addition to the VAE to increase the discriminability of the orientation inference. Via this step, like the aforementioned studies, we aim to find reliable keypoints in an image for pose regression extracting from the reconstructed primitives. Differing from previous methods, our method presents the following contributions.

- We propose a novel 6D object pose estimation network called PrimA6D, which introduces a rotation primitive reconstruction and its associated keypoints to enhance the orientation inference.
- The proposed estimation scheme mitigates the manual effort of preparing label annotated images for training. The proposed primitive is straightforward to generate given rotation in 3D CAD; the trained VAE allows us to learn from synthetic images via domain randomization robust to occlusion and symmetry.
- We verify the meaningful performance improvement of 6D object pose estimation compared to the existing SOTA 6D pose regression methods. The proposed method improved estimation over LINEMOD as 97.62% (PVNet 86.27%), Occlusion LINEMOD as 59.77% (PVNet 40.77%), and YCB-Video as 94.43% (PVNet 73.78%) in terms of ADD(-S).
- The proposed method is more generalizable than existing methods showing less performance variance between datasets. This superior generalization capability is essential for robotic manipulation and automation tasks.
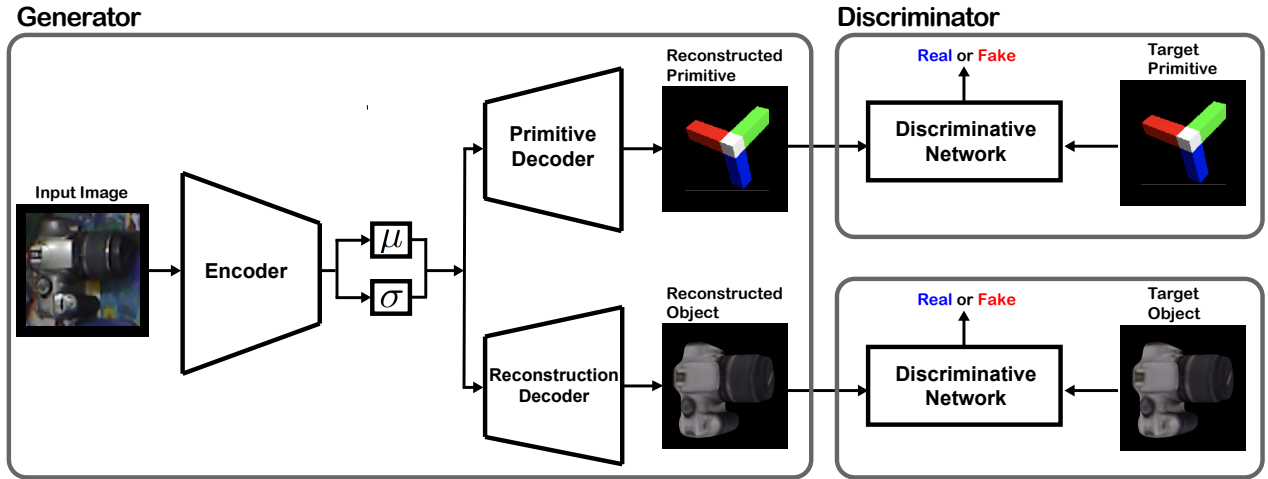
Fig. 2: The reconstruction stage of PrimA6D. In the reconstruction stage, an encoder using ResNext50 [10] and an reconstruction decoder consisted of fully deconvolutional layers form the VAE. We reconstruct both object and primitive in a separate decoder. To increase the discriminability, we additionally introduce adversarial loss and train using GAN.

II. RELATED WORKS

PnP based approaches establish 2D-3D correspondences from keypoints or patches and solve for the pose from these correspondences. Keypoint-based approaches define specific features to infer the pose from an image, such as the corner points of 3D bounding-box. These approaches chiefly involve two steps: first, the 2D keypoint of an object in the image is computed, and then an object pose is regressed using the PnP algorithm. In BB8 [11], the authors predicted the 2D projection corner points of the object's 3D bounding box. Then, they computed the object pose from 2D-3D correspondences using PnP algorithm. Similarly, [12] additionally added the central point of object to the keypoints of BB8 [11]. Unfortunately, [11, 12] revealed the weakness to the occlusion that prevents defining the correct bounding box of an object. To alleviate this issue, PVNet [1] predicted the unit vectors that point to predefined keypoints for each pixel in an object. By computing the locations of these keypoints using unit vector voting, the authors substantially improved their accuracy. However, these keypoint-based approaches may suffer when an object lacks texture, whereas the proposed primitive can be robustly reconstructed even for small and textureless objects.

The part-based approaches extract parts of an object called patches for pose regression. This method is frequently used with depth data. For instance, [13] jointly predicted the object labels and object coordinates for every pixel, then computed the object pose from the predicted 2D-3D correspondence matching. In [14], the authors extracted heatmaps of multiple keypoints from local patches then computed the object pose using the PnP algorithm. Depth data may not always be available and we propose using a single RGB as input. Recently, [15] exploited Auto Encoder (AE) to generate an image representing 3D coordinate per each pixel. Together with the reconstructed image containing 3D information, the expected error per pixel was also estimated to sort meaningful 2D-3D correspondence.

Unlike the aforementioned approaches, which consist of steps, holistic approaches utilize the overall shape of the object appearing in an image to directly regress the object pose relative to the camera. In Deep-6DPose [16], the authors found the regions where the objects were located through the Region Proposal Network (RPN). Despite solving for both object detection and 6D pose estimation, their method was vulnerable to small and symmetrical objects since they relied on the detected regions. In SSD-6D [17], the authors used the discrete 6D pose space to represent the discriminable viewpoint. SSD-6D is a Single Shot Multibox Detector (SSD)-style pose estimation model that extends the work of [18]. Their method had a faster inference time than other methods though it sacrificed accuracy.

Incorporating a mask has been widely adopted. For example, [2, 19, 3, 20] utilized an mask of the object to solve the pose estimation task. The well-known PoseCNN [19] required an Iterated Closest Point (ICP) refinement for better accuracy. Stemming from PoseCNN, [20] enabled a real-time performance with RGB data. The authors of [19, 20] also reported meaningful results, but their works were sensitive to the issue of object occlusion. Tackling the occlusion issue, SilhoNet [2] constructed an occlusion-free mask even from the occluded region, through CNN. This type of mask level restoration enhanced the performance over occlusion, however, yielded low accuracy for symmetrical objects. Unlike these methods relying on masks, we exploit the reconstructed object to not only regress the translation but further tackle the occlusion issue.

All the abovementioned methods require true pose-labeled images for training. Another stream of studies has focused on leveraging latent code without the need for labeled training images via AAE to solve this issue. Without requiring the pose label from object 6D pose estimation, [9] utilized the latent code generated from the object recovery process using AE. The object pose was determined by a similarity comparison with the predefined latent codebook for each
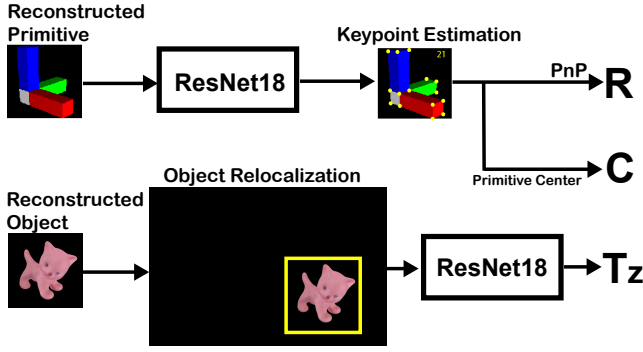
Fig. 3: The rotation and translation inference. We use keypoints learned from the reconstructed primitive for the orientation estimation using PnP. In the translation inference stage, the network input is created using the output of the reconstruction decoder through object relocalization.

pose, but the accuracy was dominated by the size of the codebook with a discretized orientation class. We have overcome this accuracy limitation and achieved substantially improved performance from a synthetic dataset.

III. METHOD

Given an RGB image, we compute the 6D pose of an object relative to the camera in terms of the its relative transform $\mathbf{T}_O^C$ between the camera coordinate frame C and the object coordinate frame O. The 6D pose is represented by quaternion and translation vectors. Similar to [15] and [9], the proposed method can be accompanied with other detection modules such as M2Det [21]. We use a cropped image using a bounding box detected from other methods as input to our network.

A. Training Set Prepration

As indicated earlier, our aim is at enabling 6D pose regression without requiring the real images. For each pair of a 3D CAD object and the associated camera pose, we prepare the training set consists of a bounding box, rendered primitive, the target keypoints (the 20 corner points and a center point), and rendered training image. The primitive images represent the rotation of the object having different colors per axis as in Fig. 1.

B. Object and Primitive Reconstruction

Using the training data, the first phase of the proposed method consists of object and primitive reconstruction as shown in Fig. 2. We train the VAE having an encoder and two reconstruction decoders. This step is partially similar to AAE in [9]. In [9], the authors verified that a trained AAE using a geometric augmentation technique learns the representation of the object's orientation and generates the latent code with data for the object orientation. We adopt the same technique in the training phase by rendering the training images with various poses of the object. The training images are augmented using the domain randomization technique [22] to bridge the gap between real and rendered images.

Augmented images are fed into the encoder, and clean target images are obtained through the reconstruction decoder using four losses as follows. Differing from [9], our training strategy is to use GAN by introducing a discriminator for the reconstructed object and primitive.

1) Object Reconstruction Loss: The first loss focuses on the reconstruction of an object by comparing the pixel-wise loss. To prevent overfitting in the reconstruction stage, we use the top-k pixel-wise L2 loss as follow.

$$\mathcal{L}_O = \frac{1}{K} \sum_{i=1}^{K} \ell_{[i]}(\|x - \hat{x}\|^2), \tag{1}$$

where $\ell_{[i]}$ is a function that extracts the pixel with the ith largest error [23]. We use $K = 128$ in this paper. In (1), x is the target image pixel and $\hat{x}$ is the predicted image pixel respectively. This initial latent code encompasses the orientation but it is not sufficiently discriminative for regression. To increase the discriminability, we introduce the rotational primitive decoder to embolden the orientation of the code.

2) Primitive Reconstruction Loss: The rendered primitive image becomes the prediction target of the primitive decoder. Using the color-coded axes in primitive image, we designed the color based axis-aligning loss function for the primitive decoder. When computing the loss for the channels, we consider a per-pixel intensity difference d_c for each channels $c \in \mathcal{C} = [R, G, B]$.

$$d_c = x_c - \hat{x}_c \tag{2}$$

$$S_c = \frac{1}{K} \sum_{i=1}^{K} \ell_{[i]}(\exp(\alpha |d_c|) \cdot \|d_c\|^2) \tag{3}$$

$$C_c = \frac{1}{K} \sum_{i=1}^{K} \ell_{[i]}(\|d_c\|^2) \tag{4}$$

$$\mathcal{L}_P = \sum_{c \in \mathcal{C}} S_c \exp\left(\frac{C_c}{\sum_{k \in \mathcal{C}} C_k}\right) \tag{5}$$

Here, the S_c measures misalignment of each axis in terms of the pixel intensity difference per channel. The C_c measures the overall misalignment per channel and discerns between channels by penalizing the channel showing a larger error when the primitive reconstruction loss L_P is constructed. The α is a weight constant ($\alpha = 5$ in this paper).

3) Overall VAE Loss: Using these two reconstruction losses with Kullback-Leibler (KL) divergence loss, we train the VAE to generate both an object image and a primitive image. By adopting VAE, the network encodes input image x to the parameters of a Gaussian distribution $q(z|x)$. In doing so, we minimize the KL divergence between $q(z|x)$ and $\mathcal{N}(0, I)$. The overall loss function for the reconstruction stack becomes as below.

$$\mathcal{L}_R = \mathcal{L}_O + \mathcal{L}_P + D_{KL}(q(z|x)\|\mathcal{N}(0, I)) \tag{6}$$

4) Adversarial Loss: We further aim to improve the quality of the reconstruction using training strategy of GAN. For instance, the reconstructed primitive sometimes becomes
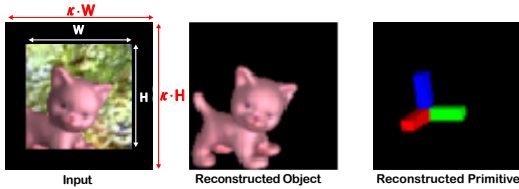
Fig. 4: The input image of the reconstruction network is (κ=1.3) larger than the actual bounding-box allowing complete object reconstruction even for inaccurate bounding-box.

ambiguous as shown in Fig. 5 and may result in incorrect keypoints learning. Regarding this issue, introducing adversarial loss improved the primitive/object prediction accuracy and thereby the overall pose inference performance. For the target primitive/object image X and the reconstructed image $\hat{X}$, we consider the following loss to train using GAN [24].

$$\mathcal{L}_A = \mathbb{E}_{\hat{X}}[\log D(\hat{X})] + \mathbb{E}_X[\log(1 - D(G(X)))]. \quad (7)$$

Here we use $D(\cdot)$ to distinguish if the predicted primitive/object is the target primitive/object from 3D model (true) or generated by $G(\cdot)$ (fake). Following the training strategy of GAN, we solve for G and D by minimizing $\mathcal{L}_A^G = \mathbb{E}_{\hat{X}}[\log D(\hat{X})]$ and maximizing $\mathcal{L}_A^D = \mathbb{E}_X[\log(1 - D(G(X)))]$.

C. Orientation Regression via PnP

From the reconstructed primitive we estimate the rotation first by learning the keypoints from it. Using generated keypoints in the training set preparation and via pixel-wise L2 loss, we adopt ResNet18 [25] and extract the keypoints of the reconstructed primitive using the loss as

$$\mathcal{L}_{rot} = \sum \|x_k - \hat{x_k}\|^2. \quad (8)$$

Since the keypoints consists of corners and the center point of the object which is the same as the primitive center, we utilize this estimated center point, $C = x_c = (u_c, v_c)$ for the translation estimation.

D. Translation Regression

Given the estimated orientation, we then infer the 3D translation by computing the scene depth T_z of the object center (i.e., primitive center). The reconstructed image is relocated to the original image using the bounding box of the object. This process is called *object relocalization* (Fig. 3).

As the sample illustration in Fig. 4 shows, the reconstruction occurs *without* assuming the bounding box is centered and thereby is robust to inaccurate object detection. The reconstruction phase handles the inaccurate bounding box location by properly reconstructing the object even at the off-centered position. The inflation scaling factor κ was chosen to handle the detected bounding box with Intersection over Union (IoU) 0.75. A small κ fails to handle inaccuracy in objection detection phase, and a large κ might deteriorate the performance with increased empty space near borders. Because this re-localization replaces an object with a reconstructed object, the translation can be regressed over occlusion-free objects. From this relocalized full image, we

again use ResNet18 to regress the scene depth T_z to the object center.

Using T_z, the bounding-box, and the camera intrinsic matrix K, we compute the T_x and T_y components of translation as follows from the estimated object center (u_c, v_c). The focal length f and principal point (u_p, v_p) are obtained from the calibration matrix while the bounding box is known when preparing the training data.

$$u_c = f_u \frac{T_x}{T_z} + u_p \quad \text{and} \quad u_v = f_v \frac{T_y}{T_z} + v_p. \quad (9)$$

IV. EXPERIMENT

In this section, we evaluate the proposed method using three different datasets.

A. Dataset

In total, three datasets were used for the evaluation: LINEMOD [28], Occlusion LINEMOD [28], and YCB-Video [19]. The LINEMOD and Occlusion LINEMOD datasets are widely exploited benchmarks for 6D object pose estimation and cover various challenges, including occlusion and texture-less. The LINEMOD dataset comprises 13 objects with 1,312 rendered images on each for the training and about 1,200 images per object for the test. The Occlusion-LINEMOD dataset shares the training images with the LINEMOD dataset. For the test, they provide pose information for eight occluded objects. A recently released dataset, the YCB-Video dataset is composed of 21 high-quality 3D models, offering 92 annotated video sequences per frame. These video sequences include various lighting conditions, noise in the capture, and occlusion.

B. Training Detail

We sampled 50,000 object poses and rendered 50,000 training images per object together with their associated primitives. The rendered object images were further augmented using the method proposed by [22] using domain randomization. We train our network mainly from synthetic images and additionally train with 180 real images for three types of dataset.

All of the networks were trained with the Adam optimizer on Titan V. The ResNet backbone was initialized with pretrained weights provided in PyTorch. All networks were trained with a batch size of 50 and a 0.0001 learning rate for 40 epochs.

C. Evaluation Metric

(i) 2D Projection error metric To evaluate the pose estimation in terms of the 2D projection error we also use the same metric as in [1] and measure the mean pixel distance between projection of the 3D model and the image pixel points. We also count the estimation is correct if the error is less than 5 pixels.

(ii) 3D Projection error metric Similar to [19], two metrics were selected for evaluation. We chose the average distance metric (ADD) [28] as the evaluation metric, which computes the distance of transformed 3D model points

methods	Holistic Approach		PnP based Approach									
			w/o refinement							w/ refinement		
	PoseCNN [19]	Deep-6D Pose [16]	BB8 [11]	Tekin [12]	Pix2Pose [15]	DPOD [26]	PVNet [1]	PrimA6D-S	PrimA6D-SR	BB8	Tekin	HybridPose [27]
ape†	-	38.8	27.9	21.62	58.1	53.28	43.62	66.5	**90.21**	40.4	65	77.6
benchvise	-	71.2	62	81.8	91	95.34	**99.9**	97.11	99.75	91.8	80	99.6
cam	-	52.5	40.1	36.57	60.9	90.36	86.86	91.5	**98**	55.7	78	95.9
can	-	86.1	48.1	68.8	84.4	94.10	95.47	93.89	**98.74**	64.1	86	93.6
cat†	-	66.2	45.2	41.82	65	60.38	79.34	88.21	**99.06**	62.6	70	93.5
driller	-	82.3	58.6	63.51	76.3	97.72	96.43	98.14	**99.57**	74.4	73	97.2
duck†	-	32.5	32.8	27.23	43.8	66.01	52.58	72.56	**90.82**	44.3	66	87.0
eggbox*	-	79.4	40	69.58	96.8	99.72	99.15	91.93	99.52	57.8	**100**	99.6
glue*	-	63.7	27	80.02	79.4	93.83	95.66	91.14	99.75	41.2	**100**	98.7
holepuncher†	-	56.4	42.4	42.63	74.8	65.83	81.92	68.71	**95.31**	67.2	49	92.5
iron	-	65.1	67	74.97	83.4	**99.80**	98.88	97.3	99.56	84.7	78	98.1
lamp	-	89.4	39.9	71.11	82	88.11	**99.33**	95.59	99.18	76.5	73	96.9
phone	-	65	35.2	47.74	45	74.24	92.41	89.38	**99.59**	54	79	98.3
average	55.95	65.2	43.55	55.95	72.38	82.98	86.27	87.84	**97.62**	62.67	76.69	94.5

TABLE I: The accuracy in terms of the ADD(-S) metric for our method and baseline methods on the LINEMOD dataset. Symmetrical objects are marked with an asterisk (*) and small objects are marked with †. PoseCNN [19] only provided the average.

method	w/o refinement					w/ refinement
	BB8	Tekin	PVNet	PrimA6D-S	PrimA6D-SR	BB8
ape†	95.3	92.1	**99.23**	97.24	99.02	96.6
benchvise	80	95.06	**99.81**	95.3	98.92	90.1
cam	80.9	93.24	99.21	97	**99.75**	86
can	84.1	97.44	**99.9**	93.64	99.24	91.2
cat†	97	97.41	99.3	95.08	**99.83**	98.8
driller	74.1	79.41	96.92	97.22	**98.9**	80.9
duck†	81.2	94.65	98.02	96.65	**98.8**	92.2
eggbox*	87.9	90.33	**99.34**	86.59	98.32	91
glue*	89	96.53	98.45	91.22	**99.26**	92.3
holepuncher†	90.5	92.86	**100**	81	97.41	95.3
iron	78.9	82.94	**99.18**	84.89	98.58	84.8
lamp	74.4	76.81	98.27	78.72	**98.77**	75.8
phone	77.6	86.07	**99.42**	93.56	99.11	85.3
average	83.92	90.37	**99.00**	91.39	98.92	89.25

method	Rotation MAE [deg]				Translation MAE [mm]			
	Tekin	PVNet	PrimA6D-S	PrimA6D-SR	Tekin	PVNet	PrimA6D-S	PrimA6D-SR
ape†	5.69	6.25	6.35	**3.63**	29.32	47.70	10.58	**5.72**
benchvise	3.14	**1.43**	3.32	1.96	18.86	6.97	6.94	**3.81**
cam	3.91	**1.75**	3.05	1.83	33.47	9.74	7.31	**4.29**
can	2.89	**1.48**	3.14	2.01	16.87	7.93	8.47	**5.55**
cat†	4.79	**1.88**	3.91	2.27	22.61	10.56	8.27	**3.35**
driller	3.42	2.42	2.30	**1.53**	19.72	16.67	4.46	**3.20**
duck†	6.39	3.00	4.87	**2.72**	28.52	15.51	8.83	**4.84**
holepuncher†	4.77	**1.99**	7.82	3.29	31.37	9.70	15.99	**4.80**
iron	6.21	2.07	3.27	**1.74**	27.87	10.95	9.41	**4.57**
lamp	4.24	**1.50**	4.11	1.62	26.78	7.90	9.64	**4.35**
phone	-	**1.82**	3.69	1.98	-	9.90	9.29	**4.46**
average	4.55	2.33	4.17	**2.23**	25.54	13.96	9.02	**4.45**

TABLE II: The accuracy in terms of the 2D projection error (left) and mean absolute error (MAE) (right) for our method and baseline methods on the LINEMOD dataset. Symmetrical objects are marked with an asterisk (*) and small objects are marked with †. Using the provided network and weight by authors, we evaluate mean absolute error. Tekin did not provide weight for the phone model. On evaluation in terms of MAE, the symmetric object is excluded because evaluation for the symmetric object requires a shape comparison rather than a specific pose comparison.

methods	w/o refinement								w/ refinement
	Tekin	PoseCNN	Oberweger[14]	Pix2Pose	PVNet	PrimA6D-S	PrimA6D-SR	w/o GAN	HybridPose
ape†	2.48	9.6	17.6	22	15.81	28.71	37.35	2.75	**53.3**
can	17.48	45.2	53.9	44.7	63.3	61.88	67.19	5.47	**86.5**
cat†	0.67	0.93	3.31	22.7	16.68	40.43	44.65	1.68	**73.4**
duck†	1.14	19.6	19.2	15	25.24	39.28	50.65	10.87	**92.8**
driller	7.66	41.4	62.4	44.7	65.65	70.42	**74.87**	9.71	62.8
eggbox*	-	22	25.9	25.2	50.17	54.31	66.01	7.02	**95.3**
glue*	10.08	38.5	39.6	32.4	49.62	71.03	73.02	7.72	**92.5**
holepuncher†	5.45	22.1	21.3	49.5	12.41	46.19	64.38	12.41	**76.7**
average	6.42	24.92	30.40	32.00	40.77	51.53	59.77	7.20	**79.2**

methods	Tekin	PoseCNN	Oberweger	PVNet	PrimA6D-S	PrimA6D-SR
ape†	7.01	34.6	**69.9**	69.14	56.58	62.64
can	11.2	15.1	82.6	**86.09**	69.26	76.8
cat†	3.62	10.4	65.1	**65.12**	54.33	58.8
duck†	5.07	31.8	61.4	61.44	64.74	**67.36**
driller	1.4	7.4	**73.8**	73.06	59.55	61.28
eggbox*	-	1.9	13.1	8.43	8.53	**16.48**
glue*	4.7	13.8	54.9	55.37	51.83	**57.38**
holepuncher†	8.26	23.1	66.4	**69.84**	50.82	67.76
average	5.89	17.26	60.90	**61.06**	51.96	58.56

TABLE III: The accuracy in terms of the ADD(-S) metric (left) and 2D projection error (right) for our method and baseline methods on the Occlusion LINEMOD dataset. All values are imported from [1] except pix2pose and hybridpose. Symmetrical objects are marked with an asterisk (*) and small objects are marked with †.

between the ground truth pose and predicted pose using ADD(-S) metric. When the average distance value is less than 10% of the 3D model's diameter, the predicted 6D pose is considered to be correct. This ADD metric, however, is not suitable to use with symmetrical objects. For symmetrical objects, the average distance is computed between the closest points.

(iii) Pose estimation metric For the original LINEMOD dataset, we directly measure the mean absolute error (MAE) for rotation and translation together with other metrics.

D. 6D Object Pose Estimation Evaluation

We evaluate the performance of our model on the three datasets. The proposed method is compared against other SOTA methods from holistic, keypoint-based and part-based approaches. We evaluate our method when training only with synthetic images (PrimA6D-S) and when training using 180 additional real images (PrimA6D-SR). Other existing methods were trained using additional real images. Qualitative results for all three datasets are presented in Fig. 7 and Fig. 8. Please also refer to `prima6d.mp4`. We excluded the comparison to the case without the primitive decoder due to drastically large error when no primitive decoder was used.

1) LINEMOD Dataset Results: Table. I and Table. II exhibit the evaluation for all 13 objects in the LINEMOD dataset. Among the keypoint or the part-based approaches, PVNet is the most recently released method with SOTA performance and thus used as the main baseline method for comparison.

Over the LINEMOD dataset, PrimA6D-SR outperformed

methods	PoseCNN	Oberweger	PVNet	PrimA6D-S	PrimA6D-SR
002_mater_chef_can	55.17	49.1	81.6	82.57	**99.71**
003_cracker_box	52.9	83.6	80.5	75.7	**99.44**
004_sugar_box	68.3	82	84.9	97.94	**99.74**
005_tomato_soup_can	66.1	**79.7**	78.2	73.51	78.39
006_mustard_bottle	80.8	91.4	88.3	94.42	**99.96**
007_tuna_fish_can†	70.6	49.2	62.2	56.77	**79.47**
008_pudding_box	62.2	90.1	85.2	52.06	**99.59**
009_gelatin_box	74.8	93.6	88.7	94.88	**99.59**
010_potted_meat_can†	59.5	79	65.1	38.72	**80.35**
011_banana	72.1	51.9	51.8	74.14	**87.01**
019_pitcher_base	53.1	69.4	91.2	71.94	**98.12**
021_bleach_cleanser	50.2	76.1	74.8	50.75	**90.04**
024_bowl*	69.8	76.9	89	96.84	**98.79**
025_mug	58.4	53.7	81.5	88.82	**89.93**
035_power_drill	55.2	82.7	83.4	92.62	**97.67**
036_wood_block*	61.8	55	71.5	95.68	**100**
037_scissors	35.3	65.9	54.8	63.89	**97.85**
040_large_marker	58.1	56.4	35.8	85.51	**94.18**
051_large_clamp*	50.1	67.5	66.3	97.72	**98.89**
052_extra_large_clamp*	46.5	53.9	53.9	88.25	**98.12**
061_foam_brick*	85.9	89	80.6	80.91	**96.15**
average	61.28	71.24	73.78	78.74	**94.43**

methods	PoseCNN	Oberweger	PVNet	PrimA6D-S	PrimA6D-SR
002_mater_chef_can	74.2	0.09	29.7	24.5	**97.24**
003_cracker_box	0.12	64.7	50.35	21.47	**93.24**
004_sugar_box	7.11	72.2	61.25	74.47	**94.11**
005_tomato_soup_can	5.21	39.8	60.69	85.31	**89.74**
006_mustard_bottle	6.44	**87.7**	82.35	74.06	87.28
007_tuna_fish_can†	2.96	38.9	45.21	97.21	**99.4**
008_pudding_box	5.14	78	52.8	69.63	**99.76**
009_gelatin_box	15.8	94.8	94.85	52.12	**97.78**
010_potted_meat_can†	23.1	41.2	62.92	69.25	**95.86**
011_banana	0.26	10.3	8.18	37.85	**87.01**
019_pitcher_base	0	5.4	**79.3**	46.54	69.64
021_bleach_cleanser	1.16	23.2	37.51	58.61	**67.11**
024_bowl*	4.43	26.1	33.99	26.59	**56.13**
025_mug	0.78	29.2	52.98	89.75	**98.3**
035_power_drill	3.31	69.5	74.74	51.78	**78.6**
036_wood_block*	0	2.1	2.06	90.54	**95.4**
037_scissors	0	12.1	56.35	60.95	**96.01**
040_large_marker	1.38	1.9	6.8	87.92	**93.94**
051_large_clamp*	0.28	24.2	44.94	59.31	**86.88**
052 extra large clamp*	0.58	1.3	7.77	21.24	**54.08**
061_foam_brick*	0	75	25	67.07	**96.84**
average	7.25	37.99	46.18	60.29	**87.35**

TABLE IV: The accuracy in terms of the ADD(-S) metric (left) and 2D projection error (right) for our method and the baseline methods on the YCB-VIDEO dataset. Symmetrical objects are marked with an asterisk (*) and small objects are marked with †.

PVNet in terms of the ADD(-S) score and presented a comparable result in terms of the 2D projection metric. The different performance can be understood from the MAE metric in Table. II. If the object center point in the image is found properly, the 2D projection metric is affected more by the rotational estimation accuracy and ADD(-S) is more critical to the translation inference performance. As can be seen, PVNet exhibited higher performance than PrimA6D-S in terms of the rotation MAE metric, which results in a performance difference on the 2D projection metric. On the contrary, PrimA6D-S outperformed PVNet in terms of the translation MAE metric, which results in a performance difference on the ADD(-S) score.

In addition, the PrimA6D(-S/-SR) presents the strength for small and texture-less objects that the drawback in the PVNet such as ape and cat. Especially, the PrimA6D(-S/-SR) has the forte on the translation estimation.

2) Occlusion LINEMOD Dataset Results: Tables in Table. III list the performance on the Occlusion LINEMOD dataset. Here, we only evaluate the performance of models that do not have the refinement step. As can be seen in two tables, the PrimA6D(-S/-SR) outperformed PVNet in terms of the ADD(-S) score and presented a comparable result in terms of the 2D projection metric. As the PrimA6D(-S/-SR) reconstruct the corresponding object and primitive, we can affirm notable improvement on the occluded case. Moreover, the PrimA6D(-S/-SR) demonstrates striking accuracy for objects that are difficult to recognize by other methods due to small size such as cat and holepuncher.

3) YCB-Video Dataset Result: We further evaluate the pose estimation performance over the YCB dataset as shown in Table. IV. The PrimA6D(-S/-SR) achieved enhanced inference capability even when trained solely from synthetic dataset (PrimA6D-S). The proposed method is capable of inferring 6D pose of objects that are difficult to recognize by other methods such as wood block.
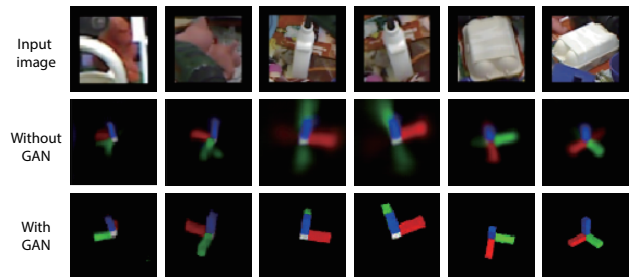
Another notable point in YCB dataset is the overall

Fig. 5: Effect of GAN on occluded and symmetrical object.

down performance from LINEMOD; all approaches including PVNET and ours shows lower accuracy in YCB than LINEMOD. This is because of the challenging and realistic dataset nature of YCB. Despite the challenging dataset, the proposed method shows the smallest performance drop from LINEMOD both in terms of ADD(-S) and 2D projection error even when solely trained from synthetic data. This reveals the generalization capability of the proposed method while others lacking.

E. Ablation Studies

We further evaluate the proposed method by conducting ablation studies in terms of the effect of real images, effect of training with GAN and bounding box accuracy.

1) Effect of real image The proposed method was designed to exploit synthetic data from CAD model. However, considering the potential discrepancy between CAD and real data, additional weight refinement using real training image shows very meaningful performance improvement. We train the network only from synthetic data (PrimA6D-S) and compare the performance improvement by adding real images in the training set (PrimA6D-SR). This comparison is provided for three datasets. Obviously we witness accuracy improvement in all metrics by adding real images in the training set.

2) Training with GAN The adversarial loss and training strategy using GAN provides a detailed refinement of the

object and primitive reconstruction. As can be seen in Fig. 5, the reconstructed result shows a blurry and obscure reconstruction when without using GAN. This indicates the potential ambiguity in the rotation and translation estimation. Additional training with GAN enables the VAE to build discriminability. Also, regarding symmetrical objects with ambiguity for one axis, incorporating adversarial training can solve the potential axis ambiguity in the reconstructed primitive and completely form three orthogonal axes. In the experiment, we confirmed that the ambiguous axis in the primitive becomes the clear orthogonal axis for the other two axes according to the qualitative result. Testing over Occlusion LINEMOD in Table. III reveals this substantial improvement from the adversarial loss.

3) Bounding box accuracy The input of the network is a cropped image from an object detection module hence the PrimA6D estimation performance may be dependent on the detection accuracy. We have examined the effect of the detected bounding box accuracy by reducing the IoU to the original bounding box in terms of ADD(-S) as this is more sensitive to the translational estimation accuracy. As can be seen in Table. V, the performance drops as we reduce the IoU from 1.0 to 0.75. As can be seen from Table. V, Table. I and Table. II, the PrimA6D-SR still outperforms all other existing methods despite the performance degradation. More notably, the PrimA6D-S shows the exceptional result on the Occlusion LINEMOD dataset even with decreased IoU, being solely trained from synthetic data.

F. Inference Time

Since our method is applied after object detection, the total inference time is also affected by the running time of the object detection model. For example, we used M2Det [21] that runs at 33.4 frames-per-second (fps). For the 6D object pose estimation of a single object in an image, the total process takes 31ms on average using a GTX 1080Ti GPU. The entire 6D pose estimation supports 16 to 17 fps.

G. Failure Cases

Fig. 6 shows failure cases when one axis is entirely obscured by the other two axes of the rotation primitive. This hidden primitive axis prevents the algorithm from extracting the correct keypoint yielding a larger rotation error. In future work, we aim to include estimated depth from the primitive to resolve this issue.

V. CONCLUSION

This paper reported on 6D pose estimation from a single RGB image by introducing novel primitive learning asso-

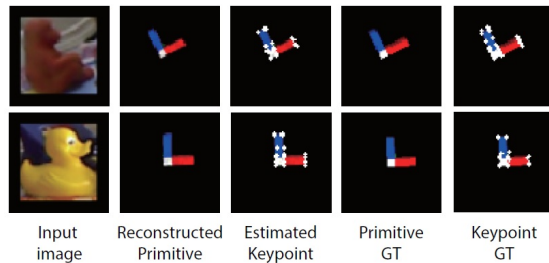| Input image | Reconstructed Primitive | Estimated Keypoint | Primitive GT | Keypoint GT |

Fig. 6: The examples of failure case.

ciated with each object. This paper presents a substantial improvement in the pose estimation even in the occluded case and comparable performance even using only synthetic images for training. The proposed method was validated using three public benchmark datasets yielding SOTA performance for occluded and small objects.

REFERENCES

[1] S. Peng, Y. Liu, Q. Huang, H. Bao, and X. Zhou, "Pvnet: Pixel-wise voting network for 6dof pose estimation," 12 2018.

[2] G. Billings and M. Johnson-Roberson, "Silhonet: An rgb method for 6d object pose estimation," *IEEE Robot. and Automat. Lett.*, 2019.

[3] A. S. Periyasamy, M. Schwarz, and S. Behnke, "Robust 6d object pose estimation in cluttered scenes using semantic segmentation and pose regression networks," in *Proc. IEEE/RSJ Intl. Conf. on Intell. Robots and Sys.*, 2018, pp. 6660–6666.

[4] 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 *Proc. IEEE Conf. on Comput. Vision and Pattern Recog.*, 2019, pp. 3343–3352.

[5] F. Manhardt, W. Kehl, and A. Gaidon, "Roi-10d: Monocular lifting of 2d detection to 6d pose and metric shape," in *Proc. IEEE Conf. on Comput. Vision and Pattern Recog.*, 2019, pp. 2069–2078.

[6] A. Saxena, J. Driemeyer, and A. Y. Ng, "Learning 3-d object orientation from images," in *Proc. IEEE Intl. Conf. on Robot. and Automat.*, 2009, pp. 794–800.

[7] E. Brachmann, A. Krull, S. Nowozin, J. Shotton, F. Michel, S. Gumhold, and C. Rother, "Dsac-differentiable ransac for camera localization," in *Proc. IEEE Conf. on Comput. Vision and Pattern Recog.*, 2017, pp. 6684–6692.

[8] E. Brachmann and C. Rother, "Learning less is more - 6d camera localization via 3d surface regression," *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 4654–4662, 2018.

[9] M. Sundermeyer, Z.-C. Marton, M. Durner, M. Brucker, and R. Triebel, "Implicit 3d orientation learning for 6d object detection from rgb images," in *Proc. European Conf. on Comput. Vision*, 2018, pp. 699–715.

[10] S. Xie, R. Girshick, P. Dollár, Z. Tu, and K. He, "Aggregated residual transformations for deep neural networks," pp. 5987–5995, 2017.

[11] M. Rad and V. Lepetit, "Bb8: A scalable, accurate, robust to partial occlusion method for predicting the 3d poses of challenging objects without using depth," in *Proc. IEEE Intl. Conf. on Comput. Vision*, 2017, pp. 3828–3836.

[12] B. Tekin, S. N. Sinha, and P. Fua, "Real-time seamless single shot 6d object pose prediction," in *Proc. IEEE Conf. on Comput. Vision and Pattern Recog.*, 2018, pp. 292–301.

[13] E. Brachmann, F. Michel, A. Krull, M. Ying Yang, S. Gumhold *et al.*, "Uncertainty-driven 6d pose estimation of objects and scenes from a single rgb image," in *Proc. IEEE*

method	PrimA6D-S			PrimA6D-SR			PVNET
IoU	1	0.9	0.75	1	0.9	0.75	-
LINEMOD	87.84	85.48	77.22	97.62	97.23	90.72	86.27
OCC-LINEMOD	51.53	50.31	43.65	59.77	59.17	50.24	40.77

TABLE V: Performance variation in terms of the ADD(-S) metric according to IoU showing the accuracy on bounding-box. PVNET results are copied from Table. I and Table. II for easy comparison.
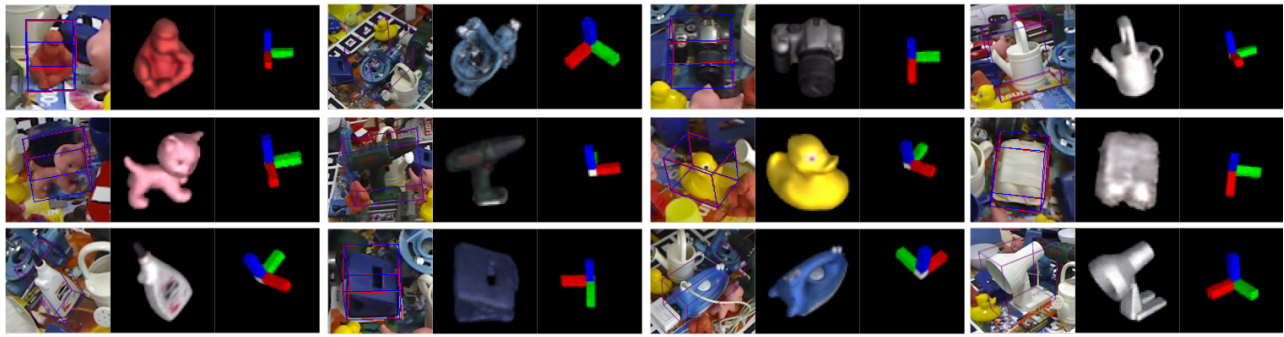
Fig. 7: Qualitative pose estimation evaluation on the LINEMOD dataset. The red box indicates the ground truth pose while blue is the estimated pose using PrimA6D. Next to the inferred pose is reconstructed object and the predicted primitive associated to each object.
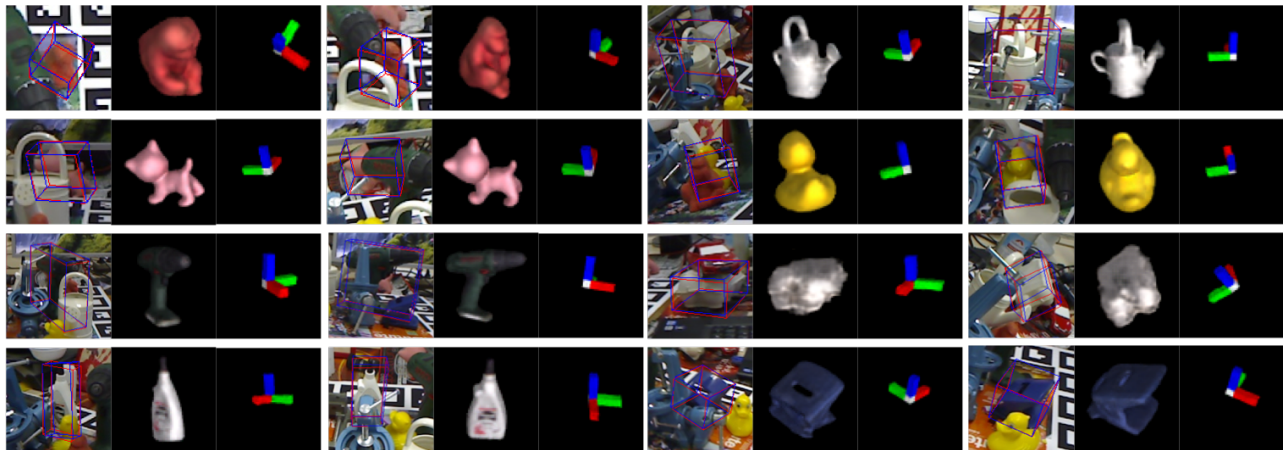

Fig. 8: Qualitative pose estimation evaluation on the Occlusion LINEMOD dataset. The red box indicates the ground truth pose while blue is the estimated pose using PrimA6D. Next to the inferred pose is reconstructed object and the predicted primitive associated to each object.

Conf. on Comput. Vision and Pattern Recog., 2016, pp. 3364–3372.

[14] M. Oberweger, M. Rad, and V. Lepetit, "Making deep heatmaps robust to partial occlusions for 3d object pose estimation," in *Proc. European Conf. on Comput. Vision*, 2018, pp. 119–134.

[15] K. Park, T. Patten, and M. Vincze, "Pix2pose: Pixel-wise coordinate regression of objects for 6d pose estimation," 2019, pp. 7667–7676.

[16] T. Do, M. Cai, T. Pham, and I. D. Reid, "Deep-6dpose: Recovering 6d object pose from a single RGB image," *CoRR*, vol. abs/1802.10367, 2018. [Online]. Available: http://arxiv.org/abs/1802.10367

[17] W. Kehl, F. Manhardt, F. Tombari, S. Ilic, and N. Navab, "Ssd-6d: Making rgb-based 3d detection and 6d pose estimation great again," in *Proc. IEEE Intl. Conf. on Comput. Vision*, 2017, pp. 1521–1529.

[18] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg, "Ssd: Single shot multibox detector," in *Proc. European Conf. on Comput. Vision*, 2016, pp. 21–37.

[19] Y. Xiang, T. Schmidt, V. Narayanan, and D. Fox, "Posecnn: A convolutional neural network for 6d object pose estimation in cluttered scenes," *Robotics: Science and Systems (RSS)*, 2018.

[20] J. Wu, B. Zhou, R. Russell, V. Kee, S. Wagner, M. Hebert, A. Torralba, and D. M. Johnson, "Real-time object pose estimation with pose interpreter networks," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2018, pp. 6798–6805.

[21] Q. Zhao, T. Sheng, Y. Wang, Z. Tang, Y. Chen, L. Cai, and H. Ling, "M2det: A single-shot object detector based on multi-level feature pyramid network," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, pp. 9259–9266, 07 2019.

[22] M. Jeon, Y. Lee, Y.-S. Shin, H. Jang, and A. Kim, "Underwater object detection and pose estimation using deep learning," vol. 52, no. 21. Elsevier, 2019, pp. 78–81.

[23] Y. Fan, S. Lyu, Y. Ying, and B. Hu, "Learning with average top-k loss," in *Advances in Neural Information Processing Sys. Conf.*, 2017, pp. 497–505.

[24] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," pp. 2672–2680, 2014. [Online]. Available: http://papers.nips.cc/paper/5423-generative-adversarial-nets.pdf

[25] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," *Proc. IEEE Conf. on Comput. Vision and Pattern Recog.*, pp. 770–778, 2016.

[26] S. Zakharov, I. Shugurov, and S. Ilic, "Dpod: 6d pose object detector and refiner," in *Proceedings of the IEEE International Conference on Computer Vision*, 2019, pp. 1941–1950.

[27] C. Song, J. Song, and Q. Huang, "Hybridpose: 6d object pose estimation under hybrid representations," June 2020.

[28] S. Hinterstoisser, V. Lepetit, S. Ilic, S. Holzer, G. Bradski, K. Konolige, and N. Navab, "Model based training, detection and pose estimation of texture-less 3d objects in heavily cluttered scenes," 2012, pp. 548–562.