Pseudo-keypoint RKHS Learning for Self-supervised 6DoF Pose Estimation

Yangzheng Wu and Michael Greenspan

RCVLab, Dept. of Electrical and Computer Engineering, Ingenuity Labs, Queen's University, Kingston, Ontario, Canada
{y.wu, michael.greenspan}@queensu.ca

Abstract. We address the simulation-to-real domain gap in six degree-of-freedom pose estimation (6DoF PE), and propose a novel self-supervised keypoint voting-based 6DoF PE framework, effectively narrowing this gap using a learnable kernel in RKHS. We formulate this domain gap as a distance in high-dimensional feature space, distinct from previous iterative matching methods. We propose an adapter network, which is pre-trained on purely synthetic data with synthetic ground truth poses, and which evolves the network parameters from this source synthetic domain to the target real domain. Importantly, the real data training only uses pseudo-poses estimated by pseudo-keypoints, and thereby requires no real ground truth data annotations. Our proposed method is called RKHSPose, and achieves state-of-the-art performance among self-supervised methods on three commonly used 6DoF PE datasets including LINEMOD (+4.2%), Occlusion LINEMOD (+2%), and YCB-Video (+3%). It also compares favorably to fully supervised methods on all six applicable BOP core datasets, achieving within −11.3% to +0.2% of the top fully supervised results.

Keywords: pose estimation · self-supervision · domain adaptation · keypoint estimation

1 Introduction

RGB-D Six Degree-of-Freedom Pose Estimation (6DoF PE) is a problem being actively explored in computer vision research. Given an RGB image and its associated depth map, the task is to detect scene objects and estimate their poses comprising 3DoF rotational angles and 3DoF translational offsets in the camera reference frame. This task enables many applications such as augmented reality [34,36,40,84], robotic bin picking [19,35,43], autonomous driving [51,85] and image-guided surgeries [21,25].

As with other machine learning (ML) tasks, fully supervised 6DoF PE requires large annotated datasets. This requirement is particularly challenging for 6DoF PE, as the annotations comprise not only the identity of the objects in the scene, but also their 6DoF pose, which makes the data relatively expensive to annotate compared to related tasks such as classification, detection and segmentation. This is due to the fact that humans are not able to qualitatively or

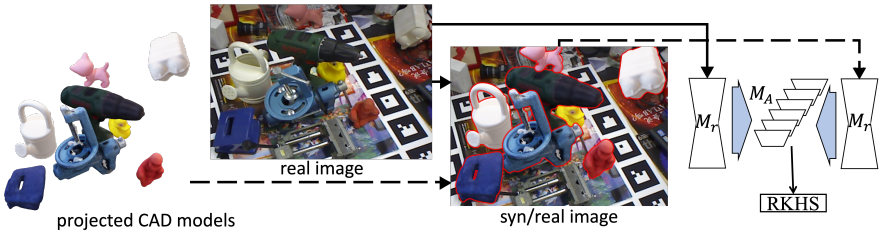

Fig. 1: RKHSPose adapts the network pretrained on synthetic data to real test scenes (left), by comparing network feature spaces with real image inputs (solid arrows), against those with syn/real image (right) inputs (dashed arrows). M_r regresses radial quantities, M_A is the Adapter network, and RKHS maps features into a higher dimensional space.

intuitively estimate 6DoF pose, which therefore requires additional instrumentation of the scene at data collection, and/or more sophisticated user annotation tools [34, 84]. Consequently, synthetic 6DoF PE datasets [36, 43] have been introduced, either as an additional complement to real datasets, or as standalone purely synthetic datasets. Annotated synthetic datasets are of course trivially inexpensive to collect, simply because precise synthetic pose annotation in a simulated environment is fully automatic. A known challenge in using synthetic data, however, is that there typically exists a domain gap between the real and synthetic data, which makes results less accurate when inferring in real data using models trained on purely synthetic datasets. Expectations of the potential benefit of synthetic datasets has led to the exploration of a rich set of Domain Adaptation (DA) methods, which specifically aim to reduce the domain gap [6,62,88] using inexpensive synthetic data for a wide variety of tasks, recently including 6DoF PE [28, 45, 47].

Early methods [42, 83] ignored the simulation-to-real (sim2real) domain gap and nevertheless improved performance by training on both synthetic and real annotated data, effectively augmenting the real images with the synthetic. However, these methods still required real labels to be sufficiently accurate and robust for practical applications, partially due to the domain gap. As shown in Fig. 1, the rendered synthetic objects (right image) have a slightly different appearance than the real objects (left image). The details of the CAD models, both geometry and texture, are not precise, as can be seen for the *can* object (which lacks a mouth and shadows), the *benchvise* (which is missing a handle), and the *holepuncher* (which has coarse geometric resolution). Several recent methods [9, 44, 73, 75, 80] have started to address the sim2real gap for 6DoF PE by first training on labeled synthetic data and then fine-tuning on unlabeled real data. Commonly known as *self-supervised*, these methods reduce the domain gap by adding extra supervision using features extracted from real images without requiring real ground truth (GT) labels. The majority of these methods are view-

point/template matching-based, and the self-supervision commonly iteratively matches 2D masks or 3D point clouds [44, 73, 80].

While the above-mentioned self-supervison works have shown promise, there exist a wealth of DA techniques that can be brought to bear to improve performance further for this task. One such technique is Reproducing Kernel Hilbert Space (*RKHS*), which is a kernel method that has been shown to be effective for DA [1, 5, 41]. RKHS was initially used to create decision boundaries for non-separable data [12, 60], and has been shown to be effective at reducing the domain gap for various tasks and applications [7, 65, 89]. The reproducing kernel guarantees that the domain gap can be statistically measured, allowing network parameters trained on synthetic data to be effectively adapted to the real data, using specifically tailored metrics.

To address the sim2real domain gap in 6DoF PE, we propose RKHSPose, which is a keypoint-based method [82, 83] trained on a mixture of a large collection of labeled synthetic data, and a small handful of unlabeled real data. RKHSPose estimates the intermediate radial voting quantity, which has been shown to be effective for estimating keypoints [83], by first training a modified FCN-Resnet-18 on purely synthetic data, with automatically labeled synthetic GT poses. The radial quantity is a map of the distance from each image pixel to each keypoint. Next, real images are passed through the synthetically trained network, resulting in a set of pseudo-keypoints. The real images and their corresponding pseudo-keypoints are used to render a set of 'synthetic-over-real' (*syn/real*) images, by first estimating the pseudo-pose from the pseudo-keypoints, and then overlaying the synthetic object, rendered with the pseudo-pose, onto the real image. The network training then continues on the syn/real images, invoking an RKHS network module with a trainable linear product kernel, which minimizes the Maximum Mean Discrepancy (MMD) loss. At the front end, a proposed keypoint radial voting network learns to cast votes to estimate keypoints from the backend-generated radial maps. The final pose is then determined using ePnP [46] based on the estimated and corresponding object keypoints.

The main contributions of this work are:

- A novel learnable RKHS-based Adapter backend network architecture to minimize the sim2real domain gap in 6DoF PE;
- A novel CNN-based frontend network for keypoint radial voting;
- A self-supervised keypoint-based 6DoF PE method, RKHSPose, which is shown to have state-of-the-art (SOTA) performance, based on our experiments and several ablation studies.

2 Related Work

2.1 6DoF PE

ML-based 6DoF PE methods [57, 61, 84] all train a network to regress quantities, such as keypoints and camera viewpoints, as have been used in classical algorithms [34]. ML-based methods, which initially became popular for the general

Table 1: Existing self-supervised 6DoF PE methods. Some methods use CAD models, labeled synthetic (syn) data (images+poses), and real data without GT labels, while others use only labeled synthetic data. Many methods require a ROI such as a bounding box or a semantic mask, manually labeled or estimated by an existing framework. SO-Pose [18] used very few real labels to improve performance.

method	mode	CAD model	syn data	real images	real poses	ROI
SO-Pose [18]	RGB	✓	✓	✓	✓	✓
TexPose [9]	RGB	✓	✓	✓	✗	✓
SMOC-Net [75]	RGB	✓	✓	✓	✗	✓
FS6D [33]	RGBD	✓	✓	✗	✗	✓
AAE [73]	RGB	✓	✓	✗	✗	✓
MHP [54]	RGB	✓	✓	✗	✗	✓
Sock et al. [68]	RGB	✓	✓	✗	✗	✓
DSC [86]	RGBD	✓	✓	✗	✗	✓
Sundermeyer [71]	RGB	✓	✓	✗	✗	✓
LatentFusion [59]	RGB	✗	✓	✗	✗	✓
OSOP [66]	RGBD	✓	✓	✗	✗	✗
Kleeberger et al. [44]	D	✓	✓	✗	✗	✗
Su et al. [69]	RGB	✓	✓	✓	✗	✗
Self6D [80]	RGBD	✓	✓	✓	✗	✗
Self6D++ [79]	RGB	✓	✓	✓	✗	✗
Deng et al. [15]	RGBD	✓	✓	✓	✗	✗
RKHSPose (Ours)	RGBD	✓	✓	✓	✗	✗

object detection task, have started to dominate the 6DoF PE literature due to their accuracy and efficiency.

There are two main categories of ML-based fully supervised methods: *feature matching*-based [29, 30, 57, 70, 84], and *keypoint*-based methods [31, 32, 61, 87]. Feature matching-based methods make use of the structures from general object detection networks most directly. The network encodes and matches features and estimates pose by either regressing elements of the pose (e.g. the transformation matrix [75], rotational angles and translational offsets [84] or 3D vertices [70]) directly, or by regressing some intermediate feature-matching representations, such as viewpoints or segments.

In contrast, keypoint-based methods encode features to estimate keypoints which are predefined within the reference frame of an object's CAD model. These methods then use (modified) classical algorithms such as PnP [22, 46], Horn's method [38], and ICP [3] to estimate the final pose from corresponding image and model keypoints. Unlike feature-matching methods, keypoint-based methods are typically more accurate due to redundancies encountered through voting schemes [32,61,83] and by generating confidence hypotheses of keypoints [31,87].

Recently, self-supervised 6DoF PE methods have been explored in order to reduce the reliance on labeled real data, which is expensive to acquire. As sum-

marized in Table 1, these methods commonly use real images without GT labels. Some methods use pure synthetic data and CAD models only, with the exception of LatentFusion [59] which trained the model using only synthetic data. The majority of these methods [9, 15, 33, 48, 66, 68, 69, 71, 75, 79, 80] are inspired by fully supervised feature-matching methods, except DPODv2 [67] and DSC [86], in which keypoint correspondences are rendered and matched.

A few methods [15, 66] fine-tuned the pose trivially by iteratively matching the template/viewpoint, whereas others [33, 44, 59, 86] augmented the training data by adding noise [44, 86], rendering textures [33, 67] and creating a latent space [59]. Some methods [54, 71, 73] also implemented DA techniques such as codebook encoding [71], Principle Component Analysis (PCA) [71] and symmetric Bingham distributions [54]. Most methods [66,68,69,79,80,86] used rendering techniques to render and match a template. There are a few methods that combined 3D reconstruction techniques, such as Neural radiance fields (Nerf) [55] and Structure from Motion (SfM) [76]. TexPose [9] matched CAD models to segments generated by Nerf, and SMOC-Net [75] used SfM to create the 3D segment and matched with the CAD model.

2.2 Kernel Methods and Deep Learning

While deep learning is the most common ML technique within the computer vision literature, kernel methods [12,60,64] have also been actively explored. Kernel methods are typically in RKHS space [74] with reproducing properties that facilitate solving non-linear problems by mapping input data into high dimensional spaces that can be linearly separated [11, 23]. Well-known early kernel methods that have been applied to computer vision are Support Vector Machines [12] and PCA [60]. A recent method [64] linked energy distance and MMD in RKHS, and showed the effectiveness of kernels in statistical hypothesis testing.

More recent studies compare kernel methods with deep learning networks [1, 4, 5, 41]. RKHS is found to perform better on classification tasks than one single block of a CNN comprising convolution, pooling and downsampling (linear) layers [41]. RKHS can also help with CNN generalization by meta-regularization on image registration problems [1]. Similarly, norms (magnitude of trainable weights) defined in RKHS help with CNN regularization [4, 5]. Further, discriminant information in label distributions in unsupervised DA is addressed and RKHS-based Conditional Kernel Bures metric is proposed [50]. Lastly, the connection between Neural Tangent Kernel and MMD is established, and an efficient MMD for two-sample tests is developed [10].

Inspired by the previous work, RKHSPose applies concepts of kernel learning to keypoint-based 6DoF PE, to provide an effective means to self-supervise a synthetically trained network on unlabeled real data.

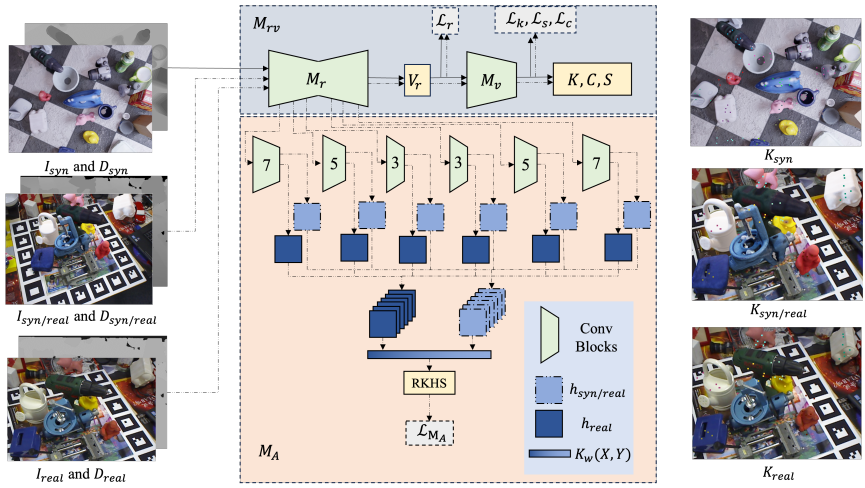

Fig. 2: RKHSPose architecture. RKHPose is first trained on synthetic labeled data (solid arrows), and then finetuned on alternating syn/real and (unlabeled) real images (dashed arrows). M_A is measured by MMD in RKHS by densely mapping the intermediate features of M_r into high dimensional spaces with conv blocks. The distance is treated as $\mathcal{L}_{M_A}$ and back-propagated through M_A and M_r.

3 Method

3.1 Network Overview

As shown in Fig. 2, RKHSPose is made up of two networks, a main network M_{rv} for keypoint regression and classification, and an Adapter network M_A. The input of M_{rv} with shape $W \times H \times 4$, is the concatenation of an RGB image I and its corresponding depth map D. This input can be synthetically generated with an arbitrary background (I_{syn} and D_{syn}), a synthetic mask overlayed on a real background ($I_{syn/real}$ and $D_{syn/real}$), or a real (I_{real} and D_{real}) image. The outputs are n projected 2D keypoints K, along with corresponding classification labels C and confidence scores S. K are organized into instance sets based on C and geometric constraints.

M_{rv} comprises two sub-networks, regression network M_r and voting network M_v. Inspired by recent voting techniques [32, 61, 82, 83, 90], M_r estimates an intermediate voting quantity, which is a radial distance map V_r [83], by using a modified Fully Connected ResNet 18 (FCN-ResNet-18). The radial voting map V_r, with shape $W \times H$, stores the Euclidean distance from each object point to each keypoint in the 3D camera world reference frame. The voting network M_v (described in Sec. 3.3) then takes V_r as input, accumulates votes, and detects peaks to estimate K, C and S.

The Adapter network M_A consists of a series of CNNs which encode pairs of feature maps from M_{rv}, and are trained on both synthetic overlayed and pure real

data. M_A encodes feature map pairs $(f_{syn/real}, f_{real})$ into corresponding high-dimensional feature maps $(h_{syn/real}, h_{real})$. The input data, $(I_{syn/real}, I_{real})$ and $(D_{syn/real}, D_{real})$, are also treated as $(f_{syn/real}, f_{real})$ during the learning of M_A. Each of these networks creates a high-dimensional latent space, essentially the Reproducing Kernel Hilbert Space (RKHS) [2] and contributes to the learning of both M_{rv} and M_A by calculating the MMD [27]. While RKHS has been applied effectively to other DA tasks, to our knowledge, this is the first time that it has been applied to 6DoF PE, and the adapter network architecture is novel.

The loss function $\mathcal{L}$ is made up of five elements: Radial regression loss $\mathcal{L}_r$ for V_r; Keypoint projection loss $\mathcal{L}_k$ for K; Classification loss $\mathcal{L}_c$ for C; Confidence loss $\mathcal{L}_s$ for S, and finally; Adapter loss $\mathcal{L}_A$ for the comparison of intermediate feature maps. The regression losses $\mathcal{L}_r$, $\mathcal{L}_k$, and $\mathcal{L}_s$ all use the smooth L1 metric, whereas classification loss $\mathcal{L}_c$ uses the cross-entropy metric $H(\cdot)$. RKHSPose losses can then be denoted as:

$$\mathcal{L}_r = smooth_{\mathcal{L}_1}(V_r, \widehat{V}_r) \tag{1}$$

$$\mathcal{L}_k = smooth_{\mathcal{L}_1}(K, \widehat{K}) \tag{2}$$

$$\mathcal{L}_c = H(p(C), \widehat{p}(C)) \tag{3}$$

$$\mathcal{L}_s = smooth_{\mathcal{L}_1}(S, \widehat{S}) \tag{4}$$

$$\mathcal{L}_A = MMD(\widehat{f}_{syn/real}, \widehat{f}_{real}) \tag{5}$$

$$\mathcal{L} = \lambda_r \mathcal{L}_r + \lambda_k \mathcal{L}_k + \lambda_c \mathcal{L}_c + \lambda_s \mathcal{L}_s + \lambda_A \mathcal{L}_A \tag{6}$$

where λ_r, λ_k, λ_c, λ_s, and λ_A are weights for adjustment during training, and all non-hatted quantities are GT values. At inference, M_{rv} takes I_{real} and D_{real} as input, and outputs K, C and S. The keypoints K are ranked and grouped by S and C, and are then forwarded into the ePnP [46] algorithm which estimates 6DoF pose values. ICP can then be optionally applied using the depth data to refine the estimated pose.

3.2 Convolutional RKHS Adapter

Reproducing Kernel Hilbert Space $\mathcal{H}$ is a commonly used vector space for Domain Adaptation [58, 77]. Hilbert Space is a complete metric space (in which every Cauchy sequence of points has a limit within the metric) represented by the inner product of vectors. For a non-empty set of data $\mathcal{X}$, a function $\mathcal{K}_{\mathcal{X}} : \mathcal{X} \times \mathcal{X} \to \mathcal{R}$ is a reproducing kernel if:

$$\begin{cases} k(\cdot, x) \in \mathcal{H} \quad \forall \quad x \in \mathcal{X} \\ \langle f(\cdot), k(\cdot, x) \rangle = f(x) \quad \forall \quad x \in \mathcal{X}, f \in \mathcal{H} \end{cases} \tag{7}$$

where $\langle a, b \rangle$ denotes the inner product of two vectors a and b, $k(\cdot, x) = \mathcal{K}_{\mathcal{X}}$ for each $x \in \mathcal{X}$, and f is a function in $\mathcal{H}$. The second equation in Eq. 7 is an expression of the reproducing property of $\mathcal{H}$.

In order to utilize $\mathcal{H}$ for DA, we expand the kernel definition to two sets of data $\mathcal{X}$ and $\mathcal{Y}$. The reproducing kernel can then be defined as:

$$\mathcal{K}(\mathcal{X}, \mathcal{Y}) = \langle \mathcal{K}_{\mathcal{X}}, \mathcal{K}_{\mathcal{Y}} \rangle_H \tag{8}$$

Here, $\mathcal{K}_{\mathcal{X}}$ and $\mathcal{K}_{\mathcal{Y}}$ are themselves inner product kernels that map $\mathcal{X}$ and $\mathcal{Y}$ respectively into their own Hilbert spaces, and $\mathcal{K}(\mathcal{X}, \mathcal{Y})$ is the joint Hilbert space kernel of $\mathcal{X}$ and $\mathcal{Y}$. Note that $\mathcal{K}(\mathcal{X}, \mathcal{Y})$ is not the kernel commonly defined for CNNs. Rather, it is the similarity function defined in kernel methods, such as is used by Support Vector Machine (SVM) techniques to calculate similarity measurements. Some recent methods [8, 52, 53, 56] named it a Convolutional Kernel Network (CKN) to distinguish it from CNNs.

Various kernels of CKN, such as the Gaussian Kernel [53], the RBF Kernel [52] and the Inner Product Kernel [56], have been shown to be comparable to shallow CNNs for various tasks, especially for Domain Adaption. The most intuitive inner product kernel is mathematically similar to a fully connected layer, where trainable weights are multiplied by the input feature map. To allow the application of RKHS methods into our CNN based Adapter network M_A, trainable weights are added to $\mathcal{K}(\mathcal{X}, \mathcal{Y})$ [49]. The trainable Kernel $\mathcal{K}_w(\mathcal{X}, \mathcal{Y})$ can then be denoted as:

$$\mathcal{K}_w(\mathcal{X}, \mathcal{Y}) = \langle \langle \mathcal{X}, W_{\mathcal{X}} \rangle, \langle \mathcal{Y}, W_{\mathcal{Y}} \rangle \rangle_H \tag{9}$$

where $W_{\mathcal{X}}$ and $W_{\mathcal{Y}}$ are trainable weights. By adding $W_{\mathcal{X}}$ and $W_{\mathcal{Y}}$, $K_w(\mathcal{X}, \mathcal{Y})$ still satisfies the RKHS constraints.

The sim2real domain gap of feature maps f being trained in M_{rv} (with few real images and no real GT labels) are hard to measure using trivial distance metrics. In contrast, RKHS can be a more accurate and robust space for comparison, since it is known to be capable of handling high-dimensional data with a low number of samples [24, 56]. To compare $(f_{syn/real}, f_{real})$ in RKHS, a series of CNN layers encodes $(f_{syn/real}, f_{real})$ into higher-dimensional features $(h_{syn/real}, h_{real})$, followed by the trainable $\mathcal{K}_w(\mathcal{X}, \mathcal{Y})$. Once mapped into RKHS, $h_{syn/real} = \{sr_i\}_{i=1}^m$ and $h_{real} = \{r_i\}_{i=1}^m$ can then be measured by Maximum Mean Discrepancy (MMD), a common DA measurement [27, 49], which is the square distance between the kernel embedding [26]:

$$MMD(h_{syn/real}, h_{real}) = \frac{1}{m} \left[\left(\sum_{i=1}^{m} \sum_{j=1}^{m} k_w(sr_i, sr_j) - \sum_{i=1}^{m} k_w(sr_i, sr_i) \right) \right.$$
$$\left. - \left(\sum_{i=1}^{m} \sum_{j=1}^{m} k_w(sr_i, r_j) - \sum_{i=1}^{m} k_w(sr_i, r_i) \right) + \left(\sum_{i=1}^{m} \sum_{j=1}^{m} k_w(r_i, r_j) - \sum_{i=1}^{m} k_w(r_i, r_i) \right) \right]^{\frac{1}{2}} \tag{10}$$

where $k_w()$ is the feature element of $\mathcal{K}_w()$.

In summary, the Adapter M_A shown in Fig. 2 measures MMD for each $(h_{syn/real}, h_{real})$ in RKHS, by increasing the $f_{syn/real}$ and f_{real} dimension using a CNN and thereby constructing a learnable kernel $\mathcal{K}_w$. The outputs are the feature maps $h_{syn/real}$ and h_{real}, which are supervised by loss $\mathcal{L}_A$ during training of the real data epochs. Based on the experiments in Sec. 5.3, our trainable inner product kernel is shown to be more accurate for our task than other known kernels [52, 53, 56] that we tested, that are often used in such kernel methods.

3.3 Keypoint Radial Voting Network

The network M_v votes for keypoints using a CNN architecture, taking the radial voting quantity V_r resulting from M_r as input. VoteNet [63] previously used a CNN approach to vote for object centers, whereas other keypoint-based techniques have implemented GPU-based parallel RANSAC [32,61] methods for offset and vector quantities. The radial quantity is known to be more accurate than the vector or offset quantities, and has been previously implemented with a CPU-based parallel accumulator space method [82,83]. Given its superior accuracy, M_v implements radial voting using a CNN to improve efficiency. Given a 2D radial map $\widehat{V}_r$ estimated by M_r and supervised by GT radial maps V_r, the task is to accumulate votes, find the peak, and estimate the keypoint location. The V_r foreground pixels (which lie on the target object) store the Euclidean distance from these pixels to each of the keypoints, with background (non-object) pixels set to value -1.

The estimated radial map $\widehat{V}_r$ is indeed an inverse heat map of the candidate keypoints' locations, distributed in a radial pattern centered at the keypoints. To forward $\widehat{V}_r$ into a CNN voting module, it is inversely normalized so that it becomes a heat map. Let v_r^{max} and v_r^{min} be the maximum and minimum global radial distances for all objects in a dataset, which can be calculated by iterating through all GT radial maps, or alternately generated from the object CAD models. An inverse radial map $\widehat{V}_r^{-1}$ can then be denoted as $\widehat{V}_r^{-1} = (v_r^{max} - \widehat{V}_r)/(v_r^{max} - v_r^{min})$. Voting network M_v takes $\widehat{V}_r^{-1}$ as the input and generates the accumulated vote map by a series of convolution, ReLu, and batch normalization layers. The complete network architecture is provided in the Supplementary material Sec. S.2. The background pixels are filtered out by a ReLu layer, and only foreground pixels contribute to voting. The accumulated vote map is then max-pooled for peak extraction, and reshaped using a fully connected layer into a $n \times 4$ output. The output represents n keypoints and comprises $n \times 2$ projected 2D keypoints K, n classification labels C, and n confidence scores S. The labels C indicate which object the corresponding keypoint belongs to, and S ranks the confidence level of keypoints before being forwarded into ePnP [46] for pose estimation. M_v is supervised by $\mathcal{L}_k$, $\mathcal{L}_c$, and $\mathcal{L}_s$ (Eqs. 2-4) and is trained end-to-end along with M_r.

4 Experiments

4.1 Datasets and Evaluation Metrics

RKHSPose uses BOP Procedural Blender [17] (PBR) synthetic images [16, 36, 37, 72] for the synthetic training phase. The images are generated by dropping synthetic objects (CAD models) onto a plane in a simulated environment using PyBullet [13], and then rendering them with synthetic textures. All objects in the synthetic images are thus automatically labeled with precise GT poses. We evaluated RKHSPose for the six BOP [37,72] core datasets (LMO [34], YCB [84], TLESS [35], TUDL [36], ITODO [20], and HB [40]), all except IC-BIN [19],

which does not include any real training or validation images and is therefore not applicable. ITODO and HB have no real images in the training set, and so for training we instead used the real images in their validation sets, which were disjoint from their test sets.

Our main results are evaluated with the ADD(S) [34] metric for the LM and LMO dataset, and the ADD(S) AUC [84] metric for the YCB dataset. These are the standard metrics commonly used to compare self-supervised 6DoF PE methods. ADD(S) is based on the mean distance (minimum distance for symmetry) between the object surfaces for GT and estimated poses, whereas ADD(S) AUC plots a curve formed by ADD(S) for various object diameter thresholds. We use the BOP average recall (AR) metrics for our ablation studies. The AR metric, based on the original ADD(S) [34], evaluates three aspects, including Visible Surface Discrepancy (AR_{VSD}), Maximum Symmetry-Aware Surface/Projection Distance (AR_{MSSD} and AR_{MSPD}) [36].

4.2 Implementation Details

RKHSPose is trained on a server with an Intel Xeon 5218 CPU and two RTX6000 GPUs with a batch size of 32. The Adam optimizer is used for the training of M_{rv}, on both synthetic and real data, and M_A is optimized by SGD. Both of the optimizers have an initial learning rate of $lr = 1e\text{-}3$ and weight decay $1e\text{-}4$ for 80 and 20 epochs respectively.

The input of the network is normalized before training, as follows. The RGB images I are normalized and standardized using ImageNet [14] specifications. The depth maps are each individually normalized by their local minima and maxima, to lie within a range of 0 to 1. The radial distances in radial map V_r and 2D projected keypoints are both normalized by the width and height of I.

A single set of four keypoints is chosen by KeyGNet [81] for the set of all objects in each dataset. One extra *background* class is added to C in order to filter out the redundant background points in K. M_{rv} is first trained for 120 epochs on synthetic data, during which M_A remains frozen. Following this, training proceeds for an additional 80 epochs which alternate between real and synthetic data. When training on real data, both M_A and M_{rv} weights are learned, whereas M_A is frozen for the alternating synthetic data training.

During real data epochs, M_{rv} initially estimates pseudo-keypoints for each real image. These pseudo-keypoints are then forwarded into ePnP for pseudo-pose estimation. Each estimated pseudo-pose is then augmented into a set of poses P_{aug} by applying arbitrary rotational and translational perturbations with respective ranges of $[-\frac{\pi}{18}, \frac{\pi}{18}]$ radians and $[-0.1, 0.1]$ along three axes within the normalized model frame, which is defined using the largest object in the dataset. The set of syn/real images are rendered by overlaying onto the real image the CAD model of each object using each augmented pose value in P_{aug}. The cardinality of P_{aug} is set to be one less than the batch size, and the adapter M_A is trained on a mini-batch of the hybrid images resulting from P_{aug}, plus the image resulting from the original estimated pseudo-pose.

Table 2: Comparison with other methods. Accuracy of RKHSPose for LM and LMO is evaluated with ADD(S), and for YCB is evaluated with ADD(S) AUC. All 'Supervision: Syn + Self' methods use real images without real labels.

Method	Real data image	Real data label	Dataset/Metric LM ADD(S)	LMO ADD(S)	YCB ADD(S) AUC	YCB ADD-S AUC
Supervision: Syn (lower bound)						
AAE	✗	✗	31.4	-	-	-
MHP	✗	✗	38.8	-	-	-
GDR (TexPose version)	✗	✗	77.4	52.9	-	-
Self6D++	✗	✗	77.4	52.9	77.8	89.4
Self6D++ with D_{ref}	✗	✗	88.0	62.5	79.2	90.1
Ours	✗	✗	78.2	54.3	76.5	90.2
Ours+ICP	✗	✗	87.9	55.7	78.3	91.3
Supervision: Syn + Self						
Sock *et al.*	✓	✗	60.6	22.8	-	-
DSC	✓	✗	58.6	24.8	-	-
Self6D	✓	✗	58.9	32.1	-	-
SMOC-Net	✓	✗	91.3	63.3	-	-
Self6D++	✓	✗	88.5	64.7	80.0	91.4
TexPose	✓	✗	91.7	66.7	-	-
Ours	✓	✗	95.8	68.6	82.8	92.4
Ours+ICP	✓	✗	**95.9**	**68.7**	**83.0**	**92.6**
Supervision: Syn + Real GT (upper bound)						
SO-Pose	✓	✓	96.0	62.3	83.9	90.9
Self6D++	✓	✓	91.0	74.4	82.6	90.7
Ours	✓	✓	96.7	70.8	85.4	92.2
Ours+ICP	✓	✓	**96.8**	**71.3**	**85.6**	**92.4**

Initially M_A is frozen, and for the first 80 epochs, the loss is set to emphasize the classification and the visibility score regression, i.e. $\lambda_c = \lambda_s = 0.6$ and $\lambda_r = \lambda_k = 0.4$. Following this, up to epoch 200, the scales of losses are then exchanged to fine-tune the localization of the keypoints, i.e. $\lambda_c = \lambda_s = 0.4$ and $\lambda_r = \lambda_k = 0.6$. After epoch 120, M_A is unfrozen each alternating epoch, and λ_D is set to 1 during the remaining M_A training epochs. This training strategy, shown in Fig. 2, minimizes the sim2real gap without any real image GT labels, and using very few (320) real images.

4.3 Results

The results are summarized in Tables 2 and 3. To our knowledge, RKHSPose outperforms all existing self-supverised 6DoF PE methods. In Table 2, the upper bound fully supervised, lower bound synthetically supervised, and middle self-supervised methods (including ours) are compared. On LM and LMO, our ADD(S) is +4.2% and +2% better than the second best method TexPose [9]. We

Table 3: Comparison with fully supervised methods. RKHSPose results on TLESS (−1.8), TUDL (−0.4), ITODD (−4.6) and HB (+0.1) compares to SOTA methods with full supervision of real GT labels. Methods annotated with * use the detection results from other detection methods.

Method	real label	Dataset						
		LM	LMO	TLESS	TUDL	ITODD	HB	YCB
SurfEmb* [30]	✓	-	76.0	82.8	85.4	65.9	86.6	79.9
RCVPose3D [82]	✓	-	72.9	70.8	**96.6**	**73.3**	86.3	84.3
RADet [87]+PFA* [39]	✓	-	**79.7**	85.0	96.0	67.6	86.9	88.8
ZebraPose [70]	✓	-	78.0	**86.2**	95.6	65.4	92.1	**89.9**
Ours	✗	95.7	68.2	85.5	96.2	68.6	92.2	83.6
Ours+ICP	✗	95.8	68.4	85.6	96.2	68.7	**92.3**	83.8

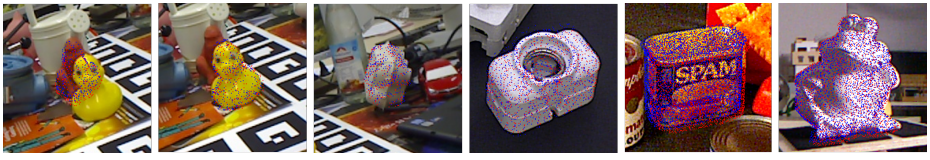

Fig. 3: Qualitative overlay results on selected images. Red dots and blue dots are projected surface points from GT poses and estimated poses, respectively.

compared our performance to Self6D++ [79], which was the only other method that evaluated using YCB, and saw a 3% improvement after ICP on ADD(S) AUC. Last but not least, in Table 3 our performance evaluated on the other four BOP core datasets is comparable to several upper bound methods from the BOP leaderboard. Some test scenes with RKHSPose results are shown in Fig. 3.

RKHSPose runs at 34 fps on an Intel i7 2.5GHz CPU and an RTX 3090 GPU with 24G VRAM. It takes on average 8.7 ms for loading data, 4.5 ms for forward inference through M_{rv} (×10 faster compared to the analytical radial voting in RCVPose [83]), and 16.2 ms for ePnP.

5 Ablation Studies

5.1 Dense Vs. Sparse Adapter

The Adapter M_A densely matches the intermediate feature maps, whereas the majority of other methods [15,75,80] only compare the final output. To show the benefits of dense comparison, we conduct an experiment with different variations of M_A. A sparse matching M_A^s network is trained on synthetic and real data comparing only a single feature map, which is the intermediate radial map. M_A^s has the exact same overall learning capacity (number of parameters) as the dense matching M_A described in Sec. 3.3. The results in Table 4 show that M_A

Table 4: AR of different adapters on LM and five BOP core datasets.

Adapter	AR_{VSD}	AR_{MSSD}	AR_{MSPD}	AR
M_A^s	78.1	77.8	77.8	77.9
M_A	84.9	84.1	84.3	**84.4**

Table 5: AR of different training strategies on LM and five BOP core datasets.

Training Type	AR_{VSD}	AR_{MSSD}	AR_{MSPD}	AR
Mixed	84.9	84.1	84.3	**84.4**
Sequantial	82.3	81.7	81.7	81.9

Table 6: [R Tab A] AR of different kernels on LM and five BOP core datasets.

Kernel	w	AR_{VSD}	AR_{MSSD}	AR_{MSPD}	AR
Linear	✗	71.6	70.8	70.6	71.0
	✓	84.9	84.1	84.3	**84.4**
RBF	✗	73.4	72.9	73.2	73.2
	✓	82.5	81.3	81.5	81.6

Table 7: AR of different metrics on LM and five BOP core datasets.

Metric	AR_{VSD}	AR_{MSSD}	AR_{MSPD}	AR
MMD	84.9	84.1	84.3	**84.4**
KL Div	78.0	77.8	78.0	77.9
Wass	80.9	80.6	80.9	80.8

surpassed M_A^s on all six datasets tested. Specifically, on ITODD, M_A is 12.1% more accurate than M_A^s. This experiment shows the effectiveness of our densely matched M_A.

5.2 Syn/Real Synchronized Training

When training RKHSPose, real epochs are alternated with synthetic epochs. In contrast, some other methods [9, 75, 79, 80] separate the synthetic/real training. We conducted an experiment to compare these two different training strategies, the results of which are shown in Table 5. The alternating training performs slightly better (+2.5% on average) than the sequential training, possibly due to the early access to real scenes thereby avoiding local minima.

5.3 Adapter Kernels and Metrics

We use a linear (dot product) kernel and MMD in RKHS for domain gap measurements. There are various other kernels and similarity measurements that can be implemented in RKHS as described in Sec. 3.2. First, we add trainable weights to the radial basis function (RBF) kernel in a similar manner as K_w defined in Eq. 9. The trainable RBF kernel on two sets of data X and Y is denoted as:

$$K_{rbf}(X, Y) = exp(-w \left\| X - Y \right\|^2) \tag{11}$$

where w are the trainable weights, which replaces the original adjustable parameter in the classical RBF kernel. We also experiment with the classical RKHS kernel functions without trainable weights, including the inner product kernel and the original RBF kernel [78], for comparison. Further, we experiment on other commonly used distance measures, including Kullback-Leibler Divergence (KL Div, i.e. relative-entropy) and Wasserstein (Wass) Distance.

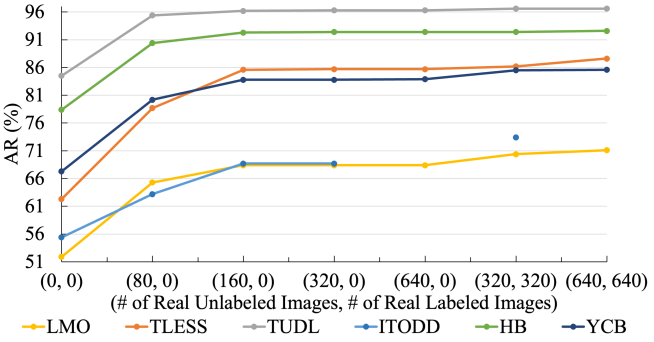

Fig. 4: Impact of # of real images with/without GT labels used during training. All datasets are evaluated by the BOP AR metric. We conduct experiments from 0 to 640 real images on all datasets, except ITODD which contained only 357 real images.

In Table 6, the RBF kernel performs similar to the linear product kernel with a slight performance dip. In Table 7, MMD minimizes the domain gap better than the Wass Distance, followed by the KL Div metric, leading to a better overall performance on *AR*. Based on these results, we used the linear product kernel with trainable weights and selected MMD as the main loss metric.

5.4 Number of Real Images and Real Labels

The objective of RKHSPose is to reduce real data usage and train without any real GT labels. To show the effectiveness of the approach, we conducted an experiment by training on different numbers of real images, the results of which are shown in Fig. 4. We used up to 640 real images in all cases, except for that of ITODD which contains only 357 real images. The *AR* of all datasets saturates at 160 images except YCB. The further improvement of YCB beyond 160 images is also only +0.1% and saturates after 320 real images. We nevertheless use 320 real unlabeled images for our main results. This experiment showed that adding more than 320 real labeled images did not significantly improve performance.

6 Conclusion

To sum up, we propose a novel self-supervised keypoint radial voting-based 6DoF PE method using RGB-D data called RKHSPose. RKHSPose fine-tunes poses pre-trained on synthetic data by densely matching features with a learnable kernel in RKHS, using real data albeit without any real GT poses. By applying this DA technique in feature space, RKHSPose achieved SOTA performance on the six applicable BOP core datasets, surpassing the performance of all other self-supervised methods. Notably, the RKHSPose performance closely approaches that of several fully-supervised methods, which indicates the strength of the approach at reducing the sim2real domain gap for this problem.

Acknowledgements: Thanks to Bluewrist Inc. and NSERC for their support of this work.

References

1. Al Safadi, E., Song, X.: Learning-based image registration with meta-regularization. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 10928–10937 (2021)
2. Aronszajn, N.: Theory of reproducing kernels. Transactions of the American mathematical society **68**(3), 337–404 (1950)
3. Besl, P.J., McKay, N.D.: Method for registration of 3-d shapes. In: Sensor fusion IV: control paradigms and data structures. vol. 1611, pp. 586–606. International Society for Optics and Photonics (1992)
4. Bietti, A., Mairal, J.: Group invariance, stability to deformations, and complexity of deep convolutional representations. The Journal of Machine Learning Research **20**(1), 876–924 (2019)
5. Bietti, A., Mialon, G., Chen, D., Mairal, J.: A kernel perspective for regularizing deep neural networks. In: International Conference on Machine Learning. pp. 664–674. PMLR (2019)
6. Bozorgtabar, B., Mahapatra, D., Thiran, J.P.: Exprada: Adversarial domain adaptation for facial expression analysis. Pattern Recognition **100**, 107111 (2020)
7. Chen, C., Fu, Z., Chen, Z., Jin, S., Cheng, Z., Jin, X., Hua, X.S.: Homm: Higher-order moment matching for unsupervised domain adaptation. In: Proceedings of the AAAI conference on artificial intelligence. vol. 34, pp. 3422–3429 (2020)
8. Chen, D., Jacob, L., Mairal, J.: Convolutional kernel networks for graph-structured data. In: International Conference on Machine Learning. pp. 1576–1586. PMLR (2020)
9. Chen, H., Manhardt, F., Navab, N., Busam, B.: Texpose: Neural texture learning for self-supervised 6d object pose estimation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 4841–4852 (2023)
10. Cheng, X., Xie, Y.: Neural tangent kernel maximum mean discrepancy. Advances in Neural Information Processing Systems **34**, 6658–6670 (2021)
11. Corcoran, P.: An end-to-end graph convolutional kernel support vector machine. Applied Network Science **5**(1), 1–15 (2020)
12. Cortes, C., Vapnik, V.: Support-vector networks. Machine learning **20**, 273–297 (1995)
13. Coumans, E., Bai, Y.: Pybullet, a python module for physics simulation for games, robotics and machine learning. `http://pybullet.org` (2016–2021)
14. Deng, J., Dong, W., Socher, R., Li, L., Kai Li, Li Fei-Fei: Imagenet: A large-scale hierarchical image database. In: 2009 IEEE Conference on Computer Vision and Pattern Recognition. pp. 248–255 (2009). `https://doi.org/10.1109/CVPR.2009.5206848`
15. Deng, X., Xiang, Y., Mousavian, A., Eppner, C., Bretl, T., Fox, D.: Self-supervised 6d object pose estimation for robot manipulation. In: 2020 IEEE International Conference on Robotics and Automation (ICRA). pp. 3665–3671. IEEE (2020)
16. Denninger, M., Sundermeyer, M., Winkelbauer, D., Olefir, D., Hodan, T., Zidan, Y., Elbadrawy, M., Knauer, M., Katam, H., Lodhi, A.: Blenderproc: Reducing the reality gap with photorealistic rendering. In: International Conference on Robotics: Sciene and Systems, RSS 2020 (2020)

17. Denninger, M., Winkelbauer, D., Sundermeyer, M., Boerdijk, W., Knauer, M., Strobl, K.H., Humt, M., Triebel, R.: Blenderproc2: A procedural pipeline for photorealistic rendering. Journal of Open Source Software **8**(82), 4901 (2023). https://doi.org/10.21105/joss.04901, https://doi.org/10.21105/joss.04901
18. Di, Y., Manhardt, F., Wang, G., Ji, X., Navab, N., Tombari, F.: So-pose: Exploiting self-occlusion for direct 6d pose estimation. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV). pp. 12396–12405 (October 2021)
19. Doumanoglou, A., Kouskouridas, R., Malassiotis, S., Kim, T.K.: Recovering 6d object pose and predicting next-best-view in the crowd. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 3583–3592 (2016)
20. Drost, B., Ulrich, M., Bergmann, P., Hartinger, P., Steger, C.: Introducing mvtec itodd-a dataset for 3d object recognition in industry. In: Proceedings of the IEEE international conference on computer vision workshops. pp. 2200–2208 (2017)
21. Gadwe, A., Ren, H.: Real-time 6dof pose estimation of endoscopic instruments using printable markers. IEEE Sensors Journal **19**(6), 2338–2346 (2018)
22. Gao, X., Hou, X., Tang, J., Cheng, H.: Complete solution classification for the perspective-three-point problem. IEEE Trans. Pattern Anal. Mach. Intell. **25**, 930–943 (2003)
23. Ghojogh, B., Ghodsi, A., Karray, F., Crowley, M.: Reproducing kernel hilbert space, mercer's theorem, eigenfunctions, nystr\" om method, and use of kernels in machine learning: Tutorial and survey. arXiv preprint arXiv:2106.08443 (2021)
24. Ghorbani, B., Mei, S., Misiakiewicz, T., Montanari, A.: When do neural networks outperform kernel methods? Advances in Neural Information Processing Systems **33**, 14820–14830 (2020)
25. Greene, N., Luo, W., Kazanzides, P.: dvpose: Automated data collection and dataset for 6d pose estimation of robotic surgical instruments. In: 2023 International Symposium on Medical Robotics (ISMR). pp. 1–7. IEEE (2023)
26. Gretton, A., Borgwardt, K., Rasch, M., Schölkopf, B., Smola, A.: A kernel method for the two-sample-problem. Advances in neural information processing systems **19** (2006)
27. Gretton, A., Borgwardt, K.M., Rasch, M.J., Schölkopf, B., Smola, A.: A kernel two-sample test. The Journal of Machine Learning Research **13**(1), 723–773 (2012)
28. Guo, S., Hu, Y., Alvarez, J.M., Salzmann, M.: Knowledge distillation for 6d pose estimation by aligning distributions of local predictions. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 18633–18642 (2023)
29. Hai, Y., Song, R., Li, J., Salzmann, M., Hu, Y.: Rigidity-aware detection for 6d object pose estimation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 8927–8936 (2023)
30. Haugaard, R.L., Buch, A.G.: Surfemb: Dense and continuous correspondence distributions for object pose estimation with learnt surface embeddings. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 6749–6758 (2022)
31. He, Y., Huang, H., Fan, H., Chen, Q., Sun, J.: Ffb6d: A full flow bidirectional fusion network for 6d pose estimation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 3003–3013 (2021)
32. He, Y., Sun, W., Huang, H., Liu, J., Fan, H., Sun, J.: 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)

33. He, Y., Wang, Y., Fan, H., Sun, J., Chen, Q.: Fs6d: Few-shot 6d pose estimation of novel objects. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 6814–6824 (2022)
34. Hinterstoisser, S., Lepetit, V., Ilic, S., Holzer, S., Bradski, G., Konolige, K., Navab, N.: Model based training, detection and pose estimation of texture-less 3d objects in heavily cluttered scenes. In: Asian conference on computer vision. pp. 548–562. Springer (2012)
35. Hodaň, T., Haluza, P., Obdržálek, Š., Matas, J., Lourakis, M., Zabulis, X.: T-LESS: An RGB-D dataset for 6D pose estimation of texture-less objects. IEEE Winter Conference on Applications of Computer Vision (WACV) (2017)
36. Hodan, T., Michel, F., Brachmann, E., Kehl, W., GlentBuch, A., Kraft, D., Drost, B., Vidal, J., Ihrke, S., Zabulis, X., et al.: Bop: Benchmark for 6d object pose estimation. In: Proceedings of the European conference on computer vision (ECCV). pp. 19–34 (2018)
37. Hodaň, T., Sundermeyer, M., Drost, B., Labbé, Y., Brachmann, E., Michel, F., Rother, C., Matas, J.: Bop challenge 2020 on 6d object localization. In: European Conference on Computer Vision. pp. 577–594. Springer (2020)
38. Horn, B.K., Hilden, H.M., Negahdaripour, S.: Closed-form solution of absolute orientation using orthonormal matrices. JOSA A **5**(7), 1127–1135 (1988)
39. Hu, Y., Fua, P., Salzmann, M.: Perspective flow aggregation for data-limited 6d object pose estimation. In: European Conference on Computer Vision. pp. 89–106. Springer (2022)
40. Kaskman, R., Zakharov, S., Shugurov, I., Ilic, S.: Homebreweddb: Rgb-d dataset for 6d pose estimation of 3d objects. In: Proceedings of the IEEE/CVF International Conference on Computer Vision Workshops. pp. 0–0 (2019)
41. Khosravi, M., Smith, R.S.: The existence and uniqueness of solutions for kernel-based system identification. Automatica **148**, 110728 (2023)
42. Kleeberger, K., Huber, M.F.: Single shot 6d object pose estimation. In: 2020 IEEE International Conference on Robotics and Automation (ICRA). pp. 6239–6245. IEEE (2020)
43. Kleeberger, K., Landgraf, C., Huber, M.F.: Large-scale 6d object pose estimation dataset for industrial bin-picking. In: 2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). pp. 2573–2578. IEEE (2019)
44. Kleeberger, K., Völk, M., Bormann, R., Huber, M.F.: Investigations on output parameterizations of neural networks for single shot 6d object pose estimation. In: 2021 IEEE International Conference on Robotics and Automation (ICRA). pp. 13916–13922. IEEE (2021)
45. Lee, T., Lee, B.U., Shin, I., Choe, J., Shin, U., Kweon, I.S., Yoon, K.J.: Udacope: unsupervised domain adaptation for category-level object pose estimation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 14891–14900 (2022)
46. Lepetit, V., Moreno-Noguer, F., Fua, P.: Ep n p: An accurate o (n) solution to the p n p problem. International journal of computer vision **81**, 155–166 (2009)
47. Li, Z., Hu, Y., Salzmann, M., Ji, X.: Sd-pose: Semantic decomposition for cross-domain 6d object pose estimation. In: Proceedings of the AAAI Conference on Artificial Intelligence. vol. 35, pp. 2020–2028 (2021)
48. Lin, J., Wei, Z., Ding, C., Jia, K.: Category-level 6d object pose and size estimation using self-supervised deep prior deformation networks. In: European Conference on Computer Vision. pp. 19–34. Springer (2022)

49. Liu, F., Xu, W., Lu, J., Zhang, G., Gretton, A., Sutherland, D.J.: Learning deep kernels for non-parametric two-sample tests. In: International conference on machine learning. pp. 6316–6326. PMLR (2020)
50. Luo, Y.W., Ren, C.X.: Conditional bures metric for domain adaptation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 13989–13998 (2021)
51. Ma, X., Wang, Z., Li, H., Zhang, P., Ouyang, W., Fan, X.: Accurate monocular 3d object detection via color-embedded 3d reconstruction for autonomous driving. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 6851–6860 (2019)
52. Mairal, J.: End-to-end kernel learning with supervised convolutional kernel networks. Advances in neural information processing systems **29** (2016)
53. Mairal, J., Koniusz, P., Harchaoui, Z., Schmid, C.: Convolutional kernel networks. Advances in neural information processing systems **27** (2014)
54. Manhardt, F., Arroyo, D.M., Rupprecht, C., Busam, B., Birdal, T., Navab, N., Tombari, F.: Explaining the ambiguity of object detection and 6d pose from visual data. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV) (October 2019)
55. Mildenhall, B., Srinivasan, P.P., Tancik, M., Barron, J.T., Ramamoorthi, R., Ng, R.: Nerf: Representing scenes as neural radiance fields for view synthesis. Communications of the ACM **65**(1), 99–106 (2021)
56. Misiakiewicz, T., Mei, S.: Learning with convolution and pooling operations in kernel methods. Advances in Neural Information Processing Systems **35**, 29014–29025 (2022)
57. Oberweger, M., Rad, M., Lepetit, V.: Making deep heatmaps robust to partial occlusions for 3d object pose estimation. In: Proceedings of the European Conference on Computer Vision (ECCV). pp. 119–134 (2018)
58. Pan, S.J., Tsang, I.W., Kwok, J.T., Yang, Q.: Domain adaptation via transfer component analysis. IEEE transactions on neural networks **22**(2), 199–210 (2010)
59. Park, K., Mousavian, A., Xiang, Y., Fox, D.: Latentfusion: End-to-end differentiable reconstruction and rendering for unseen object pose estimation. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 10710–10719 (2020)
60. Pearson, K.: Liii. on lines and planes of closest fit to systems of points in space. The London, Edinburgh, and Dublin philosophical magazine and journal of science **2**(11), 559–572 (1901)
61. Peng, S., Liu, Y., Huang, Q., Zhou, X., Bao, H.: Pvnet: Pixel-wise voting network for 6dof pose estimation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 4561–4570 (2019)
62. Pinheiro, P.O., Rostamzadeh, N., Ahn, S.: Domain-adaptive single-view 3d reconstruction. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 7638–7647 (2019)
63. Qi, C.R., Litany, O., He, K., Guibas, L.J.: Deep hough voting for 3d object detection in point clouds. In: proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 9277–9286 (2019)
64. Sejdinovic, D., Sriperumbudur, B., Gretton, A., Fukumizu, K.: Equivalence of distance-based and rkhs-based statistics in hypothesis testing. The annals of statistics pp. 2263–2291 (2013)
65. Shan, P., Bi, Y., Li, Z., Wang, Q., He, Z., Zhao, Y., Peng, S.: Unsupervised model adaptation for multivariate calibration by domain adaptation-regularization based

kernel partial least square. Spectrochimica Acta Part A: Molecular and Biomolecular Spectroscopy **292**, 122418 (2023)
66. Shugurov, I., Li, F., Busam, B., Ilic, S.: Osop: A multi-stage one shot object pose estimation framework. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 6835–6844 (2022)
67. Shugurov, I., Zakharov, S., Ilic, S.: Dpodv2: Dense correspondence-based 6 dof pose estimation. IEEE transactions on pattern analysis and machine intelligence **44**(11), 7417–7435 (2021)
68. Sock, J., Garcia-Hernando, G., Armagan, A., Kim, T.K.: Introducing pose consistency and warp-alignment for self-supervised 6d object pose estimation in color images. In: 2020 International Conference on 3D Vision (3DV). pp. 291–300. IEEE (2020)
69. Su, H., Qi, C.R., Li, Y., Guibas, L.J.: 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. pp. 2686–2694 (2015)
70. Su, Y., Saleh, M., Fetzer, T., Rambach, J., Navab, N., Busam, B., Stricker, D., Tombari, F.: Zebrapose: Coarse to fine surface encoding for 6dof object pose estimation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 6738–6748 (2022)
71. Sundermeyer, M., Durner, M., Puang, E.Y., Marton, Z.C., Vaskevicius, N., Arras, K.O., Triebel, R.: Multi-path learning for object pose estimation across domains. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 13916–13925 (2020)
72. Sundermeyer, M., Hodaň, T., Labbe, Y., Wang, G., Brachmann, E., Drost, B., Rother, C., Matas, J.: 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. pp. 2784–2793 (2023)
73. Sundermeyer, M., Marton, Z.C., Durner, M., Brucker, M., Triebel, R.: Implicit 3d orientation learning for 6d object detection from rgb images. In: Proceedings of the european conference on computer vision (ECCV). pp. 699–715 (2018)
74. Szafraniec, F.H.: The reproducing kernel hilbert space and its multiplication operators. Complex Analysis and Related Topics pp. 253–263 (2000)
75. Tan, T., Dong, Q.: Smoc-net: Leveraging camera pose for self-supervised monocular object pose estimation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 21307–21316 (2023)
76. Ullman, S.: The interpretation of structure from motion. Proceedings of the Royal Society of London. Series B. Biological Sciences **203**(1153), 405–426 (1979)
77. Venkateswara, H., Eusebio, J., Chakraborty, S., Panchanathan, S.: Deep hashing network for unsupervised domain adaptation. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 5018–5027 (2017)
78. Vert, J.P., Tsuda, K., Schölkopf, B.: A primer on kernel methods. Kernel methods in computational biology **47**, 35–70 (2004)
79. Wang, G., Manhardt, F., Liu, X., Ji, X., Tombari, F.: Occlusion-aware self-supervised monocular 6d object pose estimation. IEEE Transactions on Pattern Analysis and Machine Intelligence (2021)
80. Wang, G., Manhardt, F., Shao, J., Ji, X., Navab, N., Tombari, F.: Self6d: Self-supervised monocular 6d object pose estimation. In: European Conference on Computer Vision. pp. 108–125. Springer (2020)
81. Wu, Y., Greenspan, M.: Learning better keypoints for multi-object 6dof pose estimation. arXiv preprint arXiv:2308.07827 (2023)

82. Wu, Y., Javaheri, A., Zand, M., Greenspan, M.: Keypoint cascade voting for point cloud based 6dof pose estimation. In: 2022 International Conference on 3D Vision (3DV). pp. 176–186. IEEE (2022)
83. Wu, Y., Zand, M., Etemad, A., Greenspan, M.: Vote from the center: 6 dof pose estimation in rgb-d images by radial keypoint voting. In: European Conference on Computer Vision. pp. 335–352. Springer (2022)
84. Xiang, Y., Schmidt, T., Narayanan, V., Fox, D.: Posecnn: A convolutional neural network for 6d object pose estimation in cluttered scenes (2018)
85. Xiao, F., Liu, H., Lee, Y.J.: Identity from here, pose from there: Self-supervised disentanglement and generation of objects using unlabeled videos. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 7013–7022 (2019)
86. Xiao, Y., Qiu, X., Langlois, P.A., Aubry, M., Marlet, R.: Pose from shape: Deep pose estimation for arbitrary 3d objects. arXiv preprint arXiv:1906.05105 (2019)
87. Yang, H., Pavone, M.: Object pose estimation with statistical guarantees: Conformal keypoint detection and geometric uncertainty propagation. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 8947–8958 (2023)
88. Zhang, Y., David, P., Gong, B.: Curriculum domain adaptation for semantic segmentation of urban scenes. In: Proceedings of the IEEE international conference on computer vision. pp. 2020–2030 (2017)
89. Zhang, Z., Wang, M., Huang, Y., Nehorai, A.: Aligning infinite-dimensional covariance matrices in reproducing kernel hilbert spaces for domain adaptation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 3437–3445 (2018)
90. Zhou, J., Chen, K., Xu, L., Dou, Q., Qin, J.: Deep fusion transformer network with weighted vector-wise keypoints voting for robust 6d object pose estimation. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 13967–13977 (2023)