Pattern Recognition Letters

Joint Object Recognition and Pose Estimation using Multiple-anchor Triplet Learning of Canonical Plane

Shunsuke Yoneda[a,*], Kouki Ueno[a], Go Irie[b], Masashi Nishiyama[a], Yoshio Iwai[a]

[a]*Graduate School of Engineering, Tottori University, 101 Minami 4-chome, Koyama-cho, Tottori, 680-8550, Japan*
[b]*NTT Corporation, 1 Morinosato-wakamiya 3-chome, Atsugi, 243-0198, Japan*

ABSTRACT

Accurate object recognition and pose estimation models are essential for practical applications of robot arms, such as picking products on a shelf. Training such a model often requires a large-scale dataset with qualified labels for both object classes and pose parameters, and collecting accurate pose labels is particularly costly. A recent paper [28] proposed a triplet learning framework for joint object recognition and pose estimation without explicit pose labels by learning a spatial transformer network to estimate the pose difference of an input image from an anchor image depicting the same object in a reference pose. However, our analysis suggests that the pose estimation accuracy is severely degraded for input images with large pose differences. To address this problem, we propose a new learning approach called multiple-anchor triplet learning. The basic idea is to give dense reference poses by preparing multiple anchors so that there is at least one anchor image having a small pose difference to the input image. Our multiple-anchor triplet learning is an extension of the standard single-anchor triplet learning to the multiple-anchor case. Inspired by the idea of multiple instance learning, we introduce a selection layer that automatically chooses the best anchor for each input image and allows the network to be trained end-to-end to minimize triplet-based losses. Experiments with three benchmark datasets in product picking scenarios demonstrate that our method significantly outperforms existing methods in both object recognition and pose estimation accuracy.

1. Introduction

Real-world applications of robot arms, e.g., picking a product on a shelf, often require accurate object recognition and pose estimation. Many vision-based techniques have been developed to equip these functions targeting to realistic use cases such as warehouse automation [11, 17].

Following the success of deep learning in various image processing tasks, it has also become mainstream in both object recognition and pose estimation. Given the correlation between the two tasks, recent methods have been designed to perform both tasks simultaneously [27, 4]. These existing methods perform well when a sufficient amount of training data associated with accurate object class and pose parameter labels are available. However, collecting such a large-scale dataset with qualified labels is generally costly. It is even more so for pose parameters, due to the nature of the need for fine numerical labels. In this work, we aim to develop an efficient approach to learning a network for joint object recognition and pose estimation using a training dataset with no explicit pose parameter labels but with only object class labels.

A recent method [28] proposed a joint object recognition and pose estimation model that can be learned without explicit pose parameter labels, by combining the ideas of triplet learning [29, 24] and spatial transformer networks (STNs) [15]. The triplet learning and STN are typically used to improve object recognition accuracy. Triplet learning minimizes the feature distance between an anchor image (called anchor) and a positive image (called positive) of the same class as the anchor, and maximizes that between the anchor and a negative image (called negative) of a different class. Meanwhile, the original use of STN [15] is to apply a geometric transformation (e.g., 2D affine transformation) to the input image purely to improve recognition accuracy. Instead, [28] proposes integrating both

e-mail: nishiyama@tottori-u.ac.jp (Masashi Nishiyama)

into a unified framework to estimate the relative pose difference from the anchor to the positive. Considering that many products handled by the robot arm are locally planar (e.g., boxed foods), [28] trains the STN to match the appearance of a "dominant plane" of the object (called canonical plane) of the anchor and that of the positive within the triplet learning framework.

This elegant idea allows for pose estimation of nearly planar objects without labeling the pose parameters, and can also improve the accuracy of object recognition. However, our analysis (given later in Sec. 3) found that the training of the STN could fall to a poor local minimum depending on the choice of the anchor, resulting in severe degradation of pose estimation accuracy, especially for images with large pose differences.

In this paper, we propose a multiple-anchor triplet learning method to address this problem. Unlike the standard triple learning framework that uses only a single anchor, our approach allows us to have more than one anchor per triplet, giving denser reference poses to prevent pose estimation from being trapped in poor local minima. Moreover, we introduce a selection layer that automatically selects the best anchor from among candidate anchors to enable end-to-end learning with multiple-anchor triplet data. Experimental results with three publicly available datasets for object recognition in product picking scenarios show that our method reduces the pose estimation error while improving object recognition accuracy.

2. Related work

2.1. Object Recognition

Existing methods for object recognition can be categorized into a model-based approach and a data-driven approach. The model-based approach uses a classification model (classifier) to categorize an input image into a set of known object classes. The classifier is trained to minimize some loss function, e.g., softmax cross-entropy between prediction and the ground truth class label. A convolutional neural network such as ResNet [14] is typically used as the classifier. The model-based approach provides highly accurate classification as long as the possible object classes are prespecified. Utilizing this advantage of the model-based approach has been the mainstream for more advanced tasks such as object detection [19, 21].

The data-driven approach aims at learning distance metrics in a feature space and solving the classification task in the nearest neighbor manner. A feature extraction network is trained so that it minimizes the intra-class distances and maximizes the inter-class distances. Modern methods use convolutional neural networks for feature extraction, such as the siamese network [5, 1] and triplet network [29, 24]. More sophisticated versions have also been proposed in recent years [10, 31].

In this work, we consider the data-driven approach for object recognition. In a product sorting task in a warehouse, novel classes of objects can be frequently added to the gallery. In such a case, the model-based approach requires frequent retraining of the model, which is often time consuming, whereas the data-driven approach circumvents this. Our network is designed to extract a deep feature for an input image and is trained in the triplet learning framework. Moreover, the network is also trained so that it can estimate the pose of the object in the image by leveraging the output of the intermediate STN module.

2.2. Pose Estimation

Existing methods for pose estimation can be categorized into a keypoint-based approach and a learning-based approach. The keypoint-based approach is designed to extract discriminative three-dimensional points and match their local features in a point cloud space [8, 23]. The existing methods perform well when a sufficient number of points representing the shape of the object have been acquired. However, it is known that the performance decreases for objects with less discriminative shapes.

The learning-based approach is typically designed to match the appearance of the object in an RGB image space, which has attracted much attention recently [27, 4, 6]. Accurate pose estimation is possible when a large number of training samples with pose parameter labels are available. However, assigning accurate full three-dimensional pose parameters to each object is costly [4, 6]. Assigning ground truth three-dimensional bounding boxes to each object is still hard work [27].

Some recent studies have explored weakly supervised approaches to reduce the labeling cost, which is the most relevant type of methods to ours. The majority of the methods perform joint object recognition and pose estimation [16, 26, 18]. Kanezaki et al. [16] assumed that the object pose is captured by multiple views of objects and estimate the object category and pose using multiple images in the test stage. Sundermeyer et al. [26] and Li et al. [18] used three-dimensional models of objects for training to learn their shapes.

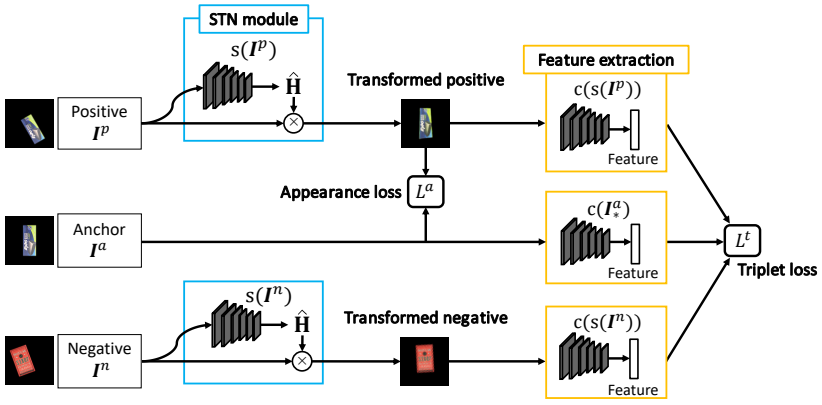
Unlike the existing methods, we instead focus on accomplishing this task without making such assumptions. We aim to estimate the parameters of linear transformations of pose changes. The network can be trained using only RGB images without requiring explicit pose labels. Furthermore, our model can estimate the object class and pose together from a single RGB image, which can be applicable to diverse robot arm systems in practical scenarios. As mentioned earlier, our method is based on [28]. However, [28] has a drawback that the pose estimation accuracy is severely reduced for images with large pose variations. In this paper, we address this problem by proposing a novel multiple-anchor triplet learning method.

3. Preliminary and Analysis

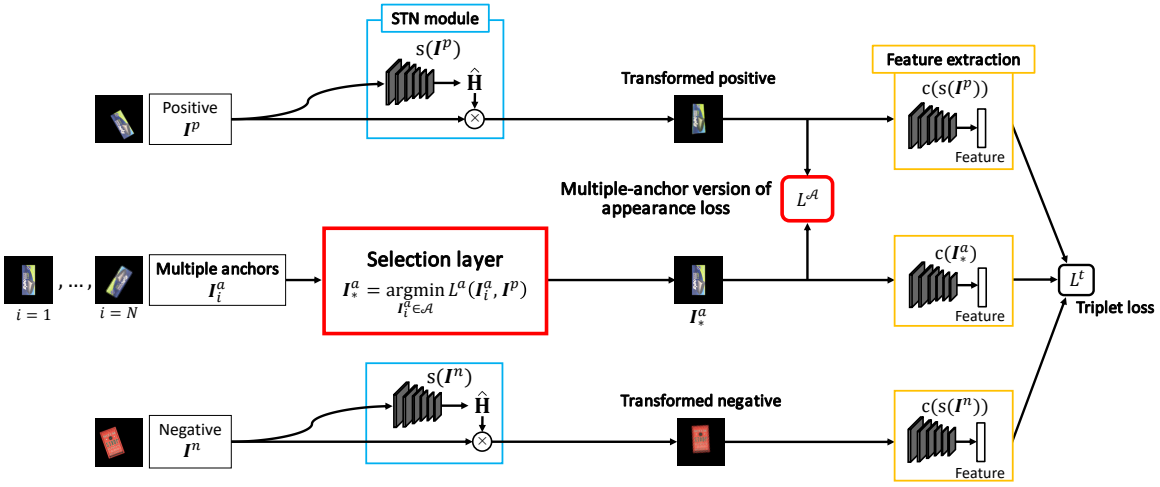
Before explaining the proposed method, we first introduce [28] which is the background framework of the proposed method. We also show our analysis to reveal that the pose estimation accuracy by [28] is degraded for an input image with a large pose difference from the anchor.

3.1. Overview of [28]

The overview of the network proposed in [28] is shown in Fig. 1(a). It mainly consists of two major trainable parts: the feature extraction network c( ) and the STN module s( ). As in the triplet learning framework [29, 24], it has three streams for the anchor, positive, and negative, and the parameters of c( ) and s( ) are shared over all the three streams. Note that the STN



(a) Architecture of [28]



(b) Architecture of ours

Fig. 1: Overview of the network architectures of (a) [28] and (b) ours. s( ) synthesizes an image by removing pose changes using the STN in terms of the homography transformation. c( ) extracts a feature using a convolutional neural network. Unlike (a) [28], (b) ours uses the selection layer to handle multiple anchors.

module s( ) is only set in the positive and negative streams, right after the input layer.

Suppose a collection of training triplets are available, where each consists of an anchor $\boldsymbol{I}^a$, positive $\boldsymbol{I}^p$, and negative $\boldsymbol{I}^n$. $\boldsymbol{I}^p$ contains the same object as the anchor $\boldsymbol{I}^a$ (but with a different pose), while $\boldsymbol{I}^n$ does a different object. Every image is assumed to be pre-processed to contain only a single foreground object, due to the fact that background models are often available in warehouse automation as described in [32], and thus can readily be removed and segmented by a standard foreground segmentation method. Given such a dataset, c( ) and s( ) are trained by minimizing the distance between the features of the anchor $\mathrm{c}(\boldsymbol{I}^a)$ and the "transformed" positive $\mathrm{c}(\mathrm{s}(\boldsymbol{I}^p))$ (by the STN module), while maximizing that between the features of the anchor $\mathrm{c}(\boldsymbol{I}^a)$ and the transformed negative $\mathrm{c}(\mathrm{s}(\boldsymbol{I}^n))$. The resulting features are expected to be consistent with the object classes.

At the same time, the STN module s( ) is trained to estimate the relative pose of the positive $\boldsymbol{I}^p$ from the anchor $\boldsymbol{I}^a$. Typical objects considered in warehouse automation scenarios can be well represented by a small number of "canonical planes," and an image of such an object is likely to capture at least one of these planes. Based on this idea, [28] first converts the full 3D pose estimation problem into a problem of 2D transformation estimation of the canonical plane, which can be directly handled by the STN module. Suppose an anchor depicts an object in a reference pose with respect to a canonical plane, and the same anchor is used for all the triplets for the same plane of the same object. In this case, the pose of the object in the positive can be predicted by estimating a 2D homography transformation matrix $\hat{\mathbf{H}}$ from the anchor to the positive. The following appearance loss L^a is introduced to enforce the positive to be always transformed to the anchor by the STN module s( ).

$$L^a(\boldsymbol{I}^a, \boldsymbol{I}^p) = \|\boldsymbol{I}^a - \mathrm{s}(\boldsymbol{I}^p)\|_1. \tag{1}$$

The loss returns a small value when $\boldsymbol{I}^a$ and $\boldsymbol{I}^p$ are similar after the transformation s( ) is applied. The minimization can be performed together with the feature learning in the triplet learning framework, without explicit pose parameter labels.

3.2. Analysis

We found that the pose estimation accuracy by this method [28] decreases rapidly when the pose difference between the positive $\boldsymbol{I}^p$ and the anchor $\boldsymbol{I}^a$ is large. This is because changes in appearance due to rotation often involve a certain level of ambiguity, leading to poor local minima for training

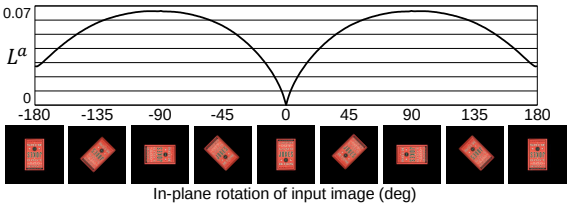


Fig. 2: Local minima of L^a with respect to in-plane rotations. Other than the global optimum at 0°, L_a has multiple poor local minima at ±180°.

the STN module s( ) using the appearance loss L^a. To be specific, we show in Fig. 2 the value of L^a when various in-plane rotations are applied to a book image. As can be seen in the example, when the image is rotated by 180°, the resulting image looks highly similar to its original (i.e., 0°). That suggests that, if the in-plane rotation angle of a positive $\boldsymbol{I}^p$ from the anchor $\boldsymbol{I}^a$ is outside of $[-90, 90]$ degrees, the appearance loss minimization is likely to be trapped at the points displaced by ±180°. Consequently, the pose estimation result may often deteriorate for an input image with a large pose difference from the anchor $\boldsymbol{I}^a$. To tackle this issue, we propose a multiple-anchor extension of the standard single-anchor triplet learning.

4. Multiple-anchor Triplet Learning

The idea behind our approach is to create a situation in which there is at least one anchor having a close angle for any positive by preparing multiple anchors with different rotation angles.

4.1. Multiple-anchor Triplet for Pose Estimation

Our triplet tuple consists of a set of N anchors denoted by $\mathcal{A} = \{\boldsymbol{I}_i^a\}_{i=1}^N$, one positive $\boldsymbol{I}^p$, and one negative $\boldsymbol{I}^n$. The set of the anchors $\mathcal{A}$ is prepared to represent the same canonical plane of an object from different rotation angles. While the simplest option is to use N images taken from N random viewpoints as $\mathcal{A}$, we adopted a more systematic and effective method. Specifically, we set up the camera so that its optical axis is perpendicular to the floor and place the object parallel to the floor surface. The optical axis of the camera is set so that it passes through the object's center of gravity. We then acquire N images for each canonical plane at a regular interval. As discussed in Sec. 5.7, this method gives better object recognition and pose estimation performance than the random acquisition case.

4.2. Appearance Loss for Multiple-anchor Case

Given the "bag of anchors" $\mathcal{A}$, inspired by the idea of multiple instance learning [2], we define our multiple-anchor version of the appearance loss $L^{\mathcal{A}}$ as:

$$L^{\mathcal{A}}(\mathcal{A}, \boldsymbol{I}^p) = \min_{I_i^a \in \mathcal{A}} L^a(\boldsymbol{I}_i^a, \boldsymbol{I}^p) = \min_{I_i^a \in \mathcal{A}} \|\boldsymbol{I}_i^a - \mathrm{s}(\boldsymbol{I}^p)\|_1. \quad (2)$$

By this, only the anchor image closest to the positive contributes to the final loss value, which mitigates the ambiguity over the angles.

4.3. Selection Layer

Our loss function Eq. 2 can be decomposed into two processes as:

$$\boldsymbol{I}_*^a = \operatorname*{argmin}_{I_i^a \in \mathcal{A}} L^a(\boldsymbol{I}_i^a, \boldsymbol{I}^p). \quad (3)$$

$$L^{\mathcal{A}}(\mathcal{A}, \boldsymbol{I}^p) = L^a(\boldsymbol{I}_*^a, \boldsymbol{I}^p). \quad (4)$$

This means that once the best anchor $\boldsymbol{I}_*^a$ in terms of the original (single-anchor version of) appearance loss L^a is chosen from $\mathcal{A}$ according to Eq. 3, the multiple-anchor version of the appearance loss $L^{\mathcal{A}}(\mathcal{A}, \boldsymbol{I}^p)$ turns into the original appearance loss $L^a(\boldsymbol{I}_*^a, \boldsymbol{I}^p)$.

Based on this fact, our multiple-anchor triplet learning can be implemented by just introducing a selection layer into the original triplet network. Our network architecture overall can be illustrated as in Fig. 1(b). It is the same as the original network Fig. 1(a) except for the selection layer that determines the best anchor according to Eq. 3. For training of the network, the gradient is computed by using only the $\boldsymbol{I}_*^a$ chosen by the selection layer, and the STN module can also be updated based on the homography transformation matrix $\hat{\mathbf{H}}$ from the selected anchor $\boldsymbol{I}_*^a$ to the given positive $\boldsymbol{I}^p$.

4.4. Training

Our network is trained to minimize the total loss function that can be expressed as

$$L(\boldsymbol{I}^{\mathcal{A}}, \boldsymbol{I}^p, \boldsymbol{I}^n) = \lambda L^{\mathcal{A}}(\mathcal{A}, \boldsymbol{I}^p) + L^t(\boldsymbol{I}_*^a, \boldsymbol{I}^p, \boldsymbol{I}^n) \quad (5)$$

$$= \lambda L^a(\boldsymbol{I}_*^a, \boldsymbol{I}^p) + L^t(\boldsymbol{I}_*^a, \boldsymbol{I}^p, \boldsymbol{I}^n), \quad (6)$$

where L^t is the standard triplet loss term which is for learning features for object recognition, and λ is a hyperparameter. The triplet loss L^t is defined as:

$$L^t(\boldsymbol{I}_*^a, \boldsymbol{I}^p, \boldsymbol{I}^n) = \max(\|\mathrm{c}(\boldsymbol{I}_*^a) - \mathrm{c}(\mathrm{s}(\boldsymbol{I}^p))\|_2^2 - \|\mathrm{c}(\boldsymbol{I}_*^a) - \mathrm{c}(\mathrm{s}(\boldsymbol{I}^n))\|_2^2 + m,\ 0), \quad (7)$$

where m is the margin. Note that the triplet loss term is computed only for the best anchor $\boldsymbol{I}_*^a$ selected by the selection layer, hence the standard form [29, 24] can be directly used. L^t returns a smaller value when the features of the selected anchor $\boldsymbol{I}_*^a$ and $\boldsymbol{I}^p$ are closer as well as those of $\boldsymbol{I}_*^a$ and $\boldsymbol{I}^n$ are farther.

4.5. Inference

Following [28], our method performs object recognition in a nearest neighbor classification manner. For each object, we store the set of N anchor images $\mathcal{A}$ of its canonical planes and their features in a database as the reference set. Given an input (query) image $\boldsymbol{I}^q$, our method first extracts its feature using the trained network, i.e., the anchor branch in Fig. 1(b), as $\mathrm{c}(\mathrm{s}(\boldsymbol{I}^q))$. Object recognition is done by finding the closest anchor image $\boldsymbol{I}_*^a$ from the database in terms of the feature distance, i.e., $\|\mathrm{c}(\boldsymbol{I}_*^a) - \mathrm{c}(\mathrm{s}(\boldsymbol{I}^q))\|$, and assigning the object class of the selected anchor image $\boldsymbol{I}_*^a$. The homography transformation matrix $\hat{\mathbf{H}}$ output by the STN module gives the estimated pose of the input $\boldsymbol{I}^q$ from the chosen anchor $\boldsymbol{I}_*^a$.

4.6. Extension to Other Types of Losses

Our multiple-anchor idea can be extended to several other types of loss functions. Here, we consider extending the noise contrastive estimation (NCE) loss to its multiple-anchor version. Many studies [9, 25, 30] have reported the effectiveness of NCE loss in various tasks, e.g., self-supervised learning and metric learning. Suppose given $\{\boldsymbol{I}^p, \boldsymbol{I}^a_*, \boldsymbol{I}^n_1, \ldots, \boldsymbol{I}^n_M\}$ with $\boldsymbol{I}^p$ as the positive, $\boldsymbol{I}^a_*$ as the anchor chosen by the selection layer from multiple anchors, and $\boldsymbol{I}^n_j (j = 1, \ldots, M)$ as the negatives, the total loss function is expressed as:

$$L(\boldsymbol{I}^p, \boldsymbol{I}^a_*, \boldsymbol{I}^n_1, \ldots, \boldsymbol{I}^n_M) = \lambda L^a(\boldsymbol{I}^p, \boldsymbol{I}^a_*) + L^N(\boldsymbol{I}^p, \boldsymbol{I}^a_*, \boldsymbol{I}^n_1, \ldots, \boldsymbol{I}^n_M), \quad (8)$$

where L^a is the selected appearance loss of Eq. 4, λ is a hyper-parameter, and L^N is the NCE loss. L^N is defined as:

$$L^N(\boldsymbol{I}^p, \boldsymbol{I}^a_*, \boldsymbol{I}^n_1, \ldots, \boldsymbol{I}^n_M) = -\mathbb{E}\left[\log \frac{\exp(\mathrm{sim}(\boldsymbol{I}^p, \boldsymbol{I}^a_*)/\tau)}{\exp(\mathrm{sim}(\boldsymbol{I}^p, \boldsymbol{I}^a_*)/\tau) + \sum_{j=1}^{M} \exp(\mathrm{sim}(\boldsymbol{I}^p, \boldsymbol{I}^n_j)/\tau)}\right], \quad (9)$$

where sim( ) is a cosine similarity function and τ is a temperature parameter. In our experiments in Sec. 5.10, we also evaluate the multiple-anchor NCE loss.

5. Experiments

5.1. Datasets

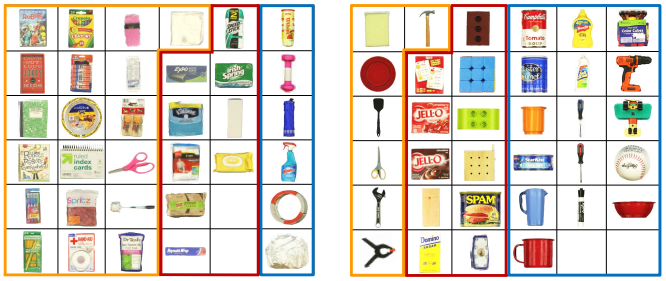
We evaluated object recognition accuracy and pose estimation error of our method by using three major datasets for product picking in warehouse scenarios: ARC dataset [3], YCB dataset [7], and APC dataset [22]. We used RGB images rendered from the object models available in these datasets so that we can have the ground truth pose parameters for evaluating pose estimation performance.

ARC dataset. Fig. 3(a) shows the examples of the images contained in ARC dataset. It contains 34 distinct objects: 19 are planar, nine are rectangular, and six are cylindrical. Following [28], the number of canonical planes are set to 2, 6, and 10, respectively. The resulting number of the anchor sets are 152, i.e., $19 \times 2 + 9 \times 6 + 6 \times 10$.

YCB dataset. Fig. 3(b) shows the examples of the images included in YCB dataset. The objects in YCB dataset have relatively complicated shapes, making it more challenging than ARC dataset. It contains 34 objects: seven are planar, 11 are rectangular, and 16 are cylindrical. We prepare the same number of the canonical planes for each object type as ARC dataset, leading to 240 sets of the anchors, i.e., $7 \times 2 + 11 \times 6 + 16 \times 10$.

APC dataset. Fig. 3(c) shows the examples of the images included in APC dataset. It contains 17 objects: six are planar, eight are rectangular, and three are cylindrical. We prepare the same number of the canonical planes for each object type as ARC dataset, leading to 90 sets of the anchors, i.e., $6 \times 2 + 8 \times 6 + 3 \times 10$.

We generated positives for triplet learning by applying a random homography matrix to the same object as in the anchor, and negatives by applying another random homography matrix to different objects. The parameters of the homography matrix are sampled in the range of $[-25, 25]$ pixels for translation, $[-180, 180]$ degrees for rotation, $[0.8, 1.2]$ for scale, and $[-0.5, 0.5]$ for the remaining parameters. We generated a set of anchors for each canonical plane by rotating it by $360/N$ degrees (We explain N in Sec. 4.1). For training, we used $20,000$ triplets for ARC dataset, $25,000$ for YCB dataset, and $10,000$ for APC dataset. The test sets were prepared so that they were disjoint from the training sets. Specifically, we stored $152 \times N$ anchors (i.e., reference images) for ARC dataset, $240 \times N$ for YCB dataset, and $90 \times N$ for APC dataset in the reference database. We randomly generated $1,000$ test input samples for each dataset in the same manner as the training samples. The size of each image was fixed to 100×100.



(a) ARC dataset (b) YCB dataset

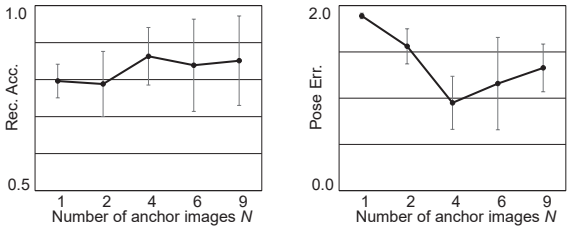
(c) APC dataset

Fig. 3: Object images from ARC dataset, YCB dataset, and APC dataset used in our experiments. The color frame indicates the type of the object shape (Orange: planar, Red: rectangular, Blue: cylindrical).

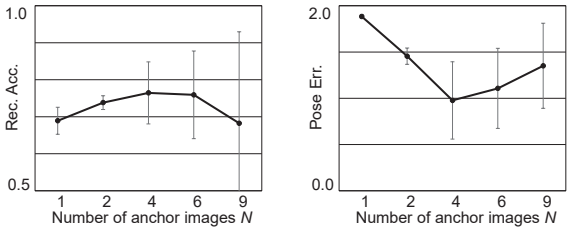
5.2. Experimental Setup

Implementation Details. Our feature extraction network is consisted of two convolution layers, one max pooling layer, and one global max pooling layer. The output feature is 128-dimensional. In the STN, we used a regressor with two convolution layers, two max pooling layers, and two fully connected layers. We trained the network using stochastic gradient descent with a learning rate of 0.01 and a momentum parameter of 0.9 for 100 epochs.

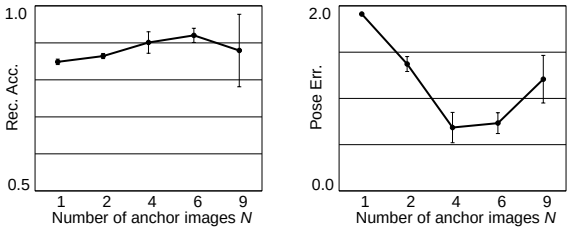
Performance Measures. We used the correct match rate to evaluate the object recognition accuracy and the Frobenius norm between the estimated homography matrix and the ground truth homography matrix to evaluate the pose estimation error. Note that we computed the pose estimation error if and only if the object classes of the test input samples were correctly recognized as those of the reference images. We report the average performance over the nine runs, each with the different training-testing splits.



(a) Rec. Acc. on ARC (b) Pose Err. on ARC



(c) Rec. Acc. on YCB (d) Pose Err. on YCB



(e) Rec. Acc. on APC (f) Pose Err. on APC

Fig. 4: Performance for different number of anchors N. (a) and (b) Results on ARC dataset. (c) and (d) Results on YCB dataset. (e) and (f) Results on APC datast.

5.3. Comparison with [28]

We first demonstrate the effectiveness of our multiple-anchor triplet learning by comparing with [28] based on the single-anchor triplet learning. We consistently set the balancing hyper-parameter $\lambda = 10$ in Eq. 5 as suggested in [28]. The results are shown in Table 1(a) and (f). We see that our method (f) outperformed [28] (a) in both recognition accuracy and pose estimation error with significant margins on all the datasets. It proved the strong advantage of our multiple-anchor triple learning in both object recognition and pose estimation. For further analysis, we evaluated the performance of our method while changing the number of anchors N for $N = 1, 2, 4, 6, 9$. Note that $N = 1$ corresponds to [28]. The results are shown in Fig. 4. The results suggest that the performance of multiple-anchor cases ($N > 1$) tend to outperform the single-anchor version ($N = 1$), which further emphasizes the effectiveness of our idea of using multiple anchors. We found that the larger N does not always improve performance. This may be because the number of training samples used per anchor may be reduced as the number of anchors grows.

Fig. 5 shows qualitative examples. We see that the images transformed by [28] have different appearances as the corresponding anchors. Meanwhile, those obtained by using our method are close to the anchors, even for objects with relatively complex contours (as the example in the second row) and input images with large rotation angles from the anchors (as the example in the third row). This clearly demonstrates that our multiple-anchor idea plays an important role in correctly estimating the relative pose changes.

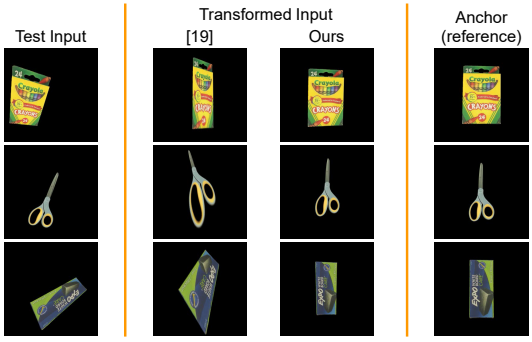


Fig. 5: Qualitative examples. In each row, from left to right, a test input image, test input images transformed by the STN module trained by [28] and ours, and the corresponding anchor (reference) image are shown, respectively.

5.4. Comparison with Triplet Learning [29] and STN Classifier [15]

We compared the performance of our method with that of the standard triplet learning method (TL) [29]. We only evaluated the object recognition accuracy because the standard triplet network was unable to estimate object poses. The results are shown in Table 1(b) and (f). Comparing the standard triplet learning method (b) with our method (f), we see that our network using the multiple-anchor triplet learning is superior to the simple triplet network.

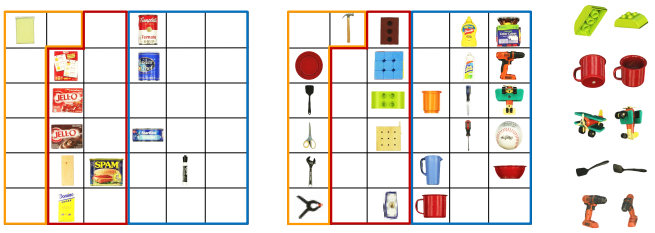
Next, we compared the performance of our method with that of the original STN [15]. For a fair comparison, the existing STN is constructed to have the same configuration as our network. That is, it consists of one STN module, two convolution layers, two max pooling layers, and two fully connected layers. Instead of triplet-based learning, the baseline network is trained with the cross-entropy loss. The results are shown in Table 1(c) and (f). Comparing the original STN (c) with our method (f), our approach of embedding an STN module in multiple-anchor triplet learning is superior to the original STN.

5.5. Evaluation of Object Recognition using Local Descriptors

Object recognition using SIFT+RANSAC and SuperPoint+RANSAC, i.e., image registration using RANSAC [13] with SIFT [20] local descriptors and SuperPoint [12] local descriptors, has been a strong approach to joint object recognition and pose estimation. We thus compare our method with them to analyze the superiority of our method. For SIFT feature extraction, we set the sigma of the Gaussian to 1.6, the contrast threshold to 0.004, and the edge threshold to $1,000$. For SuperPoint feature extraction, we set the radius of non-maximum suppression to 4, the threshold of keypoints to 0.005, and the maximum number of keypoints to 1024. The results are shown in Table 1(d) and (e). Our method (f) outperforms both SIFT+RANSAC (d) and SuperPoint+RANSAC (e) in terms of object recognition accuracy while achieving comparable performance in pose estimation.

Table 1: Performance of our method and the existing methods.

		ARC		YCB		APC	
		Rec. Acc.	Pose Err.	Rec. Acc.	Pose Err.	Rec. Acc.	Pose Err.
(a)	[28] (Single anchor, $N = 1$)	0.84±0.01	1.86±0.01	0.72±0.01	1.88±0.01	0.84±0.01	1.90±0.01
(b)	TL [29]	0.87±0.01	n/a	0.78±0.01	n/a	0.88±0.01	n/a
(c)	STN [15]	0.88±0.02	2.80±0.16	0.82±0.02	6.13±0.01	0.92±0.04	2.68±0.53
(d)	SIFT+RANSAC [20, 13]	0.82±0.02	**0.63±0.05**	0.80±0.03	0.64±0.19	0.87±0.01	0.50±0.02
(e)	SuperPoint+RANSAC [12, 13]	0.83±0.01	0.66±0.03	0.82±0.01	0.64±0.03	0.90±0.01	0.51±0.03
(f)	Ours (Multiple anchors, $\lambda = 10,\ N = 4$)	0.93±0.01	0.66±0.01	0.84±0.01	**0.60±0.01**	0.93±0.01	**0.46±0.01**
(g)	Ours (Multiple anchors, $\lambda = 0,\ N = 4$)	0.88±0.01	1.70±0.02	0.78±0.01	1.75±0.03	0.89±0.01	1.68±0.04
(h)	Ours (Multiple anchors, Random position, $\lambda = 10,\ N = 4$)	0.87±0.01	1.64±0.01	0.74±0.01	1.65±0.01	0.91±0.01	1.59±0.01
(i)	Ours (Multiple anchors, No STN module in negative, $\lambda = 10,\ N = 4$)	0.93±0.01	0.67±0.03	0.83±0.01	0.63±0.01	0.93±0.01	0.53±0.01
(j)	Ours (Multiple anchors, NCE loss, $\lambda = 10,\ N = 4$)	**0.94±0.01**	0.67±0.02	**0.89±0.01**	0.61±0.02	**0.97±0.01**	0.52±0.02



(a) (b) (c)

Fig. 6: (a) Planar objects, (b) non-planar objects, and (c) examples of non-planar objects from different views.

Table 2: Effect of planar objects and non-planar objects in YCB dataset.

		Rec. Acc.	Pose Err.
(a)	Ours (Planar)	0.91±0.02	0.57±0.03
(b)	[28] (Planar)	0.78±0.02	1.86±0.04
(c)	Ours (Non-planar)	0.81±0.02	0.68±0.05
(d)	[28] (Non-planar)	0.64±0.01	1.89±0.03

5.6. Performance without Appearance Loss

To further analyze the effectiveness of the appearance loss using multiple anchors Eq. 5, we evaluate the case when we set $\lambda = 0$. The results are shown in Table 1(g). Comparing the method without appearance loss (g) with the method with it (f), we confirm that our multiple-anchor version of the appearance loss contributes to both increasing object recognition accuracy and reducing the pose estimation error.

5.7. Comparison of Anchor Acquisition Methods

As described in Sec. 4.1, our method acquires anchor images of canonical planes in the fixed camera setting. To analyze the impact of the way to collect the anchors, we evaluated the performance with anchors acquired from random camera positions. The results when using the anchors from random position are shown in Table 1(h). Comparing with the results when using the fixed positions in (f), we confirm that the acquisition of representative canonical planes is effective in terms of improving the performance of object recognition and pose estimation.

5.8. Investigation of Effect of Non-planar Objects

We evaluated the performance of our method for planar objects and non-planar objects contained in YCB dataset. The planar objects mean simple shapes, i.e., planar, rectangular, and cylindrical, while non-planar objects mean complex shapes. Specifically, non-planar objects consist of not only canonical planes but also curved surfaces, projections, and concavities. We divided YCB dataset into 11 planar objects in Fig. 6(a) and 23 non-planar objects in (b). Examples of non-planar objects from different views are shown in (c). We used the same configuration of Table 1(f) except the datasets. We compared the performance between our method and [28].

The results for planar objects and non-planar objects in YCB dataset are shown in Table 2. For planar objects, our method (a) obtained higher performance than (b). For non-planar objects, our method (c) also obtained higher performance than (d). We believe that our multiple-anchor triple learning has the advantage for both planar objects and non-planar objects. However, the performance for the non-planar objects (c) is lower than that for planar objects (a). We consider that this is the limitation of our method because the homography estimation used in the STN module cannot sufficiently represent the complexity of the shape of non-planar objects. We need to develop further a method for pose estimation of complex shaped objects.

5.9. Performance without STN Module in Negative Stream

We evaluated the case when the STN module is removed from the negative stream of Fig. 1(b). We used the same configuration of Table 1(f) except the use of the STN module of the negative stream. The results without the STN module in the negative stream are shown in Table 1(i) and the results with it in (f). We see that the pose estimation error of (f) was slightly improved compared with the one of (i) in all the datasets. We consider that the STN module in the negative stream is meaningful to reducing the pose estimation error.

5.10. Performance with Multiple-anchor NCE Loss

We evaluated the performance of our method when using the multiple-anchor NCE loss L^N instead of the triplet loss L^t. We set $M = 64$, $\lambda = 10$, and $\tau = 0.07$. The same structures of the positive stream, the anchor stream, and the negative stream illustrated in Fig. 1(b) were used. The number of anchors in the selection layer was $N = 4$.

The results using NCE loss L^N are shown in Table 1(j). Comparing with the results of Triplet loss L^t (f), we see that the NCE loss increased the object recognition accuracy in all the datasets. However, the NCE loss did not contribute to reducing the pose estimation error in APC dataset. We consider that NCE loss is valuable in improving object recognition accuracy.

6. Conclusions

We proposed a multiple-anchor triplet learning method for joint object recognition and pose estimation without explicit pose parameter labels. Aiming at avoiding the problem of falling into poor local minima during pose estimation, our method uses multiple anchors to enable finer pose estimation. We showed that the proposed multiple-anchor triplet learning can be readily implemented by introducing a simple selection layer that determines the closest anchor to the positive image. We demonstrated that our method outperformed several existing methods with significant performance gain.

In future work, we will further evaluate our method on datasets of objects with various shapes and complex backgrounds. We are planning to expand the appearance loss term to metrics other than the L1 norm and develop a spatial transform module that can handle more transformations than just planar homography transformations.

References

[1] Ahmed, E., Jones, M., Marks, T.K., 2015. An improved deep learning architecture for person re-identification, in: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 3908–3916.

[2] Andrews, S., Tsochantaridis, I., Hofmann, T., 2002. Support vector machines for multiple-instance learning, in: Proceedings of Advances in Neural Information Processing Systems (NeurIPS), pp. 561–568.

[3] Araki, R., Yamashita, T., Fujiyoshi, H., 2018. ARC2017 RGB-D dataset for object detection and segmentation, in: Proceedings of Late Breaking Results Poster on International Conference on Robotics and Automation.

[4] Balntas, V., Doumanoglou, A., Sahin, C., Sock, J., Kouskouridas, R., Kim, T., 2017. Pose guided RGBD feature learning for 3D object pose estimation, in: Proceedings of IEEE International Conference on Computer Vision (ICCV), pp. 3876–3884.

[5] Bromley, J., Guyon, I., LeCun, Y., Sackinger, E., Shah, R., 1993. Signature verification using a "siamese" time delay neural network, in: Proceedings of Advances in Neural Information Processing Systems (NeurIPS), pp. 737–744.

[6] Bui, M., Zakharov, S., Albarqouni, S., Ilic, S., Navab, N., 2018. When regression meets manifold learning for object recognition and pose estimation, in: Proceedings of IEEE International Conference on Robotics and Automation (ICRA), pp. 1–7.

[7] Calli, B., Singh, A., Walsman, A., Srinivasa, S., Abbeel, P., Dollar, A.M., 2015. The YCB object and model set: Towards common benchmarks for manipulation research, in: Proceedings of International Conference on Advanced Robotics (ICAR), pp. 510–517.

[8] Chen, D.Y., Tian, X.P., Shen, Y.T., Ouhyoung, M., 2003. On Visual Similarity Based 3D Model Retrieval. International Journal of Computer Graphics Forum , 223–232.

[9] Chen, T., Kornblith, S., Norouzi, M., Hinton, G., 2020. A simple framework for contrastive learning of visual representations, in: Proceedings of Machine Learning Research (PMLR), pp. 1597–1607.

[10] Chen, W., Chen, X., Zhang, J., Huang, K., 2017. Beyond triplet loss: A deep quadruplet network for person re-identification. Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR) , 1320–1329.

[11] Correll, N., Bekris, K.E., Berenson, D., Brock, O., Causo, A., Hauser, K., Okada, K., Rodriguez, A., Romano, J.M., Wurman, P.R., 2018. Analysis and observations from the first Amazon Picking Challenge. IEEE Transactions on Automation Science and Engineering 15, 172–188.

[12] DeTone, D., Malisiewicz, T., Rabinovich, A., 2018. Superpoint: Self-supervised interest point detection and description, in: 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), pp. 337–33712.

[13] Fischler, M., Bolles, R., 1981. Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. Communications of the ACM 24, 381–395.

[14] He, K., Zhang, X., Ren, S., Sun, J., 2016. Deep residual learning for image recognition, in: Proceedings of Computer Vision and Pattern Recognition (CVPR), pp. 770–778.

[15] Jaderberg, M., Simonyan, K., Zisserman, A., Kavukcuoglu, K., 2015. Spatial transformer networks, in: Proceedings of Advances in Neural Information Processing Systems (NeurIPS), pp. 2017–2025.

[16] Kanezaki, A., Matsushita, Y., Nishida, Y., 2018. Rotationnet: Joint object categorization and pose estimation using multiviews from unsupervised viewpoints, in: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 5010–5019.

[17] Leitner, J., Tow, A.W., Sunderhauf, N., Dean, J.E., Durham, J.W., Cooper, M., Eich, M., Lehnert, C., Mangels, R., McCool, C., Kujala, P.T., Nicholson, L., Pham, T., Sergeant, J., Wu, L., Zhang, F., Upcroft, B., Corke, P., 2017. The ACRV picking benchmark: A robotic shelf picking benchmark to foster reproducible research, in: Proceedings of IEEE International Conference on Robotics and Automation (ICRA), pp. 4705–4712.

[18] Li, Y., Wang, G., Ji, X., Xiang, Y., Fox, D., 2018. Deepim: Deep iterative matching for 6D pose estimation, in: Proceedings of European Conference on Computer Vision (ECCV), pp. 695–711.

[19] Liu, W., Anguelov, D., Erhan, D., 2016. Ssd: Single shot multibox detector, in: Proceedings of European Conference on Computer Vision (ECCV), pp. 21–37.

[20] Lowe, D.G., 2004. Distinctive image features from scale-invariant keypoints. International Journal of Computer Vision (IJCV) 60, 91–110.

[21] Redmon, J., Farhadi, A., 2017. Yolo9000: Better, faster, stronger, in: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 6517–6525.

[22] Rennie, C., Shome, R., Bekris, K.E., Souza, A.F.D., 2016. A dataset for improved RGBD-based object detection and pose estimation for warehouse pick-and-place. IEEE Robotics and Automation Letters 1, 1179–1185.

[23] Rusu, R.B., Bradski, G., Thibaux, R., Hsu, J., 2010. Fast 3d recognition and pose using the viewpoint feature histogram, in: Proceedings of International Conference on Intelligent Robots and Systems (IROS), pp. 2155–2162.

[24] Schroff, F., Kalenichenko, D., Philbin, J., 2015. FaceNet: A unified embedding for face recognition and clustering, in: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 815–823.

[25] Sohn, K., 2016. Improved deep metric learning with multi-class n-pair loss objective, in: Proceedings of Advances in Neural Information Processing Systems (NeurIPS), pp. 1857–1865.

[26] Sundermeyer, M., Marton, Z.C., Durner, M., Brucker, M., Triebel, R., 2018. Implicit 3d orientation learning for 6D object detection from rgb images, in: Proceedings of European Conference on Computer Vision (ECCV), pp. 712–729.

[27] Tekin, B., Sinha, S., Fua, P., 2018. Real-time seamless single shot 6D object pose prediction, in: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 292–301.

[28] Ueno, K., Irie, G., Nishiyama, M., Iwai, Y., 2019. Weakly supervised triplet learning of canonical plane transformation for joint object recognition and pose estimation, in: Proceedings of International Conference on Image Processing (ICIP), pp. 2476–2480.

[29] Wang, J., Song, Y., Leung, T., Rosenberg, C., Wang, Philbin, J., Chen, B., Wu, Y., 2014. Learning fine-grained image similarity with deep ranking, in: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 1386–1393.

[30] Wu, Z., Xiong, Y., Yu, S., Lin, D., 2018. Unsupervised feature learning via non-parametric instance discrimination, in: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 3733–3742.

[31] Yu, B., Liu, T., Gong, M., Ding, C., Tao, D., 2018. Correcting the triplet selection bias for triplet loss, in: Proceedings of The European Conference on Computer Vision (ECCV), pp. 71–86.

[32] Zeng, A., Song, S., Yu, K.T., Donlon, E., Hogan, F.R., Bauza, M., Ma, D., Taylor, O., Liu, M., Romo, E., Fazeli, N., Alet, F., Dafle, N.C., Holladay, R., Morona, I., Nair, P.Q., Green, D., Taylor, I., Liu, W., Funkhouser, T., Rodriguez, A., 2019. Robotic pick-and-place of novel objects in clutter with multi-affordance grasping and cross-domain image matching. International Journal of Robotics Research , 1–16.