17th International Conference on Machine Vision Applications (MVA)
Fully Online, July 25–27, 2021.

Joint Learning of Object Detection and Pose Estimation using Augmented Autoencoder

Ryota Hayashi Asei Shimokura Takuya Matsumoto Norimichi Ukita
Toyota Technological Institute, Japan
ukita@toyota-ti.ac.jp

Abstract

This paper proposes a method for estimating the pose of a rigid object. While an appearance-based pose estimator requires a bounding box of each target object, an object detector is in general trained independently of the pose estimator. Recent pose estimators are robust to occlusion and image deviation, if the object region is correctly located by the detector. In reality, however, it is difficult to detect correct bounding-boxes, and such erroneous bounding-boxes make pose estimation inaccurate. Our proposed method integrates the object detector and the pose estimator so that they share feature maps and support to each other for improving the pose estimation accuracy. Experimental results demonstrate that the performance of our method is 7.54 times better than the SoTA pose estimation method.

1 Introduction

Object pose estimation enables various real-world systems, such as the automation of assembly robotics in factories. The performance of this object pose estimation is greatly degraded when a part of the object is not visible in the image. In the case of tools in a factory, a part of the object is obscured by hand grasping, and in the case of assembled parts in a storage case, a part of the object is obscured by mutual occlusion between parts. Augmented AutoEncoder (AAE) [1] is an image reconstruction model, which remains only the target object and removes all other foreground objects and background. That is, AAE can solve the occlusion problems in pose estimation. However, in AAE, we assume that the object region in the image is accurately located in advance by a detector that is independent of the pose estimator (Fig. 1 (a)). Therefore, the accuracy of pose estimation using AAE following object detection is also degraded because the occlusion prevents accurate object detection.

In this paper, as shown in Fig. 1 (b), we integrate a detector and a pose estimator, and learn them jointly by sharing a feature extractor. In this joint learning, the detection results of the object detector are directly end-to-end learned by the pose estimator to achieve "robust pose estimation against the detection errors of the detector" and "object detection to improve the accuracy of pose estimation".

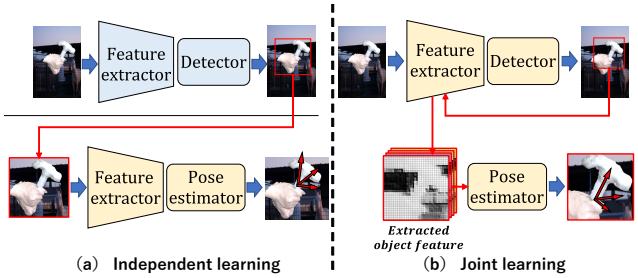

Figure 1. Comparison between (a) independent detection and pose estimation and (b) our proposed joint learning.

2 Related Work

2.1 Pose Estimation

Object pose estimation can be categorized to several approaches such as 3D model-based approaches [2, 3], appearance-based approaches [4, 5], and their combinations [6]. Deep networks further improve several key components (e.g., local feature detection and matching, appearance matching, and pose refinement [7, 8, 9, 10, 11, 12, 13, 14, 15]) for pose estimation. However, these methods are not sufficiently robust to object occlusions, background clutters, and illumination changes.

AAE [1] directly and explicitly addresses these issues. AAE extends the Denoising Autoencoder so that it is robust to arbitrary backgrounds and light sources using domain randomization [16]. AAE achieves accurate pose estimation in real images and also reduces the cost of manual annotations because the training data for AAE can be automatically generated with 3D CG models and arbitrary background images. However, AAE assumes that object-bounding boxes are known or detected in advance. In addition, if the detection accuracy is reduced by occlusion and this inaccurate detection is fed into AAE, the pose estimation accuracy of AAE is also reduced.

2.2 Object Detection

Convolutional Neural Networks (CNNs) enable accurate object detection such as Faster R-CNN [17],

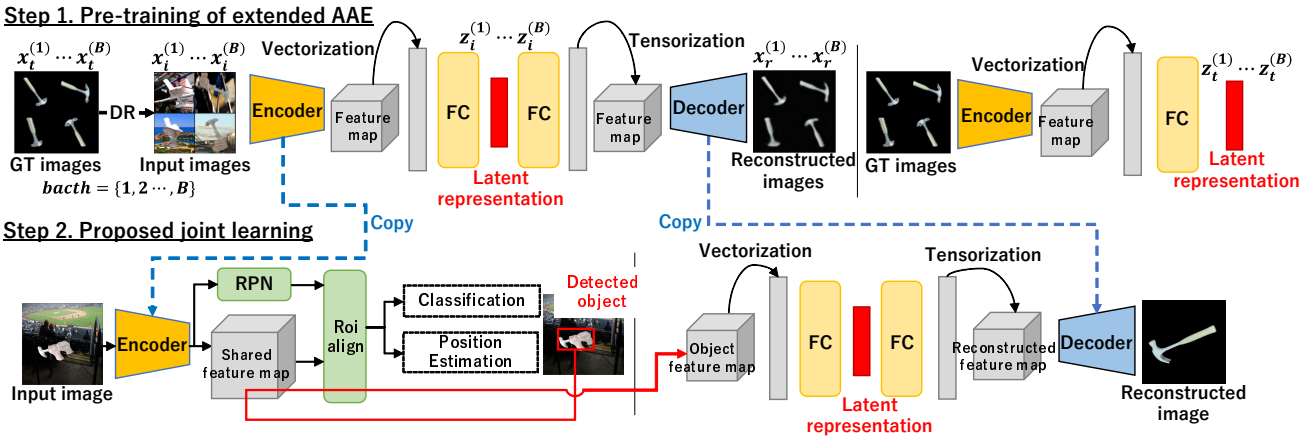

Figure 2. Overview of the proposed method. In Step 1, the Augmented AutoEncoder (AAE) is trained so that it can reconstruct the image of a target object with no other obstacles and background ("Reconstructed images" in the figure). The encoder and decoder in Step 1 are employed by the object detector and the pose estimator, respectively, in Step 2.

which enables end-to-end learning. Other object detectors are also highly-matured, such as Single Shot Multibox Detector (SSD) [18] and You Only Look Once (YOLO) [19]. Recent methods, including CenterNet [20] and CornerNet [21], detect each object not as a bounding box but as a set of points. These methods are extended in terms of a variety of aspects [22, 23, 24].

All of the above detectors use CNN to extract features for detection. Our proposed method is designed to work with any detector having such a feature extractor. In this paper, we use Faster R-CNN [17].

3 Proposed Method

The proposed method illustrated in Fig. 2 is divided into two steps. In Step 1, AAE alone is pre-trained. In Step 2, Faster R-CNN is trained jointly with AAE pre-trained in Step 1. Our propose method is possessed of the following advantages:

- Features extracted by AAE represent only a target object, and are useful for both detection and pose estimation.
- By propagating pose estimation error back to the detector, the detector is trained to predict bounding boxes that are easy for the pose estimation.
- By using detections (instead of ground-truth bounding-boxes) to train the pose estimator, the discrepancy between its training and estimation is eliminated. This makes the pose estimator robust to the deviation of detections.

3.1 Pre-training of Extended AAE

As shown in Step 1 of Fig. 2, AAE [1] is pre-trained. The encoder of AAE is shared by an object detector and a pose estimator in our joint learning network. However, in the original AAE, a small bounding-box is fed into the encoder. Therefore, a small encoder is sufficient. In our joint learning, on the other hand, a whole image where a target object is observed is fed into the encoder. Since the dimension of the object region in the feature map of the whole image is smaller than that of the object bounding box in the original AAE, this small feature map is insufficient for accurate pose estimation. To have sufficient features, we employ the VGG16, which is used also as a feature extractor in Faster R-CNN, for the encoder of AAE in our proposed method. The decoder is also designed as the inverse of VGG16 with deconvolution layers.

For training AAE, a 3D CG model of a target object is prepared in advance. This 3D model is rendered from arbitrary orientations with a black background ("GT images" in Step 1 of Fig. 2). These images are changed with arbitrary background and illuminations ("Input images" in Step 1) as with done in the original AAE. In our proposed method, furthermore, the target object is occluded for realistic scenarios. For example, tools in a factory are occluded by a user's hand. AAE consisting of the encoder, decoder and, fully-connected (FC) layers is trained so that its output ("Reconstructed images" in Step 1) is identical to its GT image.

The original AAE is trained with the MSE loss between the output image and its GT image (denoted by $\boldsymbol{x}_r$ and $\boldsymbol{x}_t$, respectively), as expressed by Eq. (1). n in Eq. (1) is the number of pixels in those images. In addition to this MSE loss, two loss functions are proposed for improving AAE. The angle loss expressed by Eq. (2) optimizes the latent representations so that, if the poses of two object images are similar, their latent representations should be close to each other. Assume

that we have image pairs ($\boldsymbol{x}_t^{(j)}$, $\boldsymbol{x}_t^{(k)}$) that the angles of objects are the closest among the batch. In Eq. (2), the latent representations corresponding to the input images $\boldsymbol{x}_t^{(j)}$ and $\boldsymbol{x}_t^{(k)}$ are $\boldsymbol{z}_i^{(j)}$ and $\boldsymbol{z}_i^{(k)}$, respectively. ${\theta_t}^{(j),(k)}$ is the relative angle between the poses of $\boldsymbol{x}_t^{(j)}$ and $\boldsymbol{x}_t^{(k)}$. The cos loss expressed by Eq. (3) optimizes the latent representations of the input image and its GT image (denoted by $\boldsymbol{z}_i$ and $\boldsymbol{z}_t$) so that they get close to each other. Our extended AAE is trained by the weighted sum of these three loss functions as expressed by Eq. (4):

$$L_{mse} = \frac{1}{n}||\boldsymbol{x}_i - \boldsymbol{x}_r||^2 \quad (1)$$

$$L_{angle} = ||\boldsymbol{z}_i^{(j)} - \boldsymbol{z}_i^{(k)}|| \times \exp(-{\theta_t}^{(j),(k)}) \quad (2)$$

$$L_{cos} = -\frac{\boldsymbol{z}_i \cdot \boldsymbol{z}_t}{||\boldsymbol{z}_i|| \cdot ||\boldsymbol{z}_t||} \quad (3)$$

$$L_{step1} = L_{mse} + \lambda_1 L_{angle} + \lambda_2 L_{cos} \quad (4)$$

where λ_1 and λ_2 denote the weights.

With AAE trained by Eq. (4), the latent representation is acquired from each input image. Before the pose estimation process, the latent representations of the target object with all possible poses are acquired. A set of these latent representations is regarded as a codebook.

In inference, the latent representation of each input image is compared with those in the codebook for finding the nearest one. The distance between two latent representations is expressed by the cosine similarity. Since each latent representation in the codebook is related with its object pose, the pose of the nearest one is considered to be that of the input object.

3.2 Proposed Joint Learning

The encoder and the decoder of AAE pre-trained in Step 1 are copied to the feature extractor of Faster R-CNN and the decoder of AAE, respectively, as illustrated by blue dashed arrows in Fig. 2.

First of all, an input image is fed into the feature extractor of Faster R-CNN. Through the region proposal network (RPN) and the RoI align [25], object bounding-boxes are detected. In a training image, only one target object is rendered and its bounding-box is known. With this assumption, we choose only one bounding-box having the largest Intersection of Union (IoU) with the ground-truth bounding-box. This selected bounding-box is passed to the following processes of Faster R-CNN for object detection. After this object detection, the detected bounding-box is feedbacked to the feature map in order to extract the feature of the detected object. This extracted feature is again fed into the RoI align via the red arrow in Fig. 2. Then, the feature is vectorized and fed into the FC layer in order to get the latent representation. The latent representation is fed into the next FC layer and the decoder to have the reconstructed image.

The reconstructed image and the latent representation are employed for training and detection as follows:

Training: Among L_{mse}, L_{angle}, and L_{cos}, only L_{mse} in Eq. (1) is used for training the whole network in Step 2. L_{angle} and L_{cos} are not used because they reduce the pose estimation accuracy in our experiments. In addition, L_{faster} proposed for training Faster R-CNN [17] is also computed with object detection results. The weighted sum of these two loss functions is used for an end-to-end joint learning of the whole network in Step 2.

$$L_{step2} = L_{faster} + \lambda L_{mse} \quad (5)$$

where λ is the weight.

Detection: The latent representation is used for retrieving the closest one from the codebook, as with in AAE, for pose estimation.

4 Experimental Results

4.1 Implementation Details and Evaluation

We experimented with a 3D CG model of a hammer, which is shown in Fig. 1.

For pre-training AAE (i.e., Step 1 shown in Fig. 2), the size of target, input, and output images is 128×128 pixels, as with the original AAE [1]. 100,000 input images were used for this training. $\lambda_1 = 10^{-6}$ and $\lambda_2 = 10^{-2}$ in Eq. 4. AdaBound [26] was used as an optimizer with a learning rate of $8e^{-6}$, a batch size = 128, and the 27 epochs.

On the other hand, the size of the input image for the detection network (i.e., Faster R-CNN) in Step 2 is 512×512 pixels. In each input image, the hammer is observed with around 128×128 pixels. 25,000 input images were trained for Step 2. In the same way as input images for training, 800 images were generated for evaluation. $\lambda = 1$ in Eq. (5). We used an SGD optimizer with an initial learning rate of 0.01, a batch size = 12, and the 30 epochs. The learning rate was reduced to $\frac{1}{10}$ of its current value at every eight epochs.

The pose estimation accuracy is evaluated by Visible Surface Discrepancy (VSD) [27] proposed for evaluating the pose of a symmetric textureless object. VSD becomes the minimum and maximum values, 0 and 1, if the estimated pose is completely identical to and different from its ground-truth pose, respectively. If VSD is below a threshold, the estimated pose can be considered to be correct. In our evaluation, the number of correctly-estimated poses is counted as an evaluation metric.

4.2 Experimental Results

Figure 3 shows experimental results. The horizontal and vertical axes of each graph indicate the threshold

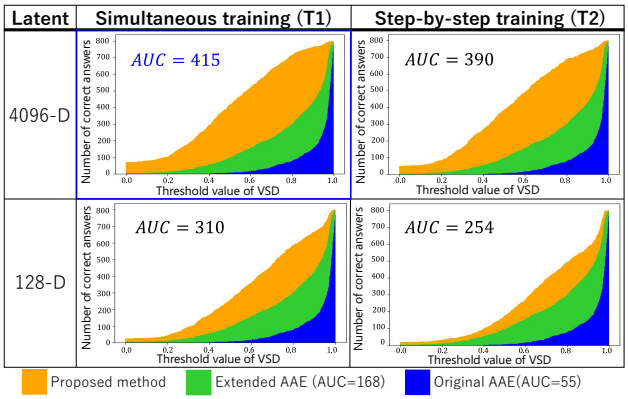

Figure 3. Evaluation results with VSD.

of VSD and the number of correctly-estimated poses, respectively. 2×2 results are shown in Fig. 3.

Row in Fig. 3 While the dimension of the latent representation in the original AAE is 128, this dimension is insufficient for representing the appearance variation of our dataset because of occlusions. For improving the representation ability, the dimension of the latent representation is increased to 4,096.

Column in Fig. 3 We trained the whole network for the joint learning simultaneously. The results of this training strategy (which is called T1) are shown in the left column in Fig. 3. On the other hand, joint learning sometimes fail because of imbalance between tasks (i.e., object detection and pose estimation in our problem). For maintaining the pose estimation ability of AAE in the joint learning, the weights of the encoder and decoder are fixed for training only RPN and FC layers in the first epoch. In the second epoch, only the encoder and decoder are trained. After the second epoch, all parameters are trained jointly. The results of this training strategy (which is called T2) are shown in the right column.

In Fig. 3, the results of our proposed method are shown with orange. Furthermore, two other methods are also evaluated. In these two methods, Faster R-CNN is trained independently of AAE. The bounding box detected by this Faster R-CNN is extracted and resized to 128×128 pixels. This resized bounding box is fed into our extended AAE and orignal AAE. The results of our extended AAE and the original AAE are shown in green and blue graphs, respectively.

As an evaluation metric, the Area Under Curve (AUC) of each graph is computed. AUC of our proposed method (i.e., orange graphs) is shown in each graph of Fig. 3. Compared to the independent detection and pose estimation approaches, our joint learning

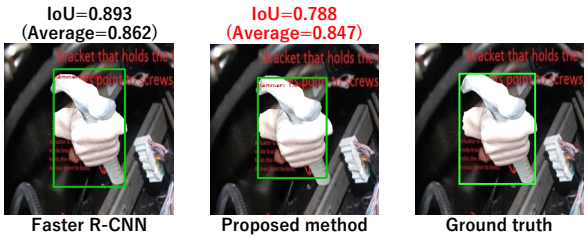

Figure 4. Examples of object detection results.

method gets higher scores in all of the four graphs. Since our extended AAE is superior to the original AAE, our proposed method is improved by the joint learning as well as by two additional loss functions (i.e., Eqs. (2) and (3)). Among the four variants, our proposed method gets the best score (AUC = 415) with the T1 training strategy and the 4096-D latent space. In this condition, AUC of the proposed method is 7.54 times and 2.47 times larger than the original AAE and the extended AAE, respectively.

The results of object detection and their IoU scores are shown in Fig. 4. Figure 4 shows IoU of this example as well as the mean IoU of all test images. In terms of IoU, Faster R-CNN trained independently of AAE (i.e, the leftmost result) is superior to the one trained jointly with AAE (i.e., the middle result). This result can be naturally interpreted that the former is better because its feature map is optimized only for the detection task. If the detection result of the Faster R-CNN is always identical to its ground-truth (i.e., IoU = 1.0), independent learning of detector and pose estimator is a good choice. However, it is practically impossible. Therefore, if the pose estimator is trained with the ground-truth bounding box, discrepancy between the ground-truth given in the training stage and its detection result given in the inference stage decreases the pose estimation performance. While the detection IoU is worse, our joint learning method appropriately learns object poses detected by Faster R-CNN.

5 Concluding Remarks

This paper proposes a joint learning approach for object detection and pose estimation. The effectiveness of the proposed approach is demonstrated with comparative experiments: 7.54 times better than the original AAE in terms of AUC of VSDs.

Future work includes experiments with real images and more objects (e.g., T-LESS dataset [28]) for validating the general-purpose effectiveness of the proposed method by comparison with other SOTA methods.

References

[1] Martin Sundermeyer, Zoltan-Csaba Marton, Maximilian Durner, Manuel Brucker, and Rudolph Triebel. Implicit 3d orientation learning for 6d object detection from RGB images. In *ECCV*, 2018.

[2] Andrew Edie Johnson and Martial Hebert. Using spin images for efficient object recognition in cluttered 3d scenes. *IEEE*, 21:433–449, 1999.

[3] Maxim Tatarchenko, Jaesik Park, Vladlen Koltun, and Qian-Yi Zhou. Tangent convolutions for dense prediction in 3d. In *CVPR*, 2018.

[4] Paul Wohlhart and Vincent Lepetit. Learning descriptors for object recognition and 3d pose estimation. In *CVPR*, 2015.

[5] Vassileios Balntas, Andreas Doumanoglou, Caner Sahin, Juil Sock, Rigas Kouskouridas, and Tae-Kyun Kim. Pose guided RGBD feature learning for 3d object pose estimation. In *ICCV*, 2017.

[6] Paul Wohlhart and Vincent Lepetit. Learning descriptors for object recognition and 3d pose estimation. In *CVPR*, 2015.

[7] Yi Li, Gu Wang, Xiangyang Ji, Yu Xiang, and Dieter Fox. Deepim: Deep iterative matching for 6d pose estimation. In *ECCV*, 2018.

[8] He Wang, Srinath Sridhar, Jingwei Huang, Julien Valentin, Shuran Song, and Leonidas John Guibas. Normalized object coordinate space for category-level 6d object pose and size estimation. In *CVPR*, 2019.

[9] Dylan Campbell, Liu Liu, and Stephen Gould. Solving the blind perspective-n-point problem end-to-end with robust differentiable geometric optimization. In *ECCV*, 2020.

[10] Yinlin Hu, Joachim Hugonot, Pascal Fua, and Mathieu Salzmann. Segmentation-driven 6d object pose estimation. In *CVPR*, 2019.

[11] Bo Chen, Álvaro Parra, Jiewei Cao, Nan Li, and Tat-Jun Chin. End-to-end learnable geometric vision by backpropagating pnp optimization. In *CVPR*, 2020.

[12] Kiru Park, Timothy Patten, and Markus Vincze. Pix2 pose: Pixel-wise coordinate regression of objects for 6d pose estimation. In *ICCV*, 2019.

[13] Sida Peng, Yuan Liu, Qixing Huang, Xiaowei Zhou, and Hujun Bao. Pvnet: Pixel-wise voting network for 6dof pose estimation. In *CVPR*, 2019.

[14] Yinlin Hu, Pascal Fua, Wei Wang, and Mathieu Salzmann. Single-stage 6d object pose estimation. In *CVPR*, 2020.

[15] Sergey Zakharov, Ivan Shugurov, and Slobodan Ilic. DPOD: 6d pose object detector and refiner. In *ICCV*, 2019.

[16] Joshua Tobin, Rachel Fong, Alex Ray, Jonas Schneider, Wojciech Zaremba, and Pieter Abbeel. Domain randomization for transferring deep neural networks from simulation to the real world. In *IROS*, 2017.

[17] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster R-CNN: towards real-time object detection with region proposal networks. In *NIPS*, 2015.

[18] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott E. Reed, Cheng-Yang Fu, and Alexander C. Berg. SSD: single shot multibox detector. In *ECCV*, 2016.

[19] Joseph Redmon, Santosh Kumar Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In *CVPR*, 2016.

[20] Xingyi Zhou, Dequan Wang, and Philipp Krähenbühl. Objects as points. In *CVPR*, 2019.

[21] Hei Law and Jia Deng. Cornernet: Detecting objects as paired keypoints. In *ECCV*, 2018.

[22] Golnaz Ghiasi, Tsung-Yi Lin, and Quoc Viet Le. Nas-fpn: Learning scalable feature pyramid architecture for object detection. In *CVPR*, 2019.

[23] Xingyi Zhou, Jiacheng Zhuo, and Philipp Krahenbuhl. Bottom-up object detection by grouping extreme and center points. In *CVPR*, 2019.

[24] Mingxing Tan, Ruoming Pang, and Quoc Viet Le. Efficientdet: Scalable and efficient object detection. In *CVPR*, 2020.

[25] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask R-CNN. In *ICCV*, 2017.

[26] Liangchen Luo, Yuanhao Xiong, Yan Liu, and Xu Sun. Adaptive gradient methods with dynamic bound of learning rate. In *ICLR*, 2019.

[27] Tomas Hodan, Jiri Matas, and Stepán Obdrzálek. On evaluation of 6d object pose estimation. In *ECCV*, 2016.

[28] Tomas Hodan, Pavel Haluza, Stepán Obdrzálek, Jiri Matas, Manolis Lourakis, and Xenophon Zabulis. T-LESS: an RGB-D dataset for 6d pose estimation of texture-less objects. In *WACV*, 2017.