Deep Multi-State Object Pose Estimation for Augmented Reality Assembly

Yongzhi Su* [1,2] Jason Rambach† [1] Nareg Minaskan[1] Paul Lesur[1] Alain Pagani[1]
Didier Stricker[1,2]

[1] German Research Center for Artificial Intelligence(DFKI), Kaiserslautern, Germany
[2] TU Kaiserslautern, Kaiserslautern, Germany

ABSTRACT

Neural network machine learning approaches are widely used for object classification or detection problems with significant success. A similar problem with specific constraints and challenges is object state estimation, dealing with objects that consist of several removable or adjustable parts. A system that can detect the current state of such objects from camera images can be of great importance for Augmented Reality(AR) or robotic assembly and maintenance applications. In this work, we present a CNN that is able to detect and regress the pose of an object in multiple states. We then show how the output of this network can be used in an automatically generated AR scenario that provides step-by-step guidance to the user in assembling an object consisting of multiple components.

Index Terms: Computing methodologies—Machine learning—Machine learning approaches—Neural networks; Computing methodologies—Artificial intelligence—Computer vision—Computer vision tasks; Human-centered computing—Human computer interaction (HCI)—Interaction paradigms—Mixed/augmented reality;

1 INTRODUCTION

The detection of objects in images and their classification into one of a number of predefined object classes has been one of the most researched topics in computer vision for several decades. Traditionally, handcrafted features such as SIFT or HOG [13] were used to train different types of classifiers for the task. The field was revolutionized by Deep Learning and Convolutional Neural Networks (CNNs) after AlexNet won the ImageNet challenge in 2012 [8]. Further improvements followed, to the point that neural networks have achieved human-like accuracy in detection of several types of objects [12, 21].

However, an important problem in this category is detecting an object in different states. Examples thereof are objects with adjustable or removable parts or objects comprised of several components. Such objects are very often encountered in real-life situations at home or industrial environments and are of great interest for Augmented Reality (AR) as well as robotic applications. Automatic recognition of object state along with their pose directly from camera images can enable AR applications that assist in the assembly/dis-assembly and maintenance of these objects while increasing safety and human error detection and prevention. Even though the practical importance of object state estimation is obvious, there is little existing work addressing it. This can be partially due to its challenging nature, since it requires correct classification between classes of very high similarity, while ambiguities between states may even exist based on the current viewing point. Thus, for feature-based approaches a difficult challenge are the repeating features between classes. Using a learning approach the significance of specific features to the classification can be learned rather than explicitly defined.

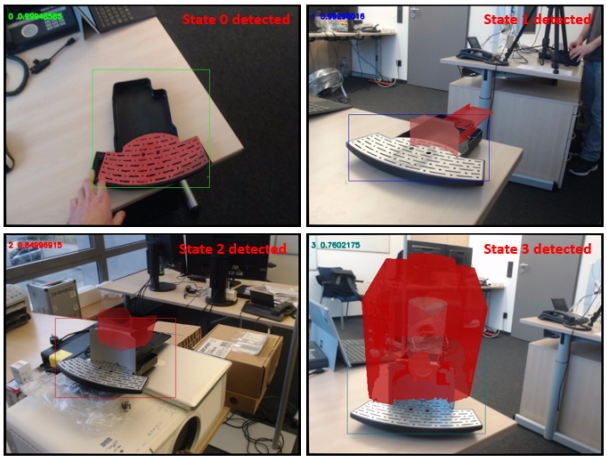

Figure 1: Proposed AR Assembly application using object state and pose estimation. The missing component to move to the next assembly state is visualized.

In this paper, we propose a CNN inspired by the Backbone TridentNet [9] and Structure Faster R-CNN [23] to estimate the current state of an object and its pose from a single RGB image. Befit from the dilated convolution with different receptive fields, the CNN can deal objects at different scales, without a large increase of the computational cost and the size of the neural network. We use 3D reconstructed models of the objects in different states and train our CNN on exclusively synthetic datasets. Based on a state evolution diagram of the object and the output of the trained neural network we are able to generate an AR application that detects and tracks the current state of the object and visualizes the next missing component in the assembly procedure. The main contributions of this work are:

- The first system to estimate the state and pose of a multi-state object with deep learning to the best of our knowledge.
- A combination of two CNN architectures into a new network with a state estimation and a pose estimation branch trained exclusively on synthetic images.
- The integration of the CNN state and pose information into an AR assembly instruction application with reduced content generation effort.

The paper is organized as follows: We first give an overview of the related work in sec. 2. The object state graph defining the possible states is introduced in sec. 3. Subsequently, we present our proposed method in sec 4 and its quantitative evaluation in sec. 6. We also propose an AR Application in sec. 5 as an example use case of this work.

*e-mail: yongzhi.su@dfki.de
†e-mail: jason_raphael.rambach@dfki.de

2 Related Work

In this section we will look at related work in the problem of object detection, classification and pose estimation. Subsequently, we give a short overview of existing approaches for AR assembly and maintenance applications.

2.1 Object Detection and Classification

Object detection is one of the most important problems in the research of computer vision. Classical detectors extract and match hand-crafted features. Although some features have been well-designed to be stable to challenges like scale or lighting variations [13], they still have limited performance with textureless objects. After CNNs [8] have achieved remarkable result in the image classification, they have also been applied in the object detection methods as the backbone to extract features from the image. Two main types of structures for CNN based detectors exist. The one-stage detectors like YOLO [21] and SSD [12], benefit from the straightforward structure, can reach a very high frame rate of more than 50 fps. They predict the bounding box and class probabilities with the predefined anchors on the image. The two-stage detector like Faster R-CNN [23] and its following works [3] contain a Region Proposal Network (RPN) to generate potential bounding boxes based on the feature map. Relying on the proposed bounding box, a fine regression of the position and the object classification follows.

Detecting objects in various scale requires the detector to contain various receptive fields thus it is hard to detect very large and very small objects with the same model. The first idea to address this issue is the use of multi-scale image pyramids. However, the computational cost is proportional to the range of scaling. To reduce the reuse of resources, the detection is performed in the feature maps with different downsampling rates in the work of SSD [12]. Furthermore, the Feature Pyramid Network(FPN) [11] applies a top-down pathway and lateral connections in addition, to keep semantic information during the increase of downsampling in the CNNs. Although the low-level feature layers have a small receptive field for the detection of smaller objects, it also has less representational power than high-level feature layers. Detection of objects in different scales of feature layers is still suboptimal.

Another idea is leveraging the dilated convolution, which enables a large sampling field in the CNN layer without increasing the number of parameters and the computational cost [9]. Following the idea proposed in SNIP [24], training and testing of the detector can be scale aware to ensure detecting of the object with the suitable receptive field.

2.2 Object Pose Estimation

The object detection problem focuses on the presence of the object and its location in the 2D image. However, in the AR and robotic operation applications, the 6DoF pose of the object instance with a known 3D model (CAD model or reconstructed model) is of greater significance.

Given the bounding box of the object, the 2D detection can be extended to 6DoF pose estimation. Generally, the object rotation in most works is obtained by processing the image in the bounding box. In SSD-6D [6], the rotation is classified into the predefined discrete viewpoints. It can also be directly regressed as in [30]. A novel encoder-decoder structure was also proposed to determine the rotation relying on the object representation in the neural network [25]. The translation is treated separately as depth and offset in the x-, y-axis. The depth information can be obtained by comparing the size of the predicted bounding box with the bounding box in a known distance [6] , or regressed from the neural network [25, 30]. The offset in x-, y-axis can be calculated by scaling the object 2D center in the image with the predicted depth [6, 25, 30].

The work of PoseNet [7] was the first to regress poses directly for relocalization using a CNN. In the work of [18] a similar network was applied for regression of object poses from images. The pencil filter was used as a domain adaptation technique to enable training exclusively with synthetic images. However, this work only estimates the pose and does not explicitly detect the object.

2.3 Augmented Reality Assembly and Maintenance

Assembly or maintenance instructions for industrial or household devices has historically been one of the most coveted use cases for Augmented Reality. The reasons for this are the clear benefits that such an intelligent guidance system can have over the use of printed manuals that draw the attention away from the object at hand and are more error-prone [4]. Initial work exploring these ideas was based on specifically selected objects and experimental setups for the task [22]. A large amount of work followed, often being object specific and relying on tracking techniques such as fiducial markers [17]. An extensive overview of the topic can be found in [29]. The work of [31] takes an approach that is similar to ours in defining an object state graph leading to the assembly, however using markers on each object component for tracking.

Many related publications present studies that compare AR-assisted assembly instructions to traditional printed manuals in order to examine the benefits of using AR mainly in terms of required time and error avoidance [26]. Results of such studies tend to be inconclusive, however an important depending factor that is often not considered is the actual quality of the AR application used in the experiments in terms of tracking and visual element quality. The latter is studied in [27]. The complexity of the task is another important factor, however most of the existing work is focused on less challenging tasks. In [26] the human factors in such a comparison are in focus.

In [16], the main addressed problem is that of automated knowledge capturing from a video example of the performed task. Along with other existing systems, tracking and 3D registration of augmentations is not in focus. The same holds true for many examples of remote expert maintenance support AR applications. A 3D object tracking system is deployed in [19] while in [20] objects that can directly be connected to in order to provide live tracking and status information are proposed. The work presented here could also be combined with these approaches in a greater AR tracking/assembly/maintenance framework. Our paper is more focused on providing a crucial novel technology that we consider necessary for many AR assistance systems rather than evaluating the performance gains from the use of AR compared to traditional assistance methods.

3 Object State Graph

An object state graph describing single components and their possible combinations leading to the assembled object is required in order to train the network and generate an AR application that guides the assembly. An example of such a graph for our selected test object (coffee machine) is given in Fig. 2. The object components are shown on the top row followed by the possible combinations leading to new states and finally to the assembled machine. As can be seen it is possible that there are multiple options in the order of combinations of components to reach the final assembled device. Reconstructed or CAD models of all single components as well as all states are required in order to generate the synthetic training set and to create augmentations of the following states in order to visually guide the user to assemble the object.

4 Object State and Pose Estimation

Dealing with objects in various scales has been a hard problem for many computer vision tasks. This is especially of importance for the AR assembly case since during the assembly of parts, initial states are typically smaller than the final state.

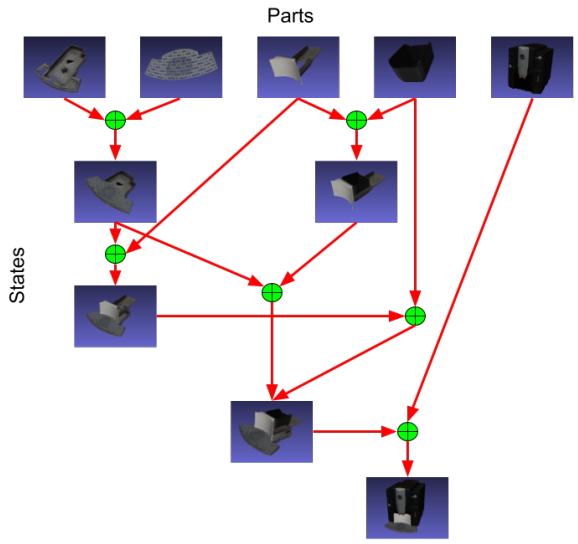

Figure 2: Object State Graph example. Single components can be combined in different ways to assemble the object. The example of a coffee machine is shown here.

Another important point is that objects with the same rotation but a different translation have a different appearance. Decoupling the rotation and translation prediction completely can thus be problematic. Estimating the rotation directly from the bounding box without other spatial information can be imprecise. Besides, the estimation of translation relying on the object 2D center requires the original of the object's local coordinate located in its geometric center. In the frame of this work, we estimate the 6DoF pose of an object in multi-state, which means the object has no fixed geometric center(see Fig. 3). We choose to decouple the pose estimation from the detected bounding box in this work.

Following the work of TridentNet [9], which implemented the dilated convolution to deal with the objects scaling, we added a branch in the Faster R-CNN architecture for pose estimation. In the section, we will first give a brief introduction of the multi-branch block in the TridentNet and then present our modified Faster R-CNN architecture as well as the used loss function.

4.1 CNN Architecture

The architecture of the CNN is depicted in Fig. 4. At first, a backbone with trident block is used to extract features from the input image. The trident block contains multiple dilated convolutions with various dilation rates. Each dilated convolution independently extracts the feature map. Those features maps are stacked together as the output. The RPN proposes potential foreground region and resizes all the regions of interest (ROI) into a fixed size, which is further processed for the classification of the object states and the bounding box regression. At the same time, the extracted feature map from the backbone are passed into another branch for the 6DoF pose estimation. The translation is regressed in the form of the offset in x-y-z-axis, and the rotation in the form of the quaternion, respectively. Similar to the work of [28], we added a confidence rate to reflect how certain the network is about the pose prediction. The use of the trident block enables a different receptive field in one feature layer. In the proposed architecture, this advantage is not only used for state detection, but also for the pose estimation. Since the 6DoF pose estimation is not relying on the detected 2D bounding box, we worked around the problem mentioned in Fig. 3.

4.2 Loss Function

It is worth noting, that each output of the network contains the prediction from every trident block branch. It is logical to train each branch with its suitable object scale. To this end, a weight w_i is defined for each branch, and a valid scale between l_i and u_i is assigned to each branch. If the size of the ground truth bounding box located in the range of l_i^2 to u_i^2, the w_i will be set as 1, otherwise 0.

To train the network jointly, a multi-task loss function is used to train the network, which can be expressed as

$$\begin{aligned} L = \boldsymbol{w}(\alpha_1 \cdot \boldsymbol{L}_{rpn\_class} + \alpha_2 \cdot \boldsymbol{L}_{rpn\_bbox} \alpha_3 \cdot \boldsymbol{L}_{rcnn\_class} \\ + \alpha_4 \cdot \boldsymbol{L}_{rcnn\_bbox} + \alpha_5 \cdot \boldsymbol{L}_{tra} + \alpha_6 \cdot \boldsymbol{L}_{rot} + \alpha_7 \cdot \boldsymbol{L}_c), \end{aligned} \tag{1}$$

where $\boldsymbol{w} = [w_0, \cdots, w_n]$, with n equals the number of branches in the trident block, and the $\alpha_1 \cdots \alpha_7$ are the scale factors.

The L_{rpn_class}, L_{rpn_bbox}, L_{rcnn_class}, L_{rcnn_bbox} are the losses used in Faster R-CNN. The softmax loss is used in the classification part, and the smooth L1 loss is used in the regression part. We used L2 loss for the translation and rotation. Considering the confidence rate $\boldsymbol{c}$, it can be formulated as

$$\boldsymbol{L}_{tra} = \boldsymbol{c} \cdot \|\boldsymbol{x} - \hat{\boldsymbol{x}}\|_2 \tag{2}$$

and

$$\boldsymbol{L}_{rot} = \boldsymbol{c} \cdot \left\| \frac{\boldsymbol{q}}{\|\boldsymbol{q}\|_2} - \hat{\boldsymbol{q}} \right\|_2 , \tag{3}$$

where $\boldsymbol{x}$ and $\boldsymbol{q}$ are the predicted translation and rotation quaternion and $\hat{\boldsymbol{x}}$ and $\hat{\boldsymbol{q}}$ are the respective ground truth values. The regularization term is added in the end to prevent the network from achieving a small loss by a strong reduction of the confidence rate, which takes form of

$$\boldsymbol{L}_{\boldsymbol{c}} = -log(\boldsymbol{c}) \, . \tag{4}$$

5 AR Assembly Application

Using the output of the network and the object state graph we can generate an AR application that provides step-by-step guidance to assemble an object. The application keeps track of the current state of the object assembly and uses the pose in order to correctly display the next required step or missing component as an AR augmentation superimposed on the object. Screenshots of the application are shown in Fig. 1.

As discussed previously our implementation is based on the two-stage detector with the pose estimation separated from the computationally heavy RPN. This allows us to use the pose estimation branch on every camera frame (up to ≈30fps) in order to keep track of the current pose and display the augmentations correctly. The object state estimation branch is invoked only on some of the incoming frames since it can process only about 2fps (or 4fps with precision fp16). Moreover, in order to make the state-changing more robust and resilient to ambiguous states, we average the state confidence values over a few frames and only change the state if there is a clear indication from the averaged values (i.e. confidence higher than 0.7).

By using a common coordinate origin for the states of all the models and using the object state graph, the augmentations displaying the next assembly state can be automatically generated by the system without a need for manual content generation and setup.

6 Experiments and Evaluation

In this section we give further details on the creation of the synthetic dataset used for training, implementation details of the network and an evaluation of the system performance on our chosen evaluation object.

Figure 3: A: Even though the object has the same rotation its appearance inside the bounding box differs because of its position with respect to the camera. Thus, it is sub-optimal to recover its rotation only relying on the image inside the bounding box. B: The different states of the object have their own geometric center (marked in red), while they share only one local coordinates(marked in blue). Thus, the pose cannot be estimated by locating the geometric center.

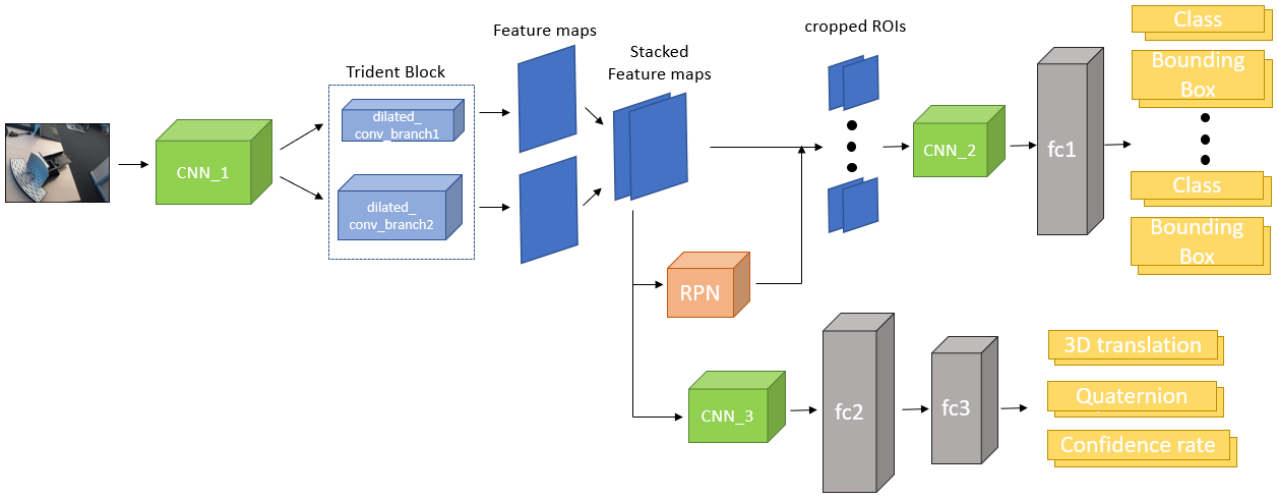

Figure 4: Our proposed architecture based on a modified TridentNet with an additional branch for pose estimation. The details of the network are summarized in the sec. 6.2

6.1 Synthetic Training Dataset

Unlike the object's class and bounding box, the 6DoF pose of the object cannot be labeled manually. Even the tools like [14] are proposed to reduce the cost of labeling, the process is still very time consuming or require specific hardware setups for the acquisition of ground truth poses. However, the quality and size of the dataset is a critical factor for the supervised learning. At the same time, with systems like [15], the 3D model of objects can be easily obtained. Additionally the 3D CAD models are typically available for industrial objects. Due to these reasons, training with synthetic data is a very attractive option.

In this work, we reconstructed the 3D model of every part of our object (i.e. a coffee machine) using a structured light approach. We merged those parts to create models of all potential states. In total, 5 states including the initial state are created for the coffee machine (see Fig. 2). The images from VOC2014 [2] and Ikea dataset [10] are used as the background image. We roughly limited the possible translation and rotation for each state, e.g. the bottom of the object will not appear in the dataset. For every 5 degrees step of each rotation axis, we randomly generate 10 angles within its ± 2.5 degree range. For each angle, translations within the predefined range are generated. When the 2D bounding box is inside of the image frame, the rendered image is passed into an additional occlusion test. Due to the self-occlusion, the differences between states can be unclear. The rendered image is only selected, when at least 30% of the newly added part under this pose is visible. In total, 72903 synthetic training images are created from the selected test object at all possible states.

6.2 Implementation Details

In the work of [9], ResNet was modified with the multi-branch block. The convolutions with kernel size 3×3 in the residual blocks were replaced in the multi-branch with different dilation rates. Typically, the multi-branch blocks are placed in the last stage of the backbone, to achieve different receptive fields with the same representational power. Experiments showed that more than 3 branches brings no further performance improvement. The multiple dilated convolutions can share the weights to reduce the number of parameters in the CNN, with only a small decrease in the performance. We implemented our work in MxNet [1] and used the ResNet50 in bottleneck style pretrained model in the MxNet model zoo. We used the image with the original size of 640×480 to train the network. The details of the architecture in Fig. 4 can be summarized as:

- CNN_1: All the component of ResNet50 until the first ResNet unit of stage_3. The output of CNN_1 is the feature map with the size of $(1024 \times 30 \times 40)$ per image in the mini-batch.
- Trident Block:The rest 5 ResNet units in the stage_3 are modified as the multi-branch block described above. We use the multi-branch block with 2 dilated convolutions with the dilation rate of 2 and 3 that share weights, and valid range for each state are set as $[0, 220]$, and $[180, \infty]$. The output of the trident block has the size of $(2 \times 1024 \times 30 \times 40)$ per image, the 3 corresponds to the 3 branches.
- RPN: At first a CNN and relu-activation extract a feature map, which has the size of $(2 \times 512 \times 30 \times 40)$ per image. The

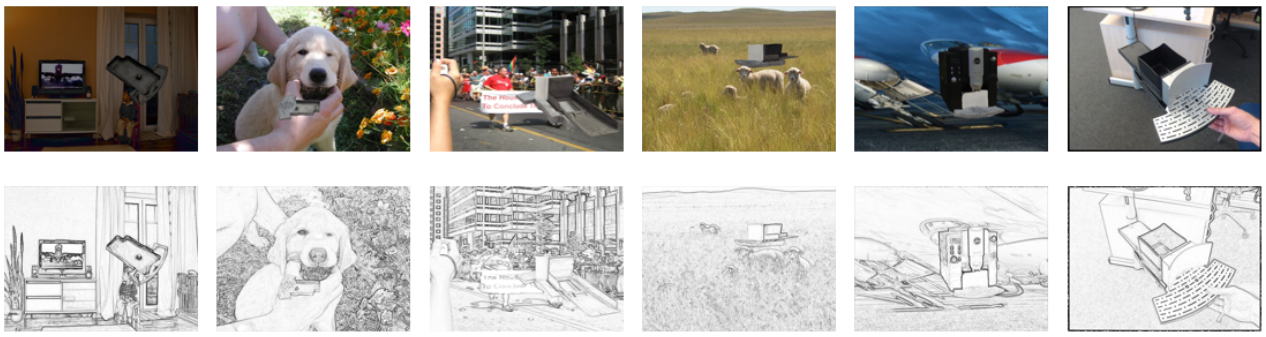

Figure 5: First row: From left to right, the first 5 images are synthetic images with random noise of the 5 state, and the last one is the real image for the validation. Second row: the images after applying the pencil filter

	Accuracy	**IoU** 0.5	**IoU** 0.9	**Recall**	**Precision**	**ADD30**	**ADD10**
Validation (Synthetic)							
State 0	1.0	1.0	1.0	1.0	1.0	1.0	0.89
State 1	1.0	1.0	1.0	1.0	1.0	1.0	0.92
State 2	1.0	1.0	0.96	1.0	1.0	1.0	0.94
State 3	1.0	1.0	0.92	1.0	1.0	1.0	1.0
State 4	1.0	1.0	0.88	1.0	1.0	1.0	1.0
Mean	1.0	1.0	0.94	1.0	1.0	1.0	0.95
Testing (Real images)							
State 0	0.97	0.98	0.37	0.98	1.0	–	–
State 1	0.46	0.47	0.08	0.47	1.0	–	–
State 2	0.83	0.84	0.13	0.84	0.92	–	–
State 3	0.71	0.71	0.08	0.71	0.85	–	–
State 4	0.94	0.94	0.14	0.94	0.94	–	–
Mean	0.78	0.79	0.16	0.79	0.94	–	–

Table 1: Evaluation results on the synthetic validation and real testing datasets. We report accuracy, IoU on the detection bounding box for the state estimation, and the ADD metric for the pose estimation.

feature map is passed to a CNN for classification for the foreground and background, and to another CNN for the bounding box regression, for the predefined number of anchors. The number of anchors is reduced by using non-maximum suppression and filtered according to the valid range of each branch. At last, the ROIs in the output of trident block are resized to $(2 \times 1024 \times 14 \times 14)$ with ROI Align [3].

- CNN_2: CNN_2 corresponds to the stage_4 in the ResNet50, it increasing the number of channels of ROIs from 1024 to 2048.
- fc_1: The CNN_2 is flattened and connected with fc_1, which outputs the object state and the bounding box of ROIs from each branch.
- CNN_3: CNN_3 consists of 4 times the combination of the relu activation and the convolution. All the convolutions have a kernel size of 3×3, 1024 channels, while the first and third convolution has a stride of 2. After CNN_3, the output from trident block is downsampled to the size of $(2 \times 1024 \times 8 \times 10)$.
- fc_2: Fully connected layer of size 2046.
- fc_3: fully connected layer for the prediction of the 6DoF pose and the confidence rate for each branch.

We train the network with Stochastic Gradient Descent(SGD) with 0.9 momentum and a weight decay of 0.0001. The network is trained on an RTX 2080Ti GPU for 8 epochs. Each mini-batch has 1 image. The learning rates start from 0.001 and are factored by 0.5 after each two epochs, until reaching 0.000125. The input position is normalized between -1 and 1. The $[\alpha_1, \cdots, \alpha_7]$ are set as $[1, 0.5, 1, 0.2, 1, 0.3, 5]$.

We used 80% of the synthetic dataset for training the network, the remaining 20% for the validation and about 400 real images for the testing/evaluation. The pencil filter is used to reduce the domain gap between the synthetic images and real images [18]. During the training, 3 types of data augmentation effects are randomly applied on the training image, namely gaussian noise, contrast and illumination changes, and motion blur. The pencil filter is applied on the input image for both training and testing. Examples of the synthetic and real images can be found in the Fig. 5.

6.3 Quantitative Evaluation

To test the network, the prediction of object state are firstly filtered according to the predicted bounding box. The object state with the highest probability is considered as the final prediction. The pose with the highest confidence rate is selected as the final pose prediction. In Table 1, we report the accuracy, recall and precision of state estimation for each state. The intersection over union (IoU0.5 and IoU0.9) values show how often these thresholds were met in the predicted bounding boxes. The ADD metric [5] is used to evaluate the accuracy of the predicted pose. Since we do not have the 6DoF ground truth poses in the real image, the ADD metric will only be reported for the synthetic images. The ADD metric measures the percentage of estimated poses where the average relative error in the estimated pose is under a certain threshold (here 10% and 30%). Since the ground truth poses of the real images are not available, we show some qualitative results in the supplementary material.

7 Conclusion

In this paper we addressed the problem of joint object state and pose estimation for objects that consist of several components. We believe that this is a problem of great significance for AR as it can serve as an important step forward for AR applications that guide object assembly and maintenance. Our proposed solution consists of a single CNN with two branches, one for object detection and current state estimation and one for 6DoF pose estimation. We show how the outputs of this network can be integrated efficiently into an AR assembly application that is capable of operating in real-time given an object state graph. We used exclusively synthetic data for the training of the network which allows making the dataset and AR content generation more generic. On our selected test object we showed promising results on the accuracy of state and pose estimation. Further evaluation could be possible if a multi-state object dataset with more objects and ground truth poses was available. A frame-to-frame object tracker could be used to further improve the accuracy of tracking since currently a tracking by detection approach directly using the network output is applied.

Acknowledgments

This work was partially funded by the INNOPROM Rheinland Pfalz/EFFRE funding program (P1-SZ2-7, 84002637) in cooperation with John Deere GmbH & Co. KG.

References

[1] https://mxnet.apache.org/.

[2] M. Everingham, L. Van Gool, C. K. Williams, J. Winn, and A. Zisserman. The pascal visual object classes (voc) challenge. *International journal of computer vision*, 88(2):303–338, 2010.

[3] K. He, G. Gkioxari, P. Dollár, and R. Girshick. Mask r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pp. 2961–2969, 2017.

[4] S. Henderson and S. Feiner. Exploring the benefits of augmented reality documentation for maintenance and repair. *IEEE transactions on visualization and computer graphics*, 17(10):1355–1368, 2011.

[5] S. Hinterstoisser, V. Lepetit, S. Ilic, S. Holzer, G. Bradski, K. Konolige, and N. Navab. Model based training, detection and pose estimation of texture-less 3d objects in heavily cluttered scenes. In *Asian conference on computer vision*, pp. 548–562. Springer, 2012.

[6] W. Kehl, F. Manhardt, F. Tombari, S. Ilic, and N. Navab. Ssd-6d: Making rgb-based 3d detection and 6d pose estimation great again. In *Proceedings of the IEEE International Conference on Computer Vision*, pp. 1521–1529, 2017.

[7] A. Kendall, M. Grimes, and R. Cipolla. Posenet: A convolutional network for real-time 6-dof camera relocalization. In *Proceedings of the IEEE international conference on computer vision*, pp. 2938–2946, 2015.

[8] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pp. 1097–1105, 2012.

[9] Y. Li, Y. Chen, N. Wang, and Z. Zhang. Scale-aware trident networks for object detection. *arXiv preprint arXiv:1901.01892*, 2019.

[10] J. J. Lim, H. Pirsiavash, and A. Torralba. Parsing IKEA Objects: Fine Pose Estimation. *ICCV*, 2013.

[11] T.-Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie. Feature pyramid networks for object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2117–2125, 2017.

[12] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg. Ssd: Single shot multibox detector. In *European conference on computer vision (ECCV)*, pp. 21–37. Springer, 2016.

[13] D. G. Lowe. Object recognition from local scale-invariant features. In *The proceedings of the seventh IEEE international conference on Computer vision,*, vol. 2, pp. 1150–1157. Ieee, 1999.

[14] P. Marion, P. R. Florence, L. Manuelli, and R. Tedrake. Labelfusion: A pipeline for generating ground truth labels for real rgbd data of cluttered scenes. *arXiv preprint arXiv:1707.04796*, 2017.

[15] R. A. Newcombe, S. Izadi, O. Hilliges, D. Molyneaux, D. Kim, A. J. Davison, P. Kohi, J. Shotton, S. Hodges, and A. Fitzgibbon. Kinectfusion: Real-time dense surface mapping and tracking. In *2011 IEEE International Symposium on Mixed and Augmented Reality*, pp. 127–136. IEEE, 2011.

[16] N. Petersen, A. Pagani, and D. Stricker. Real-time modeling and tracking manual workflows from first-person vision. In *IEEE International Symposium on Mixed and Augmented Reality (ISMAR),*, pp. 117–124. IEEE, 2013.

[17] R. Radkowski, J. Herrema, and J. Oliver. Augmented reality-based manual assembly support with visual features for different degrees of difficulty. *International Journal of Human-Computer Interaction*, 31(5):337–349, 2015.

[18] J. Rambach, C. Deng, A. Pagani, and D. Stricker. Learning 6DoF Object Poses from Synthetic Single Channel Images. In *IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*. IEEE, 2018.

[19] J. Rambach, A. Pagani, M. Schneider, O. Artemenko, and D. Stricker. 6DoF Object Tracking based on 3D Scans for Augmented Reality Remote Live Support. *Computers*, 7(1):6, 2018.

[20] J. Rambach, A. Pagani, and D. Stricker. Augmented Things: Enhancing AR Applications leveraging the Internet of Things and Universal 3D Object Tracking. In *IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*, pp. 103–108. IEEE, 2017.

[21] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR)*, pp. 779–788, 2016.

[22] D. Reiners, D. Stricker, G. Klinker, and S. Müller. Augmented reality for construction tasks: Doorlock assembly. In *IEEE and ACM IWAR*, 1988.

[23] S. Ren, K. He, R. Girshick, and J. Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in neural information processing systems*, pp. 91–99, 2015.

[24] B. Singh and L. S. Davis. An analysis of scale invariance in object detection snip. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3578–3587, 2018.

[25] M. Sundermeyer, Z.-C. Marton, M. Durner, M. Brucker, and R. Triebel. 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.

[26] A. Tang, C. Owen, F. Biocca, and W. Mou. Comparative effectiveness of augmented reality in object assembly. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pp. 73–80. ACM, 2003.

[27] B. Volmer, J. Baumeister, S. Von Itzstein, I. Bornkessel-Schlesewsky, M. Schlesewsky, M. Billinghurst, and B. H. Thomas. A Comparison of Predictive Spatial Augmented Reality Cues for Procedural Tasks. *IEEE transactions on visualization and computer graphics*, 24(11):2846–2856, 2018.

[28] C. Wang, D. Xu, Y. Zhu, R. Martín-Martín, C. Lu, L. Fei-Fei, and S. Savarese. Densefusion: 6d object pose estimation by iterative dense fusion. *arXiv preprint arXiv:1901.04780*, 2019.

[29] X. Wang, S. K. Ong, and A. Y. Nee. A comprehensive survey of augmented reality assembly research. *Advances in Manufacturing*, 4(1):1–22, 2016.

[30] Y. Xiang, T. Schmidt, V. Narayanan, and D. Fox. Posecnn: A convolutional neural network for 6d object pose estimation in cluttered scenes. *arXiv preprint arXiv:1711.00199*, 2017.

[31] J. Zauner, M. Haller, A. Brandl, and W. Hartmann. Authoring of a mixed reality assembly instructor for hierarchical structures. In *Proceedings of the 2nd IEEE/ACM International Symposium on Mixed and Augmented Reality*, p. 237. IEEE Computer Society, 2003.