Real-Time 6D Pose Estimation from a Single RGB Image

Xin Zhang, Zhiguo Jiang, and Haopeng Zhang*

Image Processing Center, School of Astronautics, Beihang University, Beijing, 100191, China

Beijing Key Laboratory of Digital Media, Beijing, 100191, China

Key Laboratory of Spacecraft Design Optimization and Dynamic Simulation Technologies, Ministry of Education, Beijing, 100191, China

Abstract

We propose an end-to-end deep learning architecture for simultaneously detecting objects and recovering 6D poses in an RGB image. Concretely, we extend the 2D detection pipeline with a pose estimation module to indirectly regress the image coordinates of the object's 3D vertices based on 2D detection results. Then the object's 6D pose can be estimated using a Perspective-n-Point algorithm without any post-refinements. Moreover, we elaborately design a backbone structure to maintain spatial resolution of low level features for pose estimation task. Compared with state-of-the-art RGB based pose estimation methods, our approach achieves competitive or superior performance on two benchmark datasets at an inference speed of 25 fps on a GTX 1080Ti GPU, which is capable of real-time processing.

Keywords: 6D pose estimation, real-time processing, coordinate localization, backbone design

1. Introduction

Determining relative 3D location and orientation between the object and the camera is a classical research issue in computer vision. Applications, such as augmented reality, autonomous driving and robotics, put forward new demands on the accuracy and speed of 6D pose estimation algorithms. In the past few years, commodity depth sensors have facilitated many RGB-D based pose estimation methods. However, active depth sensors are limited to be used in short range scope, and consume intensive energy. Therefore, RGB based 6D pose estimation methods are more practical for real-time mobile applications.

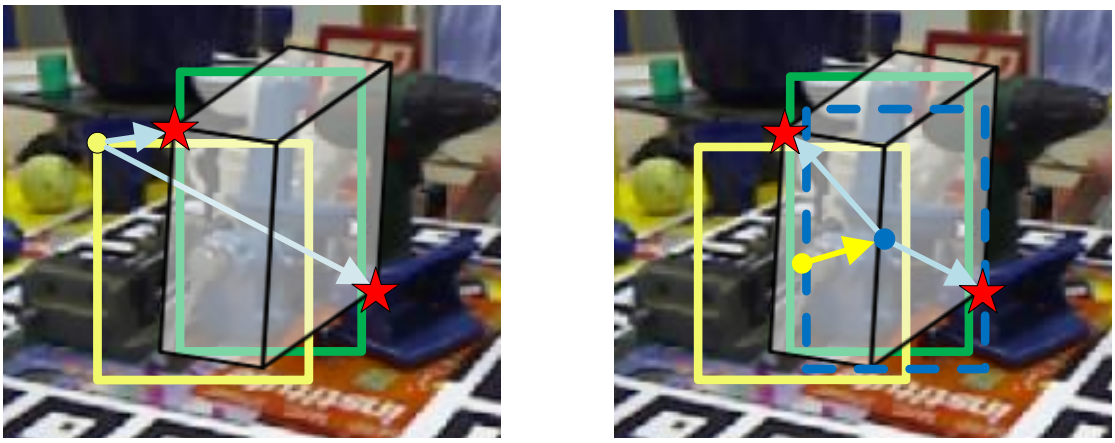
Traditional RGB based pose estimation methods mainly resort to keypoint and edge matching to establish 2D-3D correspondences. Then 6D poses are calculated using a Perspective-n-Point (PnP) algorithm. These methods are effective for high quality images of richly textured objects and scenes. Nonetheless, for poorly textured objects under occlusions and changing illuminations, the 6D pose estimation remains a challenging problem. Recently, the introduce of deep learning techniques, especially convolutional neural networks (CNN), has boosted the performance of 6D pose estimation. There exist two main strategies to employ CNN to estimate 6D poses. The first strategy is to directly regress the pose parameters [1, 2] or viewpoints [3]. These methods are typically used for pose initialization, followed by refinement to improve pose accuracy. Approaches using the second strategy learn to predict the 3D model coordinates corresponding to image pixels [4] or 2D projection locations of the object vertices [5, 6]. Benefited from the strong representation capability of CNN, they can establish 2D-3D correspondences under challenging conditions, and achieve state-of-the-art performance on multiple 6D pose benchmark datasets.

In this work, we develop an end-to-end trainable network to support new demands of real-time 6D pose estimation. Our network takes RGB images as

☆THIS IS A PREPRINT VERSION FOR PERSONAL USE ONLY. Published version can be accessed via ScienceDirect at https://doi.org/10.1016/j.imavis.2019.06.013.

*Corresponding author

Email address: zhang_xin_by@buaa.edu.cn, jiangzg@buaa.edu.cn, and zhanghaopeng@buaa.edu.cn (Xin Zhang, Zhiguo Jiang, and Haopeng Zhang)



(a) The strategy of Tekin et al. [6] (b) Proposed indirect regression strategy

Figure 1: Illustration of proposed indirect strategy for image coordinate regression. The yellow rectangles and the green rectangles represent anchors and ground truth bounding boxes, respectively. The cuboids represent 3D bounding boxes of an object. In (a), Tekin et al. [6] regress all the image coordinates of 3D bounding box vertices from the left top corner of an anchor. Whereas the proposed strategy (b) utilizes 2D detection results (blue dashed lines) as intermediaries to reduce the length and variance of regression targets.

inputs, and simultaneously detects objects and estimates their poses in single forward pass. Inspired by BB8 [5], we adopt the second strategy to predict the image coordinates of 3D bounding box vertices in pose estimation phase. Firstly, we propose an indirect strategy to regress the image coordinates based on 2D detection results for better localization precision. In contrast to a recent similar work [6], we utilize the 2D bounding boxes as intermediaries and calculate the image coordinate regression targets based on their centers. As illustrated in Fig. 1, we use short range offsets to refine error-prone long range offsets. The proposed strategy can remarkably reduce the length and variance of regression targets, which is helpful for stable training and robust prediction. Secondly, we elaborately design the network structure to maintain spatial resolution of low level features, which is demonstrated to be critical for accurate pose estimation. More specifically, we pay attention to the gap between the image classification and the pose estimation problem. Traditional backbones designed for image classification have large down-sampling factors to extract highly abstract features, which are discriminative for inter-category differences. Whereas pose estimation focuses on appearance variation from different perspectives of a few specific objects. Therefore, we attempt to preserve details and structural information by maintaining the resolution of low level features. Comprehensive experiments are performed on two wildly used 6D pose estimation benchmarks, i.e., LINEMOD dataset [7] and OCCLUSION dataset [8], and the results show that our approach competes with state-of-the-art RGB based pose estimation methods even when they are used with post-refinements involving depth information. In summary, the main contributions of our work are as follows:

- We propose an indirect regression strategy which fully utilizes 2D detection results to improve 3D vertices localization precision.
- We specifically design a backbone structure for pose estimation task by maintaining the spatial resolution of low level features.
- We achieve state-of-the-art pose accuracy on the LINEMOD dataset and the OCCLUSION dataset using RGB images only with real-time processing capability.

The rest of the paper is organized as follows. After reviewing related works in Section 2, we detail each component of our method in Section 3. Section 4 presents the ablation experiments and comparison with the state-of-the-art methods. Finally conclusions are summarized in Section 5.

2. Related Work

In this section, we briefly review the extensive literature on 6D pose estimation, mainly focusing on

recent representative works. Most previous studies on pose estimation are based on reasonable assumptions of priori knowledge and input forms. The priori knowledge generally includes calibrated cameras and available 3D models, and the input forms vary from monocular RGB images to RGB-D data or point clouds.

2.1. RGB-D methods.

In the last few years, the emergence of commercial depth cameras has facilitated the development of RGB-D based pose estimation methods. For example, Hinterstoisser et al. [7] proposed surface normal template matching for 3D point clouds. Several variants of Point Pair Feature [9, 10] were proposed to improve the robustness against background clutter and sensor noises. Kehl et al. [11] employed a convolutional auto-encoder to regress descriptors of locally-sampled RGB-D patches for 6D vote casting. Although achieving promising performance, RGB-D based pose estimation methods generally involve sampling and voting schemes, which are computationally expensive. Furthermore, acquiring depth information is energy consuming, and the depth data usually contains noises and holes due to specularity. Therefore, in this work we mainly focus on RGB based 6D pose estimation methods for efficiency and usability.

2.2. RGB methods.

Given a set of 2D-3D correspondences, 6D pose estimation of an object instance has been formulated previously as a pure geometric problem, known as the Perspective-n-Point (PnP) problem. Several closed form [12] and iterative solutions [13] were proposed in the literature. However, establishing 2D-3D correspondences between RGB images and 3D models is a non-trivial task. In terms of this issue, traditional pose estimation approaches can be categorized into keypoint-based methods and appearance-based methods. Keypoint-based methods [14, 15] resort to matching local features to establish 2D-3D correspondences, followed by a PnP solution to calculate 6D pose parameters. Despite the high precision, they are slow due to feature extraction and inadequate for addressing textureless objects. Appearance-based methods bypass the troublesome procedure of determining 2D-3D correspondences using template matching [16]. These methods can roughly determine the pose parameters, nevertheless the number of templates grows sharply when a more accurate estimation is required. Currently, the research hotspot of 6D pose estimation has focused on weakly textured objects under changing illuminations and occlusions. A large number of methods [17, 18] have adopted popular machine learning techniques, such as random forest and deep neural networks, to cope with the challenges of complex conditions.

In recent years, CNN has been successfully applied to many computer vision tasks, including 6D pose estimation. In terms of the output form, there exist two main strategies to utilize CNN for predicting 6D poses. In the first class, CNNs directly yield continuous pose parameters or discretized viewpoints. To name a few, PoseCNN [2] and DeepMDPose [1] were designed to detect and segment objects in the input image, meanwhile regress convolutional features of the objects to 6D pose parameters. SSD-6D [3] discretized the pose space in the form of viewpoint and inplane rotation, and then extended SSD [19] with a pose classification branch. Sundermeyer et al. [20] proposed Augmented Autoencoder to learn implicit representations of object orientations in latent space. These methods follow the paradigm of appearance-based methods, and usually rely on post-refinement to improve pose accuracy. Approaches in the second class adopt the philosophy of keypoint-based methods, learning to predict 2D-3D correspondences between the RGB images and the 3D models. In [5, 6, 21] the CNNs predicted 2D projection locations of 3D bounding box corners in the input images. [6] extended YOLO [22] to directly regress the coordinates, while [21] predicted heatmaps from sampled image patches to reduce the influence of occlusions. These methods are able to establish 2D-3D correspondences under challenging conditions, followed by a PnP solution to achieve accurate pose estimates on multiple 6D pose benchmark datasets.

Early works [23, 24] treated object detection and pose estimation as two separate problems. They typically relied on off-the-shelf 2D detectors to locate the objects of interest in advance. However, due to the inevitable localization errors, this multistage pipeline often suffers from inaccurate and redundant detections, which can lead to inefficiency or even failure. Moreover, the ability to identify poses of objects may in turn improve the performance of detection. Therefore, several state-of-the-art methods [1, 3, 6] attempted to augment 2D detectors for 6D pose estimation, integrating multitask supervision information. We also follow this

trend to leverage the success on 2D object detection for 6D pose estimation in our work.

3. Approach

Our goal is to develop an end-to-end framework for simultaneous detection and 6D pose estimation in real-time. Single shot 2D object detectors [19, 25] have shown impressive performance on the first task. Motivated by [6], we extend 2D detection pipeline to predict 2D projections of 3D bounding box corners for each object instance in the image. Then we can calculate the 6D pose with an efficient PnP algorithm [12] given these 2D-3D correspondences. It is worth noting that Tekin et al. [6] totally ignore the 2D detection ability of their extended version of YOLO [22], whereas we propose to indirectly regress the image coordinates based on intermediary 2D detection results to improve localization precision. Furthermore, we construct our architecture properly and demonstrate that maintaining the spatial resolution of low level features is crucial for achieving good pose estimation results. Our approach significantly boosts the accuracy of Tekin et al. [6] and meanwhile retains the capability of real-time processing. The schematic overview of proposed network is shown in Fig. 3. We now describe each part of our approach in more detail.

3.1. Problem Formulation

Pose estimation aims at retrieving the 6 Degree-of-Freedom (6-DoF) transformation of the object coordinate frame with reference to the camera coordinate frame. The geometry of the coordinate frames is presented in Fig. 2. The object self-centered frame $O_oX_oY_oZ_o$ and the camera frame $O_cX_cY_cZ_c$ are related by the 3D rigid transformation

$$\mathbf{x}_c = \mathbf{R}\mathbf{x}_o + \mathbf{t}, \quad \mathbf{R} \in SO(3) \tag{1}$$

where $\mathbf{x}_o$ and $\mathbf{x}_c$ denote 3D coordinates of the same point in object frame and camera frame, respectively. $\mathbf{R}$ is a 3×3 rotation matrix which rotates the object frame to align with the camera frame and $\mathbf{t}$ is the translation vector equaling O_CO_O. Then the perspective projection procedure can be modeled as

$$\begin{pmatrix} \mathbf{x}_p \\ 1 \end{pmatrix} \sim \mathbf{K}(\mathbf{R}|\mathbf{t}) \begin{pmatrix} \mathbf{x}_o \\ 1 \end{pmatrix} \tag{2}$$

in which $\mathbf{x}_p$ represents the 2D image projection location. The symbol $\sim$ means equal in homogeneous manner, and K is the inner calibration matrix assumed to be known. One critical issue for most pose estimation algorithms is to establish accurate and robust 2D-3D correspondences. For weakly textured objects, however, it still remains a challenge. In this work, we establish 2D-3D correspondences by means of predicting the 2D image coordinates of 3D bounding box corners, inspired by [5]. Given these 2D-3D correspondences, we calculate 6D pose parameters by solving a set of equation 2.

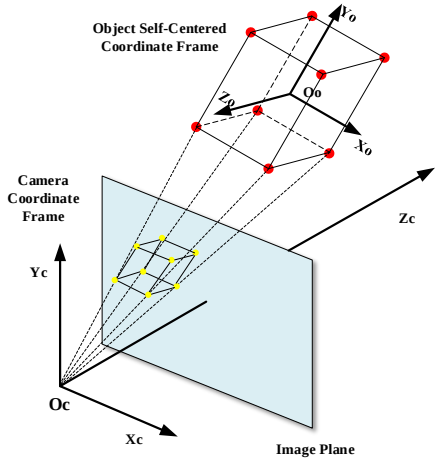


Figure 2: The geometry of the object self-centered frame and the camera frame. The red points represent the 3D virtual vertices related to the 3D model. The corresponding projection locations on the image plane are denoted by the yellow points.

3.2. Network Architecture

Tekin et al. [6] select YOLO as base framework for extreme speed/accuracy trade-off. In this work, we elaborately construct our network integrating the Feature Pyramid Network (FPN) [26] and SSD [19] for simultaneous detection and 6D pose estimation. The multi-scale architecture and anchors of various aspect ratios allow for smooth search over many differently-sized features in a single pass. As shown in Fig. 3, the input RGB images are resized to 448×448 and fed into the backbone network constructed on a modified ResNet architecture [27] which we denote by ResNet-h. The origin ResNet down-samples too fast at the first several layers, losing vast quantities of details and structural information which are critical for pose estimation. In ResNet-h, we remove the max pooling layer in Stage

1 of the original ResNet to keep high spatial resolution of low level features for accurate localization. Thereby, the output of ResNet-h Stage 3 has strides of 4 with respect to the input image. We notice that several works [28, 29] proposed to maintain the spatial resolution of features for detection and semantic segmentation. However, no similar structure has been specifically designed for 6D pose estimation task as far as we know. Furthermore, both [28] and [29] kept high spatial resolution in deeper layers, whereas we demonstrate that maintaining the spatial resolution of low level features can be more effective and efficient for pose estimation task. In Sec. 4.2, we compare our backbone structure with those of [28, 29] in terms of both pose accuracy and computational cost.

Following [25], We branch off after ResNet-h Stage 3 through Stage 5, and then attach top-down and lateral connections to extract multi-scale features over the image. P6 and P7 are successively down-sampled by a 3×3 stride-2 convolution layer to cover large objects. The top-down pathway and lateral connections compensate for the lack of semantic information due to maintaining high spatial resolution of features. We use (w_s, h_s, c_s) to denote the dimensions of the feature map at scale s, where c_s is set to 256 for all feature levels P3 through P7. Each feature map is convolved with a set of $3\times3\times c_s$ kernels to jointly classify the objects, refine the 2D bounding boxes and regress the projection locations of the 3D control points. We create N_{anchor} anchor boxes at each location of the feature maps with various sizes and ratios. Positive and negative anchor boxes are decided by the overlaps with the ground truth 2D bounding boxes. Each positive anchor box is assigned a length C one-hot vector of classification targets, and a length 4 vector of box refinement targets, along with a length $N_{pt} \times 2$ vector of coordinate regression targets. The term C denotes the number of object classes excluding the background, and N_{pt} is the number of 3D control points, which is set to 8 in our implementation. Then the output of *Detector & Pose Predictor* at scale s is a 3D tensor of size $(w_s, h_s, N_{anchor} \times (N_{pt} \times 2 + 4 + C + 1))$. We use anchors at three aspect ratios $\{1:2,\ 1:1,\ 2:1\}$, with sizes of 25^2 to 224^2 on pyramid levels P3 to P7, respectively. The total number of anchor boxes over the whole image adds up to 66836 in order to cover the variety of objects in terms of scale and shape. Meanwhile, our method still runs at a fast speed thanks to the fully-convolutional architecture.

Both SSD-6D [3] and our model adopt a multi-scale architecture to estimate poses for objects of various sizes. In contrast to their structure, our model is more efficient in two aspects. Firstly, we condense the channels of features P3 through P7 to 256, which is much smaller than 384 to 1536 in SSD-6D. Secondly, our pose predictor is considerably lighter due to an efficient pose representation. Both of these advantages can reduce the computational cost in pose estimation. Therefore, although maintaining high spatial resolution of features, our approach is substantially faster than SSD-6D as shown in Table 6.

3.3. Training and Inference

We construct synthetic training sets to solve the problem of insufficient annotated data. The training poses for each object are selected as in [5, 6] such that the upper hemisphere is sparsely covered. We take random images from MS COCO dataset [30] and resize them to 640×480 as background to avoid overfitting to the scene context. The segmented target objects are scaled by a factor of $s \in [0.8, 1.2]$ and randomly placed onto the background. we also apply various color augmentation by randomly changing the hue, saturation, exposure and contrast of the images. As suggested in SSD [19], we select hard negatives anchor boxes so that the positives-negatives ratio is 1:3, to achieve fast convergence and stable training.

We extend the MultiBox loss of SSD to take image locations regression of 3D control points into account. Given a set of positive boxes Pos and hard-mined negative boxes Neg, we train our network by minimizing the following loss function:

$$L(Pos, Neg) = \sum_{x \in Neg} L_{conf} + \sum_{x \in Pos} (L_{conf} + \alpha L_{loc} + \beta L_{pt}) \quad (3)$$

The terms L_{conf}, L_{loc} and L_{pt} denote the classification loss, 2D bounding boxes fitting loss and coordinate regression loss, respectively. In terms of L_{pt}, we indirectly regress the image coordinates of 3D bounding box vertices via the intermediate 2D detection results. Specifically, we predict offsets for the coordinates with respect to the centers of the regressed 2D bounding boxes, rather than the left-top corners of the assigned anchor boxes as in [6], i.e.

$$\delta_x = \frac{T_x - B_x}{B_w},\ \delta_y = \frac{T_y - B_y}{B_h} \quad (4)$$

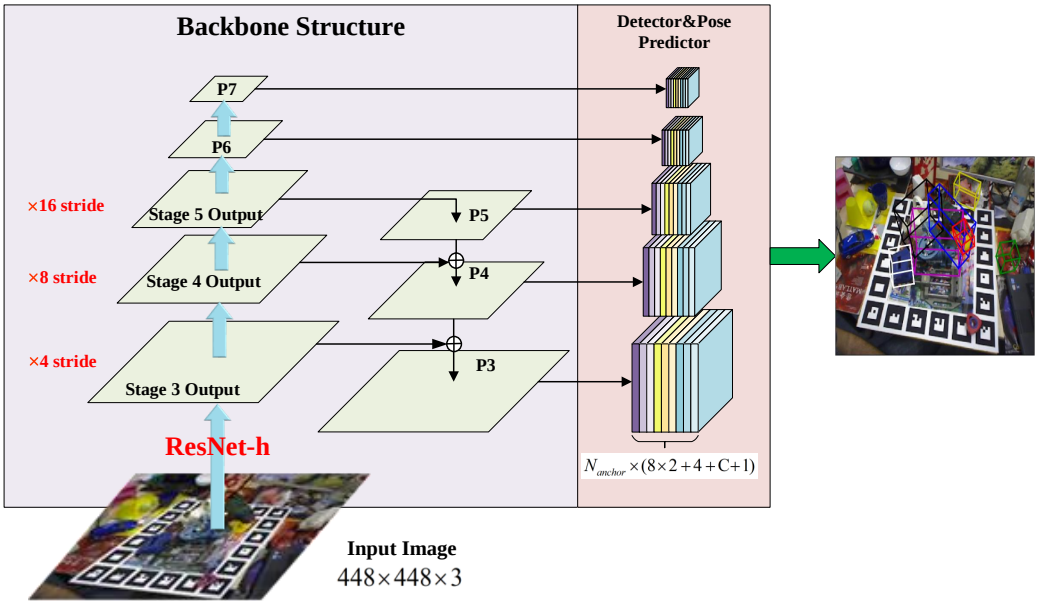


Figure 3: The schematic overview of proposed network. We extend 2D detection pipeline to predict the image coordinates of 3D bounding box vertices for each object instance in the image. We construct feature pyramid on top of ResNet-h, which is specifically designed for pose estimation task by maintaining spatial resolution of low level features.

where T and B denote ground truth coordinates and the regressed box, respectively. The subscripts x, y, w, h stand for the box's center coordinates and its width and height. The regressed bounding boxes are likely to be more fitted to the objects than the assigned anchor boxes, therefore our indirect strategy can reduce the length and variance of regression targets, leading to stable training and accurate prediction. We employ softmax cross-entropy loss for classification, whereas smooth L1-norm loss for L_{loc} and L_{pt}.

We adopt a two step training strategy for our multi-task network. Firstly, we train our network without the coordinate regression loss in equation 3 to learn to locate the objects. In the second stage, we fine-tune the model with the full loss. We freeze the first several convolution layers of the proposed ResNet-h backbone and fine-tune the network using stochastic gradient descent with 0.9 momentum, 0.0005 weight decay, and batch size 8. In our experiments, we emphasize the loss term associated with pose estimation by setting $\alpha = 1, \beta = 10$. The initial learning rate is set to 0.001 for all the sequences in our experiments. We train our network for 45k iterations in each stage and divide the learning rate by 10 at 30k and 40k iterations.

When testing, we simultaneously detect in 2D and estimate 6D poses by conducting a forward pass of our network. The network outputs the object identities with scores, 2D bounding boxes, and the 2D projections of the object's 3D control points. We only select at most 400 top-scoring predictions after thresholding confidence at 0.01. Non-maximum suppression with a threshold of 0.45 is applied to the merged predictions from all levels, yielding the final detections. We calculate the 6D pose from the 2D-3D correspondences by solving a set of equation 2 for each object instance. As in [5, 6], we utilize an efficient PnP algorithm [12] and achieve an estimate of the 6D transformation of the object coordinate frame with respect to the camera coordinate frame.

4. Experiments

Our method is implemented using MXNet [31] and ran on an Intel Core i7-6800K@3.40GHz desktop with a GeForce 1080Ti GPU. We present and compare our results with the state-of-the-art pose estimation methods on the LINEMOD [7] and OCCLUSION [8] datasets. LINEMOD is a standard benchmark for 6D pose estimation algorithms and

consists of 15 sequences of indoor scenes. In each frame, one textureless instance in the center is annotated with identity, 2D bounding box and 6D pose. OCCLUSION is an extensively annotated version of sequence 02 in the LINEMOD dataset where each image focuses on instances of 8 objects undergoing heavy occlusions in most cases.

4.1. Evaluation Metrics

We use four standard metrics to evaluate 6D pose accuracy, including 2D reprojection error, 2D Intersection over Union (IoU) score, average distance of model points (referred to as ADD metric), and $5cm$ 5° metric as in [3, 5, 6].[1] The presented results are the percentage of correctly estimated poses within certain error thresholds. To measure 2D pose errors, we project the object's model vertices into the image plane using the estimated poses and the ground truth poses. In terms of reprojection error, we consider the estimated pose to be correct when the mean distance between the 2D projections is less than 5 pixels. This metric is designed for applications such as augmented reality. In terms of 2D IoU score, we calculate the overlap of the rendered masks' bounding boxes, and provide results of correct poses at certain IoU threshold. To measure pose errors in 3D, the most extensively used error function is the ADD metric [7], which calculates the average distance between transformed vertices of object model M by the ground truth pose $\mathbf{P}$ and the estimated pose $\hat{\mathbf{P}}$.

$$e_{ADD}(\mathbf{P}, \hat{\mathbf{P}}; M) = \underset{\mathbf{x} \in M}{avg} ||\mathbf{P}\mathbf{x} - \hat{\mathbf{P}}\mathbf{x}||_2 \tag{5}$$

For symmetric objects with ambiguous poses such as *EggBox* and *Glue* in the LINEMOD dataset, we use the indistinguishable version of the ADD metric as in [5, 6]. The threshold is set to 10% of the object's diameter.

$$e_{ADI}(\mathbf{P}, \hat{\mathbf{P}}; M) = \underset{\mathbf{x}_1 \in M}{avg} \min_{\mathbf{x}_2 \in M} ||\mathbf{P}\mathbf{x}_1 - \hat{\mathbf{P}}\mathbf{x}_2||_2 \tag{6}$$

We also compare the absolute error of 6D poses using the $5cm$ 5° metric. With this metric, the estimated pose is accepted if the translation and rotation errors are below 5cm and 5°, respectively.

[1] We use the public code in https://github.com/thodan/obj_pose_eval.

4.2. Ablation Study

In this section, we analyze the effects of backbone design, regression strategy and input size on pose estimation. Ablation experiments are conducted on the LINEMOD dataset, and average results over the 13 objects (see Sec. 4.3) are presented in Table 1. We prove the validity of proposed indirect regression strategy and maintaining high resolution of features for improving pose accuracy. A trade-off between accuracy and speed can be achieved by changing the input size.

4.2.1. Backbone Design.

We use a 50 layer Residual Network as the baseline to build our model. [28] and [29] have pointed out that maintaining high spatial resolution of features can improve performance for 2D detection and semantic segmentation. However, simply adopting their structures may not be suitable for pose estimation task. Following the principle, we propose ResNet-h backbone design as illustrated in figure 4. ResNet-h removes the max pooling layer in Stage 1, and branches off after Stage 3 through Stage 5, with down-sampling factors of 4, 8 and 16, respectively. Our structure design reduces the down-sampling rate of features from the low level of the network, therefore retains accurate spatial location information. For comparison, we also construct two variants of ResNet, called ResNet-atrous and ResNet-detnet, according to the structures of [29] and [28]. Both of them utilize atrous convolution operator in Stage 4 and Stage 5 to keep high spatial resolution of deeper features. ResNet-detnet uses atrous convolution with rate 2 only at the first residual unit of Stage 4 and Stage 5. Whereas ResNet-atrous uses atrous convolution with rate 2 in all the residual units of Stage 4, and rate 4 in Stage 5. The FLOPs (floating-point operations) of proposed ResNet-h is about 60.2G at input size 448×448, while those of ResNet-atrous and ResNet-detnet are greater than 80G. As reported in Table 1, ResNet-h efficiently achieves best pose accuracy among the variants of ResNet and has real-time processing capability. The results illustrate that maintaining spatial resolution of low level features is more critical than that of top level features for 6D pose estimation task.

4.2.2. Regression Strategy.

We propose an indirect strategy to regress the image coordinates of 3D vertices based on intermediate 2D detection results. Our strategy replaces the

error-prone long range offsets using the more accurate short range offsets, leading to stable training and robust prediction. To demonstrate the validity of proposed indirect strategy, we compare with the direct strategy which regress the image coordinates with respect to the centers of anchor boxes similar to [6]. The pose accuracy results are reported in row 1 to 8 of Table 1, and we find that proposed indirect strategy can boost the performances by about 2% for all the backbones. This improvement is considerable since we hardly add any computational overhead.

4.2.3. Input Size.

We present a speed-accuracy trade-off by changing the input size to 300×300 in row 9 to 12 of Table 1. The FLOPs decrease by about two times, and our model with ResNet-h backbone can run at a speed of 53 fps with competitive pose accuracy. When reducing the resolution of input images, the performances decline naturally for all the backbone structures. However, it is worth noting that the pose accuracy of proposed ResNet-h decreases the least among the four backbones, since we attempt to maintain spatial resolution of low level features. This robustness against spatial down-sampling can be beneficial for pose estimation of small objects. Tekin et al. [6] also showed their speed-accuracy trade-off results for different input sizes. Although running fast, their best pose accuracy is still much lower than ours.

4.3. Results on LINEMOD Dataset

On the LINEMOD dataset, we evaluate our method in terms of single object detection and 6D pose estimation using RGB images only. The LINEMOD [7] dataset contains 15 sequences of indoor images in which the central object is annotated with a ground truth pose. Two sequences, *Cup* and *Bowl*, are omitted since the 3D models are incomplete. We use the same train/test split as in [5, 6] and augment the training sets as described in Sec. 3.3. We follow the evaluation protocol of [5, 6] by measuring accuracy as the percentage of correctly estimated poses in the test sets. Quantitative results of our method in terms of 2D pose accuracy and 3D pose accuracy are presented. We also provide qualitative examples of pose prediction in Figure 5.

4.3.1. 2D Pose Accuracy.

In Table 2, we compare our results with those of the state-of-the-art methods in terms of 2D reprojection error. [17] and [5] involve a multi-stage procedure and require detailed 3D models to refine the pose predictions. Whereas Tekin et al. [6] and our network can be trained in end-to-end fashion. We achieve best accuracy among all the competing methods even without post-refinement, and outperform Tekin et al. [6] by 4%. In Table 3, we perform a similar comparison in terms of the IoU metric under threshold 0.5. SSD-6D [3] requires a pose refinement, whereas Tekin et al. [6], Deep-6DPose [1] and ours do not. Our results are better than those of SSD-6D [3] and Deep-6DPose [1], but a little bit lower than that of Tekin et al. [6]. Since the pose accuracy measured by IoU metric under threshold 0.5 is almost perfect, we present our results under higher thresholds for further comparison. As can be seen, our approach can yield pose predictions that are highly overlapping with ground truth for most of the frames in the test sets.

4.3.2. 3D Pose Accuracy.

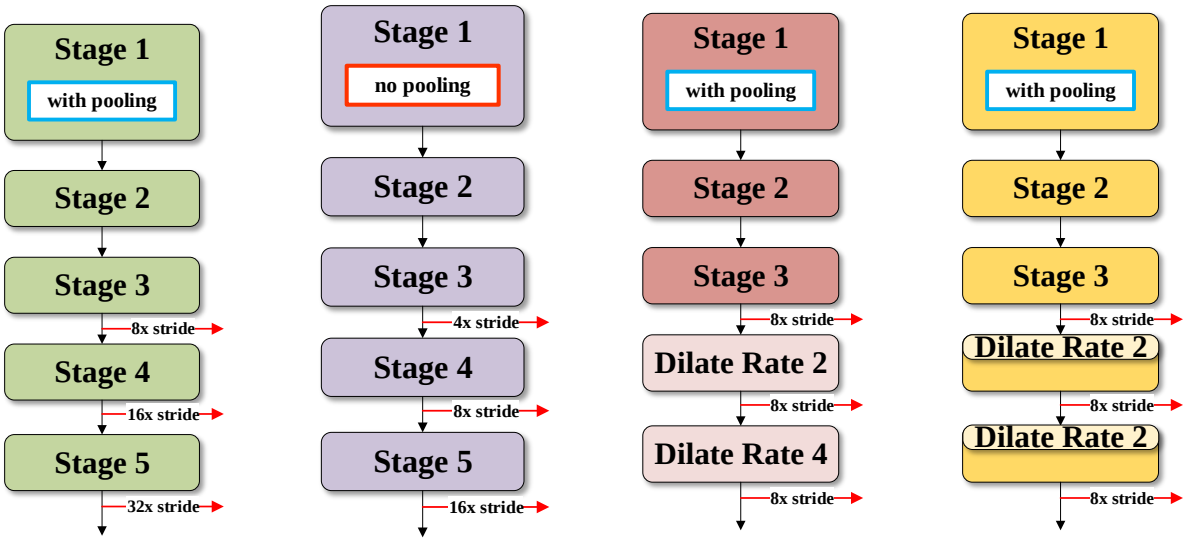
In Table 4, we compare with the state-of-the-art methods in terms of the ADD metric described in Section 4.1. The results of *EggBox* and *Glue* are measured using the ADI metric as in [5, 6]. We outperform all the competing methods when used without pose refinement. Using the 3D CAD models, BB8 [5] and SSD-6D [3] rely heavily on post refinement to increase their pose accuracy by rendering and aligning, which is computationally intensive. However, our results are still better than that of BB8 after refinement by a margin of 9%. Taking advantage of a large rendered training set, SSD-6D is able to densely sample the viewpoints and inplane rotations. In contrast, we only select about 200 viewpoints sparsely sampled from the upper hemisphere for training. In terms of small objects such as *Ape* and *Duck*, our approach has a significant advantage over Tekin et al. [6], thanks to the high spatial resolution of features and the multi-scale architecture. Table 5 presents results before and after refinement for the competing methods when the absolute pose error is less than $5cm$ and $5°$. Our approach is more stable for different objects and achieves state-of-the-art pose accuracy without any post-refinement. The inference speed of our approach for single object is reported in Table 6. Benefited from the fully convolutional architecture and no need of refinement, We can per-

Table 1: Ablation studies about the effects of backbone design, regression strategy and input size on pose estimation accuracy. We report average percentages of correctly estimated poses on the LINEMOD dataset.

Row	Methods			ADD	$5cm$ $5°$	Reproj. 5px	Inference Speed
	reg. strategy	backbone	input shape				
1	direct	Resnet-50-h	448	70.46	80.35	92.53	25 fps
2	direct	Resnet-50-a	448	63.87	77.72	85.18	18 fps
3	direct	Resnet-50-d	448	64.80	73.79	85.17	19 fps
4	direct	Resnet-50	448	62.80	68.79	81.10	45 fps
5	indirect	Resnet-50-h	448	**71.70**	**84.38**	**94.68**	25 fps
6	indirect	Resnet-50-a	448	66.97	76.16	88.90	18 fps
7	indirect	Resnet-50-d	448	66.38	75.77	89.05	19 fps
8	indirect	Resnet-50	448	63.74	71.23	84.74	45 fps
9	indirect	Resnet-50-h	300	69.34	82.82	94.29	53 fps
10	indirect	Resnet-50-a	300	61.76	70.82	84.07	37 fps
11	indirect	Resnet-50-d	300	60.34	69.41	84.92	40 fps
12	indirect	Resnet-50	300	54.24	58.18	75.49	87 fps

Table 2: Comparison of our approach with state-of-the-art algorithms on the LINEMOD dataset in terms of 2D reprojection error. **Bold face** numbers denote the best overall methods.

Method	w/o Refinement				w/ Refinement	
Object	Brachmann [17]	BB8 [5]	Tekin [6]	OURS	Brachmann [17]	BB8 [5]
Ape	-	95.3	92.10	**98.01**	85.2	96.6
Benchvise	-	80.0	**95.06**	93.56	67.9	90.1
Cam	-	80.9	93.24	**98.44**	58.7	86.0
Can	-	84.1	**97.44**	96.48	70.8	91.2
Cat	-	97.0	97.41	**98.91**	84.2	98.8
Driller	-	74.1	79.41	**87.21**	73.9	80.9
Duck	-	81.2	94.65	**98.23**	73.1	92.2
Eggbox	-	87.9	90.33	**96.83**	83.1	91.0
Glue	-	89.0	**96.53**	95.29	74.2	92.3
Holepuncher	-	90.5	92.86	**98.20**	78.9	95.3
Iron	-	78.9	82.94	**89.72**	83.6	84.8
Lamp	-	74.4	76.87	**86.17**	64.0	75.8
Phone	-	77.6	86.07	**93.75**	60.6	85.3
Average	69.5	83.9	90.37	**94.68**	73.7	89.3



(a) ResNet-50 (b) ResNet-50-h (c) ResNet-50-a (d) ResNet-50-d

Figure 4: Three variants of ResNet-50 we constructed to maintain spatial resolution of features. ResNet-h removes the max pooling layer in Stage 1. ResNet-atrous and ResNet-detnet utilize atrous convolution operators in Stage 4 and Stage 5 according to [29] and [28], respectively. We point out that ResNet-h keeps higher resolution in low level features, whereas ResNet-atrous and ResNet-detnet focus on deeper features.

Figure 5: Qualitative 6D pose estimation results on the LINEMOD dataset. The green and blue bounding boxes are rendered using ground truth poses and predicted poses, respectively. As can be seen, our method is robust to scale and light changes.

Table 3: Comparison of our approach with state-of-the-art algorithms on the LINEMOD dataset in terms of IoU metric.

Threshold	0.5				0.6	0.7	0.8	0.9
Method	SSD-6D [3]	Tekin [6]	Deep-6DPose [1]	OURS	OURS	OURS	OURS	OURS
Ape	-	99.81	99.8	99.91	99.91	99.15	96.88	89.87
Benchvise	-	99.90	100	99.81	99.42	99.13	97.88	77.31
Cam	-	100	99.7	100	100	99.90	99.32	92.48
Can	-	99.81	100	99.80	99.71	99.41	98.24	79.98
Cat	-	99.90	99.2	99.90	99.90	99.40	97.52	72.02
Driller	-	100	100	99.90	99.41	98.34	93.65	64.55
Duck	-	100	99.8	99.72	99.72	99.53	97.67	87.78
Eggbox	-	99.91	99.0	99.63	99.53	99.44	98.69	94.22
Glue	-	99.81	97.1	98.08	97.79	96.92	93.08	73.17
Holepuncher	-	99.90	98.0	100	99.91	99.72	98.39	86.36
Iron	-	100	99.7	99.90	99.80	99.50	98.29	81.45
Lamp	-	100	99.8	100	99.91	99.62	94.13	59.47
Phone	-	100	99.1	98.97	98.69	98.41	96.46	77.43
Average	99.4	99.92	99.3	99.66	99.52	99.11	96.94	79.70

form simultaneous detection and pose estimation with real-time processing capability.

4.4. Results on OCCLUSION Dataset

To demonstrate robustness with respect to occlusions, we conduct experiments for multi-object detection and 6D pose estimation on the challenging OCCLUSION dataset. Unlike in LINEMOD sequences, the object identities are not known a priori, which puts forward great difficulty for coordinate regression since the network has to learn various modalities of different objects. We construct a synthetic training set of 20,000 images as described in Sec. 3.3 using the same objects extracted from the corresponding sequences in the LINEMOD dataset, which has become a common protocol as in [5, 6, 21]. The OCCLUSION dataset is only used as test set so that the occlusion patterns are not seen in advance. The network is trained for 112.5k iterations in total and divide the learning rate by 10 at 75k and 100k iterations. Other training settings are the same as in Sec. 3.3. We report our pose estimation results in Table 7 and Fig. 6. It can be seen that our method achieves the best pose accuracy in terms of 2D reprojection error, which is the most widely used pose metric on the OCCLUSION dataset. Our approach substantially outperforms Tekin et al. [6] and PoseCNN [2] when used with only RGB images, even if PoseCNN involved semantic labeling supervision for pose estimation. [21] adopted a sampling and accumulating scheme to reduce the influence of occlusions at expense of computational efficiency. They also used a Feature Mapping (FM) [32] method to bridge the domain gap between the synthetic training data and the real-world test images, whereas we do not. For fairness, we compare with their results without FM. As shown in Table 6 and Table 7, our approach is several times faster than PoseCNN [2] and [21] meanwhile achieves competitive pose accuracy. We also provide our results under $5cm$ 5° and ADD metric for further comparison. The pose accuracy on Eggbox is significantly lower than other objects because more than 70% of close poses are not seen in the training sequences. In terms of object detection, we can report a mean Average Precision (mAP) of 0.84 at IoU threshold 0.5 over the 8 objects. Qualitative results on the OCCLUSION dataset are presented in Figure 7.

5. Conclusion

In this paper, we have developed a CNN framework to simultaneously detect objects and predict 6D poses for real-time applications using RGB images only. Following the paradigm of keypoint-based methods, we establish 2D-3D correspondences by employing CNN to regress the image coordinates of 3D virtual vertices. We propose an indirect strategy utilizing intermediate 2D detection results to improve localization precision. We also demonstrate that maintaining spatial resolution of

Table 4: Comparison of our approach with state-of-the-art algorithms on the LINEMOD dataset in terms of ADD metric. **Bold face** numbers denote the best overall methods, red numbers denote the best methods among those that do not use refinement, if different.

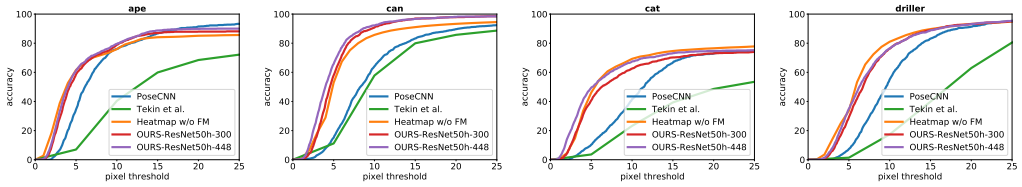
Method	w/o Refinement						w/ Refinement		
Object	[17]	BB8[5]	SSD-6D[3]	Tekin[6]	Deep-6DPose[1]	OURS	[17]	BB8[5]	SSD-6D[3]
Ape	-	27.9	0	21.62	38.8	41.48	33.2	40.4	**65**
Bvise	-	62.0	0.18	81.80	71.2	85.38	64.8	**91.8**	80
Cam	-	40.1	0.41	36.57	52.5	67.19	38.4	55.7	**78**
Can	-	48.1	1.35	68.80	**86.1**	80.47	62.9	64.1	86
Cat	-	45.2	0.51	41.82	66.2	60.32	42.7	62.6	**70**
Driller	-	58.6	2.58	63.51	**82.3**	79.79	61.9	74.4	73
Duck	-	32.8	0	27.23	32.5	44.78	30.2	44.3	**66**
Eggbox	-	40.0	8.9	69.58	79.4	96.08	49.9	57.8	**100**
Glue	-	27.0	0	80.02	63.7	87.69	31.2	41.2	**100**
Holep	-	42.4	0.30	42.63	56.4	55.59	52.8	**67.2**	49
Iron	-	67.0	8.86	74.97	65.1	81.75	80.0	**84.7**	78
Lamp	-	39.9	8.20	71.11	**89.4**	86.08	67.0	76.5	73
Phone	-	35.2	0.18	47.74	65.0	65.49	38.1	54.0	**79**
Average	32.3	43.6	2.42	55.95	65.2	71.70	50.2	62.7	**79**

Table 5: Comparison of our approach with state-of-the-art algorithms on LINEMOD in terms of 5 degrees, 5 cm metric. **Bold face** numbers denote the best overall methods.

Method	w/o Refinement		w/ Refinement	
Object	OURS	Deep-6DPose [1]	Brachmann [17]	BB8 [5]
Ape	**89.11**	57.8	34.4	80.2
Benchvise	**88.75**	72.9	40.6	81.5
Cam	**92.09**	75.6	30.5	60.0
Can	**89.94**	70.1	48.4	76.8
Cat	**85.81**	70.3	34.6	79.9
Driller	**80.49**	72.9	54.5	69.6
Duck	**84.14**	67.1	22.0	53.2
Eggbox	**89.27**	68.4	57.1	81.3
Glue	**71.73**	64.6	23.6	54.0
Holepuncher	**83.05**	70.4	47.3	73.1
Iron	**76.31**	60.7	58.7	61.1
Lamp	**84.75**	70.9	49.3	67.5
Phone	**81.44**	69.7	26.8	58.6
Average	**84.38**	68.5	40.6	69.0

Table 6: Comparison of our approach with state-of-the-art algorithms in terms of inference speed.

Method	Overall speed for 1 object	Refinement runtime
Brachmann [17]	2 fps	100 ms/object
BB8 [5]	3 fps	21 ms/object
SSD-6D [3]	10 fps	24 ms/object
PoseCNN [2]	2 fps	24 ms/object
Deep-6DPose [1]	10 fps	-
Tekin [6]	50 fps	-
Heatmap [21]	< 4 fps	-
OURS	25 fps	-



(a) ape (b) can (c) cat (d) driller

(e) duck (f) eggbox (g) glue (h) holepuncher

Figure 6: Comparison with state-of-the-art RGB based methods on the OCCLUSION dataset in terms of 2D reprojection error. We plot percentages of correctly estimated poses as a function of the pixel threshold.

Table 7: Results on the OCCLUSION dataset. **Bold face** numbers denote the best overall methods.

metric	$5cm$ 5°		ADD			Reproj. 5px				
method	[2]	ours	[2]	ours	Heatmap w/o FM [32]	BB8[5]	[2]	ours	Tekin et al. [6]	Heatmap w/o FM [32]
Ape	2.1	23.9	9.6	10.8	14.2	28.5	34.6	61.3	40.4	64.7
Can	4.1	29.9	45.2	39.1	36.9	1.2	15.1	65.8	57.8	53.0
Cat	0.3	9.5	0.9	11.0	8.82	10.4	9.6	49.5	23.3	47.9
Driller	2.5	11.8	41.4	42.5	46.6	0.0	7.4	35.0	17.4	35.1
Duck	1.8	11.7	19.6	18.7	11.1	6.8	31.8	50.0	18.2	36.1
Eggbox	0.0	0.2	22.0	18.4	22.9	-	1.9	2.6	-	10.3
Glue	0.9	8.1	38.5	32.5	39.7	4.7	13.8	39.1	26.9	44.9
Holep.	1.7	14.1	22.1	18.4	20.3	2.4	23.1	56.6	39.5	52.9
Average	1.7	**13.7**	24.9	24.0	**25.1**	7.6	19.46	**45.0**	31.9	43.1

low level features is critical for 6D pose estimation task. As an extension of 2D detection pipeline, proposed network runs fast and can be trained in end-to-end manner. Our approach is able to address textureless objects as well as occlusions between objects. We have proved the effectiveness of proposed approach for 6D pose estimation on two benchmark datasets. Experimental results verify that our method can achieve state-of-the-art pose accuracy in terms of both 2D metrics and 3D metrics.

Acknowledgments

This work was supported in part by the National Natural Science Foundation of China (Grant No. 61501009, 61771031, and 61371134), the National Key Research and Development Program of China (2016YFB0501300 and 2016YFB0501302), and the Fundamental Research Funds for the Central Universities.

References

[1] T.-T. Do, M. Cai, T. Pham, I. Reid, Deep-6DPose: Recovering 6d object pose from a single rgb image, arXiv preprint arXiv:1802.10367.

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

[3] W. Kehl, F. Manhardt, F. Tombari, S. Ilic, N. Navab, SSD-6D: Making rgb-based 3d detection and 6d pose estimation great again, in: 2017 IEEE International Conference on Computer Vision (ICCV), 2017, pp. 1530–1538. doi:10.1109/ICCV.2017.169.

Figure 7: Qualitative 6D pose estimation results on the OCCLUSION dataset. We only draw the 3D bounding boxes rendered by predicted poses. Our approach is robust to partial occlusion and illuminations.

[4] E. Brachmann, A. Krull, S. Nowozin, J. Shotton, F. Michel, S. Gumhold, C. Rother, DSAC: Differentiable ransac for camera localization, in: 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2017, pp. 2492–2500. doi:10.1109/CVPR.2017.267.

[5] M. Rad, V. Lepetit, BB8: A scalable, accurate, robust to partial occlusion method for predicting the 3d poses of challenging objects without using depth, in: 2017 IEEE International Conference on Computer Vision (ICCV), 2017, pp. 3848–3856. doi:10.1109/ICCV.2017.413.

[6] B. Tekin, S. N. Sinha, P. Fua, Real-Time Seamless Single Shot 6D Object Pose Prediction, CVPR.

[7] S. Hinterstoisser, V. Lepetit, S. Ilic, S. Holzer, G. Bradski, K. Konolige, N. Navab, Model based training, detection and pose estimation of texture-less 3d objects in heavily cluttered scenes, in: K. M. Lee, Y. Matsushita, J. M. Rehg, Z. Hu (Eds.), Computer Vision – ACCV 2012, Springer Berlin Heidelberg, Berlin, Heidelberg, 2013, pp. 548–562.

[8] E. Brachmann, A. Krull, F. Michel, S. Gumhold, J. Shotton, C. Rother, Learning 6d object pose estimation using 3d object coordinates, in: D. Fleet, T. Pajdla, B. Schiele, T. Tuytelaars (Eds.), Computer Vision – ECCV 2014, Springer International Publishing, Cham, 2014, pp. 536–551.

[9] S. Hinterstoisser, V. Lepetit, N. Rajkumar, K. Konolige, Going further with point pair features, in: European Conference on Computer Vision, Springer, 2016, pp. 834–848.

[10] J. Vidal, C.-Y. Lin, R. Martí, 6d pose estimation using an improved method based on point pair features, in: 2018 4th International Conference on Control, Automation and Robotics (ICCAR), IEEE, 2018, pp. 405–409.

[11] W. Kehl, F. Milletari, F. Tombari, S. Ilic, N. Navab, Deep learning of local rgb-d patches for 3dobject detection and 6d pose estimation, in: B. Leibe, J. Matas, N. Sebe, M. Welling (Eds.), Computer Vision – ECCV 2016, Springer International Publishing, Cham, 2016, pp. 205–220.

[12] V. Lepetit, F. Moreno-Noguer, P. Fua, EPnP: An accurate o(n) solution to the pnp problem, International Journal of Computer Vision 81 (2) (2008) 155. doi:10.1007/s11263-008-0152-6.

[13] C. P. Lu, G. D. Hager, E. Mjolsness, Fast and globally convergent pose estimation from video images, IEEE Transactions on Pattern Analysis and Machine Intelligence 22 (6) (2000) 610–622. doi:10.1109/34.862199.

[14] A. Rubio, M. Villamizar, L. Ferraz, A. Penate-Sanchez, A. Ramisa, E. Simo-Serra, A. Sanfeliu, F. Moreno-Noguer, Efficient monocular pose estimation for complex 3d models, in: 2015 IEEE International Conference on Robotics and Automation (ICRA), 2015, pp. 1397–1402. doi:10.1109/ICRA.2015.7139372.

[15] L. Svrm, O. Enqvist, M. Oskarsson, F. Kahl, Accurate localization and pose estimation for large 3d models, in: 2014 IEEE Conference on Computer Vision and Pattern Recognition, 2014, pp. 532–539. doi:10.1109/CVPR.2014.75.

[16] Z. Cao, Y. Sheikh, N. K. Banerjee, Real-time scalable 6dof pose estimation for textureless objects, in: 2016 IEEE International Conference on Robotics and Automation (ICRA), 2016, pp. 2441–2448. doi:10.1109/ICRA.2016.7487396.

[17] E. Brachmann, F. Michel, A. Krull, M. Y. Yang, S. Gumhold, C. Rother, Uncertainty-driven 6d pose estimation of objects and scenes from a single rgb image, in: 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016, pp. 3364–3372. doi:10.1109/CVPR.2016.366.

[18] A. Kendall, M. Grimes, R. Cipolla, PoseNet: A convolutional network for real-time 6-dof camera relocalization, in: 2015 IEEE International Conference on Computer Vision (ICCV), 2015, pp. 2938–2946. doi:10.1109/ICCV.2015.336.

[19] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, A. C. Berg, SSD: Single shot multibox detector, in: ECCV, 2016.

[20] M. Sundermeyer, Z.-C. Marton, M. Durner, M. Brucker, R. Triebel, Implicit 3d orientation learning for 6d object detection from rgb images, in: European Conference on Computer Vision, Springer, 2018, pp. 712–729.

[21] M. Oberweger, M. Rad, V. Lepetit, Making deep heatmaps robust to partial occlusions for 3d object pose estimation, European Conference on Computer Vision.

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

[23] S. Tulsiani, J. Malik, Viewpoints and keypoints, in: 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2015, pp. 1510–1519. doi:10.1109/CVPR.2015.7298758.

[24] H. Su, C. R. Qi, Y. Li, L. J. Guibas, Render for CNN: Viewpoint estimation in images using cnns trained with rendered 3d model views, in: 2015 IEEE International Conference on Computer Vision (ICCV), 2015, pp. 2686–2694. doi:10.1109/ICCV.2015.308.

[25] T.-Y. Lin, P. Goyal, R. Girshick, K. He, P. Dollár, Focal loss for dense object detection, in: Computer Vision (ICCV), 2017 IEEE International Conference on, IEEE, 2017, pp. 2999–3007.

[26] T.-Y. Lin, P. Dollár, R. B. Girshick, K. He, B. Hariharan, S. J. Belongie, Feature pyramid networks for object detection., in: CVPR, Vol. 1, 2017, p. 4.

[27] K. He, X. Zhang, S. Ren, J. Sun, Deep residual learning for image recognition, in: 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016, pp. 770–778. doi:10.1109/CVPR.2016.90.

[28] Z. Li, C. Peng, G. Yu, X. Zhang, Y. Deng, J. Sun, Detnet: A backbone network for object detection, IEEE Conference on Computer Vision and Pattern Recognition (CVPR).

[29] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, A. L. Yuille, Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs, IEEE transactions on pattern analysis and machine intelligence 40 (4) (2018) 834–848.

[30] G. Lin, C. Shen, Q. Shi, A. van den Hengel, D. Suter, Fast supervised hashing with decision trees for high-dimensional data, in: 2014 IEEE Conference on Computer Vision and Pattern Recognition, 2014, pp. 1971–1978. doi:10.1109/CVPR.2014.253.

[31] T. Chen, M. Li, Y. Li, M. Lin, N. Wang, M. Wang, T. Xiao, B. Xu, C. Zhang, Z. Zhang, MXNet: A flexible and efficient machine learning library for heterogeneous distributed systems, Neural Information Processing Systems, Workshop on Machine Learning Systems.

[32] M. Rad, M. Oberweger, V. Lepetit, Feature mapping

for learning fast and accurate 3d pose inference from synthetic images, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2018, pp. 4663–4672.