




# Making Deep Heatmaps Robust to Partial Occlusions for 3D Object Pose Estimation

Markus Oberweger<sup>1</sup><sup>(✉)</sup>, Mahdi Rad<sup>1</sup>, and Vincent Lepetit<sup>2,1</sup>

<sup>1</sup> Institute for Computer Graphics and Vision, Graz University of Technology,  
Graz, Austria

{oberweger,rad,lepetit}@icg.tugraz.at

<sup>2</sup> Laboratoire Bordelais de Recherche en Informatique, Université de Bordeaux,  
Bordeaux, France

**Abstract.** We introduce a novel method for robust and accurate 3D object pose estimation from a single color image under large occlusions. Following recent approaches, we first predict the 2D projections of 3D points related to the target object and then compute the 3D pose from these correspondences using a geometric method. Unfortunately, as the results of our experiments show, predicting these 2D projections using a regular CNN or a Convolutional Pose Machine is highly sensitive to partial occlusions, even when these methods are trained with partially occluded examples. Our solution is to predict heatmaps from multiple small patches independently and to accumulate the results to obtain accurate and robust predictions. Training subsequently becomes challenging because patches with similar appearances but different positions on the object correspond to different heatmaps. However, we provide a simple yet effective solution to deal with such ambiguities. We show that our approach outperforms existing methods on two challenging datasets: The Occluded LineMOD dataset and the YCB-Video dataset, both exhibiting cluttered scenes with highly occluded objects.

**Keywords:** 3D object pose estimation · Heatmaps · Occlusions

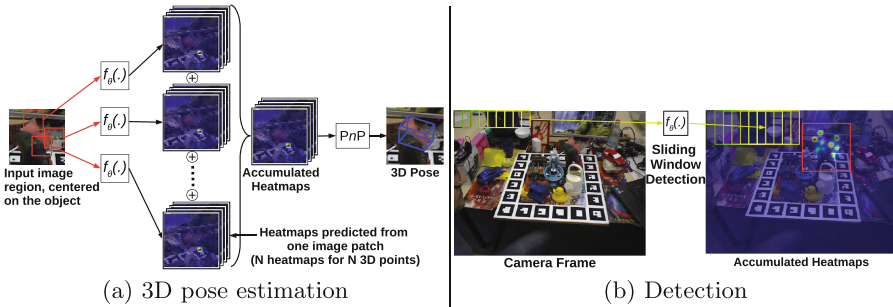
## 1 Introduction

3D object pose estimation from images is an old but currently highly researched topic, mostly due to the advent of Deep Learning-based approaches and the possibility of using large datasets for training such methods. 3D object pose estimation from RGB-D already has provided compelling results [1–4], and the accuracy of methods that only require RGB images recently led to huge progress in the field [2–8]. In particular, one way to obtain an accurate pose is to rely on

---

**Electronic supplementary material** The online version of this chapter ([https://doi.org/10.1007/978-3-030-01267-0\\_8](https://doi.org/10.1007/978-3-030-01267-0_8)) contains supplementary material, which is available to authorized users.

a Deep Network to initially predict the 2D projections of some chosen 3D points and then compute the 3D pose of the object using a PnP method [9]. Such an approach has been shown to be more accurate than the approach of directly predicting the pose used in [5–7], and, therefore, we used the former approach in the research described in this paper.



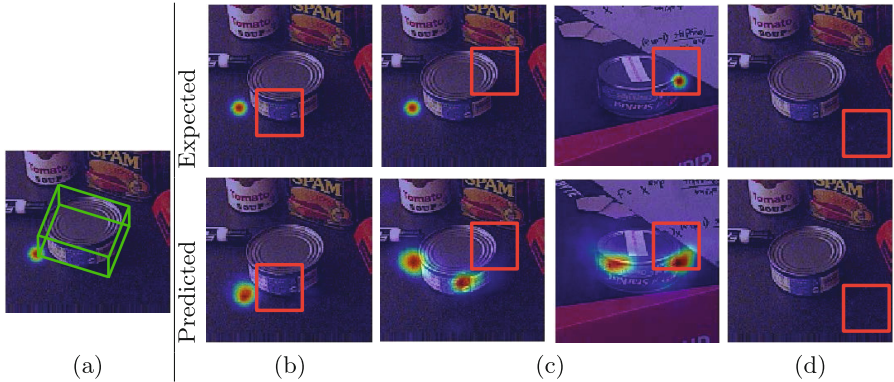
**Fig. 1.** Overview of our method. (a) Given an image region centered on the target object, we sample image patches from which we predict heatmaps for the 2D projections of the corners of the object’s 3D bounding box. This prediction is done by a Deep Network  $f_\theta(\cdot)$ . We aggregate the heatmaps and extract the global maxima for each heatmap, from which we compute the 3D object pose using a PnP algorithm. We show that  $f_\theta(\cdot)$  can be trained simply and efficiently despite the ambiguities that may arise when using small patches as input. (b) To obtain the image region centered on the object, we apply the predictor in a sliding window fashion and accumulate the heatmaps for the full camera frame. We keep the image region with the largest values after accumulation.

However, while Deep Learning methods allow researchers to predict the pose of fully visible objects, they suffer significantly from occlusions, which are very common in practice: Parts of the target object can be hidden by other objects or by a hand interacting with the object. A common *ad hoc* solution is to train the network with occluded objects in the training data. As the results of our experiments presented in this paper show, the presence of large occlusions and unknown occluders still decrease the accuracy of the predicted pose.

Instead of using the entire image of the target object as input to the network, we consider image patches, as illustrated in Fig. 1, since at least some of these are not corrupted by the occluder. Using an image patch as input, our approach learns to predict heatmaps over the 2D projections of 3D points related to the target object. By combining the heatmaps predicted from many patches, we obtain an accurate 3D pose even if some patches actually lie on the occluder or the background instead of on the object.

When moving to an image patch as input, the prediction becomes multi-modal. This is shown in Fig. 2: Some patches may appear on different parts of the target object but look similar. These patches are ambiguous, as they can correspond to different predictions. In such a case, we would like to predict heatmaps

with multiple local maxima, one for each possible prediction. The main challenge is that the ambiguities are difficult to identify: This would require us to identify the patches that have similar appearances, from all the possible viewpoints and at all the possible positions on the object.



**Fig. 2.** Predicting heatmaps from image patches. In this example, we consider predicting the projection of the 3D corner highlighted in (a) for the *tuna fish can* object of the YCB-Video dataset [3]. The red boxes show the input patch of the predicted heatmap. (b) shows a patch from which the projection can be predicted unambiguously. (c) shows two patches that are located in two different positions on the can (notice that the can is flipped and rotated between the two images) while having a similar appearance. In presence of such patches, it is only possible to predict a distribution over the possible locations for the projection. (d) shows a patch on the background, from which we predict a uniform heatmap as it does not provide additional information. See text for details.

The authors of [10] faced a similar problem in the context of 2D object detection when aiming to localize semantic parts from feature vectors of a convolutional layer computed by a CNN. As we discuss in Sect. 2, the method they proposed is complex both for training and inference, and also inaccurate. The solution we propose is much simpler yet efficient: We train a network to predict heatmaps corresponding to a single solution for training image patches using a least-squares loss function. Thanks to the properties of the least-squares loss, this makes the network naturally predict the *average* of the possible heatmap solutions for a given patch. This is exactly what we want, because it is the best information we can obtain from a single patch even if the information remains ambiguous. We then follow an ensemble approach and take the average of the heatmaps predicted for many patches, which allows us to resolve the ambiguities that arise with individual patches. We finally extract the global maximum from this average as the final 2D location.

Our main contribution is, therefore, a simple method that can be used to accurately predict the 3D pose of an object under partial occlusion. We also considered applying Transfer Learning to exploit additional synthetic training data and improve performances. However, as we show, if the input to a network contains an occluder, the occlusion significantly influences the network output even when the network has been trained with occlusion examples and simply adding more training data does not help. In our case, some of the input patches used by our method will not contain occluders, and Transfer Learning becomes useful. In practice, we use the Feature Mapping described in [11], which can be used to map the image features extracted from real images to corresponding image features for synthetic images. This step is not needed for our method to outperform the state-of-the-art but allows us to provide an additional performance boost.

In the remainder of this paper, we first discuss related work, then present our approach, and finally evaluate it and compare it to the state-of-the-art methods on the Occluded LineMOD [12] and the YCB-Video [3] datasets.

## 2 Related Work

The literature on 3D object pose estimation is extremely large. After the popularity of edge-based [13] and keypoint-based methods [14] waned, Machine Learning and Deep Learning became popular in recent years for addressing this problem [2–8, 15]. Here, we will mostly focus on recent work based on RGB images. In the Evaluation section, we compare our method to recent methods [3–5, 7].

[4, 8] proposed a cascade of modules, whereby the first module is used to localize the target objects, and the second module, to regress the object surface 3D coordinates. These coordinates then are used to predict the object pose through hypotheses sampling with a pre-emptive RANSAC [9]. Most importantly, we do not directly predict 3D points but average 2D heatmaps. Predicting 3D points for corresponding 2D points seems to be much more difficult than predicting 2D points for 3D points, as discussed in [3]. Also, surface coordinates are not adapted to deal with symmetric objects. In [5] the target object was also first detected, then the 2D projections of the corners of the object’s 3D bounding boxes were predicted and, finally, the 3D object pose from their 3D correspondences was estimated using a  $PnP$  algorithm. [7] integrated this idea into a recent object detector [16] to predict 2D projections of the corners of the 3D bounding boxes, instead of a 2D bounding box. Similarly, in [6], 2D keypoints were predicted in the form of a set of heatmaps as we do in this work. However, it uses the entire image as input and, thus, performs poorly on occlusions. It also requires training images annotated with keypoint locations, while we use virtual 3D points. In [17], 2D keypoint detection was also relied upon. The authors considered partially occluded objects for inferring the 3D object location from these keypoints. However, their inference adopted a complex model fitting and required the target objects to co-occur in near-regular configuration.

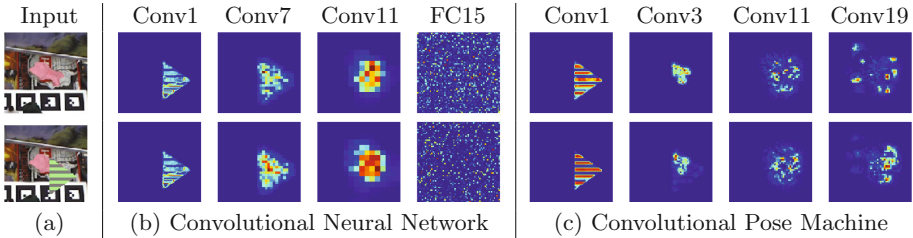
In [2], the SSD architecture [18] was extended to estimate the objects’ 2D locations and 3D rotations. In a next step, the authors used these predictions

together with pre-computed information to estimate the object’s 3D pose. However, this required a refinement step to get an accurate pose, which was influenced by occlusions. The objects were segmented in [3], and an estimate of their 3D poses was made by predicting the translation and a quaternion for the rotation, refined by ICP. Segmenting objects makes their approach robust to occlusions to some extent, however, it requires the use of a highly complex model. In [15], object parts were considered to handle partial occlusion by predicting a set of 2D-3D correspondences from each of these parts. However, the parts had to be manually picked, and it is not clear which parts can represent objects such as those we evaluate in this paper.

As mentioned in the introduction, our method is related to that described in [10]. In the context of 2D object detection, semantic parts are localized as in [10] from neighboring feature vectors using a spatial offset map. The offset maps are accumulated in a training phase. However, they need to be able to identify which feature vectors support a semantic part from these maps, and complex statistical measures are used to identify such vectors. Our method is significantly simpler, as the mapping between the input patches and the 2D projections does not have to be established explicitly.

The authors of [19] already evaluated CNNs trained on occlusions in the context of 2D object detection and recognition and proposed modifying training to penalize large spatial filters support. This yields better performance; however, this does not fully cancel out the influence of occlusions. Some recent work also describes explicitly how to handle occlusions for 3D pose estimation when dealing with 3D or RGB-D data: Like us, [20] relied on a voting scheme to increase robustness to occlusions; [21] first segmented and identified the objects from an RGB-D image. They then performed an extensive randomized search over possible object poses by considering physical simulation of the configuration. In [22], holistic and local patches were combined for object pose estimation, using a codebook for local patches and applying a nearest-neighbor search to find similar poses, as in [23,24]. In contrast to these methods, we use only color images.

Our method is also related to ensemble methods and, in particular, the Hough Forests [25], which are based on regression trees. Hough Forests also predict 2D locations from multiple patches and are multimodal. Multimodal prediction is easy to perform with trees, as the multiple solutions can be stored in the tree leaves. With our method, we aim to combine the ability of Hough Forests for multimodal predictions and the learning power of Deep Learning. [26] already reformulated a Hough Forest as a CNN by predicting classification and regression for patches of the input image. However, this method required to handle the detection separately, and each patch regressed a single vector, which was not multimodal and required clustering of the predicted vectors. In this paper, we show that carrying out a multimodal prediction with Deep Networks to address our problem is, in fact, simple.



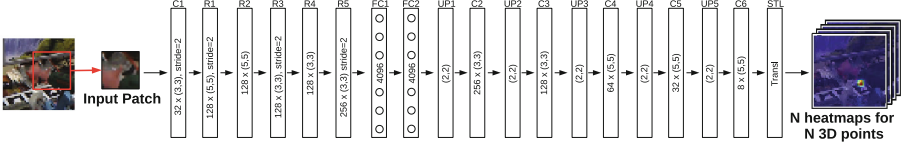
**Fig. 3.** Effect of occlusions on the feature maps of Deep Networks. (a) Input image without (top) and with (bottom) partial occlusion. (b-Top) Sums of absolute differences between the feature maps with and without occlusions for a CNN trained without occlusions. (b-Bottom) Same when the network is trained with occlusion examples. (c) Same for a Convolutional Pose Machine. The influence of the occlusion increases with the layers’ depths, as receptive fields are larger in the deeper layers than in the first layers, even when the method is trained with occlusion examples. For more details we refer to the supplementary material.

### 3 Influence of Occlusions on Deep Networks

In this section, we describe how we evaluate how much a partial occlusion influences a Deep Network, whether it is a standard Convolutional Neural Network (CNN) or a Convolutional Pose Machine (CPM) [27]. Specifically, a CPM is a carefully designed CNN that predicts dense heatmaps by sequentially refining results from previous stages. The input features are concatenated to intermediary heatmaps in order to learn spatial dependencies.

For this experiment, depicted in Fig. 3, we use an image centered on an object as input to a network—here, the *Cat* object from the Occluded LineMOD dataset [12]. We then compare the layer activations in the absence of occlusion, and when the object is occluded by an artificial object (here, a striped triangle). We consider two networks: A standard CNN trained to predict the 2D projections of 3D points as a vector [5], and a CPM [27] with 3 stages trained to predict a heatmap for each of the same 2D projections. For the 3D points, we use the corners of the 3D bounding box of the object.

As can be seen in Fig. 3, the occlusion induces changes in the activations of all the layers of both networks. For a standard CNN, the occlusion spreads to more than 20% in the last feature map, and, beyond the first fully-connected layer, more than 45% of all activations are changed. In this case, all the predictions for the 2D projections, occluded or not, are inaccurate. A similar effect can be observed for CPMs: Here, the altered activations are more specifically localized to the occluded region due to the convolutions, with more than 29% of the activations changed in the last feature map. In this case, the predictions of the 2D projections are inaccurate when the 3D points are occluded. When the 3D points are not occluded, the predicted projections are sometimes correct, because the influence of the occluder spreads less with a CPM than with a standard CNN.



**Fig. 4.** Network architecture for  $f_\theta(\cdot)$ . C denotes a convolutional layer with the number of filters and the filter size inscribed; FC, a fully-connected layer with the number of neurons; UP, an unpooling layer [28]; R, a residual module [29] with the number of filters and filter size; and STL, a Spatial Transformation Layer [30] used for translation. All layers have ReLU activations, and the output of the last layer is linear.

## 4 Minimizing the Effect of Occlusions

In this section, we first describe our training procedure given an input image region centered on the object, then the run-time inference of the pose. Finally, we explain how we identify the input image region in practice.

### 4.1 Training

Datasets for 3D pose estimation typically provide training images annotated with the objects’ poses and the 3D models of the objects. From this data, we generate our training set  $\{(I^{(i)}, \{\mathbf{p}_j^{(i)}\}_j, M^{(i)})\}_i$ , where  $I^{(i)}$  is the  $i$ -th training image;  $\mathbf{p}_j^{(i)}$ , the 2D projection of the  $j$ -th 3D corner; and  $M^{(i)}$ , the 2D mask of the object in image  $I^{(i)}$ . This mask can be obtained by projecting the 3D object model into the image using the object’s pose.

**The Unambiguous Case.** Let us first ignore the fact that some image patches can be ambiguous and that the learning problem is actually multimodal. We train a network  $f_\theta(\cdot)$  to predict a heatmap for each projection  $\mathbf{p}_j$ . The architecture we use for this network is shown in Fig. 4.  $f_\theta(\cdot)$  takes an input patch of size  $32 \times 32px$ , and predicts a set of heatmaps of size  $128 \times 128px$ , and we train it by minimizing:

$$\min_{\theta} \sum_i \sum_{u,v} \|\mathcal{H}^{(i)} - \text{Transl}(f_\theta(\mathcal{P}(I^{(i)}, u, v)), -u, -v)\|^2, \quad (1)$$

where:

- $\mathcal{P}(I^{(i)}, u, v)$  is an image patch centered on location  $(u, v)$  in image  $I^{(i)}$ ;
- $\mathcal{H}^{(i)}$  is the set of expected heatmaps for  $\mathcal{P}(I^{(i)}, u, v)$ . It contains one heatmap for each 2D projection  $\mathbf{p}_j^{(i)}$ . We describe how  $\mathcal{H}^{(i)}$  is defined in detail below;
- $f_\theta(\mathcal{P})$  returns a set of heatmaps, one for each 2D projection  $\mathbf{p}_j^{(i)}$ .
- $\text{Transl}(H, -u, -v)$  translates the predicted heatmaps  $H$  by  $(-u, -v)$ .  $f_\theta(\cdot)$  learns to predict the heatmaps with respect to the patch center  $(u, v)$ , and

this translation is required to correctly align the predicted heatmaps together. Such a translation can be efficiently implemented using a Spatial Transformation Layer [30], which makes the network trainable end-to-end.

The sum  $\sum_{(u,v)}$  is over 2D locations randomly sampled from the image. The heatmaps in  $\mathcal{H}^{(i)}$  are defined as a Gaussian distribution with a small standard deviation (we use  $\sigma = 4px$  in practice) and centered on the expected 2D projections  $\mathbf{p}_j^{(i)}$  when patch  $\mathcal{P}(I^{(i)}, u, v)$  overlaps the object mask  $M^{(i)}$ . The top row of Fig. 2 shows examples of such heatmaps.

When the patch does not overlap the object mask, the heatmaps in  $\mathcal{H}^{(i)}$  are defined as a uniform distribution of value  $\frac{1}{W \cdot H}$ , where  $W \times H$  is the heatmap’s resolution, since there is no information in the patch to predict the 2D projections. In addition, we use patches sampled from the ImageNet dataset [31] and train the network to predict uniform heatmaps as well for these patches. Considering these patches (outside the object’s mask or from ImageNet) during training allows us to correctly handle patches appearing in the background or on the occluders and significantly reduces the number of false positives observed at run-time.

**The Multimodal Case.** Let us now consider the real problem, where the prediction is multimodal: Two image patches such as the ones shown in Fig. 2(c) can be similar but extracted from different training images and, therefore, correspond to different expected heatmaps. In other words, in our training set, we can have values for samples  $i, i'$  and locations  $(u, v)$  and  $(u', v')$  such that  $\mathcal{P}(I^{(i)}, u, v) \approx \mathcal{P}(I^{(i')}, u', v')$  and  $\mathcal{H}^{(i)} \neq \mathcal{H}^{(i')}$ .

It may seem as though, in this case, training given by Eq. (1) would fail or need to be modified. *In fact, Eq. (1) remains valid.* This is because we use the least-squares loss function: For image patches with similar appearances that correspond to different possible heatmaps,  $f_\theta(\cdot)$  will learn to predict the average of these heatmaps, which is exactly what we want. The bottom row of Fig. 2 shows such heatmaps. At run-time, because we will combine the contribution of multiple image patches, we will be able to resolve the ambiguities.

## 4.2 Run-Time Inference

At run-time, given an input image  $I$ , we extract patches from randomly selected locations from the input image and feed them into the predictor  $f_\theta(\cdot)$ . To combine the contributions of the different patches, we use a simple ensemble approach and average the predicted heatmaps for each 2D projection. We take the locations of the global maxima after averaging them as the final predictions for the 2D projections.

More formally, the final prediction  $\widetilde{\mathbf{p}}_j$  for the 2D projection  $\mathbf{p}_j$  is the location of the global maximum of  $\sum_{u,v} \text{Transl}(f_\theta(\mathcal{P}(I, u, v)), -u, -v)[j]$ , the sum of the heatmaps predicted for the  $j$ -th projection, translated such that these heatmaps align correctly. The sum is performed over randomly sampled patches. An evaluation of the effect of the number of samples is provided in the supplementary



material. To compute the pose, we use a  $PnP$  estimation with RANSAC [9] on the correspondences between the corners of the object’s 3D bounding box and the  $\widetilde{\mathbf{p}}_j$  locations.

### 4.3 Two-Step Procedure

In practice, we first estimate the 2D location of the object of interest, using the same method as in the previous subsection, but instead of sampling random locations, we apply the network  $f_\theta(\cdot)$  in a sliding window fashion, as illustrated in Fig. 1 (b). For each image location, we compute a score by summing up the heatmap values over a bounding box of size  $128 \times 128$  and over the 8 corners for each object, which is done efficiently using integral images. We apply Gaussian smoothing and thresholding to the resulting score map. We use the centers-of-mass of the regions after thresholding as the centers of the input image  $I$ . Finally, we use this image as input to the method described in the previous subsection. We use a fixed size for this region as our method is robust to scale changes.

## 5 Evaluation

In this section, we evaluate our method and compare it to the state-of-the-art. For this, we use two datasets: The Occluded LineMOD dataset [12], and the YCB-Video dataset [3]. Both datasets contain challenging sequences with partially occluded objects and cluttered backgrounds. In the following, we first provide the implementation details, the evaluation metrics used and then present the results of evaluation of the two datasets, including the results of an ablative analysis of our method.

### 5.1 Implementation Details

*Training Data:* The training data consist of real and synthetic images with annotated 3D poses and object masks, as was also the case in [3]. To render the synthetic objects, we use the models that are provided with the datasets. We crop the objects of interest from the training images and paste them onto random backgrounds [32] sampled from ImageNet [31] to achieve invariance to different backgrounds. We augment the dataset with small affine perturbations in HSV color space.

*Network Training:* The network is optimized using ADAM [33] with default parameters and using a minibatch size of 64, a learning rate of 0.001, and 100k iterations. We train one network per object starting from a random initialization.

*Symmetric Objects:* We adapt the heatmap generation to symmetric objects present in the two datasets. For rotationally symmetric objects, *e.g.*, cylindrical shapes, we only predict a single position around the rotation axis. For mirror-symmetric objects, we only train on half the range of the symmetry axis, as was performed in [5].

*Feature Mapping:* Optionally, we apply the Feature Mapping method as described in [11] to compensate for a lack of real training data. We apply the mapping between the FC1 and FC2 layers shown in Fig. 4. The mapping network uses the same architecture as described in [11], but the weight for the feature loss is significantly lower ( $10^{-5}$ ).

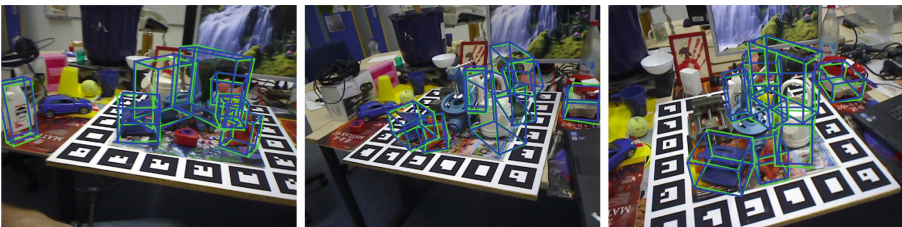
## 5.2 Evaluation Metrics

We consider the most common metrics. The 2D Reprojection error [8] computes the distances between the projections of the 3D model points when projected using the ground truth pose, and when using the predicted pose. The ADD metric [34] calculates the average distance in 3D between the model points, after applying the ground truth pose and the predicted pose. For symmetric objects, the 3D distances are calculated between the closest 3D points, denoted as the ADI metric. Below, we refer to these two metrics as  $AD\{D|I\}$  and use the one appropriate to the object. The exact formulas for these metrics are provided in the supplementary material.

## 5.3 Occluded LineMOD Dataset

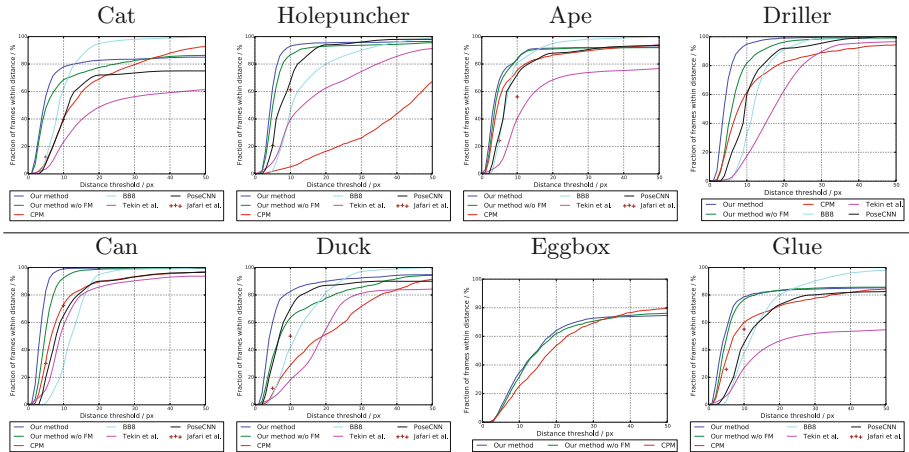
The Occluded LineMOD dataset [12] consists of a sequence of 1215 frames, each frame labeled with the 3D poses of 8 objects as well as object masks. The objects show severe occlusions, which makes pose estimation extremely challenging. The sequences were captured using an RGB-D camera with  $640 \times 480px$  images, however, we use *only the color images* for our method and all results reported.

For training the heatmap predictors, we use the LineMOD dataset [34] that contains the same objects as the Occluded LineMOD dataset. This protocol is commonly used for the dataset [3–5,7], since the Occluded LineMOD dataset only contains testing sequences. Figure 5 shows some of the qualitative results obtained. We give an extensive quantitative evaluation in the following section.



**Fig. 5.** Some qualitative results on the Occluded LineMOD dataset [12]. We show the 3D bounding boxes of the objects projected onto the color image. Ground truth poses are shown in green, and our predictions are shown in blue. More results are provided in the supplementary material.

**Quantitative Results.** Figure 6 shows the fraction of frames where the 2D Reprojection error is smaller than a given threshold, for each of the 8 objects from the dataset. A larger area under the curve denotes better results. We compare these results to those obtained from the use of several recent methods that also work only with color images, namely, BB8 [5], PoseCNN [3], Jafari et al. [4], and Tekin et al. [7]. Note that the method described in [5] uses ground truth detection, whereas ours does not. Our method performs significantly more accurately on all sequences. Notably, we also provide results for the *Eggbox* object, which, so far, was not considered since it was too difficult to learn for [4, 5, 7].



**Fig. 6.** Evaluation on the Occluded LineMOD dataset [12] using color images only. We plot the fraction of frames for which the 2D Reprojection error is smaller than the threshold on the horizontal axis. Our method provides results that significantly outperform those reported by previous work. “w/o FM” denotes without Feature Mapping.

Adding Feature Mapping [11] improves the 2D Reprojection error for a threshold of  $5px$  by 17% on average. We also tried Feature Mapping for the approach of [5], but it did not improve the results because the occlusions influence the feature maps too greatly when the network input contains occluders, as already discussed in the introduction.

Further quantitative results are given in Table 1, where we provide the percentage of frames for which the ADD or ADI metric is smaller than 10% of the object diameter, as [3] reported such results on the Occluded LineMOD dataset. This is considered a highly challenging metric. We also give the percentage of frames that have a 2D Reprojection error of less than  $5px$ . Our method significantly outperforms all other methods on these metrics by a large margin.

**Table 1.** Comparison on the Occluded LineMOD dataset [12] with color images only. We provide the percentage of frames for which the AD{D|I} error is smaller than 10% of the object diameter, and for which the 2D Reprojection error is smaller than 5px. Objects marked with a \* are considered to be symmetric.

Method	AD{D I}-10%						2D Reprojection Error-5px											
	Ape	Can	Cat	Driller	Duck	Eggbox*	Glue*	Holepun.	Average	Ape	Can	Cat	Driller	Duck	Eggbox*	Glue*	Holepun.	Average
PoseCNN [3]	9.6	45.2	0.93	41.4	<b>19.6</b>	22.0	38.5	<b>22.1</b>	24.9	34.6	15.1	10.4	7.40	31.8	1.90	13.8	23.1	17.2
Tekin et al. [7]	-	-	-	-	-	-	-	-	-	7.01	11.2	3.62	1.40	5.07	-	6.53	8.26	6.16
BB8 [5]	-	-	-	-	-	-	-	-	-	28.5	1.20	9.60	0.00	6.80	-	4.70	2.40	7.60
Jafari et al. [4]	-	-	-	-	-	-	-	-	-	24.2	30.2	12.3	-	12.1	-	25.9	20.6	20.8
CPM [27]	12.5	25.6	1.43	23.8	6.99	18.3	15.0	0.74	13.0	55.4	30.6	15.7	27.9	26.6	7.97	18.5	1.81	23.1
Our method w/o FM	16.5	42.5	2.82	47.1	11.0	24.7	39.5	21.9	25.8	64.7	53.0	47.9	35.1	36.1	10.3	44.9	52.9	43.1
Our method	<b>17.6</b>	<b>53.9</b>	<b>3.31</b>	<b>62.4</b>	<b>19.2</b>	<b>25.9</b>	<b>39.6</b>	21.3	<b>30.4</b>	<b>69.6</b>	<b>82.6</b>	<b>65.1</b>	<b>73.8</b>	<b>61.4</b>	<b>13.1</b>	<b>54.9</b>	<b>66.4</b>	<b>60.9</b>

**The Effect of Seeing Occlusions During Training.** We evaluate the importance of knowing the occluder in advance. [3, 5, 7] assumed that the occluder is another object from the LineMOD dataset and only used occlusions from these objects during training. However, in practice, this assumption does not hold, since the occluder can be an arbitrary object. Therefore, we investigated how the performance was affected by the use of occlusions during training.

We compare our results (without Feature Mapping) to two state-of-the-art approaches: our reimplementations of BB8 [5] and CPM [27]. To avoid bias introduced by the limited amount of training data in the Occluded LineMOD dataset [12], we consider synthetic images both for training and for testing here.

We investigate three different training schemes: (a) No occlusions used for training; (b) random occlusions by simple geometric shapes; (c) random occlusions with the same objects from the dataset, as described in [4, 5, 7]. We compare the different training schemes in Fig. 7. Training without occlusions clearly result in worse performance for BB8 and CPM, whereas our method is significantly more robust. Adding random geometric occlusions during training slightly increases the performance of BB8 and CPM, since the networks learn invariance to occlusions, however, mainly for these specific occlusions, whereas our approach maintains the accuracy compared to training without occlusions. Using occluders from the dataset gives the best results, since the networks learn to ignore specific features from these occluders. This, however, is only possible when the occluders are known in advance, which is not necessarily the case in practice.

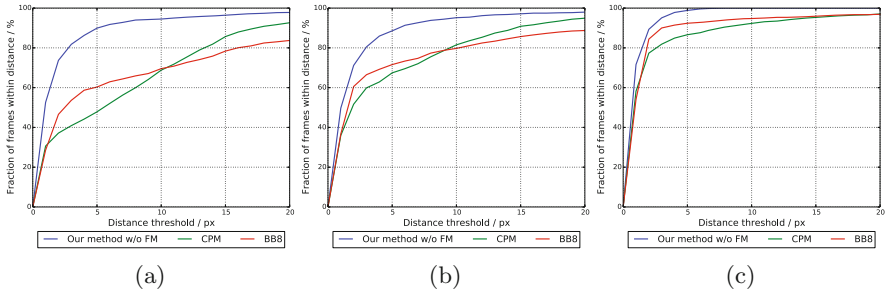
**Patch Size and Number of Patches.** We evaluated the influence of the patch size on the predicted pose accuracy. There is a range of sizes (25px to 40px) for which the performances stay very close to those presented in Table 1. Small patches seem to lack discriminative power, the 2D Reprojection metric gets 19% worse with 8px patches, and large patches are sensitive to occlusions, which leads to a decrease in the 2D Reprojection metric of 5% for 128px patches.

In the supplementary material, we provide a detail study of the influence of the number of patches on the predicted pose accuracy. The main conclusions are that the accuracy starts to flatten when more than 64 patches are used, and that — if a preprocessing algorithm could be used to provide a segmentation mask — we could reduce the number of patches to achieve the same level of accuracy.

**Runtime.** We implemented our method in Python on an Intel i7 with 3.2 GHz and 64GB of RAM, using an nVidia GTX 980 Ti graphics card. Pose estimation is 100ms for 64 patches, and detection takes 150ms on a  $640 \times 480$  camera frame. Predicting the heatmaps for a single patch takes 4ms, and the total runtime could, thus, be significantly reduced by processing the individual patches in parallel.

#### 5.4 YCB-Video Dataset

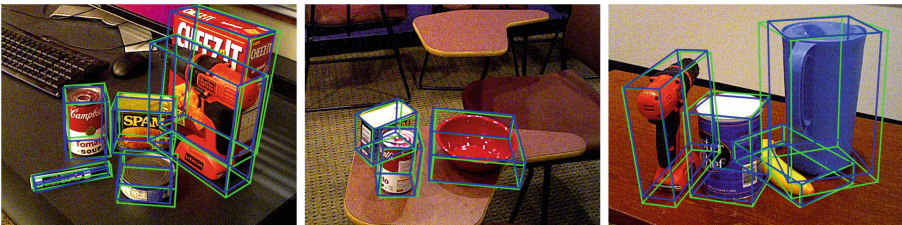
The recently proposed YCB-Video dataset [3] consists of 92 video sequences, where 12 sequences are used for testing and the remaining 80 sequences for



**Fig. 7.** Evaluation of synthetic renderings of scenes from the Occluded LineMOD dataset (see text) using the 2D Reprojection error. (a) Training without occlusions; (b) training with random geometric occlusions; and (c) training with occluding objects from the LineMOD dataset [34]. Knowing the occluders in advance significantly improves the performances of BB8 [5] and CPM [27], however, this knowledge is often not available in practice. Our method does not require this knowledge.

training. In addition, the dataset contains 80k synthetically rendered images, which can be used for training as well. There are 21 objects in the dataset, which are taken from the YCB dataset [35] and are publicly available for purchase. The dataset is captured with two different RGB-D sensors, each providing  $640 \times 480$  images, but we only use the color images. The test images are extremely challenging due to the presence of significant image noise and different illumination levels. Each image is annotated with the 3D object poses, as well as the objects’ masks. Figure 8 shows some qualitative results. We give an extensive quantitative evaluation in the following section.

**Quantitative Results.** We provide the 2D Reprojection error and the  $AD\{D|I\}$  metrics averaged over all the objects in Table 2. In [3], the area under the accuracy-threshold curve was used as a metric, which we also provide.<sup>1</sup> Again, our approach results in better performance according to these metrics.



**Fig. 8.** Qualitative results on the YCB-Video dataset [3]. The green bounding boxes correspond to the ground truth poses, the blue ones to our estimated poses. More results are provided in the supplementary material.

<sup>1</sup> The metrics are calculated from the results provided by the authors at their website.

**Table 2.** Comparison on the YCB-Video dataset [3]. We refer to the supplementary material for the object-specific numbers and additional plots. Our method clearly outperforms the baseline.

Method	PoseCNN [3]			Our method w/o FM			Our method		
	AUC	AD{D I}-10%	2D Repr-5px	AUC	AD{D I}-10%	2D Repr-5px	AUC	AD{D I}-10%	2D Repr-5px
Average	61.0	21.3	3.72	<i>61.4</i>	<i>33.6</i>	<i>23.1</i>	<b>72.8</b>	<b>53.1</b>	<b>39.4</b>

## 6 Discussion and Conclusion

In this paper, we introduced a novel method for 3D object pose estimation that is inherently robust to partial occlusions of the object. To do this, we considered only small image patches as input and merged their contributions. Because we chose to compute the pose by initially predicting the 2D projections of 3D points related to the object, the prediction can be performed in the form of 2D heatmaps. Since heatmaps are closely related to density functions, they can be conveniently applied to capture the ambiguities that arise when using small image patches as input. We showed that training a network to predict the heatmaps in the presence of such ambiguities is much simpler than it may sound. This resulted in a simple pipeline, which outperformed much more complex methods on two challenging datasets.

Our approach can be extended in different ways. The heatmaps could be merged in a way that is more robust to erroneous values than simple averaging. The pose could be estimated by considering the best local maxima rather than only the global maxima. Sampling only patches intersecting with the object mask, which could be predicted by a segmentation method, would limit the influence of occluders and background in the accumulated heatmaps even more. Predicting the heatmaps could be performed in parallel.

**Acknowledgment.** This work was funded by the Christian Doppler Laboratory for Semantic 3D Computer Vision. We would like to thank Yu Xiang for providing additional results.

## References

1. Krull, A., Brachmann, E., Michel, F., Yang, M.Y., Gumhold, S., Rother, C.: Learning analysis-by-synthesis for 6D pose estimation in RGB-D images. In: International Conference on Computer Vision (2015)
2. Kehl, W., Manhardt, F., Tombari, F., Ilic, S., Navab, N.: SSD-6D: making RGB-based 3D detection and 6D pose estimation great again. In: International Conference on Computer Vision (2017)
3. Xiang, Y., Schmidt, T., Narayanan, V., Fox, D.: PoseCNN: a convolutional neural network for 6D object pose estimation in cluttered scenes. In: Robotics: Science and Systems Conference (2018)

4. Jafari, O.H., Mustikovela, S.K., Pertsch, K., Brachmann, E., Rother, C.: iPose: instance-aware 6D pose estimation of partly occluded objects. CoRR [abs/1712.01924](#) (2017)
5. Rad, M., Lepetit, V.: BB8: a scalable, accurate, robust to partial occlusion method for predicting the 3D poses of challenging objects without using depth. In: International Conference on Computer Vision (2017)
6. Pavlakos, G., Zhou, X., Chan, A., Derpanis, K.G., Daniilidis, K.: 6-DoF object pose from semantic Keypoints. In: International Conference on Intelligent Robots and Systems (2018)
7. Tekin, B., Sinha, S.N., Fua, P.: Real-time seamless single shot 6D object pose prediction. In: Conference on Computer Vision and Pattern Recognition (2018)
8. Brachmann, E., Michel, F., Krull, A., Yang, M.M., Gumhold, S., Rother, C.: Uncertainty-Driven 6D pose estimation of objects and scenes from a single RGB image. In: Conference on Computer Vision and Pattern Recognition (2016)
9. Hartley, R., Zisserman, A.: Multiple View Geometry in Computer Vision. Cambridge University Press, New York (2000)
10. Wang, J., Xie, Q., Zhang, Z., Zhu, J., Xie, L., Yuille, A.L.: Detecting semantic parts on partially occluded objects. In: British Machine Vision Conference (2017)
11. Rad, M., Oberweger, M., Lepetit, V.: Feature mapping for learning fast and accurate 3D pose inference from synthetic images. In: Conference on Computer Vision and Pattern Recognition (2018)
12. Brachmann, E., Krull, A., Michel, F., Gumhold, S., Shotton, J., Rother, C.: Learning 6D object pose estimation using 3D object coordinates. In: European Conference on Computer Vision (2014)
13. Harris, C., Stennett, C.: RAPID—a video rate object tracker. In: British Machine Vision Conference (1990)
14. Lowe, D.: Distinctive image features from scale-invariant keypoints. *Int. J. Comput. Vis.* **20**(2), 91–110 (2004)
15. Crivellaro, A., Rad, M., Verdie, Y., Yi, K., Fua, P., Lepetit, V.: Robust 3D object tracking from monocular images using stable parts. *IEEE Trans. Pattern Anal. Mach. Intell.* **40**, 1465–1479 (2017)
16. Redmon, J., Divvala, S., Girshick, R., Farhadi, A.: You only look once: unified, real-time object detection. In: Conference on Computer Vision and Pattern Recognition (2016)
17. Hueting, M., Reddy, P., Kim, V., Carr, N., Yumer, E., Mitra, N.: SeeThrough: finding chairs in heavily occluded indoor scene images. CoRR [abs/1710.10473](#) (2017)
18. Liu, W. et al.: SSD: single shot multibox detector. In: European Conference on Computer Vision (2016)
19. Osherov, E., Lindenbaum, M.: Increasing CNN robustness to occlusions by reducing filter support. In: International Conference on Computer Vision (2017)
20. Buch, A.G., Kiforenko, L., Kraft, D.: Rotational subgroup voting and pose clustering for robust 3D object recognition. In: International Conference on Computer Vision (2017)
21. Mitash, C., Boularias, A., Bekris, K.E.: Improving 6D pose estimation of objects in clutter via physics-aware monte carlo tree search. In: International Conference on Robotics and Automation (2018)
22. Zhang, H., Cao, Q.: Combined holistic and local patches for recovering 6D object pose. In: International Conference on Computer Vision Workshops (2017)
23. Doumanoglou, A., Balntas, V., Kouskouridas, R., Kim, T.: Siamese regression networks with efficient mid-level feature extraction for 3D object pose estimation. CoRR [abs/1607.02257](#) (2016)



24. Kehl, W., Milletari, F., Tombari, F., Ilic, S., Navab, N.: Deep Learning of local RGB-D patches for 3D object detection and 6D pose estimation. In: European Conference on Computer Vision (2016)
25. Gall, J., Lempitsky, V.: Class-specific hough forests for object detection. In: Conference on Computer Vision and Pattern Recognition (2009)
26. Riegler, G., Ferstl, D., Rüther, M., Bischof, H.: Hough networks for head pose estimation and facial feature localization. In: British Machine Vision Conference (2014)
27. Wei, S.E., Ramakrishna, V., Kanade, T., Sheikh, Y.: Convolutional pose machines. In: Conference on Computer Vision and Pattern Recognition (2016)
28. Zeiler, M., Krishnan, D., Taylor, G., Fergus, R.: Deconvolutional networks. In: Conference on Computer Vision and Pattern Recognition (2010)
29. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Conference on Computer Vision and Pattern Recognition (2016)
30. Jaderberg, M., Simonyan, K., Zisserman, A., Kavukcuoglu, K.: Spatial transformer networks. In: Advances in Neural Information Processing Systems (2015)
31. Deng, J., Dong, W., Socher, R., Li, L.J., Li, K., Fei-Fei, L.: ImageNet: a large-scale hierarchical image database. In: Conference on Computer Vision and Pattern Recognition (2009)
32. Dwibedi, D., Misra, I., Hebert, M.: Cut, paste and learn: surprisingly easy synthesis for instance detection. In: International Conference on Computer Vision (2017)
33. Kingma, D.P., Ba, J.: Adam: a method for stochastic optimization. In: International Conference on Machine Learning (2015)
34. Hinterstoisser, S. et al.: Model based training, detection and pose estimation of texture-less 3D objects in heavily cluttered scenes. In: Asian Conference on Computer Vision (2012)
35. Calli, B., et al.: Yale-CMU-Berkeley dataset for robotic manipulation research. *Int. J. Robot. Res.* **36**, 261–268 (2017)