The Best of Both Worlds: Learning Geometry-based 6D Object Pose Estimation

Omid Hosseini Jafari*, Siva Karthik Mustikovela*, Karl Pertsch, Eric Brachmann, Carsten Rother
Heidelberg University

Abstract

We address the task of estimating the 6D pose of known rigid objects, from RGB and RGB-D input images, in scenarios where the objects are heavily occluded. Our main contribution is a new modular processing pipeline. The first module localizes all known objects in the image via an existing instance segmentation network. The next module densely regresses the object surface positions in its local coordinate system, using an encoder-decoder network. The third module is purely a geometry-based algorithm to output the final 6D object poses. While the first two modules are learned from data, and the last one not, we believe that this is the best of both worlds: geometry-based and learning-based algorithms for object 6D pose estimation. This is validated by achieving state-of-the-art results for RGB input and a slight improvement over state-of-the-art for RGB-D input. However, in contrast to previous work, we achieve these results with the same pipeline for RGB and RGB-D input. Furthermore, to obtain these results, we give a second contribution of a new 3D occlusion-aware and object-centric data augmentation procedure.

1. Introduction

One-shot localization of object instances has been a long-standing goal in computer vision. As the field progressed, the task evolved from simple 2D detection to full 6D pose estimation consisting of 3D position and 3D orientation relative to the observing camera. Early approaches relied on objects having sufficient texture to match salient points [24]. Later, with the advent of consumer depth cameras [31], research focused on texture-less objects [13] in increasingly cluttered environments. Today, heavy occlusion of objects is the main performance benchmark for one-shot object pose estimation methods. Solving this problem is highly relevant in applications like augmented reality or robotics.

While recent RGB-D based pose estimation approaches are robust to moderate degrees of occlusion [25, 12], achieving the same reliability for RGB input is an open research problem. Since depth sensors fail in certain environments, *e.g*. under intense sunlight, and RGB cameras are prevalent on many types of devices, RGB-based methods have high practical relevancy. In this work, we present a system for 6D pose estimation of rigid object instances from single image input, both RGB-D and RGB, with strong emphasis on object occlusion.

At the core, our method strictly decomposes the task of 6D pose estimation into a sequence of three sub-tasks, or modules, which can be solved independently taking advantage of reduced complexity. We first detect the object in 2D, then we locally reconstruct the 3D object surface, and, finally, we estimate the 6D pose of the object. In an abstract view, while the dimensionality of the output for each sub-task grows, we can remove specific aspects of the problem with each task, such as object-background and object-appearance. In the first module, the 2D detection is implemented by an existing instance segmentation component which estimates a tight mask for each object. Thus, we can separate the object from the surrounding clutter, and occluding objects, making the following surface reconstruction step invariant to object background. In the second module, we present an encoder-decoder architecture for surface reconstruction which densely regresses so-called *object coordinates* [2], *i.e*. 3D points in the local coordinate frame of the object. This reduces the final 6D pose estimation step to a purely geometric optimization task, which is done in the final module. While the first two modules are learned from data, and the last one not, we believe that this is the best of both worlds: geometry-based and learning-based algorithms.

While some prior work shares ingredients with our approach, there are notable differences. Firstly, there is a body of work [2, 22, 4, 25] which also regresses object-coordinates from image data, and successively uses it to predict the 6D object pose. In contrast to our work, they combine object coordinates prediction and object segmentation in a single module, using random forests. These two tasks are disentangled in our approach, with the clear advantage that each individual object-background mask is known for object-coordinate regression. Secondly, Rad *et al*. [27] recently presented the BB8 pipeline for object pose estima-

*Equal contribution

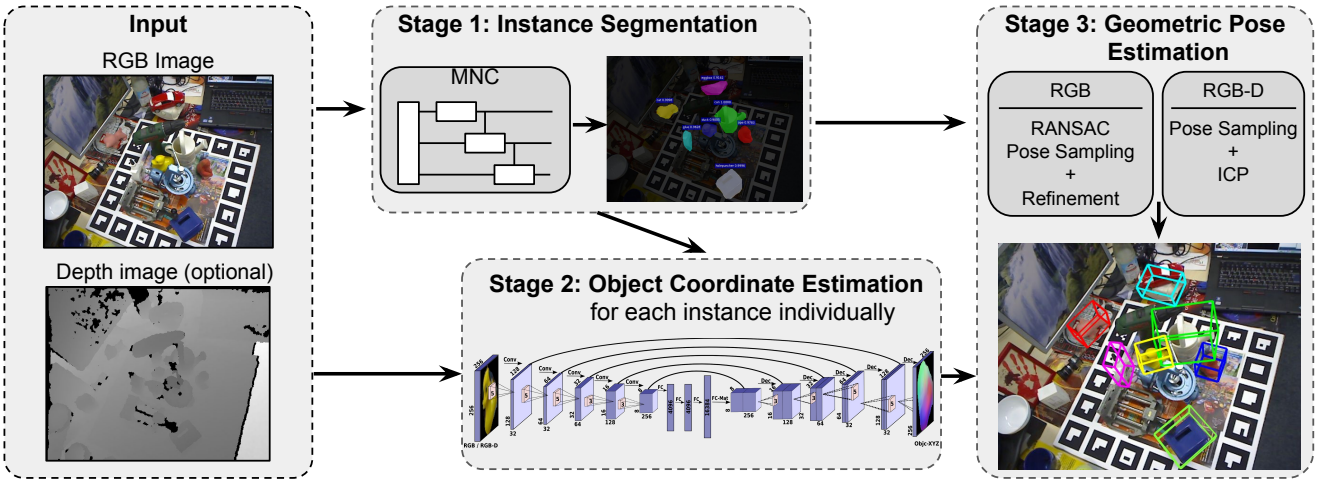

Figure 1. **Illustration of our modular, 3-stage pipeline** for both RGB and RGBD input images.

tion from RGB images. While their architecture resembles our decomposition philosophy to some extent, their processing steps are more tightly coupled. For example, their detection stage does not remove object background. Furthermore, our rich 3D surface regression step allows for a geometric hypothesize-and-verify approach that can yield a good pose estimate even if parts of the surface prediction are incorrect. Such a geometry-based step is missing in their pipeline. We exceed the accuracy of BB8 by +6% on a challenging occlusion dataset. Another difference is that we use the *same* pipeline for RGB and RGB-D input images. For RGB-D input, our method outperforms the state-of-the-art [25, 12].

One key advantage of our modular design is that the components are interchangeable. For example, the first step of our pipeline addresses the instance segmentation task, which is an active field of research with an enormous rate of progress. Any improved instance segmentation method can be incorporated into our pipeline, without the need to adjust the remaining components. Such flexibility can be an important factor for practitioners.

As an additional contribution, we propose a new data augmentation scheme, specifically designed for the task of 6D object pose estimation. Since training data is usually scarce for instance pose estimation, data augmentation is a common aspect of learning-based pose estimation methods. Previous works have placed objects at random 2D locations over arbitrary background images [4, 27, 17] which yields implausible object-scene constellations.
We summarize our main contributions:

- We introduce a new pipeline for 6D object pose estimation which can use RGB or RGB-D input. The system is modular, and thus parts can be improved individually without redesigning the entire pipeline.
- We present a new data augmentation scheme for object pose estimation which generates physically plausible occlusion patterns. With this, we are the first to successfully train a neural network for object coordinate regression of known objects.
- We set a new state of the art for 6D pose estimation of occluded objects using RGB input. For RGB-D input we outperform the competing methods.

2. Related Work

Traditional 2D object detection methods aim at matching sparse features [24] or templates [15]. Templates work well for texture-less objects where sparse feature detectors fail to identify salient points. With the introduction of Kinect [31], Hinterstoisser *et al*. proposed the LINEMOD templates [13], which combine gradient and normal cues for robust object detection. Annotating the template database with viewpoint information facilitates accurate 6D pose estimation [11]. Template methods for pose estimation have been improved, predominately in terms of scalability and speed, *e.g*. by using template hierarchies [21], cascades [28] and hashing [19, 14]. An RGB version of LINEMOD [10] is less suited for pose estimation [4]. In general, template-based methods suffer from sensitivity to occlusion [2].

With a depth channel available, good results have been achieved by geometric voting schemes [32, 7, 12]. In particular, Drost *et al*. [7] cast votes by matching point pair features which combine normal and distance information. Recently, the method was improved in [12] by a suitable sampling scheme, resulting in state-of-the-art results for occluded objects given RGB-D input. Our pipeline achieves a comparable, or even slightly higher accuracy, and can also be applied to RGB input.

With the success of neural networks, learning-based methods are increasingly popular for 6D object pose estimation. Direct regression of object pose by a neural network, *e.g*. proposed by Kendall *et al*. for camera localization [20], exhibits low accuracy [3]. Instead, Kehl *et al*. [17] propose to solve pose estimation by classifying discrete object views with subsequent refinement. Their method can learn from synthetic training data, but the authors report only moderate detection performance for occluded objects, and do not evaluate pose estimation performance for the challenging *OccludedLINEMOD* dataset. The focus of our work is to perform well for heavily occluded objects. To achieve this we do *not* predict the 6D pose directly, but instead predict 3D surface positions (*i.e*. object coordinates) which are then used to robustly obtain the 6D object pose despite strong occlusions.

Some authors combine voting schemes with machine learning. Tejani *et al*. [30] propose to cast 6D pose votes for each image patch of an RGB-D image using a Hough-forest. Doumanoglou *et al*. [6] improve the method by using features of a deep auto-encoder at forest split nodes. Similarly in [18], auto-encoder features of image patches are matched to a codebook annotated with pose information. Thus far, these voting schemes have only been applied to pose estimation from RGB-D input.

As an alternative to voting, low-dimensional image-to-object correspondences can be predicted. The object coordinate framework, proposed by Brachmann *et al*. [2], for object pose estimation from RGB-D images utilizes a random forest to match image patches to 3D points in the local coordinate frame of the object. The pose can be recovered by a robust, RANSAC-based optimization of predicted correspondences. Because few correct correspondences suffice for a pose estimate, the method is inherently robust to object occlusion. Krull *et al*. [22] improved the pose optimization stage by learning to compare rendered and observed RGB-D images. Recently, Michel *et al*. [25] substituted the RANSAC component entirely with a conditional random field which is more effective in finding geometrically consistent correspondences but depends on a depth channel. In [4], an RGB version of the object coordinate framework is proposed, but the authors perform only 2D object detection for the occluded scenario. We utilize object coordinate regression as one component of our modular system, and report improved pose accuracy for occluded objects both for RGB and RGB-D input. In this context, the *main innovation* is that we are the first to successfully train a neural network for object coordinate regression of known objects. Previous works have employed random forests for this task. To make this work, we propose a new data augmentation scheme, which generates physically plausible occlusion patterns. Note that very recently Behl *et al*. [1] have trained a network for object coordinate regression of vehicles (*i.e*. object class). However, our network, training procedure, and data augmentation scheme differs from [1].

The BB8 [27], which we already mention in the introduction, also relies on correspondence prediction but only for a sparse set of fixed object points. We show that dense object coordinate regression provides a richer output, which is essential for robust geometric pose optimization, and hence yields improved results.

To summarize, only few previous works have addressed the challenging task of pose estimation of occluded objects from RGB input. We achieve superior accuracy for this task while, using the same system, slightly improving state-of-the-art accuracy for RGB-D input with the *same* appraoch.

3. Method

This section describes our modular three stage instance-aware approach for 6D object pose estimation. An overall workflow of our method is illustrated in Fig. 1. First, we obtain all the known object instances in an image using an instance segmentation network (Sec. 3.1). Next, for each of the discovered objects in the scene we estimate 3D object coordinates using an encoder-decoder network with skip connections (Sec. 3.2). Finally, we use the correspondences between predicted object coordinates and the input scene to sample hypotheses and further refine them using geometric refinement procedure, depending on RGB or RGB-D inputs (Sec. 3.3). Finally in Sec. 3.4, we describe our occlusion aware object centric data augmentation procedure which we use to generate augmented data for training the object coordinates CNN.

3.1. Stage1: Instance Segmentation

The main goal of instance segmentation is to classify and segment all the known objects within an image. In this work, we utilize a Multi-task Network Cascade (MNC) proposed by Dai *et al*. [5] for instance segmentation. The MNC consists of a VGG-based encoder for feature extraction followed by three subnetworks for object detection, mask estimation and classification. Given an input RGB image I, the output of the network will be instance masks $M = \{M_1, ..., M_N\}$ and the classes of those masks $k_i \in \{1, ..., K\}$, where K is the number of known objects.

3.2. Stage2: Object Coordinate Estimation

Object coordinates are 3D labeling of the surface points on an object, indicating their relative location in its local coordinate system. In other words, they provide an intermediate fine-grained geometric representation of the object in its local coordinate system. We use a CNN with an encoder-decoder style architecture with skip connections to estimate the pixelwise object coordinates for each instance found by MNC. The encoder consists of 5 convolutional layers with

a stride of 2 in each layer, followed by a set of 3 fully connected layers. The decoder has 5 deconvolutional layers followed by the 3 layer output. Skip connections exist between symmetrically opposite conv-deconv layers. The input to the CNN is an RGB image in case of RGB-only setup and an additional depth image in the RGB-D setup. We train separate CNNs for RGB and RGB-D cases. Our CNN has a 3 channel output containing the pixelwise X, Y and Z values for object coordinates.

Given an input RGB/RGB-D image I/I,D and a predicted mask from the MNC M_i, we first crop the image using the mask I_{M_i} (in case of RGB-D D_{M_i}) and then resize and pad it to a fixed-size I'_{M_i}. This is passed into the object coordinate network. The output of the network will consist of the predicted object coordinates for the input crop C'_{M_i}.

3.3. Stage3: Pose Estimation

In this section, we describe the final pose estimation step of our approach for RGB and RGB-D setups. This step extensively uses the instance masks from stage 1 and the object coordinates from stage 2 in a purely geometric framework and also demonstrates the importance of these modules.

3.3.1 RGB-D setup

In the RGB-D setup, the RGB image and the 3D point cloud of the scene is available. For a detected object O in the scene, we obtain the mask M_O which helps us in determining the corresponding point cloud P_O of the object. Also, stage 2 yields the predicted object coordinates C_O. Using P_O and C_O we randomly sample 3 pixels j_1, j_2, j_3 from mask M_O from which we establish 3D-3D correspondences $(P_O^{j_1}, C_O^{j_1})$, $(P_O^{j_3}, C_O^{j_3})$, $(P_O^{j_3}, C_O^{j_3})$. We use the Kabsch algorithm to compute the pose hypothesis H_c from these correspondences. The object coordinates $C_O^{j_1}$, $C_O^{j_2}$, $C_O^{j_3}$ are then transformed to the camera coordinate system using H_c. Let these transformed points be $T_O^{j_i}$. We compute the Euclidean distance, $d(P_O^{j_i}, T_O^{j_i})$. If the distances of all 3 points are less than 10% of object diameter, we add H_c to our hypothesis pool. Through this process, we collect 210 hypotheses. For each H_c, we render the point cloud P_O^* of the object into the camera coordinate system using its CAD model. Further, we score each of these hypothesis using

$$S_{H_c} = \frac{\sum_{i=1}^{m} \left[||P_O^i - P_O^{*i}|| < d/10 \right]}{m}, \qquad (1)$$

where $[\cdot]$ is the Iverson bracket which returns 1 if the enclosed condition is true, i is the index to the points inside the predicted mask of the object, and m is the total number of points in the predicted mask. $||P_O^i - P_O^{*i}||$ returns the Euclidean distance between a 3D point P_O^{*i} from the rendered point cloud and the corresponding observed point P_O^i. We consider i as an inlier if the Euclidean distance is less than 10% of the object diameter d. Effectively, S_{H_c} computes the average number of inlier 3D points for a given hypothesis H_c. We assign such a score to each of the 210 hypotheses and select the top 20. Finally, for each hypothesis H_{c_i}, we perform an Iterative Closest Point (ICP) based refinement with P_O as the target, the CAD model vertices as the source and H_{c_i} as initialization. The pose with the lowest fitting error from ICP is chosen as the final estimated pose $H_{c_{icp}}$ for the object.

Rendering based refinement. As a final refinement step, we render the CAD model of the object using $H_{c_{icp}}$ to obtain a point cloud of the visible surface, unlike the earlier ICP step where CAD model vertices are used. We use the point cloud inside the mask of the object (M_O) as a source to fit to the observed point cloud P_O via ICP (Table. 2).

3.3.2 RGB setup

In the RGB setup, we follow Brachmann *et al*. [4] and estimate the pose of the objects through hypotheses sampling [2] and the pre-emptive RANSAC [29]. At this stage, the predicted mask M_O and the predicted object coordinates C_O inside the mask are available. For each pixel i at the 2D location p_i inside the predicted mask $i \in M_O$, the object coordinate network estimates a 3D point C_O^i in the local object coordinate system. Thus, the 2D-3D correspondences between 2D points on the images and 3D object coordinate points can be easily sampled from the area inside the mask. Our goal is to search for a hypothesis H_c by sampling 4 2D-3D correspondences which maximize the following inlier count:

$$H_c^* = \arg\max_{H_c} \sum_{i \in M_O} \left[\|p_i - A H_c C_o^i\|_2 < \tau_{in} \right] \qquad (2)$$

where A is camera projection matrix, τ_{in} is re-projection inlier threshold, and $[\cdot]$ is 1 if statement inside the bracket is true, otherwise 0. Since we have a separate mask for each detected object, the search space for sampling is limited to the points inside the mask. The goal of Eq. 2 is to find a hypothesis H_c^* which can maximize the number of inliers when the predicted object coordinates are reprojected to the image using H_c^*. We use pre-emptive RANSAC to maximize this objective function.

We start by drawing 4 correspondences from the predicted mask M_O. Then we get the hypothesis by solving *PnP* [8, 23]. If the re-projection error of these points is below a certain threshold we keep them. Afterwards, we evaluate all the accepted hypotheses by randomly sampling N points inside the mask and computing the reprojection inliers for these points. Based on the scores (number of inliers), we sort the hypotheses and drop the lower half. Further, we refine each remaining hypothesis by solving *PnP*

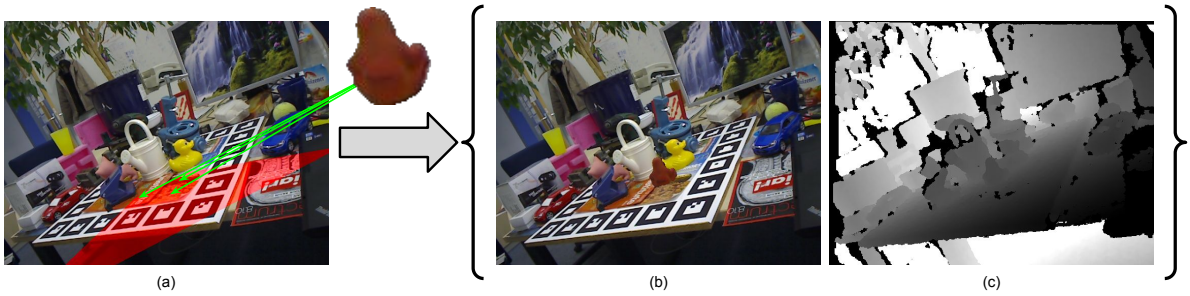

Figure 2. **Object centric data augmentation pipeline.** (a) demonstrates the area on the ground plane which potentially gives a chance to the cropped object (ape) for generating occlusion on the central target object (can). (b) demonstrates the resulting augmented RGB image. (c) shows the resulting augmented depth image.

on a random subset of their inliers. We repeat this procedure until only one hypothesis remains as the final estimated pose H_c^*.

3.4. Data augmentation

Data augmentation is crucial for training a deep model, especially when we need to train for a dense estimation task. Additionally, data augmentation can help to reduce the dataset bias and introduce novel examples for the deep model to train on. One of the straightforward ways to perform the data augmentation is to cut the target object from the existing limited dataset and paste it on a random background. Although this approach is often good enough for high level tasks such as object detection, classification and instance segmentation, it does not work well for our low level task of object coordinates prediction. We attribute this failure to the generation of random, physically implausible occlusions on the target objects. Such random augmentation of training data induces unnecessary and implausible scenarios in data samples, hence making it difficult for the network to converge and also introducing bias towards unnecessary object occlusion configurations. In the following section, we present a principled *object centric data augmentation* approach, which generates reasonable occlusion on the target objects. Furthermore, we introduce a new occlusion analysis method, *i.e. on-object occlusion*, for better analysis and comparison between the generated dataset and the base training set.

3.4.1 Object Centric Data Augmentation

The main goal of this approach is to provide a dataset with physically reasonable and realistic occlusion on target objects. We use the central objects from *unoccluded LINEMOD* sequences (*Ape, Can, Cat, Duck, Eggbox, Glue* and *Holepuncher*) as the target object in this data augmentation pipeline.

For each image in those 7 sequences of *LINEMOD*, first we compute the ground plane on which the central object stands, as well as the distance between its base point and the camera. Then, as shown in Fig.2(a)(red), a surface of interest is defined on the ground plane in front of the central object, representing a cone with an opening angle of $90°$. Next, we find objects from the other sequences whose ground plain normal is close to that of the target object, and which lie inside the defined surface of interest, based on their distance from camera. Finally, by overlaying one or more of these chosen objects in front of the target object, we can generate multiple augmented RGB and Depth images (*c.f.* Fig.2(b,c)).

Using this approach, the resulting occlusion looks physically correct for both the RGB and the depth image.

3.4.2 On-Object Occlusion Analysis

The occlusion on target objects is a key factor in determining the complexity of the task. The occlusion is introduced either by other annotated objects or unknown clutter objects. Also, truncation at the borders of the image can be considered as occlusion. Computing the percentage of the occlusion on each object in the dataset is a common way to analyze the difficulty of the dataset. However, it is beneficial to know the object-part occlusion distribution throughout the dataset. This knowledge helps us in determining why a certain model works when trained on a base training set versus our augmented dataset.

Therefore in this work, we introduce *on-object occlusion analysis* which is a visualization of the distribution of object part occlusion in a dataset. To obtain this visualization, the 3D bounding box surrounding each object is discretized in a total number of $20 \times 20 \times 20$ voxels. Each annotated object in the dataset has a mask in the image plane, which covers only the visible parts of the object, and a 6D pose. Using this 6D pose and the 3D CAD model, we can render the full mask of the object; and for each pixel inside the rendered mask, we can compute the ground truth object coordinate of the pixel. Then, the respective voxel for each pixel can be found by using its corresponding object coordinate. These voxels can be used as histogram bins and visualized as colors on the surface of the 3D CAD model.

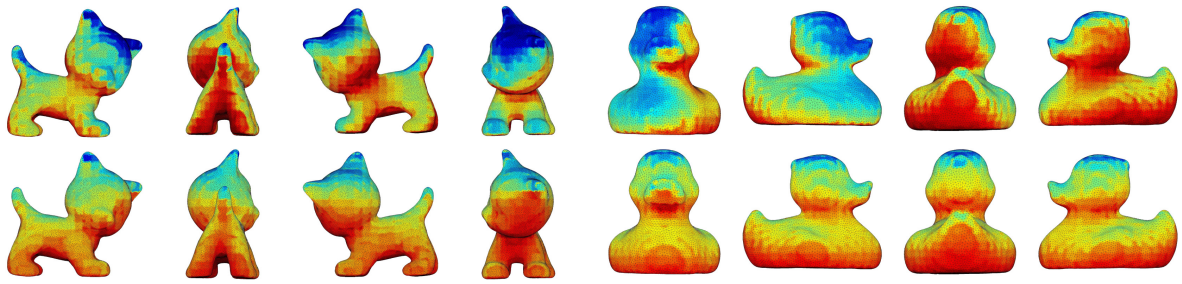

Figure 3. **Illustration of on-object occlusion distribution.** The top row illustrates the on-object occlusion distribution of the base training set before augmentation and bottom row illustrates the same for augmented data using our object centric data augmentation. For a given part on the model, red indicates that the part is often occluded as compared to blue on a part indicating rare occlusion in a given dataset.

Fig. 3 illustrates the *on-object occlusion* visualization before and after our object centric data augmentation process for Cat and Duck. Firstly, by looking at the visualization (top row), it can be noticed that the un-augmented data contains biased occlusion samples (irregular distribution of blue and red patches) which could induce overfitting on certain object parts, leading to reduced performance of the object coordinate network. In the second row, it can be seen that the augmented data has a better distribution of *on-object occlusion* thus reducing the biased occlusion patterns. The advantage of using this visualization is two fold. Firstly, it reveals the bias in the base training set. Additionally, it demonstrates the efficacy of our object centric data augmentation procedure to create unbiased training data samples.

4. Experiments

In this section, we present various experiments quantifying the performance of our approach. In Sec. 4.1, we introduce the dataset for evaluating our system. In Sec. 4.2, we compare the performance of our approach to existing RGB and RGB-D based pose estimation approaches. In Sec. 4.2.3, we analyse the contribution of various modules of our approach to the final pose estimation performance. Finally, in Sec. 4.3 and 4.4, we discuss the performance of our instance segmentation and object coordinate estimation networks.

4.1. Datasets and Implementation

We evaluate our approach on the publicly available *occludedLINEMOD* dataset published by Brachmann *et al*. [2]. It was created from the LINEMOD dataset [11] by annotating ground truth 6D poses for various objects in a sequence of 1214 RGB-D images. The objects are located on a table and embedded in dense clutter. Ground truth poses are provided for seven of these objects which, depending on the camera view, heavily occlude each other, making this dataset very challenging. We test both our RGB and RGB-D-based methods on this dataset.

To train our system, we use a separate sequence from the LINEMOD dataset which was annotated by Michel *et al*. [25] for hyper-parameter tuning. For ease of reference we call this LINEMOD-M dataset. Similar to the test sequence mentioned earlier, this training sequence comes with ground truth annotations of six objects with mutual occlusion. One object of the test sequence, namely the Driller, is not present in this training sequence, so we do not report results for it. The training sequence is extremely limited in the amount of data it provides. Some objects are only seen from few viewpoints and with little occlusion, or occlusion affects only certain object parts.

Training Instance Segmentation. To train the instance segmentation network (MNC) with a wide range of object viewpoints, and diverse occlusion examples, we create an augmented dataset by using RGB images from NYUD dataset [26] as backgrounds and randomly overlaying them with objects picked from *Unoccluded LINEMOD dataset* [11] images. Each image created this way contains one instance of all the 6 objects present in the dataset, and additional objects like lamp, camera, benchvise, *etc*. We combine these generated images with LINEMOD-M to obtain 9000 images with groundtruth instance masks. We split this dataset into training and validation sets for training the MNC. We use the standard Caffe [16] implementation and the hyperparameters provided by Dai *et al*. [5] for MNC. We initialize it with the model provided by [5] which was trained on PASCAL VOC dataset and finetune it with mentioned augmented data.

Training Object Coordinate Regression. For training the object coordinate estimation network, we found it important to utilize physically plausible data augmentation for best results. Therefore, we use the LINEMOD-M dataset along with the data obtained using our object centric data augmentation pipeline described in Sec. 3.4. Note that this data is different from the augmented data used to train MNC. We implemented our object coordinate estimation network in Caffe framework [16] and trained it by minimizing a robust and smooth Huber loss function [9] using the Adam solver. We train a separate network for each object class.

	5 px		10 px	
	BB8[27]	ours	BB8[27]	ours
Ape	28.5%	**31.6%**	**81.0%**	65.3%
Can	1.2%	**29.7%**	27.8%	**73.2%**
Cat	9.6%	**8.8%**	**61.8%**	35.9%
Duck	6.8%	**10.0%**	41.3%	**41.6%**
Glue	4.7%	**22.3%**	37.7%	**52.6%**
Hole Puncher	2.4%	**20.0%**	45.4%	**60.5%**
	8.9%	**20.5%**	49.2%	**55.2%**

Table 1. Comparison of our pose estimation accuracies in RGB only setup with the existing state-of-the-art method.

While training, we present the network with a 256x256 RGB/RGB-D scaled instance of an object and the pixel wise groundtruth object coordinates.

Note that the test sequence and our training data are strictly separated, *i.e*. we did not use parts of the test sequence for data augmentation.

4.2. Pose Estimation Accuracy

In this section, we compare the performance of our method to existing RGB as well as RGB-D-based approaches.

4.2.1 RGB Setup

We estimate object poses from RGB images without utilizing the depth channel. We measure accuracy as the percentage of correct object pose estimates. We accept a pose to be correct based on the metric proposed by Brachmann *et al*. [4]. This metric measures the average reprojection error of 3D model vertices transformed by the ground truth pose and the estimated pose. We report results for two different thresholds on the average reprojection error. For an acceptance threshold of 5px, the estimated pose provides a visually tight fit. A threshold of 10px includes pose estimates that are less aligned but still visually pleasing, see Fig. 4 for qualitative results using both thresholds.

In Table 1, we compare the performance of our approach to the current state-of-the-art method proposed by Rad *et al*. [27]. For the 5px acceptance threshold, our method outperforms [27] by a significant margin for all objects. We achieve an accuracy of 20.5% compared to 8.9% by [27], showing an improvement of 11.5%. Additionally, we also show improved performance for most of the objects using the 10px metric surpassing the state-of-the-art by 6% on average. To summarize, we estimate significantly more object poses correctly, often giving visually tight fits. Note that pose estimation of heavily occluded objects from RGB inputs is extremely challenging, and we are not aware of another method beside [27] which reports results for this scenario. Similar to [27], we do not report results for *EggBox* since we could not get any reasonable result for this object, which is extremely occluded in all test frames.

Figure 4. Qualitative results from RGB setup. The top row shows success and failure cases of our pipeline according to the 5px metric. The bottom row shows success and failure cases of our pipeline according to the 10px metric. It can be seen that although we penalize our method (red boxes) for the 5px metric, the same are accepted for the 10px metric. Such cases demonstrate that although the bounding boxes are wrong according to the 5px metric, our method produces visually accurate results in such cases for augmented reality applications.

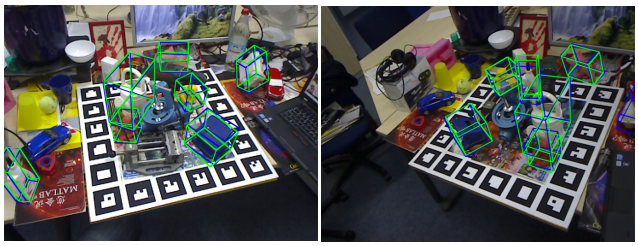

Figure 5. Qualitative results from RGB-D setup. Our approach reliably estimates pose for objects which are heavily occluded. *e.g*. Duck and Cat (left), Duck and Glue (right). The green box is the pose predicted by our approach and blue box is ground truth.

4.2.2 RGB-D Setup

In this setup, the input to our pipeline is an RGB-D image. Similar to the RGB setup, we measure accuracy as the percentage of correctly estimated object poses. Following Hinterstoisser *et al*. in [11], we accept a pose if the average 3D distance between 3D model vertices transformed using groundtruth pose and predicted pose lies below 10% of the object diameter. In Table 2, we compare the performance of our approach to Michel *et al*. [25] and Hinterstoisser *et al*. [12]. We show results of our pipeline with ICP refinement *Ours(ICP)*, and additionally using a rendering-based refinement stage *Ours(ICP+Refine)*. We outperform the state-of-the-art on average and show considerable improvements for *Glue* and *Eggbox*.

	Michel *et al*. [25]	Hinterstoisser *et al*. [12]	Ours (ICP)	Ours (ICP + Refine)
Ape	80.7%	**81.4%**	71.6%	73.3%
Can	88.5%	**94.7%**	85.1%	85.4%
Cat	**57.8%**	55.2%	53.4%	54.5%
Duck	74.4%	**79.7%**	73.5%	73.8%
Eggbox	47.6%	65.5%	70.7%	**73.4%**
Glue	73.8%	52.1%	73.1%	**74.5%**
Holepuncher	**96.3%**	95.5%	95.7%	95.3%
Average	74.2%	74.9%	74.8%	**75.7%**

Table 2. Comparison of our pose estimation accuracies in RGB-D setup with the existing state-of-the-art methods.

4.2.3 Ablation Study

We investigate the contribution of each module of our method towards the final pose estimation accuracy for the RGB-D setup. As discussed before, our method consists of three modules, namely instance mask estimation, object coordinate regression and pose estimation. We compare to the method of Brachmann *et al*. [2] which has similar steps, namely soft segmentation (class-based), object coordinate regression and a final RANSAC-based pose estimation. The first two steps in [2] are implemented using a random forest, compared to CNNs in our system. Table 3 shows the accuracies for various re-combinations of these modules. The first row is the standard baseline approach of [2] which achieves an average accuracy of 52.9%. In the second row, we replace the soft segmentation estimated by [2] with our instance segmentation masks. Our masks effectively constrain the 2D search space and it leads to better sampling of correspondences between depth points and object coordinate predictions, improving the accuracy by 3.5%. Next, we additionally replace the object coordinate predictions of the random forest of [2] with our CNN-based predictions. Although we still perform the same pose optimization, this achieves an additional 4.6% performance boost (third row), showing that our encoder-decoder network architecture predicts object coordinates more precisely. Finally, in the last row, we show a significant accuracy improvement using our full pipeline.

4.3. Instance Segmentation

Since we cannot hope to estimate a correct pose for an object that we do not detect, the performance of this stage is crucial for our overall accuracy. See Table 4 for detection rates of the MNC component for each object in the test set. Despite the limited amount of training data, and heavy object occlusion, we report very high detection rates that exceed our rates of correct pose estimation. We conclude that future efforts should focus on object coordinate regression and pose optimization rather than object detection which works well, already.

Mask	Obj. Coord.	Pose Estimation	Score
RF[2]	RF[2]	Brachmann *et al*. [2]	52.9%
MNC[5]	RF[2]	Brachmann *et al*. [2]	56.4%
MNC[5]	ours	Brachmann *et al*. [2]	61.0%
MNC[5]	ours	ours	75.7%

Table 3. Pose estimation accuracies on RGB-D dataset using various combinations of mask estimation, object coordinates estimation and pose estimation approaches.

Ape	Can	Cat	Duck	Eggb	Glue	HoleP.	Avg.
91.9	98.5	87.8	90.4	78.7	80.8	98.8	89.6

Table 4. MNC detection rate for various objects in test set.

4.4. Object coordinate estimation

We trained the networks with and without our data augmentation procedure (sec. 3.4). We measure the average inlier rate, *i.e*. object coordinate estimates that are predicted within 2cm of ground truth object coordinates to measure the performance of the network. When the network is trained only using the LINEMOD-M dataset, the average inlier rate is 44% as compared to 52% when we use the data created using our object centric data augmentation procedure. A clear 8% increase in the inlier rate shows the importance of our proposed data augmentation.

Conclusion

We presented a modular system for object pose estimation from single RGB or RGB-D images, using a new combination of machine learning techniques and robust geometric optimization. Using RGB input, our system sets a new state-of-the-art for pose estimation accuracy for heavily occluded objects. The same system can also exploit a depth channel for improved accuracy which is on-par or slightly superior to competing RGB-D-based methods. Because our individual processing stages are strictly decoupled, they can be upgraded individually without the need to re-adjust the architecture as a whole. We also presented a new data augmentation procedure that improves performance of learning based 6D pose estimation by creating physically plausible occlusion patterns. While we made progress for RGB-based pose estimation for heavily occluded objects, we exceed the average performance of existing RGB-D methods.

References

[1] A. Behl, O. H. Jafari, S. K. Mustikovela, H. A. Alhaija, C. Rother, and A. Geiger. Bounding boxes, segmentations and object coordinates: How important is recognition for 3d scene flow estimation in autonomous driving scenarios? In *ICCV*, 2017. 3

[2] E. Brachmann, A. Krull, F. Michel, S. Gumhold, J. Shotton, and C. Rother. Learning 6D object pose estimation using 3D object coordinates. In *ECCV*, 2014. 1, 2, 3, 4, 6, 8

[3] E. Brachmann, A. Krull, S. Nowozin, J. Shotton, F. Michel, S. Gumhold, and C. Rother. DSAC-Differentiable RANSAC for camera localization. In *CVPR*, 2017. 3

[4] E. Brachmann, F. Michel, A. Krull, M. Y. Yang, S. Gumhold, and C. Rother. Uncertainty-driven 6D pose estimation of objects and scenes from a single RGB image. In *CVPR*, 2016. 1, 2, 3, 4, 7

[5] J. Dai, K. He, and J. Sun. Instance-aware semantic segmentation via multi-task network cascades. In *CVPR*, 2016. 3, 6, 8

[6] A. Doumanoglou, R. Kouskouridas, S. Malassiotis, and T. Kim. 6D object detection and next-best-view prediction in the crowd. In *CVPR*, 2016. 3

[7] B. Drost, M. Ulrich, N. Navab, and S. Ilic. Model globally, match locally: Efficient and robust 3D object recognition. In *CVPR*, 2010. 2

[8] X.-S. Gao, X.-R. Hou, J. Tang, and H.-F. Cheng. Complete solution classification for the perspective-three-point problem. *IEEE Trans. on PAMI*, 2003. 4

[9] R. Girshick. Fast R-CNN. In *Proceedings of the International Conference on Computer Vision (ICCV)*, 2015. 6

[10] S. Hinterstoisser, C. Cagniart, S. Ilic, P. Sturm, N. Navab, P. Fua, and V. Lepetit. Gradient response maps for real-time detection of texture-less objects. *IEEE Trans. on PAMI*, 2012. 2

[11] S. Hinterstoisser, V. Lepetit, S. Ilic, S. Holzer, G. Bradski, K. Konolige, and N. Navab. Model based training, detection and pose estimation of texture-less 3D objects in heavily cluttered scenes. In *ACCV*, 2012. 2, 6, 7

[12] S. Hinterstoisser, V. Lepetit, N. Rajkumar, and K. Konolige. Going further with point pair features. In *ECCV*, 2016. 1, 2, 7, 8

[13] S. Hinterstoisser, S. Holzer, C. Cagniart, S. Ilic, K. Konolige, N. Navab, and V. Lepetit. Multimodal templates for real-time detection of texture-less objects in heavily cluttered scenes. In *ICCV*, 2011. 1, 2

[14] T. Hodaň, X. Zabulis, M. Lourakis, Š. Obdržálek, and J. Matas. Detection and fine 3D pose estimation of texture-less objects in RGB-D images. In *IROS*, 2015. 2

[15] D. Huttenlocher, G. Klanderman, and W. Rucklidge. Comparing images using the Hausdorff distance. *IEEE Trans. on PAMI*, 1993. 2

[16] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell. Caffe: Convolutional architecture for fast feature embedding. *arXiv preprint arXiv:1408.5093*, 2014. 6

[17] W. Kehl, F. Manhardt, F. Tombari, S. Ilic, and N. Navab. Ssd-6d: Making rgb-based 3d detection and 6d pose estimation great again. In *ICCV*, 2017. 2, 3

[18] W. Kehl, F. Milletari, F. Tombari, S. Ilic, and N. Navab. Deep learning of local RGB-D patches for 3D object detection and 6D pose estimation. In *ECCV*, 2016. 3

[19] W. Kehl, F. Tombari, N. Navab, S. Ilic, and V. Lepetit. Hashmod: A hashing method for scalable 3D object detection. In *BMVC*, 2016. 2

[20] A. Kendall, M. Grimes, and R. Cipolla. PoseNet: A convolutional network for real-time 6-DoF camera relocalization. In *ICCV*, 2015. 3

[21] Y. Konishi, Y. Hanzawa, M. Kawade, and M. Hashimoto. Fast 6D pose estimation from a monocular image using hierarchical pose trees. In *ECCV*, 2016. 2

[22] A. Krull, E. Brachmann, F. Michel, M. Y. Yang, S. Gumhold, and C. Rother. Learning analysis-by-synthesis for 6D pose estimation in RGB-D images. In *ICCV*, 2015. 1, 3

[23] V. Lepetit, F. Moreno-Noguer, and P. Fua. EPNP: An accurate O(n) solution to the PNP problem. *IJCV*, 2009. 4

[24] D. G. Lowe. Local feature view clustering for 3D object recognition. In *CVPR*, 2001. 1, 2

[25] F. Michel, A. Kirillov, E. Brachmann, A. Krull, S. Gumhold, B. Savchynskyy, and C. Rother. Global hypothesis generation for 6D object pose estimation. In *CVPR*, 2017. 1, 2, 3, 6, 7, 8

[26] P. K. Nathan Silberman, Derek Hoiem and R. Fergus. Indoor segmentation and support inference from rgbd images. In *ECCV*, 2012. 6

[27] M. Rad and V. Lepetit. BB8: A scalable, accurate, robust to partial occlusion method for predicting the 3d poses of challenging objects without using depth. In *ICCV*, 2017. 1, 2, 3, 7

[28] R. Rios-Cabrera and T. Tuytelaars. Discriminatively trained templates for 3D object detection: A real time scalable approach. In *ICCV*, 2013. 2

[29] J. Shotton, B. Glocker, C. Zach, S. Izadi, A. Criminisi, and A. W. Fitzgibbon. Scene coordinate regression forests for camera relocalization in RGB-D images. In *CVPR*, 2013. 4

[30] A. Tejani, D. Tang, R. Kouskouridas, and T.-K. Kim. Latent-class Hough forests for 3D object detection and pose estimation. In *ECCV*, 2014. 3

[31] M. C. R. WA. Kinect for Xbox 360. 1, 2

[32] C. Zach, A. Penate-Sanchez, and M.-T. Pham. A dynamic programming approach for fast and robust object pose recognition from range images. In *CVPR*, 2015. 2