iPose: Instance-Aware 6D Pose Estimation of Partly Occluded Objects

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

Visual Learning Lab - Heidelberg University (HCI/IWR)
`http://vislearn.de`

Abstract. We address the task of 6D pose estimation of known rigid objects from single input images in scenarios where the objects are partly occluded. Recent RGB-D-based methods are robust to moderate degrees of occlusion. For RGB inputs, no previous method works well for partly occluded objects. Our main contribution is to present the first deep learning-based system that estimates accurate poses for partly occluded objects from RGB-D and RGB input. We achieve this with a new instance-aware pipeline that decomposes 6D object pose estimation into a sequence of simpler steps, where each step removes specific aspects of the problem. The first step localizes all known objects in the image using an instance segmentation network, and hence eliminates surrounding clutter and occluders. The second step densely maps pixels to 3D object surface positions, so called object coordinates, using an encoder-decoder network, and hence eliminates object appearance. The third, and final, step predicts the 6D pose using geometric optimization. We demonstrate that we significantly outperform the state-of-the-art for pose estimation of partly occluded objects for both RGB and RGB-D input.

1 Introduction

Localization of object instances from single input images has been a long-standing goal in computer vision. The task evolved from simple 2D detection to full 6D pose estimation, *i.e.* estimating the 3D position and 3D orientation of the object relative to the observing camera. Early approaches relied on objects having sufficient texture to match feature points [1]. Later, with the advent of consumer depth cameras [2], research focused on texture-less objects [3] in increasingly cluttered environments. Today, heavy occlusion of objects is the main performance benchmark for one-shot pose estimation methods. Object occlusion occurs in all scenarios, apart from artificial settings, hence robustness to occlusion is crucial in applications like augmented reality or robotics.

Recent RGB-D-based methods [4,5] are robust to moderate degrees of object occlusion. However, depth cameras fail under certain conditions, *e.g.* with intense sunlight, and RGB cameras are prevalent on many types of devices. Hence, RGB-based methods still have high practical relevance. In this work, we present a
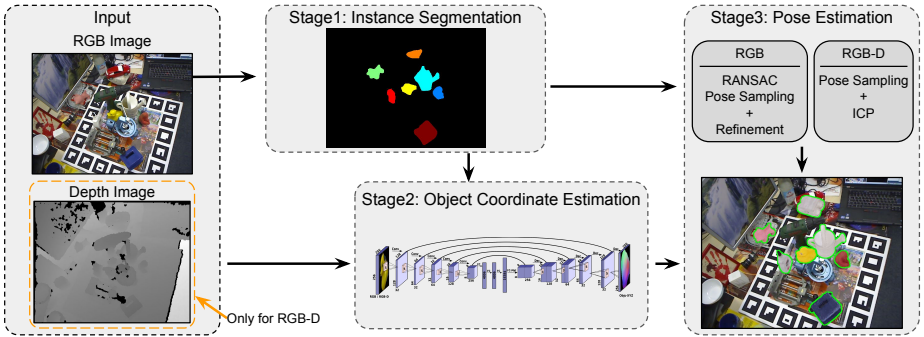
---

* Equal contribution

Fig. 1: **Illustration of our modular, 3-stage pipeline** for both RGB and RGB-D input images.

system for 6D pose estimation of rigid object instances from single input images. The system performs well for partly occluded objects. That means for both input modalities, RGB-D and RGB, it clearly outperforms the accuracy of previous methods.

During the last decade, computer vision has seen a large shift towards learning-based methods. In particular, *deep learning*, *i.e.* training multi-layered neural networks, has massively improved accuracy and robustness for many tasks, most notably object recognition [6], object detection [7,8,9] and semantic segmentation [10,11,12]. While 6D object pose estimation has also benefited from deep learning to some extent, with recent methods being able to estimate accurate poses in real time from single RGB images [13,14,15], the same does not hold when objects are partly occluded. In this case, aforementioned methods, despite being trained with partly occluded objects, either break down [14,15] or have to simplify the task by estimating poses from tight crops around the ground truth object position [13]. To the best of our knowledge, we are the first to show that deep learning can improve results considerably for objects that are moderately to heavily occluded, particularly for the difficult case of RGB input.

At the core, our method decomposes the 6D pose estimation problem into a sequence of three sub-tasks, or modules (see Fig. 1). We first detect the object in 2D, then we locally regress correspondences to the 3D object surface, and, finally, we estimate the 6D pose of the object. With each sub-task, we can remove specific aspects of the problem, such as object background and object appearance. In the first module, 2D detection is implemented by an instance segmentation network which estimates a tight mask for each object. Thus, we can separate the object from surrounding clutter and occluders, making the following steps invariant to the object environment, and allowing us to process each detected instance individually. In the second module, we present an encoder-decoder architecture for densely regressing so-called *object coordinates* [16], *i.e.* 3D points in the local coordinate frame of the object which define 2D-3D correspondences between the image and the object. The third module is a purely geometric pose optimization

which is not learned from data because all aspects of object appearance have been removed in the previous steps. Since we estimate 6D poses successively from 2D instance segmentation, we call our approach *iPose*, short for "instance-aware pose estimation".

Our decomposition strategy is conceptually simple, but we show that it is considerably superior to other deep learning-based methods that try to reason about different aspects of these steps jointly. In particular, several recent works propose to extend state-of-the-art object detection networks to output 6D object poses directly. Kehl *et al.* [14] extend the SSD object detector [9] to recognize discretized view-points of specific objects, *i.e.* re-formulating pose regression as a classification problem. Similarly, Tekin *et al.* [15] extend the YOLO object detector [8] by letting image grid cells predict object presence, and simultaneously the 6D pose. Both approaches are highly sensitive to object occlusion, as we will show in the experimental evaluation. Directly predicting the 6D pose from observed object appearance is challenging, due to limited training data and innumerable occlusion possibilities.

We see three reasons for the success of our approach. Firstly, we exploit the massive progress in object detection and instance segmentation achieved by methods like MNC [11] and Mask R-CNN [12]. This is similar in spirit to the work of [14,15], but instead of extending the instance segmentation to predict 6D poses directly, we use it as a decoupled component within our step-by-step strategy. Secondly, the rich structural output of our dense object coordinate regression step allows for a geometric hypothesize-and-verify approach that can yield a good pose estimate even if parts of the prediction are incorrect, *e.g.* due to occlusion. Such a robust geometry-based step is missing in previous deep learning-based approaches [13,14,15]. Thirdly, we propose a new data augmentation scheme specifically designed for the task of 6D object pose estimation. Data augmentation is a common aspect of learning-based pose estimation methods, since training data is usually scarce. Previous works have placed objects at random 2D locations over arbitrary background images [17,13,14], which yields constellations where objects occlude each other in physically impossible ways. In contrast, our data augmentation scheme infers a common ground plane from ground truth poses and places additional objects in a physically plausible fashion. Hence, our data augmentation results in more realistic occlusion patterns which we found crucial for obtaining good results.

We summarize our main **contributions**:

- We propose *iPose*, a new deep learning architecture for 6D object pose estimation which is remarkably robust w.r.t. object occlusion, using a new three-step task decomposition approach.
- We are the first to surpass the state-of-the-art for partly occluded objects with a deep learning-based approach for both RGB-D and RGB inputs.
- We present a new data augmentation scheme for object pose estimation which generates physically plausible occlusion patterns, crucial for obtaining good results.

2   Related Work

Below, we give an overview of previous methods for 6D object pose estimation. Note that there is a body of work regarding pose estimation of object categories, specifically in the context of autonomous driving on datasets like KITTI [18], see *e.g.* [19,20,21,22]. Because of intra-class variability, these approaches often estimate coarse viewpoints or constrained poses, *e.g.* 3D poses on a ground plane. In this work, we consider the different task of estimating full 6D poses of specific, rigid object instances.

Early pose estimation methods were based on matching sparse features [1] or templates [23]. Templates work well for texture-less objects where sparse feature detectors fail to identify salient points. Hinterstoisser *et al.* proposed the LINEMOD templates [3], which combine gradient and normal cues for robust object detection given RGB-D inputs. Annotating the template database with viewpoint information facilitates accurate 6D pose estimation [24,25,26,27,28]. An RGB version of LINEMOD [29] is less suited for pose estimation [17]. In general, template-based methods suffer from sensitivity to occlusion [16].

With a depth channel available, good results have been achieved by voting-based schemes [30,31,32,33,34,5]. In particular, Drost *et al.* [34] cast votes by matching point-pair features which combine normal and distance information. Recently, the method was considerably improved in [5] by a suitable sampling scheme, resulting in a purely geometric method that achieves state-of-the-art results for partly occluded objects given RGB-D inputs. Our deep learning-based pipeline achieves higher accuracy, and can also be applied to RGB images.

Recently, deep learning-based methods have become increasingly popular for object pose estimation from RGB images. Rad and Lepetit [13] presented the BB8 pipeline which resembles our decomposition philosophy to some extent. However, their processing steps are more tightly coupled. For example, their initial detection stage does not segment the object, and can thus not remove object background. Also, they regress the 6D pose by estimating the 2D location of a sparse set of control points. We show that dense 3D object coordinate regression provides a richer output which is essential for robust geometric pose optimization. Rad and Lepetit [13] evaluate BB8 on occluded objects but restrict pose prediction to image crops around the ground truth object position[1]. Our approach yields superior results for partly occluded objects *without* using prior knowledge about object position.

Direct regression of a 6D pose vector by a neural network, *e.g.* proposed by Kendall *et al.* for camera localization [35], exhibits low accuracy [36]. The works discussed in the introduction, *i.e.* Kehl *et al.* [14] and Tekin *et al.* [15], also regress object pose directly but make use of alternative pose parametrizations, namely discrete view point classification [14], or sparse control point regression

---

[1] Their experimental setup which relies on ground truth crops is not explicitly described in [13], but we verified this information from a private email exchange with the authors of [13].

[15] similar to BB8 [13]. We do *not* predict the 6D pose directly, but follow a step-by-step strategy to robustly obtain the 6D pose despite strong occlusions.

Object coordinates have been used previously for object pose estimation from RGB-D [16,37,4] or RGB inputs [17]. In these works, random forest matches image patches to 3D points in the local coordinate frame of the object, and the pose is recovered by robust, geometric optimization. Because few correct correspondences suffice for a pose estimate, these methods are inherently robust to object occlusion. In contrast to our work, they combine object coordinate 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 mask is known for object coordinate regression. In this context, we are also the first to successfully train a neural network for object coordinate regression of known objects. Overall, we report superior pose accuracy for partly occluded objects using RGB and RGB-D inputs. Note that recently Behl *et al.* [38] have trained a network for object coordinate regression of vehicles (*i.e.* object class). However, our network, training procedure, and data augmentation scheme differ from [38].

To cope well with limited training data, we propose a new data augmentation scheme which generates physically plausible occlusion patterns. While plausible data augmentation is becoming common in object class detection works, see *e.g.* [39,40,41], our scheme is tailored specifically towards object instance pose estimation where previous works resorted to pasting 2D object crops on arbitrary RGB backgrounds [17,13,14]. We found physically plausible data augmentation to be crucial for obtaining good results for partly occluded objects.

To summarize, only few previous works have addressed the challenging task of pose estimation of partly occluded objects from single RGB or RGB-D inputs. We present the first viable deep learning approach for this scenario, improving state-of-the-art accuracy considerably for both input types.

3   Method

In this section, we describe our three-stage, instance-aware approach for 6D object pose estimation. The overall workflow of our method is illustrated in Fig. 1. Firstly, we obtain all object instances in a given image using an instance segmentation network (Sec. 3.1). Secondly, we estimate dense 3D object coordinates for each instance using an encoder-decoder network (Sec. 3.2). Thirdly, we use the pixel-wise correspondences between predicted object coordinates and the input image to sample 6D pose hypotheses, and further refine them using an iterative geometric optimization (Sec. 3.3). In Sec. 3.4, we describe our object-centric data augmentation procedure which we use to generate additional training data with realistic occlusions for the encoder-decoder network of step 2.

We denote the RGB input to our pipeline as I and RGB-D input as I-D. $\mathcal{K} = \{1, ..., K\}$ is a set of all known object classes, a subset of which could be present in the image. The goal of our method is to take an image I/I-D

containing n objects $\mathcal{O} = \{O_1, ..., O_n\}$, each of which has a class from $\mathcal{K}$, and to estimate their 6D poses. Below, we describe each step of our pipeline in detail.

3.1 Stage 1: Instance Segmentation

The first step of our approach, instance segmentation, recognizes the identity of each object, and produces a fine grained mask. Thus we can separate the RGB(-D) information pertaining only to a specific object from surrounding clutter and occluders. To achieve this, we utilize instance segmentation frameworks such as [11,12]. Given an input I, the output of this network is a set of n instance masks $\mathcal{M} = \{M_1, ..., M_n\}$ and an object class $k \in \mathcal{K}$ for each mask.

3.2 Stage 2: Object Coordinate Regression

An object coordinate denotes the 3D position of an object surface point in the object's local coordinate frame. Thus given a pixel location p and its predicted object coordinate C, a (p, C) pair defines a correspondence between an image I and object O. Multiple such correspondences, at least three for RGB-D data and four for RGB data, are required to recover the 6D object pose (see Sec. 3.3). In order to regress pixelwise object coordinates C for each detected object, we use a CNN with an encoder-decoder style architecture with skip connections. 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 corresponding to 3-dimensional object coordinates. Skip connections exist between symmetrically opposite conv-deconv layers. As input for this network, we crop a detected object using its estimated mask M, resize and pad the crop to a fixed size, and pass it through the object coordinate network. The output of this network has 3 channels containing the pixelwise X, Y and Z values of object coordinates C for mask M. We train separate networks for RGB and RGB-D inputs.

3.3 Stage 3: Pose Estimation

In this section, we describe the geometric pose optimization step of our approach for RGB-D and RGB inputs, respectively. This step is not learned from data, but recovers the 6D object pose from the instance mask M of stage 1 and the object coordinates C of stage 2.

RGB-D Setup. Our pose estimation process is inspired by the original object coordinate framework of [16]. Compared to [16], we use a simplified scoring function to rank pose hypotheses, and an Iterative Closest Point (ICP) refinement.

In detail, we use the depth channel and the mask M_O to calculate a 3D point cloud P_O associated with object O w.r.t. the coordinate frame of the camera. Also, stage 2 yields the pixelwise predicted object coordinates C_O. We seek the

6D pose H_O^* which relates object coordinates C_O with the point cloud P_O. For ease of notation, we drop the subscript O, assuming that we are describing the process for that particular object instance. We randomly sample three pixels j_1, j_2, j_3 from mask M, from which we establish three 3D-3D correspondences (P^{j_1}, C^{j_1}), (P^{j_2}, C^{j_2}), (P^{j_3}, C^{j_3}). We use the Kabsch algorithm [42] to compute the pose hypothesis H_i from these correspondences. Using H_i, we transform C^{j_1}, C^{j_2}, C^{j_3} from the object coordinate frame to the camera coordinate frame. Let these transformed points be T^j. We compute the Euclidean distance, $\|P^j, T^j\|$, and if the distances of all three points are less than 10% of the object diameter, we add H_i to our hypothesis pool. We repeat this process until we have collected 210 hypotheses. For each hypothesis H, we obtain a point cloud $P^*(H)$ in the camera coordinate system via rendering the object CAD model. This lets us score each hypothesis using

$$S_{\text{RGB-D}}(H) = \frac{\sum_{j \in M} \left[||P^j - P^{*j}(H)|| < d/10 \right]}{|M|}, \qquad (1)$$

where $[\cdot]$ returns 1 if the enclosed condition is true, and the sum is over pixels inside the mask M and normalized. The score $S_{\text{RGB-D}}(H)$ computes the average number the pixels inside the mask for which the rendered camera coordinates $P^{*j}(H)$ and the observed camera coordinates P^j agree, up to a tolerance of 10% of the object diameter d. From the initial pool of 210 hypotheses we select the top 20 according to the score $S_{\text{RGB-D}}(H)$. Finally, for each selected hypothesis, we perform ICP refinement with P as the target, the CAD model vertices as the source, and H_i as initialization. We choose the pose with the lowest ICP fitting error H_{ICP} for further refinement.

Rendering-Based Refinement. Under the assumption that the estimate H_{ICP} is already quite accurate, and using the instance mask M, we perform the following additional refinement: using H_{ICP}, we render the CAD model to obtain a point cloud P_r of the visible object surface. This is in contrast to the previous ICP refinement where all CAD model vertices were used. We fit P_r inside the mask M to the observed point cloud P via ICP, to obtain a refining transformation H_{ref}. This additional step pushes P_r towards the observed point cloud P, providing a further refinement to H_{ICP}. The final pose is thus obtained by $H_{\text{RGB-D}}^* = H_{\text{ICP}} * H_{\text{ref}}$.

Our instance-based approach is a clear advantage in both refinement steps, since we can use the estimated mask to precisely carve out the observed point cloud for ICP.

RGB Setup. Given RGB data, we follow Brachmann *et al.* [17] and estimate the pose of the objects through hypotheses sampling [16] and pre-emptive RANSAC [43]. At this stage, the predicted object mask M and the predicted object coordinates C inside the mask are available. For each pixel j at the 2D position p_j inside M, the object coordinate network estimates a 3D point C^j in the local object coordinate system. Thus, we can sample 2D-3D correspondences between 2D points of the image and 3D object coordinate points from the area inside the object mask. Our goal is to search for a pose hypothesis H^* which
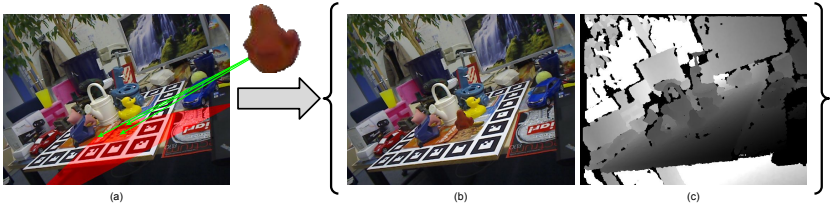
Fig. 2: **Object centric data augmentation pipeline.** (a) If the cropped object (Ape) is inserted within the red area, it can cause a physically plausible occlusion for the center object (Can). (b) shows the resulting augmented RGB image, and (c) shows the resulting augmented depth image.

maximizes the following score:

$$S_{\mathrm{RGB}}(H) = \sum_{j \in M} \left[\|p_j - AHC^j\|_2 < \tau_{\mathrm{in}} \right], \tag{2}$$

where A is the camera projection matrix, τ_{in} is a threshold, and $[\cdot]$ is 1 if the statement inside the bracket is true, otherwise 0. The score $S_{\mathrm{RGB}}(H)$ counts the number of pixel-residuals of re-projected object coordinate estimates which are below τ_{in}. We use pre-emptive RANSAC to maximize this objective function. We start by drawing four correspondences from the predicted mask M. Then, we solve the perspective-n-point problem (PnP) [44,45] to obtain a pose hypothesis. If the re-projection error of the initial four correspondences is below threshold τ_{in} we keep the hypothesis. We repeat this process until 256 pose hypotheses have been collected. We score each hypothesis with $S_{\mathrm{RGB}}(H)$, but only using a sub-sampling of N pixels inside the mask for faster computation. We sort the hypotheses by score and discard the lower half. We refine the remaining hypotheses by re-solving PnP using their inlier pixels according to $S_{\mathrm{RGB}}(H)$. We repeat scoring with an increased pixel count N, discarding and refining hypotheses until only one hypothesis H^*_{RGB} remains as the final estimated pose.

3.4   Data Augmentation

Data augmentation is crucial for creating the amount of data necessary to train a deep neural network. Additionally, data augmentation can help to reduce dataset bias, and introduce novel examples for the network to train on. One possibility for data augmentation is to paste objects on a random background, where mutually overlapping objects occlude each other. This is done *e.g.* in [17,13,14] and we found this strategy sufficient for training our instance segmentation network in step 1. However, the resulting images and occlusion patterns are highly implausible, especially for RGB-D data where objects float in the scene, and occlude each other in physically impossible ways. Training the object coordinate network in step 2 with such implausible data made it difficult for the network to converge and also introduced bias towards impossible object occlusion configurations. In the following, we present an object-centric data augmentation strategy which generates plausible object occlusion patterns, and analyze its impact on
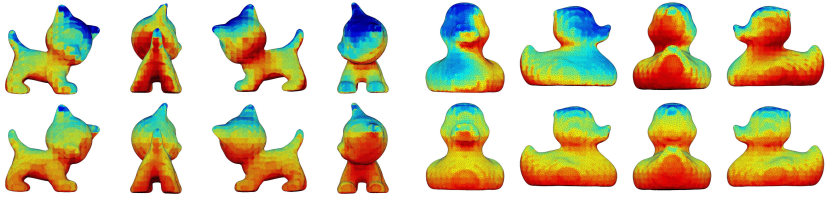
Fig. 3: **Impact of our data augmentation.** The top row illustrates the on-object occlusion distribution of the base training set before augmentation and the bottom row shows the same for augmented data using our object centric data augmentation. For a given part of the model, red indicates that the part is often occluded, while blue corresponds to rare occlusion in a given dataset.

the dataset. We assume that for each target object k in the set of all known objects $\mathcal{K}$, a sequence of images is available where the object is not occluded. For each image, we compute the ground plane on which the target object stands on, 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 target object, representing a cone with an opening angle of $90°$. Next, we search for images of other objects in $\mathcal{K}$, where the ground plane normal is close to that of the target object, and which are located in the defined surface of interest, based on their distance from the 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.

To analyze the impact of our data augmentation scheme, we visualize the distribution of partial occlusion on the object surface in the following way: we first discretize the 3D bounding box surrounding each object into $20 \times 20 \times 20$ voxels. Using the ground truth 6D pose and the 3D CAD model, we can render the full mask of the object. Each pixel that lies inside the rendered mask but not inside the ground truth mask is occluded. We can look-up the ground truth object coordinate of each occluded pixel, and furthermore the associated bounding box voxel. We use the voxels as histogram bins and visualize the occlusion frequency as colors on the surface of the 3D CAD model.

The impact of our object-centric data augmentation for two objects of the LINEMOD dataset [24] is illustrated in Fig. 3. Firstly, by looking at the visualization (top row), we notice 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 of step 2. In the second row, we see that the augmented data has a more regular distribution of occlusion. This visualization reveals the bias in the base training set, and demonstrates the efficacy of our object-centric data augmentation procedure in creating 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 which we use 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, we analyze 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. Please see the supplemental materials for a complete list of parameter settings of our pipeline.

4.1   Datasets and Implementation

We evaluate our approach on *occludedLINEMOD*, a dataset published by Brachmann *et al.* [16]. It was created from the LINEMOD dataset [24] 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 eight 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.* [4]. For ease of reference we call this the LINEMOD-M dataset. LINEMOD-M comes with ground truth annotations of seven 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 our instance segmentation network with a wide range of object viewpoints and diverse occlusion examples, we create synthetic images in the following way. We use RGB backgrounds from the NYUD dataset [46], and randomly overlay them with objects picked from the original LINEMOD dataset [24]. While this data is physically implausible, we found it sufficient for training the instance segmentation component of our pipeline. We combine these synthetic images with LINEMOD-M to obtain 9000 images with ground truth instance masks. We use Mask R-CNN [12] as our instance segmentation method. For training, we use a learning rate of $1e$-3, momentum of 0.9 and weight decay of $1e$-4. We initialize Mask R-CNN with weights trained on ImageNet [47], and finetune on our training set.

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 the test sequence and our training data are strictly separated, *i.e.* we did not use parts of the test sequence for data augmentation. We trained our object coordinate network by minimizing a robust Huber loss

function [7] using ADAM [48]. We train a separate network for each object. We rescale inputs and ground truth outputs for the network to 256x256px patches.

4.2   Pose Estimation Accuracy

RGB Setup. We estimate object poses from RGB images ignoring the depth channel. We evaluate the performance using the *2D Projection* metric introduced by Brachmann *et al.* [17]. This metric measures the average re-projection error of 3D model vertices transformed by the ground truth pose and the estimated pose. A pose is accepted if the average re-projection error is less than a threshold.

In Table 1, we compare the performance of our pipeline to existing RGB-based methods using two different thresholds for the 2D projection metric. We see that our approach outperforms the previous works for most of the objects significantly. Our RGB only pipeline surpasses the state-of-the-art for a 5 pixel threshold by 13% and for a 10 pixel threshold by 39% on average. Note that the results of BB8 [13] were obtained from image crops around the ground truth object position. Similar to [13] and [15], we do not report results for *EggBox* since we could not get reasonable results for this extremely occluded object using RGB only. Note that SSD-6D [14] and SSS-6D [15] completely fail for partly occluded objects. We obtained the results of SSS-6D directly from [15], and of SSD-6D [14] using their publicly available source code and their pre-trained model. However, they did not release their pose refinement method, thus we report their performance without refinement. In the supplement, we show the accuracy of SSD-6D using different 2D re-projection thresholds. Most of the detections of SSD-6D are far off (see also their detection performance in Fig. 7, right), therefore we do not expect refinement to improve their results much. We show qualitative pose estimation results for the RGB setting in Fig 4.

Table 1: **Results using RGB only.** Comparison of our pose estimation accuracy for RGB inputs with competing methods. *Italic* numbers were generated using ground truth crops, thus they are not directly comparable.

| | Acceptance Threshold: 5 px | | | Acceptance Threshold: 10 px | | | | |
	BB8[13] (GT crops)	Brachmann [17]	**Ours**	BB8[13] (*GT crops*)	Brachmann [17]	SSD-6D [14]	SSS-6D [15]	**Ours**
Ape	*28.5%*	**31.8%**	24.2%	*81.0%*	51.8%	0.5%	0%	**56.1%**
Can	*1.2%*	4.5%	**30.2%**	*27.8%*	19.1%	0.6%	0%	**72.4%**
Cat	*9.6%*	1.1%	**12.3%**	*61.8%*	7.1%	0.1%	0%	**39.7%**
Duck	*6.8%*	1.6%	**12.1%**	*41.3%*	6.4%	0%	5%	**50.1%**
Glue	*4.7%*	0.5%	**25.9%**	*37.7%*	6.4%	0%	0%	**55.1%**
HoleP.	*2.4%*	6.7%	**20.6%**	*45.4%*	2.6%	0.3%	1%	**61.2%**
Avg	*8.9%*	7.7%	**20.8%**	*49.2%*	17.1%	0.3%	0.01%	**56.0%**

RGB-D Setup. Similar to the RGB setup, we measure accuracy as the percentage of correctly estimated poses. Following Hinterstoisser *et al.* [24], we accept a pose if the average 3D distance between object model vertices transformed using ground truth pose and predicted pose lies below 10% of the object diameter.

Fig. 4: **Qualitative results from the RGB setup.** From left to right: input image, our results, results of SSD-6D [14].

Fig. 5: **Qualitative results from the RGB-D setup.** Our approach reliably estimates poses for objects which are heavily occluded. The middle column shows estimated object masks of our instance segmentation step.

In Fig. 6, left, we compare the performance of our approach to Michel *et al.* [4] and Hinterstoisser *et al.* [5]. We significantly outperform the state-of-the-art on average by 6%, and show massive improvements for some objects. Fig. 5 shows qualitative results from our pipeline. Fig. 6, right represents the percentage of correct poses as a function of occluded object surface. We see that for cases of mild occlusion, our method surpasses accuracy of 90% for all objects. For cases of heavy occlusion (above 60%) our method can still recover accurate poses.

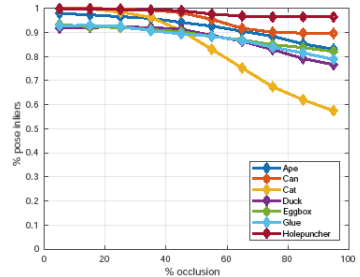| Object	Michel et al. [4]	Hinterstoisser et al. [5]	Ours
Ape	80.7%	81.4%	**83.0%**
Can	88.5%	**94.7%**	89.6%
Cat	**57.8%**	55.2%	57.5%
Duck	74.4%	**79.7%**	76.6%
Eggbox	47.6%	65.5%	**82.1%**
Glue	73.8%	52.1%	**78.8%**
Holep.	96.3%	95.5%	**97.0%**
Avg.	74.2%	74.9%	**80.7%**

Fig. 6: **Left.** Comparison of our pose estimation accuracy (RGB-D) with competing methods. **Right.** The percentage of correctly estimated poses as a function of the level of occlusion.

Ablation Study. We investigate the contribution of each step of our method towards the final pose estimation accuracy for the RGB-D setup. As discussed before, our method consists of three steps, namely instance mask estimation, object coordinate regression and pose estimation. We compare to the method of Brachmann *et al.* [16] which has similar steps, namely soft segmentation (not instance-aware), object coordinate regression, and a final RANSAC-based pose estimation. The first two steps in [16] are implemented using a random forest, compared to two separate CNNs in our system. Fig 7, left shows the accuracy for various re-combinations of these modules. The first row is the standard baseline approach of [16] which achieves an average accuracy of 52.9%. In the second row, we replace the soft segmentation estimated by [16] with a standard instance segmentation method, namely Multi-task Network Cascades (MNC) [11]. The instance masks effectively constrain the 2D search space which leads to better sampling of correspondences between depth points and object coordinate predictions. Next, we replace the object coordinate predictions of the random forest with our CNN-based predictions. Although we still perform the same pose optimization, this achieves an 4.6% performance boost, showing that our encoder-decoder network architecture predicts object coordinates more precisely. Next, we use the instance masks as above and object coordinates from our network with our geometric ICP-based refinement which further boosts the accuracy to 75.7%. Finally, in the last row, we use our full pipeline with masks from Mask R-CNN followed by our other modules to achieve state-of-the-art performance

of 80.7%. The table clearly indicates that the accuracy of our pipeline as a whole improves when any of the modules improve, *e.g.* by better instance segmentation.

Mask	Obj. Coord.	Pose Estimation	Accuracy		Method	MAP
RF[16]	RF[16]	Brachmann [16]	52.9%		Hinterstoisser [3]	0.21
Ours (MNC)	RF[16]	Brachmann [16]	56.4%		Brachmann [17]	0.51
Ours (MNC)	Ours (CNN)	Brachmann [16]	61.0%		SSD-6D [14]	0.38
Ours (MNC)	Ours (CNN)	Ours	75.7%		SSS-6D [15]	0.48
Ours (Mask R-CNN)	Ours (CNN)	Ours	**80.6%**		Ours	**0.84**

Fig. 7: **Left.** Pose estimation accuracies on the RGB-D dataset using various combinations of mask estimation, object coordinates estimation and pose estimation approaches. **Right.** Comparison of 2D detection performance.

4.3   Instance Segmentation

Since we cannot hope to estimate a correct pose for an object that we do not detect, the performance of instance segmentation is crucial for our overall accuracy. Fig. 7, right shows the mean average precision of our method for a 2D bounding box IoU > 0.5 compared to other methods. Since our RGB only instance segmentation network is used for both, the RGB and RGB-D setting, the MAP is equal for both settings. We significantly outperform all the other pose estimation methods, showing that our decoupled instance segmentation step can reliably detect objects, making the task for the following modules considerably easier.

4.4   Object Coordinate Estimation

We trained our object coordinate network 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. 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.

5   Conclusion

We have presented *iPose*, the first deep learning-based approach capable of estimating accurate poses of partly occluded objects. Our approach surpasses the state-of-the-art for both image input modalities, RGB and RGB-D. We attribute the success of our method to our decomposition philosophy, and therefore the ability to leverage state-of-the-art instance segmentation networks. We are also the first to successfully train an encoder-decoder network for dense object coordinate regression, that facilitates our robust geometric pose optimization.

References

1. Lowe, D.G.: Local feature view clustering for 3D object recognition. In: CVPR. (2001)
2. WA, M.C.R.: Kinect for Xbox 360
3. Hinterstoisser, S. Holzer, S., Cagniart, C., Ilic, S., Konolige, K., Navab, N., Lepetit, V.: Multimodal templates for real-time detection of texture-less objects in heavily cluttered scenes. In: ICCV. (2011)
4. Michel, F., Kirillov, A., Brachmann, E., Krull, A., Gumhold, S., Savchynskyy, B., Rother, C.: Global hypothesis generation for 6D object pose estimation. In: CVPR. (2017)
5. Hinterstoisser, S., Lepetit, V., Rajkumar, N., Konolige, K.: Going further with point pair features. In: ECCV. (2016)
6. Krizhevsky, A., Sutskever, I., Hinton, G.E.: ImageNet Classification with Deep Convolutional Neural Networks. In: NIPS. (2012)
7. Girshick, R.: Fast R-CNN. In: ICCV. (2015)
8. Redmon, J., Divvala, S.K., Girshick, R.B., Farhadi, A.: You only look once: Unified, real-time object detection. In: CVPR. (2016)
9. Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S.E., Fu, C., Berg, A.C.: SSD: single shot multibox detector. In: ECCV. (2016)
10. Long, J., Shelhamer, E., Darrell, T.: Fully convolutional networks for semantic segmentation. In: CVPR. (2015)
11. Dai, J., He, K., Sun, J.: Instance-aware semantic segmentation via multi-task network cascades. In: CVPR. (2016)
12. He, K., Gkioxari, G., Dollr, P., Girshick, R.: Mask r-cnn. In: ICCV. (2017)
13. 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: ICCV. (2017)
14. Kehl, W., Manhardt, F., Tombari, F., Ilic, S., Navab, N.: SSD-6D: Making RGB-Based 3D Detection and 6D Pose Estimation Great Again. In: ICCV. (2017)
15. Tekin, B., Sinha, S.N., Fua, P.: Real Time Seamless Single Shot 6D Object Pose Prediction. In: CVPR. (2018)
16. Brachmann, E., Krull, A., Michel, F., Gumhold, S., Shotton, J., Rother, C.: Learning 6D object pose estimation using 3D object coordinates. In: ECCV. (2014)
17. Brachmann, E., Michel, F., Krull, A., Yang, M.Y., Gumhold, S., Rother, C.: Uncertainty-driven 6D pose estimation of objects and scenes from a single RGB image. In: CVPR. (2016)
18. Geiger, A., Lenz, P., Urtasun, R.: Are we ready for Autonomous Driving? The KITTI Vision Benchmark Suite. In: CVPR. (2012)
19. Chen, X., Kundu, K., Zhu, Y., Berneshawi, A.G., Ma, H., Fidler, S., Urtasun, R.: 3D object proposals for accurate object class detection. In: NIPS. (2015)
20. Chen, X., Kundu, K., Zhang, Z., Ma, H., Fidler, S., Urtasun, R.: Monocular 3D object detection for autonomous driving. In: CVPR. (2016)
21. Chen, X., Ma, H., Wan, J., Li, B., Xia, T.: Multi-view 3D object detection network for autonomous driving. In: CVPR. (2017)
22. Chabot, F., Chaouch, M., Rabarisoa, J., Teulière, C., Chateau, T.: Deep MANTA: A coarse-to-fine many-task network for joint 2D and 3D vehicle analysis from monocular image. CVPR (2017)
23. Huttenlocher, D., Klanderman, G., Rucklidge, W.: Comparing images using the Hausdorff distance. IEEE Trans. on PAMI (1993)

24. Hinterstoisser, S., Lepetit, V., Ilic, S., Holzer, S., Bradski, G., Konolige, K., Navab, N.: Model based training, detection and pose estimation of texture-less 3D objects in heavily cluttered scenes. In: ACCV. (2012)
25. Rios-Cabrera, R., Tuytelaars, T.: Discriminatively trained templates for 3D object detection: A real time scalable approach. In: ICCV. (2013)
26. Hodaň, T., Zabulis, X., Lourakis, M., Obdržálek, Š., Matas, J.: Detection and fine 3D pose estimation of texture-less objects in RGB-D images. In: IROS. (2015)
27. Kehl, W., Tombari, F., Navab, N., Ilic, S., Lepetit, V.: Hashmod: A hashing method for scalable 3D object detection. In: BMVC. (2016)
28. Konishi, Y., Hanzawa, Y., Kawade, M., Hashimoto, M.: Fast 6D pose estimation from a monocular image using hierarchical pose trees. In: ECCV. (2016)
29. Hinterstoisser, S., Cagniart, C., Ilic, S., Sturm, P., Navab, N., Fua, P., Lepetit, V.: Gradient response maps for real-time detection of texture-less objects. IEEE Trans. on PAMI (2012)
30. Tejani, A., Tang, D., Kouskouridas, R., Kim, T.K.: Latent-class Hough forests for 3D object detection and pose estimation. In: ECCV. (2014)
31. Zach, C., Penate-Sanchez, A., Pham, M.T.: A dynamic programming approach for fast and robust object pose recognition from range images. In: CVPR. (2015)
32. Doumanoglou, A., Kouskouridas, R., Malassiotis, S., Kim, T.: 6D object detection and next-best-view prediction in the crowd. In: CVPR. (2016)
33. 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: ECCV. (2016)
34. Drost, B., Ulrich, M., Navab, N., Ilic, S.: Model globally, match locally: Efficient and robust 3D object recognition. In: CVPR. (2010)
35. Kendall, A., Grimes, M., Cipolla, R.: PoseNet: A convolutional network for real-time 6-DoF camera relocalization. In: ICCV. (2015)
36. Brachmann, E., Krull, A., Nowozin, S., Shotton, J., Michel, F., Gumhold, S., Rother, C.: DSAC-Differentiable RANSAC for camera localization. In: CVPR. (2017)
37. 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: ICCV. (2015)
38. Behl, A., Hosseini Jafari, O., Mustikovela, S.K., Alhaija, H.A., Rother, C., Geiger, A.: Bounding boxes, segmentations and object coordinates: How important is recognition for 3D scene flow estimation in autonomous driving scenarios? In: ICCV. (2017)
39. Gaidon, A., Wang, Q., Cabon, Y., Vig, E.: Virtual worlds as proxy for multi-object tracking analysis. In: CVPR. (2016)
40. Alhaija, H.A., Mustikovela, S.K., Mescheder, L., Geiger, A., Rother, C.: Augmented reality meets deep learning for car instance segmentation in urban scenes. In: BMVC. (2017)
41. Li, C., Zia, M.Z., Tran, Q., Yu, X., Hager, G.D., Chandraker, M.: Deep supervision with shape concepts for occlusion-aware 3D object parsing. In: CVPR. (2017)
42. Kabsch, W.: A solution for the best rotation to relate two sets of vectors. Acta Crystallographica (1976)
43. Shotton, J., Glocker, B., Zach, C., Izadi, S., Criminisi, A., Fitzgibbon, A.W.: Scene coordinate regression forests for camera relocalization in RGB-D images. In: CVPR. (2013)
44. Gao, X.S., Hou, X.R., Tang, J., Cheng, H.F.: Complete solution classification for the perspective-three-point problem. IEEE Trans. on PAMI (2003)

45. Lepetit, V., Moreno-Noguer, F., Fua, P.: EPNP: An accurate O(n) solution to the PNP problem. IJCV (2009)
46. Nathan Silberman, Derek Hoiem, P.K., Fergus, R.: Indoor segmentation and support inference from rgbd images. In: ECCV. (2012)
47. Deng, J., Dong, W., Socher, R., Li, L.J., Li, K., Fei-Fei, L.: ImageNet: A Large-Scale Hierarchical Image Database. In: CVPR. (2009)
48. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. In: ICLR. (2015)