PoET: Pose Estimation Transformer for Single-View, Multi-Object 6D Pose Estimation

Anonymous Author(s)
Affiliation
Address
email

Abstract: Accurate 6D object pose estimation is an important task for a variety of robotic applications such as grasping or localization. It is a challenging task due to object symmetries, clutter, occlusion and different scenes, but it becomes even more challenging when additional information, such as depth and 3D models, is not provided. We present a transformer-based approach that takes an RGB image as input and predicts a 6D pose for each object in the image. Besides the image, our network does not require any additional information such as depth maps or 3D object models. First, the image is passed through an object detector to generate feature maps and to detect objects. Second, these feature maps are fed into a transformer while the detected bounding boxes are provided as additional information. Afterwards, the output object queries are processed by a separate translation and rotation head. We achieve state of the art results for RGB-only approaches on the challenging YCB-V dataset. We illustrate the suitability of the resulting model as pose sensor for a 6 DoF state estimation task. Code is available at <placeholder>.

Keywords: 6D Pose Estimation, Transformer, Single View, Multi Object, Object-Relative Localization

1 Introduction

Accurately estimating the 6D pose of objects from RGB images is essential for robotics tasks such as grasping or localization [1, 2]. Grasping tasks require the robot to know the exact position of the object such that it can place its end-effector effectively. In autonomous driving it is of utmost importance that the vehicle has sufficient knowledge of its surroundings, which includes the relative 6D pose of all objects in its vicinity. For unmanned aerial vehicle (UAV) navigation, especially in close proximity to people or infrastructure, realizing precise control depends on the estimation of 6D object poses.

In recent years, the task of vision-based 6D pose estimation has received a lot of attention especially with the rise of deep learning. Approaches differ in terms of input data, network architecture, post processing and number of viewpoints. Observing objects from multiple viewpoints introduces constraints to the pose of objects and helps with the estimation. Availability of 3D object models allows for an iterative refinement of an initial pose estimate by either iterative closest point (ICP) [3] matching of pointclouds or by matching keypoints with an perspective-n-point (PnP) [4] algorithm. However, these algorithms are computationally demanding. Besides using 3D object models for post processing, they can be utilized as an additional input to the network. This may include model keypoints and corresponding features or information about the object such as symmetry axes and planes. Additional input information is not limited to the 3D model, but it can also be prior information about the object class or a depth map corresponding to the input RGB image which improves the networks ability to estimate the objects distance to the camera. Although additional input information can greatly benefit the accuracy of the final estimated pose, apart of requiring a

Submitted to the 6th Conference on Robot Learning (CoRL 2022). Do not distribute.

more detailed data base, it results in higher computational complexity and thus, longer run time in comparison to the same architecture taking only RGB images.

While the requirement for real-time performance depends on the specific application, more often the availability of high-quality, detailed 3D models or depth maps is limited. Depending on the type of additional input information, it also requires additional sensor hardware which may not be available during inference. Hence, in this work we focus on a pose estimation framework purely based on RGB images and information provided by a backbone object detector. To the best of our knowledge, we are the first to incorporate global image context information to the pose estimation task by passing multi-scale feature maps to a transformer network and relying only on 2D image information. Our approach does not depend on the number of objects present. Moreover, our framework dubbed PoET (Pose Estimation Transformer) can be used on top of any 2D object detector. We evaluate our approach on the difficult YCB-V [5] dataset and compare our results to state-of-the-art approaches. Finally, we illustrate the suitability of the obtained model as a pose sensor for a 6 DoF state estimation task. Our contributions are the following:

- We present a transformer-based framework that takes a single RGB-image as input, estimates the 6D pose for every object present in the image and can be trained on top of any object detector framework.
- The framework is independent of any additional information which is not contained in the raw RGB image. In particular, it does not depend on depth maps, object symmetries or 3D object models. Hence, our results are achieved without iterative refinement and the whole network can be trained using 3D model independent loss functions.
- We achieve state of the art results on the YCB-V [5] dataset for RGB only methods and competitive results in comparison to approaches utilizing 3D models.
- We show the feasibility of the resulting model to be used as a pose sensor in a 6-DoF localization task.

The rest of the paper is organized as follows. In Section 2 related work for 6D pose estimation is reviewed. Following the presentation of our method and implementation details in Section 3, the conducted experiments and the corresponding results are discussed in Section 4. Additionally, we shortly showcase how PoET and its relative 6D pose estimates can be used for localization in Section 4.2. Finally, the limitations are discussed in Section 5 and the paper is concluded in Section 6. Furthermore, we refer the reader to the supplementary material for qualitative results, an extensive ablation study and additional results on the LM-O [6] dataset.

2 Related Work

Classical image-based 6D pose estimation approaches can be split into feature-based methods and template-based methods. For the latter one, object templates are generated from the corresponding 3D models and pose is determined by matching the template against the input image [7, 8]. While template-based approaches work well on texture-less objects, they are prone to fail in scenarios where objects are occluded. In contrast to that, in feature-based methods local features are extracted from the image and then matched to the 3D model to determine correspondences [9, 10]. Based on this 2D-3D correspondences, the 6D pose of the object can be derived. While these approaches can handle occlusion of objects, they require textures in order to perform the matching. If RGB-D images are available, the additional information provided by the depth can be used to improve the initial pose estimate by iterative refinement [11, 12] using algorithms such as e.g. ICP [3]. Even though those methods usually achieve state-of-the-art performance, they are computationally expensive and the availability of depth maps is not guaranteed.

In recent years, the advancements made in deep learning methods for computer vision tasks have been applied to single-view, image-based 6D pose estimation, either to replace components of classical approaches [13, 14, 15, 16], or as end-to-end learned methods, where the 6D pose is

directly estimated from the input using convolutional neural networks (CNNs). Xiang et al. [5] proposed with PoseCNN a CNN-based method to regress the 3D translation and rotation of each object present in the image. Moreover, during training they employ an object symmetry-aware loss function. Li et al. [17] proposed a framework that introduces prior knowledge about the object class into the network and perform pose estimation by binning the translation and rotation.

Aside from depth images, using 3D object models for the pose estimation task yielded promising results [18, 19, 20, 21, 22]. The approaches differ in terms of how the 3D model is used. Kehl et al. [18] and Li et al. [19] predict the 6D pose of objects and then refine the estimated pose. With GDR-Net, Wang et al. [23] are able to integrate PnP into an end-to-end trainable network. Similarly, Li et al. [21] use the 3D model of an object for iterative pose refinement. Given an initial pose estimate, a network is trained to iteratively refine the pose by matching a rendered image created from the 3D model to the original image. Labbé et al. [22] takes this approach a step further and applies it to multiple viewpoints of the same scene resulting in improved pose estimation. A different approach was chosen by Billings and Johnson-Roberson [20] as they provide the 3D model as an additional input to their network dubbed SilhoNet. Their network predicts the object silhouette and a 3D translation vector derived from the 2D bounding box position. Based on the predicted silhouette, the 3D rotation of the object is estimated and corrected for silhouette symmetric objects - a significant restriction, as the real error in rotation for symmetric objects can be very large.

Similar to our approach, Amini et al. [24] presented a transformer-based architecture to directly regress the 6D pose for multiple objects contained in a single image. By extending the Detection Transformer [25] with translation and rotation heads and using an extended Hungarian loss, they are able to train the whole network in an end-to-end fashion. However, they require the object 3D model as they use a symmetry aware loss [5].

Our approach does not require any additional information such as depth maps, 3D models or known object symmetries but instead directly estimates the translation and rotation of objects in the camera coordinate frame from a single RGB image. In contrast to other methods that work with regions of interest for pose estimation, we keep the complete image feature map and provide the regions of interest as an additional input to our transformer network.

3 Method

We present a novel transformer-based neural network for the 6D pose estimation task. Taking a single RGB image as its input, the 6D pose of every object detected in the image is predicted simultaneously. After generating (multi-scale) feature maps by passing the image through an object detector, they are processed by a transformer architecture. At the end, translation and rotation are estimated in a decoupled manner. In this section we, first present the general structure of our network. Afterwards, we talk about specific implementation details and data preparation.

3.1 Network Architecture

Fig. 1 shows a detailed overview of our network architecture which consists of three steps. First, the input image is passed through a backbone object detector network. Both the generated (multi-scale) feature maps and the predicted bounding boxes are used in subsequent processing. Important to note is that PoET can be trained on top of any object detector architecture and thus extend a pre-trained object detection framework to include 6D object pose estimation.

Second, the (multi-scale) feature maps used for the object detection step are passed to the encoder of a multi-head attention-based transformer. Our transformer architecture is a modified version of the *Deformable DETR* transformer module proposed by Zhu et al. [27]. In contrast to the original *DETR* [25], the deformable transformer allows to process multi-scale feature maps similar to state of the art object detectors. Moreover, by only attending to a limited number of feature map keypoints in the decoder, the transformer achieves a higher pass through rate. Additionally, a deformable transformer shows faster convergence rates than a regular transformer architecture. The main idea

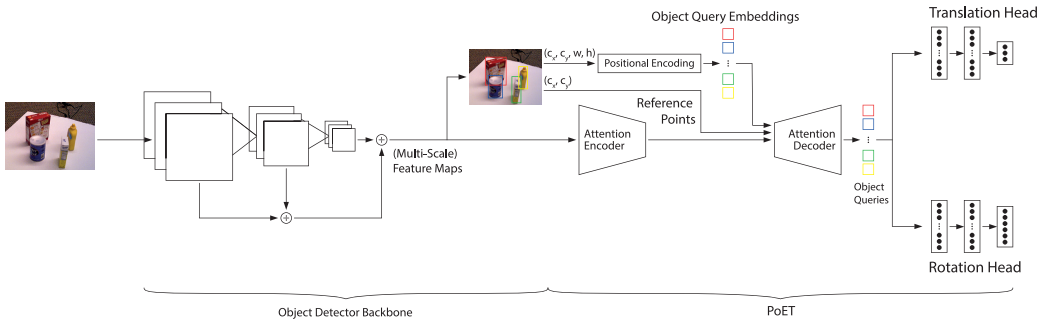

Figure 1: Overview of the PoET network architecture for single-view, multi-object 6D pose estimation. Bounding box information for each detected object is passed to the transformer as an object query. Afterwards, for each object query the egocentric 3D translation and 6D rotation is predicted. From the 6D rotation the 3D rotation is determined according to [26].

behind using a transformer architecture for feature map refinement is to generate features that capture the global information contained in the image. For example, such additional information might be the image location of other objects present or general information of the overall scene extracted from the image. While the encoder of the deformable transformer is kept unchanged, we modified the decoder to incorporate more information from the object detection step:

First, the learned object query embedding is replaced by bounding box information. For each detected object, the bounding box center coordinates (c_x, c_y), the width w and the height h are normalized and then positional encoded using gamma encoding [28] to generate the object query embeddings. The embedding dimension L is chosen such that $2 \cdot n_p \cdot L$ equals the hidden dimension d_h of the transformer. In our case, the number of parameters n_p equals to 4. Moreover, the inter-query attention heads ensure that information is properly propagated between the different object queries. Second, as mentioned earlier, the *Deformable DETR* originally only attends to a limited number of keypoints which are randomly sampled around reference points. Instead of predicting reference points from query embeddings by a trainable fully connected layer, we directly feed the normalized center coordinates (c_x, c_y) as the reference points to the decoder. By feeding this additional information to the decoder along with the encoder refined image feature maps and the inter-query attention heads, the decoder generates new object query embeddings which should not only contain local information regarding the object but also global information extracted from the image. Third, the object queries outputted by the transformer are passed through a translation and rotation head. This allows us to simultaneously estimate the pose for multiple objects independent of how many objects are present and which class they belong to. As we approach the pose estimation problem from a global image context perspective by extracting features from the whole image, the network directly estimates the translation and rotation with respect to the camera.

Our translation head is a simple multi-layer perceptron (MLP) with input dimension d_h, one hidden layer and output dimension 3. We directly predict the translation $\tilde{t} = (\tilde{t}_x, \tilde{t}_y, \tilde{t}_z)$ with respect to the camera frame. Given the ground truth translation t, our translation head is trained with a simple L2-loss defined as

$$L_t = ||t - \tilde{t}||_2 \,. \tag{1}$$

The rotation prediction head is identical to the translation head besides the output dimension. For estimating the rotation, we use the 6D rotation representation proposed by Zhou et al. [26] as this representation does not suffer from discontinuities with respect to learning as e.g. quaternions do. Hence, the network predicts a 6 dimensional output vector. Afterwards, the 6D representation is used to determine the estimated rotation matrix $\tilde{R} \in SO(3)$ as described in [26]. The rotation head is trained using a geodesic loss [29] given by

$$L_{rot} = \arccos \frac{1}{2} \left(Tr \left(R\tilde{R}^T \right) - 1 \right) , \tag{2}$$

where $Tr(\cdot)$ is the matrix trace operator. To ensure numerical stability of the loss during training, the argument of the $\arccos$ is clamped between $-1 + \epsilon$ and $1 - \epsilon$, where $\epsilon = 1e - 6$. Our whole

network is then trained with a weighted multi-task loss expressed as

$$L = \lambda_t L_t + \lambda_{rot} L_{rot} \ , \tag{3}$$

where the loss is calculated for each object and then averaged across all objects present across all images in the batch. Moreover, PoET can be trained either class specific or class agnostic. In the class specific case, the translation and rotation output dimension are changed to $3 \cdot n_{cls}$ and $6 \cdot n_{cls}$, respectively, and the final output is chosen depending on the class predicted for the bounding box. However, no class information is fed into the transformer.

3.2 Implementation Details & Data Preparation

While PoET can be trained on top of any object detection framework, we used Scaled-YOLOv4 [30] as the backbone object detector as it offers a good trade-off between speed and accuracy. An MS-COCO [31] pre-trained Scaled-YOLOv4 network is fine tuned for 10 epochs on the YCB-V dataset for the object detection task. During the training of PoET, the weights of the object detector backbone are frozen.

In order to utilize the benefits of batch processing, the number of input object queries has to be constant across all images. This is usually not the case, as the number of objects present in an image can vary significantly. Therefore, the number of object queries n_q is fixed for a specific dataset such that it is equal to the maximum number of objects present in an image across the whole dataset. If fewer objects are present in the image than n_q, the remainder of the object queries is filled up with dummies. Such a dummy object is assigned a bounding box $(c_x, c_y, w, h) = (-1, -1, -1, -1)$, a dummy class and a dummy query embedding. In case that the object detector predicts more objects than the number of allowed object queries, the top-n_q predictions are chosen based on the classification score. Due to the transformer's attention mechanism, it should learn to not focus on dummy embedding feature vectors. Besides that, for the loss calculation as well as the evaluation, dummy object queries are disregarded as object queries are matched to ground truth objects based on bounding box center distance, predicted class and generalized intersection over union (GIoU). We want to highlight that dummy embeddings are not utilized in inference as only single images are processed. During training, the ground truth bounding box and class are used instead of the predictions made by the object detector backbone.

We implement PoET using PyTorch [32] and train it on a single NVIDIA GeForce RTX 3090 for 50 epochs using AdamW [33] with a learning rate of $2e-5$ and a batch size of 16. Our best performing network has five encoder and decoder layers, $d_h = 256$, 16 attention heads and a positional embedding dimension of $L = 32$. The network is simultaneously trained for 3D translation and 3D rotation estimation and the weighting parameters are set to $\lambda_t = 2$ and $\lambda_{rot} = 1$, such that both losses are in the same value range after scaling. During evaluation, our whole pipeline consumes around 5.3GB of VRAM and runs with 71 FPS.

YCB-V[5] contains 21 objects [34] and consists of 92 video sequences totalling 133,827 frames (`real`). Each video sequence contains multiple objects and the level of object occlusion and scene clutter varies between the sequences. Throughout the whole dataset at most 9 objects are contained in one scene and thus we fix n_q to 10. Following the proposed and commonly used data split [5, 35, 21, 22, 17, 20], we train our network on 80 of those 92 video sequences and reserve the remaining 12 sequences for testing. Out of these 12 sequences the challenging keyframes are used for evaluation. Moreover, the original authors of the dataset provide 80,000 synthetic images generated from 2D projections of the 3D object models (`synt`). We include these synthetic images during training for better network generalization. The background of the synthetic images is blank and thus, to improve network generalization even further, we randomly choose an image from the MS COCO dataset [31] to function as a background. Even though, the BOP challenge also provides photorealistic images (`pbr`) for YCB-V, we do not utilize this data at all.

Besides adding background images for synthetic data, we also perform random RGB augmentation including not only modifying the image color, sharpness, brightness and contrast, but also blurring the image and converting it to grayscale. However, images are not scaled, flipped or cropped.

Table 1: Comparison with state of the art on YCB-V. We report the AUC of ADD-S. *gt* denotes results achieved with providing ground truth ROIs to network. The 3D model row indicates how the object model is used in the method. Either to calculate the loss function, as an additional input, for symmetry reduction or for PnP or iterative refinement (IR) based pose estimation. (*) denotes symmetric objects. Bold and italic values indicate the state of the art for results for methods not based on PnP/IR, when working on ground truth or predicted ROIs respectively

Method	PoseCNN [5]	SilhoNet [20]	SilhoNet$_{gt}$	MCN[17]	MCN$_{gt}$	T6D [24]	PoET$_{gt}$	PoET	GDR-Net [23]	CosyPose [22]	DeepIM[21]
3D Model	Loss	Input + Sym	Input + Sym	2D	2D	Loss	2D	2D	PnP	IR	IR
master chef can	84.0	84.0	83.6	87.8	91.2	*91.9*	**92.9**	88.4	96.6	-	93.1
cracker box	76.9	73.5	88.4	64.3	78.5	*86.6*	**90.4**	80.5	84.9	-	91.0
sugar box	84.3	86.6	88.8	82.4	85.1	90.3	**94.5**	*92.4*	98.3	-	96.2
tomato soup can	80.9	88.7	89.4	87.9	93.3	88.9	**94.0**	*91.4*	96.1	-	92.4
mustard bottle	90.2	89.8	91.0	92.5	91.9	*94.7*	**94.8**	91.7	99.5	-	95.1
tuna fish can	87.9	89.5	89.9	84.7	**95.2**	*92.2*	94.0	90.4	95.1	-	96.1
pudding box	79.0	60.1	89.1	51.0	84.9	85.1	**93.8**	*89.0*	94.8	-	90.7
gelatin box	87.1	*92.7*	**94.6**	86.4	92.1	86.9	92.7	91.7	95.3	-	94.3
potted meat can	78.5	78.8	84.8	83.1	90.8	83.5	**94.1**	*91.2*	82.9	-	86.4
banana	85.9	80.7	88.7	79.1	70.0	*93.8*	**94.3**	89.5	96.0	-	72.3
pitcher base	76.8	91.7	91.8	84.8	91.1	*92.3*	**94.3**	91.7	98.8	-	94.6
bleach cleanser	71.9	73.6	72.0	76.0	86.8	83.0	**92.6**	*85.4*	94.4	-	90.3
bowl*	69.7	79.6	72.5	76.1	85.0	*91.6*	**92.1**	90.5	84.0	-	81.4
mug	78.0	86.8	92.1	*91.4*	91.9	89.8	**94.1**	*91.4*	96.9	-	91.3
power drill	72.8	56.5	82.9	76.0	87.2	*88.8*	**94.3**	*88.8*	91.9	-	92.3
wood block*	65.8	66.2	79.2	54.0	87.2	*90.7*	**92.0**	75.7	77.3	-	81.9
scissors	56.2	49.1	78.3	71.6	80.2	*83.0*	**92.5**	75.2	68.4	-	75.4
large marker	71.4	75.0	**83.1**	60.1	66.4	74.9	81.6	*81.2*	87.4	-	86.2
large clamp*	49.9	69.2	84.5	66.8	86.5	78.3	**95.7**	*88.6*	69.3	-	74.3
extra large clamp*	47.0	72.3	88.4	61.1	79.5	54.7	**96.0**	*83.5*	73.6	-	73.2
foam brick*	87.8	77.9	88.4	60.9	79.2	*89.9*	**89.7**	81.3	90.4	-	81.9
All	75.9	79.6	85.8	75.1	86.9	86.2	**92.8**	*87.1*	89.1	89.8	88.1

4 Results & Experiments

In this section, we present PoET's performance and its evaluation results on the YCB-V benchmark dataset for the 6D pose estimation task. For evaluation, we use the AUC of ADD(-S) metric [5] and additionally report the average translation and rotation error in cm and degree respectively as done by [20]. We compare our results to state of the art, single-view, RGB-based approaches and list the results of other approaches as reported in the corresponding work. Afterwards, we showcase how PoET's relative pose estimates can be used for camera localization. In the supplementary material, PoET's performance on the LM-O dataset [6] is reported, an extensive ablation study is conducted and qualitative results are presented. The ablation study investigates the influence of the object detector backbone, transformer modifications, data augmentation, network architecture and rotation representation on PoET's performance.

4.1 6D Pose Estimation

Our best performing network is class specific, consists of 5 encoder and decoder layers with 16 attention heads, and trained according to Section 3.2. Methods that rely on an object detector to predict regions of interest (ROI) usually do not elaborate whether and, if so, how multiple predictions, bad predictions and missing predictions are treated within their approach. Therefore, to allow for a fair comparison we also report the results of PoET and other approaches given ground truth bounding boxes in addition to the results on predicted bounding boxes.

In Table 1 we report our results for the AUC of ADD-S metric. Both on predicted and ground truth bounding boxes, our PoET framework outperforms, overall and also for most classes by a significant margin, other state of the art RGB-based methods that either work on the whole input image [5, 24] or that predict ROIs for pose regression [17]. We also outperform SilhoNet [20] which incorporates additional information from the 3D model into its network input and reduces predicted rotations by known geometric symmetries of the underlying objects. This shows that PoET and its feature maps containing global image context information reduces the need for any additional inputs. This is especially highlighted when comparing the performance of PoET to the ROI-based approaches SilhoNet and MCN [17] in the case where all networks are provided with ground truth bounding boxes. Only models that explicitly utilize 3D object models during inference, either through PnP during the estimation [23] or by performing iterative refinement after estimating an initial pose [22, 21], achieve slightly better results but this comes at the cost of significantly increased computational complexity and the need for accurate 3D models to be known a priori.

Table 2: Comparison of average translation error in cm on YCB-V. Bold and italic values indicate the state of the art for results, when working on ground truth or predicted ROIs respectively

Method	PoseCNN[5]	SilhoNet [20]	SilhoNet$_{gt}$	PoET$_{gt}$	PoET
master chef can	3.29	3.02	3.14	**1.37**	*2.26*
cracker box	4.02	5.24	2.38	**1.48**	*3.14*
sugar box	3.06	2.10	1.67	**0.94**	*1.42*
tomato soup can	3.02	2.40	2.24	**1.09**	*1.62*
mustard bottle	1.72	1.65	1.41	**0.94**	*1.42*
tuna fish can	2.41	*1.57*	1.49	**0.95**	1.79
pudding box	3.69	7.15	1.91	**1.01**	*1.94*
gelatin box	2.49	*1.09*	**0.79**	1.20	1.41
potted meat can	3.65	4.30	2.74	**1.13**	*1.75*
banana	2.43	4.12	2.59	**1.06**	*1.95*
pitcher base	4.43	*1.31*	1.29	**0.95**	1.55
bleach cleanser	4.86	3.60	3.99	**1.09**	*2.47*
bowl*	5.23	3.30	4.08	**1.51**	*1.76*
mug	4.00	2.61	1.43	**1.28**	*1.85*
power drill	4.59	6.77	3.19	**0.98**	*2.29*
wood block*	6.34	5.59	3.23	**1.41**	*4.75*
scissors	6.40	9.91	2.59	**1.38**	*3.72*
large marker	3.89	3.24	**2.31**	2.68	*2.75*
large clamp*	9.79	6.27	3.51	**0.98**	*2.33*
extra large clamp*	8.36	4.86	2.12	**0.91**	*3.10*
foam brick*	*2.48*	3.98	2.31	**1.90**	3.42
All	4.16	3.49	2.45	**1.20**	*2.12*

Table 3: Comparison of average rotation error in degrees on YCB-V. † indicates that the rotation predictions are reduced by geometric symmetries as described in [20]

Method	PoseCNN [5]	PoseCNN†	SilhoNet†[20]	SilhoNet$^{\dagger}_{gt}$	PoET$_{gt}$	PoET
master chef can	50.7	7.57	*1.21*	**1.11**	89.25	80.12
cracker box	*19.69*	*19.69*	19.86	**9.53**	9.68	21.87
sugar box	9.29	9.29	12.28	11.50	**3.95**	*4.40*
tomato soup can	18.23	8.40	*1.91*	**1.82**	50.97	49.29
mustard bottle	9.94	9.59	*5.78*	**5.07**	23.71	27.73
tuna fish can	32.80	12.74	*1.46*	**1.50**	60.30	63.72
pudding box	10.20	10.20	20.95	18.39	**6.36**	*6.87*
gelatin box	*5.25*	*5.25*	12.52	8.48	**6.69**	7.19
potted meat can	28.67	19.74	*7.27*	10.93	**5.06**	*6.75*
banana	*15.48*	*15.48*	16.29	**5.70**	7.90	20.40
pitcher base	11.98	11.98	*6.64*	**6.61**	7.51	8.04
bleach cleanser	*20.85*	*20.85*	51.28	48.42	**16.32**	21.93
bowl*	75.53	75.53	49.95	53.95	**16.06**	*25.71*
mug	19.44	19.44	18.14	7.02	**3.86**	*5.59*
power drill	9.91	9.91	30.54	10.66	**5.92**	*6.45*
wood block*	23.63	23.63	25.52	23.23	**5.88**	*14.32*
scissors	43.98	43.98	155.53	154.82	**3.19**	*6.27*
large marker	92.44	13.59	*10.44*	**10.72**	24.95	25.91
large clamp*	38.12	38.12	*3.54*	6.03	**2.61**	4.88
extra large clamp*	34.18	34.18	29.18	7.30	**2.38**	*26.01*
foam brick*	22.67	22.67	*13.84*	**17.36**	37.20	36.34
All	27.79	17.82	*16.04*	**13.48**	23.65	27.26

The improved performance of our PoET compared to other RGB-based models in terms of the ADD-S score is due to a better estimate of the 3D translation as can be seen in Table 2. Again, ground truth-based results are also presented. In contrast to directly estimating the 3D translation like PoET, PoseCNN as well as SilhoNet determine the 3D translation by estimating the depth and center pixel coordinates of an object and then reprojecting it using the known camera intrinsic parameters, which is considered the easier task to learn [5]. MCN treats the translation estimation as a classification problem by binning the translation space. In addition to the global image information provided by the multi-scale feature maps, our approach also learns to model the camera intrinsics which results in a more accurate estimation of translation.

Regarding the 3D rotation estimation, we achieve the same average error as regular PoseCNN. Not surprisingly, networks that reduce the possible rotation space based on known symmetries of the objects achieve a better result but with limited applicability to real world scenarios. The mean average 3D rotation error is shown in Table 3. The influence of reducing the rotation by geometrical symmetries (†) is highlighted for PoseCNN. Since PoET makes use of the full multi-scale RGB feature maps, it outperforms SilhoNet for objects that have no symmetries in the silhouette space and only performs significantly worse for objects with rotational symmetries around an axis in the 3D space, however without requiring a priori knowledge about the 3D shape of objects or restricting the rotation space based on symmetries.

We refer the reader to the supplementary material for an extensive ablation study, additional results and analysis. Moreover, we also report there our networks' performance for the stricter BOP[35] and AUC of ADD [5] metrics as well as for another benchmark dataset, namely LM-O [6].

4.2 Localization

PoET is well suited for vision-based object-relative localization. Given a set of landmarks with known ground truth pose (R^i_{wo}, R^i_{wo}), a single frame that captures at least one of those landmarks can be used in combination with PoET to localize the camera by estimating the relative pose $(\tilde{R}^i_{co}, \tilde{t}^i_{co})$ to all landmarks present in the frame. For each landmark, the estimated camera pose $(\tilde{R}^i_{wc}, \tilde{t}^i_{wc})$ can be determined by

$$\tilde{R}^i_{wc} = R^i_{wo}\tilde{R}^{i^T}_{co} \quad \text{and} \quad {}^W\tilde{t}^i_{wc} = {}^Wt^i_{wo} - \tilde{R}^i_{wc}\,{}^W\tilde{t}^i_{co}. \tag{4}$$

The final camera pose $(\tilde{R}_{wc}, \tilde{t}_{wc})$ can then be determined by taking the average over all landmarks present in the image. YCB-V's test sequences offer 12 different camera trajectories and each with a different constellation of multiple objects serving as landmarks. For each individual frame we estimate the camera pose and compare it to the ground truth. In Fig. 2 we show an example trajectory for a single sequence. For further examples we refer the reader to the supplementary material.

We compare three different approaches: using all detected objects (`all`), perform simple outlier rejection by taking and choosing the hypothesis the majority of objects agree on (`out`) or by incorporating the camera pose estimate from the previous frame into the outlier rejection in cases

with multiple hypothesis having the same number of votes (`prev`). For the sake of comparison, we also calculate the camera pose by choosing the estimated camera pose being closest to the current ground truth camera pose (`best`). Moreover, we differentiate between PoET being provided either object detections from the backbone (`bb`) or ground truth bounding box information (`gt`). We calculate the sample error mean and standard deviation across all frames for the position and attitude and summarize them in Table 4. Simple outlier rejection greatly benefits the localization error. This is due to wrong relative object pose estimates having a large impact on the final estimated camera pose when taking a simple average. Incorporating the camera pose estimate from the previous frame only slightly improves the localization as it only helps in multi-hypothesis cases. In comparison to the estimated trajectories using the best estimated camera pose, our outlier rejection based trajectories perform worse with around factor two. Nonetheless, the localization results when considering only individual frames are remarkable. The difference in performance between backbone and ground truth detections is due to the backbone potentially missing an object or assigning a wrong class and thus throwing off the estimate. Summarizing, PoET achieves sufficient localization accuracy such that it can be used as a pose sensor for robotics.

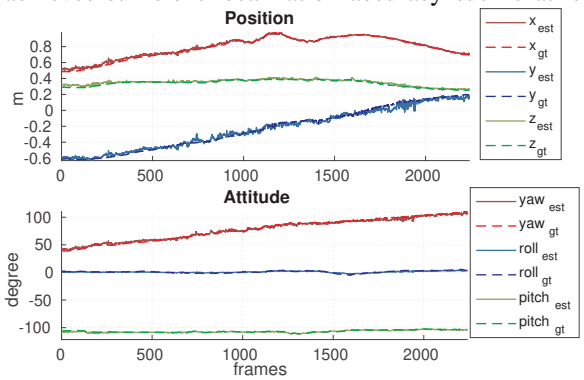

Figure 2: Example trajectory of a single sequence for `gt` and `out`. We plot the ground truth and estimated position and attitude.

detection mode		(x, y, z) $\pm$ (σ_x, σ_y, σ_z) [mm]
bb	all	(119, 157, 52)$\pm$(121, 131, 59)
	out	(56, 54, 28) $\pm$ (100, 106, 36)
	prev	(42, 43, 26) $\pm$ (56, 59, 27)
	best	(28, 27, 23) $\pm$ (31, 35, 22)
gt	all	(102, 138, 41) $\pm$ (107, 117, 53)
	out	(38, 34, 19) $\pm$ (80, 80, 28)
	prev	(32, 28, 19) $\pm$ (45, 39, 21)
	best	(21, 19, 17) $\pm$ (28, 32, 18)
		(yaw, roll, pitch) $\pm$ (σ_y, σ_r, σ_p) [deg]
bb	all	(19.6, 7.1, 12.4) $\pm$ (18.6, 11.8, 14.9)
	out	(5.4, 1.6, 2.2) $\pm$ (10.4, 2.8, 4.3)
	prev	(4.1, 1.4, 1.9) $\pm$ (5.0, 2.2, 2.1)
	best	(2.6, 1.5, 1.7) $\pm$ (3.0, 1.8, 1.7)
gt	all	(18.0, 6.7, 11.0) $\pm$ (16.5, 10.8, 12.9)
	out	(3.7, 1.2, 3.0) $\pm$ (8.9, 2.0, 3.0)
	prev	(2.8, 1.2, 1.4) $\pm$ (3.7, 2.2, 1.5)
	best	(2.0, 1.3, 1.2) $\pm$ (3.0, 1.4, 1.3)

Table 4: Sample error mean and standard deviation of camera pose across all 12 test sequence frames. Position and attitude are reported in mm and degree.

5 Limitations

In Section 4.1 it was discussed that PoET is outperformed by methods utilizing 3D object information for rotation estimation in particular for objects that have rotation symmetric silhouettes. Wrongly estimated rotations lead to the assumption that the camera views the objects from a different angle resulting in wrong hypotheses for the localization task as discussed in Section 4.2. Moreover, the low resolution of the images in the YCB-V dataset means that RGB textures are not as dominant, especially when augmentation is used. This is the main reason why PoET has difficulties to estimate silhouette rotation symmetric objects that are not symmetric in RGB space.

6 Conclusion

In this work we presented a novel, transformer-based framework for multi-object 6D pose estimation. It can be used on top of any object detector and the only input required is a single RGB image. The image is passed through an object detector backbone to create (multi-scale) image feature maps and to detect objects. Bounding box information of detected objects is fed into the transformer decoder which improves the learning. By taking the whole image into consideration during the estimation process, our framework does not rely on any additional information. We outperform other RGB-based methods by a wide margin on the YCB-V dataset. This is especially important for scenarios where no detailed 3D models or prior object information is available, or where computational efficiency is required and thus any input besides the RGB image has to be dropped. Moreover, we highlighted how PoET's relative 6D object pose estimation can be used as a pose sensor for robotics tasks.

References

[1] H. Song, W. Choi, and H. Kim. Robust vision-based relative-localization approach using an rgb-depth camera and lidar sensor fusion. *IEEE Transactions on Industrial Electronics*, 63(6): 3725–3736, 2016.

[2] V. Loing, R. Marlet, and M. Aubry. Virtual training for a real application: Accurate object-robot relative localization without calibration. *International Journal of Computer Vision*, 126 (9):1045–1060, 2018.

[3] P. J. Besl and N. D. McKay. Method for registration of 3-d shapes. In *Sensor Fusion IV: Control Paradigms and Data Structures*, volume 1611, pages 586–606. International Society for Optics and Photonics, 1992.

[4] M. A. Fischler and R. C. Bolles. Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6):381–395, 1981.

[5] Y. Xiang, T. Schmidt, V. Narayanan, and D. Fox. Posecnn: A convolutional neural network for 6d object pose estimation in cluttered scenes. 2018.

[6] 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 *Proceedings of the IEEE international conference on computer vision*, pages 954–962, 2015.

[7] S. Hinterstoisser, C. Cagniart, S. Ilic, P. Sturm, N. Navab, P. Fua, and V. Lepetit. Gradient response maps for real-time detection of textureless objects. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(5):876–888, 2011.

[8] Z. Cao, Y. Sheikh, and N. K. Banerjee. Real-time scalable 6dof pose estimation for textureless objects. In *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2441–2448. IEEE, 2016.

[9] D. G. Lowe. Object recognition from local scale-invariant features. In *Proceedings of the seventh IEEE International Conference on Computer Vision*, volume 2, pages 1150–1157. Ieee, 1999.

[10] G. Pavlakos, X. Zhou, A. Chan, K. G. Derpanis, and K. Daniilidis. 6-dof object pose from semantic keypoints. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2011–2018. IEEE, 2017.

[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 *Asian Conference on Computer Vision*, pages 548–562. Springer, 2012.

[12] F. Michel, A. Kirillov, E. Brachmann, A. Krull, S. Gumhold, B. Savchynskyy, and C. Rother. Global hypothesis generation for 6d object pose estimation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 462–471, 2017.

[13] K. Park, T. Patten, and M. Vincze. Pix2pose: Pixel-wise coordinate regression of objects for 6d pose estimation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 7668–7677, 2019.

[14] T. Hodan, D. Barath, and J. Matas. Epos: Estimating 6d pose of objects with symmetries. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11703–11712, 2020.

[15] 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 *Proceedings of the IEEE International Conference on Computer Vision*, pages 3828–3836, 2017.

[16] S. Peng, Y. Liu, Q. Huang, X. Zhou, and H. Bao. Pvnet: Pixel-wise voting network for 6dof pose estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4561–4570, 2019.

[17] C. Li, J. Bai, and G. D. Hager. A unified framework for multi-view multi-class object pose estimation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 254–269, 2018.

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

[19] Z. Li, G. Wang, and X. Ji. Cdpn: Coordinates-based disentangled pose network for real-time rgb-based 6-dof object pose estimation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 7678–7687, 2019.

[20] G. Billings and M. Johnson-Roberson. Silhonet: An rgb method for 6d object pose estimation. *IEEE Robotics and Automation Letters*, 4(4):3727–3734, 2019.

[21] Y. Li, G. Wang, X. Ji, Y. Xiang, and D. Fox. Deepim: Deep iterative matching for 6d pose estimation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 683–698, 2018.

[22] Y. Labbé, J. Carpentier, M. Aubry, and J. Sivic. Cosypose: Consistent multi-view multi-object 6d pose estimation. In *European Conference on Computer Vision*, pages 574–591. Springer, 2020.

[23] G. Wang, F. Manhardt, F. Tombari, and X. Ji. Gdr-net: Geometry-guided direct regression network for monocular 6d object pose estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16611–16621, 2021.

[24] A. Amini, A. S. Periyasamy, and S. Behnke. T6d-direct: Transformers for multi-object 6d pose direct regression. In *DAGM German Conference on Pattern Recognition*, pages 530–544. Springer, 2021.

[25] N. Carion, F. Massa, G. Synnaeve, N. Usunier, A. Kirillov, and S. Zagoruyko. End-to-end object detection with transformers. In *European Conference on Computer Vision*, pages 213–229. Springer, 2020.

[26] Y. Zhou, C. Barnes, J. Lu, J. Yang, and H. Li. On the continuity of rotation representations in neural networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5745–5753, 2019.

[27] X. Zhu, W. Su, L. Lu, B. Li, X. Wang, and J. Dai. Deformable detr: Deformable transformers for end-to-end object detection. *arXiv preprint arXiv:2010.04159*, 2020.

[28] B. Mildenhall, P. P. Srinivasan, M. Tancik, J. T. Barron, R. Ramamoorthi, and R. Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. In *European Conference on Computer Vision*, pages 405–421. Springer, 2020.

[29] S. Mahendran, H. Ali, and R. Vidal. 3d pose regression using convolutional neural networks. In *Proceedings of the IEEE International Conference on Computer Vision Workshops*, pages 2174–2182, 2017.

[30] C.-Y. Wang, A. Bochkovskiy, and H.-Y. M. Liao. Scaled-yolov4: Scaling cross stage partial network. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13029–13038, 2021.

[31] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick. Microsoft coco: Common objects in context. In *European Conference on Computer Vision*, pages 740–755. Springer, 2014.

[32] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala. Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc., 2019. URL http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf.

[33] I. Loshchilov and F. Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017.

[34] B. Calli, A. Singh, A. Walsman, S. Srinivasa, P. Abbeel, and A. M. Dollar. The ycb object and model set: Towards common benchmarks for manipulation research. In *2015 International Conference on Advanced Robotics (ICAR)*, pages 510–517. IEEE, 2015.

[35] T. Hodaň, M. Sundermeyer, B. Drost, Y. Labbé, E. Brachmann, F. Michel, C. Rother, and J. Matas. Bop challenge 2020 on 6d object localization. In *European Conference on Computer Vision*, pages 577–594. Springer, 2020.