17th International Conference on Intelligent Autonomous Systems (IAS), Zagreb, Croatia, June 2022.
Best Paper Award.

YOLOPose: Transformer-based Multi-Object 6D Pose Estimation using Keypoint Regression

Arash Amini*, Arul Selvam Periyasamy*, and Sven Behnke

Autonomous Intelligent Systems, University of Bonn, Germany
periyasa@ais.uni-bonn.de
*Equal contribution.

Abstract. 6D object pose estimation is a crucial prerequisite for autonomous robot manipulation applications. The state-of-the-art models for pose estimation are convolutional neural network (CNN)-based. Lately, Transformers, an architecture originally proposed for natural language processing, is achieving state-of-the-art results in many computer vision tasks as well. Equipped with the multi-head self-attention mechanism, Transformers enable simple single-stage end-to-end architectures for learning object detection and 6D object pose estimation jointly. In this work, we propose YOLOPose (short form for You Only Look Once Pose estimation), a Transformer-based multi-object 6D pose estimation method based on keypoint regression. In contrast to the standard heatmaps for predicting keypoints in an image, we directly regress the keypoints. Additionally, we employ a learnable orientation estimation module to predict the orientation from the keypoints. Along with a separate translation estimation module, our model is end-to-end differentiable. Our method is suitable for real-time applications and achieves results comparable to state-of-the-art methods.

Autonomous robotic object manipulation in real-world scenarios depends on high-quality 6D object pose estimation. Such object poses are also crucial in many other applications like augmented reality, autonomous navigation, and industrial bin picking. In recent years, with the advent of convolutional neural networks (CNNs), significant progress has been made to boost the performance of object pose estimation methods. Due to the complex nature of the problem, the standard methods favor multi-stage approaches, i.e., feature extraction followed by object detection and/or instance segmentation, target object crop extraction, and, finally, 6D object pose estimation. In contrast, Carion et al. [2] introduced DETR, a Transformer-based single-stage architecture for object detection. In our previous work [1], we extended the DETR model with the T6D-Direct architecture to perform multi-object 6D pose direct regression. Compared to multi-stage CNN-based methods that employ components like bounding box proposals, region of interest (RoI) pooling, non-maximum suppression (NMS) to construct end-to-end differentiable pipelines, the T6D-Direct model learns object detection and 6D pose estimation jointly. Taking advantage of the *pleasingly parallel* nature of the Transformer architecture, the T6D-Direct model predicts

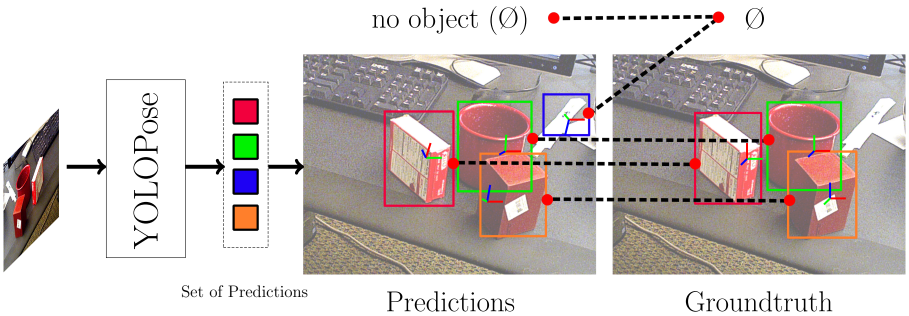

Fig. 1. Proposed YOLOPose approach. Our model predicts a set with a fixed cardinality N. Each element in the set corresponds to an object prediction and after predicting all the objects in the given input image, the rest of the elements are padded with Ø as no object predictions. The predicted and the groundtruth sets are matched using bipartite matching and the model is trained to minimize the Hungarian loss between the matched pairs. Our model is end-to-end differentiable.

6D poses for all the objects in an image in one forward-pass. Despite the advantages of the architecture and its impressive performance, the overall 6D pose estimation accuracy of T6D-Direct, which directly regresses translation and orientation components of the 6D object poses, is inferior to state-of-the-art CNN-based methods, especially in rotation estimation. Instead of directly regressing the translation and orientation components, the keypoint-based methods predict the 2D pixel projection of 3D keypoints and use the PnP algorithm to recover the 6D pose. We extend our T6D-Direct model to learn sparse 2D-3D correspondences. Our proposed model performs keypoint direct regression instead of the standard heatmaps for predicting the spatial position of the keypoints in a given RGB image and uses a multi layer perceptron (MLP) to learn the orientation component of 6D object pose from the keypoints. Another independent MLP serves as the translation estimator. In short, our contributions include:

1. A Transformer-based real-time single-stage model for multi-object monocular 6D pose estimation using keypoint regression,
2. a learnable rotation estimation module to estimate object orientation from a set of keypoints to develop an end-to-end differentiable architecture for pose estimation, and
3. results comparable to state-of-the-art pose estimation methods on the YCB-Video dataset while being capable of real-time inference.

1 Related Work

1.1 RGB Object Pose Estimation

The recent significant progress in the task of 6D object pose estimation from RGB images is driven—like in many computer vision tasks—by deep learning methods. The methods for the object pose estimation from RGB images can be broadly classified into three major categories, namely direct regression methods, keypoint-based methods, and refinement-based methods. Direct regression methods formulate the problem of pose estimation as a regression of continuous translation and rotation components, whereas keypoint-based methods predict the location of projection of some of the specific keypoints or the 3D coordinates of the visible pixels of an object in an image and use the PnP algorithm to recover the 6D pose from the estimated 2D-3D correspondences. The PnP algorithm is used in conjunction with RANSAC to improve the robustness of the pose estimation.

Some examples for direct regression methods include [1, 22, 29, 30] and examples for keypoint-based methods include [8, 9, 21, 24, 28]. One important detail to note is that, except for [1] all the other methods use multi-staged CNNs. In the first stage, the model performs object detection and/or semantic or instance segmentation to detect the objects in the given RGB image. Using the object detections from the first stage, a crop containing the target object is extracted. In the second stage, the model predicts the 6D pose of the target object from the image crop. In terms of the 6D pose prediction accuracy, keypoint-based methods perform considerably better than the direct regression methods [7], though this performance gap is shrinking [1].

The third category of the pose estimation methods is refinement-based. These methods formulate the task of pose estimation as iterative pose refinement, i.e., the target object is rendered according to the current pose estimate, and a model is trained to estimate a pose update that minimizes the pose error between the groundtruth and the current pose prediction. Refinement-based methods [11, 15, 19, 23, 26] achieve the highest pose prediction accuracy among all categories [7].

1.2 Learned PnP

Given a set of 3D keypoints and their corresponding 2D projections and the camera intrinsics, the PnP algorithm is used to recover the 6D object pose. The standard PnP algorithm [5] and its variant EPnP [13] are used in combination with RANSAC to improve the robustness against outliers. Both PnP and RANSAC are not trivially differentiable. In order to realize an end-to-end differentiable pipeline for the 6D object pose estimation, Wang et al. [29], and Hu et al. [8] proposed a learning-based PnP module. Similarly, Li et al. [14] introduced a learnable 3D Lifter module to estimate vehicle orientation. Lately, Chen et al. [3] proposed to differentiate PnP using the implicit function theorem. Although a generic differentiable PnP has many potentials, due to the overhead incurred during training, we opt for a simple MLP that estimates the orientation component given the 2D keypoints.

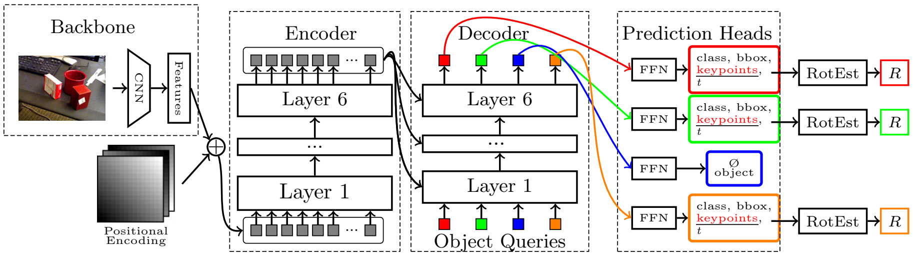

Fig. 2. YOLOPose architecture in detail. Given an RGB input image, we extract features using the standard ResNet model. The extracted features are supplemented with positional encoding and provided as input to the Transformer encoder. The encoder module consists of 6 standard encoder layers with skip connections. The output of the encoder module is provided to the decoder module along with N object queries and the decoder module also consists of 6 standard decoder layers with skip connections generating N output embeddings. The output embeddings are processed with FFNs to generate a set of N elements in parallel. Each element in the set is a tuple consisting of bounding box, class probability, translation and interpolated bounding box keypoints. A learnable rotation estimation module is employed to estimate object orientation R from the predicted keypoints.

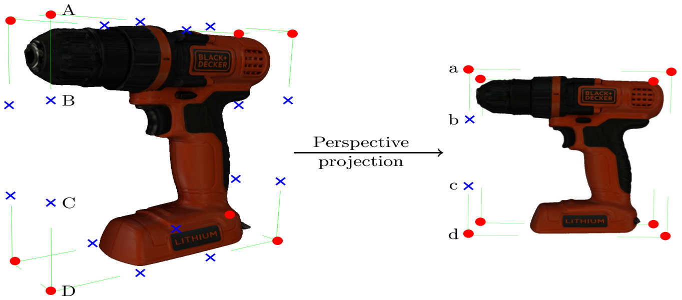

Fig. 3. Interpolated bounding box points. Bounding box points are indicated with red dots, and the interpolated points are indicated with blue crosses. The cross-ratio of every four collinear points is preserved during perspective projection,e.g., the cross-ratio the points A, B, C, and D remains the same in 3D and, after perspective projection, in 2D.

2 Method

2.1 Multi-Object Keypoint Regression as Set Prediction

Following DETR [2] and T6D-Direct [1], we formulate the problem of keypoint regression as a set prediction problem. Given an RGB input image, our model outputs a set of elements with a fixed cardinality N. Each element in the set is a tuple containing 2D bounding boxes, class probability, translation, and keypoints. The 2D bounding boxes are represented with the center coordinates, height, and width proportional to the image size. The class probability is predicted using a softmax function. We regress the translation component directly, where our translation representation follows Xiang et al. [30]. The exact choice of the keypoints is discussed in Section 2.2. The number of objects present in an image varies. Therefore, to enable output sets with fixed cardinality, we choose N to be larger than the expected maximum number of objects in an image in the dataset and introduce a null object class $\emptyset$. The $\emptyset$ class is analogous to the background class used in semantic segmentation models. In addition to predicting the corresponding classes for objects present in the image, our model is trained to predict $\emptyset$ class for the rest of the elements in the set.

A set is an unordered collection of elements. To facilitate comparing the groundtruth set and the predicted set, we use bipartite matching [2, 10, 27] to find the permutation of the predicted elements that minimizes the matching cost. Given the $\emptyset$ class padded groundtruth set y of cardinality N containing labels $y_1, y_2, ..., y_n$ for n elements, the predicted set denoted by $\hat{y}$, we search for the optimal permutation $\hat{\sigma}$ among the possible permutations $\sigma \in \mathfrak{S}_N$ that minimizes the matching cost $\mathcal{L}_{match}$. Formally,

$$\hat{\sigma} = \underset{\sigma \in \mathfrak{S}_N}{\arg\min} \sum_{i}^{N} \mathcal{L}_{match}(y_i, \hat{y}_{\sigma(i)}). \tag{1}$$

Although each element of the set is a tuple containing four components, bounding box, class probability, translation, and keypoints, we use only the bounding box and class probability components to define the cost function. In practice, omitting the other components in the cost function definition does not hinder the model's ability in learning to predict the keypoints.

2.2 Keypoints Representation

An obvious choice for 3D keypoints is the eight corners of the 3D bounding box [20]. Peng et al. [21] argued that predicting the projection of 3D bounding boxes of an object might be difficult for a CNN-based model since the projection might lie outside of the object boundary in the RGB image. To alleviate this issue, they proposed to use the Farthest Point Sampling algorithm (FPS) to sample eight keypoints on the surface of the object meshes. Li et al. [14] defined the 3D representation of an object as sparse interpolated bounding boxes (IBBs), depicted in Fig. 3, and exploited the property of perspective projection that the cross-ratio of every four collinear points in 3D (A, B, C, and D as illustrated in Fig. 3) is preserved under perspective projection in 2D [6]. The cross-ratio consistency is enforced by an additional component in the loss function that the model learns to minimize during training. We further investigate these keypoints representations in Section 4 and present our results in Table 3.

2.3 RotEst

For each object, from the estimated pixel coordinates of the 32 keypoints (the eight corners of the 3D bounding box and the 24 intermediate bounding box keypoints), the RotEst module predicts the object orientation represented as the 6D continuous representation in SO(3) [31]. Formally, the RotEst module takes a 64-dimensional vector (32 pixel coordinates) and generates a 6D object orientation estimate. We implement the RotEst module using six fully connected layers with hidden dimension 1024 and dropout probability of 0.5.

2.4 Loss Function

The Hungarian loss consists of four components: class probability loss, bounding box loss, keypoint loss, and pose loss. The class probability loss and the bounding box loss follow the DETR model [2].

Class Probability Loss The class probability loss function is the standard negative log-likelihood. Since we choose the cardinality of the set to be higher than the expected maximum number of objects in an image, the Ø class appears disproportionately often. Thus, we weigh the loss for the Ø class with a factor of 0.4.

Bounding Box Loss We use a weighted combination of the Generalized IoU (GIoU) [25] and ℓ_1-loss with 2 and 5 factors, respectively, for the bounding box loss.

Keypoint Loss Having the groundtruth K_i and the model output $\hat{K}_{\hat{\sigma}(i)}$ our keypoints loss can be represented as:

$$\mathcal{L}_{keypoints}(K_i, \hat{K}_{\hat{\sigma}(i)}) = \gamma ||K_i - \hat{K}_{\hat{\sigma}(i)}||_1 + \delta \mathcal{L}_{\mathcal{CR}}, \tag{2}$$

where γ and δ are hyperparameters. The first part of the keypoints loss is the ℓ_1 loss, and for the second part, we employ the cross-ratio loss $\mathcal{L}_{\mathcal{CR}}$ provided in Equation 3 to enforce the cross-ratio consistency in the keypoints loss as proposed by Li et al. [14]. This loss is self-supervised by preserving the cross-ratio of each line to be 4/3. The reason is that after camera projection of the 3D bounding box on the image plane, the cross-ratio of every four collinear points remains the same.

$$\mathcal{L}_{\mathcal{CR}} = Smooth\ell_1(\mathrm{CR}^2 - \frac{||c-a||^2||d-b||^2}{||c-b||^2||d-a||^2}), \tag{3}$$

where CR^2 is chosen since $||.||^2$ can be easily computed using vector inner product. Given four collinear points A, B, C and D and their predicted 2D projections a, b, c, and d, the groundtruth cross-ratio CR is defined as:

$$\text{CR} = \frac{||C - A||\ ||D - B||}{||C - B||\ ||D - A||} = \frac{4}{3}. \tag{4}$$

Pose Loss We supervise the rotation R and the translation t individually via employing PLoss and SLoss from [30] for rotation and ℓ_1 loss for translation:

$$\mathcal{L}_{pose}(R_i, t_i, \hat{R}_{\sigma(i)}, \hat{t}_{\sigma(i)}) = \mathcal{L}_{rot}(R_i, \hat{R}_{\sigma(i)}) + ||t_i - \hat{t}_{\sigma(i)}||_1, \tag{5}$$

$$\mathcal{L}_{rot} = \begin{cases} \frac{1}{|\mathcal{M}_i|} \displaystyle\sum_{\mathrm{x}_1 \in \mathcal{M}_i} \min_{\mathrm{x}_2 \in \mathcal{M}_i} ||R_i \mathrm{x}_1 - \hat{R}_{\sigma(i)} \mathrm{x}_2||_1 \text{ if sym,} \\ \frac{1}{|\mathcal{M}_i|} \displaystyle\sum_{\mathrm{x} \in \mathcal{M}_i} ||R_i \mathrm{x} - \hat{R}_{\sigma(i)} \mathrm{x}||_1 \text{ otherwise,} \end{cases} \tag{6}$$

where $\mathcal{M}_i$ indicates the set of 3D model points. Here, we subsample 1.5K points from meshes provided with the dataset. R_i is the groundtruth rotation, and t_i is the groundtruth translation. $\hat{R}_{\sigma(i)}$ and $\hat{t}_{\sigma(i)}$ are the predicted rotation and translation, respectively.

2.5 Model Architecture

The proposed YOLOPose architecture is inspired by T6D-Direct [1]. The model consists of a ResNet backbone followed by an encoder-decoder based Transformer and MLP prediction heads to predict a set of tuples described in Section 2.1. CNN architectures have several inductive biases designed into them [4, 12]. These strong biases enable CNNs to learn efficient local spatial features in a fixed neighborhood defined by the receptive field to perform well on many computer vision tasks. In contrast, Transformers, aided by self-attention, are suitable for learning spatial features over the entire image. This makes the Transformer architecture ideal for multi-object pose estimation. In this section, we describe the individual components of the YOLOPose architecture.

Backbone Network We use a ResNet50 backbone for extracting features from the given RGB image. For an image size of height H and width W, the backbone network extracts 2048 low-resolution feature maps of size H/32×W/32. We then use 1×1 convolution to reduce the 2048 feature dimensions to a lower number of d dimensions. The standard Transformer models are designed to process vectors. Therefore, to enable processing the d×H/32×W/32 features, we vectorize them to $d \times \frac{H}{32}\frac{W}{32}$.

Positional Encodings Multi-head self-attention, the core component of the Transformer model, is permutation-invariant. Thus, the Transformer architecture ignores the order of the input vectors. We employ the standard solution of supplementing the input vectors with absolute positional encoding following Carion et al. [2] to provide the Transformer model with spatial information of the pixels. The positional embeddings are added elementwise to the backbone feature vectors before feeding them to the Transformer encoder as input.

Encoder The Transformer encoder module consists of six encoder layers with skip connections. Each layer performs multi-head self-attention of the input vectors. The embeddings used in our model are 256-dimensional vectors.

Decoder From the encoder output embedding and N positional embedding inputs, the decoder generates N output embeddings using the multi-head self-attention and cross-attention mechanisms, where N is the cardinality of the predicted set. Unlike the fixed positional encoding used in the encoder, we use learnable positional encoding in the decoder, called *object queries*. From the N decoder output embeddings, we use feed-forward prediction heads to generate a set of N output tuples independently.

FFN For each object query decoder output, we use four feed-forward prediction heads shared across object queries to predict the class probability, bounding box, translation, and keypoints independently. Prediction heads are straightforward three-layer MLPs with hidden dimension 256 in each layer and ReLU activation.

3 Evaluation

In this section, we evaluate the performance of our proposed YOLOPose model and compare it with other state-of-the-art 6D pose estimation methods.

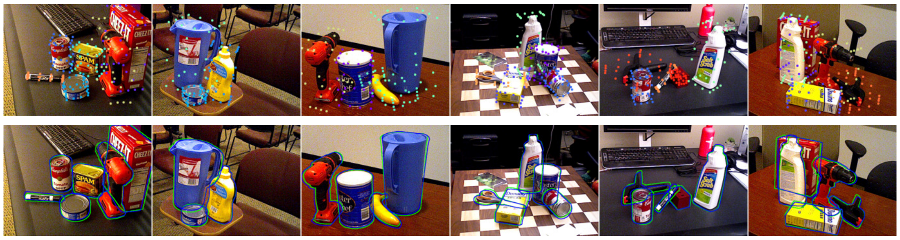

Fig. 4. Qualitative results on YCB-V test set. Top row: The predicted IBB keypoints overlaid on the input images. Bottom row: Groundtruth and predicted object poses visualized as object contours in green and blue colors, respectively.

3.1 Dataset

We use the challenging YCB-Video (YCB-V) [30] dataset to evaluate the performance of our model. YCB-V provides bounding box, segmentation, and 6D pose annotation for 133,936 RGB-D images. Since our model is RGB-based, we do not use the provided depth information. The dataset is generated by capturing video sequences of a random subset of objects from a total of 21 objects placed in tabletop configuration. There are 92 video sequences, out of which twelve are used for testing and the rest are used for training. The objects used exhibit varying geometric shapes, reflectance properties, and symmetry. Thus, YCB-V is a challenging dataset for benchmarking 6D object pose estimation methods. YCB-V also provides high-quality meshes for all 21 objects. Mesh points from these objects are used in computing the evaluation metrics discussed in Section 3.2. Hodaň et al. [7] provided a variant of YCB-V[1] as a part of the BOP challenge in which the centers of the 3D bounding boxes are aligned with the origin of the model coordinate system, and the groundtruth annotations are converted correspondingly. We use the BOP variant of the YCB-V dataset. Additionally, we use the COCO dataset [17] for pretraining our model on the task of object detection.

3.2 Metrics

Xiang et al. [30] proposed area under the curve (AUC) of the ADD and ADD-S metrics for evaluating the accuracy of non-symmetric and symmetric objects, respectively. Given the groundtruth 6D pose annotation with rotation and translation components R and t, and the predicted rotation and translation components $\hat{R}$ and $\hat{t}$, the ADD metric is the average ℓ_2 distance between the subsampled mesh points $\mathcal{M}$ in the groundtruth and the predicted pose. In contrast, the symmetry aware ADD-S metric is the average distance between the closest subsampled mesh points $\mathcal{M}$ in the groundtruth and predicted pose. A predicted pose is considered correct if the ADD metric is below $0.1\,\mathrm{m}$. The ADD(-S) metric is the combination of ADD-S for symmetric objects and ADD for the non-symmetric objects. Formally,

$$\mathrm{ADD} = \frac{1}{|\mathcal{M}|} \sum_{x \in \mathcal{M}} \|(Rx + t) - (\hat{R}x + \hat{t})\|, \tag{7}$$

$$\text{ADD-S} = \frac{1}{|\mathcal{M}|} \sum_{x_1 \in \mathcal{M}} \min_{x_2 \in \mathcal{M}} \|(Rx_1 + t) - (\hat{R}x_2 + \hat{t})\|. \tag{8}$$

3.3 Hyperparameters

The γ and δ hyperparameters in $\mathcal{L}_{keypoints}$ (Eq. (2)) are set to 10 and 1, respectively. While computing the Hungarian loss, the pose loss component is weighted

[1] `https://bop.felk.cvut.cz/datasets/`

down by a factor 0.02. The cardinality of the predicted set N is set to 20. The model takes images of the size 640 × 480 as input and is trained using the AdamW optimizer [18] with an initial learning rate of $2e^{-4}$ for 335K iterations. The learning rate is decayed to $2e^{-5}$ after 271K iterations, and the batch size is 32. Moreover, gradient clipping with a maximal gradient norm of 0.1 is applied. In addition to YCB-V dataset images, we use the synthetic dataset provided by PoseCNN for training our model.

3.4 Results

Table 1. Comparison of keypoints-based method with state-of-the-art methods on YCB-V. P.E=1 denotes one model for all objects, whereas P.E.=(N) denotes the usage of object specific models. The symmetric objects are denoted by *, and the best results are shown in bold.

Method	PoseCNN [30]		PVNet [21]	GDR-Net [29]		T6D-Direct [1]		YOLOPose (Ours)		DeepIM [15]	
P.E.	1		N	1		1		1		1	
Metric	AUC of ADD-S	AUC of ADD(-S)	AUC of ADD(-S)	AUC of ADD-S	AUC of ADD(-S)	AUC of ADD-S	AUC of ADD(-S)	AUC of ADD-S	AUC of ADD(-S)	AUC of ADD-S	AUC of ADD(-S)
master_chef_can	84.0	50.9	81.6	96.6	71.1	91.9	61.5	91.3	64.0	93.1	71.2
cracker_box	76.9	51.7	80.5	84.9	63.5	86.6	76.3	86.8	77.9	91.0	83.6
sugar_box	84.3	68.6	84.9	98.3	93.2	90.3	81.8	92.6	87.3	96.2	94.1
tomato_soup_can	80.9	66.0	78.2	96.1	88.9	88.9	72.0	90.5	77.8	92.4	86.1
mustard_bottle	90.2	79.9	88.3	99.5	93.8	94.7	85.7	93.6	87.9	95.1	91.5
tuna_fish_can	87.9	70.4	62.2	95.1	85.1	92.2	59.0	94.3	74.4	96.1	87.7
pudding_box	79.0	62.9	85.2	94.8	86.5	85.1	72.7	92.3	87.9	90.7	82.7
gelatin_box	87.1	75.2	88.7	95.3	88.5	86.9	74.4	90.1	83.4	94.3	91.9
potted_meat_can	78.5	59.6	65.1	82.9	72.9	83.5	67.8	85.8	76.7	86.4	76.2
banana	85.9	72.3	51.8	96.0	85.2	93.8	87.4	95.0	88.2	91.3	81.2
pitcher_base	76.8	52.5	91.2	98.8	94.3	92.3	84.5	93.6	88.5	94.6	90.1
bleach_cleanser	71.9	50.5	74.8	94.4	80.5	83.0	65.0	85.3	73.0	90.3	81.2
bowl*	69.7	69.7	89.0	84.0	84.0	91.6	91.6	92.3	92.3	81.4	81.4
mug	78.0	57.7	81.5	96.9	87.6	89.8	72.1	84.9	69.6	91.3	81.4
power_drill	72.8	55.1	83.4	91.9	78.7	88.8	77.7	92.6	86.1	92.3	85.5
wood_block*	65.8	65.8	71.5	77.3	77.3	90.7	90.7	84.3	84.3	81.9	81.9
scissors	56.2	35.8	54.8	68.4	43.7	83.0	59.7	93.3	87.0	75.4	60.9
large_marker	71.4	58.0	35.8	87.4	76.2	74.9	63.9	84.9	76.6	86.2	75.6
large_clamp*	49.9	49.9	66.3	69.3	69.3	78.3	78.3	92.0	92.0	74.3	74.3
extra_large_clamp*	47.0	47.0	53.9	73.6	73.6	54.7	54.7	88.9	88.9	73.3	73.3
foam_brick*	87.8	87.8	80.6	90.4	90.4	89.9	89.9	90.7	90.7	81.9	81.9
MEAN	75.9	61.3	73.4	89.1	80.2	86.2	74.6	**90.1**	**82.6**	88.1	81.9

In this section, we present the quantitative and qualitative results of our method. In Section 3.4, we provide the quantitative per class area under the accuracy curve (AUC) of the ADD-S and ADD(-S) metrics. Except for DeepIM, a refinement-based method and PVNet, an indirect method, all other methods estimate the 6D pose directly. Our method outperforms all of the competing approaches. Additionally in Table 2, we present Average Recall (AR) of ADD(-S) 0.1d, and AUC of ADD-S and ADD(-S) of the state-of-the-art methods. In terms of the AR of ADD(-S) and AUC of ADD-S metrics, our method achieves state-of-the-art results among the pose estimators. Note that the pose refinement approach, CosyPose, achieves the best result in terms of the AUC of ADD(-S)

Table 2. Results on YCB-V.

Method	ADD(-S)	AUC of ADD-S	AUC of ADD(-S)	Inference Time [s]
CosyPose[†] [11]	-	89.8	**84.5**	0.395
PoseCNN [30]	21.3	75.9	61.3	-
SegDriven [9]	39.0	-	-	-
Single-Stage [8]	53.9	-	-	-
GDR-Net [29]	49.1	89.1	80.2	0.065
T6D-Direct [1]	48.7	86.2	74.6	**0.017**
YOLOPose (Ours)	**65.0**	**90.1**	82.6	**0.017**

[†] Refinement-based method.

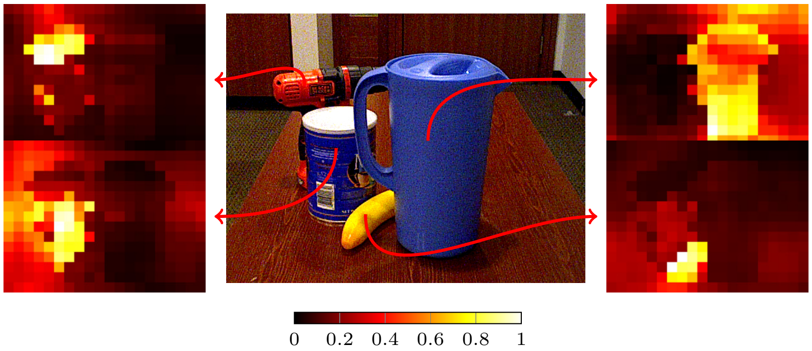

Fig. 5. Encoder self-attention. We visualize the self-attention maps for four pixels belonging to four objects in the image. Note that for each object, roughly all the corresponding pixels are attended.

metric. However, in terms of the approach pose refinement methods are orthogonal to the pose estimation methods and can benefit from the improved pose estimation accuracy. Furthermore, pose estimation models with admissible accuracy avoid the need for training an additional pose refinement model and enable faster inference time. We present exemplar qualitative results in Fig. 4. In Fig. 5, we visualize encoder self-attention of four different pixels belonging to four different objects and in Fig. 6, we visualize the decoder cross-attention corresponding to four different object detections. In both visualizations, the attended regions correspond to the spatial position of the object in the image very well.

3.5 Inference Time Analysis

In terms of the inference speed, one of the major advantages of our architecture is that the feed-forward prediction networks (FFN) can be executed in parallel for each object. Thus, irrespective of the number of objects in an image, our model generates pose predictions in parallel. In Table 2, we present the inference

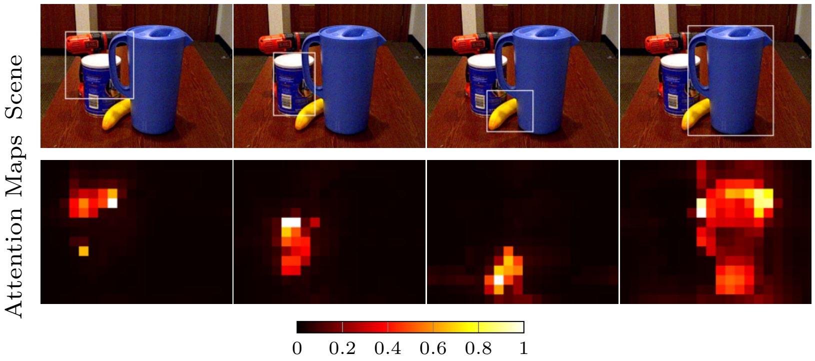

Fig. 6. Top: Object detections predicted by bounding boxes in a given image. Bottom: Decoder cross-attention maps for the object queries corresponding to the predictions in the first row.

time results for 6D pose estimation. Our method operates at ∼59 FPS on an NVIDIA 3090 GPU and Intel 3.70 GHz CPU and is, hence, suitable for real-time applications.

4 Ablation Study

In contrast to the standard approach of estimating the 2D keypoints and using PnP solver—which is not trivially differentiable—to estimate the 6D object pose, we use the learnable RotEst module to estimate the object orientation from a set of predicted interpolated keypoints. In this section, we analyze the effectiveness of our RotEst module and the choice of the keypoint representation.

4.1 Effectiveness of Keypoints Representations

We compare various keypoint representations, namely 3D bounding box keypoints (BB), random keypoints sampled using the FPS algorithm and our representation of choice, the interpolated bounding box keypoints (IBB). We use the OpenCV implementation of the RANSAC-based EPnP algorithm with the same parameters to recover 6D object pose from the predicted keypoints. Since EPnP does not contain any learnable components, this experiment serves the goal of evaluating the ability of the YOLOPose model to estimate different keypoint representations in isolation. YOLOPose is trained using only the ℓ_1 loss in the case of BB and FPS representations, whereas in the case of IBB representation, ℓ_1 is combined with the cross-ratio loss described in Section 2.4. In our experiments presented in Table 3, when used in conjunction with the EPnP solver, the FPS keypoints performed worse than all other representations. In contrast, the IBB keypoints representation yields the best performance. We conjecture that the additional cross-ratio loss employed helps our model in learning the IBB keypoint projections better.

4.2 Effectiveness of RotEst

After deciding on the keypoint representation, we compare the performance of the learnable feed-forward rotation and translation estimators against the analytical EPnP algorithm. Based on the observation that the rotation and translation components impacted by different factors [16], we decide to estimate rotation and translation separately. As shown in Table 3, using only the rotation from the EPnP result and directly regressing the translation improved the accuracy significantly. In general, RotEst performs slightly better than using EPnP orientation and direct translation estimation. Furthermore, the RotEst module and the translation estimators are straightforward MLPs and, thus, do not add much execution time overhead. This enables YOLOPose to perform inference in real-time.

Table 3. Ablation study on YCB-V. We present the comparison results of different keypoint representations and the effectiveness of RotEst. The top section of the table corresponds to different keypoint representations in combination with the standard EPnP algorithm, and the bottom section corresponds to the effectiveness of the learnable RotEst module using IBB keypoints.

Method	ADD(-S)	AUC of ADD(-S)
FPS + EPnP	31.4	56.9
handpicked + EPnP	31.5	55.7
IBB + EPnP	**56.0**	**74.7**
IBB + EPnP for R; head for t	63.9	82.3
IBB + RotEst for R and head for t	**65.0**	**82.6**

5 Discussion & Conclusion

We presented YOLOPose, a Transformer-based single-stage multi-object pose estimation method using keypoint regression. Our model jointly estimates bounding boxes, class labels, translation vectors, and pixel coordinates of 3D keypoints for all objects in the given input image. Employing the learnable RotEst module to estimate the object orientation from the predicted keypoints coordinate enables the model to be end-to-end differentiable. We evaluated our model on the widely-used YCB-Video dataset and reported results comparable to the state-of-the-art approaches while being real-time capable. In the future, we plan to extend our model to video sequences and exploit temporal consistency to improve the pose estimation accuracy further.

6 Acknowledgment

This work has been funded by the German Ministry of Education and Research (BMBF), grant no. 01IS21080, project “Learn2Grasp: Learning Human-like Interactive Grasping based on Visual and Haptic Feedback”.

Bibliography

[1] Amini, A., Periyasamy, A.S., Behnke, S.: T6D-Direct: Transformers for multi-object 6D object pose estimation. In: DAGM German Conference on Pattern Recognition (GCPR) (2021)

[2] Carion, N., Massa, F., Synnaeve, G., Usunier, N., Kirillov, A., Zagoruyko, S.: End-to-end object detection with transformers. In: European Conference on Computer Vision (ECCV), pp. 213–229 (2020)

[3] Chen, B., Parra, A., Cao, J., Li, N., Chin, T.J.: End-to-end learnable geometric vision by backpropagating pnp optimization. In: IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 8100–8109 (2020)

[4] Cohen, N., Shashua, A.: Inductive bias of deep convolutional networks through pooling geometry. In: International Conference on Learning Representations, ICLR 2017, Toulon, France, OpenReview.net (2017)

[5] Gao, X., Hou, X., Tang, J., Cheng, H.: Complete solution classification for the perspective-three-point problem. IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI) **25**, 930–943 (2003)

[6] Hartley, R., Zisserman, A.: Multiple View Geometry in Computer Vision. Cambridge University Press, 2 edn. (2004), doi:10.1017/CBO9780511811685

[7] Hodaň, T., Sundermeyer, M., Drost, B., Labbé, Y., Brachmann, E., Michel, F., Rother, C., Matas, J.: BOP challenge 2020 on 6D object localization. In: European Conference on Computer Vision (ECCV), pp. 577–594 (2020)

[8] Hu, Y., Fua, P., Wang, W., Salzmann, M.: Single-stage 6D object pose estimation. In: IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 2930–2939 (2020)

[9] Hu, Y., Hugonot, J., Fua, P., Salzmann, M.: Segmentation-driven 6D object pose estimation. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 3385–3394 (2019)

[10] Kuhn, H.W.: The hungarian method for the assignment problem. Naval Research Logistics Quarterly **2**(1-2), 83–97 (1955)

[11] Labbe, Y., Carpentier, J., Aubry, M., Sivic, J.: CosyPose: Consistent multi-view multi-object 6D pose estimation. In: European Conference on Computer Vision (ECCV) (2020)

[12] LeCun, Y., Bengio, Y., et al.: Convolutional networks for images, speech, and time series. The handbook of brain theory and neural networks **3361**(10), 1995 (1995)

[13] Lepetit, V., Moreno-Noguer, F., Fua, P.: EPnP: An accurate O(n) solution to the PnP problem. International Journal of Computer Vision(IJCV) **81**(2), 155 (2009)

[14] Li, S., Yan, Z., Li, H., Cheng, K.T.: Exploring intermediate representation for monocular vehicle pose estimation. In: IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 1873–1883 (2021)

[15] Li, Y., Wang, G., Ji, X., Xiang, Y., Fox, D.: DeepIM: Deep iterative matching for 6D pose estimation. In: European Conference on Computer Vision (ECCV), pp. 683–698 (2018)

[16] Li, Z., Wang, G., Ji, X.: CDPN: Coordinates-based disentangled pose network for real-time RGB-based 6-DoF object pose estimation. In: Proceedings of the IEEE International Conference on Computer Vision (ICCV), pp. 7678–7687 (2019)

[17] Lin, T.Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., Zitnick, C.L.: Microsoft COCO: Common objects in context. In: European conference on computer vision (ECCV), pp. 740–755 (2014)

[18] Loshchilov, I., Hutter, F.: Decoupled weight decay regularization. In: International Conference on Learning Representations (ICLR) (2017)

[19] Manhardt, F., Kehl, W., Navab, N., Tombari, F.: Deep model-based 6D pose refinement in RGB. In: European Conference on Computer Vision (ECCV), pp. 800–815 (2018)

[20] Oberweger, M., Rad, M., Lepetit, V.: Making deep heatmaps robust to partial occlusions for 3D object pose estimation. In: European Conference on Computer Vision (ECCV) (2018)

[21] Peng, S., Liu, Y., Huang, Q., Zhou, X., Bao, H.: PVNet: Pixel-wise voting network for 6DOF pose estimation. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 4561–4570 (2019)

[22] Periyasamy, A.S., Schwarz, M., Behnke, S.: Robust 6D object pose estimation in cluttered scenes using semantic segmentation and pose regression networks. In: International Conference on Intelligent Robots and Systems (IROS) (2018), doi:10.1109/IROS.2018.8594406

[23] Periyasamy, A.S., Schwarz, M., Behnke, S.: Refining 6D object pose predictions using abstract render-and-compare. In: IEEE-RAS International Conference on Humanoid Robots (Humanoids), pp. 739–746 (2019)

[24] 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: IEEE International Conference on Computer Vision (ICCV), pp. 3828–3836 (2017)

[25] Rezatofighi, H., Tsoi, N., Gwak, J., Sadeghian, A., Reid, I., Savarese, S.: Generalized intersection over union: A metric and a loss for bounding box regression. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 658–666 (2019)

[26] Shao, J., Jiang, Y., Wang, G., Li, Z., Ji, X.: PFRL: Pose-Free reinforcement learning for 6D pose estimation. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2020)

[27] Stewart, R., Andriluka, M., Ng, A.Y.: End-to-end people detection in crowded scenes. In: IEEE conference on computer vision and pattern recognition (CVPR), pp. 2325–2333 (2016)

[28] Tekin, B., Sinha, S.N., Fua, P.: Real-time seamless single shot 6D object pose prediction. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2018)

[29] Wang, G., Manhardt, F., Tombari, F., Ji, X.: GDR-Net: Geometry-guided direct regression network for monocular 6D object pose estimation. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2021)

[30] Xiang, Y., Schmidt, T., Narayanan, V., Fox, D.: PoseCNN: A convolutional neural network for 6D object pose estimation in cluttered scenes. arXiv:1711.00199 (2017)

[31] Zhou, Y., Barnes, C., Lu, J., Yang, J., Li, H.: On the continuity of rotation representations in neural networks. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 5745–5753 (2019)