CRT-6D: Fast 6D Object Pose Estimation with Cascaded Refinement Transformers

Pedro Castro
Imperial College London
p.castro18@imperial.ac.uk

Tae-Kyun Kim
Imperial College London, KAIST
tk.kim@imperial.ac.uk

Abstract

*Learning based 6D object pose estimation methods rely on computing large intermediate pose representations and/or iteratively refining an initial estimation with a slow render-compare pipeline. This paper introduces a novel method we call **C**ascaded Pose **R**efinement **T**ransformers, or **CRT-6D**. We replace the commonly used dense intermediate representation with a sparse set of features sampled from the feature pyramid we call **OSKFs**(**O**bject **S**urface **K**eypoint **F**eatures) where each element corresponds to an object keypoint. We employ lightweight deformable transformers and chain them together to iteratively refine proposed poses over the sampled OSKFs. We achieve inference runtimes 2× faster than the closest real-time state of the art methods while supporting up to 21 objects on a single model. We demonstrate the effectiveness of CRT-6D by performing extensive experiments on the LM-O and YCB-V datasets. Compared to real-time methods, we achieve state of the art on LM-O and YCB-V, falling slightly behind methods with inference runtimes one order of magnitude higher. The source code is available at:* `https://github.com/PedroCastro/CRT-6D`

1. Introduction

Estimating the 6D pose of objects given an RGB image remains a challenging computer vision task yet indispensable in many real world applications from autonomous vehicle perception, robotics as well as augmented reality. This task entails the retrieval of a target object's 3D rotation and translation, relative to a camera, by overcoming difficult issues such as occlusion, illumination and symmetries. Depth information can used to great effect when available [21, 28], while monocular methods tend to underperform due to lack of information.

Recent methods utilizing Convolutional Neural Networks (CNNs) have surpassed prior classical approaches and are at the core of most recent state-of-the-art 6D object pose estimators [41, 6, 34, 39, 35, 47, 9, 24, 40, 32, 3]. The computation pipeline of these methods can be roughly defined by 3 steps: 1.) The object is detected in the image (this usually done using an off-the-shelf object detector [37, 36]) ; 2.) Features are extracted from a cropped image, around the 2D area containing the object, using an established CNN pre-trained architecture [13, 42]; 3.) These

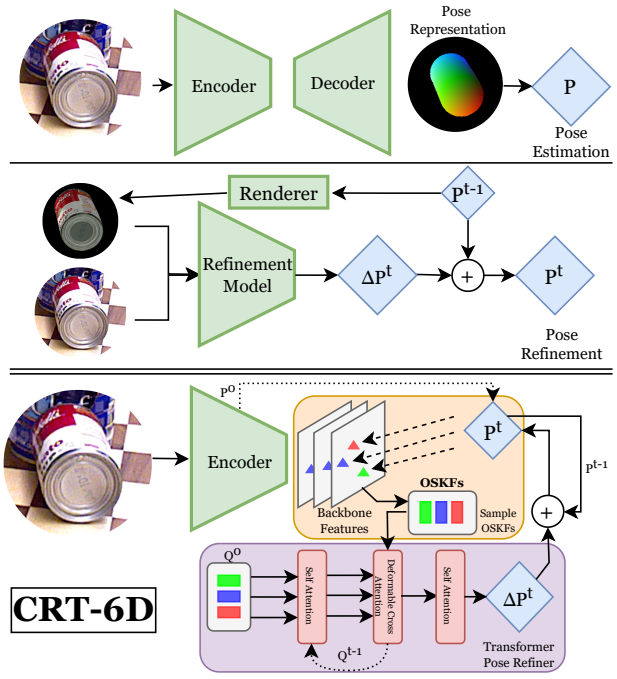

Figure 1: **Illustrative diagram of CRT-6D.** CRT-6D removes the decoder and pose representation from pose estimation methods, and the renderer and refinement model from the standard refinement pipelines. Instead CRT-6D replaces them with a deformable attention based refinement module achieving pose estimation and refinement within the same model. Each refinement iteration takes less than 3ms making **CRT-6D 2× faster than prior real time methods and simultaneously achieving better accuracy.**

features are transformed into an intermediate representation [34, 32, 47, 35, 15] which are then used to extract pose (using PnP [25] or other variations [40, 15]) or pose is extracted directly [7, 47, 9, 24]. By assuming that the necessary information to extract the pose is performed by step 2.), the pose extraction step is the key to a fast and accurate estimation. Prior art has proposed several intermediate representations, e.g. NOCS, keypoint heatmaps [48, 35, 34, 51]. These representations cover the full input crop area thereby computing it at every pixel on a significant spatial dimension, regardless of the area of the image occupied by the object, resulting in a large amount of unnecessary and expensive computations. Moreover, some require an additional slow RANSAC PnP step. Others methods propose to directly learn the PnP operation [47, 7, 9], and while they are shown to be faster and more precise, introduce more complexity into the model without removing the information-less regions from the computational pipeline.

On top of these, an application might choose to refine the predicted erroneous pose. The most commonly used methods rely on a costly *render-compare* iterative process [28, 51, 24, 46], making them unsuitable for real-time applications. *Ad-hoc* refinement methods require large models, designed and trained only for refinement and leaving the initial pose estimation as an exercise for other methods [24, 28, 51]. More recently, specifically designed approaches perform a trade-off between runtime and initialization: RePose while fast requires a great initialization [20] and SurfEmb [12] is very precise and robust to occlusions and symmetries but is extremely slow at inference time .

In this paper, we introduce a novel method that removes redundant computations around areas where the object is not present, while oversampling the image regions where it is. We achieve this by using a simple yet effective intermediate offset representation: *Object Surface Keypoint Features* (OSKFs). Given an initial coarse pose, we project predetermined object surface keypoints into the image plane. We generate OSKFs by sampling the extracted feature pyramid at each keypoint current 2D location. Given that the intial pose is not guaranteed to be precise, we use deformable attention to guide our sampling around the original 2D location, overcoming possible errors in the coarse pose. Therefore, we propose *OSKF-PoseTransformers* (OSKF-PT), a transformer module with deformable attention mechanisms [53], where self-attention and cross-attention operations are performed over the OSKFs set, outputting an improved pose. Since OSKFs are an inexpensive representation in terms of computation, we chain together multiple OSKF-PT in a novel Cascaded Pose Refinement (CPF) module to iteratively refine the pose in a cascaded fashion, which can be trained end-to-end.

In summary, this paper's contributions are:

- We propose Object Surface Keypoint Features (OSKF), a lightweight intermediate 6d pose offset representation, which is significantly less noisy, ignores unusable information from feature maps resulting in a more accurate pose estimation when compared to prior art and is considerably cheaper to generate than intermediate pose representations.
- We propose OSFK-PoseTransformer (OSFK-PT), a module that utilizes a chain of self-attention and deformable-attention layers to iteratively update an initial pose guess. Due to the lightweight nature of OSKFs, our refinement is faster than any prior refinement method, taking less than 3ms per iteration.
- We introduce CRT-6D, a fast end-to-end 6d pose estimation model, that leverages a cascaded iterative refinement over a chain of OSFK-PTs to achieve state of the art accuracy for real time 6D pose estimators on two challenging datasets, with its inference time being 100% faster than the fastest prior methods.

2. Literature Review

Keypoint Detection. Object pose estimation can be seen as the inverse of camera pose estimation. One can extract 6D pose by solving the PnP problem which means we can detect the pixel position of keypoints, creating the necessary 2D-3D correspondence set. Early works started by choosing the 3D bounding box of the objects as keypoints [43, 35, 30]. However, the projected 3D bounding box keypoints usually lie outside the silhouette of the objects , which potentially reduces the local information extraction. This shortfall was noticed by PVNet [34], which suggests the use of the surface region to find suitable keypoints.

Dense Object Coordinate Estimation. Instead of preselecting a few keypoints, NOCS [48] was proposed where for every pixel in the silhouette of the object is used to estimate the coordinate of the surface of the object (in normalized space) projected at that pixel. In other words, every point in the surface would become a keypoint and could be used for the 2D-3D correspondence set to solve PnP. Inspired by NOCS, Pix2Pose [32] proposed the use of a GAN to solve issues with occlusion. DPOD [51, 15, 47, 40] suggested using UV maps and object regions instead of a 3D coordinate system, ensuring that every point estimated lied within the object's surface. Each of these methods has an increasingly more complex model, and while performance has been improved by each method, runtime has been overlooked.

Direct Pose Estimation. Posenet[22] proposed learning quaternions to predict rotation on camera pose estimation tasks. In 6D object pose estimation field, PoseCNN [50] used Lie algebra instead. SSD-6D [21] discretized the viewpoint space and learned to classify it, while using the mask to regress the distance to the camera. These methods are

more susceptible to noise and occlusion due to their holistic approach. Moreover, these usually require an extra step to solve the ambiguity caused by egocentric orientations. Some methods learn a mapping from an intermediate representation to emulate a PnP solver, making them differentiable w.r.t. the final pose [47, 9, 17, 7]. This step can be used to reduce the need for symmetry hacks on dense methods.

Pose Estimation Refinement If depth is available, then Iterative Closest Point (ICP) is the most commonly used algorithm [2]. The ICP algorithm finds the correspondence between points by iteratively refining the pose that takes to align them. However, it is heavily dependent on initial pose and might converge in a local minimum. Recent learned approaches mostly rely a *render-compare* pipeline, with slight variations among these methods [28, 51, 24]. More recently, Repose [20] introduced a fast iterative refinement algorithm however it requires a great initialization.

Transformers in Pose Estimation. Given the rising effectiveness of transformers in computer vision tasks, there have been attempts to use transformers to improve human [19, 27], hand [18] and object [31, 1, 54, 38, 11] pose estimation. For object pose, such approaches are aimed at improving results [31, 1], category level estimation [54, 38] or hand-object interaction [11]. However, these improvements come at the cost of runtime, making them unsuitable for real-time applications. Our novel approach, not only improves results, but also decreases the runtime when compared to prior object pose estimation methods.

3. Methodology

In this section, we detail each step of **CRT-6D**, our novel 6D pose estimation method. Given an image $\mathcal{I}$, the goal of CRT-6D is to predict the $P_i = [R_i|t_i]$, the 6D pose of the objects targeted by the camera, where i refers to the i^{th} object in the set of N objects $\mathcal{O} = \{\mathcal{O}_i \mid i = 0, , N-1\}$ present in the image. We follow the setup of other methods [47, 9, 23] and disentangle pose from object detection which means we use an off-the-shelf detector to crop out regions of the image where objects are present. These regions are then independently processed and fed to CRT-6D for pose estimation.

3.1. Coarse Pose Estimation

We use a Resnet34 [13] as the backbone for CRT-6D. While other backbones could have been used, Resnets present fairer comparisons with prior art as they are used by most 6D pose approaches[47, 9, 40, 20, 29]. We use the backbone to build a multi-scale feature pyramid $\mathcal{F} = \{\mathcal{F}_l \mid l = 1, , L\}$ of sizes $[s/4, s/8, s/16, s/32]$, with $L = 4$s.

CRT-6D starts by estimating a coarse pose $P^0_{6D} = [R^0|t^0]$ using a simple MLP: $FC_\theta(F_3) = R^0_{6D}, \tilde{t}^0$, where

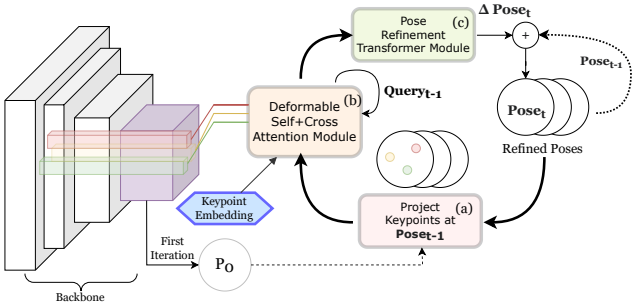

Figure 2: **Illustration of CRT-6D.** We start by generating our feature pyramid. Using the last feature level, we generate a pose estimation P^0, which will serve as the initial reference for the refinement module. **(a)** Using P^0, we project a set of surface keypoints $\mathcal{S}$ into the image plane. **(b)** Our deformable attention mechanism uses the location of the set of projected keypoints to sample the feature pyramid. **(c)** We then perform self-attention over the sampled feature set and estimates an offset pose ΔP such that $P^t = \hat{P}^{t-1} + \Delta P^t$, with t being the refinement step.

θ are learnable parameters. To recover the rotation matrix $R = [R_1, R_2, R_3]$, we use the 6D rotation representation $R_{6D} = [r_1, r_2]$ introduced in [52] and used in prior methods [47, 9, 24] to great success:

$$\begin{cases} R_1 = r_1 \\ R_3 = R_1 \times r_2 \\ R_2 = R_3 \times R_2 \end{cases}, \tag{1}$$

where $r1$ and $r2$ are unit vectors. Due to the projective function applied by the camera, the appearance of an object is affected not only by its orientation but also by its position. Since we are working with a cropped image and camera intrinsics K are known, predicting P becomes a one-to-many function, where the same appearance might correspond to different egocentric orientations as pointed out by 3D-RCNN [23]. A common strategy to solve this problem, which we also adopt, is to estimate the allocentric orientation and apply a transformation to recover the egocentric orientation at inference time.

Wang *et al*. [47] showed that choosing an adequate translation representation has an effect on the performance of the method. While our goal is the global translation $t = [t_x, t_y, t_z]$, this information cannot be directly recovered by CRT-6D due to the cropping step. Therefore we use $t' = [O_x, O_y, t_z]$ which can be used to recover t via back-projection. CRT-6D adopts the scale-invariant representation $\tilde{t} = [\gamma_x, \gamma_y, \gamma_z]$ [47, 9]:

$$\begin{cases} \gamma_x = (O_x - c_x)/s_{bbox} \\ \gamma_y = (O_y - c_y)/s_{bbox} \\ \gamma_z = t_z/r_{bbox} \end{cases} \tag{2}$$

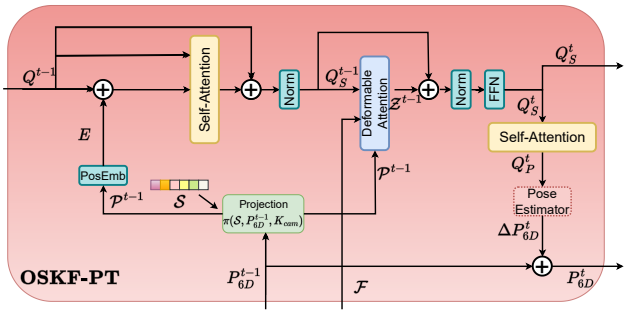

Figure 3: **Illustration of OSKF-PT.** OSKF-PT receives as an input the previous Q^{t-1}, P_{6D}^{t-1} and $\mathcal{F}$, the last one being the same on every iteration. It outputs a new refined pose P_{6D}^t and Q^t, an initial query to guide the next iteration.

where the scale of the crop bounding box $s_{bbox} = max(w_{bbox}, h_{bbox})$ and ratio $r = s_{bbox}/s$, with s referring to the original size of the image.

3.2. Object Surface Keypoint Features - OSKF

Refinement methods usually rely on a prior independent powerful pose estimator [24, 20, 28] slowing both inference and training. In contrast, CRT-6D is designed to estimate and refine by reusing the multi-scale feature pyramid. We replace the rendering step used by pose refiners [24, 28, 20] and instead of using a pose representation, we generate a pose offset representation $\mathcal{Z}$, a compact set of backbone features sampled at 2D keypoint locations $\mathcal{P}^t = \{\pi(\mathcal{S}_k^t, P_{6D}^t, K_{cam}) \mid k = 1, ..., K\}$, where π is the projection function, K_{cam} are the known camera intrinsic parameters, P_{6D}^t is the reference pose, $\mathcal{S}$ is a set of predefined keypoints, with t denotes the iteration step. Since the input image I suffers no perturbations from the pose estimation step to the refinement step, or even between refinement iterations, we recycle these features instead of recomputing them at every iteration, in contrast to prior methods [51, 28, 24, 20].

We propose OSKFs $\mathcal{Z} = \{\mathcal{Z}_k \mid k = 1, ..., K\}$, a lightweight offset pose representation generated by sampling the feature pyramid at all spatial scales:

$$\mathcal{Z} = \{\mathcal{F}_l(\mathcal{P}_k) \mid k = 1, ..., K,\ l = 1, ..., L\} \tag{3}$$

where l denotes the l^{th} element of the feature pyramid.

The set of object keypoints $\mathcal{S} = \{\mathcal{S}_k \mid k = 1, ..., K\}$ chosen for OSKFs are generated using the farthest point sampling algorithm [34], where K is a hyperparameter of the number of used keypoints. Each of these features represent the local information around $\mathcal{P}$, with coarse information at higher feature levels and finer on lower. The following modules will learn to recover the pose offset embedded in $\mathcal{Z}$.

3.3. OSKF Pose Transformer (OSKF-PT)

The untopologized set of OSKFs $\mathcal{Z}$ presents a use case for transformers which are able to process data without introducing implicit spatial bias [45] allowing the attention mechanisms to learn spatial and structural relationships. CRT-6D is designed to interpret $\mathcal{Z}$ and learn to output the offset pose ΔP such that:

$$P_{6D}^t = \hat{P}_{6D}^{t-1} + \Delta P_{6D}^t, \tag{4}$$

at the t refinement step.

However, the estimated pose is not guaranteed to be correct nor the keypoints unoccluded. For this reason, the optimal information for a keypoint might not lie close to the 2D position of that object. Inspired by Deformable-DETR [53] we introduce OFKF-PT where we make use of deformable attention mechanisms to learn to sample information around the reference keypoint position. A diagram of the pose transformer module OFKF-PT is showed in Fig 3.

OFKF-PT are composed by a self-attention layer, a deformable self attention layer and another attention layer for pose estimation, in this order. We start by performing self-attention over a query matrix Q, where each row corresponds to a keypoint in the set $\mathcal{S}$. The multi-head attention operation takes $\hat{Q}$, $\hat{Q}$ and Q as query, key and value, where $\hat{Q} = Q + E$. $E_k = PosEmb(\mathcal{P}_k)$ are the positional embeddings for keypoint k and *PosEmb* is the high frequency cosine embedding for position parameterization [45]. Following [53], we then apply residual addition and layer normalization. We denote the output of this operation as Q_S.

The multi-scale deformable attended OSKFs are formally defined by:

$$\tilde{\mathcal{Z}}_{kl} = \sum_{j=0}^{J} A_{ljk} W \mathcal{F}_l(\mathcal{P}_k + \Delta\mathcal{P}_{ljk}) \tag{5}$$

where $\Delta\mathcal{P}_{ljs}$ refers to the predicted sampling offset for the j^{th} deformable position, spatial level l and object keypoint k. J refers to the number of sampling points used for deformation. $\Delta\mathcal{P}_l js$ and A_{ljs} are computed by a simple linear transformations of Q_S.

We apply a self-attention operation over $\tilde{\mathcal{Z}}$ in order to capture long distance information which we denote as Q_P. We chose to use a global pooling operation over Q_P instead of using a CLS-Token (which is shown to make no difference in practice [10]), followed by a small MLP to output $[\Delta R, \Delta\gamma_x, \Delta\gamma_y, \Delta\gamma_z]^t$. We update the reference pose P^t:

$$\begin{cases} R^t = \Delta R^t\ R^{t-1} \\ \gamma_{\{x,y\}}^t = \gamma_{\{x,y\}}^{t-1} + \Delta\gamma_{\{x,y\}}^t \\ \gamma_z^t = \gamma_z^{t-1} \cdot (1 + tanh(\Delta\gamma_z^t)) \end{cases}. \tag{6}$$

We found that using a *tanh* parameterization over $\Delta\gamma_z$ performed better than linear [24] or exponential [28].

For our experiments, we chain together 3 OFKF-PT. The output of the deformable attention operation is reused on the next iteration, as the input matrix becomes $Q^t = \tilde{\mathcal{Z}}^{t-1}$. The initial query Q^0 is a learnable embedding.

3.4. Objective Functions

The main goal of CRT-6D is to process an image and produce the 6D pose of the target object. To perform this operation, it goes through an iterative refinement process. At each iteration, we compute a valid 6d pose and reuse it on the next iteration. Due to the simplified objective of CRT-6D, which does not output an intermediate pose representation, our overall objective $\mathcal{L}$ is guided by the pose estimation error and can be defined as:

$$\mathcal{L} = \lambda \mathcal{L}^0 + (1 - \lambda) \sum_{i=1}^{N} \mathcal{L}^i \tag{7}$$

with i indicates the refinement iteration, $N = 3$ is fixed for all experiments and $\mathcal{L}^0$ the loss for the coarse estimation. The pose loss $\mathcal{L}$ is disentangled and separated into rotation and position loss:

$$\mathcal{L} = \alpha \mathcal{L}_R + \mathcal{L}_{pos}. \tag{8}$$

Recalling the pose parameterization described in Sec. 3.1, the loss functions are defined as:

$$\begin{cases} \mathcal{L}_R = \underset{x \in \mathcal{P}}{avg} ||Rx - \hat{R}x||_1 \\ \mathcal{L}_{pos} = ||\gamma_x - \hat{\gamma_x}, \gamma_y - \hat{\gamma_y}, \gamma_z - \hat{\gamma_z}||_1 \end{cases}, \tag{9}$$

where $\hat{\cdot}$ refers to the groundtruth data. When the target object is symmetric, a variation of $\mathcal{L}_R$ is used [48], while $\mathcal{L}_{pos}$ is invariant to symmetries. Note that while our refinement module outputs an offset pose ΔP, the transformation in Eq. 4 is differentiable which means CRT-6D can be directly optimized through the set of predicted poses $P_{6D} = \{P_{6D}^t | t = 0, ..., N\}$

4. Experiments

We conducted experiments on two benchmark dataset LM-O[14] and YCB-V[4] where we present strong evidence of our method's potential. We also show through ablation studies our key contributions, including the improvements stemming from the use of iterative refinement and the high accuracy it achieves with an impressively low inference time. Results for all BOP datasets [16] are available on the challenge website.

4.1. Experimental Setup

Datasets Setup. The commonly used Linemod dataset (LM) [14] has become sasturated with most recent methods achieving over 95% accuracy [24, 47, 9]. For this reason we adopt our experiments on the more challenging Linemod Occlusion (LM-O), a subset of 1214 LM images, where $\sim$ 8 objects are annotated on every image. For LM-O, in accordance with prior art [47, 9, 40], we make use of the available LM real images, where $\sim$ 1200 images are available per object. We also present experiments on YCB-V, a larger dataset with 21 target objects, some with very challenging symmetries. For this dataset over 100k real images are available for training. However, the dataset is generated through video resulting in similar frames where the objects are seldom fully visible.

On top of real images, we also make use of synthetic data. For a fair comparison, we employ the readily available PBR splits [8], available for both LM-O and YCB-V, a dataset of photo and physically realistic synthetic images containing the target models with challenging poses and under heavy occlusion. We also perform common on-the-fly image augmentations such as color jittering, blur and noise as well as more complex operations such as in-plane image rotations and background removal. For experiments under the BOP [16] setup, LM-O methods are trained only with PBR synthetic data. We also implement the Dynamic Zoom-In (DZI) [29] in order to be robust to detection errors. During training we apply uniform perturbations to the center and scale of the bounding box. At test time, we found that increasing the detection bounding boxes by 20%, to ensure the object is fully visible, yielded the best results.

Implementation details. We implement CRT-6D using PyTorch [33]. We use 8 heads and 4 points for deformable attentions as suggested in Deformable-DETR [53]. The model is trained in an end-to-end fashion, including the cascaded refinement step. All ablation experiments are optimized with same number of training iterations. For LM-O experiments, CRT-6D is optimized for 250k iterations with batch size of 32, with PBR images composing 50% of the batch, or 100% if under BOP standards [16]. YCB-V is trained for 350k, with the same PBR ratio and batch size. We use the Ranger optimizer [49] starting at a learning rate 10^{-4} with a cosine annealing schedule starting at 85% of training. Unlike similar methods [5, 53, 26], we found in early experiments that choosing a lower learning rate for the backbone weights was not ideal. For the first 20% of the iterations we set $\lambda = 0$ because P^0 starts with very poor pose estimations, which does not allow CRT-6D to learn. For the rest of training $\lambda = \frac{N-1}{N}$ where N is the number of pose refiners used. We set $\alpha = 3$ for all experiments.

During testing, we use the same detections per dataset for all experiments. For LM-O we use the publicly available 2D detections from Faster-RCNN utilized by [47, 9, 40]. For YCB-V, we also use publicly available detections from FCOS[44] trained by [29]. For BOP results, we the standardized detections provided by the challenge.

Type	Pose Estimation					Refinement		Hybrid
N.S.O.	1	8	1	8	1	1	8	8
Method	PVNet [34]	GDR [47]	GDR [47]	SO-Pose [9]	ZebraPose [40]	RePose [20]	DeepIM [28]	**CRT-6D**
Ape	15.8	44.9	46.8	48.4	55.2	31.1	**59.2**	53.4
Can	63.3	79.7	90.8	85.8	**94.9**	80.0	63.5	92.0
Cat	16.7	30.6	40.5	32.7	**56.6**	25.6	26.2	42.0
Driller	65.7	67.8	82.6	77.4	**94.7**	73.1	55.6	81.4
Duck	25.2	40.0	46.9	48.9	**60.9**	43.0	52.4	44.9
Eggbox*	50.2	49.8	54.2	52.4	**64.7**	51.7	63.0	62.7
Glue*	49.6	73.7	75.8	78.3	**84.5**	54.3	71.7	80.2
Holepuncher	36.1	62.7	60.1	75.3	**83.2**	53.6	52.5	74.3
Average	40.8	56.1	62.2	62.3	**74.3**	51.6	55.5	66.3

Table 1: **Comparison study on LM-O.** We present the results for ADD(-S) metric and compare them to state of the art. We are outperformed only by ZebraPose [40], a method with an inference time of $\sim$ 191ms for a single object, while **CRT-6D** estimates the pose for all objects in a single LM-O image ($\sim$ 8 objects) in 36ms. Best results are bolded while second best are underlined. *N.S.O.* refers to the number of objects supported by a model and * denotes symmetric objects.

Evaluation Metrics. For a fair comparison, we emulate the same practice as prior methods. For LM-O, we present our results over the ADD(-S) metric [35, 16]. Under the ADD(-S) metric, a pose is considered correct if the distance of the transformed model points from the groundtruth points is below $d \cdot 10\%$, d being the object's diameter. For symmetric objects, we take the distance from each transformed point to the closest groundtruth and apply the same threshold [16, 14]. When experimenting on the YCB-V dataset, we also take the Area Under the Curve (AUC) of the ADD metric, with a maximum threshold distance of 10cm [50]. For more detailed experiments we also measure accuracy in terms of n°, n cm which considers a pose valid if both the rotation and translation fall under the defined thresholds. Under BOP standards, we present the average recalls used by the challenge: AR_{vsd}, AR_{mssd} and AR_{mspd}, along with their mean. We refer the reader to the BOP challenge [16] for more information about these metrics. The inference time measurements, for both CRT-6D and prior methods, were all made using publicly available code on a GTX1080ti. For simplicity, we ignore detection time as most methods are evaluated using the same detections.

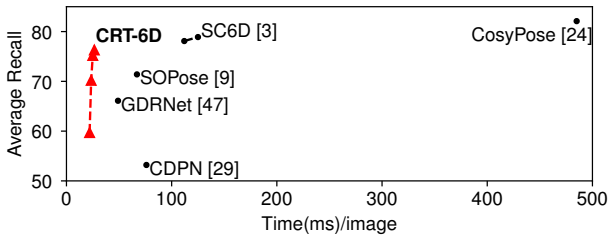

Figure 4: **Visualization of the inference time difference between state of the art methods.** CRT-6D is faster, even after 3 refinement steps, than prior approaches and has very competitive results compared to methods 10$\times$ slower. We measure the average time it takes to estimate the pose for all objects in an image ($\sim$4.75 per image on YCB-V). ZebraPose [40] and SurfEmb [12] are omitted as their results would lie outside the runtime range, with estimation taking over 250ms and 2000ms per crop, respectively.

4.2. Runtime comparison to state of the art.

DeepIM [28] is a landmark work on learned pose refinement. It is however very slow, inference time being 41ms per object per iteration (two are recommended), without taking into account the model used for initialization. More recently CosyPose [24] performs a two-in-one model, where two CNN are employed and fully trained, one performs an initial coarse estimation while the second one refines it. However, due to the massive size of the models used, their inference time is $\sim$ 100ms for a single object. Repose [20] proposed a faster refinement method at 18ms with 5 iterations however they require a good initialization (they use PVNet [34] which itself takes over 25ms) and it only support a single object per model.

In contrast, CRT-6D takes on average 26ms and 34ms for YCB-V and LM-O images (each with $\sim$ 4 and $\sim$ 6 object instances) for all objects on a single model, making our method one order of magnitude faster than other refinement methods. Recent state of the art real time methods, GDR-Net [47]and SO-Pose [9] take 2$\times$ and 3$\times$ longer, respectively. Moreover, CRT-6D not only is faster but more accurate than both these methods, on both LM-O and YCB-V. A more detail comparison to other methods w.r.t. inference time can be seen in Fig. 4.

4.3. Accuracy comparison to the State of the Art

LM-O results. We present the results under the ADD(-S) metric for LM-O on Tab. 1. We present competitive results,

Method	LM-O				YCB-V				Mean AR
	AR_{VSD}	AR_{MSSD}	AR_{MSPD}	**AR**	AR_{VSD}	AR_{MSSD}	AR_{MSPD}	**AR**	
EPOS [15]	0.389	0.501	0.750	0.547	0.626	0.677	0.783	0.695	0.621
GDR-Net [47]	-	-	-	-	0.584	0.674	0.726	0.661	-
SO-Pose [9]	0.442	0.581	0.817	0.613	0.652	0.731	0.763	0.715	0.664
SurfEmb [12]	-	-	-	0.656	-	-	-	0.718	0.687
CosyPose [24]	0.480	0.606	0.812	0.633	**0.772**	**0.842**	**0.850**	**0.821**	**0.727**
SC6D [3]	-	-	-	-	0.695	0.796	0.804	0.765	-
CRT-6D	**0.504**	**0.640**	**0.837**	**0.660**	0.707	0.776	0.774	0.752	0.706

Table 2: **LM-O and YCB-V datasets under the BOP standards [16].** We present the results for all metrics used on the challenge with *Mean AR* referring to average *AR* over both datasets. Best results are **bolded** while second best are underlined.

Figure 5: **Deformable attention sampling locations.** The red circle indicates the reference point while the white circles indicate deformed sample positions. On the last column, we show attention sampling for the same object and keypoint, with the first row having the keypoint visible whereas on the bottom two the keypoint is self or externally occluded. When the keypoint is occluded, attention learns to sample clues elsewhere on the object silhouette.

with the second best overall accuracy behind ZebraPose [40]. Compared with real-time methods, CRT-6D achieves a 6.4% improvement over SO-Pose [9], the former state of the art. Under the BOP challenge rules, we achieve state of the art performance, reaching an Average Recall of 67.2%, beating all other RGB based methods, regardless of their inference time. We achieve this result because our method is extremely robust to occlusion due to our use of deformable attention and the fact that it can *attent* to regions far from the reference position when the the keypoint is occluded (see Fig. 5).

YCB-V results. On YCB-V, we present extremely competitive results by achieving 72.1% and and a state of the art 87.5 ADD-(S) and AUC of ADD(-S), respectively. Compared to closest real-time methods, we outperform GDR-Net[47] and SO-Pose[9] by 46% and 27% on the ADD(-S) metric, 9% and 4% on AUC of ADD(-S). When compared to slower methods, our method is surpassed on the ADD(-S) only by ZebraPose [40], which is an order of magnitude slower than our method. We also present YCB-V results under the BOP standard, where we are only outperformed by a slower method.

4.4. Ablation studies

Ablation study on number of OSKFs On Fig. 6 we show that CRT-6D is not highly reliant in a high number of reference keypoints. While we find that 8 points does not offer sufficient accuracy for our standards, the difference between 16 and 128 keypoints is not significant. Regardless, we use $K = 64$ keypoints on all our experiments as it provides the best results for both LM-O and YCB-V.

Cascaded Pose Refinement. We proposed a method that iteratively refines the initial pose of an object in a cascaded

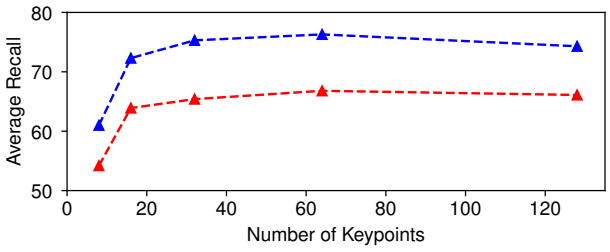

Figure 6: **Experiments over number of OSKFs used.** With this ablation, we show that CRT-6D is not reliant on a large number of keypoints. Although significant decreases in performance occur when using 8 keypoints, we see small differences for larger amounts, with the optimal number of keypoints being 64 for both datasets.

Dataset	LM-O		YCB-V		
Metric	ADD-(S)	AR	ADD-(S)	AUC	AR
0	40.5	59.8	50.6	81.0	59.7
1	61.4	69.0	63.2	84.9	70.2
2	64.9	71.1	70.8	87.1	75.2
3	66.3	71.5	72.1	87.5	76.3

Table 3: **Effects of the Cascaded Pose Refinement**. We present the improvement with each refinement step. We can observe the diminishing returns where the first refinement improvement is larger than all other iterations combined.

Initial Pose	ADD(-S)			2°2cm	5°5cm	Total
	0.02d	0.05d	0.1d			
GDR [47]	3.42	26.5	56.1	3.14	35.1	24.9
SO-Pose [9]	**4.55**	**31.4**	**62.3**	**3.39**	**39.0**	**28.1**
CRT-6D-Coarse	1.53	16.3	40.5	1.99	29.45	18.0
GDR* [47]	6.08	34.4	62.5	5.67	43.5	30.43
SO-POSE* [9]	6.22	34.8	63.4	5.38	44.4	30.84
CRT-6D	**6.23**	**36.2**	**66.3**	**5.48**	**45.4**	**31.9**

Table 4: **Initial Pose ablation study.** We show the experimental results when using our refinement on top of prior art. Results for initial poses are on the top 3 rows while bottom 3 rows are the refined poses. * indicates the use of CRT-6D refinement on the respective method's estimated pose.

fashion. On Tab. 3 we show the impact multiple iterations have on the accuracy. We find a decrease of marginal improvements when applying multiple refinements. The reduction in error on the first refinement iteration is larger than the consequent iterations combined.

Effect of coarse pose accuracy. In Table 4 we show the impact of the initial coarse pose on the OSKF-TP module. We show OSKF-TP can serve as an independent refiner module, such as DeepIM [28] or Repose [20], as it can receive pose estimations from other approaches and refine them to a state of the art level. Using 3 OSKF-TP modules, we improve GDR [47] and SO-Pose [9] results by 22% and 10% respectively. This experiment is done for the sake of completion: CRT-6D must generate the feature pyramid $\mathcal{F}$ to compute OSKFs, which is the most expensive operation in the pipeline, making the use of these initial poses redundant and unnecessary as we achieve better results with our coarse initialization.

4.5. Qualitative Results

Visualization of Deformable Attention. In Fig 5 we visualize the attention sampling points generated by the deformable attention operation. We only show sampling positions (white circles) for high attention weights $A > 0.25$ (see Eq. 5). When the keypoints is visible, the deformations occur near the keypoint projection location, whereas for occluded keypoints attention is scattered around the image. The transformer can guide its sampling to regions with clues to recover subtle pose differences.

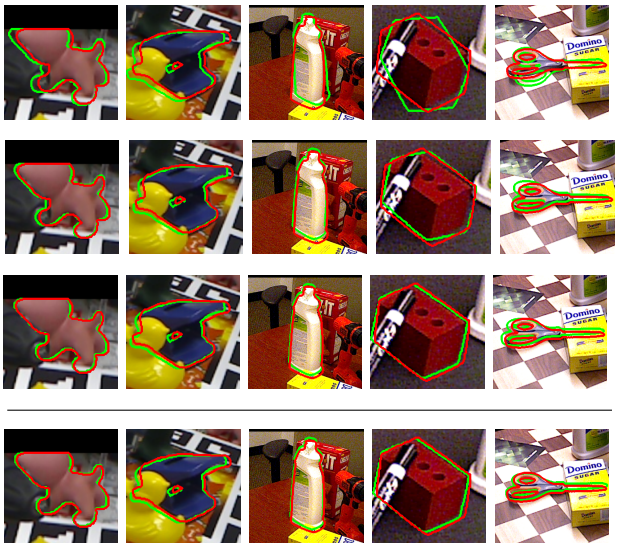

Figure 7: **Qualitative results on refinement steps.** We show the progress in the pose accuracy over the iterative refinements, with the first row being the initial coarse pose and last row the final results. Although the initial pose is very poor for evaluation standards it proves to suffice as a reference for refinement.

Refinement Qualitative Examples. On Fig.7 we present qualitative results showing the impact of each refinement step. We can see that for most objects the original pose is significantly inaccurate. Nonetheless, CRT-6D can recover an accurate pose after 3 iterations of the refinement module. The last column shows the results on the *scissors* object, the hardest object in YCB-V, CRT-6D was not able to recover the correct pose due to a poor initial pose.

5. Conclusion

We have proposed a novel approach to 6d object pose estimation CRT-6D based on iterative pose refinements. The input to our refinement modules is a set of feature vectors OSKFs, sampled from feature pyramid at the location of known keypoints, 2D projected using a coarse pose. OSKFs are the representation of the pose offset representation and are fed into an OSKF-PoseTransformer to extract the refined pose. We achieve state of the art on multiple datasets while being at least $2\times$ faster than similar methods.

For future directions, we are looking to extend CRT-6D into category level object pose estimation, where specific keypoints cannot be used.

AcknowledgementS. This work is in part sponsored by KAIA grant (22CTAP-C163793-02, MOLIT), NST grant (CRC 21011, MSIT), KOCCA grant (R2022020028, MCST) and the Samsung Display corporation.

References

[1] Arash Amini, Arul Selvam Periyasamy, and Sven Behnke. Yolopose: Transformer-based multi-object 6d pose estimation using keypoint regression, 2022.

[2] P.J. Besl and Neil D. McKay. A method for registration of 3-d shapes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14(2):239–256, 1992.

[3] Dingding Cai, Janne Heikkilä, and Esa Rahtu. Sc6d: Symmetry-agnostic and correspondence-free 6d object pose estimation. *arXiv preprint arXiv:2208.02129*, 2022.

[4] Berk Calli, Aaron Walsman, Arjun Singh, Siddhartha Srinivasa, Pieter Abbeel, and Aaron M Dollar. Benchmarking in manipulation research: The ycb object and model set and benchmarking protocols. In *IEEE International Conference on Advanced Robotics (ICAR)*, 2015.

[5] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *European conference on computer vision*, pages 213–229. Springer, 2020.

[6] Pedro Castro, Anil Armagan, and Tae-Kyun Kim. Accurate 6d object pose estimation by pose conditioned mesh reconstruction. In *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4147–4151, 2020.

[7] Bo Chen, Alvaro Parra, Jiewei Cao, Nan Li, and Tat-Jun Chin. End-to-end learnable geometric vision by backpropagating pnp optimization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020.

[8] Maximilian Denninger, Martin Sundermeyer, Dominik Winkelbauer, Youssef Zidan, Dmitry Olefir, Mohamad Elbadrawy, Ahsan Lodhi, and Harinandan Katam. Blenderproc, 2019.

[9] Yan Di, Fabian Manhardt, Gu Wang, Xiangyang Ji, Nassir Navab, and Federico Tombari. So-pose: Exploiting self-occlusion for direct 6d pose estimation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2021.

[10] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

[11] Shreyas Hampali, Sayan Deb Sarkar, Mahdi Rad, and Vincent Lepetit. Keypoint transformer: Solving joint identification in challenging hands and object interactions for accurate 3d pose estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11090–11100, 2022.

[12] Rasmus Laurvig Haugaard and Anders Glent Buch. Surfemb: Dense and continuous correspondence distributions for object pose estimation with learnt surface embeddings. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6749–6758, 2022.

[13] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016.

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

[15] Tomas Hodan, Daniel Barath, and Jiri Matas. Epos: Estimating 6d pose of objects with symmetries. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020.

[16] Tomáš Hodaň, Frank Michel, Eric Brachmann, Wadim Kehl, Anders Glent Buch, Dirk Kraft, Bertram Drost, Joel Vidal, Stephan Ihrke, Xenophon Zabulis, Caner Sahin, Fabian Manhardt, Federico Tombari, Tae-Kyun Kim, Jiří Matas, and Carsten Rother. Bop: Benchmark for 6d object pose estimation. *ECCV*, 2018.

[17] Yinlin Hu, Pascal Fua, Wei Wang, and Mathieu Salzmann. Single-stage 6d object pose estimation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020.

[18] Lin Huang, Jianchao Tan, Ji Liu, and Junsong Yuan. Hand-transformer: Non-autoregressive structured modeling for 3d hand pose estimation. In *European Conference on Computer Vision*, pages 17–33. Springer, 2020.

[19] A S M Iftekhar, Hao Chen, Kaustav Kundu, Xinyu Li, Joseph Tighe, and Davide Modolo. What to look at and where: Semantic and spatial refined transformer for detecting human-object interactions. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5353–5363, June 2022.

[20] Shun Iwase, Xingyu Liu, Rawal Khirodkar, Rio Yokota, and Kris M. Kitani. Repose: Fast 6d object pose refinement via deep texture rendering. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2021.

[21] Wadim Kehl, Fabian Manhardt, Federico Tombari, Slobodan Ilic, and Nassir Navab. SSD-6D: Making RGB-based 3D detection and 6D pose estimation great again. In *ICCV*, 2017.

[22] Alex Kendall, Matthew Grimes, and Roberto Cipolla. Posenet: A convolutional network for real-time 6-dof camera relocalization. In *ICCV*, 2015.

[23] Abhijit Kundu, Yin Li, and James M Rehg. 3D-RCNN: Instance-level 3D object reconstruction via render-and-compare. In *CVPR*, 2018.

[24] Yann Labbé, Justin Carpentier, Mathieu Aubry, and Josef Sivic. Cosypose: Consistent multi-view multi-object 6d pose estimation. In *European Conference on Computer Vision*, 2020.

[25] Vincent Lepetit, Francesc Moreno-Noguer, and Pascal Fua. Epnp: An accurate o (n) solution to the pnp problem. *International journal of computer vision*, 81(2):155, 2009.

[26] Hui Li, Zidong Guo, Seon-Min Rhee, Seungju Han, and Jae-Joon Han. Towards accurate facial landmark detection via cascaded transformers. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4176–4185, 2022.

[27] Wenhao Li, Hong Liu, Hao Tang, Pichao Wang, and Luc Van Gool. Mhformer: Multi-hypothesis transformer for 3d

human pose estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 13147–13156, June 2022.

[28] Yi Li, Gu Wang, Xiangyang Ji, Yu Xiang, and Dieter Fox. Deepim: Deep iterative matching for 6d pose estimation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018.

[29] Zhigang Li, Gu Wang, and Xiangyang Ji. Cdpn: Coordinates-based disentangled pose network for real-time rgb-based 6-dof object pose estimation. In *ICCV*, 2019.

[30] Markus Oberweger, Mahdi Rad, and Vincent Lepetit. Making deep heatmaps robust to partial occlusions for 3D object pose estimation. In *ECCV*, 2018.

[31] Jaewoo Park and Nam Ik Cho. Dprost: 6-dof object pose estimation using space carving and dynamic projective spatial transformer. *arXiv preprint arXiv:2112.08775*, 2021.

[32] Kiru Park, Timothy Patten, and Markus Vincze. Pix2pose: Pix2pose: Pixel-wise coordinate regression of objects for 6d pose estimation. In *ICCV*, 2019.

[33] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith 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.

[34] Sida Peng, Yuan Liu, Qixing Huang, Xiaowei Zhou, and Hujun Bao. Pvnet: Pixel-wise voting network for 6dof pose estimation. In *CVPR*, 2019.

[35] Mahdi Rad and Vincent Lepetit. Bb8: A scalable, accurate, robust to partial occlusion method for predicting the 3d poses of challenging objects without using depth. In *ICCV*, 2017.

[36] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You Only Look Once: Unified, real-time object detection. In *CVPR*, 2016.

[37] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *NeurIPS*, 2015.

[38] Ivan Shugurov, Fu Li, Benjamin Busam, and Slobodan Ilic. Osop: A multi-stage one shot object pose estimation framework. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6835–6844, 2022.

[39] Juil Sock, Guillermo Garcia-Hernando, Anil Armagan, and Tae-Kyun Kim. Introducing pose consistency and warp-alignment for self-supervised 6d object pose estimation in color images. In *2020 International Conference on 3D Vision (3DV)*, 2020.

[40] Yongzhi Su, Mahdi Saleh, Torben Fetzer, Jason Rambach, Nassir Navab, Benjamin Busam, Didier Stricker, and Federico Tombari. Zebrapose: Coarse to fine surface encoding for 6dof object pose estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6738–6748, 2022.

[41] Martin Sundermeyer, Zoltan-Csaba Marton, Maximilian Durner, Manuel Brucker, and Rudolph Triebel. Implicit 3D orientation learning for 6D object detection from RGB images. In *ECCV*, 2018.

[42] Mingxing Tan and Quoc Le. Efficientnet: Rethinking model scaling for convolutional neural networks. In *International conference on machine learning*, pages 6105–6114. PMLR, 2019.

[43] Bugra Tekin, Sudipta N Sinha, and Pascal Fua. Real-time seamless single shot 6d object pose prediction. In *CVPR*, 2018.

[44] Zhi Tian, Chunhua Shen, Hao Chen, and Tong He. Fcos: Fully convolutional one-stage object detection. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 9627–9636, 2019.

[45] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[46] Chen Wang, Danfei Xu, Yuke Zhu, Roberto Martín-Martín, Cewu Lu, Li Fei-Fei, and Silvio Savarese. DenseFusion: 6D object pose estimation by iterative dense fusion. *arXiv preprint arXiv:1901.04780*, 2019.

[47] Gu Wang, Fabian Manhardt, Federico Tombari, and Xiangyang 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 (CVPR)*, 2021.

[48] He Wang, Srinath Sridhar, Jingwei Huang, Julien Valentin, Shuran Song, and Leonidas J. Guibas. Normalized object coordinate space for category-level 6d object pose and size estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.

[49] Less Wright. Ranger-Deep-Learning-Optimizer, 2019.

[50] Yu Xiang, Tanner Schmidt, Venkatraman Narayanan, and Dieter Fox. PoseCNN: A convolutional neural network for 6D object pose estimation in cluttered scenes. *Robotics: Science and Systems (RSS)*, 2018.

[51] Sergey Zakharov, Ivan Shugurov, and Slobodan Ilic. Dpod: 6d pose object detector and refiner. In *ICCV*, 2019.

[52] Yi Zhou, Connelly Barnes, Lu Jingwan, Yang Jimei, and Li Hao. On the continuity of rotation representations in neural networks. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.

[53] Xizhou Zhu, Weijie Su, Lewei Lu, Bin Li, Xiaogang Wang, and Jifeng Dai. Deformable detr: Deformable transformers for end-to-end object detection. In *International Conference on Learning Representations*, 2020.

[54] Lu Zou, Zhangjin Huang, Naijie Gu, and Guoping Wang. 6d-vit: Category-level 6d object pose estimation via transformer-based instance representation learning, 2021.