Rheinische
Friedrich-Wilhelms-
Universität Bonn

Institute for Computer Science
Department VI
Autonomous Intelligent Systems

Rheinische Friedrich-Wilhelms-Universität Bonn

Master Thesis

Transformers for 6D Object Pose Estimation

Author:
Arash Amini

First Examiner:
Prof. Dr. Sven Behnke

Second Examiner:
Prof. Dr. Jürgen Gall

Advisor:
Arul Selvam Periyasamy

Date: February 23, 2022

Declaration

I hereby declare that I am the sole author of this thesis and that none other than the specified sources and aids have been used. Passages and figures quoted from other works have been marked with appropriate mention of the source.

Place, Date

Signature

Abstract

6D object pose estimation is the task of predicting the orientation and translation of objects in a given input image. This information is crucial in many fields, e.g., robotics, autonomous driving, and augmented reality. In recent years, the Transformer has dominated the natural language processing area and embarked on achieving state-of-the-art results in many computer vision tasks. One of the notable Vision Transformers is DETR, which models object detection as a set prediction problem and demonstrates impressive results without any handcrafted components. In this thesis, inspired by DETR performance, we formulate the multi-object 6D pose estimation from a monocular image as a set prediction problem and investigate the usage of Transformers for this task. We propose a real-time single-stage architecture that directly regresses the translation and rotation components of the 6D object pose. To further improve the performance, we extend our approach by employing keypoints as 2D projected sparse correspondences to estimate rotation. The proposed methods yield performance competitive with or better than state-of-the-art approaches on the challenging YCB-Video dataset. Moreover, the inference time analysis shows the superiority of our models in terms of inference time, making them ideal for real-world applications.

Contents

1 Introduction

Object pose estimation is one of the crucial tasks in the computer vision field which aims to determine the 3D rotation and 3D translation of an object in camera-centered coordinates having the 3D model points as illustrated in Figure 1.1. Researchers have tried to tackle this problem due to its wide range of applications in many areas such as robotics, autonomous driving, and augmented reality. In robotics for instance, autonomous robotic object manipulation in real-world scenarios like industrial bin picking depends on high-quality 6D object pose estimation. The main challenges for this task is to handle symmetric, occluded and texture-less objects.

Many approaches have been introduced that rely on both RGB and depth data (W. Chen et al. 2020; Choi and Christensen 2016; Y. He et al. 2020; Wada et al. 2020; C. Wang et al. 2019). These methods obtain top performance in terms of accuracy; however, recently the approaches that rely on only RGB images have shown stronger results when training with synthetic data provided by the BOP challenge (Hodaň, Sundermeyer, et al. 2020). Additionally, more methods lately focus on considering only the RGB information since the depth cameras information is not always accessible due to failing of cameras in open air or capturing objects and surfaces made from transparent, reflective, and absorptive materials. The depth sensor also has the drawback of being energy-consuming (Rad and Lepetit 2017).

In recent years, with the advent of Convolutional Neural Networks (CNNs),

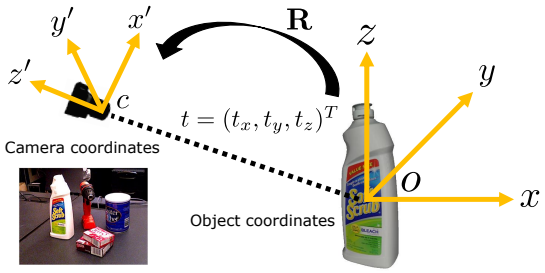

Figure 1.1: Illustration of the 6D object pose estimation task. The image is adapted from (Xiang et al. 2017).

significant progress has been made to boost the pose estimation methods. Due to the complex nature of the problem, the proposed pipelines favor multi-stage architectures, i.e., object detection and/or instance segmentation to detect the objects in a given RGB image, extraction of the target object crop, and finally processing the cropped patch to estimate the 6D object pose. Motivated by the success of deep learning methods for computer vision, in a strive for end-to-end differentiable pipelines, many of the traditional components like Non-Maximum Suppression (NMS) and Region of Interest cropping (RoI) have been replaced by their differentiable counterparts (Girshick 2015; Hosang, Benenson, and Schiele 2017; Ren et al. 2015). Despite these advancements, the pose estimation accuracy still heavily depends on the initial object detection stage.

Recently due to the impressive performance of Transformers in natural language processing tasks, they also received attention in computer vision community, e.g., Carion et al. (2020) introduced DETR, a single-stage non-autoregressive Transformer model that treats object detection as a set prediction problem. In this thesis, we develop end-to-end transformer-based approaches evolved from DETR for multi-object 6D pose estimation from a single RGB image.

We formulate multi-object 6 Degrees-of-Freedom (6DoF) pose estimation as a set prediction problem and introduce two real-time differentiable architectures for this task. Our first proposed approach utilizes independent Multi Layer Perceptrons (MLPs) to directly regress the rotation and translation components. To improve the overall 6D pose estimation accuracy of the proposed direct model, and since the key idea is to target the 6D pose estimation task rather than improving the architecture, we extend our approach by exploiting keypoints as 2D projected sparse correspondences to estimate 3D rotation using a fully connected module, called Rotation Estimation (RotEst).

In this thesis, we attempt to employ Vision Transformers for 6D object pose estimation from a single RGB image. Taking advantage of the non-autoregressive Transformer architecture, our proposed approaches predict 6D pose for all the objects in an image in one forward-pass. We compare our proposed approaches with the state-of-the-art methods on the multi-object single-instance 6D pose dataset, YCB-Video introduced by Xiang et al. (2017).

A part of the material presented in this thesis has been published in the DAGM German Conference on Pattern Recognition (GCPR), 2021 (Amini, Periyasamy, and Behnke 2021). The organization of the thesis is as follows:

- *Chapter 2 - Background*
  The key materials associated with the main topic of this thesis (6D object pose estimation) and the proposed architectures are presented in the back-

ground chapter.

- *Chapter 3 - Related Work*
  In this chapter, we review works that have addressed the 6D object pose estimation. The related work is divided into the following parts: (i) classical approaches, (ii) deep learning based models.

- *Chapter 4 - Method*
  This chapter presents the proposed approaches in this thesis. Each part of the proposed methods and the choices we consider for the corresponding part are explained in detail. The main feature of our approaches is to be single-stage models generating the multi-object 6D poses in one shot.

- *Chapter 4 - Experiments*
  In this chapter, we provide the details of the dataset and metrics utilized to evaluate the proposed methods as well as the description of comprehensive experiments conducted for the proposed approaches.

- *Chapter 5 - Conclusion*
  This chapter summarizes the main contributions of the thesis and discusses the observed results and future works.

2 Background

In this chapter, we present the necessary background material for this thesis. Section 2.1 provides an overview of the original Transformer architecture proposed by Vaswani et al. (2017) and describes its usage in tackling computer vision problems. As the proposed methods are derived from DETR (Carion et al. 2020), the transformer-based architecture, we introduce this model in Section 2.2.

Since the choice of rotation representations has a significant effect on the model performance for 6DoF pose estimation, we briefly explain the commonly used rotation representations in Section 2.3. Finally, in Section 2.4, we describe the PnP algorithms that utilize 2D-3D correspondences to recover the 6D object pose.

2.1 Transformer

Transformer (Vaswani et al. 2017) is a multi-layered architecture based on a self-attention mechanism that enables neural networks to learn long-term dependencies in the input data (see Figure 2.1). One of the main advantages of this attention-based model is the ability of parallel sequence generation, which makes them the best choice for handling long sequences. As a result, Transformers have achieved great success in natural language processing and speech recognition tasks (Devlin et al. 2018; Y. Liu et al. 2019; Radford et al. 2019; Synnaeve et al. 2019). The Transformer can be primarily utilized in three modes: encoder-only, decoder-only, and encoder-decoder. In this thesis, we leverage the encoder-decoder model with a multi-head self-attention mechanism in both encoder and decoder and a multi-head cross-attention mechanism in the decoder.

Each layer of the Transformer encoder is comprised of a multi-head self-attention mechanism, a feed-forward network, layer normalization (Ba, Kiros, and Hinton 2016) modules, and residual connections. The embed of input is added to the positional encoding and passed through the encoder layer. Positional encodings can be either fixed like sinusoidal functions (Vaswani et al. 2017) or learnable embeddings. The sinusoidal fixed encodings allow the model to attend the relative positions since for any fixed offset k, the positional encoding at $pos + k$ can be represented as a linear function of positional encoding at pos (Vaswani et al. 2017).

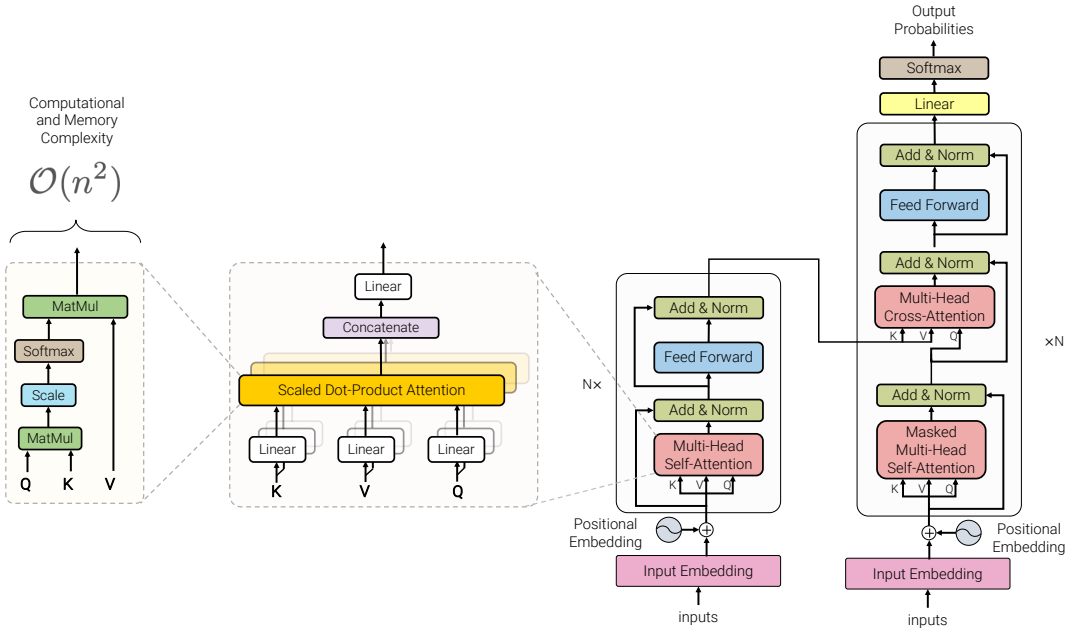

Figure 2.1: The Transformer architecture. The figure is taken from (Tay et al. 2020).

As illustrated in Figure 2.1, a Transformer decoder layer enjoys the same components as the encoder layer along with a multi-head cross-attention mechanism that provides the decoder with information from the encoder.

2.1.1 Transformers in Computer Vision

Lately, Transformers are also gaining attraction in the computer vision community. Architectures that replace CNNs completely with Transformers are achieving state-of-the-art results in image classification (Dosovitskiy et al. 2020; Z. Liu, Y. Lin, et al. 2021; El-Nouby et al. 2021; Touvron, Cord, Douze, et al. 2020; Touvron, Cord, Sablayrolles, et al. 2021; Zhang et al. 2021), object recognition (Dosovitskiy et al. 2020; Z. Liu, Y. Lin, et al. 2021; Zhang et al. 2021), instance and semantic segmentation (El-Nouby et al. 2021; W. Wang et al. 2021a), semantic segmentation (Z. Liu, Y. Lin, et al. 2021; Strudel et al. 2021; H. Wang et al. 2020; Zheng et al. 2021), image synthesis (M. Chen et al. 2020; Esser, Rombach, and Ommer 2020; Jiang, Chang, and Z. Wang 2021), object tracking (Meinhardt et al. 2021; Sun et al. 2020; Xu et al. 2021; Zeng et al. 2021), and video analysis (Arnab et al. 2021; Z. Liu, Ning, et al. 2021). The proposed methods in this thesis are derived from DETR (Carion et al. 2020), a single-stage Vision Transformer model for objection detection, which is described in the next section.

2.2 DETR

Carion et al. (2020) introduce DETR, an end-to-end differentiable object detection model using the Transformer architecture. In contrast to the modern models which address object detection, the problem of estimating the bounding boxes and class label probabilities, as an indirect set prediction problem, DETR formulate this problem as a direct set prediction task. The DETR architecture includes three main components: a CNN backbone for feature extraction, an encoder-decode Transformer with non-autoregressive parallel decoding, and a Feed Forward Network (FFN) to generate the final predictions. Given an RGB input image, the DETR model outputs a set of tuples with fixed cardinality. Each tuple consists of the bounding box and class label probability of an object. To allow an output set with a fixed cardinality, a larger cardinality is chosen, and a special class id $\emptyset$ as "no object class" is used for padding the rest of the tuples in addition to the actual object detections. The tuples in the predicted set and the ground truth target set are then matched by bipartite matching using the Hungarian algorithm.

The DETR model achieves competitive results for both object detection and panoptic segmentation on the COCO dataset (T.-Y. Lin et al. 2014) compared to standard CNN-based architectures. Although this model yields better performance on detecting large objects, it shows poor results for small ones. The DETR also has a slower convergence than the existing object detectors. One of the recent works which addresses the mentioned limitations of DETR is Deformable DETR (Zhu et al. 2021) in which perfectly combines the sparse spatial sampling of deformable convolutions and the capability of Transformers in relation modeling.

2.3 Rotation Representation

Theoretically, it is possible to represent any 3D rotation with three dimension. However, due to instabilities representing of this way, recent methods utilize more compact rotation representations such as axis-angles and quaternions. Any 3D rotation can be specified by an axis of rotation and a rotation angle around that axis. Therefore, an axis-angle rotation can be represented by four numbers $(\theta, \hat{x}, \hat{y}, \hat{z})$, where $[\hat{x}, \hat{y}, \hat{z}]^T$ is a unit vector that defines the axis of rotation, and θ is the amount of rotation around this vector. Generally, a quaternion is represented by four elements (q_0, q_1, q_2, q_3) as follows (Vince 2011):

$$q = q_0 + iq_1 + jq_2 + kq_3, \tag{2.1}$$

where q_0, q_1, q_2 and q_3 are real numbers, and i, j and k are mutually orthogonal imaginary unit vectors. The q_0 term is referred to as the real component, and the remaining three terms are the imaginary components. Given the axis-angle components $(\theta, \hat{x}, \hat{y}, \hat{z})$, we can convert to a rotation quaternion q as follows:

$$\begin{aligned} q_0 = \cos(\theta/2),\ q_1 = \hat{x}\sin(\theta/2), \\ q_2 = \hat{y}\sin(\theta/2),\ q_3 = \hat{z}\sin(\theta/2). \end{aligned} \tag{2.2}$$

Zhou et al. (2019) show that there cannot exist any continuous representation for 3D rotation in the Euclidean space with four or fewer dimensions. Moreover, the mentioned representations are not ideal for the convergence of deep neural networks; therefore, they introduce a novel 6D continuous representation for the rotation matrix R in $SO(3)$, which has shown superior performance compared to other representations. Following the state-of-the-art methods, we utilize the 6D continuous rotation representation, which is further explained in the method chapter.

2.4 P*n*P Algorithms and Variants

Given a set of n 3D keypoints and their corresponding 2D projections as well as the camera intrinsic matrix, the goal of the Perspective-n-Point (PnP) problem is to recover the 6D object pose consisting of rotation and translation. To solve this problem, there has been proposed different algorithms including standard PnP (Gao et al. 2003) and non-iterative EPnP (Lepetit, Moreno-Noguer, and Fua 2009).

Based on (Lepetit, Moreno-Noguer, and Fua 2009), four points are generally sufficient to estimate the pose in the P*3*P problem. However, recent approaches exploit dense 2D-3D correspondences; therefore, to deal with these correspondences, the PnP algorithms are commonly used in combination with the RANdom SAmple Consensus (RANSAC) algorithms. The RANSAC algorithm is employed to improve the robustness against outliers. This algorithm aims to find the best hypothesis that fits the given data containing outliers. It first finds the model parameters that fit the measurements sampled randomly. Then the fitting model is checked for all the measurements. If they fit, they are counted as inliers. The RANSAC algorithm repeats the mentioned steps, and the estimated model with the highest number of inliers is selected. Afterward, the model is refined by reestimating it using all inliers.

Although the PnP-RANSAC algorithms provide promising results, they are not trivially differentiable. In order to realize an end-to-end differentiable pipeline for

6D object pose estimation, the state-of-the-art approaches (B. Chen et al. 2020; Hu, Fua, et al. 2020; S. Li et al. 2021; G. Wang et al. 2021) present a learning-based PnP module. For instance, the learnable Patch-PnP module is introduced by G. Wang et al. (2021) which contains three CNN layers in combination with Group Normalization (Y. Wu and K. He 2018) and ReLU activation followed by two Fully connected (FC) layers to reduce the dimension. This module takes dense 2D-3D correspondences and surface region attention as inputs, and finally, two parallel FC layers generate the 3D rotation and 3D translation components. B. Chen et al. (2020) propose the differentiable BPnP module which can be backpropagated through PnP by using the Implicit Function Theorem (Krantz and Parks 2002) to compute the gradients.

3 Related Work

This chapter reviews the literature proposed for 6D object pose estimation. First, we briefly introduce the top-performance classical (RGB-D and depth-based) approaches in Section 3.1. The Point Pair Features-based (PPF-based) methods show the best performance among the classical approaches. In recent studies, architectures using the power of CNNs have been able to reach the PPF-based methods.

Similar to many computer vision tasks, the recent significant progress on the task of 6D object pose estimation from a single RGB image is driven by deep learning methods. Therefore, an overview of the CNN-based approaches are provided in Section 3.2. These methods can be broadly classified into three major categories, namely direct, indirect, and refinement-based approaches. Direct methods regress object pose-related parameter presentations directly from an image. In contrast, indirect approaches aim to retrieve the 6D pose from the estimated 2D-3D correspondences using PnP algorithm. PnP algorithm is usually used in conjunction with RANSAC for improving the robustness of the pose estimation. Refinement-based methods formulate 6D pose estimation as an iterative refinement problem. These approaches are often combined with direct or indirect methods, i.e., direct or indirect methods produce initial pose estimate, and the refinement-based methods are used to refine the initial pose estimate to predict the final accurate pose.

3.1 Classical Methods

Researchers have attempted to tackle the task of 6D object pose estimation for a long time; therefore, there are numerous approaches proposed before the appearance of CNNs. The classical methods can be divided into four categories:

First, PPF-based approaches (Drost et al. 2010; Vidal et al. 2018): where the aim is to detect 3D objects in point clouds. Drost et al. (2010) utilize matching oriented point pairs between the point cloud of the test scene and the object model and group the matches with a local voting scheme. During training, point pairs from the model are sampled and stored in a hash table. At test time, reference points are fixed in the scene, and a low-dimensional parameter space for the voting

scheme is created, which is restricted to those poses that align the reference point with the model. Point pairs between the reference point and other scene points are generated, similar model point pairs are searched through the hash table, and a vote is cast for each matching point pair. Finally, pose candidates are extracted from the peaks in the accumulator space, which are then refined by the coarse-to-fine Iterative Closest Point (ICP) algorithm and re-scored by the relative amount of visible model surface. While providing promising results, the PPF-based methods are vulnerable to fail in the presence of sensor noise and background clutter.

Second, template matching methods (Hodaň, Zabulis, et al. 2015): templates are the representations of an object utilizing images that depict the object from different angles. The task can be considered as matching an image region to the best fit within the templates. Although working perfectly on non-textured objects, template matching approaches generally do not generate good results for the occluded object.

Third, learning-based approaches (Brachmann, Krull, et al. 2014; Brachmann, Michel, et al. 2016; Kehl, Milletari, et al. 2016; Tejani et al. 2014): in (Brachmann, Krull, et al. 2014) a regression forest estimates the object identity and coordinate for each pixel of an input image. Each object coordinate prediction defines a 3D-3D correspondence between the image and the 3D object model. A RANSAC-based algorithm samples sets of feature correspondences to create a pool of pose hypotheses. The final hypothesis is selected and iteratively refined to maximize the alignment of predicted correspondences and observed depth with the object model.

Finally, methods based on 3D local features (Buch, Petersen, and Krüger 2016; Glent Buch, Kiforenko, and Kraft 2017): Buch, Petersen, and Krüger (2016) propose a RANSAC-based method that iteratively samples three feature correspondences between the object model and the scene. These correspondences are generated by matching 3D local shape descriptors and are used to estimate the 6D object pose which is refined by ICP. All of the mentioned methods take advantage of RGB-D or depth information, and the approaches based on PPF provide superior performance.

3.2 CNN-based Methods

3.2.1 Direct Approaches

One of the existing ways to estimate 6DoF pose is to formulate the pose estimation problem as a regression of continuous rotation and translation components directly from the RGB images. The state-of-the-art examples that follow the di-

rect approach for 6D object pose estimation include (Periyasamy, Schwarz, and Behnke 2018; Xiang et al. 2017).

Xiang et al. (2017) propose an end-to-end regression method called PoseCNN that decouples the object pose estimation into three related tasks (see Figure 3.1): 1) semantic labeling, i.e., prediction of an object label for each pixel in the input image, 2) 3D translation estimation by localizing the 2D coordinates of the object center via predicting a unit vector from each pixel towards the center and estimating the distance of the object center with respect to the camera, and 3) estimating 3D rotation by regressing a quaternion representation. Instead of regressing the 3D translation directly, PoseCNN proposes to localize the 2D object center in the image and estimate object distance from the camera. The 3D translation is finally recovered using the known camera intrinsic matrix, 2D object center, and its distance. Since the 2D object center can be occluded, it is not possible to directly detect the center point. Therefore, PoseCNN network regresses the center direction for each pixel in the image, and Hough voting is used to localize the 2D object center in which each pixel adds votes for image locations along the ray predicted from the network. Moreover, this method introduces a novel loss function named ShapeMatch-Loss, which concentrates on matching the 3D shape of an object. This new loss function handles symmetric objects during training, as different orientations of symmetric objects may generate identical observations.

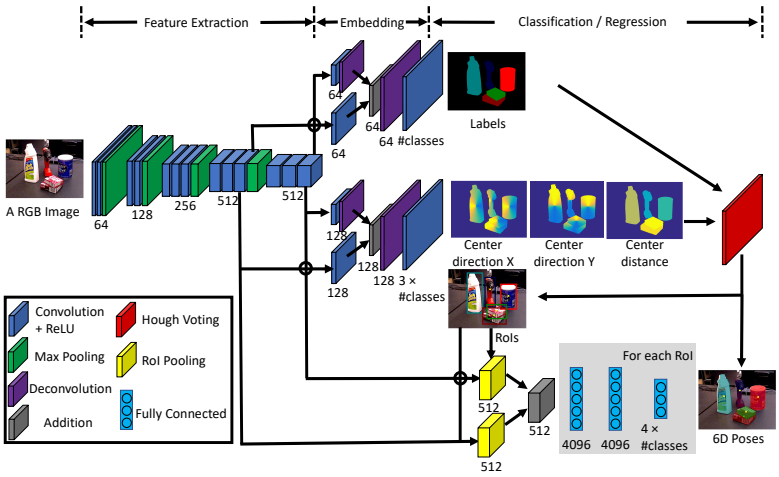

Figure 3.1: PoseCNN architecture, a direct method proposed for 6D object pose estimation. The figure is taken from (Xiang et al. 2017).

In (Periyasamy, Schwarz, and Behnke 2018) authors present a direct pose estimation network based on regression of a 5D pose, i.e., 3D rotation and 2D translation in the image plane. The 2D translation is then transformed into 3D using depth information. In contrast to the RGB-D methods, their approach utilizes

depth only for the capturing process before training the model and transforming the inferred 5D pose estimate to 6D.

Conversely, Kehl, Manhardt, et al. (2017) and Sundermeyer et al. (2018a) predict the discretized rotation component instead of regression. SSD-6D (Kehl, Manhardt, et al. 2017) introduces a direct prediction pipeline, which is the extension of the 2D object detection architecture, SSD (W. Liu et al. 2016). The idea of SSD is that instead of considering input as discrete points, which results in continuous output, we can treat the input as a dense space on the whole image and the output as a discretized space into many overlapping bounding boxes of different shapes and sizes. Following this approach, SSD-6D discretizes the rotation space into classifiable viewpoint bins and handles rotation estimation as a classification problem.

The direct methods have struggled to compete in terms of accuracy with indirect architectures. The aim of indirect models is to establish 2D-3D correspondences and solve them to predict pose-related parameters, which is introduced in the following section. Although the indirect approaches have achieved state-of-the-art results, these methods cannot be trained end-to-end and require extra computation for pose optimization. Therefore, recently there has been an intensive research effort to modify the indirect approaches to direct methods (Hu, Fua, et al. 2020; Z. Li, G. Wang, and Ji 2019; G. Wang et al. 2021).

Hu, Fua, et al. (2020) present the Single-Stage method to adapt the indirect methods to direct methods via using neural networks to estimate 2D-3D correspondences directly and leveraging deep learning networks to approximate the Perspective-n-Point (PnP). The proposed network directly regresses the 6D pose from 2D-3D correspondences associated with each 3D object keypoints. The architecture is motivated from PointNet (Qi et al. 2017) with the main difference that the order of the groups is considered fixed and corresponds to specific 3D keypoints, which is opposite of the PointNet design that is invariant to rigid transformations. Moreover, the Single-Stage framework is invariant to the order of the correspondences in each group; therefore, they design a grouped feature aggregation method to extract the representation of each cluster. They integrate the proposed network with two state-of-the-art keypoints-based methods SegDriven (Hu, Hugonot, et al. 2019) and PVNet (Peng et al. 2019). The results show that the new end-to-end trainable methods yield better accuracy and running time performance than the original architectures. Although the results prove that the Single-Stage approach works well, disregarding the fact that the correspondences are organized by the image pixels leads to decline the performance strongly as shown in (Ma et al. 2020).

Coordinates-based Disentangled Pose Network (CDPN) is introduced by Z. Li,

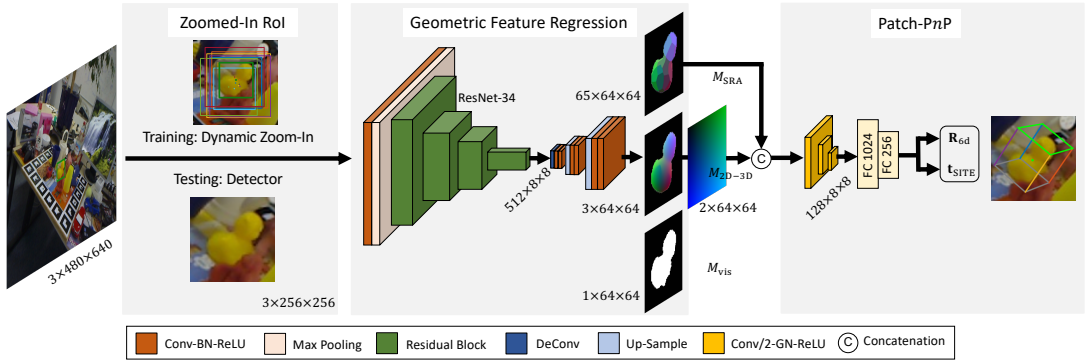

Figure 3.2: The architecture of GDR-Net. This model has end-to end differentiable pipeline via leveraging learning-based PnP module. The figure is taken from (G. Wang et al. 2021).

G. Wang, and Ji (2019) to disentangle the estimation of rotation and translation components. The CDPN method first utilizes a lightweight detector for detecting all objects. The model then zooms in on the target object one at a time via the proposed Dynamic Zoom In (DZI) to predict the 6D object pose using segmentation and coordinates maps which are fed to a PnP-RANSAC algorithm for rotation estimation, and translation is directly estimated with the proposed Scale-Invariant representation for Translation Estimation (SITE) based on the local image patches. Even though CDPN shows robustness with respect to occlusion and clutter, this method ignores the importance of handling symmetries.

Inspired by the CDPN method, G. Wang et al. (2021) present a framework to estimate 6D pose via exploiting dense correspondence-based intermediate representations in a fully differentiable way, called Geometry-guided Direct Regression Network (GDR-Net). After detecting all objects using an off-the-shelf detector as illustrated in Figure 3.2, GDR-Net zooms in on the corresponding Region of Interest (RoI). The zoomed-in RoI will be fed to the proposed network to predict three intermediate geometric feature maps: the 2D-3D Correspondences Map (M_{2D-3D}), the Surface Region Attention Map (M_{SRA}), and the Visible Object Mask (M_{vis}). Finally, the 6D pose of the target object is directly regressed from M_{2D-3D} and M_{SRA} using a learnable Patch-PnP solver consisting of 2D CNNs and fully connected layers. The authors utilize the 6D continuous representation to parameterize the 3D rotation proposed in (Zhou et al. 2019), and they follow the method proposed in the CDPN approach for translation estimation.

3.2.2 Indirect Approaches

As explained earlier, the indirect methods (Hodaň, Baráth, and Jiří Matas 2020; Hu, Fua, et al. 2020; Hu, Hugonot, et al. 2019; Peng et al. 2019; Rad and Lepetit 2017; Tekin, Sudipta N Sinha, and Fua 2018a) have a two-stage paradigm: first using a deep network to estimate correspondences between 3D object points and their 2D image projections, followed by a RANSAC based Perspective-n-Point (PnP) algorithm to compute the 6D pose parameters.

As the pioneer of indirect approaches with a multi-stage framework, BB8 (Rad and Lepetit 2017) exploits the object appearance itself instead of surface keypoints to predict the 6D object pose. The target objects in 2D are detected using object segmentation, and then another network is utilized to estimate 2D projections of the 3D bounding box corners to create 2D-3D correspondences. Finally, the 3D pose is computed by solving a PnP algorithm based on these 2D-3D correspondences. The BB8 framework has two main limitations: not being end-to-end trainable and time-consuming for inference.

EPOS (Hodaň, Baráth, and Jiří Matas 2020) represent the target object via compact surface fragments. The correspondences between densely sampled pixels and the surface fragments are estimated by utilizing an encoder-decoder convolutional neural network. The proposed network generates three outputs per pixel: the probability of each object's presence, the probability of the fragments given the object's presence, and the exact 3D location on each fragment. The object pose estimation from many-to-many 2D-3D correspondences is then estimated using a variant of PnP-RANSAC algorithm, which is explained in the following, integrated with the Progressive-X scheme (Barath and Jiri Matas 2019). Pose hypotheses are proposed by GC-RANSAC (Barath and Jiri Matas 2018) which utilizes the spatial coherence of correspondences, and the pose is estimated from sampled correspondences using the EPnP solver (Lepetit, Moreno-Noguer, and Fua 2009) followed by the Levenberg-Marquardt optimization (Moré 1978). Efficiency is achieved by the PROSAC sampler (Chum and J. Matas 2005) that prioritizes correspondences with a high predicted probability. The results prove that this method is capable of handling objects with global or partial symmetries.

To be able to handle the poses of occluded or truncated objects, (Peng et al. 2019) propose the Pixel-wise Voting Network (PVNet) as illustrated in Figure 3.3. Instead of directly regressing image coordinates of keypoints, PVNet predicts unit vectors that represent pixel-wise directions of the object pointing to the keypoints. Keypoints need to be on the object surface, and they should be spread out on the surface to find a stable solution of the PnP problem. Therefore, PVNet exploit the Farthest Point Sampling (FPS) algorithm to select eight keypoints. These unit

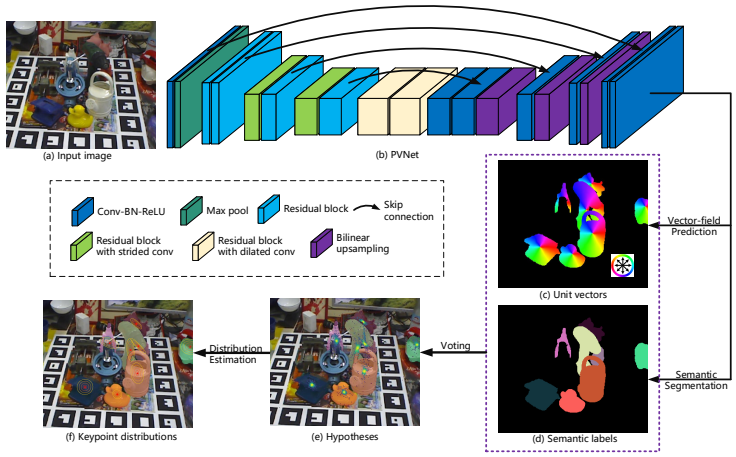

Figure 3.3: The diagram of PVNet, one of the indirect methods. The figure is taken from (Peng et al. 2019).

vectors then vote for the keypoint locations based on RANSAC (Fischler and Bolles 1981). As explained earlier, PoseCNN (Xiang et al. 2017) introduced a similar idea for object detection. Generating a vector-field representation for keypoint localization instead of coordinates or heatmaps has two advantages. The vector-field representation makes the network focus on local features of objects and spatial relations between object parts, which robustly estimates the occluded or truncated keypoints from the visible parts and yields good performance. Moreover, the dense outputs generated by the vector-field representation provide the uncertainty-driven PnP algorithm in which the spatial probability distribution for each keypoint due to the RANSAC-based voting helps the PnP solver to estimate the final pose based on consistent correspondences.

Hu, Hugonot, et al. (2019) propose a segmentation-driven 6D pose estimation network called SegDriven, where instead of a single global prediction, it utilizes predictions of multiple local patches to estimate 6D poses robust to occlusions. Each visible object patch contributes a pose estimate for the object it belongs to in the form of the predicted 2D projections of predefined 3D keypoints. Then, using confidence values also predicted by the SegDriven network, the most reliable 2D projections for each 3D keypoints are combined. A RANSAC-based PnP algorithm on these 2D-3D correspondences is applied to infer a pose per object.

While indirect methods are dominating in terms of accuracy, they suffer from several drawbacks: 1) the loss function trained in this paradigm is designed to handle a surrogate objective rather than the 6D pose estimation error, 2) due to being multi-staged, these approaches are not differentiable and end-to-end trainable, and 3) the iterative process of RANSAC is very time-consuming for dense

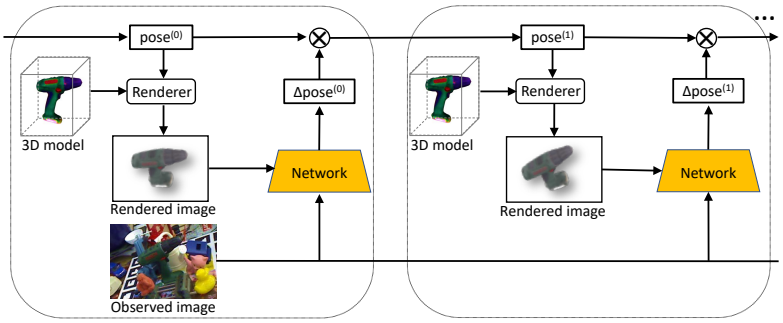

Figure 3.4: The DeepIM network, which is a refinement-based approach. The figure is taken from (Yi Li et al. 2018).

2D-3D correspondences.

3.2.3 Refinement-based Approaches

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 minimize the error between the current pose prediction and the ground truth. Refinement-based methods (Labbe et al. 2020; Yi Li et al. 2018; Manhardt, Kehl, et al. 2018; Periyasamy, Schwarz, and Behnke 2019; Shao et al. 2020) achieve the highest pose prediction accuracy among the mentioned categories (Hodaň, Sundermeyer, et al. 2020); however, these approaches are often slower than the direct and indirect methods, and their speed depends heavily on the used renderer and the number of iterations (Fan et al. 2021).

Yi Li et al. (2018) propose DeepIM as a deep learning-based pose refinement network for iterative 6D pose matching. Given an initial pose estimation, DeepIM generates the relative rotation and translation that can refine the initial 6D pose estimation via minimizing the differences between the observed image and the rendered image under the current pose as illustrated in Figure 3.4. The refined pose is then utilized as the initial pose for the next iteration. This procedure continues until the refined pose converges or iterations reach a pre-defined number.

CosyPose is developed by Labbe et al. (2020) to integrate the information from multiple views to jointly predict poses of multiple objects for the reconstruction of the input scene. The proposed method consists of three steps and generates a scene model, including the number of per type objects, their corresponding 6D poses, and the relative poses of the cameras. In the first step, 6D object pose hypotheses are estimated from each view by following the render-and-compare approach of DeepIM (Yi Li et al. 2018). In the second step, the object-level matching is

conducted using RANSAC to optimize the overall scene consistency. Finally, the 6D poses of all objects and cameras are refined to minimize a global reprojection error.

3.3 Our Approach

In this thesis, we propose two real-time transformer-based methods that estimate multi-object 6D pose directly from a single RGB image in one forward-pass. Similar to the state-of-the-art RGB-based methods, our proposed approaches are direct.

Following the DETR (Carion et al. 2020) approach, we model 6DoF pose estimation as a set prediction problem and utilize a loss based on the Hungarian algorithm to find a bipartite matching between prediction and ground truth. In our first approach, we directly regress the 3D rotation and 3D translation components. To improve the performance of rotation, we extend our direct method by leveraging keypoints as 2D projected sparse correspondence.

Although in terms of the 6D pose prediction performance, refinement-based approaches achieve considerably better results than the direct methods, in this thesis we will demonstrate that without any pose refinement, this gap in performance is shrinking. Furthermore, we will show that our proposed methods enjoy the fastest inference time compared to the state-of-the-art pose estimators.

4 Method

4.1 6D Object Pose

The object pose can be represented by a 4×4 matrix $P \in SE(3)$ constitutes of a 3×3 rotation matrix R in $SO(3)$ and 3×1 translation vector $t \in \mathbb{R}^3$. The matrix P transforms a 3D homogeneous point x_m in the model coordinate system to a 3D point x_c in the camera coordinate system:

$$x_c = \overbrace{[R, t; 0_{1\times 3}, 1]}^{P} x_m. \tag{4.1}$$

As the choices of rotation and translation representations have major impacts on the performance of 6D pose estimation, we describe our choices for representation of each component in this section.

4.1.1 Parameterization of Rotation

Following the state-of-art-arts approaches, we exploit the 6D continuous rotation representation introduced in (Zhou et al. 2019). The authors advance the continuous representation definition by taking into account the training of deep neural networks. As a result, they mathematically prove that for 3D rotations, all representations, e.g., axis-angles and quaternions are discontinuous in the real Euclidean spaces of four or fewer dimensions. Therefore, they present the 5D and 6D continuous representations, which among them the 6D representation yields the best performance in practice. The 6D representation denoted as R_{6d} is defined as the first two columns of the rotation matrix $R = [R_{.1} \,|R_{.2}|\, R_{.3}]$:

$$R_{6d} = [R_{.1} \mid R_{.2}] \tag{4.2}$$

Having a 6-dimensional vector $R_{6d} = [r_1 \mid r_2]$ and using the vector normalization operation $\phi(.)$, the rotation matrix R can be calculated as follows:

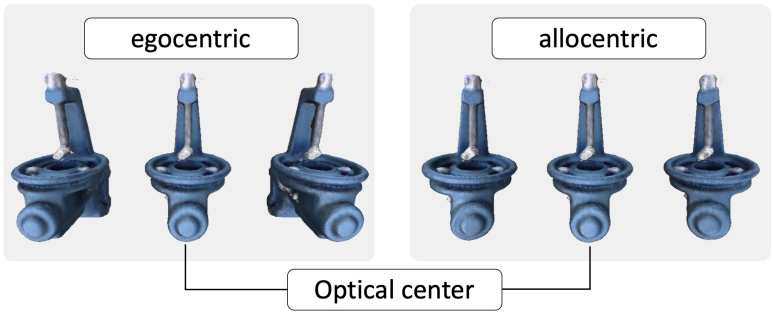

Figure 4.1: The egocentric *vs* allocentric rotation. The figure is taken from (Manhardt, G. Wang, et al. 2020).

$$R = \begin{cases} R_{.1} = \phi\left(r_1\right) \\ R_{.3} = \phi\left(R_{.1} \times r_2\right) \\ R_{.2} = R_{.3} \times R_{.1} \end{cases} \tag{4.3}$$

Generally, the object rotation representation can be either egocentric, rotation with respect to camera, or allocentric, rotation with respect to object (see Figure 4.1). In the case of the object's egocentric representation, the azimuth of the object does not change while the object appearance considerably changes. Conversely, objects with similar allocentric rotation will also have analogous appearance (Kundu, Yin Li, and Rehg 2018).

4.1.2 Parameterization of Translation

Considering the translation $t = [t_x, t_y, t_z]^T \in \mathbb{R}^3$ as the coordinate of the object origin in the camera coordinate system, the naive approach for estimating t is to regress it directly in 3D space. This method is not generalizable as objects can appear in any location in the image; moreover, it cannot deal with multiple object instances of the same category (Xiang et al. 2017). Therefore, we can consider the proposed method in PoseCNN by Xiang et al. (2017) in which decouples the estimation of t into directly regressing the object's distance from the camera t_z and the 2D location of projected 3D object's centroid in the image plane $[c_x, c_y]^T$. Given the intrinsic camera matrix K which consists of the focal lengths of the camera f_x, f_y and the principal point $[x_0, y_0]$ parameters (Hartley and Zisserman

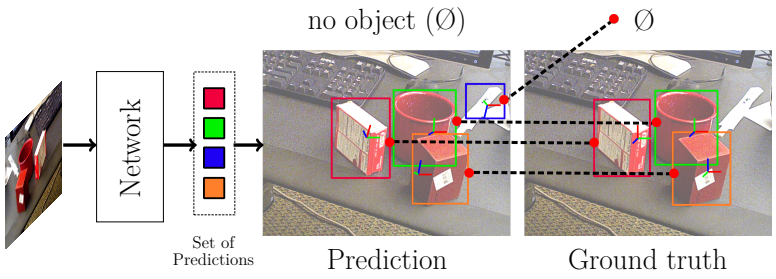

Figure 4.2: The overall of our proposed pipeline for 6D object pose estimation.

2004), t_x and t_y can be recovered as follows:

$$\begin{bmatrix} c_x \\ c_y \end{bmatrix} = \frac{1}{t_z} \overbrace{\begin{bmatrix} f_x & 0 & x_0 \\ 0 & f_y & y_0 \\ 0 & 0 & 1 \end{bmatrix}}^{K} \begin{bmatrix} t_x \\ t_y \\ t_z \end{bmatrix} \tag{4.4}$$

4.2 Overall Pipeline

The overview of the pipeline proposed in this thesis for 6DoF pose estimation is demonstrated in Figure 4.2. We formulate 6D object pose estimation as a set prediction problem and develop transformer-based architectures that take a single RGB image as input and simultaneously generate a set of N elements where each element corresponds to an object prediction. To facilitate the prediction of a varying number of objects in the image, we choose N to be larger than the expected number of objects in the image, enabling the network to have enough options for freely embedding each object. After predicting all objects in the given image, the rest of the elements are padded with Ø as no object predictions (see Figure 4.2). The predicted and ground truth sets are then matched using bipartite matching, and the model is trained to minimize the Hungarian loss between the matched pairs.

4.3 Network Architecture

Our proposed end-to-end differentiable architecture is depicted in Figure 4.3. Given an RGB input image, we extract lower-resolution image features using the standard ResNet (K. He et al. 2016) model in the backbone network and flatten them to create feature vectors suitable for an encoder-decoder Transformer model. The flattened features are then supplemented with positional encodings and fed

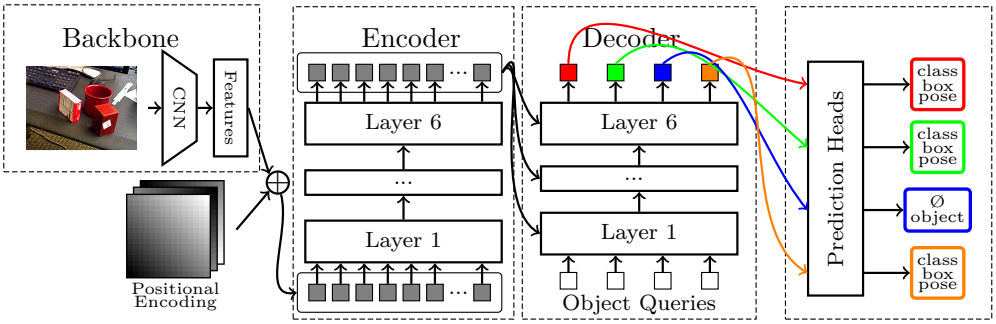

Figure 4.3: The proposed architecture in detail.

to the Transformer encoder. The encoder module consists of six standard encoder layers. The output of the encoder is provided to the decoder module along with N learnable positional encodings, called object queries. Similar to the encoder, the decoder module also includes six decoder layers generating N output embeddings (object queries). Each decoder output is processed independently by shared MLP prediction heads to generate a set of N tuples in parallel.

4.3.1 Backbone

Among CNN-based models, the ResNet (K. He et al. 2016) model is commonly chosen as the backbone network; however, only single-scale low-resolution features from the last layer are used. The early layers of ResNet have more spatial information, whereas the network extracts more semantic information in the final layers. As a result, the RefineNet (G. Lin et al. 2017) model, which improves ResNet by exploiting multi-scale features, can be another candidate for the CNN-based backbone. Additionally, we have the option of using vision Transformers for the backbone network, namely Swin Transformer (Z. Liu, Y. Lin, et al. 2021), Pyramid Vision Transformer (W. Wang et al. 2021b), or Focal Transformer (Yang et al. 2021).

Since in this thesis, our main focus is to tackle the 6D pose estimation task rather than improving the architecture, between the mentioned models, we opt for ResNet due to the availability of its pretrained weights and our computational limitation; although, having a more powerful backbone model yields better performance as shown in DETR (Carion et al. 2020).

The first layer of ResNet is a 7×7 convolutional with stride 2, followed by a max-pooling layer. The rest of the network consists of four modules of residual blocks with higher depths and lower resolutions as the number of modules increases. Each residual block includes two or three convolutional layers depending on the selected

ResNet architecture, followed by batch normalization and ReLU activation, and a shortcut connection. Moreover, we can consider the dilated version of ResNet models where the feature resolution is increased by adding a dilation to the last stage of the model and removing a stride from the first convolution of this stage. This adaption expands the resolution twice while increasing the overall computing cost two times.

In this thesis, we utilize the ResNet50 (K. He et al. 2016) backbone model pretrained on ImageNet (Deng et al. 2009) with frozen batch normalization layers. For an image size of height H and width W, the ResNet50 backbone network extracts 2048 low-resolution feature maps of size $\frac{H}{32} \times \frac{W}{32}$. We then reduce the dimension of the feature maps to d using 1×1 convolution, and finally flatten the features into $d \times \frac{H}{32} \frac{W}{32}$ feature vectors to prepare them for the input of Transformer model.

4.3.2 Positional Encoding

A multi-head self-attention mechanism, the core component of the Transformer model, is permutation-invariant, i.e., Transformer ignores the order of the input sequences. To address this limitation, we provide some information about the position of tokens via positional encodings.

The positional encoding used in transformer-based architectures can be divided into two categories: absolute and relative. In absolute positional encoding, the positional encoding is added to the input embed, and the self-attention module considers the absolute position (Vaswani et al. 2017). This absolute positional encoding can be either fixed encodings using sinusoidal functions with different frequencies (Vaswani et al. 2017) or the learnable encodings (C. Li et al. 2018; Vaswani et al. 2017). Recently, there are methods proposed to encode the relative positional between the input elements through embedding the positional encoding vectors inside the self-attention module (Dai et al. 2019; Shaw, Uszkoreit, and Vaswani 2018; K. Wu et al. 2021). In this thesis, we utilize the fixed absolute positional encoding for the Transformer encoder and learnable absolute positional encoding in decoder. More details are explained in Sections 4.3.3 and 4.3.4.

4.3.3 Transformer Encoder

The Transformer encoder module has the standard architecture as proposed by (Vaswani et al. 2017) and consists of six layers, where each layer performs multi-head self-attention of the input sequences. The inputs and output of the multi-head self-attention block are connected by residual connections followed by dropout of

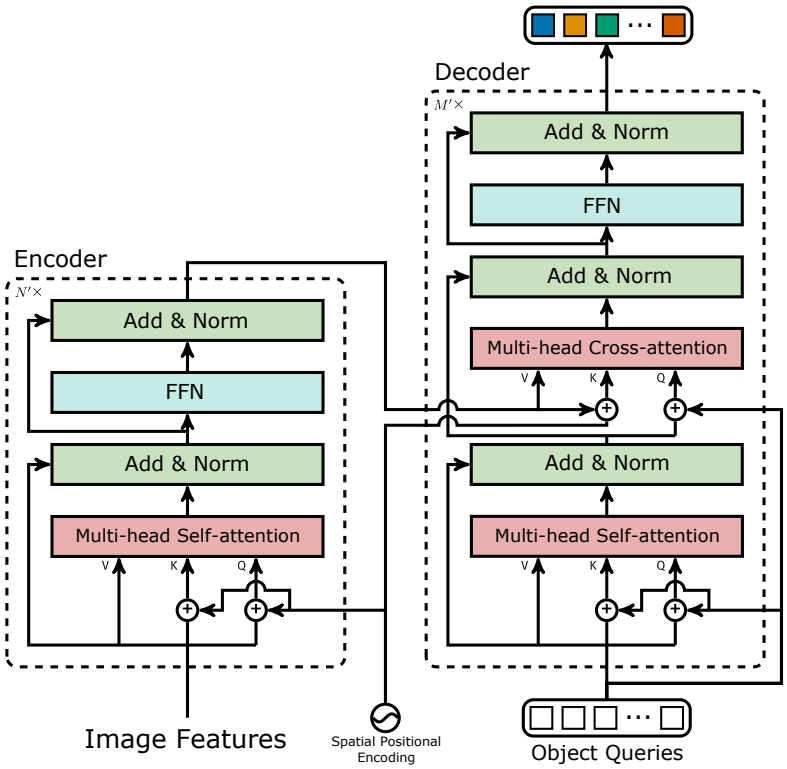

Figure 4.4: The encoder-decoder Transformer architecture in detail. The Transformer takes $d \times \frac{H}{32} \frac{W}{32}$ feature vectors as input and generates N object queries of size d. The figure is adapted from (Carion et al. 2020).

0.1 and normalization layer as illustrated in Figure 4.4. The output of the multi-head self-attention block is then passed to a two-layered Feed Forward Network (FFN) with hidden size of 2048, where its input/output is similarly connected in a residual fashion with dropout of 0.1 and layer normalization. As a result, the Transformer encoder layer can be represented as:

$$\begin{aligned} \mathrm{X} &= \mathit{LayerNorm}\left(\mathit{MultiHeadSelfAttention}\left(\mathrm{I}\right) + \mathrm{I}\right), \\ \mathrm{O} &= \mathit{LayerNorm}\left(\mathit{FFN}(\mathrm{X}) + \mathrm{X}\right), \end{aligned} \tag{4.5}$$

where I is the input of the Transformer encoder layer and O is the layer output. The multi-head self-attention block is elaborated in the following section.

4.3.3.1 Multi-head Self-attention

The Transformer model takes advantage of a multi-head self-attention mechanism in both encoder and decoder modules and a multi-head cross attention mechanism in the decoder as shown in Figure 4.4.

Having the query sequence of length $N_q, X_q \in \mathbb{R}^{d \times N_q}$, and the key-value sequence of length $N_{kv}, X_{kv} \in \mathbb{R}^{d \times N_{kv}}$, and their corresponding additional positional encodings $P_q \in \mathbb{R}^{d \times N_q}$ and $P_{kv} \in \mathbb{R}^{d \times N_{kv}}$, the operation for a single attention head of dimension d can be defined as:

$$\begin{gathered} [Q; K; V] = [W_1'(X_q + P_q); W_2'(X_{kv} + P_{kv}); W_3'(X_{kv})], \\ \mathit{Attn_i}(X_q, X_{kv}, W') = \sum_{j=1}^{N_{kv}} \alpha_{i,j} V_j, \\ Z_i = \sum_{j=1}^{N_{kv}} e^{\frac{1}{\sqrt{d/M}} Q_i^T K_j + Mask}, \alpha_{i,j} = \frac{e^{\frac{1}{\sqrt{d/M}} Q_i^T K_j + Mask}}{Z_i}, \end{gathered} \tag{4.6}$$

where the weight matrix $W' \in \mathbb{R}^{3 \times d/N_H \times d}$ is the concatenation of W_1', W_2', and W_3' in which $N_H = 8$ is the number of heads, and $\alpha_{i,j}$ is the attention weight corresponding to a query index i and a key-value index j. As the size of input images can be different, the padding is conducted in the input vectors to make them have the same size. $Mask \in \mathbb{R}^{N_q \times N_{kv}}$ is used to mask out the padding values by setting them to $-\infty$ so that they don't participate in the attention score. All the embeddings used in our models are 256-dimensional vectors ($d = 256$).

The multi-head attention is the concatenation of N_H single attention heads followed by a linear projection with weight matrix $\tilde{W} \in \mathbb{R}^{d \times d}$, which can be repre-

sented as:

$$X'_q = [Attn\,(X_q, X_{kv}, W_1)\,; \ldots ; Attn\,(X_q, X_{kv}, W_{N_H})]\,,$$
$$\tilde{X}_q = LayerNorm(X_q + Dropout(\tilde{W} X'_q)), \tag{4.7}$$

where [;] denotes concatenation on the channel axis, and the output $\tilde{X}_q \in \mathbb{R}^{d \times N_q}$ has the same size as the input query sequence. Note that the multi-head self-attention is the special case of the described multi-head attention where $X_q = X_{kv}$.

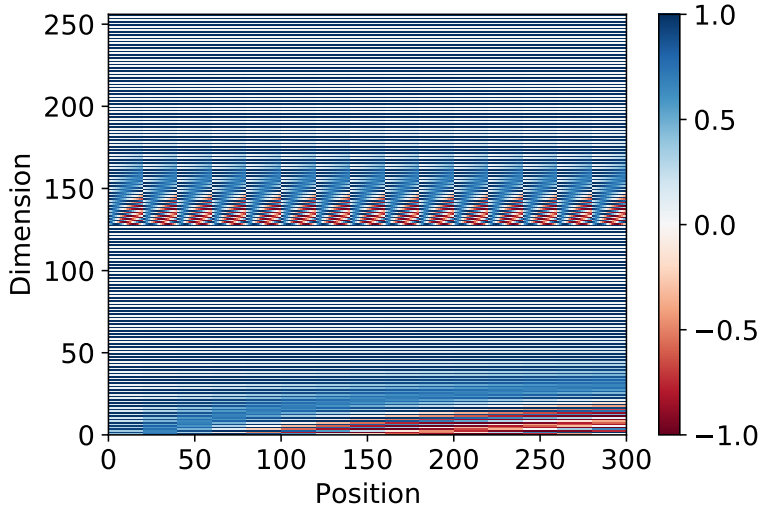

Figure 4.5: The demonstration of absolute sinusoidal positional encodings for 256 feature vectors of size 15×20.

For the encoder, we utilize the absolute positional encoding as the original Transformer (Vaswani et al. 2017). As the DETR (Carion et al. 2020) model shows using learnable positional encoding does not improve the results, we use the fixed sinusoidal positional encoding of 256-dimension with different frequencies that can be specified as:

$$P_{(pos,\ 2i)} = \sin(pos/10000^{2i/d}), \tag{4.8}$$
$$P_{(pos,\ 2i+1)} = \cos(pos/10000^{2i/d}),$$

where $d = 256$, pos is the position of sequence, and i is the index of dimension. Since we need to represent two coordinates, we use $d/2$ of the dimensions to encode the row and the other $d/2$ to encode the column as depicted in Figure 4.5. We can add these positional embeddings to backbone feature vectors and pass them once at the Transformer encoder input similar to the original Transformer or add them to queries at every encoder attention layer, where they are shared across all

encoder layers. The conducted experiments by DETR demonstrate that the latter method leads to better performance. Therefore, following the DETR approach, we pass the positional encodings directly to the attentions.

4.3.4 Transformer Decoder

The difference between the used Transformer decoder and the original version proposed by Vaswani et al. (2017) is that our model simultaneously generates the N output tuples at each decoder layer since objects that appeared in an image are independent. In contrast, the original autoregressive model predicts the output sequence one element at a time. The reason is that the autoregressive decoding mechanism utilized in the original Transformer requires previously generated outputs to produce each new token.

In our proposed architecture, from the encoder output embedding and N positional embedding inputs, the decoder generates N output embeddings of size d using the multi-head self-attention and cross-attention mechanisms as shown in Figure 4.4, where N is the cardinality of the predicted set.

Based on DETR (Carion et al. 2020), the decoder positional encodings, called object queries, are required to be learnable. These learnable positional encodings can be passed once at the decoder input or passed directly through the attention of each decoder layer. We use the latter approach that yields better performance, as mentioned in DETR.

We utilize an additional shared normalization layer to normalize the outputs of decoder layers. From the N decoder output embeddings, we use feed-forward prediction heads to generate a set of N output tuples, which more details are explained in Section 4.3.5. Note that each tuple in the set is produced from a decoder output embedding independently—lending itself for efficient parallel processing.

Computational Complexity In the following analysis, we skip the computations that are negligible. Every encoder self-attention block has complexity $\mathcal{O}(d^2HW + d(HW)^2) : O(d^2/M)$ is the computational cost of a single query/key/value embeddings, while the computational cost of the attention weights for one head is $\mathcal{O}(d(HW)^2/M)$. In the decoder, self-attention and cross-attention are in $\mathcal{O}(d^2N + dN^2)$ and $\mathcal{O}(d^2(N + HW) + dNHW)$, respectively. In practice, we have $N << HW$; therefore, the computing cost in the decoder is much lower than the encoder.

4.3.5 Prediction Heads

Each Transformer decoder output (object query) is processed independently by four feed-forward prediction heads shared across object queries to generate a set of N tuples in parallel. Prediction heads are three-layer MLPs with hidden dimension 256 in each layer and ReLU activation.

Each tuple contains class probability, 2D bounding box, rotation and translation components of 6D pose. The class probability is predicted using a softmax function. The model predicts 2D bounding boxes as the center coordinates, height, and width, all of which are normalized by the input image size. The model's output for rotation is the 6-dimensional vector R_{6d}, and for translation it directly predicts the t vector in meter.

4.4 Pose Estimation as Set Prediction

We model the problem of 6D pose estimation as a direct set prediction problem, inspired by DETR (Carion et al. 2020). The output of the proposed method is a set of tuples. To facilitate the 6D pose prediction of a varying number of objects in an image, we fix the cardinality of the predicted set to N, which is a hyperparameter, and we choose it to be larger than the expected maximum number of objects in the image. We also introduce a no object class $\emptyset$, which is analogous to the background class used in semantic segmentation models. In addition to predicting the classes of objects presented in the image, our model is trained to predict $\emptyset$ class for the rest elements in the set.

4.4.1 Bipartite Matching

To compare the predicted and ground truth sets, we perform bipartite matching to find the permutation of the predicted elements that minimizes the matching cost. Given a predicted set denoted by $\hat{y}$ and n ground truth objects $\{y_1, y_2, ..., y_n\}$, where we pad $\emptyset$ objects to create a ground truth set y of cardinality N, we search for a permutation $\hat{\sigma}$ among the possible permutations $\sigma \in \mathfrak{S}_N$ that minimizes the matching cost $\mathcal{L}_{match}$. Formally, it can be written as:

$$\hat{\sigma} = \underset{\sigma \in \mathfrak{S}_N}{\arg\min} \sum_{i}^{N} \mathcal{L}_{match}(\hat{y}_{\sigma(i)}, y_i), \tag{4.9}$$

where $\mathcal{L}_{match}(\hat{y}_{\sigma(i)}, y_i)$ is the pair-wise matching cost between the prediction at index $\sigma(i)$ and the ground truth tuple y_i. We have two options for defining

$\mathcal{L}_{match}(y_i, \hat{y}_{\sigma(i)})$. One option is to use the same definition proposed by the DETR model (Carion et al. 2020), i.e., to include only bounding box b_i and class probability p_i. We call this variant of matching cost $\mathcal{L}_{match\_object}$:

$$\mathcal{L}_{match\_object}(\hat{y}_{\sigma(i)}, y_i) = -\mathbb{1}_{c_i \neq \emptyset} \hat{p}_{\sigma(i)}(c_i) + \mathbb{1}_{c_i \neq \emptyset} \mathcal{L}_{box}(\hat{b}_{\sigma(i)}, b_i). \tag{4.10}$$

The second option is to consider the pose predictions in the matching cost as well, which can be represented as:

$$\mathcal{L}_{match\_pose}(\hat{y}_{\sigma(i)}, y_i) = \mathcal{L}_{match\_object}(\hat{y}_{\sigma(i)}, y_i) + \mathbb{1}_{c_i \neq \emptyset} \lambda_{pose} \mathcal{L}_{pose}(\hat{R}_{\sigma(i)}, \hat{t}_{\sigma(i)}, R_i, t_i). \tag{4.11}$$

There are different choices for defining $\mathcal{L}_{pose}$ which will be explained thoroughly in Section 4.4.2.3. Among them, we opt for the disentangled pose loss, as calculating the coupled pose loss, especially its symmetric aware version over all the possible permutations is significantly computationally expensive. Therefore, $\mathcal{L}_{pose}$ can be written as:

$$\mathcal{L}_{rot}(\hat{R}_{\sigma(i)}, R_i) + \mathcal{L}_{trans}(\hat{t}_{\sigma(i)}, t_i), \tag{4.12}$$

where $\mathcal{L}_{rot}$ is the angular distance between the predicted and ground truth rotations, and $\mathcal{L}_{trans}$ is the ℓ_1 loss.

4.4.2 Loss Function

After establishing the matching pairs using the bipartite matching, our proposed architecture is trained to minimize the Hungarian loss between the predicted and ground truth target sets consisting of probability loss, bounding box loss, and pose loss as described in Equation 4.13. In the following, further explanation for each term is provided. It is worth mentioning that except for the class probability loss, other terms of the Hungarian loss are normalized by the number of objects in each batch.

$$\mathcal{L}_{Hungarian}(\hat{y}, y) = \sum_{i=1}^{N} [-\log \hat{p}_{\hat{\sigma}(i)}(c_i) + \mathbb{1}_{c_i \neq \emptyset} \mathcal{L}_{box}(\hat{b}_{\hat{\sigma}(i)}, b_i) + \mathbb{1}_{c_i \neq \emptyset} \lambda_{pose} \mathcal{L}_{pose}(\hat{R}_{\hat{\sigma}(i)}, \hat{t}_{\hat{\sigma}(i)}, R_i, t_i)]. \tag{4.13}$$

4.4.2.1 Class Probability Loss

The first component in the Hungarian loss is the class probability loss. We use the standard negative log-likelihood as the class probability loss function. Since we choose the cardinality of the set to be larger than the expected maximum number of objects in an image, the number of Ø classes are imbalanced in comparison to the other object classes. Thus, we weight the log probability loss for the Ø class with a factor of 0.4.

4.4.2.2 Bounding Box Loss

The commonly used bounding box loss is ℓ_1 which suffers from the major disadvantage of having different scales for bounding boxes which their relative errors are similar. Therefore, for bounding box loss $\mathcal{L}_{box}(\hat{b}_{\sigma(i)}, b_i)$, we follow the DETR method and use a weighted combination of the Generalized IoU (GIoU) introduced by Rezatofighi et al. (2019) and the ℓ_1 loss as provided in Equation 4.14. GIoU similar to IoU has the appealing property of being scale-invariant. Furthermore, GIoU has the advantage of being always differentiable , giving it the privilege to be utilized for optimization.

$$\mathcal{L}_{box}(\hat{b}_{\sigma(i)}, b_i) = \alpha\mathcal{L}_{iou}(\hat{b}_{\sigma(i)}, b_i) + \beta||\hat{b}_{\sigma(i)} - b_i||_1, \tag{4.14}$$
$$\mathcal{L}_{iou}(\hat{b}_{\sigma(i)}, b_i) = 1 - \left(\frac{|\hat{b}_{\sigma(i)} \cap b_i|}{|\hat{b}_{\sigma(i)} \cup b_i|} - \frac{|B(\hat{b}_{\sigma(i)}, b_i) \setminus \hat{b}_{\sigma(i)} \cup b_i|}{|B(\hat{b}_{\sigma(i)}, b_i)|}\right),$$

where $\alpha, \beta \in \mathbb{R}$ are hyperparameters. $|.|$ means "area", and the union and intersection of box coordinates are used as shorthands for the boxes themselves. The areas of unions or intersections are computed by min/max of the linear functions of $\hat{b}_{\sigma(i)}$ and b_i, making the $\mathcal{L}_{iou}$ loss sufficiently well-behaved for stochastic gradients. $B(\hat{b}_{\sigma(i)}, b_i)$ is the largest box containing both the prediction $\hat{b}_{\sigma(i)}$ and the ground truth b_i.

4.4.2.3 Pose Loss

Consider the predicted pose $\hat{P}_{\sigma(i)} = [\hat{R}_{\sigma(i)}, \hat{t}_{\sigma(i)}]$, the ground truth pose $P_i = [R_i, t_i]$, and $\mathcal{M}_i$ which indicates the set of 3D model points for each object i, where in this thesis, 1.5K points are subsampled from meshes provided for training. We can supervise the 6D object pose either coupled or disentangled, i.e., the pose loss $\mathcal{L}_{pose}(\hat{R}_{\sigma(i)}, \hat{t}_{\sigma(i)}, R_i, t_i)$ can be presented as coupled or disentangled.

Coupled Pose Loss The coupled pose loss, which is introduced in (Yi Li et al. 2018) as Point Matching loss, computes the average distance between points on the transformed estimated model and their correspondings transformed by the ground truth. The Point Matching loss is described as follows:

$$\mathcal{L}_{pose}(\hat{R}_{\sigma(i)}, \hat{t}_{\sigma(i)}, R_i, t_i) = \frac{1}{|\mathcal{M}_i|} \sum_{\mathrm{x} \in \mathcal{M}_i} \|(\hat{R}_{\sigma(i)}\mathbf{x} + \hat{t}_{\sigma(i)}) - (R_i\mathbf{x} + t_i)\|. \tag{4.15}$$

Considering Point Match loss, we can handle the symmetric objects explicitly via employing symmetric aware loss from (Xiang et al. 2017). In this case, we measure the average closest point distance for symmetric objects which can be expressed as:

$$\mathcal{L}_{pose}(\hat{R}_{\sigma(i)}, \hat{t}_{\sigma(i)}, R_i, t_i) = \begin{cases} \frac{1}{|\mathcal{M}_i|} \sum\limits_{\mathrm{x}_1 \in \mathcal{M}_i} \min\limits_{\mathrm{x}_2 \in \mathcal{M}_i} ||(\hat{R}_{\sigma(i)}\mathrm{x}_1 + \hat{t}_{\sigma(i)}) - (R_i\mathrm{x}_2 + t_i)|| \text{ if sym,} \\ \frac{1}{|\mathcal{M}_i|} \sum\limits_{\mathrm{x} \in \mathcal{M}_i} ||(\hat{R}_{\sigma(i)}\mathrm{x} + \hat{t}_{\sigma(i)}) - (R_i\mathrm{x} + t_i)|| \text{ otherwise.} \end{cases} \tag{4.16}$$

Disentangled Pose Loss In disentangled pose loss, we individually supervise the rotation and translation components of the 6D pose. Therefore, the disentangled pose loss can be written as:

$$\mathcal{L}_{pose}(\hat{R}_{\sigma(i)}, \hat{t}_{\sigma(i)}, R_i, t_i) = \mathcal{L}_R(\hat{R}_{\sigma(i)}, R_i) + ||\hat{t}_{\sigma(i)} - t_i||, \tag{4.17}$$

where the translation loss computes the distance between the predicted translation $\hat{t}_{\sigma(i)}$ and the ground truth translation t_i. The first option for rotation loss $\mathcal{L}_R(\hat{R}_{\sigma(i)}, R_i)$ is to measure the average distance between each point on the estimated model pose and its corresponding point on the ground truth model pose which can be defined as:

$$\mathcal{L}_R(\hat{R}_{\sigma(i)}, R_i) = \frac{1}{|\mathcal{M}_i|} \sum_{\mathrm{x} \in \mathcal{M}_i} ||\hat{R}_{\sigma(i)}\mathrm{x} - R_i\mathrm{x}||. \tag{4.18}$$

Moreover, similar to the coupled pose loss, we can utilize symmetric aware loss introduced in (Xiang et al. 2017):

$$\mathcal{L}_{R,sym}(\hat{R}_{\sigma(i)}, R_i) = \begin{cases} \frac{1}{|\mathcal{M}_i|} \sum\limits_{\mathrm{x}_1 \in \mathcal{M}_i} \min\limits_{\mathrm{x}_2 \in \mathcal{M}_i} ||\hat{R}_{\sigma(i)}\mathrm{x}_2 - R_i\mathrm{x}_1|| \text{ if sym,} \\ \frac{1}{|\mathcal{M}_i|} \sum\limits_{\mathrm{x} \in \mathcal{M}_i} ||\hat{R}_{\sigma(i)}\mathrm{x} - R_i\mathrm{x}|| \text{ otherwise.} \end{cases} \tag{4.19}$$

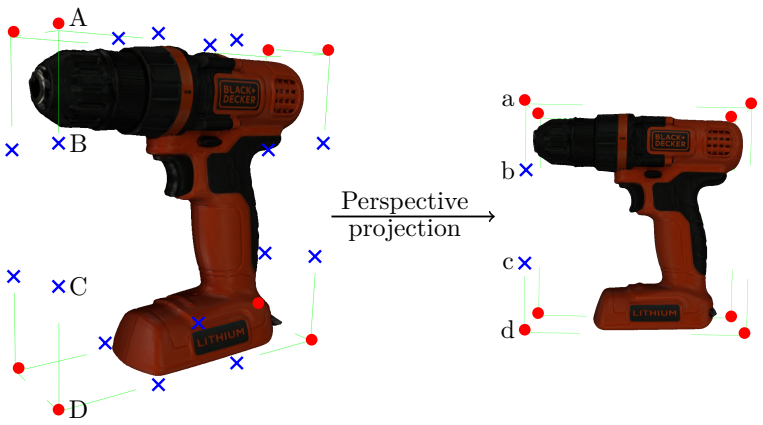

Figure 4.6: The representation of Interpolated Bounding Box (IBB) keypoints. The corners of the bounding box and the intermediate keypoints are indicated with red dots and blue crosses, respectively.

The symmetric aware rotation loss is minimized when the estimated and ground truth 3D models match each other. As a result, rotations that are equivalent due to the 3D shape symmetry of the object are not penalized (Xiang et al. 2017). The other option for rotation loss is to compute the angular distance as follows:

$$\mathcal{L}_R(\hat{R}_{\sigma(i)}, R_i) = \frac{1}{2}\ (\text{tr}(\hat{R}_{\sigma(i)}^T R_i) - 1), \tag{4.20}$$

where the tr(.) is the trace operation. The angular distance is limited to $[-1, 1]$ for numerical stability and then normalized to $[0, 1]$.

It has to mention that to compute the variants of the pose loss provided in this section, we can also use the ℓ_1 distance.

4.5 Keypoints-based Method

Formerly, we proposed the method that directly performs multi-object 6D pose estimation, which we call T6D-Direct. To further improve the pose estimation accuracy, we extend our direct approach to utilize keypoints as 2D projected sparse correspondences. In this section, we explain the candidates for keypoints representation and the methods to recover the 6D object pose from the predicted keypoints. We finalize the section with describing the Hungarian loss of our Keypoints-based approach.

Figure 4.7: Handpicked keypoints, which are objects of the YCB-Video dataset with keypoints manually defined and their interconnections. The figure is taken from (Zappel, Bultmann, and Behnke 2021).

4.5.1 Keypoints Representation

The recent approaches utilize different keypoints representations: *3D bounding box corners*, an obvious choice for selecting 3D keypoints is the 8 corners of the 3D bounding box (Oberweger, Rad, and Lepetit 2018; Tekin, Sudipta N. Sinha, and Fua 2018b). *Farthest Point Sampling (FPS) algorithm* utilized in PVNet (Peng et al. 2019). Since the PVNet model generates keypoints using the vectors that start at the object pixels, the longer distance between the keypoints representation and the object pixels results in larger localization errors. Therefore, predicting the projections of the 3D bounding box corners into the image plane, which are far away from the object pixels in the image, is not an appropriate choice. The authors instead exploit the FPS algorithm to automatically sample 8 keypoints on the surface of the object meshes, which are also spread out on the object to help the PnP algorithm find a more stable solution. The FPS algorithm starts with the object center as the initial keypoint set. Afterward, it repeatedly searches for a point on the object surface that is farthest to the current keypoint set and adds it to the set until reaching the target size of the keypoint set—the center point is not part of the final keypoint set.

Handpicked Keypoints, Zappel, Bultmann, and Behnke (2021) manually define 8 keypoints specific for each object. To facilitate the CNN-based detection, they select keypoints that represent the object contour and locate on easy-to-find spots of the object geometry and texture. The selected keypoints form a bounding box (if applicable) as shown in Figure 4.7. The authors show that the FPS method has inferior performance. The reason is that the FPS keypoints are less intuitively placed; therefore, harder to infer especially for multi-object 6D pose estimation using a single model due to geometrical differences between objects.

Interpolated Bounding Boxes (IBBs), S. Li et al. (2021) define the 3D representation of an object as sparse Interpolated Bounding Box (IBB) depicted in

Figure 4.6. Considering the 3D bounding box of an object with 8 keypoints and 12 lines as $\{\mathbf{l}_i\}_{i=1}^{12}$, where each line is represented with start and end keypoints, $\mathbf{l}_i = [k_i^s, k_i^e]^T$, where k is a 3-dimensional vector representing the keypoint's location in the world coordinate system. We can derive n more keypoints from each line given a pre-defined interpolation matrix $M \in \mathbb{R}^{n\times 2}$ as follows:

$$\begin{bmatrix} k_i^1 \\ k_i^2 \\ \dots \\ k_i^n \end{bmatrix} = M \begin{bmatrix} k_i^s \\ k_i^e \end{bmatrix} = \begin{bmatrix} m_1 & 1-m_1 \\ m_2 & 1-m_2 \\ \dots & \dots \\ m_n & 1-m_n \end{bmatrix} \begin{bmatrix} k_i^s \\ k_i^e \end{bmatrix} \tag{4.21}$$

Therefore, the 8 keypoints of the 3D bounding box and $12n$ intermediate keypoints located on 12 lines form a set of $8 + 12n$ keypoints. Although larger n provides denser correspondences, it increases the complexity; thus, n is set to 2 and $M = \begin{bmatrix} \frac{3}{4} & \frac{1}{4} \\ \frac{1}{4} & \frac{3}{4} \end{bmatrix}$ in practice. This representation allows the authors to take into account the property that the cross-ratio of every four collinear points is preserved during perspective projection (Hartley and Zisserman 2004), i.e., the cross-ratio of the keypoints A, B, C, and D in 3D remains the same after perspective projection in 2D as illustrated in Figure 4.6. The cross-ratio consistency is enforced by an additional component in the loss function that the model learns to minimize during training called the cross-ratio loss function, which will be explained in Section 4.5.4. We opt for the IBB keypoints representation since, unlike other methods, it makes sure geometrical representation is consistent across all object classes due to preserving the geometrical features via the cross-ratio property. We examine the efficiency of the mentioned keypoints representations in the following chapter.

4.5.2 Prediction Heads

For each object query, we use FFNs to estimate the class probability, bounding box, and Interpolated Bounding Box (IBB) keypoints independently. Except for the class probability, we normalize the predictions by the size of the input image.

4.5.3 Recover 6D Object Pose

After establishing the sparse 2D-3D correspondences via using IBB keypoints, we need to use a variant of the PnP-RANSAC algorithm to estimate the components of the 6D pose that can be rotation and/or translation. Between the existing algorithms such as EPnP (Lepetit, Moreno-Noguer, and Fua 2009) and learnable

PnP algorithms (B. Chen et al. 2020; S. Li et al. 2021; G. Wang et al. 2021), we utilize the RANSAC-based EPnP (Lepetit, Moreno-Noguer, and Fua 2009) algorithm from OpenCV (Fischler and Bolles 1981) as our first choice. EPnP presents a non-iterative method with $\mathcal{O}(n)$ complexity for all $n \geq 4$ by expressing the n 3D points as a weighted sum of four points, called virtual control points and then estimating the 2D correspondences of these points.

As declared in (Z. Li, G. Wang, and Ji 2019), the factors that impact rotation and translation components are different, e.g., the rotation is highly affected by the object's appearance in a given image. In contrast, the translation is more vulnerable to the size and location of the object in the image. Therefore, we decide to estimate rotation and translation separately. RANSAC-based EPnP algorithm is utilized for the rotation estimation, while the translation is directly estimated using a prediction head, where it estimates t_z in meter and the 2D location of projected 3D object's centroid $[c_x, c_y]^T$ normalized by the input image dimension.

The EPnP-RANSAC algorithm suffers from two main disadvantages of being time-consuming due to the iterative RANSAC and non-differentiable. Therefore, to have a real-time module that is also differentiable, inspired by the learnable PnP algorithms, we introduce a fully connected module called Rotation Estimation (RotEst) to predict the rotation given the 2D keypoints.

4.5.3.1 Rotation Estimation Module

Given a set of regressed keypoints, we use the fully connected Rotation Estimation (RotEst) module to estimate the object rotation. For each object prediction, the RotEst module takes 64-dimensional input corresponding to 32 keypoints coordinates and predicts the object rotation represented as the 6D continuous representation as explained in Section 4.1.1. The RotEst module consists of six fully connected layers with hidden dimension 1024 and dropout of 0.5.

4.5.4 Set Prediction Loss

As described in Section 4.4.1, for the matching cost, we have the option of using only the bounding box predictions and class probabilities ($\mathcal{L}_{match\_object}$) or adding the pose predictions as well ($\mathcal{L}_{match\_pose}$). In the case of the Keypoints-based method, we can also consider the keypoints predictions in the matching cost, called this variant $\mathcal{L}_{match\_keypoints}$:

$$\mathcal{L}_{match\_keypoints}(\hat{y}_{\sigma(i)}, y_i) = \mathcal{L}_{match\_pose}(\hat{y}_{\sigma(i)}, y_i) + \mathcal{L}_{keypoints}(\hat{K}_{\sigma(i)}, K_i), \quad (4.22)$$

where $\mathcal{L}_{keypoints}$ is the ℓ_1 loss between the predicted 2D keypoints $\hat{K}_{\sigma(i)}$ and ground truth 2D keypoints K_i.

Furthermore, for the Keypoints-based method, the Hungarian loss presented in Section 4.4.2 enjoys an additional term related to keypoints loss as provided in Equation 4.23. Similar to the T6D-Direct method, except for the class probability loss, other terms of the Hungarian loss are normalized by the number of objects in each batch.

$$\mathcal{L}_{Hungarian}(\hat{y}, y) = \sum_{i}^{N}[-\log\hat{p}_{\hat{\sigma}(i)}(c_i) + \mathbb{1}_{c_i\neq\emptyset}\mathcal{L}_{box}(\hat{b}_{\hat{\sigma}(i)}, b_i)+ \\ \mathbb{1}_{c_i\neq\emptyset}\lambda_{pose}\mathcal{L}_{pose}(\hat{R}_{\hat{\sigma}(i)}, \hat{t}_{\hat{\sigma}(i)}, R_i, t_i) + \mathbb{1}_{c_i\neq\emptyset}\mathcal{L}_{keypoints}(\hat{K}_{\hat{\sigma}(i)}, K_i)]. \quad (4.23)$$

It has to mention that when we estimate the components of 6d pose by RANSAC-based EPnP algorithm, i.e., rotation and/or translation, the corresponding loss will be eliminated from the Hungarian loss.

Keypoints Loss Having γ and δ as hyperparameters, our keypoints loss can be represented as:

$$\mathcal{L}_{keypoints}(\hat{K}_{\hat{\sigma}(i)}, K_i) = \gamma||\hat{K}_{\hat{\sigma}(i)} - K_i||_1 + \delta\mathcal{L}_{\mathcal{CR}}. \quad (4.24)$$

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 4.25 to enforce the cross-ratio consistency in the keypoints loss as proposed by S. Li et al. (2021). 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.

$$\mathcal{L}_{\mathcal{CR}} = Smooth\ell_1(\mathrm{CR}^2 - \frac{||c-a||^2||d-b||^2}{||c-b||^2||d-a||^2}), \quad (4.25)$$

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 ground truth cross-ratio CR is defined as:

$$\mathrm{CR} = \frac{||C-A||\ ||D-B||}{||C-B||\ ||D-A||} = \frac{4}{3}. \quad (4.26)$$

5 Experiments

The main goal of this chapter is to provide the experiments conducted to assess the performance of the proposed T6D-Direct and Keypoints-based methods. Therefore, the utilized dataset for the evaluation is described in Section 5.1, followed by the explanation of baselines and metrics. Furthermore, the quantitative and qualitative results are presented in Section 5.4. The chapter will be finalized by the inference time analysis.

5.1 Dataset

We evaluate the performance of our models on the single-instance multi-object 6D pose dataset YCB-Video (YCB-V) created by PoseCNN (Xiang et al. 2017). This dataset is generated by capturing video sequences from cluttered indoor scenes of a total of 21 objects placed in tabletop configuration (see Figures 5.1 and 5.2). These 21 objects are a subset of the YCB objects (Calli et al. 2015), selected due to their high-quality 3D models and good visibility in depth (Xiang et al. 2017). YCB-V is a challenging dataset for benchmarking 6D object pose estimation methods. The reason is that the dataset includes textureless and symmetric objects, where the objects exhibit varying geometric shapes, reflectance properties, and very strong occlusions. Moreover, there are multiple objects in each image. It is worth mentioning that the symmetric objects of the YCB-V dataset are "bowl", "wood block", "large clamp", "extra large clamp", and "foam brick".

YCB-V provides bounding box, segmentation masks, and 6D pose annotation for 133,936 RGB-D images, where the resolution of each image is 640×480. Since our proposed models are RGB-based, we do not use the provided depth information. YCB-V also provides high-quality meshes for all 21 objects. 3D mesh points of the objects are used in computing the evaluation metrics, which will be discussed in Section 5.3.

There are 92 video sequences in total, out of which 12 sequences are held out for the test set with 20,738 images, and the rest are used for training. The final test set is the subset of 2,949 key frames from 12 test scenes, and we employ the BOP YCB-V test set, which will be explained in the following for the validation

set. Additionally, we utilize the 80K synthetic data, generated by PoseCNN via randomly placing objects in a scene, during training and add the backgrounds to these synthetic images randomly from PASCAL VOC 2012 (M. Everingham et al. n.d.) images. The statistic details of YCB-V and its synthetic data are as follows:

Table 5.1: Statistics of the YCB-V dataset and its synthetic data.

Features	YCB-V	Synthetic
#Object Classes	21	21
Min Object Count	3	5
Max Object Count	9	8
Mean Object Count	4.58	6.49
#Images	133,936	80,000
Resolution	640×480	640×480

Recently, the BOP challenge on 6DoF pose estimation (Hodaň, Sundermeyer, et al. 2020) has captured more attention. Generally, the goal of BOP is to introduce a benchmark for 6D pose estimation via proposing datasets with a unified format and a standard evaluation methodology (Hodan et al. 2018). YCB-V is also one of the BOP challenge core datasets; however, the BOP variant of YCB-V[1] has two main differences compared to the original YCB-V dataset. First, for the test set, only a subset of 75 images, which have higher-quality ground truth poses, are manually selected from each of the 12 test scenes resulting in 900 images. Second, the original 3D object meshes are converted from meters to millimeters, and the centers of their 3D bounding boxes are aligned with the origin of the model coordinate system and the ground truth annotations are converted correspondingly. Moreover, apart from the original YCB-V symmetric objects, “master chef can” and “large marker” are also considered symmetric.

Furthermore, the BOP challenge 2020 introduced 50K synthetic training images for each BOP dataset, including YCB-V. These synthetic images are generated by the light-weight Physically-Based Renderer (PBR) tool proposed by the BOP challenge 2020. The idea behind preparing this kind of photorealistic rendering is to help reduce the synthetic-to-real domain gap in object detection and pose estimation. However, we do not use the BOP synthetic data in our experiments for fairness.

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

Figure 5.1: The subset of 21 YCB objects selected for the YCB-V dataset. The figure is taken from (Xiang et al. 2017).

Figure 5.2: An instance scene from the YCB-V dataset.

5.2 Baselines

The proposed architectures in this thesis are compared with all three categories of the state-of-the-art 6D pose estimation approaches based on RGB images. We utilize PoseCNN, CDPNv2, GDR-Net as direct baselines, Oberweger, PVNet, Seg-Driven, Pix2Pose, EPOS, and Single-Stage as indirect baselines, and the refinement-based methods are DeepIM and CosyPose.

- *PoseCNN (Xiang et al. 2017):* the extracted multi-scale feature maps from the backbone network are embedded into low-dimensional features to be the model input. The model then conducts semantic labeling, 3D translation estimation by decoupling translation representation, and quaternion regression to estimate the 6D pose.

- *CDPNv2 (Z. Li, G. Wang, and Ji 2019):* Coordinates-based Disentangled Pose Network, utilizes a lightweight detector for detecting all objects and then zooms in on the target object via the proposed Dynamic Zoom In (DZI). The rotation is estimated by a PnP-RANSAC algorithm from 2D-3D correspondences, which are extracted using confidence and coordinates maps. The translation is estimated based on the local image patches. For evaluation, we consider the modified CDPN proposed for the BOP challenge 2020. The authors make the following adjustments to the original CDPN: they substitute the lightweight detector with a more powerful one. The color augmentation like (Sundermeyer et al. 2018b) and the truncation domain randomization proposed in (Z. Li, Hu, et al. 2020) are exploited to

improve the robustness of the model to occlusion. Finally, in addition to modifying the original structure, they use ResNet-34 instead of ResNet-18 for the backbone network.

- *GDR-Net (G. Wang et al. 2021):* Geometry-guided Direct Regression Network, exploits the intermediate geometric features organized as image-like 2D patches which enables the model to recover the 6D pose directly by using the proposed 2D convolutional Patch-PnP.

- *Oberweger (Oberweger, Rad, and Lepetit 2018):* using patches from the image that is centered on the target object, their approach predict the 2D heatmaps corresponding to the corners of the 3D bounding box for each image patch. The predicted heatmaps from patches are then aggregated to extract the global maxima for each heatmap. Finally, the object pose is recovered from the heatmaps using a PnP algorithm with RANSAC.

- *PVNet (Peng et al. 2019):* Pixel-wise Voting Network, estimates unit vectors that represent pixel-wise directions of the object pointing to the keypoints. These unit vectors then vote for the keypoint locations based on RANSAC.

- *SegDriven (Hu, Hugonot, et al. 2019):* Segmentation-Driven 6D pose estimation, consists of two main parts: object segmentation to predict the label of the object observed at each grid location, and 2D projections regression to estimate the 2D keypoint locations for that object. Finally, 6D pose is inferred from the most reliable 2D projections using the PnP-RANSAC strategy.

- *Pix2Pose (Park, Patten, and Vincze 2019):* is an auto-encoder architecture with Generative Adversarial Network (GAN) which takes the cropped image of a target object and predicts pixel-wise 3D coordinates of an object. The proposed method first adjusts the input bounding box and removes unnecessary pixels, i.e., background and uncertain pixels. Then, the refined input from the first stage in which each pixel forms a 2D-3D correspondence is used to estimate object poses using the RANSAC-based PnP algorithm.

- *EPOS (Hodaň, Baráth, and Jiří Matas 2020):* the proposed method represents an object as compact surface fragments to infer 2D-3D correspondences used with a robust and efficient variant of the PnP-RANSAC algorithm to estimate the 6D pose.

- *Single-Stage (Hu, Fua, et al. 2020):* utilizes SegDriven (Hu, Hugonot, et al. 2019) method to extract 2D-3D correspondences, which will be used in

the following parts of the proposed architecture. This model includes a local feature extraction module, a feature aggregation module operating on the 2D projections correspond to each 3D keypoint, and a global inference module consisting of simple fully connected layers for 6D pose estimation as a quaternion and translation.

- *DeepIM (Yi Li et al. 2018):* Deep Iterative Matching network, using an initial estimated pose of an object and the object's 3D model, the method generates the rendered image of the target object. Then, the rendered image along with the input image are fed to the network to predict a relative transformation, applied to refine the input pose. The refined pose can be utilized as the input pose for the next iteration.

- *CosyPose (Labbe et al. 2020):* Consistent multi-view multi-object 6D pose estimation, this refinement-based method consists of three main stages. In the first stage, the initial object candidates in each view is separately estimated. In the second stage, these object candidates are matched across views to recover a single consistent scene. In the third stage, all object and camera poses are globally refined to minimize multi-view reprojection error.

5.3 Evaluation Metrics

We utilize various metrics to evaluate our proposed models in both 3D and 2D. For the 3D evaluation, the average distance metric ADD is employed from (Hinterstoisser et al. 2013) for evaluating the accuracy of non-symmetric objects. As for the symmetric objects, the matching between points is ambiguous for some views; we use the ADD-S metric for such objects. Given the set of 3D model points $\mathcal{M}$, predicted rotation and translation components $\hat{R}$ and $\hat{t}$, and the ground truth 6D pose with rotation and translation components R and t, ADD metric is the average distance between the 3D mesh points transformed according to the ground truth pose and the estimated pose. In contrast, the symmetry aware metric ADD-S is computed utilizing the closest point distance. The ADD(-S) metric combines the mentioned metrics, where ADD-S and ADD are used for symmetric and the rest of objects, respectively. Formally, these metrics can be represented as:

$$\text{ADD}(\hat{R}, \hat{t}, R, t) = \frac{1}{|\mathcal{M}|} \sum_{x \in \mathcal{M}} \|(\hat{R}x + \hat{t}) - (Rx + t)\|, \tag{5.1}$$

$$\text{ADD-S}(\hat{R}, \hat{t}, R, t) = \frac{1}{|\mathcal{M}|} \sum_{x_1 \in \mathcal{M}} \min_{x_2 \in \mathcal{M}} \|(\hat{R}x_2 + \hat{t}) - (Rx_1 + t)\|, \tag{5.2}$$

$$\text{ADD(-S)} = \begin{cases} \text{ADD-S} & \text{symmetric objects,} \\ \text{ADD} & \text{otherwise.} \end{cases} \tag{5.3}$$

For the YCB-V dataset, we follow PoseCNN and report the Area Under the accuracy-threshold Curve (AUC) of ADD, ADD-S, and ADD(-S) metrics. The reason is that using a fixed threshold in computing pose accuracy cannot reveal how a method performs on the incorrect poses with respect to that threshold. Therefore, we aggregate all results for distance thresholds of maximum 0.1 m and measure the AUC for pose evaluation. We also provide the Average Recall (AR) of ADD, ADD-S, and ADD(-S) metrics. Recall can be defined as the fraction of annotated object instances, for which a correct pose is estimated. The AR metrics measure whether the average deviation of the transformed model points from ground truth lies below 10% of the object's diameter ($0.1d$), where d is the largest distance between any pair of model points.

Another commonly used 3D metric is the n°, n cm introduced by Shotton et al. (2013). In this metric, an estimated pose is accepted if its rotation error is less than n° and its translation error is below n cm.

Recently, the evaluation protocol proposed by BOP (Hodaň, Sundermeyer, et al. 2020) has become more popular. Therefore, we also present the results of our proposed models on YCB-V under BOP setup. Based on BOP, the error of an estimated pose $\hat{P}$ with respect to the ground truth pose P in a test image I can be measured by three different metrics. *Visible Surface Discrepancy (VSD)*: to calculate this metric, an object 3D model $\mathcal{M}$ is rendered in the ground truth pose and the estimated pose to produce distance maps D and $\hat{D}$, respectively. A distance map stores at a pixel p the distance from the camera center to the 3D point x_p that projects to p. Following the method proposed in (Hodaň, Jiří Matas, and Obdržálek 2016), the distance maps are then compared with the distance map D_I of the test image I to obtain the visibility masks V and $\hat{V}$, i.e., the sets of pixels where the model $\mathcal{M}$ is visible in the image I. Given a misalignment tolerance τ, the metric is calculated as:

$$\text{VSD}(\hat{D}, \hat{V}, D, V, \tau) = \text{avg}_{p \in \hat{V} \cup V} \begin{cases} 0 & \text{if } p \in \hat{V} \cap V \ \wedge \ |\hat{D}(p) - D(p)| < \tau, \\ 1 & \text{otherwise.} \end{cases} \tag{5.4}$$

According to (Hodan et al. 2018), due to the existence of multiple fits of the visible part of an object surface to the entire object surface in a given image, multiple poses can be existed that are indistinguishable. The visible part is determined through (self-)occlusion and the multiple surface fits are induced by global or partial object symmetries (Hodan et al. 2018). As the VSD metric is only computed

over the visible part of the model surface, the indistinguishable poses are treated as equivalent.

For the next two BOP evaluation metrics, *Maximum Symmetry-Aware Surface Distance (MSSD)* and *Maximum Symmetry-Aware Projection Distance (MSPD)*, we need to compute a set of global symmetry transformations. The set of global symmetry transformations of an object is identified in two steps. Firstly, a set of candidate symmetry transformations $S'_{\mathcal{M}}$ is obtained by calculating the Hausdorff distance between points of object model $\mathcal{M}$ in the canonical and transformed locations. Then, the symmetry transformation S is selected if its corresponding distance lies below $\max(15\text{ mm}, 0.1d)$. Secondly, the final set of symmetry transformations $S_{\mathcal{M}}$ is determined as a subset of $S'_{\mathcal{M}}$ consisting of those symmetry transformations that cannot be resolved by the model texture. Having a set of global symmetry transformations $S_{\mathcal{M}}$ and $V_{\mathcal{M}}$ as a set of mesh vertices of object model $\mathcal{M}$, MSSD is defined as follows:

$$\text{MSSD}(\hat{P}, P, S_{\mathcal{M}}, V_{\mathcal{M}}) = \min_{S \in S_{\mathcal{M}}} \max_{x \in V_{\mathcal{M}}} \|\hat{P}x - PSx\| \tag{5.5}$$

In contrast to the average distance used in the variants of ADD metrics which is dominated by higher-frequency surface parts, the MSSD metric due to the maximum distance, is less dependent on the sampling of model points.

The third BOP metric MSPD is represented as:

$$\text{MSPD}(\hat{P}, P, S_{\mathcal{M}}, V_{\mathcal{M}}) = \min_{S \in S_{\mathcal{M}}} \max_{x \in V_{\mathcal{M}}} \|\,\text{proj}(\hat{P}x) - \text{proj}(PSx)\|, \tag{5.6}$$

where proj(.) is the 2D projection operation with result in pixels. The MSPD metric considers global object symmetries and replaces the average by the maximum distance to decrease the vulnerability to the sampling of model points which demonstrates the superiority of this metric over other 2D metrics.

An estimated pose $\hat{P}$ is considered correct with respect to the ground truth pose P for each BOP metric $\in$ {VSD, MSSD, MSPD} if the metric is less than θ_{metric}, where θ_{metric} is the threshold of correctness specific to each metric. The Average Recall with respect to metric denoted as AR_{metric}, is defined as the average of the recall rates calculated for multiple settings of threshold θ_{metric}, and also for multiple settings of the misalignment tolerance τ in the case of VSD. The choice of the misalignment tolerance τ and the correctness threshold θ_{metric} depends on the target application (Hodaň, Jiří Matas, and Obdržálek 2016). For evaluation of our models under BOP setup, we report AR_{VSD} for the misalignment tolerance τ with the range of $0.05d$:$0.05d$:$0.5d$ and the threshold of correctness θ_{VSD} with the range of 0.05:0.05:0.5. Moreover, AR_{MSSD} is calculated for θ_{MSSD} ranging

in $0.05d$:$0.05d$:$0.5d$, and AR_{MSPD} is computed for θ_{MSPD} ranging from $5r$ to $50r$ with a step of $5r$, where $r = w/640$ and w is the image width in pixels. Finally, the performance of our proposed architectures on YCB-V dataset is also measured by $\mathrm{AR} = \frac{1}{3}(\mathrm{AR}_{VSD} + \mathrm{AR}_{MSSD} + \mathrm{AR}_{MSPD})$.

It has to mention that based on our experiments, we notice that monitoring object detection through bounding box estimations and keypoints predictions play an important role to trace the performance of the proposed models. For both cases, we follow the evaluation metrics used by COCO (T.-Y. Lin et al. 2014), before elaborating on these metrics, we need to explain the calculation of the AP metric. Based on the PASCAL VOC (Mark Everingham et al. 2010) definition, the AP summarises the shape of the precision-recall curve and is defined as the mean precision at a set of equally spaced recall levels, $r \in \{0{:}0.01{:}1\}$ for each object:

$$\mathrm{AP} = \frac{1}{101} \sum_{r \in \{0, 0.01, \ldots, 1\}} p_{\mathrm{interp}}(r). \tag{5.7}$$

The precision at each recall level r is interpolated by taking the maximum precision measured for a method for which the corresponding recall exceeds r:

$$p_{\mathrm{interp}}(r) = \max_{\tilde{r}:\tilde{r} \geq r} p(\tilde{r}), \tag{5.8}$$

where $p(\tilde{r})$ is the measured precision at recall $\tilde{r}$. To follow the utilized evaluation metrics by COCO, we compute the average of AP over multiple thresholds, i.e, IoU values as Equation 5.9 for bounding box evaluation and OKS values provided in Equation 5.10 for keypoint evaluation and then report the mean of averaged APs over objects.

For evaluation of bounding boxes, we utilize the commonly used metric, Intersection over Union (IoU) between the predicted bounding box $\hat{B}$ and the ground truth bounding box B, presented in Equation 5.9. We measure the AP, AP^{50}, AP^{75}, AP^{M}, AP^{L}, and AR metrics. AP is the mean average precision over 10 IoU thresholds = [0.50:0.05:0.95]; however, AP^{50} and AP^{75} are computed at a single IoU threshold of 0.50 and 0.75, respectively. AP^{M} is for medium scale object instances (area $< 32^2$), and AP^{L} is for large scale instances (area $> 96^2$), where the area of an object is measured as the number of pixels in the corresponding segmentation mask. Finally, AR computes the mean of average recall over 10 OKS thresholds.

$$\mathrm{IoU} = \frac{\mathrm{area}(\hat{B} \cap B)}{\mathrm{area}(\hat{B} \cup B)} \tag{5.9}$$

For the keypoints evaluation, we use the Object Keypoint Similarity (OKS) metric from the COCO keypoint dataset. The OKS of an object o between the estimated ($\hat{y}_{o_i}$) and its corresponding ground truth keypoint (y_{o_i}) can be written as follows:

$$\mathrm{OKS}(\hat{y}_o, y_o) = \frac{\sum_i e^{-\frac{\|\hat{y}_{o_i} - y_{o_i}\|_2^2}{2a^2k_i^2}} \delta(v_i > 0)}{\sum_i \delta(v_i > 0)}, \tag{5.10}$$

where k_i is a constant specific to each keypoint, here we set k_i to 0.2 for all keypoints. a is the segment area of the object instance measured in pixels, and v_i is the keypoint visibility flag in the ground truth ($v_i = 0$ for the invisible keypoint). The OKS metric is robust to the number of visible keypoints as it gives equal importance to the object instances with different numbers of visible keypoints. To evaluate the keypoints, we report the following metrics: AP (the mean average precision over 10 OKS thresholds = [0.50:0.05:0.95]), AP^{50} (AP at OKS threshold = 0.50), AP^{75}, AP^{M}, AP^{L}, and AR (the mean of average recall over 10 OKS thresholds).

5.4 Experimental Details

We implemented the proposed methods using PyTorch (Paszke et al. 2017). The models take an RGB input image which is normalized based on the mean and standard deviation of ImageNet dataset (Deng et al. 2009). For all experiments, we use the synthetic data generated by PoseCNN (Xiang et al. 2017) for training our models to avoid overfitting.

For evaluation, we require to use the official YCB-V toolbox[2] provided by PoseCNN in MATLAB. In this thesis, as we opt for the BOP version of the YCB-V dataset, for a fair comparison, the original YCB-V meshes are converted to the BOP variant accordingly, and the 2,949 YCB-V key frames are employed to report the results. We re-implemented the evaluation toolbox in Python and verified our results. In addition, we convert the BOP format to COCO, as it allows us to use the COCO API[3] for object detection and keypoints evaluation, and it supports selecting subsets of object classes as well. Finally, the publicly available BOP toolkit[4] is utilized for evaluation under BOP setup.

[2] `https://github.com/yuxng/YCB_Video_toolbox`
[3] `https://github.com/cocodataset/cocoapi`
[4] `https://github.com/thodan/bop_toolkit`

5.4.1 T6D-Direct

5.4.1.1 Training Details

To tackle the DETR drawback of having a slow convergence (Zhu et al. 2021), we initialize our T6D-Direct model using the provided pretrained weights on the COCO dataset (T.-Y. Lin et al. 2014) by DETR, and then train the complete T6D-Direct model on the YCB-V dataset. After initializing our model with the pretrained weights, there are two possible strategies while training for the pose estimation task. In the first approach, we train the model for both object detection and 6D pose estimation tasks simultaneously; therefore, the total loss function is the Hungarian loss brought in Section 4.4.2. In the second approach, we employ a multi-stage scheme, i.e., first train the model for the object detection task and then only train prediction heads corresponding to 6D pose estimation and freeze the rest of the network. Investigation on these training methods is conducted in the following sections.

5.4.1.2 Hyperparameters

The best hyperparameters are selected as follows. α and β in computing the bounding box loss $\mathcal{L}_{box}$ are set to 2 and 5, respectively, and λ_{pose} related to the pose loss is set to 0.05. The model generates the predicted set with the cardinality of $N = 20$ and is trained using AdamW optimizer with an initial learning rate of $1e^{-4}$ for 78K iterations where the learning rate is decayed to $1e^{-5}$ after 70K iterations. The batch size is 32, and the gradient clipping with a maximal gradient norm of 0.1 is applied.

5.4.1.3 Qualitative Results

In this section, we explore the effect of the mentioned training strategies, different loss functions, and egocentric vs. allocentric rotation representations on the T6D-Direct performance for the YCB-V dataset.

Effectiveness of Training Strategies As discussed in Section 5.4.1.1, there are two training strategies: single-stage and multi-stage. In the multi-stage scheme, we train the Transformer model for object detection and only train the prediction heads for 6D pose estimation, i.e., rotation and translation. In contrast, we train the complete model in one stage in the single-stage scheme. In our experiments, as shown in Table 5.2, multi-stage training (row 2) yields inferior results, although both schemes are pretrained on the COCO dataset. The results demonstrate that

the Transformer model is learning the features specific to the 6D object pose estimation on YCB-V; therefore, COCO fine-tuning mainly helps in faster convergence during training and not in more accurate pose estimations. Thus, we believe that most large-scale image datasets can serve as pretraining data sources.

We also examine the effect of utilizing either $\mathcal{L}_{match\_object}$ as the bipartite matching cost that consists of the class probabilities and bounding boxes or $\mathcal{L}_{match\_pose}$ considering the pose predictions as well. As we can see from Table 5.2, including the pose component (row 3) does not provide any considerable advantage and it is computationally expensive in terms of memory usage. Therefore, we exploit only the class probabilities and bounding boxes in the bipartite matching cost for all further experiments.

Effectiveness of Loss Functions We examine the performance of our T6D-Direct method using either the coupled pose loss, known as Point Matching loss, or the disentangled pose loss which are presented in Section 4.4.2. Based on Table 5.2, exploiting the symmetric aware version of the Point Matching loss with ℓ_1 distance (row 1) results in the best AUC of ADD(-S) metric.

In the case of disentangled pose loss, in which rotation and translation components are supervised individually, we train our model with the disentangled pose loss in which both rotation and translation components use ℓ_1 distance (row 5). As we can observe from Table 5.2, interestingly, the ADD(-S) result of the model trained using the symmetric aware Point Match loss (row 1) is only slightly better than the model trained using the disentangled pose loss with $\mathcal{L}_R$. The model is finally trained by the disentangled pose loss, including the symmetric aware rotation loss ($\mathcal{L}_{R,sym}$) with ℓ_1 distance as well as translation loss with ℓ_1 distance that provides the best ADD(-S) result (row 6). It is worth mentioning that utilizing the symmetric aware version in both loss cases does not make a significant contribution.

Table 5.2: Results of our method with different loss functions and training schemes on YCB-V. The ADD(-S) metric reports the AR of ADD(-S).

Row	Method	ADD(-S)	AUC of ADD(-S)
1	T6D-Direct + Point Matching loss	47.0	**75.6**
2	1 + multi-stage training	20.5	59.1
3	1 + $\mathcal{L}_{match\_pose}$	42.8	71.7
4	1 + allocentric R_{6d}	42.9	74.4
5	T6D-Direct + $\mathcal{L}_R$	45.8	74.4
6	T6D-Direct	**48.7**	74.6

Effectiveness of Rotation Representations Furthermore, we compare the performance of egocentric vs. allocentric rotation representations. The egocentric rotation representation (row 1) performs slightly better than the allocentric representation (row 4). We hypothesize that supplementing RGB images with positional encoding allows the Transformer model to learn spatial features efficiently. Therefore, the allocentric representation does not have any advantage over the egocentric representation.

The results of the previous investigations provided in Table 5.2 demonstrate that training the model with single-stage fashion, in which we train the complete model in one stage, using ℓ_2 symmetric-aware disentangled loss as well as parameterizing rotation with the 6D continuous representation provides the best results. Therefore, the T6D-Direct method with the mentioned settings will be utilized for the following experiments.

Comparison to the State-of-the-art Methods The T6D-Direct is evaluated against state-of-the-art approaches. In terms of approach, our method is comparable to PoseCNN and Oberweger; all of them are direct methods, whereas DeepIM is a refinement-based approach.

In Table 5.3, we present the per object AUC of ADD and ADD-S results of T6D-Direct on the YCB-V dataset. For a fair comparison, we follow the object symmetry definition and evaluation procedure proposed by YCB-V. In terms of the AUC of ADD-S metric, T6D-Direct outperforms PoseCNN and outperforms all the competing approaches, even the refinement-based method DeepIM. The pose refinement approach achieves the best AUC of ADD result. However, in terms of approach, refinement-based methods are orthogonal to the other methods, as they 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 that enables faster inference time.

5.4.1.4 Quantitative Results

To further understand the behavior of the mentioned training schemes, i.e., single-stage and multi-stage, we demonstrate the decoder attention maps for the object queries corresponding to the predictions. In Figure 5.4, the top row shows the object detections predicted by bounding boxes. The middle and bottom rows depict the attention maps from the complete and partial trained models, respectively, corresponding to the predictions in the top row. The partial trained model has higher activations along the object boundaries. These activations are the result of training the partial model only on the object detection task. In this case, since

Table 5.3: Comparison of T6D-Direct per object results with state-of-the-art methods on YCB-V. * denotes the symmetric objects. The best results are shown in bold.

Method	PoseCNN		Oberweger		T6D-Direct (Ours)		DeepIM	
Metric	AUC of ADD	AUC of ADD-S	AUC of ADD	AUC of ADD-S	AUC of ADD	AUC of ADD-S	AUC of ADD	AUC of ADD-S
master_chef_can	50.2	83.9	81.9	91.4	61.5	91.9	65.2	87.8
cracker_box	53.1	76.9	83.6	90.0	76.3	86.6	82.6	89.8
sugar_box	68.4	84.2	82.1	89.8	81.8	90.3	89.7	93.8
tomato_soup_can	66.2	81.0	79.8	89.5	72.0	88.9	81.4	90.1
mustard_bottle	81.0	90.4	91.5	95.0	85.7	94.7	90.3	94.4
tuna_fish_can	70.7	88.0	48.7	71.7	59.0	92.2	85.4	94.5
pudding_box	62.7	79.1	90.2	94.1	72.7	85.1	84.9	91.8
gelatin_box	75.2	87.2	93.7	95.9	74.4	86.9	87.7	91.6
potted_meat_can	59.5	78.5	79.1	90.0	67.8	83.5	70.0	78.2
banana	72.3	86.0	51.7	67.8	87.4	93.8	83.3	92.0
pitcher_base	53.3	77.0	69.4	85.0	84.5	92.3	88.7	93.7
bleach_cleanser	50.3	71.6	76.2	85.5	65.0	83.0	75.9	86.8
bowl*	3.3	69.6	3.6	78.1	5.1	91.6	41.5	77.8
mug	58.5	78.2	53.9	75.8	72.1	89.8	70.3	86.1
power_drill	55.3	72.7	82.9	90.8	77.7	88.8	90.7	94.6
wood_block*	26.6	64.3	0.0	57.0	52.3	90.7	26.2	60.1
scissors	35.8	56.9	65.3	79.6	59.7	83.0	45.5	61.8
large_marker	58.3	71.7	56.5	70.2	63.9	74.9	68.1	77.5
large_clamp*	24.6	50.2	57.2	73.1	26.6	78.3	45.5	72.1
extra_large_clamp*	16.1	44.1	23.6	54.6	29.0	54.7	29.1	70.0
foam_brick*	40.2	88.0	32.1	88.9	72.0	89.9	70.5	83.0
MEAN	53.7	75.8	66.2	82.4	64.1	**86.2**	**70.1**	84.2

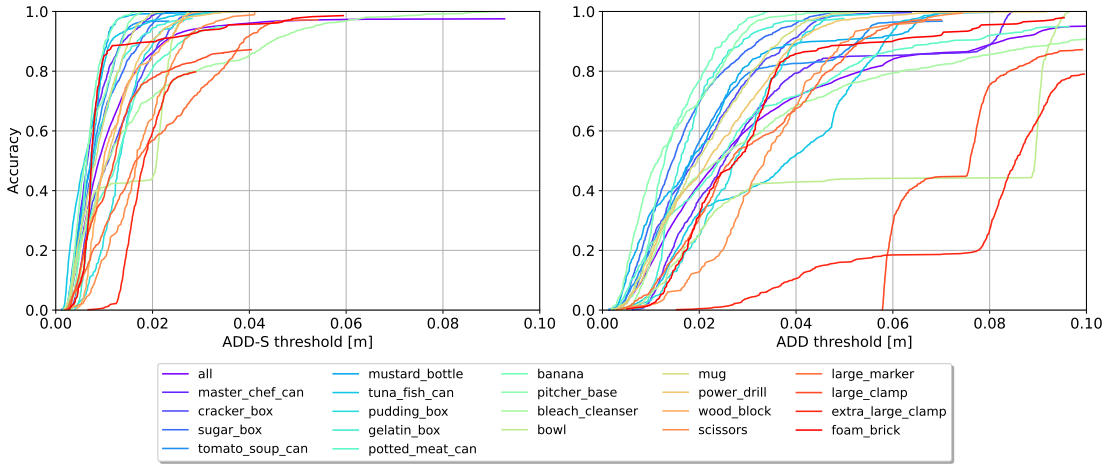

Figure 5.3: The per-object area under the accuracy-threshold curve (AUC) of ADD-S and ADD for distance threshold of maximum 0.1 m related to our T6D-Direct method on the YCB-V dataset.

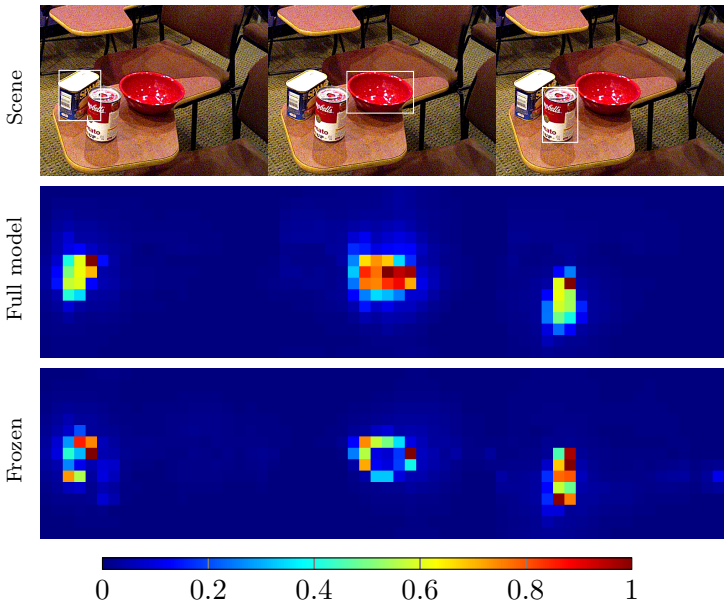

Figure 5.4: Object detections predicted by bounding boxes in a given image (first row) and decoder attention maps for the object queries (second and third rows). In second row, single-stage training is utilized, and multi-stage training is used in third row. Attention maps are visualized using the jet color map (shown above for reference).

we freeze the transformer model and train only the prediction heads of 6D pose estimation, the prediction heads have to rely on the features already learned. However, the complete trained model has denser activations than the partial-trained model, and the activations are spread over the whole object and not just the object boundaries. Thus, training the complete enables the model to learn more suitable features for 6D pose estimation than object detection.

In Figure 5.5, we illustrate the self-attention maps for four pixels belonging to four objects in the image. As we can see the encoder is able to separate individual instances. Figure 5.6 also shows some qualitative results of the T6D-Direct predictions compared to PoseCNN (Xiang et al. 2017) on the YCB-V dataset.

5.4.2 Keypoints-based

In this section, we evaluate the performance of our Keypoints-based approach. We utilize the 6D continuous representation for rotation like the T6D-Direct method and represent translation via decoupling estimations of object distance from camera and the 2D object center in the image plane.

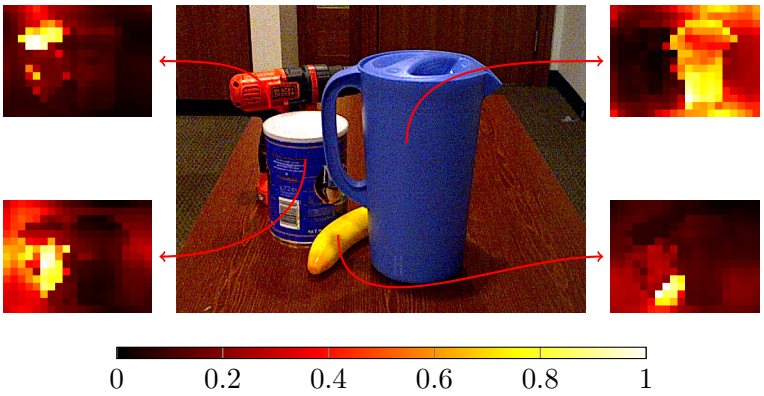

Figure 5.5: Encoder self-attention. We visualize the self-attention maps for four pixels belonging to four objects in the image.

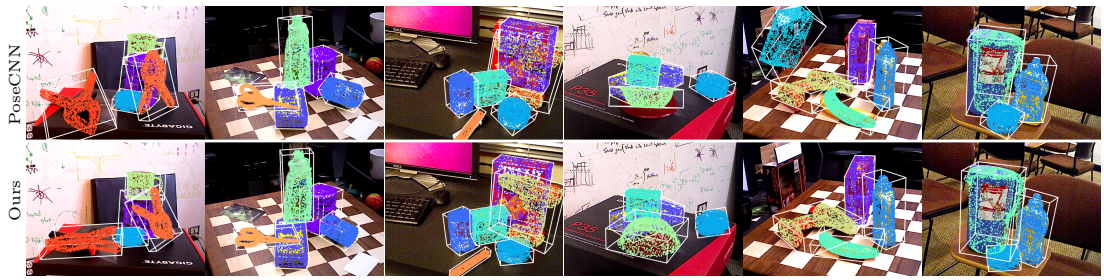

Figure 5.6: Qualitative examples from the YCB-V Dataset. Top row: PoseCNN (Xiang et al. 2017). Bottom row: our predictions.

5.4.2.1 Training Details

We train the Keypoints-based model with AdamW optimizer using the ℓ_1 symmetric-aware disentangle pose loss. The best hyperparameters are adjusted as follows. The γ and δ in $\mathcal{L}_{keypoints}$ are set to 10 and 1, respectively. λ_{pose} related to the pose loss is set to 0.02. Other hyperparameters have the same settings as the T6D-Direct approach. The model is trained with the initial learning rate of $2e^{-4}$ for 335K iterations and the learning rate is decayed to $2e^{-5}$ after 271K iterations. We also use color augmentation for training the Keypoints-based model that has the following procedure: (1) we convert 2% of all pixels to black on the 50% of image size; (2) apply Gaussian blur to the images; (3) add random values between -25 and 25 to all image, in 50% of all images the added value is different for each channel while in the other half, the added value is the same for all channels; (4) reverse the value of all pixels for all images; (5) for each channel, multiply 50% of images by a value sampled uniformly from the interval $[0.6, 1.4]$; (6) multiply images by a random value sampled uniformly from the interval $[0.6, 1.4]$ that makes

some images darker and others brighter, and finally modify the contrast of images with linear contrast function.

5.4.2.2 Qualitative Results

We first investigate the performance of our model with different 3D keypoints representations explained in Section 4.5.1. Then, we evaluate the effect of the proposed fully connected Rotation Estimation (RotEst) module.

Table 5.4: The effectiveness of keypoints representations and the RotEst module evaluated on YCB-V.

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 + heads for R and t	**65.0**	**82.6**

Effectiveness of 3D Keypoints Representations We compare different keypoints representations, i.e., keypoints sampled using the Farthest Point Sampling (FPS) algorithm (Peng et al. 2019), handpicked keypoints representation (Zappel, Bultmann, and Behnke 2021), 8 corners of the 3D bounding box (BB), and the Interpolated Bounding Box keypoints (IBB) proposed by S. Li et al. (2021). For evaluating the ability of the model to estimate keypoints representations, in the experiments, we use the OpenCV implementation of the RANSAC-based EPnP algorithm with the same parameters to recover the 6D object pose from the predicted keypoints. Except for the case of IBB representation, the proposed model is trained using only the ℓ_1 loss. The BB representation provides results slightly better than the handpicked keypoints. As the IBB representation is an extension of BB, and due to utilizing the cross-ratio loss, we expect IBB provides the best results.

Based on our experiments as provided in Tables 5.4 and 5.5, using the model in conjunction with the RANSAC-based EPnP solver, the FPS performs worse than all other representations. The reason is that the locations of FPS keypoints are less intuitive, making them more difficult to predict, especially for our proposed model that needs to deal with all objects in the YCB-V dataset. In contrast, the IBB keypoints representation yields the best performance. We conjecture that as the cross-ratio loss based on the prior geometric knowledge preserves the keypoints

geometrically, this representation is the appropriate choice for our method where a single model is trained for all objects.

Effectiveness of Rotation Estimation Module In contrast to the standard approach of estimating the 2D keypoints with the RANSAC-based PnP solver, which is not trivially differentiable, we exploit the proposed learnable Rotation Estimation (RotEst) module to estimate the object rotation from a set of predicted keypoints. After deciding on the keypoints representation, i.e., IBB keypoints, we compare the performance of the learnable feed-forward rotation and translation estimators against the analytical RANSAC-based EPnP algorithm. Based on Table 5.4, interestingly, if we only estimate the rotation from the EPnP-RANSAC result and directly regress the translation, the accuracy improves significantly. The reason is that in the IBB keypoints representation, due to geometrical enforcement by cross-ratio, a small error in prediction of the 2D projected correspondences of IBB results in dropping the translation performance considerably in contrast to rotation which tends to be less vulnerable to the noise. Therefore, it is not beneficial to utilize the IBB representation for translation. We can observe that the RotEst module performs slightly better than using EPnP rotation and direct translation estimation. However, it is worth noting that the proposed module and the translation estimator are MLPs and thus do not add any overhead to running time, unlike the EPnP-RANSAC algorithm. This advantage enables our Keypoints-based model to perform inference in real-time.

Table 5.5: Ablation study on keypoints representations. The best results are shown in bold.

Method	AP	AP^{50}	AP^{75}	AP^{M}	AP^{L}	AR	AR^{50}	AR^{75}	AR^{M}	AR^{L}
FPS	83.7	93.0	86.3	78.1	**88.0**	87.8	**95.1**	90.2	83.0	89.9
Handpicked	83.5	**92.9**	86.4	79.5	86.6	87.7	**95.1**	89.9	85.6	88.9
IBB Keypoints	**86.2**	**92.9**	**88.3**	**89.1**	86.0	**90.2**	95.0	**91.8**	**91.2**	**90.0**

Object Detection Evaluation As the object detection performance of the model directly impacts the 6D pose estimation results, we examine the object detection performance of our T6D-Direct and Keypoints-based methods in Table 5.6. It has to mention that we utilize Generalized IoU (GIoU) introduced by Rezatofighi et al. (2019) for the bounding box loss. As we can see, the Keypoints-based method provides superior results for all metrics.

Comparison to the State-of-the-art Methods In Table 5.7, we provide the results of the AUC of ADD-S and ADD(-S) for each class. Except for DeepIM as a

Table 5.6: Objection detection results of the proposed approaches. The best results are shown in bold.

Method	AP	AP^{50}	AP^{75}	AP^{M}	AP^{L}	AR	AR^{50}	AR^{75}	AR^{M}	AR^{L}
T6D-Direct	79.8	98.0	92.1	69.5	82.9	83.7	83.8	83.8	70.6	88.4
Keypoints-based	**84.5**	**99.4**	**97.8**	**80.9**	**86.6**	**88.2**	**88.3**	**88.3**	**82.0**	**90.7**

refinement-based method and PVNet as an indirect approach, other methods estimate the 6D pose directly. The results show that our Keypoints-based method outperforms all of the competing approaches. The area under the accuracy-threshold curves using both ADD-S and ADD metrics are also illustrated in Figure 5.7. We vary the threshold to the maximum value 10 cm for the average distance and then compute the pose accuracy.

Table 5.7: Comparison of Keypoints-based with the state-of-the-art methods on YCB-V. The P.E. clarifies that one model is trained per object (N) or a single model is trained for all objects (1). The symmetric objects are denoted by *, and the best results are shown in bold.

Method	PoseCNN		PVNet	GDR-Net		T6D-Direct		Keypoints-based		DeepIM	
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

We present AR of ADD(-S) and AUC of ADD-S/ADD(-S) of our proposed approaches compared to the state-of-the-art methods in Table 5.8. As we can see, our Keypoints-based method achieves the state-of-the-art results among the pose estimators in terms of AUC of ADD-S and AR of ADD(-S) $0.1d$ in which the error is computed based on the diameter of an object. Therefore, it shows that our model's 6D object pose estimations are more accurate. Note that the pose

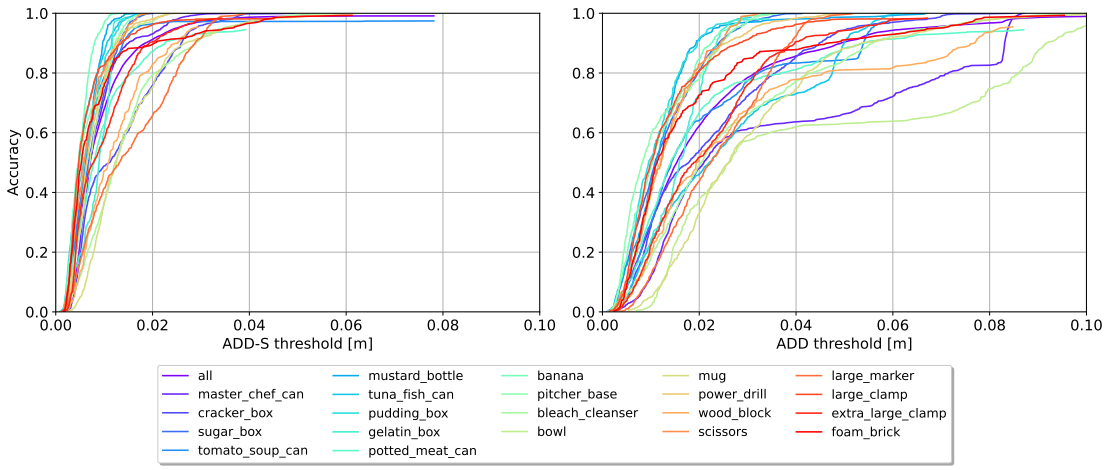

Figure 5.7: The per-object area under the accuracy-threshold curve (AUC) of ADD-S and ADD for distance threshold of maximum 0.1 m related to the Keypoints-based method on the YCB-V dataset.

refinement approach CosyPose achieves the best result in terms of the AUC of ADD(-S) metric; however, it requires training an additional pose refinement model. It is worth mentioning that our method can be combined with an additional refiner like CosyPose to improve the results.

Table 5.8: Comparison with the state-of-the-art methods on YCB-V. * indicates that the method is refinement-based. The best results are shown in bold.

Method	ADD(-S)	AUC of ADD-S	AUC of ADD(-S)
PoseCNN	21.3	75.9	61.3
SegDriven	39.0	-	-
Single-Stage	53.9	-	-
GDR-Net	49.1	89.1	80.2
T6D-Direct (Ours)	48.7	86.2	74.6
Keypoints-based (Ours)	**65.0**	**90.1**	82.6
CosyPose*	-	89.8	**84.5**

As explained in Section 5.3, we also compare the performance of our proposed methods with the state-of-the-art RGB-based approaches under BOP setup in Table 5.9. Pix2Pose and EPOS are indirect approaches, and CosyPose is a refinement-based method, whereas other approaches directly regress the 6D object pose. The results of the baselines are reported from the BOP challenge leaderboard[5]. We can observe that in terms of the AR metric, our Keypoints-based

[5] `https://bop.felk.cvut.cz/leaderboards`

Table 5.9: Performance comparison of the methods on YCB-V under BOP setup. The time is the average over the averaged inference time of images in the BOP test set. The P.E. indicates one network is trained per object (N) or a single network is trained for all objects (1). * indicates that the method is refinement-based, and the best results are shown in bold.

Method	P.E.	AR_{VSD}	AR_{MSSD}	AR_{MSPD}	AR	Time [s]
Pix2Pose	1	37.2	42.9	57.1	45.7	1.025
EPOS	1	62.6	67.7	78.3	69.6	0.572
CDPNv2	N	39.6	57.0	63.1	53.2	0.143
GDR-Net	1	58.4	67.4	72.6	66.1	0.065
T6D-Direct (Ours)	1	53.9	56.2	53.6	54.6	**0.036**
Keypoints-based (Ours)	1	66.2	72.3	76.4	71.6	0.038
CosyPose*	1	**77.2**	**84.2**	**85.0**	**82.1**	0.241

method outperforms the direct GDR-Net approach. Comparison of the T6D-Direct and Keypoints-based results show the beneficial effect of using keypoints as 2D intermediate correspondences on improving the estimation performance, especially for the rotation component. Furthermore, the results of the inference time confirm the speed superiority of our proposed methods compared to other pose estimators.

5.4.2.3 Quantitative Results

In Figure 5.8, we depict the object bounding boxes and the decoder cross-attention feature maps corresponding to the detected objects. We can observe that the attended regions indicate the spatial position of the objects in the image very well. To further show the performance of our Keypoints-based method for the 6D pose estimation task, we demonstrate the predicted IBB keypoints and estimated object poses on the YCB-V test set in Figure 5.9.

5.5 Inference Time Analysis

In terms of the inference speed, one of the major advantages of our models is to generate a set of N predictions in parallel; therefore, the inference of the models is not dependent on the number of objects in an image. We set N to 20 since having a smaller cardinality of the prediction set results in estimating fewer object queries and facilitates faster inference time. We compute the inference time on an NVIDIA 2080 GPU and Intel 2.20 GHz CPU. Our proposed models run almost similarly at 27 fps. The number of parameters in T6D-Direct is 41.5M, and Keypoints-based has extra 6.8M parameters due to the lightweight RotEst module.

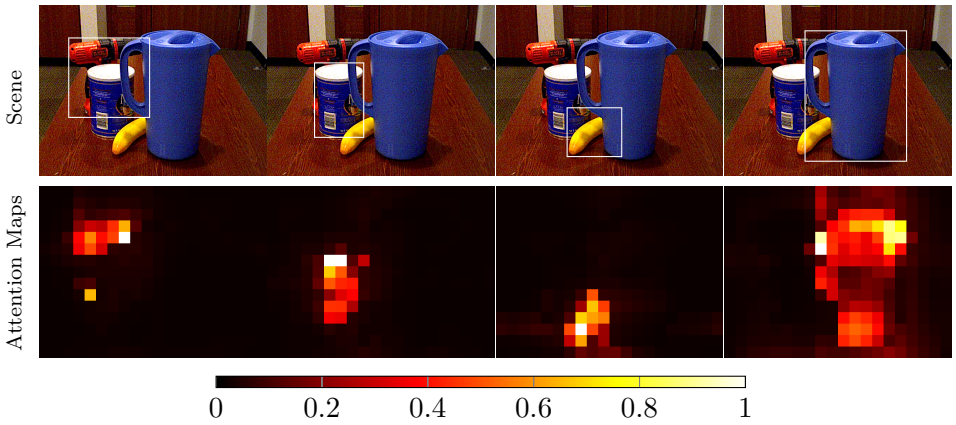

Figure 5.8: Top: Object detections predicted by bounding boxes in a given image. Bottom: Decoder cross-attention maps for the object queries corresponding to the predictions in the first row.

Figure 5.9: Qualitative results on YCB-V test set. The predicted IBB keypoints overlaid on the input images. Ground truth and predicted object poses are visualized as object contours in green and blue colors, respectively.

To demonstrate the superiority of our proposed approaches in terms of real-time performance, we present the AR score vs. inference frame rate under BOP setup for the state-of-the-art methods in Figure 5.10. As we can see, our methods enjoy the fastest inference among the state-of-art-approaches.

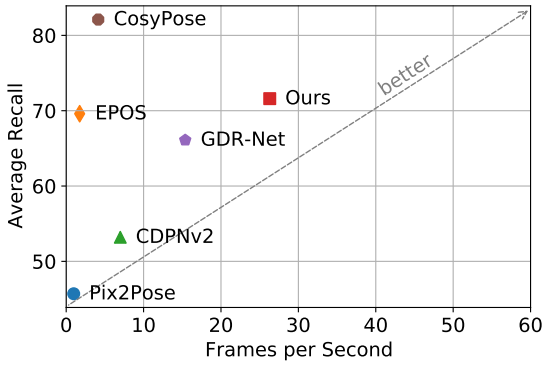

Figure 5.10: Comparison of AR score and inference frame rate (fps) under BOP setup.

6 Conclusion

In this thesis, inspired by DETR (Carion et al. 2020), we model the multi-object monocular 6D pose estimation as a direct set prediction problem and propose two end-to-end differentiable transformer-based methods, T6D-Direct and its extension Keypoints-based. We evaluate the performance of the developed models on a challenging multi-object single-instance 6D pose dataset, YCB-V. Our T6D-Direct approach directly regresses the 3D rotation and 3D translation of the 6D pose, achieving competitive results compared to the state-of-art methods. To further improve the model performance, we extend T6D-Direct by utilizing keypoints as 2D projected sparse correspondences from 3D Interpolated Bounding Box (IBB) for rotation estimation. We employ the fully connected Rotation Estimation (RotEst) module to estimate the object rotation from the predicted keypoints. Our Keypoints-based approach outperforms other competing pose estimators without leveraging intermediate geometric feature maps.

Our proposed approaches in this thesis offer two main advantages. First, these single-stage models, equipped with a non-autoregressive Transformer including a multi-head attention mechanism, enable estimating the 6D pose of all objects in a given image in one forward pass. Second, in contrast to the existing methods, inference time is invariant to the number of objects in an image. The conducted inference time analysis shows that our proposed architectures enjoy the highest frame rate that gives our models the privilege to be utilized for real-world applications.

The main goal of this thesis is to utilize Vision Transformers for 6D object pose estimation. During developing our models, we mainly focused on the 6D object pose estimation problem and explored the potential of Transformers. Therefore, there are several directions remain to be investigated.

For future work, one of the possible extensions to our Keypoints-based model is to examine variants of Transformer such as Swin Transformer (Z. Liu, Y. Lin, et al. 2021), Pyramid Vision Transformer (W. Wang et al. 2021a), Focal Transformer (Yang et al. 2021) that aim to advance the vanilla Transformer by benefiting from CNN inductive biases and hierarchical representation. We expect that using adapted Transformers will increase the accuracy and/or provide lighter models.

Although our experiments showed that adding keypoints improves the results of our T6D-Direct approach, the Keypoints-based model has reached its limitation and requires denser intermediate geometric features to achieve more robust results, as 6D pose estimation is applicable on a wide range of objects with different characteristics. Thus, another venue to be explored is to exploit additional intermediate features for recovering a more accurate 6D object pose since our Keypoints-based method estimates the 6D pose of all objects with various levels of occlusion in a scene using a single model. Moreover, we can consider other geometrical prior knowledge apart from preserving the cross-ratio as suggested by S. Li et al. (2021).

List of Figures

List of Tables

Bibliography

Amini, Arash, Arul Selvam Periyasamy, and Sven Behnke (2021). "T6D-Direct: transformers for multi-object 6D object pose estimation." In: *Dagm german conference on pattern recognition (GCPR)*, pp. 530–544.

Arnab, Anurag, Mostafa Dehghani, Georg Heigold, Chen Sun, Mario Lučić, and Cordelia Schmid (2021). "ViViT: a video vision transformer." In: *Proceedings of the ieee international conference on computer vision (ICCV)*.

Ba, Jimmy Lei, Jamie Ryan Kiros, and Geoffrey E Hinton (2016). "Layer normalization." In: *Arxiv:1607.06450*.

Barath, Daniel and Jiri Matas (2018). "Graph-cut RANSAC." In: *Conference on computer vision and pattern recognition (CVPR)*.

– (2019). "Progressive-x: efficient, anytime, multi-model fitting algorithm." In: *Proceedings of the ieee international conference on computer vision (ICCV)*, pp. 3780–3788.

Brachmann, Eric, Alexander Krull, Frank Michel, Stefan Gumhold, Jamie Shotton, and Carsten Rother (2014). "Learning 6D object pose estimation using 3D object coordinates." In: *Proceedings of the european conference on computer vision (ECCV)*, pp. 536–551.

Brachmann, Eric, Frank Michel, Alexander Krull, Michael Ying Yang, Stefan Gumhold, and carsten Rother (June 2016). "Uncertainty-driven 6D pose estimation of objects and scenes from a single RGB image." In: *Proceedings of the ieee conference on computer vision and pattern recognition (CVPR)*.

Buch, Anders G, Henrik G Petersen, and Norbert Krüger (2016). "Local shape feature fusion for improved matching, pose estimation and 3D object recognition." In: *Springerplus* 5.1, pp. 1–33.

Calli, Berk, Arjun Singh, Aaron Walsman, Siddhartha Srinivasa, Pieter Abbeel, and Aaron M. Dollar (2015). "The YCB object and model set: towards common benchmarks for manipulation research." In: *2015 international conference on advanced robotics (icar)*, pp. 510–517.

Carion, Nicolas, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko (2020). "DETR: end-to-end object detection with transformers." In: *Proceedings of the european conference on computer vision (ECCV)*, pp. 213–229.

Chen, Bo, Alvaro Parra, Jiewei Cao, Nan Li, and Tat-Jun Chin (2020). "End-to-end learnable geometric vision by backpropagating pnp optimization." In: *Proceedings of the ieee conference on computer vision and pattern recognition (CVPR)*, pp. 8100–8109.

Chen, Mark, Alec Radford, Rewon Child, Jeffrey Wu, Heewoo Jun, David Luan, and Ilya Sutskever (2020). "Generative pretraining from pixels." In: *International conference on machine learning (ICML)*, pp. 1691–1703.

Chen, Wei, Xi Jia, Hyung Jin Chang, Jinming Duan, and Ales Leonardis (2020). "G2l-net: global to local network for real-time 6D pose estimation with embedding vector features." In: *Proceedings of the ieee conference on computer vision and pattern recognition (CVPR)*, pp. 4233–4242.

Choi, Changhyun and Henrik I. Christensen (2016). "RGB-D object pose estimation in unstructured environments." In: *Robotics and autonomous systems* 75, pp. 595–613.

Chum, O. and J. Matas (2005). "Matching with prosac - progressive sample consensus." In: *Proceedings of the ieee conference on computer vision and pattern recognition (CVPR)*. Vol. 1, 220–226 vol. 1.

Dai, Zihang, Zhilin Yang, Yiming Yang, Jaime Carbonell, Quoc V Le, and Ruslan Salakhutdinov (2019). "Transformer-XL: attentive language models beyond a fixed-length context." In: *Arxiv:1901.02860*.

Deng, Jia, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei (2009). "Imagenet: a large-scale hierarchical image database." In: *Proceedings of the ieee conference on computer vision and pattern recognition (CVPR)*, pp. 248–255.

Devlin, Jacob, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova (2018). "BERT: pre-training of deep bidirectional transformers for language understanding." In: *Arxiv:1810.04805*.

Dosovitskiy, Alexey, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. (2020). "An image is worth 16x16 words: transformers for image recognition at scale." In: *Arxiv:2010.11929*.

Drost, Bertram, Markus Ulrich, Nassir Navab, and Slobodan Ilic (2010). "Model globally, match locally: efficient and robust 3d object recognition." In: *2010 ieee computer society conference on computer vision and pattern recognition*. Ieee, pp. 998–1005.

Esser, Patrick, Robin Rombach, and Björn Ommer (2020). "Taming transformers for high-resolution image synthesis." In: *Arxiv:2012.09841*.

Everingham, M., L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman (n.d.). *The PASCAL visual object classes challenge 2012 (VOC2012) results*. http://www.pascal-network.org/challenges/VOC/voc2012/workshop/index.html.

Everingham, Mark, Luc Van Gool, Christopher K. I. Williams, John Winn, and Andrew Zisserman (2010). "The PASCAL visual object classes (VOC) challenge." In: *International journal of computer vision (IJCV)* 88, pp. 303–338.

Fan, Zhaoxin, Yazhi Zhu, Yulin He, Qi Sun, Hongyan Liu, and Jun He (2021). "Deep learning on monocular object pose detection and tracking: a comprehensive overview." In: *Arxiv:2105.14291*.

Fischler, Martin A. and Robert C. Bolles (June 1981). "Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography." In: *Communications of the ACM* 24.6, pp. 381–395.

Gao, X., Xiaorong Hou, Jianliang Tang, and H. Cheng (2003). "Complete solution classification for the perspective-three-point problem." In: *Ieee transactions on pattern analysis and machine intelligence (PAMI)* 25, pp. 930–943.

Girshick, Ross (2015). "Fast R-CNN." In: *Proceedings of the ieee conference on computer vision and pattern recognition (CVPR)*, pp. 1440–1448.

Glent Buch, Anders, Lilita Kiforenko, and Dirk Kraft (Oct. 2017). "Rotational subgroup voting and pose clustering for robust 3D object recognition." In: *Proceedings of the ieee international conference on computer vision (ICCV)*.

Hartley, Richard and Andrew Zisserman (2004). *Multiple view geometry in computer vision*. 2nd ed. Cambridge University Press.

He, Kaiming, Xiangyu Zhang, Shaoqing Ren, and Jian Sun (2016). "Deep residual learning for image recognition." In: *Proceedings of the ieee conference on computer vision and pattern recognition (CVPR)*, pp. 770–778.

He, Yisheng, Wei Sun, Haibin Huang, Jianran Liu, Haoqiang Fan, and Jian Sun (2020). "PVN3D: a deep point-wise 3D keypoints voting network for 6DOF pose estimation." In: *Proceedings of the ieee conference on computer vision and pattern recognition (CVPR)*, pp. 11632–11641.

Hinterstoisser, Stefan, Vincent Lepetit, Slobodan Ilic, Stefan Holzer, Gary Bradski, Kurt Konolige, and Nassir Navab (2013). "Model based training, detection and pose estimation of texture-less 3D objects in heavily cluttered scenes." In: *Asian conference on computer vision (ACCV)*. Berlin, Heidelberg, pp. 548–562.

Hodan, Tomas, Frank Michel, Eric Brachmann, Wadim Kehl, Anders GlentBuch, Dirk Kraft, Bertram Drost, Joel Vidal, Stephan Ihrke, Xenophon Zabulis, Caner Sahin, Fabian Manhardt, Federico Tombari, Tae-Kyun Kim, Jiri Matas, and Carsten Rother (Sept. 2018). "BOP: benchmark for 6D object pose estimation." In: *Proceedings of the european conference on computer vision (ECCV)*.

Hodaň, Tomáš, Dániel Baráth, and Jiří Matas (2020). "EPOS: estimating 6D pose of objects with symmetries." In: *Proceedings of the ieee conference on computer vision and pattern recognition (CVPR)*.

Hodaň, Tomáš, Jiří Matas, and Štěpán Obdržálek (2016). "On evaluation of 6D object pose estimation." In: *Proceedings of the european conference on computer vision (ECCV)*, pp. 606–619.

Hodaň, Tomáš, Martin Sundermeyer, Bertram Drost, Yann Labbé, Eric Brachmann, Frank Michel, Carsten Rother, and Jiří Matas (2020). "BOP challenge 2020 on 6D object localization." In: *Proceedings of the european conference on computer vision (ECCV)*, pp. 577–594.

Hodaň, Tomáš, Xenophon Zabulis, Manolis Lourakis, Štěpán Obdržálek, and Jiří Matas (2015). "Detection and fine 3D pose estimation of texture-less objects in RGB-D images." In: *2015 ieee/rsj international conference on intelligent robots and systems (IROS)*. IEEE, pp. 4421–4428.

Hosang, Jan, Rodrigo Benenson, and Bernt Schiele (2017). "Learning non-maximum suppression." In: *Proceedings of the ieee conference on computer vision and pattern recognition (CVPR)*, pp. 4507–4515.

Hu, Yinlin, Pascal Fua, Wei Wang, and Mathieu Salzmann (2020). "Single-stage 6D object pose estimation." In: *Proceedings of the ieee conference on computer vision and pattern recognition (CVPR)*, pp. 2930–2939.

Hu, Yinlin, Joachim Hugonot, Pascal Fua, and Mathieu Salzmann (2019). "Segmentation-driven 6D object pose estimation." In: *Proceedings of the ieee conference on computer vision and pattern recognition (CVPR)*, pp. 3385–3394.

Jiang, Yifan, Shiyu Chang, and Zhangyang Wang (2021). "Transgan: two transformers can make one strong gan." In: *Arxiv:2102.07074*.

Kehl, Wadim, Fabian Manhardt, Federico Tombari, Slobodan Ilic, and Nassir Navab (2017). "SSD-6D: making RGB-based 3D detection and 6D pose estimation great again." In: *Proceedings of the ieee conference on computer vision and pattern recognition (CVPR)*, pp. 1521–1529.

Kehl, Wadim, Fausto Milletari, Federico Tombari, Slobodan Ilic, and Nassir Navab (2016). "Deep learning of local RGB-D patches for 3D object detection and 6D pose estimation." In: *Proceedings of the european conference on computer vision (ECCV)*. Ed. by Bastian Leibe, Jiri Matas, Nicu Sebe, and Max Welling, pp. 205–220.

Krantz, Steven George and Harold R Parks (2002). *The implicit function theorem: history, theory, and applications*. Springer Science & Business Media.

Kundu, Abhijit, Yin Li, and James M. Rehg (June 2018). "3D-RCNN: instance-level 3D object reconstruction via render-and-compare." In: *Proceedings of the ieee conference on computer vision and pattern recognition (CVPR)*.

Labbe, Y., J. Carpentier, M. Aubry, and J. Sivic (2020). "CosyPose: consistent multi-view multi-object 6D pose estimation." In: *Proceedings of the european conference on computer vision (ECCV)*.

Lepetit, Vincent, Francesc Moreno-Noguer, and Pascal Fua (2009). "EPnP: an accurate o(n) solution to the PnP problem." In: *International journal of computer vision (IJCV)* 81.2, p. 155.

Li, Chen, Zhen Zhang, Wee Sun Lee, and Gim Hee Lee (June 2018). "Convolutional sequence to sequence model for human dynamics." In: *Proceedings of the ieee conference on computer vision and pattern recognition (CVPR)*.

Li, Shichao, Zengqiang Yan, Hongyang Li, and Kwang-Ting Cheng (2021). "Exploring intermediate representation for monocular vehicle pose estimation." In: *Proceedings of the ieee conference on computer vision and pattern recognition (CVPR)*, pp. 1873–1883.

Li, Yi, Gu Wang, Xiangyang Ji, Yu Xiang, and Dieter Fox (2018). "DeepIM: deep iterative matching for 6D pose estimation." In: *Proceedings of the european conference on computer vision (ECCV)*, pp. 683–698.

Li, Zhigang, Yinlin Hu, Mathieu Salzmann, and Xiangyang Ji (2020). "Robust RGB-based 6-DoF pose estimation without real pose annotations." In: *Arxiv:2008.08391*.

Li, Zhigang, Gu Wang, and Xiangyang Ji (2019). "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.

Lin, Guosheng, Anton Milan, Chunhua Shen, and Ian Reid (July 2017). "Refinenet: multi-path refinement networks for high-resolution semantic segmentation." In: *Proceedings of the ieee conference on computer vision and pattern recognition (CVPR)*.

Lin, Tsung-Yi, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick (2014). "Microsoft COCO: common objects in context." In: *Proceedings of the european conference on computer vision (ECCV)*, pp. 740–755.

Liu, Wei, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C Berg (2016). "SSD: single shot multibox detector." In: *Proceedings of the european conference on computer vision (ECCV)*, pp. 21–37.

Liu, Yinhan, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov (2019). "RoBERTa: a robustly optimized BERT pretraining approach." In: *Arxiv:1907.11692*.

Liu, Ze, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo (2021). "Swin Transformer: hierarchical vision transformer using shifted windows." In: *Arxiv:2103.14030*.

Liu, Ze, Jia Ning, Yue Cao, Yixuan Wei, Zheng Zhang, Stephen Lin, and Han Hu (2021). "Video swin transformer." In: *Arxiv:2106.13230*.

Ma, Xinzhu, Shinan Liu, Zhiyi Xia, Hongwen Zhang, Xingyu Zeng, and Wanli Ouyang (2020). "Rethinking pseudo-lidar representation." In: *Proceedings of the european conference on computer vision (ECCV)*.

Manhardt, Fabian, Wadim Kehl, Nassir Navab, and Federico Tombari (2018). "Deep model-based 6D pose refinement in RGB." In: *Proceedings of the european conference on computer vision (ECCV)*, pp. 800–815.

Manhardt, Fabian, Gu Wang, Benjamin Busam, Manuel Nickel, Sven Meier, Luca Minciullo, Xiangyang Ji, and Nassir Navab (2020). "CPS++: improving class-level 6D pose and shape estimation from monocular images with self-supervised learning." In: *Arxiv:2003.05848*.

Meinhardt, Tim, Alexander Kirillov, Laura Leal-Taixe, and Christoph Feichtenhofer (2021). "TrackFormer: multi-object tracking with transformers." In: *Arxiv:2101.02702*.

Moré, Jorge J. (1978). "The levenberg-marquardt algorithm: implementation and theory." In: *Numerical analysis*, pp. 105–116.

El-Nouby, Alaaeldin, Hugo Touvron, Mathilde Caron, Piotr Bojanowski, Matthijs Douze, Armand Joulin, Ivan Laptev, Natalia Neverova, Gabriel Synnaeve, Jakob Verbeek, et al. (2021). "XCiT: cross-covariance image transformers." In: *Arxiv:2106.09681*.

Oberweger, Markus, Mahdi Rad, and Vincent Lepetit (2018). "Making deep heatmaps robust to partial occlusions for 3D object pose estimation." In: *Proceedings of the european conference on computer vision (ECCV)*, pp. 119–134.

Park, Kiru, Timothy Patten, and Markus Vincze (Oct. 2019). "Pix2Pose: pixel-wise coordinate regression of objects for 6D pose estimation." In: *Proceedings of the ieee international conference on computer vision (ICCV)*.

Paszke, Adam, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer (2017). "Automatic differentiation in PyTorch." In: *NIPS-W*.

Peng, Sida, Yuan Liu, Qixing Huang, Xiaowei Zhou, and Hujun Bao (2019). "PVNet: pixel-wise voting network for 6DOF pose estimation." In: *Proceedings of the ieee conference on computer vision and pattern recognition (CVPR)*, pp. 4561–4570.

Periyasamy, Arul Selvam, Max Schwarz, and Sven Behnke (2018). "Robust 6D object pose estimation in cluttered scenes using semantic segmentation and pose regression networks." In: *International conference on intelligent robots and systems (IROS)*.

– (2019). "Refining 6D object pose predictions using abstract render-and-compare." In: *Ieee-ras international conference on humanoid robots (Humanoids)*, pp. 739–746.

Qi, Charles R, Hao Su, Kaichun Mo, and Leonidas J Guibas (2017). "PointNet: deep learning on point sets for 3D classification and segmentation." In: *Proceedings of the ieee conference on computer vision and pattern recognition*, pp. 652–660.

Rad, Mahdi and Vincent Lepetit (2017). "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 (ICCV)*, pp. 3828–3836.

Radford, Alec, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever (2019). "Language Models are Unsupervised Multitask Learners." In.

Ren, Shaoqing, Kaiming He, Ross Girshick, and Jian Sun (2015). "Faster R-CNN: towards real-time object detection with region proposal networks." In: *Arxiv:1506.01497*.

Rezatofighi, Hamid, Nathan Tsoi, JunYoung Gwak, Amir Sadeghian, Ian Reid, and Silvio Savarese (2019). "Generalized intersection over union: a metric and a loss for bounding box regression." In: *Proceedings of the ieee conference on computer vision and pattern recognition (CVPR)*, pp. 658–666.

Shao, Jianzhun, Yuhang Jiang, Gu Wang, Zhigang Li, and Xiangyang Ji (2020). "PFRL: Pose-Free reinforcement learning for 6D pose estimation." In: *Proceedings of the ieee conference on computer vision and pattern recognition (CVPR)*.

Shaw, Peter, Jakob Uszkoreit, and Ashish Vaswani (June 2018). "Self-attention with relative position representations." In: *Proceedings of the 2018 conference*

of the north American chapter of the association for computational linguistics: human language technologies, volume 2 (short papers), pp. 464–468.

Shotton, Jamie, Ben Glocker, Christopher Zach, Shahram Izadi, Antonio Criminisi, and Andrew Fitzgibbon (June 2013). "Scene coordinate regression forests for camera relocalization in RGB-D images." In: *Proceedings of the ieee conference on computer vision and pattern recognition (CVPR)*.

Strudel, Robin, Ricardo Garcia, Ivan Laptev, and Cordelia Schmid (2021). "Segmenter: transformer for semantic segmentation." In: *Arxiv:2105.05633*.

Sun, Peize, Yi Jiang, Rufeng Zhang, Enze Xie, Jinkun Cao, Xinting Hu, Tao Kong, Zehuan Yuan, Changhu Wang, and Ping Luo (2020). "TransTrack: multiple-object tracking with transformer." In: *Arxiv:2012.15460*.

Sundermeyer, Martin, Zoltan-Csaba Marton, Maximilian Durner, Manuel Brucker, and Rudolph Triebel (2018a). "Implicit 3D orientation learning for 6D object detection from RGB images." In: *Proceedings of the european conference on computer vision (ECCV)*.

– (Sept. 2018b). "Implicit 3D orientation learning for 6D object detection from RGB images." In: *Proceedings of the european conference on computer vision (ECCV)*.

Synnaeve, Gabriel, Qiantong Xu, Jacob Kahn, Tatiana Likhomanenko, Edouard Grave, Vineel Pratap, Anuroop Sriram, Vitaliy Liptchinsky, and Ronan Collobert (2019). "End-to-end ASR: from supervised to semi-supervised learning with modern architectures." In: *Arxiv:1911.08460*.

Tay, Yi, Mostafa Dehghani, Dara Bahri, and Donald Metzler (2020). "Efficient transformers: a survey." In: *Arxiv:2009.06732*.

Tejani, Alykhan, Danhang Tang, Rigas Kouskouridas, and Tae-Kyun Kim (2014). "Latent-class hough forests for 3D object detection and pose estimation." In: *Proceedings of the european conference on computer vision (ECCV)*. Ed. by David Fleet, Tomas Pajdla, Bernt Schiele, and Tinne Tuytelaars, pp. 462–477.

Tekin, Bugra, Sudipta N Sinha, and Pascal Fua (2018a). "Real-time seamless single shot 6D object pose prediction." In: *Proceedings of the ieee conference on computer vision and pattern recognition (CVPR)*.

– (June 2018b). "Real-time seamless single shot 6D object pose prediction." In: *Proceedings of the ieee conference on computer vision and pattern recognition (CVPR)*.

Touvron, Hugo, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Hervé Jégou (2020). "Training data-efficient image transformers & distillation through attention." In: *Arxiv:2012.12877*.

Touvron, Hugo, Matthieu Cord, Alexandre Sablayrolles, Gabriel Synnaeve, and Herve Jegou (Oct. 2021). "Going deeper with image transformers." In: *Proceedings of the ieee international conference on computer vision (ICCV)*, pp. 32–42.

Vaswani, Ashish, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin (2017). "Attention is all you need." In: *Arxiv:1706.03762*.

Vidal, Joel, Chyi-Yeu Lin, Xavier Lladó, and Robert Martí (2018). "A method for 6D pose estimation of free-form rigid objects using point pair features on range data." In: *Sensors* 18.

Vince, John (2011). *Quaternions for computer graphics*. Springer Science & Business Media.

Wada, Kentaro, Edgar Sucar, Stephen James, Daniel Lenton, and Andrew J Davison (2020). "Morefusion: multi-object reasoning for 6D pose estimation from volumetric fusion." In: *Proceedings of the ieee conference on computer vision and pattern recognition (CVPR)*, pp. 14540–14549.

Wang, Chen, Danfei Xu, Yuke Zhu, Roberto Martín-Martín, Cewu Lu, Li Fei-Fei, and Silvio Savarese (2019). "DenseFusion: 6D object pose estimation by iterative dense fusion." In: *Proceedings of the ieee conference on computer vision and pattern recognition (CVPR)*, pp. 3343–3352.

Wang, Gu, Fabian Manhardt, Federico Tombari, and Xiangyang Ji (2021). "GDR-Net: geometry-guided direct regression network for monocular 6D object pose estimation." In: *Proceedings of the ieee conference on computer vision and pattern recognition (CVPR)*.

Wang, Huiyu, Yukun Zhu, Bradley Green, Hartwig Adam, Alan Yuille, and Liang-Chieh Chen (2020). "Axial-DeepLab: stand-alone axial-attention for panoptic segmentation." In: *Proceedings of the european conference on computer vision (ECCV)*, pp. 108–126.

Wang, Wenhai, Enze Xie, Xiang Li, Deng-Ping Fan, Kaitao Song, Ding Liang, Tong Lu, Ping Luo, and Ling Shao (Oct. 2021a). "Pyramid vision transformer: a versatile backbone for dense prediction without convolutions." In: *Proceedings of the ieee international conference on computer vision (ICCV)*, pp. 568–578.

– (Oct. 2021b). "Pyramid vision transformer: a versatile backbone for dense prediction without convolutions." In: *Proceedings of the ieee international conference on computer vision (ICCV)*, pp. 568–578.

Wu, Kan, Houwen Peng, Minghao Chen, Jianlong Fu, and Hongyang Chao (2021). "Rethinking and improving relative position encoding for vision transformer." In: *Proceedings of the ieee international conference on computer vision (ICCV)*, pp. 10033–10041.

Wu, Yuxin and Kaiming He (Sept. 2018). "Group normalization." In: *Proceedings of the european conference on computer vision (ECCV)*.

Xiang, Yu, Tanner Schmidt, Venkatraman Narayanan, and Dieter Fox (2017). "PoseCNN: a convolutional neural network for 6D object pose estimation in cluttered scenes." In: *Arxiv:1711.00199*.

Xu, Yihong, Yutong Ban, Guillaume Delorme, Chuang Gan, Daniela Rus, and Xavier Alameda-Pineda (2021). "Transcenter: transformers with dense queries for multiple-object tracking." In: *Arxiv:2103.15145*.

Yang, Jianwei, Chunyuan Li, Pengchuan Zhang, Xiyang Dai, Bin Xiao, Lu Yuan, and Jianfeng Gao (Dec. 2021). "Focal attention for long-range interactions in vision transformers." In: *Neurips 2021*.

Zappel, Moritz, Simon Bultmann, and Sven Behnke (2021). "6D object pose estimation using keypoints and part affinity fields." In: *Arxiv:2107.02057*.

Zeng, Fangao, Bin Dong, Tiancai Wang, Cheng Chen, Xiangyu Zhang, and Yichen Wei (2021). "MOTR: end-to-end multiple-object tracking with transformer." In: *Arxiv:2105.03247*.

Zhang, Pengchuan, Xiyang Dai, Jianwei Yang, Bin Xiao, Lu Yuan, Lei Zhang, and Jianfeng Gao (Oct. 2021). "Multi-scale vision longformer: a new vision transformer for high-resolution image encoding." In: *Proceedings of the ieee international conference on computer vision (ICCV)*, pp. 2998–3008.

Zheng, Sixiao, Jiachen Lu, Hengshuang Zhao, Xiatian Zhu, Zekun Luo, Yabiao Wang, Yanwei Fu, Jianfeng Feng, Tao Xiang, Philip H.S. Torr, and Li Zhang (2021). "Rethinking semantic segmentation from a sequence-to-sequence perspective with transformers." In: *Proceedings of the ieee conference on computer vision and pattern recognition (CVPR)*.

Zhou, Yi, Connelly Barnes, Jingwan Lu, Jimei Yang, and Hao Li (2019). "On the continuity of rotation representations in neural networks." In: *Proceedings of the ieee conference on computer vision and pattern recognition (CVPR)*, pp. 5745–5753.

Zhu, Xizhou, Weijie Su, Lewei Lu, Bin Li, Xiaogang Wang, and Jifeng Dai (2021). "Deformable DETR: deformable transformers for end-to-end object detection." In: *International conference on learning representations (ICLR)*.