Resolving Symmetry Ambiguity in Correspondence-based Methods for Instance-level Object Pose Estimation

Yongliang Lin, Yongzhi Su, Sandeep Inuganti, Yan Di,
Naeem Ajilforoushan, Hanqing Yang, Yu Zhang, Jason Rambach

***Abstract*—Estimating the 6D pose of an object from a single RGB image is a critical task that becomes additionally challenging when dealing with symmetric objects. Recent approaches typically establish one-to-one correspondences between image pixels and 3D object surface vertices. However, the utilization of one-to-one correspondences introduces ambiguity for symmetric objects. To address this, we propose SymCode, a symmetry-aware surface encoding that encodes the object surface vertices based on one-to-many correspondences, eliminating the problem of one-to-one correspondence ambiguity. We also introduce SymNet, a fast end-to-end network that directly regresses the 6D pose parameters without solving a PnP problem. We demonstrate faster runtime and comparable accuracy achieved by our method on the T-LESS and IC-BIN benchmarks of mostly symmetric objects. The code is available at https://github.com/lyltc1/SymNet.**

***Index Terms*—object pose estimation, symmetry ambiguity, deep learning for visual perception, representation learning.**

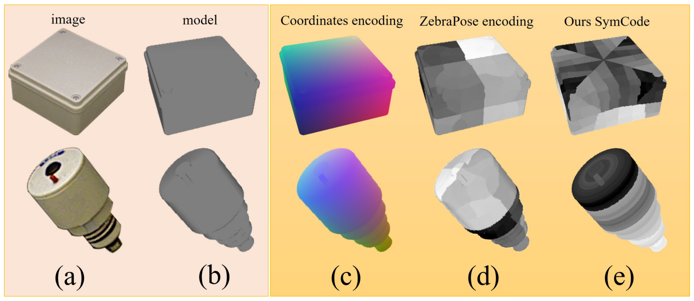


Fig. 1. **Comparison of Surface Encoding.** (a) Object images. (b) Textureless models. (c) 3D coordinate encoding. (d) ZebraPose encoding [13]. (e) Our proposed SymCode. The 3D coordinate encoding and ZebraPose encoding, based on one-to-one correspondences, do not consider symmetry. In contrast, SymCode, based on one-to-many correspondences, explicitly preserves symmetry information.

I. Introduction

OBJECT pose estimation is a crucial task in computer vision, which attracted significant research attention in recent years. Our goal is to accurately estimate the 6D poses of known objects from a single RGB image. Precise pose perception is crucial for manipulation [1], augmented reality [2] and autonomous driving applications [3]. Unlike methods incorporating depth information [4]–[8], RGB-only approaches [9]–[11] offer a broader range of applications at a lower cost.

Currently, high performing frameworks [12], [13], establish correspondences between 2D image pixels and 3D object surface vertices. A Perspective-n-Point (PnP) algorithm variant [14] is then used to estimate the pose. Correspondences are typically utilized in two ways: (1) as auxiliary targets [13], with the PnP algorithm applied subsequently to estimate the pose, or (2) as intermediate variables to output the pose through a network [12]. However, the PnP algorithm requires a one-to-one mapping between 2D and 3D points, which poses a challenge for symmetric objects. For instance, in the case of a texture-less ball, any pixel in the image can be associated with any vertex on the object's surface. The problem becomes more pronounced when the target correspondences are specified by a single ground truth pose.

Previous works [13], [15], [16] demonstrated that symmetrical ambiguity can be mitigated by employing powerful but computationally intensive methods like RANSAC-PnP variants to handle high outlier correspondence ratios. Other works [12], [17] train end-to-end networks to directly regress pose parameters, bypassing the PnP problem. However, ambiguity issues arise in cases of severe occlusion, leading to a significant decline in pose estimation accuracy due to a notable drop in the accuracy of correspondence prediction.

To address this issue, we propose SymCode, a dense correspondence encoding that encodes one-to-many 2D-3D correspondences, benefiting not only symmetrical objects but also objects that are "almost symmetrical", meaning that they exhibit symmetry to some extent but may have slight variations or irregularities that deviate from perfect symmetry. For asymmetrical objects, the one-to-many 2D-3D correspondences converge to one-to-one 2D-3D correspondences.

Manuscript received May 26, 2024; revised Dec 13 2024.
Corresponding authors: Yu Zhang and Jason Rambach.

Yongliang Lin, Hanqing Yang, Naeem Ajilforoushan and Yu Zhang are with State Key Laboratory of Industrial Control Technology, the College of Control Science and Engineering, Zhejiang University, Hangzhou 310027, China (email: lyl1020@zju.edu.cn; hanqing.yang@zju.edu.cn; Naeem_Ajilforoushan@zju.edu.cn; zhuangyu80@zju.edu.cn)

Yongzhi Su, Sandeep Inuganti are with the German Research Center for Artificial Intelligence (DFKI GmbH), 67663 Kaiserslautern, Germany, and also with the Faculty of Computer Science, RPTU Kaiserslautern, 67663 Kaiserslautern, Germany (e-mail: suyongzhi526@gmail.com; sain01@dfki.de)

Yan Di is with the Faculty of Computer Science, Technische Universität München, 85748 Garching bei München, Germany (e-mail: yan.di@tum.de)

Jason Rambach is with the German Research Center for Artificial Intelligence (DFKI GmbH), 67663 Kaiserslautern, Germany (email: jason.rambach@dfki.de)

SymCode vs. ZebraPose Encoding. We were inspired by the surface encoding introduced in ZebraPose [13] to assign a discrete code to each correspondence, which facilitates the network training. SymCode, as shown in Figure 1, internally captures symmetry information but does not adhere to the one-to-one correspondence assumption of the PnP algorithm. A similar situation is faced by SurfEmb [18], which generates a large number of pose candidates using P3P and employs a refinement process to obtain the final pose. However, this process is time-consuming. In contrast, we introduce an end-to-end network, called SymNet, to directly output the pose from SymCode along with a segmentation mask.

To summarize, the main contributions of our work are as follows:

1) We propose SymCode, a symmetry-aware binary surface encoding method that allows the estimation of one-to-many 2D-3D correspondences for pose estimation.
2) We introduce SymNet, a network that recovers object pose from one-to-many 2D-3D correspondences without relying on the PnP-RANSAC method, leading to significant improvements in inference time.
3) We investigate the impact of object symmetries as well as the performance of different symmetry-aware methods in correspondence-based object pose estimation through experiments on high-symmetry datasets such as T-LESS and IC-BIN.

II. Related Work

In this section, we briefly review works including learning-based methods using RGB data and methods incorporating mechanisms for handling of symmetries.

A. Learning-based Methods using RGB Data

Learning-based methods using RGB data can be seperated into direct methods and correspondence-based methods. Direct methods [19]–[24] explore parameters to represent rotation and translation with different network architectures and loss functions in the early stage. Correspondence-based methods [25]–[27] typically employ PnP/RANSAC modules to solve the pose. The performance of such methods [13] is usually higher and more robust to occlusion compared to direct methods [20]. Researchers employ various descriptors to represent dense 2D-3D correspondences. Pix2Pose [16], CDPN [26] consider a pixel's corresponding 3D vertex coordinates as descriptors. GDR-Net [12] introduced a direct regression network to estimate pose by utilizing dense correspondence-based intermediate representations. SO-Pose [17] generates both 2D-3D correspondences and self-occlusion information to directly regress pose parameters. ZebraPose [13] divides the model surface iteratively into halves and assigns binary labels. In this manner, they simplify the learning objective to a multi-label classification problem.

B. Handling Symmetries in Object Pose Estimation

Symmetry or pose ambiguity refers to the phenomenon where images of an object under different poses appear identical, making it impossible to determine the object's pose uniquely.

Direct methods need to compromise among all possible poses and sometimes resort to producing the average pose, which is not necessarily a valid solution. Early works such as SSD6D [19] and BB8 [28] constrain the range of rotation to remove ambiguity within the training set. Pitteri et al. [29] offers a general analytic approach to map symmetrical identities to the same pose, but results in discontinuity of the mapped pose, potentially causing abrupt changes in the network output. CosyPose [30] utilizes the symmetry-aware ADD-S loss, which measures pose accuracy by considering the closest symmetric pose of the object as ground truth. ES6D [31] categorizes symmetry into five categories and proposes a symmetry-invariant A(M)GPD loss based on these categories, which outperforms the ADD(S) loss. However, this method cannot be directly applied to correspondence-based methods.

Recent correspondence-based methods typically learn one-to-one correspondences as auxiliary targets. However, symmetry can introduce correspondence ambiguity, causing pixels in the image to have non-unique corresponding vertices on the model, thereby leading to pose ambiguity. Jesse Richter-Klug et al. [32] proposes a representation called "closed symmetry loop" to obtain one-to-many correspondences, where information represents the angle between different points and the object origin. This information is fused to achieve one-to-one correspondence. EPOS [33] utilizes surface fragments to address symmetry and establishes one-to-many 2D-3D correspondences. However, the final pose is estimated from a sampled triplet of correspondences by the P3P solver. This approach still has some limitations in continuous symmetry for regressing 3D coordinates on fragment coordinate systems. GDR-Net [12] guides pose regression by computing the loss with respect to the closest symmetric pose and incorporates surface fragments from EPOS [33]. SurfEmb [18] learns one-to-many correspondences in a self-supervised manner but samples one-to-one correspondences from the one-to-many correspondences to adapt to the RANSAC-P3P algorithm input. To the best of our knowledge, our work is the first to directly recover pose from one-to-many correspondences.

III. Method

This section offers a comprehensive description of our proposed one-to-many correspondence-based and symmetry-aware training method for 6D object pose estimation.

We aim to address the symmetrical ambiguity issue beyond the one-to-one correspondence approach. Drawing inspiration from the successful application of binary codes in one-to-one correspondence methods [8], [13], we extend this approach to encode one-to-many correspondences, which are then fed into a network to obtain the final pose without RANSAC or the need of refinement.

A. Problem Definition and Notation

Given an RGB image with known intrinsic parameters and a set of CAD models of objects, our objective is to estimate the pose of each object (Rotation matrix $\mathbf{R} \in SO(3)$ and translation vector $\mathbf{t} \in \mathbb{R}^3$) with respect to the camera in the image.

There is only one ground-truth pose $\mathbf{T} = [\mathbf{R}|\mathbf{t}]$ for an asymmetric object, while a symmetric object has multiple possible ground-truth poses $\mathbf{T}_k, k = 1, 2, ..., n$, where n is the number of ground-truth poses, which can be infinite.

ZebraPose [13] constructs a binary encoding of the N object model vertices, that defines binary codes $\mathbf{c}_j$ of d bits that uniquely correspond to vertices $\mathbf{P}_j$. The binary encoding is built iteratively, by splitting the mesh into parts of equal amount of vertices at each step, and assigning a bit to each group. In the iteration $it, it \in \{0, 1, \ldots, d-1\}$ of the surface partition, we have 2^{it+1} separate sub-groups. Given a group of vertices, the partitioning is carried out using balanced k-means, resulting in the formation of two sub-groups. For asymmetrical objects, the one-to-many correspondences are equivalent to the one-to-one correspondences since there is no ambiguity due to symmetry.

In the following sections, we compare one-to-one correspondences with one-to-many correspondences in detail. Subsequently, we present our approach for estimating the 6DoF object pose, which involves the entire process from surface encoding to the final pose estimation.

B. 2D-3D Correspondences

One-to-one Correspondences. Extracting one-to-one correspondences is a key component in pose estimation, which represents a match between a 2D point $\mathbf{p_i} = (u_i, v_i)$ from the observed image and a 3D point $\mathbf{P}_j = (x_j, y_j, z_j)$ from the object model, denoted as $\mathbf{o}_r = (\mathbf{p}_i, \mathbf{P}_j)$, where $\mathbf{p}_i \in \mathbb{R}^2$ and $\mathbf{P}_j \in \mathbb{R}^3$. The correspondence can be derived based on the ground-truth pose $\mathbf{T}$ point-wise:

$$\mathbf{p}_i = \pi(\mathbf{T} \cdot \mathbf{P}_j) \tag{1}$$

where $\pi(\cdot)$ is the projection function of a pinhole camera model using intrinsic matrix. We define a one-to-one correspondence set $\mathbf{O} = \{\mathbf{o}_1, \mathbf{o}_2, ..., \mathbf{o}_m\}$ containing m one-to-one correspondences, where $\mathbf{o}_r = (\mathbf{p}_i, \mathbf{P}_j)$. The Perspective-n-Point (PnP) module aims to recover the pose $\mathbf{T}$ given a set of one-to-one correspondences. Most correspondence-based methods use the one-to-one correspondence set as an intermediate geometric representation. For an asymmetric object, the one-to-one correspondence is unique and can be recovered without ambiguity, but this is not the case for symmetric objects.

For each ground-truth pose $\mathbf{T}_k$ of a symmetric object, we can obtain a one-to-one correspondence set using Equation (1). However, in the training stage, setting similar images with vastly different correspondence sets as learning targets can lead to convergence problems. We illustrate different one-to-one correspondence sets for a cube and a cylinder in Figure 2.

One-to-many Correspondences. We define a one-to-many correspondence to represent a match between a 2D point $\mathbf{p}_i = (u_i, v_i)$ and a set of all possible 3D points $\mathbf{Y}_j = \{\mathbf{P}_{j,1}, \mathbf{P}_{j,2}, ..., \mathbf{P}_{j,n}\}$, denoted as $\mathbf{o}_r = (\mathbf{p}_i, \mathbf{Y}_j)$. The correspondence should satisfy the following condition:

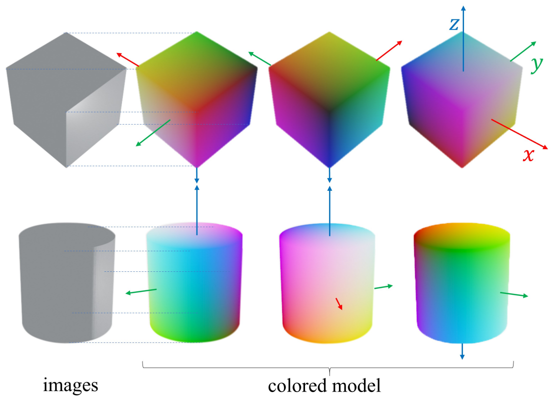


Fig. 2. **Multiple possible one-to-one correspondence sets.** Left column: Images of an untextured cube and cylinder. Top: three possible correspondence sets for the cube image. Bottom: three possible correspondence sets for the cylinder image. The color of the models in the right three columns represents the coordinates in the object frame, with red, green, and blue representing the coordinates of the x-axis, y-axis, and z-axis, respectively. The object frame is represented by colored arrows as coordinates. The dashed lines show 2D-3D correspondences. Best viewed in color mode.

$$\mathbf{p}_i = \pi(\mathbf{T}_k \cdot \mathbf{P}_{j,k}), \quad k = 1, 2, ..., n \tag{2}$$

$\mathbf{T}_k$ are all the possible ground-truth poses. We define a one-to-many correspondence set $\mathbf{O}_{\text{sym}} = \{\mathbf{o}_1, \mathbf{o}_2, ..., \mathbf{o}_m\}$ containing m one-to-many correspondences, where $\mathbf{o}_r = (\mathbf{p}_i, \mathbf{Y}_j)$, as depicted in Figure 3. The one-to-many correspondence demonstrate that from a pixel $\mathbf{p}_i$, it is impossible to determine the matching 3D point coordinates $\mathbf{P}_j$, but it is possible to determine the corresponding 3D point set $\mathbf{Y}_j$. The one-to-many correspondence presents an easier learning task for the network compared to the one-to-one correspondences due to its lack of ambiguity.

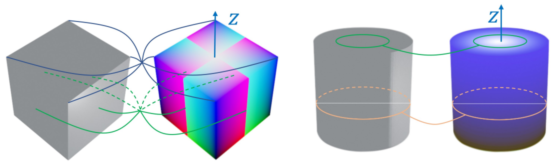


Fig. 3. **One-to-many correspondences set.** All 3D points in a one-to-many correspondence are textured with the same color. Left: two specific one-to-many correspondences for corners and centers of side faces, assuming the cube has only 4 symmetries rotated along the z-axis with 0, 90, 180, 270 degrees. Invisible parts are connected by dashed lines, and visible parts are connected by solid lines. Right: two specific one-to-many correspondences for the side surface and top surface.

C. Symmetry-aware Surface Encoding

Our objective is to encode each one-to-many correspondence $\mathbf{o}_r$ with a discrete binary code $\mathbf{c}_r$. Figure 4 presents a schematic overview of the proposed one-to-many surface encoding pipeline.

Generate One-to-Many Correspondences. When an object possesses symmetry, there are multiple rigid motions that

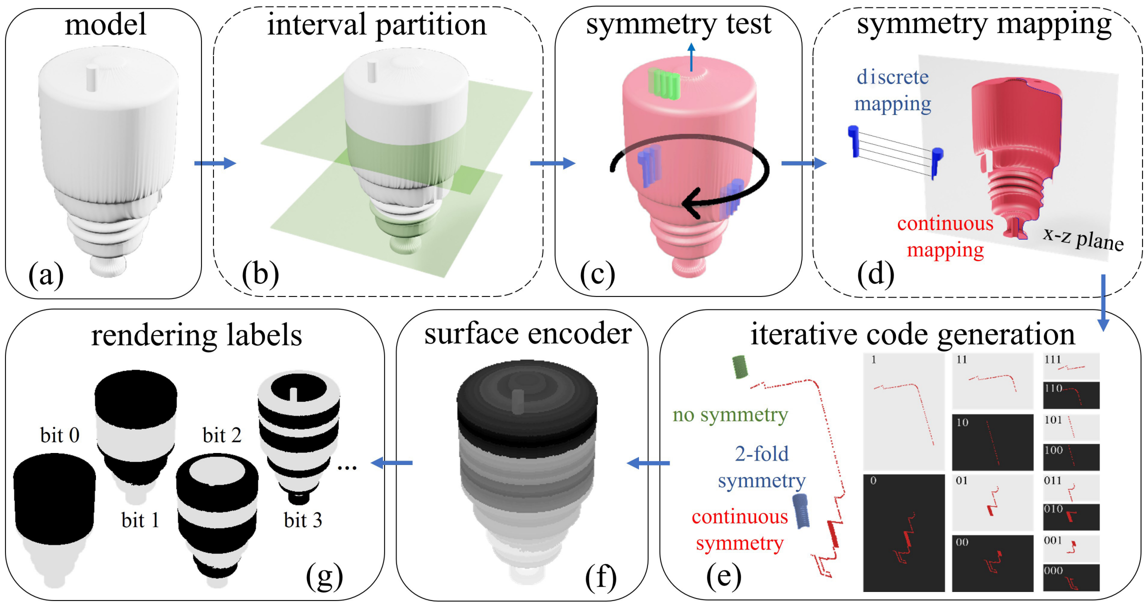


Fig. 4. **The generation process of SymCode and label rendering.** (a) Object CAD models. (b) For complex models, we manually partition the model into several parts to ensure higher accuracy in the following process. (c) Testing the symmetry type of a vertex based on symmetry priority. (d) Finding the main vertex of each one-to-many correspondence; continuous symmetry and discrete symmetry are processed differently. (e) Each correspondence is assigned a unique binary code $\mathbf{c}_i$. (f) The surface encoder encodes each vertex in the model with a binary code, which inherently preserves symmetry information. (g) The rendered label is used as an intermediate target for the network. Optional processes are enclosed in dashed boxes.

can map one ground-truth pose to other ground-truth poses. Formally, we consider the set as follows:

$$\mathcal{M} = \{\mathbf{m} \in SE(3) \mid \mathbf{T}_1 \cdot \mathbf{m} \in \{\mathbf{T}_1, \mathbf{T}_2, ..., \mathbf{T}_n\}\} \tag{3}$$

where $\mathbf{T}_1$ represents any single ground-truth pose, $\mathbf{T}_k$ represents another ground-truth pose where n denotes the total number of possible ground-truth poses.

Symmetries can be categorized into two types: discrete and continuous. In the case of discrete symmetry, the set $\mathcal{M}$ consists of a finite number of elements. For a vertex $\mathbf{P}$, we generate all other possible 3D vertices belonging to the same one-to-many correspondence as

$$\mathbf{P}_{j,k} = \mathbf{m} \cdot \mathbf{P}, \quad k = 1, 2, .., n, \quad \mathbf{m} \in \mathcal{M} \tag{4}$$

In the case of continuous symmetry, the vertices on the surface undergo rotation along the axis of symmetry, resulting in their alignment on a plane, as illustrated in Figure 4(d). By doing so, vertices that undergo rotation and end up at the same position on the plane are consolidated into the same correspondence. We can formalize the rotation process as follows:

$$\tilde{\mathbf{P}} = \begin{bmatrix} \sqrt{x^2 + y^2} & 0 & z \end{bmatrix}^T \tag{5}$$

Here, $\tilde{\mathbf{P}}$ is the transferred vertex of origin vertex $\mathbf{P}$, and is referred to as the main vertex. x, y, z represent the three components of $\mathbf{P}$. It is assumed that the rotation occurs along the z-axis, and the resulting plane is the x-y plane, as illustrated in Figure 4(d).

Handling Mixture Symmetry. For the T-LESS dataset, different objects are only annotated with the main type of symmetry. We can directly generate one-to-many correspondences based on this symmetry annotation. However, many objects consist of a mixture of symmetric parts. Consider an object constructed by combining a cube and a cylinder together. Although considering the entire object may suggest discrete symmetry, utilizing correspondences constructed based on discrete symmetry in occluded scenarios would still result in ambiguity.

Additionally, some objects are considered "almost symmetrical," meaning they exhibit symmetry except for a small detail. To tackle these challenges, we offer an advanced annotation tool that enables more precise labeling of object models. This is the only part of the process that is still manual, but can be completed within a few minutes only. If the object is perfectly symmetrical, this step is not necessary, making the entire annotation process fully automated, relying on symmetry annotation information from BOP [34].

Figure 4(a) provides an example of this scenario. We explicitly categorize points on objects into the following four categories (1) no symmetry, (2) continuous symmetry, (3) discrete symmetry, and (4) n-fold symmetry. The n-fold symmetry is a special case of discrete symmetry, i.e. the angle of symmetry is

$$\theta = i \cdot 2\pi/n, \quad i \in 1, ..., n \tag{6}$$

along the axis. The classification is determined by the average distance between each vertex and its nearest vertex after applying the transformation in Equation (4). Since different types of symmetry inherently possess an inclusion relationship, continuous symmetry is included within discrete symmetry, discrete symmetry is included within n-fold symmetry, and a vertex that does not belong to any of the aforementioned types will be considered as having no symmetry. We assess symmetry in the following order: continuous symmetry, discrete symmetry, n-fold symmetry, and finally, no symmetry, which we refer to as symmetry priority. The primary symmetry type is provided in the dataset's meta information. However, for the details that break the symmetry, human intervention is required to specify them and ensure that the correct symmetry type is identified. Additionally, thresholds of error derived from empirical or experimental observations can also be used to assist in identifying the symmetry type sym. For a given symmetry type sym, we will generate the correspondence set $\mathcal{M}$ based on that symmetry and calculate the error under this specific symmetry type sym as follows:

$$\mathbf{e}_{j,sym} = \sum_{m \in \mathcal{M}} \|\mathbf{P}_{\text{near}} - \mathbf{mP}\| \tag{7}$$

where $\mathbf{P}_{\text{near}}$ refers to the nearest vertex to the transformed vertex $\mathbf{mP}$ on the object surface.

After removing the vertices that belong to high-priority symmetry, the remaining vertices can be classified into the low-priority symmetry category. Any vertices that are not categorized as belonging to any symmetry type will be recognized as having no symmetry.

In order to attain a more precise classification, the model can be subdivided into smaller intervals, enabling the identification of different forms of potential symmetry within each interval. This approach allows for a more detailed analysis of the symmetry structure, as depicted in Figure 4(b).

Encoding Correspondence. We have established the one-to-many correspondence set $\mathbf{O}_{\text{sym}}$ wherein each group $\mathbf{Y}_j$ comprises vertices denoted as $\mathbf{P}_{j,k}$. Additionally, each group $\mathbf{Y}_j$ is associated with a symmetry type sym. We introduce a method for encoding the one-to-many correspondence set. The object surface is encoded by incorporating a hierarchical binary grouping scheme for the one-to-many correspondence set.

The encoding indicates which corresponding group $\mathbf{Y}_j$ this pixel matches to. The encoding should ideally meet the following criteria: 1) To provide ample information for pose estimation, the encoding should exhibit significant differences for two groups $\mathbf{Y}_j$ and $\mathbf{Y}'_j$ with distinct symmetry types. 2) To facilitate easier learning for the network, correspondences that are in close proximity should have more similar encodings. 3) Each $\mathbf{Y}_j$ corresponds to a unique encoding. In general, we select one 3D vertex, referred to as the main vertex, from each correspondence while considering the spatial proximity of neighboring correspondences. The main vertex will be utilized for implementing binary encoding in the next step.

Next, we introduce how to obtain the main vertex for each correspondence. For correspondences exhibiting continuous symmetry, we select the vertex mapped to the plane as the main vertex using Equation (5). For other types of symmetry, we select the main vertex based on a simple criterion explained below, which can be replaced with alternative approaches as long as it satisfies the criteria (2). In our implementation, the used criterion calculates the sum of the absolute values of each coordinate component for each vertex in the correspondence. The main vertex, denoted as $\tilde{\mathbf{P}}$, is then chosen according to the equation:

$$\tilde{\mathbf{P}} = \max_{\mathbf{P}_{j,k}}(|x_{j,k}| + |y_{j,k}| + |z_{j,k}|), \quad \mathbf{P}_{j,k} \in \mathbf{Y}_j \tag{8}$$

As illustrated in Figure 4(e), the main vertices for this object are represented by the colored vertices. Only one vertex in the correspondence is not symmetrical, which will be the main vertex, colored green. For continuous symmetry, the main vertex, colored red, lies on the plane calculated by Equation (5). Regarding 2-fold symmetry, the vertices on the portion with higher coordinate values will serve as the main vertices, colored blue.

Iterative code generation. This part is inspired by the binary encoding used in the one-to-one correspondence method [13]. Given the one-to-many correspondences set, we want to represent each correspondence $\mathbf{o}_i$ with a binary code $\mathbf{c}_i \in 0, 1^d$, where d is the length of the binary code. We construct such encodings based on the main vertex of each correspondence.

In Figure 4(e), we illustrate the process of performing d iterations of grouping the main vertices. The grouping iteration begins with collections of all main vertices G. For a group G_i with a binary code $\mathbf{c}_i$, we divide it into two groups using k-means. The resulting groups are assigned codes $\mathbf{c}_i \ll 1$ and $(\mathbf{c}_i \ll 1) + 1$, where the operation $\ll$ denotes binary left shift. Eventually, we obtain 2^d groups, each with its binary code. These binary codes can be represented in decimal form from 0 to $2^d - 1$. The surface encoder is depicted in Figure 4(f), where the color represents the decimal value of the binary code $\mathbf{c}_i$.

D. Network Architecture

Our network is inspired by ZebraPose [13] and GDR-Net [12]. In Figure 5, we provide the Binary Encoding Module with a zoomed-in region of interest (RoI) with dimensions $256 \times 256 \times 3$. The outputs, namely the amodal mask M_{amo} with dimensions 128×128, visible mask M_{vis} with dimensions 128×128, and SymCode maps M_{code} with dimensions $128 \times 128 \times 16$, serve as intermediate variables. Subsequently, these intermediate variables are utilized as inputs to the Correspondence-based Pose Regression (CPR) module.

The Binary Encoding Module includes a ResNet-34 backbone [35] and utilizes Atrous Spatial Pyramid Pooling (ASPP) [36], which employs several atrous convolutions in parallel that are effective for resampling features at different scales. We also experimented with different convolution strategies and upsampling methods as part of the network architecture from CDPN [26], but the results did not show significant differences.

We utilize a straightforward Correspondence-based Pose Regression (CPR) module to directly regress the 6D pose from the visible mask M_{vis}, amodal mask M_{amo}, and SymCode maps M_{code}. The CPR module comprises three convolutional layers, each followed by Group Normalization and ReLU activation. Subsequently, two fully connected layers are applied to the flattened features. Finally, two parallel fully connected layers output the 3D rotation parameterized as $\mathbf{R}_{6d}$ [37] and the 3D

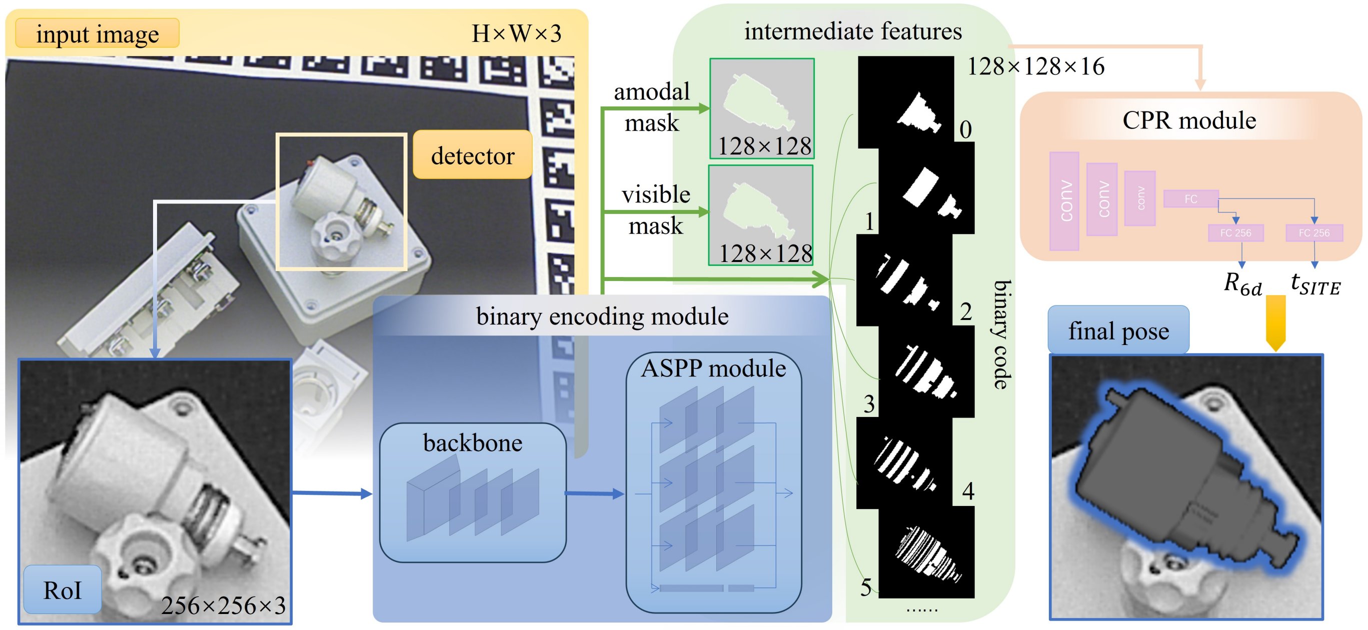


Fig. 5. **Framework of SymNet.** Given an RGB image, our SymNet takes the zoomed-in Region of Interest (RoI) as input and predicts intermediate features, including masks and binary code maps. The CPR module then directly regresses the 6D object pose. The entire process is an end-to-end procedure, eliminating the need for refinement or RANSAC processes for PnP.

translation parameterized as $\mathbf{t}_{\text{SITE}}$ [26], respectively. In all, the total parameters of our network amount to $63.2M$.

We utilize $L1$ loss for both the visible mask, amodal mask and SymCode maps. Our training loss is defined as

$$Loss = L_{\text{masks}} + L_{\text{code}} + L_{\text{params}} + L_{\text{ADD-S}} \tag{9}$$

L_{params} corresponds to the loss for end-to-end training utilizes the $L1$ loss for $\mathbf{R}_{6d}$ and $\mathbf{t}_{\text{SITE}}$. Additionally, the term $L_{\text{ADD-S}}$ is derived from the work of PoseCNN [38].

In terms of the training process, all losses are trained simultaneously without any parameters being frozen. In our preliminary experiments, we also explored a non-end-to-end training strategy, where $L_{\text{masks}} + L_{\text{code}}$ was used to train the Binary Encoding Module, while $L_{\text{params}} + L_{\text{ADD-S}}$ was used solely to train the CPR module; however, this led to a slight decrease in results.

IV. EXPERIMENTS

In this section, we introduce evaluation metrics and briefly introduce a symmetry-aware version of ZebraPose [13] - ZebraPoseSAT, then compare our method with other model-based approaches on the T-LESS [39] and IC-BIN [40] datasets. These datasets encompass a variety of symmetric objects, while a large synthetic-to-real domain gap exists because of texture mismatch. This makes T-LESS and IC-BIN particularly suitable for evaluating the performance of methods on symmetric objects. Furthermore, we present ablation experiments on various hyperparameters and visualizations to intuitively demonstrate the impact of SymCode.

A. Evaluation Protocol

The Visible Surface Discrepancy (VSD) evaluates the proportion of visible pixels for which the depth absolute discrepancy falls below a threshold of $\tau = 20mm$. We report the recall of correct 6D object poses at $e_{VSD} < 0.3$ [29]. We also adhere to the evaluation protocol outlined in the BOP challenge [34] using three metrics: Visible Surface Discrepancy (VSD), Maximum Symmetry-aware Surface Distance (MSSD), and Maximum Symmetry-aware Projection Distance (MSPD).

B. ZebraPoseSAT: Alternative Symmetry-aware Solution

To provide a symmetry-aware version of ZebraPose [13] for comparison, we additionally implement ZebraPoseSAT (SAT standing for Symmetry-Aware Training), which utilizes an analytical approach [29] to map all ground truth poses $\mathbf{T}_i$ to a unique $\mathbf{T}$ based on the Frobenius norm, prior to generating the ZebraPose encoding. ZebraPoseSAT emerged as the winner of the Best RGB-Only Method in the BOP 2023 challenge [34], providing a strong comparison baseline for SymNet.

C. Comparison to State of the Art

Results on T-LESS. In Table I, we show a comparison with previous methods on the T-LESS dataset for $e_{VSD} < 0.3$ recall. We generate the 2D detections using the FCOS [43] detector. The results demonstrate that SymNet outperforms all other methods, achieving a remarkable 25.0% improvement over the results in RGB setting reported by Pitteri et al. [29]. It is worth noting that the number of recent works providing a comparison based on this specific metric is limited, as the BOP metric offers a more comprehensive and accurate evaluation pipeline. Pitteri et al. [29] provide the most recent results of $e_{VSD} < 0.3$ recall we could find in an RGB setting. Unfortunately, the Retina detector code is outdated and unable to locate detection results. Therefore, we have included results

TABLE I
T-LESS: OBJECT RECALL FOR $err_{vsd} < 0.3$ ON ALL PRIMESENSE TEST SCENES. THE RESULTS FOR THE 30 OBJECTS ARE GROUPED BASED ON THEIR SYMMETRY TYPE.

Method	AAE [41]		Pix2Pose [16]	EdgeEnhance [42]	Pitteri [29]	CosyPose [30]	Ours(pbr)	Ours(pbr+real)
Detector	SSD	Retina	Retina	Retina	Faster-RCNN	Retina	FCOS(pbr)	FCOS(pbr+real)
Symmetry type	RGB	RGB	RGB	RGB	RGB	RGB-D	RGB	RGB
Asymmetry (3)	25.98	16.95	24.73	28.76	36.533	-	66.36	**69.47**
Continuous (11)	11.90	17.98	36.27	39.05	46.65	-	61.52	**63.72**
Discrete (16)	14.45	18.86	25.79	32.77	38.47	-	69.51	**78.02**
Mean	14.67	18.35	29.5	34.67	41.27	62.6	66.27	**71.92**

TABLE II
BOP RESULTS ON DATASET T-LESS. THE TIME IS THE RUNTIME PER IMAGE AVERAGED OVER THE DATASET.

6D object pose estimation method	Input type	Training type	AR	AR_{VSD}	AR_{MSSD}	AR_{MSPD}	Time(s)
CDPNv2 [26]	RGB	pbr	0.407	0.303	0.338	0.579	1.849
CosyPose [30]	RGB	pbr	0.640	0.571	0.589	0.761	0.493
EPOS [33]	RGB	pbr	0.467	0.380	0.403	0.619	1.992
ZebraPose [13]	RGB	pbr	0.677	0.597	0.636	0.466	0.25
ZebraPoseSAT-EffnetB4 [13]	RGB	pbr	0.723	0.659	**0.695**	0.817	0.25
SurfEmb [18]	RGB	pbr	0.735	**0.661**	0.686	0.857	9.043
SymNet(Ours)	RGB	pbr	**0.736**	0.631	0.693	**0.883**	**0.093**
DPODv2 [44]	RGB-D	pbr	0.699	0.646	0.716	0.736	0.320
ZebraPose [13]	RGB	real+pbr	0.775	0.696	0.740	0.889	0.25
SymNet(Ours)	RGB	real+pbr	0.767	0.674	0.739	0.883	0.058

TABLE III
BOP RESULTS ON DATASET IC-BIN [40]. THE TIME IS THE RUNTIME PER IMAGE AVERAGED OVER THE DATASET.

6D object pose estimation method	Input type	Training type	AR	AR_{VSD}	AR_{MSSD}	AR_{MSPD}	Time(s)
CRT-6D [45]	RGB	pbr	0.537	**0.477**	0.517	0.618	0.120
ZebraPoseSAT-EffnetB4 [13]	RGB	pbr	0.545	0.475	**0.535**	0.625	0.25
SymNet(Ours)	RGB	pbr	**0.547**	0.450	0.511	**0.678**	**0.088**

from CosyPose [30], which also provides BOP scores and $e_{VSD} < 0.3$ recall simultaneously. We provide these results to facilitate comparisons with earlier works. Moreover, we have also included results obtained using real images for training. The performance gap between the models trained solely on synthetic data and those incorporating real images is not substantial. This demonstrates the strong generalization capabilities of our method across synthetic and real-world domains. The objects of T-LESS are grouped in 3 categories based on their symmetry type and we provide the average scores for all objects in each category.

Results on BOP Benchmark. We compare our results to other methods that are fully trained on synthetic data, as shown in Table II and Table III. We use the default detections provided by BOP challenge 2023 [34]. Since our method does not rely on a time-consuming pose refinement step and directly obtains the pose for every detection, our runtime is significantly reduced compared to the other methods. Our method achieves excellent results in terms of both accuracy and runtime. Specifically, our results match the accuracy of ZebraposeSAT-EffnetB4 but with a smaller backbone, and only at one third of the runtime. Note that we compare the run-time per object instance. However, it should not significantly increase the run-time if inferring multiple objects through parallelization.

D. Ablation Studies

TABLE IV
COMPARE CPR MODULE WITH PNP MODULE ON DATASET T-LESS.

CPR	EPnP	AR	AR_{VSD}	AR_{MSSD}	AR_{MSPD}
	✓	0.283	0.166	0.190	0.493
✓		0.736	0.631	0.693	0.883

Compare with PnP solver. In Table IV, We compare the performance of our CPR module with that of RANSAC/PnP. To the best of our knowledge, this is the only approach to solving one-to-many correspondence scenarios, as demonstrated in SurfEmb [18]. We randomly sample a one-to-one match from each one-to-many correspondence. Following ZebraPose, we employ the EPnP algorithm [14], using 150 iterations, with all correspondences utilized in a single computation step. We believe that with more parallel processing, like that in SurfEmb, we can achieve higher accuracy using RANSAC/PnP, albeit at the cost of time. The results reveal that using PnP alone is challenging in terms of obtaining accurate correspondences, which are crucial for achieving reliable pose estimation. On

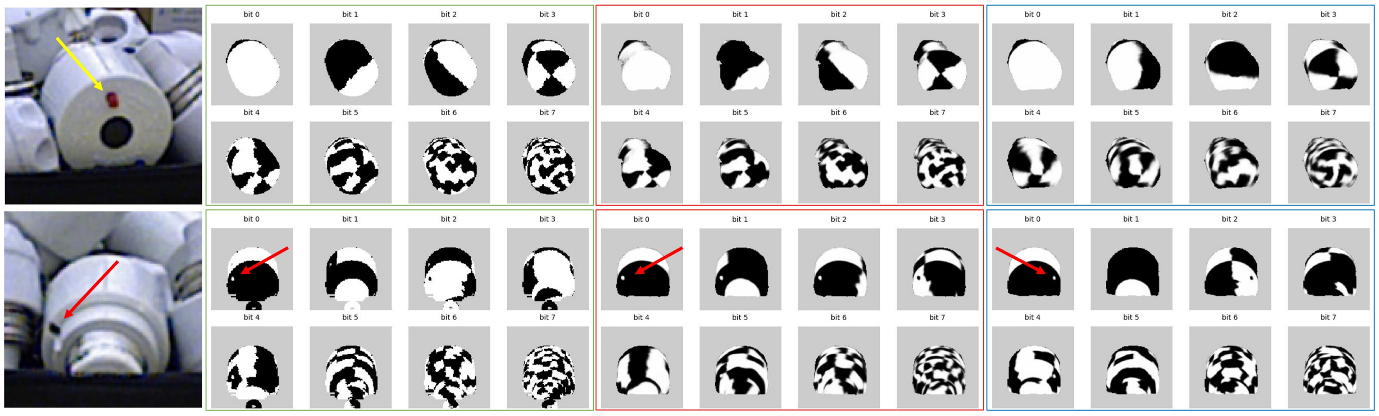


Fig. 6. **ZebraPose(real) vs. ZebraPose(pbr).** Left: input image. The green box indicates the ground truth, the red box shows the ZebraPose encoding trained with real data, and the blue box represents the ZebraPose encoding trained with synthetic data. We visualize two cases for the T-LESS obj04 object, which features a red section (indicated by the yellow arrow) at the top and two dents (indicated by the red arrows) in the middle, disrupting its continuous symmetry. It can be observed that the results of the model trained on real data are closer to the ground truth, as the network can leverage subtle surface distinctions to avoid ambiguity during training with real images. These subtle distinctions exist in the simulation data but are less pronounced; for instance, color information is absent in the T-LESS synthetic training data.

the other hand, our CPR module is capable of accurately calculating the final pose in real-time.

Compare with ZebraPose. Due to the limitation of one-to-one correspondence, we initially assumed that ZebraPose would not perform well on symmetrical objects. However, in the BOP benchmark, we observe that ZebraPose achieves a satisfactory average recall score of 0.775, which is higher than our result of 0.767 when trained with real data. We attribute this to the utilization of real images during training. Since most objects are not perfectly symmetrical, the network trained with real data can capture subtle variations and mitigate the symmetry problem. We compare ZebraPose trained on real images with that trained on synthetic images in Figure 6 to support our claims. The results we can access are specifically for the model with a larger backbone, referred to as ZebraPoseSAT-EffnetB4, which has been trained solely on PBR data. However, the results for the ZebraPose model trained exclusively on PBR data are not publicly available. We performed the experiment ourselves using open-source project code, and the average recall score is 0.677. In the same setting, our score 0.736 exhibits improvements in both accuracy and inference time.

Length of SymCode. The default setting of length for d is 16 as used in ZebraPose [13]. We investigate the impact of varying d on the training process, as illustrated in Figure 7. The results indicate that our approach is robust to variations in the length of the binary code. Indeed, the accuracy achieved with different values of d can be unpredictable, and there is no clear way to determine an optimized value beforehand. As a result, in our main results, we have chosen to fix the length of the binary code to the default value of 16. As shown in Figure 8, the network performs poorly on the last few bits, which can also be observed in ZebraPose. From Figure 7, it is evident that a length of 16 is the worst choice. This suggests that there is still room for optimization in the length of the code. So we performed comparisons using lengths of 16 and 10 for the add-s metric on YCB-V dataset, with results improving from 73.68 to 74.66. Since this improvement was not substantial, we did not explore other lengths further.

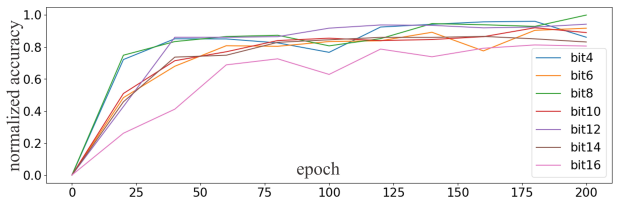


Fig. 7. We performed an ablation study on object 27 from T-LESS dataset, which exhibits discrete symmetry, to determine the optimal length of the binary code d. We normalized the BOP score to represent accuracy. The resulting curve demonstrates that even with a small number of bits (8 bits), our method is capable of capturing the object's pose.

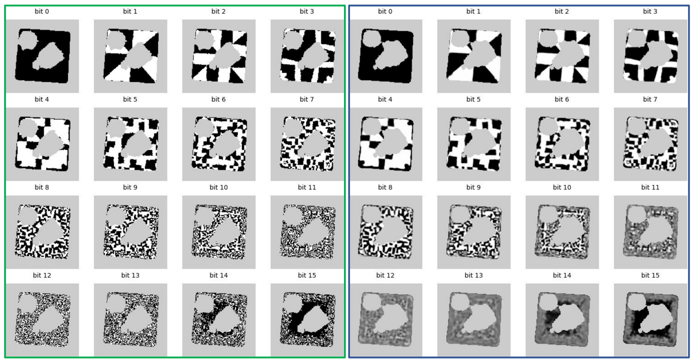


Fig. 8. Left: Ground truth of the 16-bit SymCode. Right: Estimated SymCode. This suggests that there is still room for optimization in the length of the code.

Challenge of learning one-to-one correspondences. We conducted experiments involving training our end-to-end network using ZebraPose encodings, which is an efficient way to encode one-to-one correspondences. In this case, the network achieved a BOP recall of 0.612, which is relatively weaker compared to SymNet's performance of 0.736. When we removed all the end-to-end loss and focused solely on training the ZebraPose encodings, Figure 9 illustrates the challenges associated with learning one-to-one correspondence-based encodings.

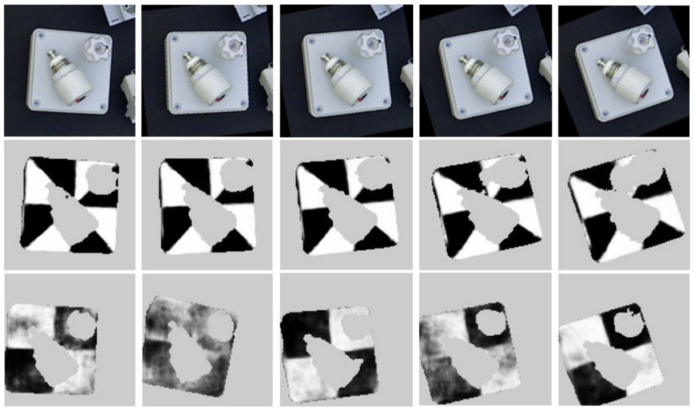


Fig. 9. **The challenge of learning one-to-one correspondences.** First line: The images with similar viewpoints. Second line: the 2nd bit prediction of SymCode. Third line: the 2nd bit prediction of ZebraPose Code. In order to achieve high accuracy pose estimation, ZebraPose Code relies on RANSAC-PnP to filter out inconsistent correspondences. However, there are still cases (second column) where ZebraPose struggles due to the ambiguity arising from symmetries. Please note that the result was obtained using our SymNet network trained with ZebraPose Code. To achieve a clear visualization, we have applied a mask based on the predicted mask to remove the background. In the visualization of the predictions, we have represented a value of 0 as black and a value of 1 as white. Any value between 0 and 1 is displayed as a shade of gray.

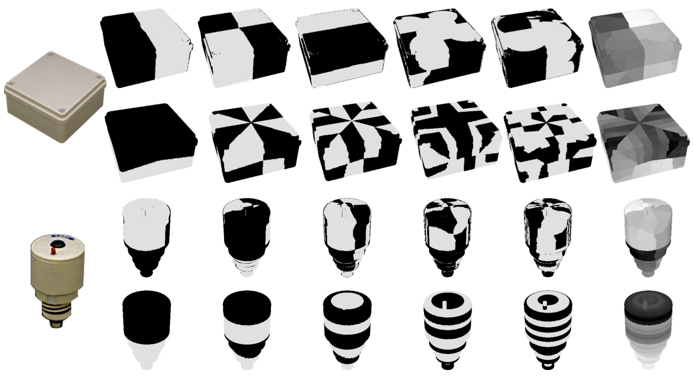


Fig. 10. **Comparison of code maps.** Detailed binary code maps representing SymCode and ZebraPose are visualized for two objects. The first and third rows depict the code maps for ZebraPose, while the second and fourth rows illustrate those for SymCode. The final columns aggregate the results for all the bits.

Visualization. Detailed binary code maps representing SymNet and ZebraPose are visualized in Figure 10, which shows our binary codes contain symmetry information. Qualitative results on T-LESS [39] can be found in Figure 11. A visual representation of the ground truth and predicted code maps for ZebraPose and SymNet is shown in Figure 12. From the visualization, it is evident that ZebraPose struggles to accurately reproduce the ground truth, whereas SymNet exhibits better performance in this regard.

V. CONCLUSION

Our method incorporates a symmetry discrete surface encoding technique to effectively handle symmetries. Furthermore, we demonstrate the ability to recover the pose from one-to-many 2D-3D correspondences. This approach holds the potential to influence not only other correspondence-based methods but also various other fields related to pose estimation.

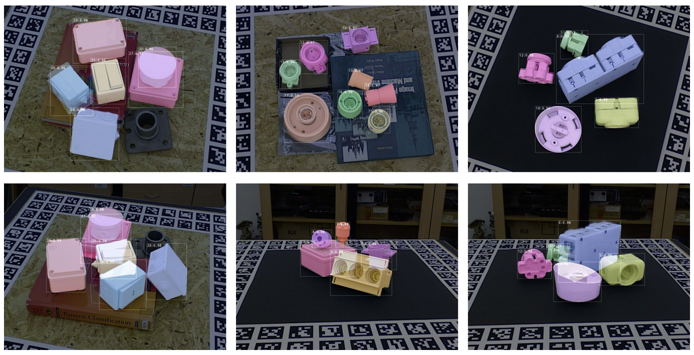


Fig. 11. **Qualitative Result on T-LESS [39].** We render the objects with the estimated pose on top of the original images. The presented confidence scores are from the 2D object detection provided by BOP challenge 2023 [34]. Each column represents a scene captured from a different viewpoint. The visualizations demonstrate that our method is capable of effectively handling severe occlusion in cluttered environment.

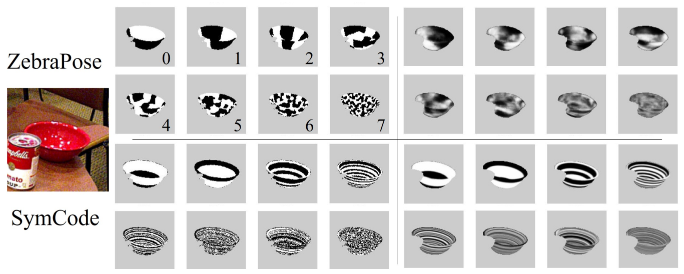


Fig. 12. **Comparison with ZebraPose.** Upper Left: Ground truth of ZebraPose. Upper Right: Predicted output of ZebraPose. Bottom Left: Ground truth of SymCode. Bottom Right: Predicted output of SymCode. Only the first 8 bits are displayed. It appears that SymNet tends to be more confident in its outputs, often producing predictions closer to extreme values of 0 or 1. On the other hand, ZebraPose's predictions tend to be more centered around 0.5, indicating a lower level of confidence.

ACKNOWLEDGMENTS

This research was partially supported by National Key R&D Program of China under Grant 2023YFB4704402, in part by NSFC 62088101 Autonomous Intelligent Unmanned Systems, and in part by Zhejiang Provincial Natural Science Foundation of China under Grant No. LD24F030001 and was partially funded by the EU Horizon Europe Framework Program under Grant Agreement 101058236 (HumanTech) and the German Ministry for Economics and Climate Action (BMWK) under Grant Agreement 13IK010 (TWIN4TRUCKS).

APPENDIX A
RESULTS ON YCB-V

In Table V, we compare our results to other methods on YCB-V dataset. The objects with relatively poor experimental results include 036 wood block, 037 scissors, 024 bowl, and 010 potted meat can. Interestingly, among these objects, 036 and 024 are symmetrical. So, we further visualized the correct and incorrect examples, as seen in Figure 13. We found

TABLE V
BOP RESULTS ON DATASET YCB-V [38]. THE TIME IS THE RUNTIME PER IMAGE AVERAGED OVER THE DATASET.

6D object pose estimation method	Input type	Training type	AR	AR_{VSD}	AR_{MSSD}	AR_{MSPD}	Time(s)
CDPNv2 [26]	RGB	pbr	0.532	0.396	0.570	0.631	0.143
EPOS [33]	RGB	pbr	0.499	0.411	0.464	0.621	0.764
SurfEmb [18]	RGB	pbr	0.647	0.548	0.620	0.773	5.427
ZebraPoseSAT-EffnetB4 [13]	RGB	pbr	**0.691**	**0.607**	**0.686**	**0.780**	0.25
Symnet(Ours)	RGB	pbr	0.653	0.557	0.646	0.758	**0.085**

that under significant occlusion, the output binary codes still exhibit characteristics of a specific pose, although this pose is incorrect. We suspect that this issue arises from insufficient occlusion in the training data, and we suggest to investigate this further in the future.

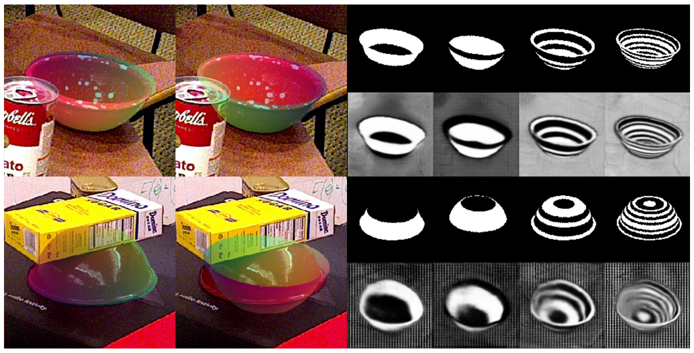


Fig. 13. **Visualization of the YCB-V dataset.** The first column is the ground truth, and the second column shows the estimated poses. The first example is a correctly estimated case, while the second example is one of incorrect estimation. The first and third rows on the right side show the ground truth of SymCode, while the second and fourth rows display the estimations. We found that under significant occlusion, the output binary codes still exhibit characteristics of a specific pose, although this pose is incorrect.

APPENDIX B
DETAILS OF LOSSES

The 6D rotation parameter $\mathbf{R}_{6d}$, as proposed in Zhou et al. [37], is defined as the first two columns of the rotation matrix $\mathbf{R}$:

$$\mathbf{R}_{6d} = [\mathbf{R}_{.1}|\mathbf{R}_{.2}] \tag{10}$$

Given a 6-dimensional vector $\mathbf{R}_{6d} = [\mathbf{r}_1|\mathbf{r}_2]$, the rotation matrix $\mathbf{R} = [\mathbf{R}_{.1}|\mathbf{R}_{.2}|\mathbf{R}_{.3}]$ can be computed as follows:

$$\begin{cases} \mathbf{R}_{.1} = \phi(\mathbf{r}_1) \\ \mathbf{R}_{.3} = \phi(\mathbf{R}_{.1} \times \mathbf{r}_2) \\ \mathbf{R}_{.2} = \mathbf{R}_{.3} \times \mathbf{R}_{.1} \end{cases} \tag{11}$$

We employ a Scale-Invariant representation for Translation Estimation (SITE), as proposed by CDPN [26]. Specifically, considering the size s and center (c_x, c_y) of the detected square bounding box, and the zoom-in size s_{zoom}, the network regresses the scale-invariant translation parameters $t_{\text{SITE}} = [\delta_x, \delta_y, \delta_z]^T$, where

$$\begin{cases} \delta_x = (o_x - c_x) \,/\, s \\ \delta_y = (o_y - c_y) \,/\, s \\ \delta_z = t_z \times s \,/\, s_{\text{zoom}} \end{cases} \tag{12}$$

Here, (o_x, o_y) is the projected 3D centroid in the image and t_z denotes the distance of the object from the camera.

REFERENCES

[1] A. Billard and D. Kragic, "Trends and challenges in robot manipulation," *Science*, vol. 364, no. 6446, p. eaat8414, 2019. [Online]. Available: https://www.science.org/doi/abs/10.1126/science.aat8414

[2] Y. Su, J. Rambach, N. Minaskan, P. Lesur, A. Pagani, and D. Stricker, "Deep multi-state object pose estimation for augmented reality assembly," in *2019 IEEE International Symposium on Mixed and Augmented Reality Adjunct (ISMAR-Adjunct)*. IEEE, 2019, pp. 222–227.

[3] Y. Su, Y. Di, G. Zhai, F. Manhardt, J. Rambach, B. Busam, D. Stricker, and F. Tombari, "OPA-3D: Occlusion-aware pixel-wise aggregation for monocular 3d object detection," *IEEE Robotics and Automation Letters*, vol. 8, no. 3, pp. 1327–1334, 2023.

[4] F. Hagelskjær and A. G. Buch, "PointVoteNet: Accurate object detection and 6 dof pose estimation in point clouds," in *2020 IEEE International Conference on Image Processing (ICIP)*, 2020, pp. 2641–2645.

[5] J. Guo, X. Xing, W. Quan, D.-M. Yan, Q. Gu, Y. Liu, and X. Zhang, "Efficient center voting for object detection and 6d pose estimation in 3d point cloud," *IEEE Transactions on Image Processing*, vol. 30, pp. 5072–5084, 2021.

[6] L. Zou, Z. Huang, N. Gu, and G. Wang, "6D-ViT: Category-level 6d object pose estimation via transformer-based instance representation learning," *IEEE Transactions on Image Processing*, vol. 31, pp. 6907–6921, 2022.

[7] C. Papaioannidis, V. Mygdalis, and I. Pitas, "Domain-translated 3d object pose estimation," *IEEE Transactions on Image Processing*, vol. 29, pp. 9279–9291, 2020.

[8] Y. Lin, Y. Su, P. Nathan, S. Inuganti, Y. Di, M. Sundermeyer, F. Manhardt, D. Stricker, J. Rambach, and Y. Zhang, "HiPose: Hierarchical binary surface encoding and correspondence pruning for rgb-d 6dof object pose estimation," in *Conference on Computer Vision and Pattern Recognition*, 2024.

[9] C. Papaioannidis and I. Pitas, "3D object pose estimation using multi-objective quaternion learning," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 30, no. 8, pp. 2683–2693, 2020.

[10] G. Billings and M. Johnson-Roberson, "SilhoNet: An RGB method for 6d object pose estimation," *IEEE Robotics and Automation Letters*, vol. 4, no. 4, pp. 3727–3734, 2019.

[11] C. Song, J. Song, and Q. Huang, "HybridPose: 6d object pose estimation under hybrid representations," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 431–440.

[12] G. Wang, F. Manhardt, F. Tombari, and X. Ji, "GDR-Net: Geometry-guided direct regression network for monocular 6d object pose estimation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 16 611–16 621.

[13] Y. Su, M. Saleh, T. Fetzer, J. Rambach, N. Navab, B. Busam, D. Stricker, and F. Tombari, "ZebraPose: Coarse to fine surface encoding for 6dof object pose estimation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 6738–6748.

[14] V. Lepetit, F. Moreno-Noguer, and P. Fua, "EPnP: An accurate o(n) solution to the pnp problem," *International Journal of Computer Vision*, vol. 81, pp. 155–166, 2009.

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

[16] K. Park, T. Patten, and M. Vincze, "Pix2Pose: Pixel-wise coordinate regression of objects for 6d pose estimation," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019, pp. 7668–7677.

[17] Y. Di, F. Manhardt, G. Wang, X. Ji, N. Navab, and F. Tombari, "SO-Pose: Exploiting self-occlusion for direct 6d pose estimation," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 12 396–12 405.

[18] R. L. Haugaard and A. G. Buch, "SurfEmb: Dense and continuous correspondence distributions for object pose estimation with learnt surface embeddings," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 6749–6758.

[19] W. Kehl, F. Manhardt, F. Tombari, S. Ilic, and N. Navab, "SSD-6D: Making rgb-based 3d detection and 6d pose estimation great again," in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 1521–1529.

[20] Y. Bukschat and M. Vetter, "EfficientPose: An efficient, accurate and scalable end-to-end 6d multi object pose estimation approach," 2020.

[21] T. Do, T. Pham, M. Cai, and I. Reid, "Real-time monocular object instance 6d pose estimation," in *British Machine Vision Conference 2018*. BMVC Press, 2019.

[22] T.-T. Do, M. Cai, T. Pham, and I. Reid, "Deep-6DPose: Recovering 6d object pose from a single rgb image," *arXiv preprint arXiv:1802.10367*, 2018.

[23] F. Manhardt, W. Kehl, N. Navab, and F. Tombari, "Deep model-based 6d pose refinement in rgb," in *Proceedings of the European Conference on Computer Vision (ECCV)*, September 2018, pp. 800–815.

[24] J. Rambach, C. Deng, A. Pagani, and D. Stricker, "Learning 6dof object poses from synthetic single channel images," in *2018 IEEE International Symposium on Mixed and Augmented Reality Adjunct (ISMAR-Adjunct)*. IEEE, 2018, pp. 164–169.

[25] O. Hosseini Jafari, S. K. Mustikovela, K. Pertsch, E. Brachmann, and C. Rother, "iPose: instance-aware 6d pose estimation of partly occluded objects," in *Computer Vision–ACCV 2018: 14th Asian Conference on Computer Vision, Perth, Australia, December 2–6, 2018, Revised Selected Papers, Part III 14*. Springer, 2019, pp. 477–492.

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

[27] S. Zakharov, I. Shugurov, and S. Ilic, "DPOD: 6d pose object detector and refiner," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2019, pp. 1941–1950.

[28] M. Rad and V. Lepetit, "BB8: A scalable, accurate, robust to partial occlusion method for predicting the 3d poses of challenging objects without using depth," in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 3828–3836.

[29] G. Pitteri, M. Ramamonjisoa, S. Ilic, and V. Lepetit, "On object symmetries and 6d pose estimation from images," in *2019 International conference on 3D vision (3DV)*. IEEE, 2019, pp. 614–622.

[30] Y. Labbé, J. Carpentier, M. Aubry, and J. Sivic, "CosyPose: Consistent multi-view multi-object 6d pose estimation," in *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XVII 16*. Springer, 2020, pp. 574–591.

[31] N. Mo, W. Gan, N. Yokoya, and S. Chen, "ES6D: A computation efficient and symmetry-aware 6d pose regression framework," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 6718–6727.

[32] J. Richter-Klug and U. Frese, "Handling object symmetries in cnn-based pose estimation," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2021, pp. 13 850–13 856.

[33] T. Hodan, D. Barath, and J. Matas, "EPOS: Estimating 6d pose of objects with symmetries," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 11 703–11 712.

[34] T. Hodan, M. Sundermeyer, Y. Labbe, V. N. Nguyen, G. Wang, E. Brachmann, B. Drost, V. Lepetit, C. Rother, and J. Matas, "BOP challenge 2023 on detection segmentation and pose estimation of seen and unseen rigid objects," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, June 2024, pp. 5610–5619.

[35] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[36] L.-C. Chen, G. Papandreou, F. Schroff, and H. Adam, "Rethinking atrous convolution for semantic image segmentation," *arXiv preprint arXiv:1706.05587*, 2017.

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

[38] Y. Xiang, T. Schmidt, V. Narayanan, and D. Fox, "PoseCNN: A convolutional neural network for 6d object pose estimation in cluttered scenes," in *Robotics: Science and Systems (RSS)*, 2018.

[39] T. Hodan, P. Haluza, Š. Obdržálek, J. Matas, M. Lourakis, and X. Zabulis, "T-LESS: An rgb-d dataset for 6d pose estimation of texture-less objects," in *2017 IEEE Winter Conference on Applications of Computer Vision (WACV)*, 2017, pp. 880–888.

[40] A. Doumanoglou, R. Kouskouridas, S. Malassiotis, and T. Kim, "Recovering 6d object pose and predicting next-best-view in the crowd," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. Los Alamitos, CA, USA: IEEE Computer Society, jun 2016, pp. 3583–3592.

[41] M. Sundermeyer, Z.-C. Marton, M. Durner, M. Brucker, and R. Triebel, "Implicit 3d orientation learning for 6d object detection from rgb images," in *Proceedings of the european conference on computer vision (ECCV)*, 2018, pp. 699–715.

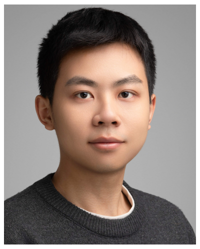
[42] Y. Wen, H. Pan, L. Yang, and W. Wang, "Edge enhanced implicit orientation learning with geometric prior for 6d pose estimation," *IEEE Robotics and Automation Letters*, vol. 5, no. 3, pp. 4931–4938, 2020.

[43] Z. Tian, C. Shen, H. Chen, and T. He, "FCOS: Fully convolutional one-stage object detection," in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, October 2019.

[44] I. Shugurov, S. Zakharov, and S. Ilic, "DPODv2: Dense correspondence-based 6 dof pose estimation," *IEEE transactions on pattern analysis and machine intelligence*, vol. 44, no. 11, pp. 7417–7435, 2021.

[45] P. Castro and T.-K. Kim, "CRT-6D: Fast 6d object pose estimation with cascaded refinement transformers," in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, 2023, pp. 5746–5755.

C Biography Section



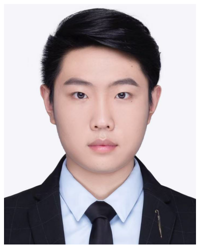
Yongliang Lin Yongliang Lin is a Ph.D. candidate at Zhejiang University, China, supervised by Prof. Yu Zhang. His research primarily focuses on computer vision, particularly object pose estimation and its applications. He has a strong interest in robotics and has interned as a robotics algorithm developer at Microsoft Research Asia, Noetix Robotics, and CoreNetic. He also serves as a reviewer for prestigious conferences and journals, including CVPR, ECCV.

Yongzhi Su Yongzhi Su is a Ph.D. candidate at the University of Kaiserslautern, supervised by Prof. Didier Stricker and Dr. Jason Rambach. His work primarily focuses on computer vision and deep learning, with a particular emphasis on object pose estimation and its applications. During his doctoral studies, he was also a visiting researcher at TU Munich, where he worked under the supervision of Prof. Dr. Nassir Navab and PD Dr. Federico Tombari. Beyond his research, he actively contributes to the academic community by serving as a teaching assistant, mentoring students in projects and theses. He also serves as a reviewer for prestigious conferences and journals, including CVPR, ICCV, IROS, RA-L, and ISMAR.

Sandeep Inuganti Sandeep Inuganti is a Researcher at DFKI and a Ph.D. student at the Department of Computer Science in RPTU Kaiserslautern. Previously he received his M.S. degree from the University of Tuebingen, Germany and Bachelors from Indian Institute of Technology Dhanbad, India. In DFKI he is a team member of “Spatial Sensing and Machine Perception”. His research interests lie in 3D Scene Understanding, particularly 6DoF Object Pose Estimation and Tracking.

Yan Di Yan Di is a Ph.D. candidate in the Department of Computer Science at the Technical University of Munich. His research interests include object pose estimation and its applications in SLAM, robotics, and other downstream visual tasks. He serves as a reviewer for several leading conferences, including ICCV, ECCV, and CVPR.

Naeem Ajilforoushan Naeem Ajilforoushan is currently pursuing a Ph.D. in Control Science and Engineering at Zhejiang University, China, under the supervision of Prof. Zhang. His research focuses on improving control strategies using Reinforcement Learning with Model Predictive Control (MPC). He received his M.Sc. in Mechatronics and B.Sc. in Control Science and Engineering from Iran. His expertise includes advanced control systems, robotics, and machine learning, with experience in UAVs, UGVs, and robotic manipulators.



Hanqing Yang Hanqing Yang received the B.S. degree from Sichuan University in 2019, and the Ph.D. degree from Zhejiang University in 2024. He is currently serving as a Senior Algorithm Engineer at Alibaba Group. His research interests encompass object perception, few-shot learning, and multimodal foundation models.

Yu Zhang Yu Zhang received his B.S. degree in information engineering from Xi’an Jiaotong University, Xi’an, China, in 2003, and his M.S. and Ph.D. degrees in computer science from Tsinghua University, Beijing, China, in 2009. He was a postdoctoral researcher at Tsinghua University from 2009 to 2011 and a visiting scholar at Carnegie Mellon University from 2013 to 2014. He is now a professor in the College of Control Science and Engineering at Zhejiang University, China. His research interests include visual navigation, intelligent control, computer vision and intelligent autonomous systems. He has published more than 70 papers indexed by SCI and EI, mainly in international journals and conferences such as IEEE TPAMI, IEEE TIE, IEEE RAL, CVPR, ICRA, IROS, AAAI, NeurIPS, ICCV, ECCV, etc.

Jason Rambach Jason Rambach received his PhD in Computer Science in 2020 from the University of Kaiserslautern for his dissertation entitled “Learning Priors for Augmented Reality Tracking and Scene Understanding”. Currently, he is a Senior Researcher at DFKI leading the team “Spatial Sensing and Machine Perception” tackling Geometric/Semantic scene understanding using Machine Learning.

His research interests include SLAM and Semantic Scene Understanding, Object Pose Estimation and Tracking, Anomaly Detection, Robotic Vision and Augmented Reality. He has over 50 publications in leading Computer Vision and Augmented Reality conferences, a best paper award from the IEEE International Symposium on Mixed and Augmented Reality (ISMAR) 2017 and five awards at the BOP Object Pose Estimation challenge 2022 and 2023 at ECCV and ICCV. He is a reviewer for several scientific journals and conferences (IEEE PAMI, IEEE TIP, CVPR, ICCV, ECCV, ICRA, WACV, BMVC).