International Journal of Computer Vision (2026) 134:212
https://doi.org/10.1007/s11263-026-02770-x

Towards Symmetry-sensitive Pose Estimation: A Rotation Representation for Symmetric Object Classes

Andreas Kriegler[1,2] · **Csaba Beleznai**[1] · **Margrit Gelautz**[2]

Received: 16 December 2024 / Accepted: 26 January 2026

Abstract

Symmetric objects are common in daily life and industry, yet their inherent orientation ambiguities that impede the training of deep learning networks for pose estimation are rarely discussed in the literature. To cope with these ambiguities, existing solutions typically require the design of specific loss functions and network architectures or resort to symmetry-invariant evaluation metrics. In contrast, we focus on the numeric representation of the rotation itself, modifying trigonometric identities with the degrees of symmetry derived from the objects' shapes. We use our representation, SARR, to obtain canonic (symmetry-resolved) poses for the symmetric objects in two popular 6D pose estimation datasets, T-LESS and ITODD, where SARR is unique and continuous w.r.t. the visual appearance. This allows us to use a standard CNN for 3D orientation estimation whose performance is evaluated with the symmetry-sensitive cosine distance AR_C. Our networks outperform the state of the art using AR_C and achieve satisfactory performance when using conventional symmetry-invariant measures. Our method does not require any 3D models but only depth, or, as part of an additional experiment, texture-less RGB/grayscale images as input. We also show that networks trained on SARR outperform the same networks trained on rotation matrices, Euler angles, quaternions, standard trigonometrics or the recently popular 6d representation – even in inference scenarios where no prior knowledge of the objects' symmetry properties is available. Code and a visualization toolkit are available at https://github.com/akriegler/SARR.

Keywords Symmetry · Rotation representation · Pose estimation · Evaluation metrics

1 Introduction

6D pose estimation is a prerequisite to enable robot grasping and manipulation tasks such as bin picking, as well as virtual and augmented reality applications. With the adoption of deep learning approaches and the usage of large training datasets, significant progress has been made for object pose estimation (Chen et al., 2023; He et al., 2022; Irshad et al., 2022; Liu et al., 2021; Pitteri et al., 2021; Periyasamy et al., 2022; Pitteri et al., 2019), yet the challenges related to rotationally symmetric objects, specifically the ambiguities that arise, are rarely discussed thoroughly.

It is sometimes possible to resolve these ambiguities by using symmetry-breaking visual features from the RGB domain such as non-symmetric color patterns like distinctive design elements on manufactured goods. At the same time, an important motivation of our work is that texture-less objects are common in industrial or robotics applications (Drost et al., 2017; Hodaň et al., 2017; Kalra et al., 2024). Furthermore, depth cameras or 3D LiDAR pointclouds are common modalities in robotics (Raj et al., 2022). Thus, we focus on the challenging task of object pose estimation where *geometric* symmetries are intrinsic properties due to an object's shape (Brégier et al., 2018; Periyasamy et al., 2022) and assume no symmetry-breaking RGB texture to help resolve the ambiguities.

These ambiguities are prohibitive when optimizing a network to estimate the orientation (Hara et al., 2017), since the bijec-

Communicated by Jian Sun.

✉ Andreas Kriegler
andreas.kriegler@tuwien.ac.at

Csaba Beleznai
csaba.beleznai@ait.ac.at

Margrit Gelautz
margrit.gelautz@tuwien.ac.at

[1] Assistive and Autonomous Systems, AIT Austrian Institute of Technology, Giefinggasse 4, 1210 Vienna, Austria

[2] Visual Computing and Human-Centered Technology, TU Wien, Favoritenstraße 9-11, 1040 Vienna, Austria

Published online: 07 April 2026

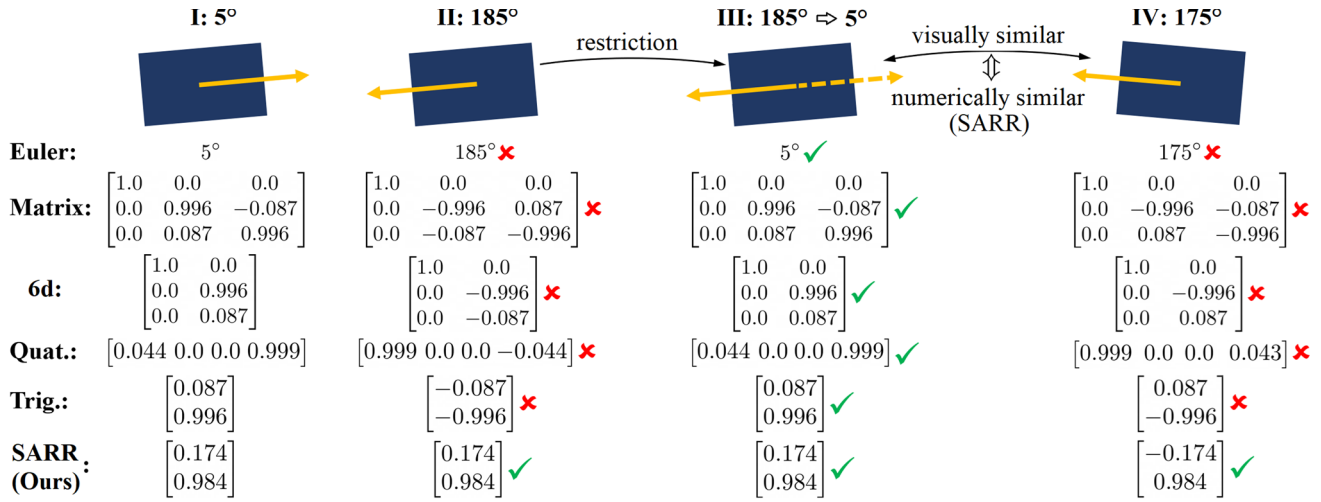


Fig. 1 Comparing rotation representations on a symmetric shape. Using a rotation of 5° as reference (I), rotating the rectangle by another 180° creates ambiguities due to visual equality but numeric differences (II). A space restriction can resolve this, numerically treating the object as if it were rotated by 5° (dashed arrow) (III). *Yet this does not ensure continuity for visually similar poses close to the restriction* boundary (*175° ↔ 5°*) (IV). Contrarily, our formulation is both unique across the rotation space and continuous across the symmetry boundary transition. ✓ marks a correct/similar representation, ✗ is incorrect. "6d" denotes the 6d representation proposed by Zhou et al. (2019)

tive relation between the *visual* representation $\mathcal{V}(\boldsymbol{p})$, e.g. an image of an object in pose $\boldsymbol{p}$, and the corresponding *numeric* rotation representation $\mathcal{N}(\boldsymbol{p})$, e.g. a rotation matrix, no longer holds: multiple orientations represent the same visual input and the resulting multi-modal distribution cannot be learned unambiguously. To better understand the problems raised by symmetric objects, let us consider the rectangles in columns I and II of Fig. 1. For other rotation representations – we consider Euler angles, rotation matrices, the 6d representation (Zhou et al., 2019), quaternions and trigonometric identities – two numerically different representations apply to visually identical poses $\boldsymbol{p}_{\mathrm{I}}$, $\boldsymbol{p}_{\mathrm{II}}$:

$$\mathcal{V}(\boldsymbol{p}_{\mathrm{I}}) = \mathcal{V}(\boldsymbol{p}_{\mathrm{II}}) \text{ but } \mathcal{N}(\boldsymbol{p}_{\mathrm{I}}) \neq \mathcal{N}(\boldsymbol{p}_{\mathrm{II}}). \tag{1}$$

While the rectangles are visually indistinguishable, "physically" they are oriented differently (marked by the yellow arrow) and thus have different numeric representations. There is therefore no function $\mathcal{F}$: $\mathcal{V}(\boldsymbol{p}) \mapsto \mathcal{N}(\boldsymbol{p})$ for all possible $\boldsymbol{p} \in \mathrm{SO}(3)$. As Pitteri et al. (2019) state, an attempt to learn such a function with a Machine Learning (ML) algorithm such as a Convolutional Neural Network (CNN) would fail for many loss functions, as the network converges naively to the mean of possible poses (if at all). This is true not only for rotation matrices but also other rotation representations (Huynh, 2009).

Current pose estimation methods typically attempt to solve this problem in two ways: Firstly, the network architectures and/or loss functions are modified to resolve ambiguities. This was originally proposed by Pitteri et al. (2019), who split the rotation space into multiple bins, train a regression network for each bin as well as a classifier to pick the regressor to invoke. A disadvantage of this discretization approach is that it requires multiple networks and thus extends training times. Secondly, evaluation often relies on metrics that are symmetry-invariant; i.e. metrics that give a full score as long as the network predicts 1 out of the n correct poses. Examples include the VSD, MSSD and MSPD metrics (Hodaň et al., 2020), which have become standard for pose estimation through the BOP benchmark (Hodaň et al., 2018). This means there is less incentive for methods to consider the ambiguities that stem from object symmetries - something we consider problematic, in line with Bregier et al. (2017)'s reluctance to rely on ambiguity-invariant error functions.

A third approach deals with the representation $\mathcal{N}$ itself by restricting the rotation space to only the canonic object poses (Rad & Lepetit, 2017), see column III of Fig. 1. The angle is clamped to the canonic space, numerically setting the orientation as marked by the dashed arrow. This restores uniqueness, i.e.:

$$\mathcal{V}(\boldsymbol{p}_{\mathrm{I}}) = \mathcal{V}(\boldsymbol{p}_{\mathrm{III}}) \Leftrightarrow \mathcal{N}(\boldsymbol{p}_{\mathrm{I}}) = \mathcal{N}(\boldsymbol{p}_{\mathrm{III}}), \tag{2}$$

however the mathematical properties and consequences were not analysed in detail in that work. Most obviously, any representation built on angles from this canonic space is discontinuous across the boundary of the space, e.g. $180° \pm \epsilon \leftrightarrow 0° \pm \epsilon$ for the shown rectangle. We take inspiration from their approach as we explore these object symmetries more formally and propose the **S**ymmetry-**A**ware-**R**otation-**R**epresentation (SARR), which implements a space restriction while remaining continuous across the boundary (col-

umn IV in Fig. 1). We evaluate SARR using the symmetry classes of the texture-less objects in the T-LESS (Hodaň et al., 2017) and ITODD (Drost et al., 2017) datasets. These have, as part of the BOP benchmark suite, become highly popular choices for evaluating object pose estimation methods. Our contributions are therefore:

1. We formally describe the ambiguities due to object symmetries, targeting the symmetries and rotations of T-LESS and ITODD objects
2. We propose SARR, a rotation representation that is unique and continuous for those objects
3. We present rotation estimation results, comparing to methods from the benchmark and networks trained on common representations

After reviewing related works on symmetry and pose estimation in Sect. 2, we provide a formal problem description and introduce the SARR representation in Sect. 3. We present extensive evaluation results on T-LESS and ITODD in Sect. 4, before concluding in Sect. 5.

2 Related Work

Pose estimation methods for symmetric objects can be split into two families (Periyasamy et al., 2022), a categorization we also follow. Methods from the first category typically estimate only one pose (2.1), whereas methods from the second category estimate the full distribution of valid poses (2.2). We finish with a brief discussion on the desirable properties of rotation representation for machine learning tasks (2.3).

2.1 Single Valid Pose

Rad and Lepetit (2017)'s work was one of the first to explicitly consider the problem of ambiguities due to symmetries. For their solution, they halve the rotation space of an object with a symmetry of degree 2 and train a classifier to detect if a given object falls outside this restricted space. If so, the image and predicted pose are mirrored. For higher degrees of symmetry, a similar discretization of the viewing sphere was done by Corona et al. (2018). Pitteri et al. (2019) provide an analytic solution and implement a mapping algorithm for ambiguous rotations. In their work multiple regressors have to be trained dependent on an object's symmetry. A classifier is additionally required, which gets invoked to choose the correct regressor. Mapping to a canonic pose and introducing empirically set balance parameters was done in Chen et al. (2021), with four such parameters required to control network training.

Morrison et al. (2020) have recognized the usefulness of the trigonometric representation to learn a grasping angle which is symmetric at $\pm\frac{\pi}{2}$. They restrict the angle to $[-\frac{\pi}{2}, \frac{\pi}{2}]$ and compute the vector $\boldsymbol{v} = (\sin 2\theta, \cos 2\theta)$ for network training. The same approach was taken by Ayoub et al. (2023) for grasping tree-logs. Nevertheless, only one axis of symmetry was considered in both of these works that use the trigonometric representation.

Another popular idea is to use a specific distance metric (Labbé et al., 2020; Mo et al., 2022; Shi et al., 2021; Wang et al., 2019; Zhao et al., 2023), implemented via a custom loss that fits closely to the evaluation metric. For example, Mo et al. (2022) proposed the symmetry-invariant metric A(M)GPD (average (maximum) grouped primitives distance) which we also include in our experiments for evaluation. Shi et al. (2021) had to train two separate networks, one for symmetric objects and one for asymmetric objects, each with a proper loss.

It is also possible to implicitly learn representations via an Autoencoder, as was done by Sundermeyer et al. (2018, 2020). The learned latent representation is (presumably) ambiguity invariant, as visually identical poses will map to the same code. Unfortunately, their method relies strongly on accurate 3D models. Haugaard et al. (2023) more recently proposed a way to learn 2D-3D correspondence distributions from color images without prior knowledge about symmetries. The authors admit that these distributions are accurate only "up to" symmetry ambiguities and postulate that such global symmetries could be modeled explicitly. In our work we specifically propose such a solution to model the symmetries of T-LESS and ITODD objects with a well-defined rotation representation.

2.2 Distribution of Pose Hypotheses

Methods of this family estimate not just one pose, but the full set of potential poses according to the object's symmetry. This is commonly done through correspondence matching with a probabilistic approach, where multiple likelihoods for the pose hypotheses are estimated. For example, Zhao et al. (2023) learn many-to-many correspondences in two input point clouds, or Li et al. (2024) use a render-and-compare approach and soft labels for the classification task. The possible distribution of 3D orientations was learned in the form of a SO(3) encoder in (Cai et al., 2022). As Li et al. (2024) mention, the performance of matching-based methods can degrade significantly for imperfect CAD models. As these methods do not resolve to a single canonic pose, they have to resort to reporting results using a symmetry-invariant metric. In contrast, we train our networks without using any 3D models and report results using a symmetry-sensitive metric.

2.3 Desired Properties of a Rotation Representation

While *uniqueness* may be sufficient for analytic settings, for optimization algorithms such as deep neural networks, the target-variables and thus the target function $\mathcal{F}$ should also be *continuous*. This continuity property is desirable, as discontinuities can inhibit learning (Hara et al., 2017). Xu and Cao (2005, 2004) have shown that functions that are smoother or have stronger continuity properties also have lower approximation errors for a given number of neurons. None of the common representations fulfill this, as they do not consider the symmetries of an object but only its orientation in the form of an abstract, three-dimensional Cartesian coordinate system.

To summarize, the works most closest to ours include Pitteri et al. (2019)'s mapping to canonic poses, but our proposed SARR requires only a single regressor and preserves continuity, as well as the modified trigonometric representations used by Morrison et al. (2020) and Ayoub et al. (2023), but we generalize their concept to multiple axes of symmetry and higher degrees of symmetry.

3 Methodology

We first formally describe the notions of *uniqueness* and *continuity* (3.1) before reviewing the BOP datasets and selecting T-LESS and ITODD for our analysis (3.2). For the remainder of Section 3 we focus on the symmetric objects in T-LESS (3.3)[1]. We then present our rotation representation SARR (3.4) and its inverse mapping (3.5). The section concludes with a visual validation of our representation (3.6) and a discussion of its limitations (3.7).

3.1 Problem Description

Types of symmetries include translational, reflectional or rotational symmetry in 2D (see Stewart (2013), pg. 3). Yet in 3D, a reflectional symmetry can be understood as rotational symmetry and translational symmetry is less relevant for Computer Vision, since translationally symmetric objects (e.g. a long building with identical parts) are uncommon and CNNs are generally accepted to be translation equivariant (Lenc & Vedaldi, 2019). We therefore limit our discussion to geometric symmetries that arise purely from rotations. We refer to the finite set of symmetric poses of an object as "discrete" and use "continuous" for the infinite circular symmetries such as those of a cone or sphere. Note that a cylinder actually has both discrete and continuous symmetries, as we define this property per axis. For the subsequent task of pose estimation, since we focus on rotational symmetry, we disregard object translation. That is, we analyse the 3D orientation of objects and not their full 6D pose, i.e. assume $\boldsymbol{p} \in \mathrm{SO}(3)$. Then, let $\mathcal{V}(\boldsymbol{p})$ be the visual representation of an arbitrary object in pose $\boldsymbol{p}$, for example a depth image $\boldsymbol{d} \in \mathbb{R}^{w \times h}$ (or an RGB image of a texture-less object). An object is said to be symmetric if there exist one or more rigid motions $\boldsymbol{r}$ which, if applied to the object pose, do not change the appearance of the object (Pitteri et al., 2019). In this case there exists a non-empty set $\mathcal{R}$:

$$\begin{aligned}\mathcal{R} = \{&\boldsymbol{r} \in \mathrm{SO}(3) \setminus I_3 \text{ s.t.} \\ &\forall \boldsymbol{p} \in \mathrm{SO}(3)\colon \mathcal{V}(\boldsymbol{p}) = \mathcal{V}(\boldsymbol{r} \cdot \boldsymbol{p}).\end{aligned} \tag{3}$$

Unlike Pitteri et al. we exclude the identity motion I_3 as otherwise every object could be considered symmetric. For a rotation representation $\mathcal{N}$ to be considered unique, i.e. to resolve the ambiguities (I and II in Fig. 1), we require that:

$$\mathcal{V}(\boldsymbol{p}) = \mathcal{V}(\boldsymbol{r} \cdot \boldsymbol{p}) \Leftrightarrow \mathcal{N}(\boldsymbol{p}) = \mathcal{N}(\boldsymbol{r} \cdot \boldsymbol{p}). \tag{4}$$

Operating on the numerical representation level, this can be done by restricting the rotations to the space of only canonic object poses $C \subset \mathrm{SO}(3)$. Such a restriction is both necessary and sufficient to satisfy Eq. (3). However, this effectively disregards all object symmetries and treats objects as non-symmetrical. An obvious drawback is the introduced discontinuity: objects near the symmetry boundary look visually similar (due to their symmetry) yet their numerical representations are drastically different (due to the clamping). More formally, we require that for small rotations $\boldsymbol{\epsilon}$ which result in small visual changes, the numeric representations must only differ slightly - within the canonic space *and across its boundary* - to be considered continuous:

$$\mathcal{V}(\boldsymbol{p}|_C) \approx \mathcal{V}(\boldsymbol{\epsilon} \cdot \boldsymbol{r} \cdot \boldsymbol{p}|_C) \Leftrightarrow \mathcal{N}(\boldsymbol{p}|_C) \approx \mathcal{N}(\boldsymbol{\epsilon} \cdot \boldsymbol{r} \cdot \boldsymbol{p}|_C). \tag{5}$$

Pitteri et al. (2019) partially solve this problem by training multiple regressors and an additional classifier. Each regressor is continuous within its respective domain C_n and invoked via the classifier prediction. They then evaluate their method on T-LESS objects. Our method requires no space discretization and we evaluate not only on the T-LESS dataset.

3.2 Symmetries in BOP Datasets

The BOP benchmark features 17 datasets (at the time of writing) with numerous objects – column # in Table 1 gives the total number of objects. For every dataset i, let $\mathcal{S}_i$ denote the set of its symmetric objects, i.e. all objects with non-empty $\mathcal{R}$. $\mathcal{S}_i$ itself was never published by the BOP organizers. Organizers did identify $\mathcal{S}_i$ – by using the HALCON software and calculating the Hausdorff distance h between vertices of

[1] See Appendix A for ITODD objects and 3D primitives.

Table 1 BOP datasets overview

Dataset	#	$\lvert\mathcal{S}'_i\rvert$	$\lvert\mathcal{S}_{i,t}\rvert$	u	GT_V	GT_T
HANDAL (Guo et al., 2023)	40	7	7	2	✓	✗
HB (Kaskman et al., 2019)	33	5	3	2	✓	✗
HOPE (Tyree et al., 2022)	28	28	0	0	✓	✗
HOT3D (Banerjee et al., 2025)	33	21	10	6	✓	✗
IC-BIN (Doumanoglou et al., 2016)	2	1	1	1	✗	✓
IC-MI (Tejani et al., 2014)	6	3	2	1	✗	✓
IPD (Kalra et al., 2024)	10	5	5	5	✓	✓
ITODD (Drost et al., 2017)	28	18	18	11	✓	✗
ITODD-MV (Drost et al., 2017)	28	18	18	11	✓	✗
LM (Hinterstoisser et al., 2013)	15	3	3	2	✗	✓
LM-O (Brachmann et al., 2014)	8	2	2	1	✗	✓
RU-APC (Rennie et al., 2016)	14	8	1	1	✗	✓
T-LESS (Hodaň et al., 2017)	30	27	27	5	✗	✓
TUD-L (Hodaň et al., 2018)	3	0	0	0	✗	✓
TYO-L (Hodaň et al., 2018)	21	13	8	2	✗	✓
XYZ-IBD (Huang et al., 2025)	15	9	9	5	✓	✓
YCB-V (Calli et al., 2017)	21	14	7	5	✗	✓

an object model in the canonical and transformed locations, identifying a symmetry if $h < \max(15mm, 0.1d)$, where d is the diameter of the model[2] – yet $\mathcal{S}_i$ is not made public[3]. We estimated the sets $\mathcal{S}'_i$ instead via visual inspection of the CAD models. Column $|\mathcal{S}'_i|$ gives the size of these sets, i.e. the total number of symmetric objects. Different from our approach of considering symmetries defined solely by an object's geometry, the BOP organizers used texture to derive $\mathcal{S}_{i,t} \subseteq \mathcal{S}_i$. Specifically, $\mathcal{S}_{i,t}$ consists of objects with "symmetry transformations that cannot be resolved by the model texture"[4]. For some datasets such as HOPE, which contains objects with highly distinctive textures, this removes the majority of objects from consideration (column $|\mathcal{S}_{i,t}|$). Nevertheless, we consider $\mathcal{S}_{i,t}$ as the starting point for our analysis, as to not deviate from established conventions ($\mathcal{S}_{i,t}$ is provided with every dataset). For every dataset, column $u := u(\mathcal{S}_{i,t})$ gives the number of unique symmetry classes amongst all the objects in that dataset. Observing columns $|\mathcal{S}_{i,t}|$ and u, T-LESS, ITODD, ITODD-MV (ITODD with additional multi-view images), HOT3D, XYZ-IBD and TYO-L all feature a large and diverse set of objects and symmetries. We disregard TYO-L, because the leaderboard has been inactive since 2019, HOT3D, because only RGB fisheye image streams are available as a modality (no depth) and XYZ-IBD, because training images were rendered with CAD models whose object origins are defined differently than those from the test set, causing a mismatch. Instead we focus on T-LESS and ITODD, both highly active datasets on the leaderboard. T-LESS has ground-truth 6D pose labels available for its test set (column GT_T), whereas these labels are only provided for the validation set in ITODD (column GT_V).

3.3 Symmetries in T-LESS

Mo et al. (2022) categorized the symmetric objects in T-LESS and YCB-V following the Hausdorff distance of model vertices, using a clustering algorithm for simplification while relying on a relaxation-threshold for continuous symmetries. We also classify the symmetries of T-LESS objects (see Table 2), but use the original BOP information without any extra parameters to resolve continuous symmetries. All 30 objects are split into one of 5 symmetry classes $i \in \{\text{I, II, III, IV, V}\}$ following u. Symmetry class I includes all non-symmetric objects. Class II and III are for objects with two-fold and four-fold discrete symmetry about z, respectively. Class IV contains all objects with continuous symmetry about z, while class V exhibits a two-fold symmetry about y. This leads to another way of expressing an object's symmetry properties using the symmetry-vectors $\boldsymbol{\kappa}_i = [\boldsymbol{\kappa}_{i,\alpha}, \boldsymbol{\kappa}_{i,\beta}, \boldsymbol{\kappa}_{i,\gamma}]$, made up of the degrees of symmetry per object axis. For class II, $\boldsymbol{\kappa}_{\text{II}} = [1, 1, 2]$ can be read as: "within a full 360° rotation about z, there are two poses that are visually identical ($\gamma = 0°, \gamma = 180°$); only one such pose exists for axes x and y (no symmetry)". These 30 T-LESS objects appear in a rotation space that is already restricted to partially eliminate duplicate images due to sym-

[2] See 7.2 at https://bop.felk.cvut.cz/challenges/bop-challenge-2019/ (Accessed: 2026-01-12)

[3] See https://github.com/thodan/bop_toolkit/issues/50 (Accessed: 2026-01-12).

[4] See 7.2 at https://bop.felk.cvut.cz/challenges/bop-challenge-2019/ (Accessed: 2026-01-12).

Table 2 Symmetry classes of T-LESS objects. Each of the five symmetry classes has a representation SARR_i, vector $\boldsymbol{\kappa}_i$, a symmetry type, a set of rotations $\mathcal{R}_i$ and a rotation space T_i. T-LESS class IDs of the visualized objects are indicated **bold** in the bottom row. The default 3D coordinate frame, as shown for class II, is oriented the same for all objects

Class	I	II	III	IV	V
SARR_i	SARR_I	SARR_II	SARR_III	SARR_IV	SARR_V
$\boldsymbol{\kappa}_i$	$\boldsymbol{\kappa}_\text{I} = [1, 1, 1]$	$\boldsymbol{\kappa}_\text{II} = [1, 1, 2]$	$\boldsymbol{\kappa}_\text{III} = [1, 1, 4]$	$\boldsymbol{\kappa}_\text{IV} = [1, 1, \infty]$	$\boldsymbol{\kappa}_\text{V} = [1, 2, 1]$
Type	None	Discrete	Discrete	Continuous	Discrete
$\mathcal{R}_i$	{}	$\boldsymbol{R}_z^\gamma \vert \gamma \in \{\pi\}$	$\boldsymbol{R}_z^\gamma \vert \gamma \in \{\frac{\pi}{2}, \pi, \frac{3\pi}{4}\}$	$\boldsymbol{R}_z^\gamma \vert \gamma \in \{\mathbb{R}\}$	$\boldsymbol{R}_y^\beta \vert \beta \in \{\pi\}$
T_i	T	T	T	T	T_1
T-LESS IDs	**21**, 22, 18	**11**, 5, 6, 7, 8, 9, 10, 12, 25, 26, 28, 29	**27**	**2**, 17, 1, 3, 4, 13, 14, 15, 16, 24, 30	**23**, 19, 20

metry (Hodaň et al., 2016). To analyse this, we build up the rotation space $T \subset \text{SO}(3)$ of the T-LESS training set. Image acquisition was divided into two steps, where rotations $\alpha_1 \in \{5, 15, \ldots, 85\}°$, $\beta_1 = 0°$, $\gamma_1 \in \{0, 5, \ldots, 355\}°$ and $\alpha_2 \in \{275, 285, \ldots, 355\}°$, $\beta_2 = 0°$, $\gamma_2 \in \{0, 5, \ldots, 355\}°$ describe all object poses $\boldsymbol{p}|_T$: $T_1 = \{\boldsymbol{R}_{z''}^{\gamma_1} \boldsymbol{R}_{y'}^{\beta_1} \boldsymbol{R}_x^{\alpha_1}\}$ and $T_2 = \{\boldsymbol{R}_{z''}^{\gamma_2} \boldsymbol{R}_{y'}^{\beta_2} \boldsymbol{R}_x^{\alpha_2}\}$, with $T = T_1 \cup T_2$[5]. T represents a "full view sphere" (Hodaň et al., 2016) of these objects with $|T| = 1296$ training images per object class. The jump of 180° in α was not done for two objects, IDs 19, 20, meaning for those $T = T_1$[6]. The fact that $\boldsymbol{p}|_T$ still includes ambiguous poses, and that the default T-LESS annotations do not satisfy Eq. (4) let alone Eq. (5), can be verified by looking at the images in Fig. 3; it also becomes apparent when considering, for example, that $\gamma \in \{0, 5, \ldots, 355\}°$ for continuously symmetric objects of class IV.

3.4 SARR Representation

To motivate our representation SARR, let us consider object 11 from symmetry class II (see Table 2) and only a rotation about axis z. We can then imagine some function f (for example a perceptual hash) that computes the similarity between the visual representations of the object in its default, unrotated pose and a second, rotated pose: $f(\mathcal{V}(\boldsymbol{p}), \mathcal{V}(\boldsymbol{R}_z^\gamma \cdot \boldsymbol{p}))$. It follows that f is periodic, being maximal at symmetrical poses, i.e. where $\gamma \in \{0, \pi, \ldots, \frac{2n\pi}{\kappa_{\text{II},\gamma}}\} \equiv \{n\pi\}, n \in \mathbb{Z}$. Similarly, f is minimal exactly halfway, where the object poses are orthogonal. Considering f instead of only the set $\mathcal{R}_\text{II}$, one can derive that the numeric representation should be designed to show a similar periodic behavior. This is both intuitive for humans to understand and desirable from an optimization standpoint, as it does not introduce any discontinuities since f is smooth. Analogous arguments can be made for all symmetry classes of T-LESS: the degree of symmetry for every axis defines the frequency of f for rotations about that axis. For example, the frequency of f for class III and z rotations is $\frac{2n\pi}{4}$ since $\boldsymbol{\kappa}_{\text{III},\gamma} = 4$.

We propose a rotation representation that implements such periodic behaviour by modifying trigonometric identities $\boldsymbol{v} = (\sin\theta, \cos\theta)$, where $\boldsymbol{v}$ represents a point on the unit circle as seen in (Hara et al., 2017). To this end, we formalize and generalize the concept used in (Ayoub et al., 2023; Morrison et al., 2020). In contrast to those works, SARR is formulated generically and can be used to treat different axes of symmetry (y, z) and different degrees of symmetry. For the trigonometric representation $\mathcal{N}_\text{trig}$ the composition of trigonometric identities $\boldsymbol{v}$ of angles α, β, γ defines the orientation of an object:

$$\mathcal{N}_\text{trig}(\alpha, \beta, \gamma) = \begin{bmatrix} \sin\alpha & \sin\beta & \sin\gamma \\ \cos\alpha & \cos\beta & \cos\gamma \end{bmatrix}. \tag{6}$$

Combining Eq. (6) with the discussion on f, we can define the rotation representation SARR_i for all objects of all five symmetry classes in T-LESS as:

[5] Intrinsic rotations in "XYZ"-order: $\boldsymbol{R}_x^\alpha$ then $\boldsymbol{R}_{y'}^\beta$ and $\boldsymbol{R}_{z''}^\gamma$.

[6] The combined α and γ rotations effectively implement a 180° flip about that object's symmetry axis, axis y, since objects 19 and 20 belong to symmetry class V. According to Hodan et al.'s own definition, object 23 also belongs to class V, yet images of object 23 in T_2 rotations do exist. We consider this an oversight, as images from T_1 and T_2 are visually identical (ignoring texture/lighting changes), and thus disregard these T_2 images.

$$\mathrm{SARR}_i(\alpha|_{T_i}, \beta|_{T_i}, \gamma|_{T_i}) = \begin{bmatrix} s_{i,\alpha} & s_{i,\beta} & s_{i,\gamma} \\ c_{i,\alpha} & c_{i,\beta} & c_{i,\gamma} \end{bmatrix} := \begin{bmatrix} \sin(\kappa_{i,\alpha}\alpha|_{T_i}) & \sin(\kappa_{i,\beta}\beta|_{T_i}) & \sin(\lambda_i\gamma|_{T_i}) \\ \cos(\kappa_{i,\alpha}\alpha|_{T_i}) & \cos(\kappa_{i,\beta}\beta|_{T_i}) & \cos(\lambda_i\gamma|_{T_i}) \end{bmatrix}. \tag{7}$$

To resolve the continuous symmetry of class IV, we define

$$\lambda_i = \begin{cases} 0 & \text{if } i == \mathrm{IV} \\ \kappa_{i,\gamma} & \text{otherwise.} \end{cases} \tag{8}$$

This formulation allows extension to other object symmetries and datasets. Specifically, the representation for symmetry classes of ITODD objects and 3D primitives is presented in Appendix A.

3.5 SARR Inverse Mapping

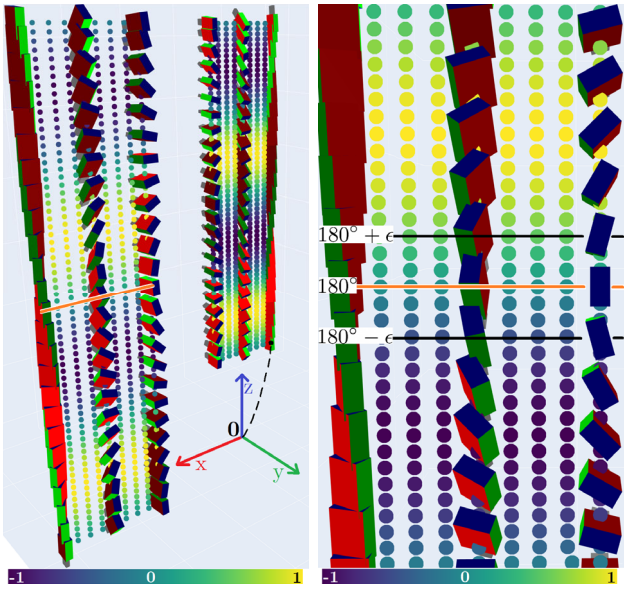
While SARR is designed for machine learning tasks, evaluation and visualization requires an inverse mapping to Euler angles $\alpha_i|_C$, $\beta_i|_C$ and $\gamma_i|_C$, from which other representations such as rotation matrices can be derived:

$$\theta_i|_C = \begin{cases} 0 & \text{if } i == \mathrm{IV}, \\ \frac{1}{\kappa_{i,\theta}}(2\pi - \arccos(c_{i,\theta})) & \text{if } s_{i,\theta} < 0, \\ \frac{1}{\kappa_{i,\theta}}\arccos(c_{i,\theta}) & \text{otherwise,} \end{cases} \tag{9}$$

for $\theta \in \{\alpha, \beta, \gamma\}$ (angles are in radians). Angles $\alpha_i|_C$, $\beta_i|_C$, $\gamma_i|_C$ are now restricted to the subspace $\cdot|_C$ and any representations derived from them do not have the continuity property of SARR, but are unique since they define canonic object poses and are thus pertinent for evaluation. Any permutations of these rotations that look visually identical are resolved. Algorithm 1 in Appendix B shows the full process of forward and inverse mapping.

3.6 Validation of the Representation

Figure 2 shows a visual validation for symmetry class II, symmetric about z. A cuboid-like object, with a detail to break the symmetry for axes x and y, is rotated across the space T represented by the grid, with some angular sparsity to avoid cluttering the plot. The colors of this grid follow a heatmap, representing the values of $s_{\mathrm{II},\gamma}$. Visually identical objects have dots of equal color (uniqueness), whereas visually similar objects, such as those near the symmetry-boundary at 180° (horizontal orange line), have similarly colored dots (continuity). This is shown in more detail in the right-hand plot. The visualization tool used to create these plots, which we also make publicly available, can be used to analyse new symmetry classes and extended rotation spaces.



(a) Plot for $s_{\mathrm{II},\gamma}$ (b) $s_{\mathrm{II},\gamma}$ near 180°

Fig. 2 **Visual verification of** $\mathrm{SARR}_{\mathrm{II}}$. The entire plot for $s_{\mathrm{II},\gamma}$ is shown in (a) and zoomed in near the symmetry boundary (orange line) in (b). Color of the grid represents the values of $s_{\mathrm{II},\gamma}$

3.7 Limitations

Uniqueness and continuity of SARR for these symmetry classes is guaranteed within T but not necessarily for the full space of rotations SO(3). This is because rotations about two non-symmetric axes can result in an object that is visually identical to one rotated about its symmetry axis, for example $\mathcal{V}_{\mathrm{V}}(180, 0, 180) = \mathcal{V}_{\mathrm{V}}(0, 180, 0)$ but $\mathrm{SARR}_{\mathrm{V}}(180, 0, 180) \neq \mathrm{SARR}_{\mathrm{V}}(0, 0, 0)$. Early results show that an additional space restriction, or mapping to an auxiliary, intermediate representation, can alleviate this problem. Regarding other symmetries, SARR is also valid for objects with multiple axes of symmetry, such as the ITODD objects or 3D primitives, although additional terms in the representation are necessary and inverse mapping can require extra steps.

4 Experiments

We use the SARR representation to train neural networks to estimate the orientation of objects in images from the T-LESS and ITODD datasets, using depth as the principal input modality. To this end, we first present results of mapping T-LESS rotation annotations $\boldsymbol{R}$ to $\boldsymbol{R}|_C$, as a form of sanity check to verify the correctness of SARR (4.1). We then describe evaluation metrics (4.2) and outline the experimental setup (4.3). For the estimation task we compare against both other methods from the leaderboard as well as our network but

trained with different rotation representations. We present results on the T-LESS 4.4 and ITODD 4.5 datasets, including an ablation study of SARR-networks trained and tested using other input modalities besides depth. The section concludes with a summary and discussion of our results 4.6.

4.1 T-LESS Annotation Mapping

To illustrate the advantage of using our rotation representation over default T-LESS annotations, Fig. 3 presents mapping results for the symmetry classes with ambiguous object poses in T: II, III, IV. For every class there are two rows of object images in poses that are visually (nearly) identical, yet numerically, at least according to the T-LESS annotations, very different. For rows 5 and 6 this discrepancy is especially noticeable: class IV requires a discretization using T-LESS annotations, per default resulting in 315 distinct yet correct rotation matrices. The columns show (a) the real, pre-processed depth maps $\boldsymbol{d}$ we use for training our networks (see Appendix C.1), (b) the RGB training images used in our ablation study with two of the multiple ambiguous T-LESS rotation matrices $\boldsymbol{R}$ overlayed, (c) our unique annotations $\boldsymbol{R}|_C$, derived from the canonic Euler angles $\alpha_i|_C, \beta_i|_C, \gamma_i|_C$ (d) network predictions $\hat{\boldsymbol{R}}$ using our annotations and (e) depth difference δ as false color maps. Depth difference was computed between rendered depth images[7] of objects in poses according to our ground truth $\boldsymbol{R}|_C$ ($\boldsymbol{d}_{Ours}$) vs. T-LESS ground truth $\boldsymbol{R}$ ($\boldsymbol{d}_{T-LESS}$):

$$\delta = \sum_{O} \sum_{\boldsymbol{d} \in \mathcal{D}} \sum_{\text{px} \in \text{mask}} |\boldsymbol{d}_{\text{Ours}} - \boldsymbol{d}_{\text{T-LESS}}|, \tag{10}$$

for all 30 objects O, all depth images $\mathcal{D}$ of those objects and all pixels px inside the respective visibility masks. We report an average depth discrepancy of $\delta = 0.5$mm across the dataset, proving that our process of mapping followed by inverse mapping results in poses that are visually identical, up to deviations due to symmetry imperfections which fall below the threshold used in the HALCON scripts (see Sect. 3.2). Per-object depth difference is highest for object 25, which is a light switch defined to be symmetric, yet the tilted switch-panel noticeably breaks this symmetry in many views. We also report a much higher average difference between real camera depth and rendered T-LESS depth of $\delta_{\text{Cam, T-LESS}} = 7.4$mm as well as $\delta_{\text{Cam, Ours}} = 7.8$mm for camera depth and our rendered depth due to the camera depth showing various artifacts and corruptions.

[7] These depth images were rendered only for the presented calculation.

4.2 Evaluation Metrics

For the orientation estimation experiments we calculate five different error functions: e, VSD, MSSD, MSPD and A(M)GPD from which we derive three evaluation metrics AR_C, AR_B and AR_G.

In line with the BOP benchmark, we report AR_B as the average of recalls under VSD, MSSD, and MSPD scores (Hodaň et al., 2020) to evaluate object orientation estimates $\hat{\boldsymbol{R}}$. We use the official BOP toolkit[8] to calculate these results, leaving all parameters unchanged.

We also calculate the rotational error $e \in [0, 180]°$ derived from the cosine distance (Hodaň et al., 2016):

$$e(\hat{\boldsymbol{R}}, \boldsymbol{R}|_C) = \arccos\left(\frac{\text{Tr}(\hat{\boldsymbol{R}}\boldsymbol{R}|_C{}^{-1}) - 1}{2}\right)\frac{180}{\pi}. \tag{11}$$

In the rare event that a method did not provide a prediction we assign the maximum error of 180°[9]. We then calculate the average recall AR_C taking the average of recall scores across the e thresholds $\{2, 5, 10, 15, 25, 40\}°$ (Kriegler et al., 2023, 2022).

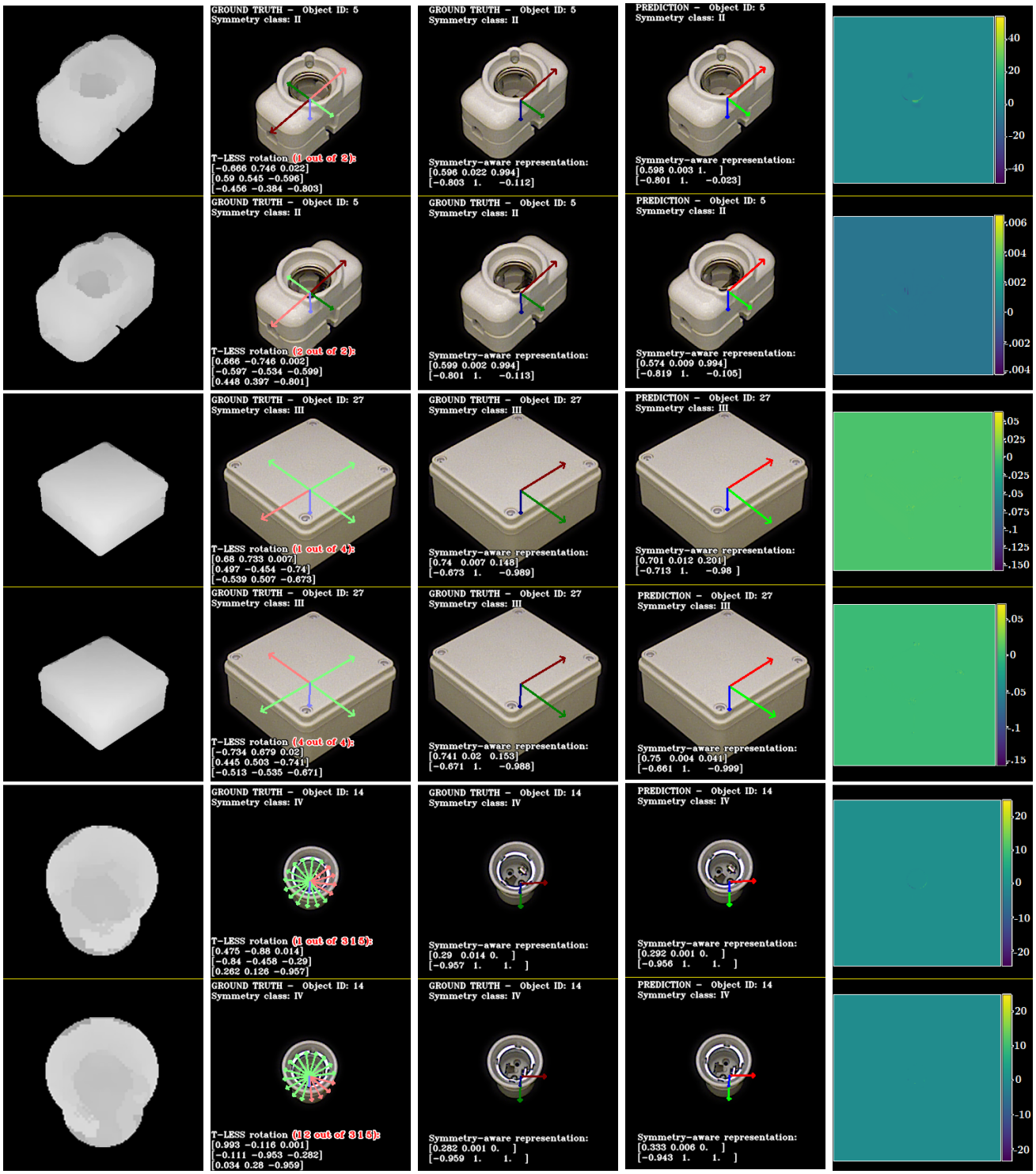
We also provide results using the A(M)GPD metric proposed by Mo et al. (2022), which we denote AR_G for brevity. AR_G is a pose distance metric that uses the ground-truth and estimated poses, the object class and predefined sets of grouped primitives. These grouped primitives incorporate information on symmetry permutations and were derived from the 3D model by Mo et al.. We reimplemented the calculation of the AR_G metric using Equations 12 and 13 from (Mo et al., 2022) as well as their source-code[10]. We can only report AR_G results on T-LESS, because the sets of grouped primitives were not defined for the ITODD objects.

Metrics AR_B and AR_G are both symmetry-invariant by design due to the properties of their underlying error functions, meaning that any one of the n potential poses for the same visual appearance is accepted. In contrast, for AR_C there is always only one correct orientation: $\boldsymbol{R}|_C$. In the context of this work, we consider AR_C to be a stricter metric, punishing methods that are ambiguous in their estimations.

[8] See https://github.com/thodan/bop_toolkit, specifically the script `eval_bop19_pose.py` (Accessed: 2026-01-12).

[9] When evaluating our networks we encountered two T-LESS depth maps that had NaN values. For other methods listed in Table 3 it is unknown why some estimates are missing.

[10] Specifically, we created a stand-alone, GPU-independent reimplementation of the `eval_metric` function from file `tless_gadd_evaluator.py`: https://github.com/GANWANSHUI/ES6D/blob/master/lib/tless_gadd_evaluator.py. This reimplementation is available in our repository: https://github.com/akriegler/SARR/blob/main/source/metrics/amgpd.py. (Accessed: 2026-01-12).



(a) Depth map $\boldsymbol{d}$ (b) T-LESS GT $\boldsymbol{R}$ (c) Our GT $\boldsymbol{R}|_C$ (d) Predictions $\hat{\boldsymbol{R}}$ (e) Depth-diff. δ

Fig. 3 SARR annotation mapping of T-LESS symmetry classes II, III and IV

AR_B and AR_G on the other hand are tolerant towards such ambiguous orientation estimations.

Lastly, we report results for two tasks: estimating the orientation for only the most visible instance of every object class in every image (SiSo), as well as for the n-most visible instances (ViVo), where n is provided with the annotation and is different for every object and image. When calculating results for the ViVo task, it is necessary to match the sets of pose ground-truths and pose estimates, for all instances of each object class apparent in any image. We treat this as a linear assignment problem, where the sets resemble the two partites, and solve it via minimum weight matching, computing a cost matrix using the error function e from Eq. (11). We do this for metrics AR_C and AR_G, while the BOP toolkit handles the matching for AR_B. We also report the average inference time of the orientation estimation for the ViVo task, where measurement starts right after an input sample i is loaded and ends when the orientation prediction $\hat{\boldsymbol{R}}_i$ is available (as the final 3×3 rotation matrix).

4.3 Experimental Setup

Columns "Training" and "Test" in Table 3 provide information regarding the training/validation data used for T-LESS, Table 5 shows the same information for ITODD. Specifically, for training our networks on T-LESS we use the real "PrimeSense" depth images as input and no 3D-models, while for our ablation study we train the SARR-networks using RGB images from the same "PrimeSense" sensor instead. For training on ITODD we use the synthetic images from a physically-based renderer (PBR) instead: depth images per default and RGB images converted to grayscale for the ablation study (Appendix C.1 explains the data preprocessing steps).

We evaluate on the real test depth/RGB images of T-LESS and the real validation depth/grayscale images of ITODD. The ITODD validation/test sets include no RGB but only grayscale images, and we use the validation set because we require ground truth 6D pose labels to calculate AR_C, AR_B and AR_G, which are not publicly available for the test but only the validation set (see Table 1). We use ground-truth translation for all results that we report: the networks we trained ourselves as well as results from other methods. For other methods we downloaded the original result files from the leaderboard and replaced the estimated translation with the ground-truth.

Considering other works from the benchmark leaderboard, it is common practice to train one network for every single object class, sometimes also across the entire dataset. We extend this approach and train multiple additional networks, one for every symmetry class, and thus identify these different scopes for our experiments:

$object$: one CNN for each object class, hence 30 for T-LESS and 28 for ITODD,
$symmetry$: one CNN for each symmetry class, hence 5 for T-LESS and 9 for ITODD,
$dataset$: one CNN for T-LESS, one for ITODD,
$dataset*$: one CNN for T-LESS, one for ITODD – both with an additional symmetry classification task.

Dataset-networks in the literature are sometimes designed in such a way that they do not require external object class information during inference. But since we need to know the symmetry-class for inverse mapping SARR, which we normally derive from the object-class, we define the additional scope *dataset**. Here the SARR-*dataset** networks have to predict the symmetry class of a given object from dataset i from the symmetry classes $u(\mathcal{S}_{i,t})$.

We use PyTorch (Paszke et al., 2019) to train a modified CenterNet (Zhou et al., 2019) network with HardNet (Chao et al., 2019) as backbone and optimize using Adam (Kingma & Ba, 2017). We use cosine distance and L1 loss for optimizing the rotation parameters. We do not use the geodesic loss (Mahendran et al., 2017) for rotation matrices as this would require modifying the network, e.g. by adding another layer (Salehi et al., 2019), which would result in an unfair comparison. We use FocalLoss (Lin et al., 2017) for the *dataset** networks since we embed the symmetry classification task in heatmaps. For more information regarding implementation and loss functions see Appendix C.2 and Appendix C.3, respectively.

4.4 T-LESS

We compare to some of the best performing methods from the BOP T-LESS leaderboard in Table 3. This includes not only RGB-D based methods, which have historically performed the best, but also RGB-only methods, which have become increasingly better in recent times, as well as depth-only methods. Under metric AR_B, SARR–Depth networks achieve satisfactory results, especially considering that we use only a small number of real depth (D: R) images but no 3D models (see columns "Training" and "Model") and we did not place significant emphasis on using the latest network architecture. Our networks already outperform some existing methods under AR_G and finally outperform all other methods under the symmetry-sensitive metric AR_C. The SARR–Depth-*dataset** network, trained with the additional task of symmetry classification, performed only marginally worse in terms of orientation estimation than the SARR–Depth-*dataset* network, while achieving a mean classification accuracy of 78.6% for the SiSo task, and 77.8% for ViVo (see Appendix D, specifically Fig. 5a to Fig. 5d). As can be seen in column 'Symm.', besides Drost et al. (2010) and Vidal et al. (2018), our method is the only one to augment the

Table 3 T-LESS orientation estimation. SARR–Depth-networks outperform all other methods under AR_C

Method	Training	Test	Scope	Model	Symm.	SiSo			ViVo			t[s]
						$AR_C\uparrow$	$AR_B\uparrow$	$AR_G\uparrow$	AR_C	AR_B	AR_G	
Hodaň et al. (2015)	templates	RGBD	unknown	CAD	evaluation	17.1	70.1	66.2	14.8	61.0	65.4	80.1
Sundermeyer et al. (2020)	RGB: R,S	RGBD	dataset	CAD	network	14.0	62.3	62.4	12.7	59.7	64.7	.531
Wang et al. (2021)	RGBD: R,S	RGBD	object	CAD	loss	33.5	**90.4**	*90.7*	30.9	91.2	*87.3*	6.63
Su et al. (2022)	RGBD: R,S	RGBD	object	reconstr.	data	38.5	85.6	**91.2**	36.7	90.6	87.2	2.62
ModalOcc.rgbd (2024)	RGBD: S	RGBD	unknown	CAD	unknown	30.8	79.8	90.4	29.3	94.1	87.6	7.24
Liu et al. (2025)	RGBD: R,S	RGBD	object	reconstr.	loss	32.0	89.4	91.1	29.6	**95.1**	86.8	2.48
Cai et al. (2022)	RGB: S	RGB	dataset	✗	network	30.2	75.1	85.9	27.4	87.1	86.0	50.8
Castro and Kim (2023)	RGB: S	RGB	dataset	CAD	loss	24.8	73.9	83.7	23.2	80.5	80.9	.059
ModalOcc.rgb (2024)	RGBD: S	RGB	unknown	CAD	unknown	32.6	69.1	90.0	30.3	93.1	**88.3**	7.75
Liu et al. (2025)	RGB: R,S	RGB	object	reconstr.	loss	33.7	*89.1*	90.6	30.2	*93.4*	85.6	.214
Drost et al. (2010)	templates	D	unknown	reconstr.	repres.	17.9	70.3	64.2	15.5	60.9	63.8	9.20
Vidal et al. (2018)	templates	D	unknown	reconstr.	repres.	22.6	76.0	74.0	19.5	68.9	71.5	7.06
ZTE_PPF (2022)	templates	D	unknown	reconstr.	unknown	21.9	79.3	73.8	19.3	75.1	74.5	.846
ModalOcc.depth (2024)	RGBD: S	D	unknown	CAD	unknown	27.3	81.8	83.4	23.6	80.6	82.0	4.72
SARR–Depth	D: R	D	*object*	✗	repres.	45.0	48.0	68.4	41.9	58.1	67.0	.077
	D: R	D	*symmetry*	✗	repres.	47.5	56.3	70.0	44.0	60.1	69.4	.078
	D: R	D	*dataset*	✗	repres.	**48.0**	50.3	69.8	**44.8**	60.0	69.4	.074
	D: R	D	*dataset**	✗	repres.	46.7	54.9	68.8	42.9	59.0	66.7	.077
SARR–RGB	RGB: R	RGB	*object*	✗	repres.	32.6	46.8	58.2	30.8	45.8	58.8	.077
	RGB: R	RGB	*symmetry*	✗	repres.	29.1	42.4	49.9	28.9	42.1	53.3	.077
	RGB: R	RGB	*dataset*	✗	repres.	*47.3*	49.0	69.3	*43.6*	58.7	69.1	.078
	RGB: R	RGB	*dataset**	✗	repres.	45.8	52.1	69.1	41.1	58.7	67.4	.080

1st, 2nd and 3rd best methods are **bold**, underlined and *italic* respectively. Column "t[s]" shows the inference time in seconds. D denotes depth, R real and S synthetic images. Methods ModalOcc.rgbd/rgb/depth and ZTE_PPF are unpublished

Table 4 T-LESS representation comparison. SARR is best suited for pose estimation of symmetric objects, outperforming the other representations across the different scopes, tasks and evaluation metrics

Representation	SiSo - AR_C			SiSo - AR_B			SiSo - AR_G		
	object	*symmetry*	*dataset(*)*	*object*	*symmetry*	*dataset(*)*	*object*	*symmetry*	*dataset(*)*
Euler	16.1	8.50	7.80	39.5	39.3	44.1	54.1	47.1	49.7
Rotation-Matrix	6.80	4.40	3.30	14.7	28.5	27.6	46.1	46.8	41.7
6d	5.70	7.40	1.10	16.9	27.8	19.6	35.1	38.1	34.5
Quaternion	7.80	9.10	1.40	41.9	51.0	21.0	52.3	52.6	30.6
Trigonometric	9.90	6.90	10.5	42.4	43.2	43.1	56.7	49.0	58.1
$\text{Euler}\|_C$	32.2	30.3	22.5	39.9	48.3	46.9	56.1	53.9	53.5
$\text{Rotation-Matrix}\|_C$	*38.8*	*41.3*	24.2	42.5	*54.0*	46.1	64.1	*63.7*	51.3
$\text{6d}\|_C$	34.8	40.7	*41.5*	41.6	48.0	*51.1*	58.4	63.3	*65.0*
$\text{Quaternion}\|_C$	33.7	38.0	28.5	47.4	53.3	**57.6**	60.2	61.6	56.8
$\text{Trigonometric}\|_C$	40.0	43.7	41.1	*46.7*	54.5	49.4	*63.9*	66.3	64.1
SARR–Depth	**45.0**	**47.5**	**48.0**(46.7)	**48.0**	**56.3**	50.3(54.9)	**68.4**	**70.0**	**69.8**(68.8)
Representation	ViVo - AR_C			ViVo - AR_B			ViVo - AR_G		
	object	*symmetry*	*dataset(*)*	*object*	*symmetry*	*dataset(*)*	*object*	*symmetry*	*dataset(*)*
Euler	15.3	8.00	7.70	44.4	37.7	41.3	54.9	49.1	52.0
Rotation-Matrix	6.00	3.60	3.30	29.1	24.2	24.5	44.9	48.8	43.7
6d	5.30	7.10	1.10	26.1	27.5	23.0	38.3	40.3	37.6
Quaternion	6.90	8.30	1.40	38.9	41.1	23.2	52.3	53.2	35.3
Trigonometric	8.50	6.10	9.10	44.6	39.5	47.6	56.5	49.8	57.4
$\text{Euler}\|_C$	30.5	29.7	20.9	46.2	45.3	44.0	56.9	56.4	54.5
$\text{Rotation-Matrix}\|_C$	*36.3*	*38.7*	22.2	*52.1*	*54.0*	40.6	63.7	*64.1*	52.9
$\text{6d}\|_C$	32.3	38.3	*38.6*	47.4	53.7	*55.5*	56.7	63.9	*64.2*
$\text{Quaternion}\|_C$	31.1	36.2	25.9	47.6	51.5	46.1	59.6	62.6	57.2
$\text{Trigonometric}\|_C$	37.2	41.0	38.3	52.9	56.4	54.9	*62.5*	66.6	63.4
SARR–Depth	**41.9**	**44.0**	**44.8**(42.9)	**58.1**	**60.1**	**60.0**(59.0)	**67.0**	**69.4**	**69.4**(66.7)

1st, 2nd and 3rd best methods are given in bold, underlined and italic respectively

numeric rotation representation itself to deal with symmetry ambiguities[11]. Other methods rely on the evaluation metrics, network/loss modifications or additional data. The last four rows show the 3D object orientation estimation results obtained using RGB images instead. Performance decreases notably compared to SARR–Depth for the *object* and *symmetry* scope networks, but for the *dataset(*)* networks, trained on a much larger single training set, performance stays comparable. See Fig. 6a through Fig. 6d for symmetry classification confusion matrices of the SARR–RGB-*dataset** network.

To show the merit of using SARR for rotation estimation of symmetric objects in comparison to other representations, we trained 30 additional networks using depth as input modality: One for each of the five other rotation representations (Euler angles, rotation matrices, the 6d representation from (Zhou et al., 2019), quaternions[12] and trigonometrics) and for each of the three scopes, with annotations derived from either the ambiguous T-LESS rotation matrices or our canonic Euler angles $\cdot|_C$. Table 4 shows these results for both tasks, SiSo on the top and ViVo on the bottom. With the exception of the AR_B score of $\text{Quaternion}|_C$-*dataset* on SiSo, networks trained on the SARR representation outperform all other representations across the different scopes, evaluation metrics and tasks. The table highlights the merit of paying respect to object symmetries, as all networks trained on unrestricted representations (upper halves in the SiSO and ViVO blocks) performed poorly, especially under the metric AR_C but also using the symmetry-invariant metrics AR_B and AR_G.

[11] Vidal et al. actually used the representation proposed by Drost et al. to handle the symmetries.

[12] For quaternions we map the redundant double cover to the "canonic" single cover, see https://docs.scipy.org/doc/scipy/reference/generated/scipy.spatial.transform.Rotation.as_quat.html (Accessed: 2026-01-12).

Table 5 ITODD orientation estimation. SARR–Depth-networks outperform SC6D under the strict AR_C metric, the *dataset**-network gives the best results. Changing the input to grayscale substantially decreases performance

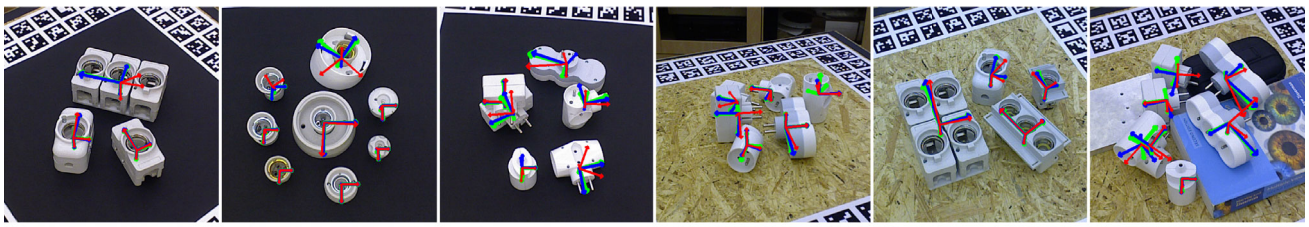
Method	Training	Test	Scope	Model	Symmetry	SiSo		ViVo		t[s]
						AR_C	AR_B	AR_C	AR_B	
Cai et al. (2022)	RGB: S	RGB	dataset	✗	network	36.7	**74.0**	30.6	**82.0**	.053
SARR–Depth	D: S	D	*object*	✗	representation	38.3	62.1	42.1	*67.7*	.071
	D: S	D	*symmetry*	✗	representation	41.0	68.7	43.0	63.7	.079
	D: S	D	*dataset*	✗	representation	41.0	*63.7*	*42.1*	65.1	.066
	D: S	D	*dataset**	✗	representation	**44.8**	56.3	**48.1**	71.1	.078
SARR–Gray	RGB2Gray: S	Gray	*object*	✗	representation	10.8	23.6	14.2	33.7	.080
	RGB2Gray: S	Gray	*symmetry*	✗	representation	7.1	24.0	8.1	28.7	.088
	RGB2Gray: S	Gray	*dataset*	✗	representation	2.8	6.9	4.6	15.4	.073
	RGB2Gray: S	Gray	*dataset**	✗	representation	6.5	22.5	6.6	24.2	.082

1st, 2nd and 3rd best methods are given in bold, underlined and italic respectively

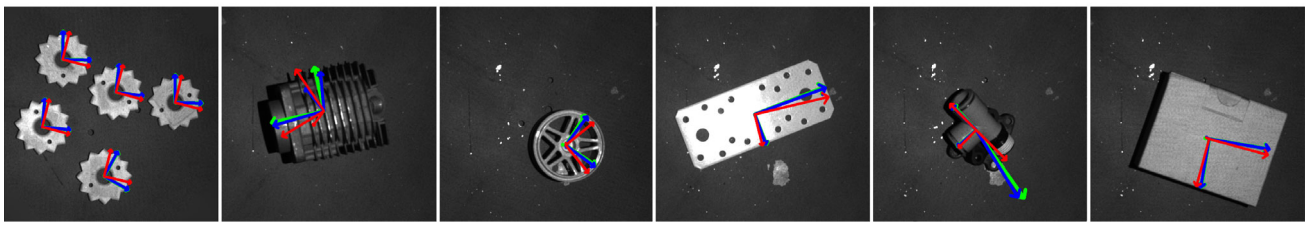
Table 6 The SARR representation outperforms most other representations

Representation	SiSo - AR_C			SiSo - AR_B			ViVo - AR_C			ViVo - AR_B		
	obj.	*symm.*	*dataset(*)*	*obj.*	*symm.*	*dataset(*)*	*obj.*	*symm.*	*dataset(*)*	*obj.*	*symm.*	*dataset(*)*
Euler	17.9	8.6	9.6	51.0	40.2	32.5	20.7	10.4	10.8	52.4	37.9	32.8
Rot.-Matrix	11.7	4.3	0.0	30.2	15.5	15.6	12.1	4.6	0.1	35.4	22.6	12.5
6d	8.6	0.3	1.2	24.4	14.8	14.9	9.6	1.1	2.2	38.4	15.3	18.9
Quaternion	13.3	1.2	0.9	27.0	29.6	17.0	10.6	1.9	0.9	45.1	34.7	23.9
Trigonometric	19.4	20.5	6.8	49.5	51.7	38.3	19.2	8.8	5.7	*60.0*	53.1	42.7
Euler$\|_C$	27.8	21.6	9.0	**68.5**	51.0	39.2	29.8	22.9	10.0	58.3	43.8	34.6
Rot.-Matrix$\|_C$	35.5	*31.2*	*34.6*	58.8	*52.2*	*59.6*	*34.4*	*34.6*	*35.5*	55.4	*52.9*	*61.3*
6d$\|_C$	38.9	23.1	18.5	56.0	42.6	46.9	33.9	22.8	19.2	54.5	41.0	41.3
Quaternion$\|_C$	28.7	24.7	11.7	44.4	47.1	32.9	26.8	26.4	11.8	48.9	44.5	31.1
Trigonometric$\|_C$	**40.1**	39.2	34.0	*61.2*	60.8	61.4	41.9	39.3	34.6	66.0	62.7	60.7
SARR–Depth	*38.3*	**41.0**	41.0(**44.8**)	62.1	**68.7**	**63.7**(56.3)	**42.1**	**43.0**	42.1(**48.1**)	**67.7**	**63.7**	65.1(**71.1**)

1st, 2nd and 3rd best methods are given in bold, underlined and italic respectively



(a) **T-LESS visualizations.** SARR–Depth-*dataset* predictions in blue, Trigonometric$|_C$-*dataset* in red.



(b) **ITODD visualizations.** SARR–Depth-*symmetry* predictions in blue, Trigonometric$|_C$-*object* in red.

Fig. 4 T-LESS and ITODD pose predictions. Ground-truths are given in green (Color figure online).

4.5 ITODD

While the ITODD BOP leaderboard is well populated with different methods, public results are only available for the test set, not the validation set. This means that we have to obtain pose estimations, and calculate results, for other methods ourselves, which in turn requires a maintained, publicly available framework. We additionally had to exclude all methods from consideration that used the real validation images for fine-tuning (which is a common practice), since the validation set constitutes our test set. With this in mind, we were able to set-up and deploy the SC6D method by Cai et al. (2022) for comparison. Their method also puts emphasis on object symmetries and does not use the 3D CAD models, like ours. Table 5 displays results for this comparison. While SC6D outperforms our networks under AR_B, using our SARR representation and depth images leads to better results following the stricter AR_C metric. Interestingly, the SARR–Depth-*dataset** network, which achieved symmetry classification accuracies of 91.0% for SiSo and 94.2% for ViVo (see Appendix D, specifically Fig. 5e to Fig. 5h), also performed the best in terms of orientation estimation. It seems that the additional task of learning to classify the symmetry positively impacted the rotation estimation optimization. We hypothesize that the symmetry class labels may act as guidance for the optimizer to more easily find and distinguish the respective areas of the loss surface, since these areas are topologically quite different from one another when using the SARR representation. Additional experiments with the SARR–Gray networks yielded poor results, most likely because the synthetic grayscale training images converted from RGB exhibit substantial difference in appearance compared to the real grayscale evaluation images.

We compare against other rotation representations in Table 6. Using our representation again yields networks that provide the best results in almost all comparisons. When training the Quaternion$|_C$-*dataset* network, we observed a NaN error that occurred during backpropagation in epoch 15 out of 40. This effectively halted the optimization at that point and might be an indicator of poor stability of the Quaternion representation for this learning task.

4.6 Results Summary

By averaging the representation comparison results from Table 4 and Table 6 across the three scopes and two tasks, we summarize our results in Table 7, which leads to four key observations. Firstly, using our proposed SARR representation leads to an absolute performance increase over the second best representation Trigonometric$|_C$ of about $3-5\%$, or a relative increase of up to 11%. Secondly, the representation generalizes well across two diverse datasets, with T-LESS and ITODD sharing only a couple of symmetry classes. Thirdly, comparing the last two rows shows that the additional task of classifying the object symmetry (*dataset**) is barely detrimental and sometimes even conducive to orientation estimation, which is relevant for scenarios where the symmetry class is not known a priori. Finally, relying solely on the symmetry-agnostic nature of an evaluation metric and using standard rotation representations leads to poor performance due to the ambiguities during optimization (upper half).

Table 7 Summary of results

Representation		T-LESS			ITODD	
		AR_C	AR_B	AR_G	AR_C	AR_B
Euler		10.6	41.1	51.2	13.0	41.1
Rotation-Matrix		4.57	24.8	45.3	5.47	22.0
6d		4.62	23.5	37.3	3.83	21.1
Quaternion		5.82	36.2	46.1	4.80	29.6
Trigonometric		8.50	43.4	54.6	13.4	49.2
Euler$\|_C$		27.7	45.1	55.2	20.2	49.2
Rotation-Matrix$\|_C$		33.6	48.2	60.0	34.3	56.7
6d$\|_C$		37.7	49.6	61.9	26.1	47.1
Quaternion$\|_C$		32.2	50.6	59.7	21.7	41.5
Trigonometric$\|_C$		*40.2*	*52.5*	*64.5*	*38.2*	*62.1*
SARR	/w *dataset*	**45.2**	55.5	**69.0**	41.3	**65.2**
	/w *dataset**	44.7	**56.1**	68.4	**42.9**	64.9

1st, 2nd and 3rd best methods are given in bold, underlined and italic respectively

Lastly, Fig. 4 shows visualizations of predicted orientations from networks trained on depth images and with different rotation representations, featuring various objects from both T-LESS and ITODD. While in some cases the other representations yield clearly wrong estimates, the advantages of the SARR networks are well visible, especially in the difficult T-LESS images (textured background in Fig. 4a) and the first ITODD image (Fig. 4b).

5 Conclusion

In this work, we focused on the challenges that arise when training a ML method to estimate the 3D orientation of symmetric objects. We focused on the symmetry classes in the popular T-LESS and ITODD datasets and proposed a representation which builds on trigonometric identities that resolves symmetry ambiguities to a canonic pose, while preserving continuity across symmetry boundaries. Our method operates on the annotations directly, allowing any pose estimation network to take into account the symmetries of objects, solving the problem of symmetries more naturally than existing works. The proposed representation scheme is generic and could be extended to additional symmetry

classes beyond the ones analysed in this paper. Therefore, future work includes the analysis of other object symmetries in different datasets, further extending the rotation space, and incorporating translation prediction to allow full 6D pose estimation. Disentangling the symmetry classification and pose regression tasks could also prove beneficial. Fusing depth and RGB information while remaining texture-agnostic in our symmetry definitions could open new avenues as well. Finally, we expect a performance increase from more powerful network architectures such as Transformers.

Appendix A SARR Generalization

This section generalizes our proposed symmetry-aware rotation representation SARR. The following definitions were verified for the symmetry classes of ITODD and the chosen 3D geometric primitives (taking into account the limitations discussed in Sect. 3.7). For any such 3D objects with symmetry class i and symmetry vector $\boldsymbol{\kappa}_i$, we restrict the Euler angles to the respective canonic space C_i:

$$\begin{aligned}\alpha\big|_{C_i} = \alpha \in \left[0, \ldots, \frac{2\pi}{\boldsymbol{\kappa}_{i,\alpha}}\right[,\\ \beta\big|_{C_i} = \beta \in \left[0, \ldots, \frac{2\pi}{\boldsymbol{\kappa}_{i,\beta}}\right[,\\ \gamma\big|_{C_i} = \gamma \in \left[0, \ldots, \frac{2\pi}{\boldsymbol{\kappa}_{i,\gamma}}\right[.\end{aligned} \tag{12}$$

Assuming intrinsic rotations in 'XYZ'-order, the representation SARR is defined as:

$$\begin{aligned}&\mathrm{SARR}_i(\alpha\big|_{C_i}, \beta\big|_{C_i}, \gamma\big|_{C_i}) = \begin{bmatrix} s_{i,\alpha} & s_{i,\beta} & s_{i,\gamma} \\ c_{i,\alpha} & c_{i,\beta} & c_{i,\gamma} \end{bmatrix} := \\ &\begin{bmatrix} \sin(\boldsymbol{\lambda}_\alpha \alpha\big|_{C_i}) & \sin(\boldsymbol{\lambda}_\beta \beta\big|_{C_i})\nu_\alpha & \sin(\boldsymbol{\lambda}_\gamma \gamma\big|_{C_i})\nu_\alpha\nu_\beta \\ \cos(\boldsymbol{\lambda}_\alpha \alpha\big|_{C_i}) & \cos(\boldsymbol{\lambda}_\beta \beta\big|_{C_i}) & \cos(\boldsymbol{\lambda}_\gamma \gamma\big|_{C_i}) \end{bmatrix}.\end{aligned} \tag{13}$$

The terms

$$\begin{aligned}\boldsymbol{\lambda}_\alpha &= \begin{cases} 0 & \text{if } \boldsymbol{\kappa}_{i,\alpha} \text{ is } \infty, \\ \boldsymbol{\kappa}_{i,\alpha} & \text{otherwise,} \end{cases}\\ \boldsymbol{\lambda}_\beta &= \begin{cases} 0 & \text{if } \boldsymbol{\kappa}_{i,\beta} \text{ is } \infty, \\ \boldsymbol{\kappa}_{i,\beta} & \text{otherwise,} \end{cases}\\ \boldsymbol{\lambda}_\gamma &= \begin{cases} 0 & \text{if } \boldsymbol{\kappa}_{i,\gamma} \text{ is } \infty, \\ \boldsymbol{\kappa}_{i,\gamma} & \text{otherwise,} \end{cases}\end{aligned} \tag{14}$$

resolve continuous symmetries as shown in Eq. (8). The difference to the representation for T-LESS are the additional terms

$$\begin{aligned}\nu_\alpha &= \begin{cases} \cos(\alpha\big|_{C_i}) & \text{if } 1 < \boldsymbol{\kappa}_{i,\alpha} < \infty, \\ 1.0 & \text{otherwise,} \end{cases}\\ \nu_\beta &= \begin{cases} \cos(\beta\big|_{C_i}) & \text{if } 1 < \boldsymbol{\kappa}_{i,\beta} < \infty, \\ 1.0 & \text{otherwise.} \end{cases}\end{aligned} \tag{15}$$

These additional terms are necessary to ensure that visually distinct poses have different numeric representations. For example, one can imagine the rectangular cuboid, $\boldsymbol{\kappa}_{\text{CUBOID}} = [2, 2, 2]$, with distinct images $\mathcal{V}_{\text{CUBOID}}(0, 0, 45) \neq \mathcal{V}_{\text{CUBOID}}(0, 180, 45)$, yet without ν_α, ν_β, the representation would be identical.

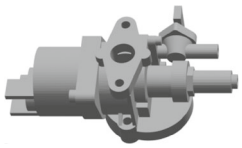
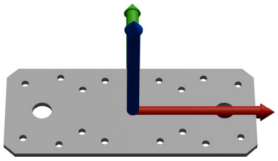
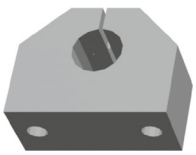
Inverse mapping the SARR representation is done sequentially, starting with the rotation first in chain, α. The ν terms are computed as in Eq. (15), but done so now on the fly, using the calculated angle from the previous step:

$$\begin{aligned}\alpha\big|_{C_i} &= \begin{cases} 0 & \text{if } \boldsymbol{\kappa}_{i,\alpha} \text{ is } \infty, \\ \frac{1}{\boldsymbol{\kappa}_{i,\alpha}}(2\pi - \arccos(c_{i,\alpha})) & \text{if } s_{i,\alpha} < 0, \\ \frac{1}{\boldsymbol{\kappa}_{i,\alpha}} \arccos(c_{i,\alpha}) & \text{otherwise,} \end{cases}\\ \beta\big|_{C_i} &= \begin{cases} 0 & \text{if } \boldsymbol{\kappa}_{i,\beta} \text{ is } \infty, \\ \frac{1}{\boldsymbol{\kappa}_{i,\beta}}(2\pi - \arccos(c_{i,\beta})) & \text{if } \frac{s_{i,\beta}}{\nu_\alpha} < 0, \\ \frac{1}{\boldsymbol{\kappa}_{i,\beta}} \arccos(c_{i,\beta}) & \text{otherwise,} \end{cases}\\ \gamma\big|_{C_i} &= \begin{cases} 0 & \text{if } \boldsymbol{\kappa}_{i,\gamma} \text{ is } \infty, \\ \frac{1}{\boldsymbol{\kappa}_{i,\gamma}}(2\pi - \frac{\arccos(c_{i,\gamma})}{\boldsymbol{\kappa}_{i,\gamma}}) & \text{if } \frac{s_{i,\gamma}}{\nu_\alpha\nu_\beta} < 0, \\ \frac{1}{\boldsymbol{\kappa}_{i,\gamma}} \arccos(c_{i,\gamma}) & \text{otherwise.} \end{cases}\end{aligned} \tag{16}$$

As can be seen from equations Eq. (13) and Eq. (16), the ν terms introduce a singularity at 90° for symmetry classes of type $\boldsymbol{\kappa} = [\{2, 3\}, \{2, 3\}, n], n \in \mathbb{Z}$. Fortunately, this is mostly a theoretic problem, because it is exceedingly rare that the symmetry class II objects from ITODD, or any object in the real world for that matter, have a rotation of *exactly* 90° – our networks trained on those objects have converged successfully. For symmetry class V of ITODD, and when considering the full SO(3) rotation space, inverse mapping is done not via Eq. (16) but instead via:

$$\begin{aligned}\alpha\big|_{C_i} &= \begin{cases} 2\pi - \arccos(c_{i,\alpha}) & \text{if } s_{i,\alpha} < 0, \\ \arccos(c_{i,\alpha}) & \text{otherwise,} \end{cases}\\ \beta\big|_{C_i} &= \begin{cases} \frac{2\pi}{\boldsymbol{\kappa}_{i,\beta}} - \frac{\arccos(c_{i,\beta})}{\boldsymbol{\kappa}_{i,\beta}} & \text{if } s_{i,\beta} < 0, \\ \frac{1}{\boldsymbol{\kappa}_{i,\beta}} \arccos(c_{i,\beta}) & \text{otherwise,} \end{cases}\\ \gamma'\big|_{C_i} &= \begin{cases} 2\pi - \arccos(c_{i,\gamma}) & \text{if } s_{i,\gamma} < 0, \\ \arccos(c_{i,\gamma}) & \text{otherwise,} \end{cases}\\ \gamma\big|_{C_i} &= \begin{cases} -\gamma'\big|_{C_i} & \text{if } s_{i,\beta} < 0, \\ \gamma'\big|_{C_i} & \text{otherwise.} \end{cases}\end{aligned} \tag{17}$$

Table 8 Symmetry classes of ITODD objects. Each of the symmetry classes i has a representation SARR_i, vector $\boldsymbol{\kappa}_i$, a symmetry type and a set of rotations $\mathcal{R}_i$. T-LESS class IDs of the visualized objects are indicated **bold** in the bottom rows. The default 3D coordinate frame, as shown for class II, is oriented the same for all objects

				
Class	I	II	III	IV
SARR$_i$ (ITODD)	SARR_I	SARR_II	SARR_III	SARR_IV
$\boldsymbol{\kappa}_i$	$\boldsymbol{\kappa}_\text{I} = [1, 1, 1]$	$\boldsymbol{\kappa}_\text{II} = [2, 2, 2]$	$\boldsymbol{\kappa}_\text{III} = [1, 1, \infty]$	$\boldsymbol{\kappa}_\text{IV} = [1, 1, 5]$
Type	None	Discrete	Continuous	Discrete
$\mathcal{R}_i$	{}	$\boldsymbol{R}^{\alpha,\beta,\gamma}_{x,y,z} \vert \alpha, \beta, \gamma \in \{\pi\}$	$\boldsymbol{R}^{\gamma}_{z} \vert \gamma \in \{\mathbb{R}\}$	$\boldsymbol{R}^{\gamma}_{z} \vert \gamma \in \{\frac{2n\pi}{5}\}$, $n \in \{1, \ldots, 4\}$
ITODD IDs	**20**, 1, 2, 4, 5, 6, 10, 13, 15, 16, 21, 22, 26	**11**, 3, 19	**23**, 7, 24, 27	**8**
				
V	VI	VII	VIII	IX
SARR_V	SARR_VI	SARR_VII	SARR_VIII	SARR_IX
$\boldsymbol{\kappa}_\text{V} = [1, 2, 1]$	$\boldsymbol{\kappa}_\text{VI} = [1, 1, 18]$	$\boldsymbol{\kappa}_\text{VII} = [1, 1, 23]$	$\boldsymbol{\kappa}_\text{VIII} = [1, 1, 12]$	$\boldsymbol{\kappa}_\text{IX} = [2, 2, \infty]$
Discrete	Discrete	Discrete	Discrete	Discrete & Continuous
$\boldsymbol{R}^{\beta}_{y} \vert \beta \in \{\pi\}$	$\boldsymbol{R}^{\gamma}_{z} \vert \gamma \in \{\frac{2n\pi}{18}\}$, $n \in \{1, \ldots, 17\}$	$\boldsymbol{R}^{\gamma}_{z} \vert \gamma \in \{\frac{2n\pi}{23}\}$, $n \in \{1, \ldots, 22\}$	$\boldsymbol{R}^{\gamma}_{z} \vert \gamma \in \{\frac{2n\pi}{12}\}$, $n \in \{1, \ldots, 11\}$	$\boldsymbol{R}^{\alpha,\beta,\gamma}_{x,y,z} \vert \alpha, \beta \in \{\pi\}$, $\gamma \in \{\mathbb{R}\}$
9, 18	**14**	**17**	**25**	**12**, 28

An overview of all the ITODD symmetry classes is shown in Table 8. We provide an alternative definition for the symmetry classes of some ITODD objects. Specifically, we define object 23, a screw, to be continuously symmetric, i.e. to belong to class III. We also consider objects 2, 4 and 5 non-symmetric. In each case, their object centroid in the 3D model was defined at a location s.t. no single rotation $\boldsymbol{r}$ about any axis could result in a visually identical pose. For object 2, because the centroid is shifted far off the symmetry-axis thus requiring a rototranslation, and for objects 4 and 5 because the transformation requires chaining multiple elementary rotations. We use this alternative symmetry classification for training all our networks and evaluating under the AR_C metric, yet we use the original definition from BOP when computing AR_B scores.

Regarding 3D geometric primitives, the above definition also works for many common primitives. The ones we consider are cuboids, cylinders, torii and spheres. Cuboids themselves can be split into five different symmetry classes:

$CUBOID$: rectangular prism, every face is a non-square rectangle,
CUB_XY : the two faces parallel to the XY plane are squares, the other four are non-square rectangles,
CUB_XZ : the two faces parallel to the XZ plane are squares, the other four are non-square rectangles,
CUB_YZ : the two faces parallel to the YZ plane are squares, the other four are non-square rectangles,
$CUBE$: a cube, all faces are squares.

Assuming upright default pose (see the default coordinate frame in Table 8), the symmetry vectors for these primitives are: $\boldsymbol{\kappa}_\text{CUBOID} = [2, 2, 2]$, $\boldsymbol{\kappa}_\text{CUB\_XY} = [2, 2, 4]$, $\boldsymbol{\kappa}_\text{CUB\_XZ} = [2, 4, 2]$, $\boldsymbol{\kappa}_\text{CUB\_YZ} = [4, 2, 2]$, $\boldsymbol{\kappa}_\text{CUBE} = [4, 4, 4]$, $\boldsymbol{\kappa}_\text{CYLINDER} \equiv \boldsymbol{\kappa}_\text{TORUS} = [2, 2, \infty]$, $\boldsymbol{\kappa}_\text{SPHERE} = [\infty, \infty, \infty]$.

Appendix B SARR Algorithm

Algorithm 1 shows the entire process of mapping a rotation representation, here a 3×3 rotation matrix, to our proposed SARR representation, followed by inverse mapping back to a rotation matrix $\mathbf{R}|_C$. While the SARR representation, which is used in the algorithm as an intermediate result, fulfills the uniqueness and continuity properties, $\mathbf{R}|_C$ on the other hand is also unique for the given symmetry classes w.r.t. the visual appearance, but not continuous. This generalized algorithm is valid for the symmetry classes and rotation spaces as shown in Table 2 and Table 8 with the exception of class V (both datasets). Mapping for symmetry class V, due to its specific property of having one axis of symmetry which is not the last in the defined rotation order, requires some modifications which are shown at the bottom of the algorithm.

Appendix C Experimental Setup

This section provides details regarding data pre-processing, implementation details of our networks and a discussion regarding loss functions.

C. 1 Image Preprocessing

T-LESS: We take a centered crop and set the pixels not part of the ground-truth object mask to 0. Masking allows the network to focus on the object by removing spurious background signals, which is especially relevant for the test images, as there is significant object clutter in the test scenes. For depth images, we calculate the mean pixel value using the visibility mask (a different mask, which only includes actually visible pixels), and use this mean to fill pixels that fall outside the defined minimum-maximum depth range: $d_{min} = 530, d_{max} = 929$ millimeters (Hodaň et al., 2016). We then cut out the object using the object mask, resize and pad the patch to 384×384 while preserving the aspect ratio (patches of size 384×384 are expected by CenterNet). We then clamp the depth map to $[d_{min}, d_{max}]$, since resizing can result in depth values outside this range, before inversing the depth, i.e. further objects now appear darker. We use these depth and RGB patches to calculate the mean and standard deviation for data standardization, for every object class, symmetry class and across the entire dataset, in line with the different scopes used during network training.

For test images we perform the same augmentations, but before masking we also divide every depth map $\boldsymbol{d}_{c,i}$ by $\bar{t}_{c,z}$, where $\boldsymbol{d}_{c,i}$ is the depth map of instance i of object class c, and $\bar{t}_{c,z}$ is the average translation in z-direction, or "center-depth", of all training instances of object class c. This effectively removes the impact of translations on the depth distribution, resulting in depth maps with variance that stems only from

Algorithm 1 Mapping a standard rot.-mat. $\mathbf{R}$ to a canonic rot.-mat. $\mathbf{R}|_C$ via the SARR representation.

1: **function** CANONICVIASARR($\mathbf{R}$, sym_cls) ▷ $\mathbf{R} \in$ SO(3), sym_cls $\in$ {I, ..., IX}
2: $\boldsymbol{\kappa} \equiv [\kappa_\alpha, \kappa_\beta, \kappa_\gamma] := \kappa_{\text{sym\_cls}}$ ▷ $\kappa_\alpha, \kappa_\beta, \kappa_\gamma \in \mathbb{Z}^+$
3: SARR $\leftarrow$ FORWARD($\mathbf{R}, \boldsymbol{\kappa}$) ▷ SARR $\in \mathbb{R}^{2\times 3}$, SARR$_{i,j} \in [-1, 1]$
4: $\mathbf{R}|_C \leftarrow$ INVERSE(SARR, $\boldsymbol{\kappa}$) ▷ $\mathbf{R}|_C \in C$ with $C \subset$ SO(3) if any(κ_i) > 1, else $C \equiv$ SO(3)
5: **end function**

6: **function** FORWARD($\mathbf{R}, \boldsymbol{\kappa}$)
7: $\alpha, \beta, \gamma \leftarrow$ R_to_Euler($\mathbf{R}$, 'XYZ', 'radians') ▷ Convert $\mathbf{R}$ to Euler angles, in intrinsic 'XYZ' order
8: $\alpha|_C \leftarrow (\alpha \bmod \frac{2\pi}{\kappa_\alpha}) \frac{\kappa_\alpha \bmod 10^3}{\kappa_\alpha}$ ▷ Clamp the three angles to unique subspaces
9: $\beta|_C \leftarrow (\beta \bmod \frac{2\pi}{\kappa_\beta}) \frac{\kappa_\beta \bmod 10^3}{\kappa_\beta}$ ▷ $\kappa_i = 10^3$ handles continuous symmetry
10: $\gamma|_C \leftarrow (\gamma \bmod \frac{2\pi}{\kappa_\gamma}) \frac{\kappa_\gamma \bmod 10^3}{\kappa_\gamma}$ ▷ A different clamping is required for class V, see lines 29-33
11: $v_\alpha \leftarrow \cos(\alpha|_C)$ if $2 <= \kappa_\alpha < 10^3$ else 1 ▷ Handles objects with multiple symmetry axes
12: $v_\beta \leftarrow \cos(\beta|_C)$ if $2 <= \kappa_\beta < 10^3$ else 1
13: $s_\alpha \leftarrow \sin(\kappa_\alpha \alpha|_C)$ ▷ Build multi-valued trigonometrics
14: $c_\alpha \leftarrow \cos(\kappa_\alpha \alpha|_C)$ ▷ Six parameters make up the SARR representation
15: $s_\beta \leftarrow \sin(\kappa_\beta \beta|_C) v_\alpha$
16: $c_\beta \leftarrow \cos(\kappa_\beta \beta|_C)$
17: $s_\gamma \leftarrow \sin(\kappa_\gamma \gamma|_C) v_\alpha v_\beta$
18: $c_\gamma \leftarrow \cos(\kappa_\gamma \gamma|_C)$
19: **return** SARR $\leftarrow$ concat_transpose($s_\alpha, c_\alpha, s_\beta, c_\beta, s_\gamma, c_\gamma$) ▷ Get SARR in desired shape, i.e. 2×3
20: **end function**

21: **function** INVERSE(SARR, $\boldsymbol{\kappa}$) ▷ Reconstruct the angles
22: $\alpha|_C \leftarrow 0$ if $\kappa_\alpha = 10^3$ else $\frac{2\pi - \arccos(c_\alpha)}{\kappa_\alpha}$ if $s_\alpha < 0$ else $\frac{\arccos(c_\alpha)}{\kappa_\alpha}$
23: $v_\alpha \leftarrow \cos(\alpha|_C)$ if $2 <= \kappa_\alpha < 10^3$ else 1 ▷ via unit-circle endpoints
24: $\beta|_C \leftarrow 0$ if $\kappa_\beta = 10^3$ else $\frac{2\pi - \arccos(c_\beta)}{\kappa_\beta}$ if $\frac{s_\alpha}{v_\alpha} < 0$ else $\frac{\arccos(c_\alpha)}{\kappa_\alpha}$
25: $v_\beta \leftarrow \cos(\beta|_C)$ if $2 <= \kappa_\beta < 10^3$ else 1 ▷ v_i are computed on-the-fly
26: $\gamma|_C \leftarrow 0$ if $\kappa_\gamma = 10^3$ else $\frac{1}{\kappa_\gamma}(2\pi - \frac{\arccos(c_\gamma)}{\kappa_\gamma})$ if $\frac{s_\alpha}{v_\alpha v_\beta} < 0$ else $\frac{\arccos(c_\gamma)}{\kappa_\gamma}$
27: **return** $\mathbf{R}|_C \leftarrow$ Euler_to_R($\alpha|_C, \beta|_C, \gamma|_C$, 'XYZ') ▷ Converts Euler angles to a rot.-mat.
28: **end function**

29: **if** $\alpha \bmod 2\pi > \pi$ **then** ▷ Replaces lines 8-10 for symmetry class V
30: $\alpha|_C \leftarrow (\alpha - \pi) \bmod 2\pi$
31: $\beta|_C \leftarrow -\beta$
32: $\gamma|_C \leftarrow (\pi - \gamma) \bmod 2\pi$
33: **end if**

34: $\alpha|_C \leftarrow 2\pi - \arccos(c_\alpha)$ if $s_\alpha < 0$ else $\arccos(c_\alpha)$ ▷ Replaces lines 22-26 for symmetry class V
35: $\beta|_C \leftarrow \frac{2\pi - \arccos(c_\beta)}{\kappa_\beta}$ if $s_\beta < 0$ else $\frac{\arccos(c_\beta)}{\kappa_\beta}$
36: $\gamma|_C \leftarrow (-1 \text{ if } s_\beta < 0)(2\pi - \arccos(c_\gamma)$ if $s_\gamma < 0$ else $\arccos(c_\gamma))$

the shape and orientation of the objects, which in turn aligns training and test samples more closely.

ITODD: We use the 50k synthetic PBR training images, from which we select only the samples with a visibility of at least 60% and require that the center-depth, given via t_z, satisfies $695mm < t_z < 761mm$. This depth range was derived from the statistics of the validation set. We then crop out and mask the object instance. We resize-pad to 384×384 while preserving the aspect ratio, clamp the depth map to [650, 770] millimeters, scaled to [0, 1]. For the ablation study we convert the RGB patches to grayscale using `cv2.cvtColor(img, cv2.COLOR_RGB2GRAY)`. We do not perform any standardization or depth map scaling via the average center-depth like we did for T-LESS.

The real depth images from the validation set used for the evaluation feature many invalid depth pixels due to reflections on the surfaces of the metallic objects. We therefore fill the depth patch after cropping and masking in a reverse-watershed like manner, using simple dilation, guided by the object mask. Depth and grayscale patches are then resize-padded, depth samples are additionally scaled like the training patches.

C. 2 Network Implementation

We use PyTorch (Paszke et al., 2019) to train a modified CenterNet (Zhou et al., 2019) network, using HardNet (Chao et al., 2019) as the backbone. We train all networks for 40 epochs, optimizing with Adam (Kingma & Ba, 2017). We use PyTorch's `ReduceLROnPlateau`[13] function as a learning rate scheduler. We use `'sum'` for loss reduction (taking the sum of the losses of all samples in the batch for backpropagation) and a batch size of 4 per GPU. Loss terms for the three elementary rotations are weighted equally with 1. We use the heatmap head to learn the symmetry class when training the SARR-*dataset** networks, using FocalLoss (Lin et al., 2017). Number of total epochs, initial learning rate, learning rate decay parameters and batch size were initially optimized by observing the loss-plots of networks trained using SARR on T-LESS. Table 9 shows all relevant hyperparameters of our experiments.

C. 3 Loss Functions

To ensure a fair comparison between the different rotation representations we want to train all networks as similar as possible, yet each representation has slightly different mathematical properties (dimensionality, range). A common ground was found using the cosine distance, interpreted as the cosine similarity loss function. For ground truths and predictions, the quaternion ($n = 4$), trigonometric ($n = 6$), rotation matrix ($n = 9$), 6d ($n = 6$) and SARR ($n = 6$) representations are flattened to a $1 \times n$ vector and the difference between true and predicted vectors is computed as the cosine distance. Since the cosine similarity loss does not enforce magnitude but only directionality between the two vectors, resulting predictions are normalized in a post-processing step for visualization and evaluation. This normalization happens column-wise for the trigonometric, rotation matrix and SARR representations, where each column is typically unit-length. Versors (quaternions that represent rotations) are also unit-length and thus the entire quaternion can be normalized. When mapping the 6d representation to a rotation matrix, normalization happens by definition. Euler angles do not allow normalization - we therefore use standard L1 loss for Euler angles. Regarding the geodesic loss (Mahendran et al., 2017) that is often used for rotation matrices: the trace-operations involved in the calculation of the loss during training require matrices to already have unit-length columns. This means the output of the network has to be constrained in some way, for example by adding another layer (Salehi et al., 2019). The comparison would then no longer be with networks of identical architecture, resulting in an unfair comparison.

Appendix D Symmetry Classification

The SARR-*dataset** networks were trained to classify the symmetry classes (5 for T-LESS and 9 for ITODD). This section provides detailed classification results.

T-LESS: As can be seen by the confusion matrices (CFMs) in Fig. 5 on the left, the network achieves a mean classification accuracy of 78.6% and 77.8% for the SiSo and ViVo tasks, respectively. Class I gets confused with class II most likely because many fine details that break the symmetries for objects of class I are lost in the depth map (e.g. one-sided holes getting filled). Class V gets confused with class II, most likely due to the shapes being virtually identical in their symmetry, the only difference being the definition of the unrotated pose, or "uprightness": if the objects of class V (shown in Table 2 of the main paper) were to be put upright, symmetry would be identical to class II. This is amplified by the fact that these class V objects in the test images are often placed upright, so if we were to change some symmetry definitions of the dataset, we would first suggest treating objects of class V like those of class II, as the two are equivalent from a symmetry standpoint. The absolute CFMs (subplots (b) and (d)) further show the strong class imbalance, with only 96 samples belonging to class III. The SARR–RGB-*dataset** network achieves mean classification accuracy of 71.2% and 68.0% for the SiSo and ViVo tasks, respectively, with additional objects from across the dataset erroneously

[13] https://docs.pytorch.org/docs/stable/generated/torch.optim.lr_scheduler.ReduceLROnPlateau.html (Accessed: 2026-01-12)

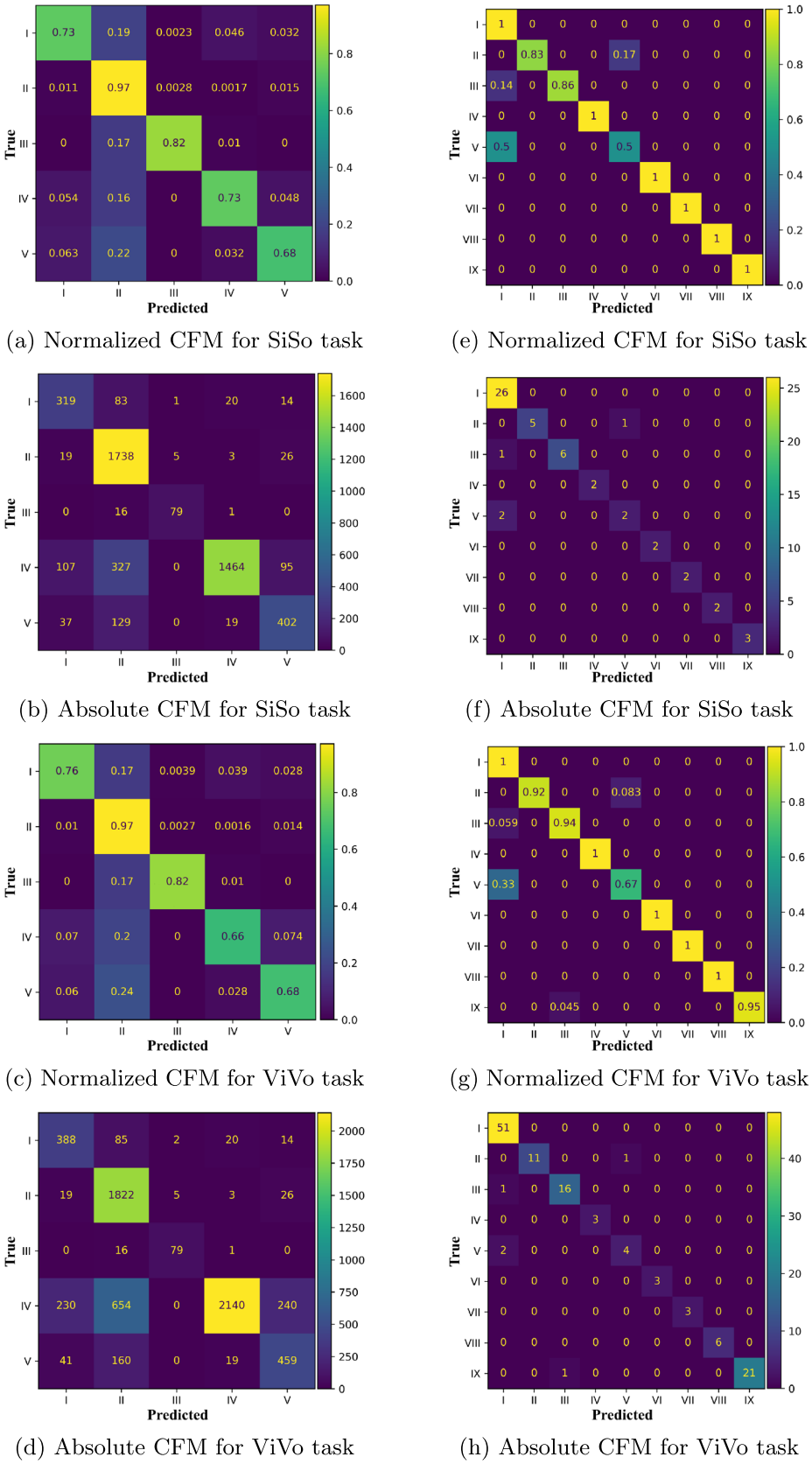


Fig. 5 SARR–Depth-*dataset** confusion matrices for T-LESS (left) and ITODD (right)

(a) Normalized CFM for SiSo task

(e) Normalized CFM for SiSo task

(b) Absolute CFM for SiSo task

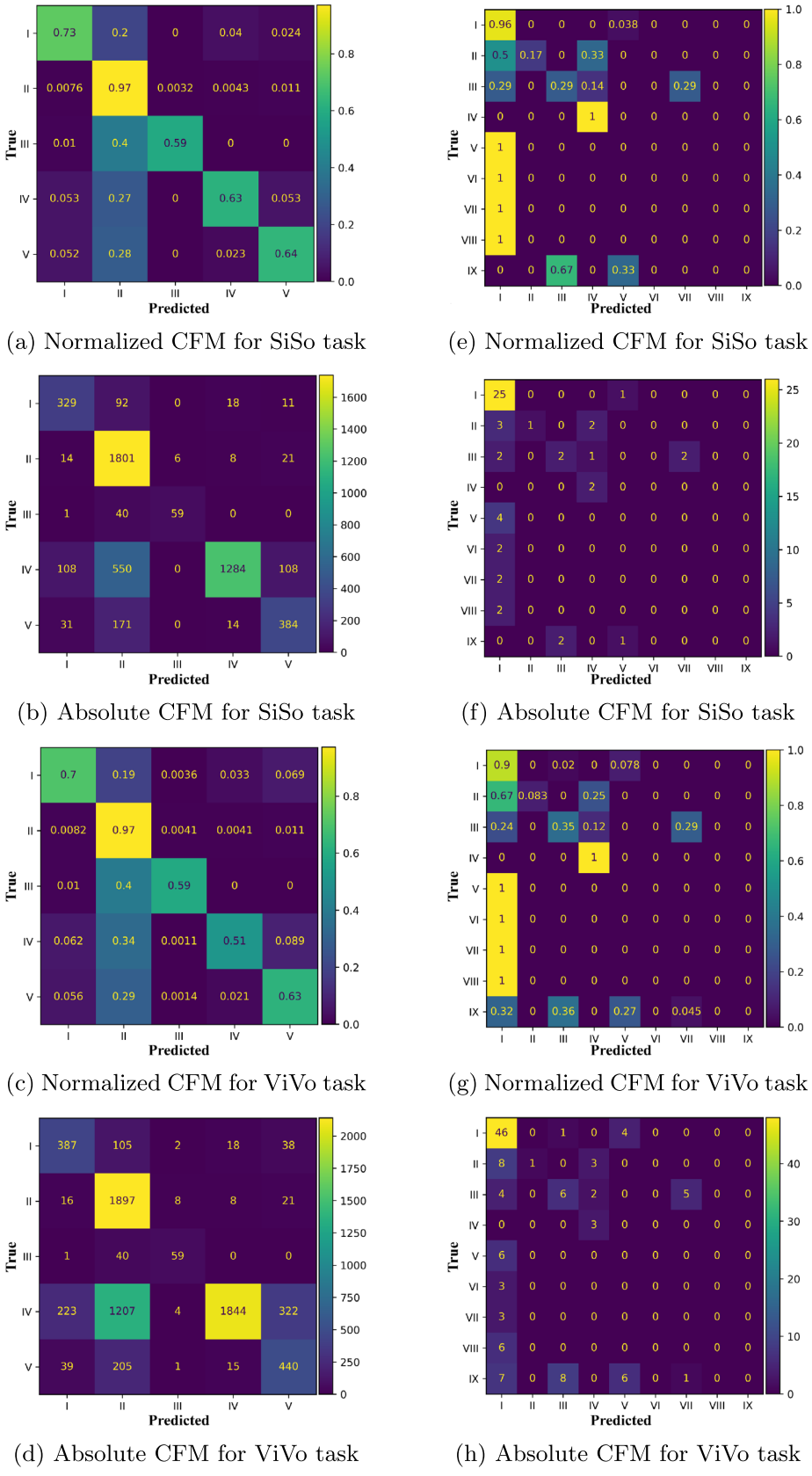
(f) Absolute CFM for SiSo task

(c) Normalized CFM for ViVo task

(g) Normalized CFM for ViVo task

(d) Absolute CFM for ViVo task

(h) Absolute CFM for ViVo task



(a) Normalized CFM for SiSo task

(e) Normalized CFM for SiSo task

(b) Absolute CFM for SiSo task

(f) Absolute CFM for SiSo task

(c) Normalized CFM for ViVo task

(g) Normalized CFM for ViVo task

(d) Absolute CFM for ViVo task

(h) Absolute CFM for ViVo task

Fig. 6 SARR–RGB-*dataset** confusion matrices for T-LESS on the left and SARR–Gray-*dataset** matrices for ITODD on the right

Table 9 Network hyperparameters

Parameter	T-LESS	ITODD
Backbone	HardNet	HardNet
Epochs	40	40
Batch-size	4	12
Input-size	384 × 384	384 × 384
Optimizer	Adam	Adam
Rotation-loss	Cosine/L1	Cosine/L1
Symm.-Class-loss	FocalLoss	FocalLoss
Loss-reduction	`'sum'`	`'sum'`
Loss-weights	1.0 for all	1.0 for all
Learning rate	$6 * 10^{-4}$	$6 * 10^{-4}$
LR-scheduler	`ReduceLROnPlateau`	`ReduceLROnPlateau`
LR-factor	0.5	0.5
LR-patience	2	2
LR-threshold	0.15	0.1
LR-cooldown	1	1
GPUs	1 × RTX 3090	3 × RTX 3090

classified as belonging to symmetry class II. This is visible in Fig. 6 on the left.

ITODD: As shown on the right side of Fig. 5, we obtained mean accuracies of 91.0% and 94.2% for SiSo and ViVo tasks, respectively. Class I receives the most false positives, most likely due to the strong dataset imbalance, see subplots (f) and (h). Symmetry classification accuracies of 26.9% and 25.9% from the SARR–Gray-*dataset** network is substantially worse compared to the depth-based counterpart, with numerous classes receiving no true-positive predictions, as is shown in Fig. 6 on the right.

Supplementary Information The online version contains supplementary material available at https://doi.org/10.1007/s11263-026-02770-x.

Author Contributions All authors contributed to the conception of the paper, the formulation of the method, and the interpretation of the results. The concept of the rotation representation, the implementation and the experiments were developed by Andreas Kriegler. The first draft of the manuscript was written by Andreas Kriegler, and all authors continuously revised the manuscript. All authors read and approved the final manuscript.

Funding Open access funding provided by AIT Austrian Institute of Technology GmbH. This work was funded by the Lighthouse Project AI-Enabled Sustainable Automation and Robotics of the Austrian Institute of Technology (AIT).

Data Availability All experiments make use of the T-LESS dataset (version 2), publicly available at https://huggingface.co/datasets/bop-benchmark/tless (Accessed: 2026-01-12). Predictions used to calculate the results from Tables 3, 4, 5 and 6 (other methods and ours) are available at https://github.com/akriegler/SARR/tree/main/results (Accessed: 2026-01-12).

Declarations

Competing Interests The authors have no interests to declare that had an influence on the content of this article.

Code Availability Code has been released on GitHub alongside this article and is available at https://github.com/akriegler/SARR (Accessed: 2026-01-12).

Materials Availability Not applicable.

Ethics Approval Not applicable.

Consent for Publication Not applicable.

Consent for Participation Not applicable.

References

Ayoub, E., Levesque, P., & Sharf, I. (2023). Grasp Planning with CNN for Log-loading Forestry Machine. In 2023 IEEE International Conference on Robotics and Automation (ICRA), pp. 11802–11808. IEEE, London, United Kingdom. https://doi.org/10.1109/ICRA48891.2023.10161562

Bregier, R., Devernay, F., Leyrit, L., & Crowley, J.L. (2017). Symmetry Aware Evaluation of 3D Object Detection and Pose Estimation in Scenes of Many Parts in Bulk. In 2017 IEEE International Conference on Computer Vision Workshops (ICCVW), pp. 2209–2218. IEEE, Venice, Italy. DOI: https://doi.org/10.1109/ICCVW.2017.258

Brégier, R., Devernay, F., Leyrit, L., & Crowley, J. L. (2018). Defining the Pose of Any 3D Rigid Object and an Associated Distance. *International Journal of Computer Vision (IJCV), 126*(6), 571–596. https://doi.org/10.1007/s11263-017-1052-4

Brachmann, E., Krull, A., Michel, F., Gumhold, S., Shotton, J., & Rother, C. (2014). Learning 6D Object Pose Estimation Using 3D Object Coordinates. In European Conference on Computer Vision (ECCV), vol. 8690, pp. 536–551. Springer, Zurich, Switzerland. https://doi.org/10.1007/978-3-319-10605-2_35

Banerjee, P., Shkodrani, S., Moulon, P., Hampali, S., Han, S., Zhang, F., Zhang, L., Fountain, J., Miller, E., Basol, S., Newcombe, R., Wang, R., Engel, J.J., & Hodan, T. (2025). HOT3D: Hand and Object Tracking in 3D from Egocentric Multi-View Videos. In: 2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 7061–7071. IEEE, Nashville, TN, USA. https://doi.org/10.1109/CVPR52734.2025.00662

Cai, D., Heikkilä, J., & Rahtu, E. (2022). SC6D: Symmetry-agnostic and Correspondence-free 6D Object Pose Estimation. In 2022 International Conference on 3D Vision (3DV), pp. 536–546. IEEE, Prague, Czech Republic. https://doi.org/10.1109/3DV57658.2022.00065

Chen, W., Jia, X., Chang, H.J., Duan, J., Shen, L., & Leonardis, A. (2021). FS-Net: Fast Shape-based Network for Category-Level 6D Object Pose Estimation with Decoupled Rotation Mechanism. In 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 1581–1590. IEEE, Nasville, TN, USA. https://doi.org/10.1109/CVPR46437.2021.00163

Chen, K., James, S., Sui, C., Liu, Y.-H., Abbeel, P., & Dou, Q. (2023). StereoPose: Category-Level 6D Transparent Object Pose Estimation from Stereo Images via Back-View NOCS. In 2023 IEEE International Conference on Robotics and Automation (ICRA), pp. 2855–2861. IEEE, London, United Kingdom. https://doi.org/10.1109/ICRA48891.2023.10160780

Castro, P., & Kim, T.-K. (2023). CRT-6D: Fast 6D Object Pose Estimation with Cascaded Refinement Transformers. In 2023 IEEE/CVF Winter Conference on Applications of Computer Vision (WACV), pp. 5735–5744. IEEE, Waikoloa, HI, USA. DOI: https://doi.org/10.1109/WACV56688.2023.00570

Corona, E., Kundu, K., & Fidler, S. (2018). Pose Estimation for Objects with Rotational Symmetry. In 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 7215–7222. IEEE, Madrid, Spain. https://doi.org/10.1109/IROS.2018.8594282

Chao, P., Kao, C.-Y., Ruan, Y., Huang, C.-H., & Lin, Y.-L. (2019). HarDNet: A Low Memory Traffic Network. In 2019 IEEE/CVF International Conference on Computer Vision (ICCV), pp. 3551–3560. IEEE, Seoul, Korea (South). https://doi.org/10.1109/ICCV.2019.00365

Calli, B., Singh, A., Bruce, J., Walsman, A., Konolige, K., Srinivasa, S., Abbeel, P., & Dollar, A. M. (2017). Yale-CMU-Berkeley dataset for robotic manipulation research. *International Journal of Robotics Research, 36*(3), 261–268. https://doi.org/10.1177/0278364917700714

Doumanoglou, A., Kouskouridas, R., Malassiotis, S., & Kim, T.-K. (2016). Recovering 6D Object Pose and Predicting Next-Best-View in the Crowd. In 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 3583–3592. IEEE, Las Vegas, NV, USA. https://doi.org/10.1109/CVPR.2016.390

Drost, B., Ulrich, M., Bergmann, P., Hartinger, P., & Steger, C. (2017). Introducing MVTec ITODD – A Dataset for 3D Object Recognition in Industry. In 2017 IEEE International Conference on Computer Vision Workshops (ICCVW), pp. 2200–2208. IEEE, Venice, Italy. https://doi.org/10.1109/ICCVW.2017.257

Drost, B., Ulrich, M., Navab, N., & Ilic, S. (2010). Model globally, match locally: Efficient and robust 3D object recognition. In 2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR), pp. 998–1005. IEEE, San Francisco, CA, USA . https://doi.org/10.1109/CVPR.2010.5540108

Guo, A., Wen, B., Yuan, J., Tremblay, J., Tyree, S., Smith, J., & Birchfield, S. (2023). HANDAL: A Dataset of Real-World Manipulable Object Categories with Pose Annotations, Affordances, and Reconstructions. In 2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 11428–11435. IEEE, Detroit, MI, USA. https://doi.org/10.1109/IROS55552.2023.10341672

Haugaard, R.L., Hagelskjær, F., & Iversen, T.M. (2023). SpyroPose: SE(3) Pyramids for Object Pose Distribution Estimation. In 2023 IEEE/CVF International Conference on Computer Vision Workshops (ICCVW), pp. 2074–2083. IEEE, Paris, France . DOI: https://doi.org/10.1109/ICCVW60793.2023.00222

Hodaň, T., Haluza, P., Obdrzalek, S., Matas, J., Lourakis, M., & Zabulis, X. (2017). 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), pp. 880–888. IEEE, Santa Rosa, CA, USA. https://doi.org/10.1109/WACV.2017.103

Huang, J., Liang, J., Hu, J., Sundermeyer, M., Yu, P.K., Navab, N., & Busam, B. (2025). XYZ-IBD: A High-precision Bin-picking Dataset for Object 6D Pose Estimation Capturing Real-world Industrial Complexity. arXiv. https://doi.org/10.48550/arXiv.2506.00599

Hinterstoisser, S., Lepetit, V., Ilic, S., Holzer, S., Bradski, G., Konolige, K., & Navab, N. (2013). Model Based Training, Detection and Pose Estimation of Texture-Less 3D Objects in Heavily Cluttered Scenes. In Hutchison, D., Kanade, T., Kittler, J., Kleinberg, J.M., Mattern, F., Mitchell, J.C., Naor, M., Nierstrasz, O., Pandu Rangan, C., Steffen, B., Sudan, M., Terzopoulos, D., Tygar, D., Vardi, M.Y., Weikum, G., Lee, K.M., Matsushita, Y., Rehg, J.M., Hu, Z. (eds.) Computer Vision - ACCV 2012, vol. 7724, pp. 548–562. Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-642-37331-2_42

Hodaň, T., Michel, F., Brachmann, E., Kehl, W., Buch, A.G., Kraft, D., Drost, B., Vidal, J., Ihrke, S., Zabulis, X., Sahin, C., Manhardt, F., Tombari, F., Kim, T.-K., Matas, J., & Rother, C. (2018). BOP: Benchmark for 6D Object Pose Estimation. In European Conference on Computer Vision (ECCV), pp. 19–35. Springer, Munich, Germany. https://doi.org/10.1007/978-3-030-01249-6_2

Hodaň, T., Matas, J., & Obdržálek, Š. (2016). On Evaluation of 6D Object Pose Estimation. In European Conference on Computer Vision Workshops (ECCVW), pp. 606–619. Springer, Amsterdam, Netherlands. https://doi.org/10.1007/978-3-319-49409-8_52

Hodaň, T., Sundermeyer, M., Drost, B., Labbé, Y., Brachmann, E., Michel, F., Rother, C., & Matas, J. (2020). BOP Challenge 2020 on 6D Object Localization. In European Conference on Computer Vision Workshops (ECCVW), pp. 577–594. Springer, Online. https://doi.org/10.1007/978-3-030-66096-3_39

Huynh, D. Q. (2009). Metrics for 3D Rotations: Comparison and Analysis. *Journal of Mathematical Imaging and Vision, 35*(2), 155–164. https://doi.org/10.1007/s10851-009-0161-2

Hara, K., Vemulapalli, R., & Chellappa, R. (2017). Designing Deep Convolutional Neural Networks for Continuous Object Orientation Estimation. https://doi.org/10.48550/arXiv.1702.01499

He, Y., Wang, Y., Fan, H., Sun, J., & Chen, Q. (2022). FS6D: Few-Shot 6D Pose Estimation of Novel Objects. In 2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), New Orleans, LA, USA, pp. 6804–6814. https://doi.org/10.1109/CVPR52688.2022.00669

Hodaň, T., Zabulis, X., Lourakis, M., Obdrzalek, S., & Matas, J. (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), pp. 4421–4428. IEEE, Hamburg, Germany. https://doi.org/10.1109/IROS.2015.7354005

Irshad, M.Z., Kollar, T., Laskey, M., Stone, K., & Kira, Z. (2022). CenterSnap: Single-Shot Multi-Object 3D Shape Reconstruction and Categorical 6D Pose and Size Estimation. In 2022 International Conference on Robotics and Automation (ICRA), pp. 10632–10640. IEEE, Philadelphia, PA, USA. https://doi.org/10.1109/ICRA46639.2022.9811799

Kingma, D.P., & Ba, J. (2017). Adam: A Method for Stochastic Optimization. In International Conference on Learning Representations (ICLR). Ithaca, NY: ArXiv, San Diego, CA, US. https://hdl.handle.net/11245/1.505367

Kriegler, A., Beleznai, C., Gelautz, M., Murschitz, M., & Göbel, K. (2023). PrimitivePose: Generic Model and Representation for 3D Bounding Box Prediction of Unseen Objects. *International Journal of Semantic Computing, 17*(3), 387–410. https://doi.org/10.1142/S1793351X23620027

Kriegler, A., Beleznai, C., Murschitz, M., Göbel, K., & Gelautz, M. (2022). PrimitivePose: 3D Bounding Box Prediction of Unseen Objects via Synthetic Geometric Primitives. In 2022 Sixth IEEE International Conference on Robotic Computing (IRC), pp. 190–197. IEEE, Naples, Italy. https://doi.org/10.1109/IRC55401.2022.00040

Kalra, A., Stoppi, G., Marin, D., Taamazyan, V., Shandilya, A., Agarwal, R., Boykov, A., Chong, T.H., & Stark, M. (2024). Towards Co-Evaluation of Cameras, HDR, and Algorithms for Industrial-Grade 6DoF Pose Estimation. In 2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 22691–22701. IEEE, Seattle, WA, USA. https://doi.org/10.1109/CVPR52733.2024.02141

Kaskman, R., Zakharov, S., Shugurov, I., & Ilic, S. (2019). HomebrewedDB: RGB-D Dataset for 6D Pose Estimation of 3D Objects. In 2019 IEEE/CVF International Conference on Computer Vision Workshop (ICCVW), pp. 2767–2776. IEEE, Seoul, Korea (South). https://doi.org/10.1109/ICCVW.2019.00338

Labbé, Y., Carpentier, J., Aubry, M., & Sivic, J. (2020). CosyPose: Consistent Multi-view Multi-object 6D Pose Estimation. In European Conference on Computer Vision (ECCV), pp. 574–591. Springer, Online . https://doi.org/10.1007/978-3-030-58520-4_34

Lin, T.-Y., Goyal, P., Girshick, R., He, K., & Dollar, P. (2017). Focal Loss for Dense Object Detection. In 2017 IEEE International Conference on Computer Vision (ICCV), pp. 2999–3007. IEEE, Venice, Italy. https://doi.org/10.1109/ICCV.2017.324

Liu, X., Iwase, S., & Kitani, K.M. (2021). StereOBJ-1M: Large-scale Stereo Image Dataset for 6D Object Pose Estimation. In 2021 IEEE/CVF International Conference on Computer Vision (ICCV), pp. 10850–10859. IEEE, Montreal, QC, Canada. https://doi.org/10.1109/ICCV48922.2021.01069

Li, Y., Mao, Y., Bala, R., & Hadap, S. (2024). MRC-Net: 6-DoF Pose Estimation with MultiScale Residual Correlation. In 2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 10476–10486. IEEE, Seattle, WA, USA. https://doi.org/10.1109/CVPR52733.2024.00997

Lenc, K., & Vedaldi, A. (2019). Understanding Image Representations by Measuring Their Equivariance and Equivalence. *International Journal of Computer Vision, 127*(5), 456–476. https://doi.org/10.1007/s11263-018-1098-y

Liu, X., Zhang, R., Zhang, C., Wang, G., Tang, J., Li, Z., & Ji, X. (2025). GDRNPP: A Geometry-guided and Fully Learning-based Object Pose Estimator. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI), 47*(7), 5742–5759. https://doi.org/10.1109/TPAMI.2025.3553485

Mahendran, S., Ali, H., & Vidal, R. (2017). 3D Pose Regression Using Convolutional Neural Networks. In 2017 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), pp. 494–495. IEEE, Honolulu, HI, USA. https://doi.org/10.1109/CVPRW.2017.73

Morrison, D., Corke, P., & Leitner, J. (2020). Learning robust, real-time, reactive robotic grasping. *The International Journal of Robotics Research, 39*(2–3), 183–201. https://doi.org/10.1177/0278364919859066

Mo, N., Gan, W., Yokoya, N., & Chen, S. (2022). ES6D: A Computation Efficient and Symmetry-Aware 6D Pose Regression Framework. In 2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 6708–6717. IEEE, New Orleans, LA, USA. https://doi.org/10.1109/CVPR52688.2022.00660

Pitteri, G., Bugeau, A., Ilic, S., & Lepetit, V. (2021). 3D Object Detection and Pose Estimation of Unseen Objects in Color Images with Local Surface Embeddings. In Asian Conference on Computer Vision (ACCV), pp. 38–54. Springer, Cham. https://doi.org/10.1007/978-3-030-69525-5_3

Periyasamy, A.S., Denninger, L., & Behnke, S. (2022). Learning Implicit Probability Distribution Functions for Symmetric Orientation Estimation from RGB Images Without Pose Labels. In 2022 Sixth IEEE International Conference on Robotic Computing (IRC), pp. 221–228. IEEE, Naples, Italy. https://doi.org/10.1109/IRC55401.2022.00044

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Kopf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., Bai, J., & Chintala, S. (2019). PyTorch: An Imperative Style, High-Performance Deep Learning Library. In Proceedings of the 33rd International Conference on Neural Information Processing Systems (NIPS), pp. 8026–8037. ACM, Vancouver, BC, Canada. https://doi.org/10.5555/3454287.3455008

Pitteri, G., Ramamonjisoa, M., Ilic, S., & Lepetit, V. (2019). On Object Symmetries and 6D Pose Estimation from Images. In 2019 International Conference on 3D Vision (3DV), pp. 614–622. IEEE, Québec City, QC, Canada. https://doi.org/10.1109/3DV.2019.00073

Raj, P., Kumar, A., Sanap, V., Sandhan, T., & Behera, L. (2022). Towards Object Agnostic and Robust 4-DoF Table-Top Grasping. In 2022 IEEE 18th International Conference on Automation Science and Engineering (CASE), Mexico City, Mexico, pp. 963–970. https://doi.org/10.1109/CASE49997.2022.9926708

Rad, M., & Lepetit, V. (2017). BB8: A Scalable, Accurate, Robust to Partial Occlusion Method for Predicting the 3D Poses of Challenging Objects without Using Depth. In 2017 IEEE International Conference on Computer Vision (ICCV), pp. 3848–3856. IEEE, Venice, Italy. https://doi.org/10.1109/ICCV.2017.413

Rennie, C., Shome, R., Bekris, K. E., & De Souza, A. F. (2016). A Dataset for Improved RGBD-Based Object Detection and Pose Estimation for Warehouse Pick-and-Place. *IEEE Robotics and Automation Letters, 1*(2), 1179–1185. https://doi.org/10.1109/LRA.2016.2532924

Shi, Y., Huang, J., Xu, X., Zhang, Y., & Xu, K. (2021). StablePose: Learning 6D Object Poses from Geometrically Stable Patches. In 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 15217–15226. IEEE, Nashville, TN, USA. https://doi.org/10.1109/CVPR46437.2021.01497

Salehi, S. S. M., Khan, S., Erdogmus, D., & Gholipour, A. (2019). Real-Time Deep Pose Estimation With Geodesic Loss for Image-to-Template Rigid Registration. *IEEE Transactions on Medical Imaging, 38*(2), 470–481. https://doi.org/10.1109/TMI.2018.2866442

Sundermeyer, M., Marton, Z.-C., Durner, M., Brucker, M., & Triebel, R. (2018). Implicit 3D Orientation Learning for 6D Object Detection from RGB Images. In: European Conference on Computer Vision (ECCV), pp. 712–729. Springer, Munich, Germany. https://doi.org/10.1007/978-3-030-01231-1_43

Sundermeyer, M., Marton, Z.-C., Durner, M., & Triebel, R. (2020). Augmented Autoencoders: Implicit 3D Orientation Learning for 6D Object Detection. *International Journal of Computer Vision, 128*(3), 714–729. https://doi.org/10.1007/s11263-019-01243-8

Su, Y., Saleh, M., Fetzer, T., Rambach, J., Navab, N., Busam, B., Stricker, D., & Tombari, F. (2022). ZebraPose: Coarse to Fine Surface Encoding for 6DoF Object Pose Estimation. In 2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 6728–6738. IEEE, New Orleans, LA, USA. https://doi.org/10.1109/CVPR52688.2022.00662

Stewart, I. (2013). Symmetry: A Very Short Introduction, 1st edn. Very Short Introductions, vol. 353. Oxford University Press, Oxford.

Tejani, A., Tang, D., Kouskouridas, R., & Kim, T.-K. (2014). Latent-Class Hough Forests for 3D Object Detection and Pose Estimation. In European Conference on Computer Vision (ECCV), pp. 462–477. Springer, Zurich, Switzerland. https://doi.org/10.1007/978-3-319-10599-4_30

Tyree, S., Tremblay, J., To, T., Cheng, J., Mosier, T., Smith, J., & Birchfield, S. (2022). 6-DoF Pose Estimation of Household Objects for Robotic Manipulation: An Accessible Dataset and Benchmark. In 2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pp. 13081–13088. IEEE, Kyoto, Japan. https://doi.org/10.1109/IROS47612.2022.9981838

Vidal, J., Lin, C.-Y., Lladó, X., & Martí, R. (2018). A Method for 6D Pose Estimation of Free-Form Rigid Objects Using Point Pair Features on Range Data. *Sensors,18*(8), 2678. https://doi.org/10.3390/s18082678

Wang, G., Manhardt, F., Tombari, F., & Ji, X. (2021). GDR-Net: Geometry-Guided Direct Regression Network for Monocular 6D Object Pose Estimation. In 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 16606–16616. IEEE, Nashville, TN, USA. https://doi.org/10.1109/CVPR46437.2021.01634

Wang, C., Xu, D., Zhu, Y., Martin-Martin, R., Lu, C., Fei-Fei, L., & Savarese, S. (2019). DenseFusion: 6D Object Pose Estimation by Iterative Dense Fusion. In 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 3338–3347. IEEE, Long Beach, CA, USA. https://doi.org/10.1109/CVPR.2019.00346

Xu, Z., & Cao, F. (2004). The essential order of approximation for neural networks. *Science in China Series F: Information Sciences, 47*(1), 97–112. https://doi.org/10.1360/02yf0221

Xu, Z.-B., & Cao, F.-L. (2005). Simultaneous Lp-approximation order for neural networks. *Neural Networks, 18*(7), 914–923. https://doi.org/10.1016/j.neunet.2005.03.013

Zhou, Y., Barnes, C., Lu, J., Yang, J., & Li, H. (2019). On the Continuity of Rotation Representations in Neural Networks. In 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 5738–5746. IEEE, Long Beach, CA, USA. https://doi.org/10.1109/CVPR.2019.00589

Zhao, H., Wei, S., Shi, D., Tan, W., Li, Z., Ren, Y., Wei, X., Yang, Y., & Pu, S. (2023). Learning Symmetry-Aware Geometry Correspondences for 6D Object Pose Estimation. In 2023 IEEE/CVF International Conference on Computer Vision (ICCV), pp. 13999–14008. IEEE, Paris, France. https://doi.org/10.1109/ICCV51070.2023.01291