CRISP: Object Pose and Shape Estimation with Test-Time Adaptation

Jingnan Shi, Rajat Talak, Harry Zhang, David Jin, and Luca Carlone
Laboratory for Information & Decision Systems (LIDS)
Massachusetts Institute of Technology
{jnshi, talak, harryz, jindavid, lcarlone}@mit.edu

Abstract

We consider the problem of estimating object pose and shape from an RGB-D image. Our first contribution is to introduce CRISP, a category-agnostic object pose and shape estimation pipeline. The pipeline implements an encoder-decoder model for shape estimation. It uses FiLM-conditioning for implicit shape reconstruction and a DPT-based network for estimating pose-normalized points for pose estimation. As a second contribution, we propose an optimization-based pose and shape corrector that can correct estimation errors caused by a domain gap. Observing that the shape decoder is well behaved in the convex hull of known shapes, we approximate the shape decoder with an active shape model, and show that this reduces the shape correction problem to a constrained linear least squares problem, which can be solved efficiently by an interior point algorithm. Third, we introduce a self-training pipeline to perform self-supervised domain adaptation of CRISP. The self-training is based on a correct-and-certify approach, which leverages the corrector to generate pseudo-labels at test time, and uses them to self-train CRISP. We demonstrate CRISP (and the self-training) on YCBV, SPE3R, and NOCS datasets. CRISP shows high performance on all the datasets. Moreover, our self-training is capable of bridging a large domain gap. Finally, CRISP also shows an ability to generalize to unseen objects. Code, pre-trained models and videos of sample results are available on the project webpage.[1]

1. Introduction

Accurately estimating the geometry of objects is necessary in many vision applications ranging from augmented reality [54] to robotics [11], and autonomous docking in space [5]. While significant progress has been made in solving object perception [47, 51, 53], real-world deployments remain challenging due to lack of generalizable models and in-domain training data [38].

Existing approaches to object perception tasks focus on instance-level 6D pose estimation [19, 20], category-level object 6D pose and shape estimation [44, 53], 3D bounding box estimation [2, 28], and more recently category-agnostic object pose and shape estimation [27]. Approaches for the first two tasks usually require object CAD model or category-level priors at inference time. On the other hand, 3D bounding box may be too coarse a representation to be useful in many applications (*e.g.* autonomous space debris removal [31]). Another line of recent work focuses on generalized unseen object pose estimation, but requires an object onboarding stage [13, 42, 50]. Recent work on conditional diffusion models have shown promise in single-view 3D reconstruction [17, 25]. However, their inference time remains prohibitively high for real-time operation [25].

In addition, the domain gap issue, where the distribution of the test data differs from training, remains a major concern. Developing an object perception model is not enough as a domain gap can render its estimates completely useless, or worse still, hazardous in safety-critical applications [38]. Therefore, in conjunction to the object perception model, it

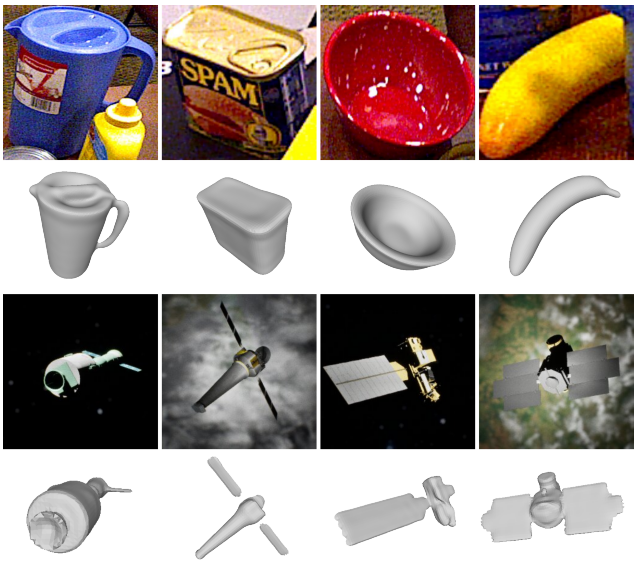


Figure 1. We introduce CRISP, a category-agnostic object pose and shape estimation pipeline, and a test-time adaptive self-training method CRISP-ST to bridge domain gaps. *Top:* Qualitative examples of CRISP on the YCBV dataset [51]. *Bottom:* Qualitative examples of CRISP on the SPE3R dataset [31].

[1] https://web.mit.edu/sparklab/research/crisp_object_pose_shape/

is necessary to develop methods that can bridge any large domain gap and work well at test-time.

Towards addressing these issues, this paper presents three main contributions:

- We introduce CRISP, an object pose and shape estimation pipeline. CRISP combines a pre-trained vision transformer (ViT) backbone with a dense prediction transformer (DPT) and feature-wise linear modulation (FiLM) conditioning to estimate the 6D pose and shape of the 3D object from a single RGB-D image [35, 37]. CRISP is category-agnostic (*i.e.*, it does not require knowledge of the object category at test time) and requires no object onboarding.
- We propose an optimization-based pose and shape corrector that can correct estimation errors. The corrector is a bi-level optimization problem and we use block coordinate descent to solve it. We approximate the shape decoder in CRISP by an active shape model, and show that (i) this is a reasonable approximation, and (ii) doing so turns it into a constrained linear least squares problem, which can be solved efficiently using interior-point methods and yields just as good shapes as the trained decoder.
- We adapt a *correct-and-certify* approach to self-train CRISP and bridge any large domain gap. During self-training, we use the corrector to correct for pose and shape estimation errors. Then, we assert the quality of the output of the corrector using an observable correctness certificate inspired by [40], and create pseudo-labels using the estimates that pass the certificate check. Finally, we train the model on these pseudo-labels with standard stochastic gradient descent. Contrary to [27, 34], we do not need access to synthetic data during self-training.

We demonstrate CRISP on the YCBV, SPE3R, and NOCS datasets. CRISP shows high performance on all the datasets. Moreover, our self-training is able to bridge a large domain gap. Finally, we observe that CRISP is able to generalize and estimate pose and shape of objects unseen during training, and performs inference at 8 Hz.

2. Related Work

Object Pose and Shape Estimation. The task of object pose and shape estimation is to recover the 3D poses and shape of an object under observation. This is different from the task of instance-level pose estimation, which usually assumes object CAD models are provided during test-time inference [8, 19, 20, 23, 46, 48, 51]. Category-level approaches have been proposed, which aim to simultaneously estimate the shape and pose of the objects, despite large intra-class shape variations. One common paradigm in category-level pose and shape estimation is to first regress some intermediate representations, and solve for poses and shapes [33, 47, 53]. Wang *et al.* propose Normalized Object Coordinates (NOCS) to represent the objects as well as use them to recover pose and shape [47]. Many recent works adopt category-specific shape priors, which are deformed to match the observation during inference [44, 53]. However, the use of category-level priors limits their scalability.

Recently, there have been a lot of works focusing on the task of generalized unseen object pose estimation [3, 13, 42, 50]. The typical setup involves an object onboarding phase, where 3D meshes or posed images and masks of the exact test objects are provided [15]. This limits the applicability of such approaches to situations where the test objects are available prior to inference.

There have also been a number of works using neural implicit fields for object pose and shape estimation [16, 27, 34]. Peng *et al.* [34] combine PointNet [36] with a DeepSDF decoder [30] for representing shapes. Irshad *et al.* [16] learn a disentangled implicit shape and apperance model for joint appearance, shape and pose optimization. Comparing with conditional diffusion models that are prevalent in the shape reconstruction literature [18, 25], such regression-based methods are significantly faster for inference, making them suitable for real-time applications.

Test-Time Adaptation. Test-time adaptation allows methods to handle distribution shifts between training and testing, leading to better performance on unseen domains [52]. Wang *et al.* [45] train a pose estimation model on synthetic RGB-D data, and then refine it further with differentiable rendering on real, unannotated data. However, the approach is limited to objects with known CAD models. Lunayach *et al.* [27] use a self-supervised chamfer loss to provide training signals on unannotated data. However, it requires access to a set of synthetic labeled data during self-supervision to stablize training, making it less suitable for test-time adaption in real-world robotics deployments. Lee *et al.* [22] propose TTA-COPE, a test-time adaptation method that does not require access of source domain data, but it does not perform shape reconstruction. Talak *et al.* [43] propose a correct-and-certify approach to self-train pose estimation models based on a corrector and binary certificates, which is further extended in [40] to handle outliers. However, the approach assumes known CAD models at test-time.

3. Object Pose and Shape Estimation

We consider the problem of object pose and shape estimation from an RGB-D image $(\mathcal{I}, \boldsymbol{X})$; where $\mathcal{I}$ is the RGB image of the detected object and $\boldsymbol{X} = [\boldsymbol{x}_1, \boldsymbol{x}_2, \dots \boldsymbol{x}_n] \in \mathbb{R}^{3\times n}$ is the depth point cloud of the segmented object. Each point $\boldsymbol{x}_i$ corresponds to a pixel in the image $\mathcal{I}$. Given $(\mathcal{I}, \boldsymbol{X})$, our goal is to estimate the shape $f(\cdot \mid \mathcal{I})$ and pose $(\boldsymbol{R}, \boldsymbol{t}) \in \mathrm{SO}(3) \times \mathbb{R}^3$ of the object, where $f(\cdot \mid \mathcal{I})$ is the signed distance field (SDF) of the object in a pose-normalized frame. We can write this as an optimization problem:

$$\begin{aligned} \underset{f,\boldsymbol{R},\boldsymbol{t}}{\text{Minimize}} \quad & \sum_{i=1}^{n} |f(\boldsymbol{R}\boldsymbol{x}_i + \boldsymbol{t} \mid \mathcal{I})|^2 \\ \text{subject to} \quad & (\boldsymbol{R}, \boldsymbol{t}) \in \mathrm{SO}(3) \times \mathbb{R}^3 \\ & f(\cdot \mid \mathcal{I}) \in \mathcal{F}. \end{aligned} \tag{1}$$

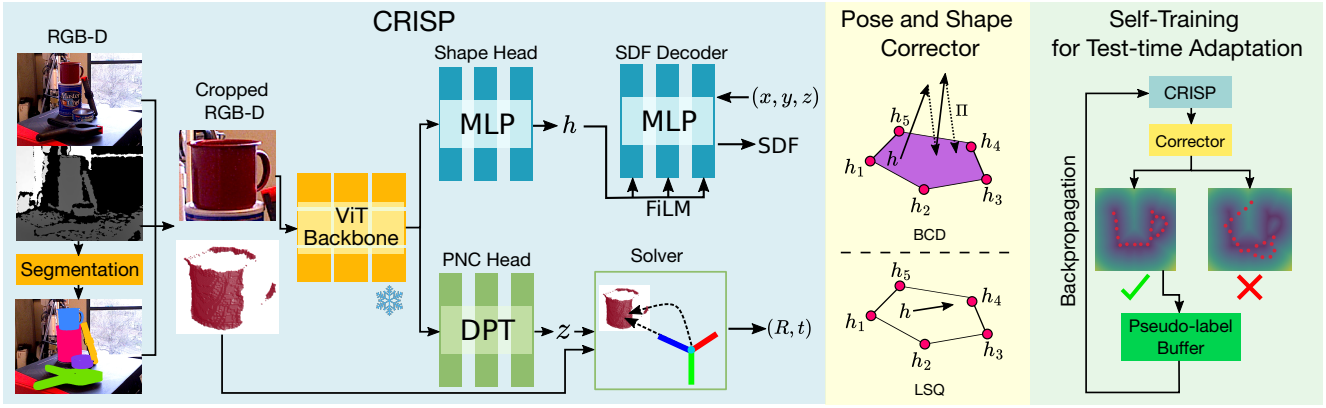


Figure 2. Overview of our contributions. Given an segmented RGB image $\mathcal{I}$ and depth points $\boldsymbol{X}$ of the object, CRISP extracts features from the cropped image. It estimates the object pose, by estimating pose-normalized coordinates (PNC) $\boldsymbol{Z}$, and shape, by reconstructing the signed distance field (SDF) of the object. The pose and shape estimates are corrected by the corrector which solves a bi-level optimization problem using two solvers: BCD (Alg. 1) and LSQ (Alg. 2). The self-training uses corrected estimates that pass the observable certification check (12) as pseudo-labels. The SDF decoder is fixed during self-training.

where the objective forces the depth points measured on the object surface to match the zero-crossing of the SDF. This problem is hard because the space of shapes $\mathcal{F}$ and the input $\mathcal{I}$ are not easy to characterize. Secondly, the problem is not well specified, if $\mathcal{F}$ is not sufficiently constrained by a set of priors. Thirdly, estimating $(\boldsymbol{R}, \boldsymbol{t})$, by solving (1) is difficult as the objective is non-linear and non-convex, and the set of object poses, *i.e.*, $\mathrm{SO}(3) \times \mathbb{R}^3$, is a manifold.

We now present how learning-based components can be used to obviate these issues and approximate the optimization problem (1) by performing forward passes on the learning-based components. This forms our category-agnostic pose and shape estimation pipeline CRISP (Fig. 2).

3.1. Shape Estimation

We use a deep encoder-decoder architecture to estimate the object shape from an image (Fig. 2). An encoder network estimates a latent shape code $\boldsymbol{h} = f_e(\mathcal{I}) \in \mathbb{R}^d$, from the image $\mathcal{I}$. A shape decoder produces an SDF $f_d(\cdot \mid \boldsymbol{h})$ of the object, given the latent shape code $\boldsymbol{h}$. The shape $f_d(\cdot \mid \boldsymbol{h})$ is in a pose-normalized frame, *i.e.*, centered at origin and is invariant to the pose of the object in the image.

Architecture Details. Given an RGB-D image, we first extract features tokens with a pre-trained ViT backbone (*e.g.*, DINOv2 [29]). These are concatenated and passed through a multi-layer perceptron (MLP) to regress the latent shape code $\boldsymbol{h}$. This forms our encoder $\boldsymbol{h} = f_e(\mathcal{I})$. The shape decoder is a MLP with sinusoidal activations [41]. We use FiLM [35] to condition the decoder. FiLM conditioning allows us to condition the SDF without any explicit category labels. This has been shown to produce better reconstruction results for neural implicit fields [4], in comparison to other popular methods like conditioning-by-concatenation.

If the encoder estimates the correct shape, the pose and shape estimation problem (1) reduces to

$$\begin{aligned} \underset{\boldsymbol{R},\boldsymbol{t}}{\text{Minimize}} \quad & \sum_{i=1}^{n} |f_d(\boldsymbol{R}\boldsymbol{x}_i + \boldsymbol{t} \mid \boldsymbol{h})|^2 \\ \text{subject to} \quad & (\boldsymbol{R}, \boldsymbol{t}) \in \mathrm{SO}(3) \times \mathbb{R}^3, \boldsymbol{h} = f_e(\mathcal{I}), \end{aligned} \tag{2}$$

where the pose estimation only depends on the decoder network f_d. This is still a hard problem as it requires one to optimize a non-convex function over a manifold.

3.2. Pose Estimation

We now use another network to simplify the pose estimation problem (2). Inspired by [47], we use a deep neural network to estimate, for each pixel and depth point $\boldsymbol{x}_i$, a corresponding point $\boldsymbol{z}_i$ in the pose-normalized frame. We estimate these pose normalized coordinates (PNC) $\boldsymbol{Z} = [\boldsymbol{z}_1, \boldsymbol{z}_2, \ldots \boldsymbol{z}_n]$ directly from the image, *i.e.*

$$\boldsymbol{Z} = [\phi_1(\mathcal{I}), \phi_2, (\mathcal{I}), \ldots \phi_n(\mathcal{I})] = \Phi(\mathcal{I}). \tag{3}$$

Architecture Details. We use a Dense Prediction Transformer (DPT) to estimate the PNC [37]. The DPT uses feature tokens extracted from the image $\mathcal{I}$ and passes them through trainable reassemble blocks. These are then progressively added and upsampled together to produce a fine-grained prediction. The final output head is a CNN that directly regresses the PNC $\boldsymbol{Z}$. See Fig. 2.

If the estimated PNC $\boldsymbol{Z}$ are correct, then the object pose $(\boldsymbol{R}, \boldsymbol{t})$, in (2), can be obtained by directly solving

$$\boldsymbol{z}_i = \boldsymbol{R}\boldsymbol{x}_i + \boldsymbol{t} \;\; \forall \, i \in [n], \tag{4}$$

for $\boldsymbol{R} \in \mathrm{SO}(3)$ and $\boldsymbol{t} \in \mathbb{R}^3$. We solve (4) using Arun's method [1], which obtains $(\boldsymbol{R}, \boldsymbol{t})$ by minimizing the least squares error in satisfying (4).

Remark 1 (Difference from NOCS [47])**.** *Contrary to NOCS [47], we do not normalize $\boldsymbol{Z}$. This is crucial for self-training: involving scale estimation leads to degeneracies of PNC, causing $\boldsymbol{Z}$ to drift away from the ground truth (see Appendix E).*

3.3. Supervised Training

CRISP can be trained on both synthetic and real-world annotated data. We require no category labels — any collection of CAD models can be used and they may belong to different categories. We use a soft L_1 loss on PNC $\boldsymbol{Z}$ and the SDF loss specified in [41] (Appendix D). In Section 5, we will see how to use the corrector to self-train CRISP, when the trained model suffers a domain gap at test-time.

4. Pose and Shape Correction

In Section 3, we saw that the pose and shape estimation problem (1) is obviated if CRISP estimates the correct latent shape code and PNC. In this section, we re-instate the optimization problem and tackle the case when the estimates from the learning-based components deviate from the ground-truth. We derive an optimization-based corrector that helps correct the estimated object pose and shape (Section 4.1). We then propose an active shape decoder, which simplifies the shape correction problem to a constrained linear least squares problem (Section 4.2). This active shape decoder enables faster refinement (Section 6).

4.1. Pose and Shape Corrector

We now re-state the pose and shape optimization problem (2), and treat the latent shape code $\boldsymbol{h}$ and the PNC $\boldsymbol{Z}$ (as in (4)) as variables. We obtain:

$$\begin{aligned} \underset{\boldsymbol{Z},\boldsymbol{h},\boldsymbol{R},\boldsymbol{t}}{\text{Minimize}} \quad & \sum_{i=1}^{n} |f_d(\, \boldsymbol{z}_i |\, \boldsymbol{h})|^2 \\ \text{subject to} \quad & \boldsymbol{z}_i = \boldsymbol{R}\boldsymbol{x}_i + \boldsymbol{t} \;\forall\, i \in [n]. \end{aligned} \tag{5}$$

Let $\boldsymbol{h}^*$ and $\boldsymbol{Z}^*$ denote the global optima of problem (5). The goal of the corrector is to ensure that the network estimates, namely,

$$\boldsymbol{Z} = \Phi(\mathcal{I}) \quad \text{and} \quad \boldsymbol{h} = f_e(\mathcal{I}), \tag{6}$$

get corrected to $\boldsymbol{Z}^*$ and $\boldsymbol{h}^*$.

Remark 2 (Correcting to Ground Truth)**.** *Assume that the shape decoder is well trained,* i.e.*, for any object observed at test-time there is a latent shape code $\boldsymbol{h}$ that generates the signed distance field of the object via $f_d(\cdot \mid \boldsymbol{h})$. Let $\boldsymbol{h}^*$ and $\boldsymbol{Z}^*$ denote the ground-truth latent shape code and PNC, respectively. Then, it is trivial to see that the corrector objective will be zero, as $f_d(\boldsymbol{z}_i^* \mid \boldsymbol{h}^*) = 0$ for all i.*

Ideally, one could solve (5) to global optimality without any network estimates. However, the problem (5) does not have an easy global solver. Therefore, we use the network estimates (6) as an initialization, followed by gradient descent (GD) to obtain a solution to (5). If the model estimates provide a good initialization, the corrector will obtain a solution that is close to the global optima.

Corrector Implementation. First, we simplify the corrector problem (5) by relaxing the hard constraints (*i.e.* $\boldsymbol{z}_i = \boldsymbol{R}\boldsymbol{x}_i + \boldsymbol{t}$). In Appendix A, we show that (5) is equivalent to the bi-level optimization problem:

$$\begin{aligned} \underset{\boldsymbol{Z}\in\mathbb{R}^{3\times n},\boldsymbol{h}\in\mathbb{R}^d}{\text{Minimize}} \quad & \sum_{i=1}^{n} |f_d(\hat{\boldsymbol{R}}\boldsymbol{x}_i + \hat{\boldsymbol{t}} |\, \boldsymbol{h})|^2 \\ \text{subject to} \quad & (\hat{\boldsymbol{R}}, \hat{\boldsymbol{t}}) = \underset{(\boldsymbol{R},\boldsymbol{t})}{\text{argmin}} \sum_{i=1}^{n} \left\| \boldsymbol{R}\boldsymbol{x}_i + \boldsymbol{t} - \boldsymbol{z}_i \right\|^2, \end{aligned} \tag{7}$$

such that the global optima of (5) can be obtained from the global optima of (7). Note that $(\hat{\boldsymbol{R}}, \hat{\boldsymbol{t}})$ depend on $\boldsymbol{Z}$. Denote $F(\boldsymbol{Z} \mid \boldsymbol{h}) = \sum_{i=1}^{n} |f_d(\hat{\boldsymbol{R}}\boldsymbol{x}_i + \hat{\boldsymbol{t}} |\, \boldsymbol{h})|^2$ where $(\hat{\boldsymbol{R}}, \hat{\boldsymbol{t}})$ are given by the constraint in (7). The bi-level optimization problem can be written as:

$$\underset{\boldsymbol{Z}\in\mathbb{R}^{3\times n},\boldsymbol{h}\in\mathbb{R}^d}{\text{Minimize}} \quad F(\boldsymbol{Z} \mid \boldsymbol{h}). \tag{8}$$

The corrector solves this using model estimates as initialization (6) and block coordinate descent (BCD); see Alg. 1. We first solve (8) using gradient descent, with $\boldsymbol{Z}$ as variable and use network estimate as initialization (Alg. 1, Line 2). We then solve (8) with $\boldsymbol{h}$ as variable (initialized with network estimate) using projected gradient descent, initialize $\boldsymbol{Z}$ with $\hat{\boldsymbol{Z}}$ (Alg. 1, Line 3). The rationale behind using projection is due to our observation that the shape decoder is well behaved within the simplex, and not outside it.

To be more precise, let $\boldsymbol{h}_1, \boldsymbol{h}_2, \ldots \boldsymbol{h}_K$, be the latent shape codes for the K objects in the train set. Let $\boldsymbol{h}$ take the values in the affine space $\boldsymbol{h} = \alpha_1\boldsymbol{h}_1 + \alpha_2\boldsymbol{h}_2 + \cdots \alpha_K\boldsymbol{h}_K$, with $\sum_{k=1}^{K} \alpha_k = 1$. When $\boldsymbol{h}$ is interpolated (*i.e.*, $1 \geq \alpha_k \geq 0$) the decoder $f_d(\cdot \mid \boldsymbol{h})$ is well behaved. When it is extrapolated, the decoder produces implausible object shapes (see Fig. 3). Therefore, we use projected gradient descent (Line 3) in Alg. 1. We project the iterates onto the simplex S_K of all latent shape codes learnt in training [39]:

$$\mathsf{S}_K = \left\{ \boldsymbol{h} = \sum_{i=1}^{K} \alpha_k \boldsymbol{h}_k \;\middle|\; \sum_{k=1}^{K} \alpha_k = 1 \;\text{ and }\; \alpha_k \geq 0 \right\}.$$

For our experiments, we extend the BCD algorithm for multi-view pose and shape correction (Remark 3).

Remark 3 (Multi-View Pose and Shape Corrector)**.** *The corrector problem (5) and the BCD algorithm is specified for a single input image,* i.e.*, it corrects the pose and shape*

Algorithm 1: BCD: Block Coordinate Descent Solver for the Pose and Shape Correction (8)

1 $\boldsymbol{Z} = \Phi(\mathcal{I}), \boldsymbol{h} = f_e(\mathcal{I})$;
2 $\hat{\boldsymbol{Z}} \leftarrow \underset{\boldsymbol{Z}}{\text{argmin}}\; F(\boldsymbol{Z} \mid \boldsymbol{h})$ using Grad. Descent;
3 $\hat{\boldsymbol{h}} \leftarrow \underset{\boldsymbol{h}}{\text{argmin}}\; F(\hat{\boldsymbol{Z}} \mid \boldsymbol{h}); \boldsymbol{h} \in \mathsf{S}_K = \left\{ \boldsymbol{h} = \sum_{i=1}^{K} \alpha_k \boldsymbol{h}_k \;\middle|\; \sum_{k=1}^{K} \alpha_k = 1 \;\text{ and }\; \alpha_k \geq 0 \right\}$ using Proj. Grad. Descent;
4 Return: $\hat{\boldsymbol{h}}, \hat{\boldsymbol{Z}}$;

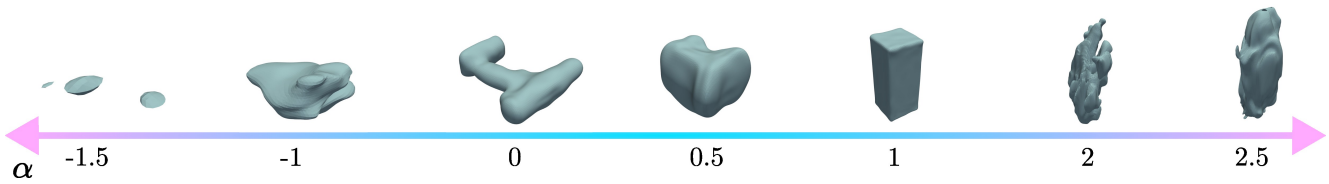


Figure 3. Visualization of the mesh extracted from SDF produced by the shape decoder $f_d(\cdot \mid \boldsymbol{h})$ as the latent shape code takes values $\boldsymbol{h} = \alpha \boldsymbol{h}_1 + (1-\alpha)\boldsymbol{h}_2$ given two shapes codes $\boldsymbol{h}_1$ and $\boldsymbol{h}_2$. The trained decoder does not produce plausible shapes at extrapolation.

of an object, estimated from a single-view. However, the corrector problem and algorithm is easily modified to correct shape using estimates from multiple views. This is done by aggregating the estimated PNC $\boldsymbol{Z}$ across views, and using the $F(\boldsymbol{Z} \mid \boldsymbol{h})$, with this aggregated $\boldsymbol{Z}$, when updating the latent shape code $\boldsymbol{h}$ in Line 3, Algorithm 1. We implement both single-view and multi-view shape corrector. In the multi-view corrector, the PNC get corrected for each view (i.e. *Line 2, Algorithm 1, remains constrained to single-view); however, using corrected shape code informed by multi-view data shows better results (Section 6).*

4.2. Active Shape Decoder

Inspired by our observation that the trained shape decoder $f_d(\cdot \mid \boldsymbol{h})$ is reliable only in the simplex S_K of $\boldsymbol{h}$, we now construct a new active shape decoder f_a, given f_d, that is reliable enough over the simplex S_K, reducing the corrector problem to a constrained linear least squares problem.

Let $\boldsymbol{h} = f_e(\mathcal{I})$ be the latent shape code estimated by the model, and $\boldsymbol{h}_1, \boldsymbol{h}_2, \dots \boldsymbol{h}_K$ be the latent shape codes of all the CAD models used in the supervised training (Section 3.3). We define an active shape decoder f_a as:

$$f_a(\boldsymbol{z} \mid \boldsymbol{c}) = c_0 d_0 f_d(\boldsymbol{z} \mid \boldsymbol{h}) + \sum_{k=1}^{K} c_k d_k f_d(\boldsymbol{z} \mid \boldsymbol{h}_k), \quad (9)$$

where $\boldsymbol{c} = [c_0, c_1, \dots c_K]^\mathsf{T}$ is a $(K+1)$-dimensional vector and d_k are some positive constants. As in (9), we limit $\boldsymbol{c}$ to be in the simplex, *i.e.*, $\sum_{k=0}^{K} c_k = 1$ and $c_k \geq 0$. The positive constants d_k prove useful in normalizing the signed distance field (see Appendix E). Note that using latent shape code $\boldsymbol{h} = f_e(\mathcal{I})$ as a "basis" in (9) is essential for the approximation to work well. We empirically observe that using the active shape decoder without the $\boldsymbol{h}$ leads to worse shape reconstruction performance (see Appendix E).

For the active shape model f_a, we now show that the shape correction (7), given PNC $\boldsymbol{Z}$, reduces to a constrained linear least squares problem. This is useful as we can directly use a constrained linear least squares solver to update the shape code in BCD (Line 3, Algorithm 1). First, note that, given $\boldsymbol{Z}$, the shape estimation problem (7) for the active shape decoder f_a can be written as

$$\begin{aligned} \underset{\boldsymbol{c} \in \mathbb{R}^{K+1}}{\text{Minimize}} \quad & \sum_{i=1}^{n} |f_a(\boldsymbol{z}_i \mid \boldsymbol{c})|^2 \\ \text{subject to} \quad & \sum_{k=1}^{K} c_k = 1 \;\text{ and }\; c_k \geq 0. \end{aligned} \quad (10)$$

Algorithm 2: LSQ: Solver for the Pose and Shape Correction (8).

1 $\boldsymbol{Z} = \Phi(\mathcal{I}), \boldsymbol{h} = f_e(\mathcal{I})$;
2 $\hat{\boldsymbol{Z}} \leftarrow \underset{\boldsymbol{Z}}{\text{argmin}}\, F(\boldsymbol{Z} \mid \boldsymbol{h})$ using Grad. Descent;
3 $\hat{\boldsymbol{c}} \leftarrow \underset{\boldsymbol{c} \geq 0, \mathbf{1}^\mathsf{T}\boldsymbol{c}=1}{\text{argmin}} \left\| \boldsymbol{F}(\hat{\boldsymbol{Z}})\boldsymbol{D}\boldsymbol{c} \right\|^2$ using Interior Point;
4 $\hat{\boldsymbol{h}} \leftarrow \hat{c}_0 d_0 \boldsymbol{h} + \sum_{k=1}^{K} \hat{c}_k d_k \boldsymbol{h}_k$;
5 Return: $\hat{\boldsymbol{h}}$, $\hat{\boldsymbol{Z}}$;

We obtain this by simply replacing the trained decoder $f_d(\cdot \mid \boldsymbol{h})$ with the active shape decoder f_a. Using (9), it is easy to show that the objective function in (10) can be simplified as $\sum_{i=1}^{n} |f_a(\boldsymbol{z}_i \mid \boldsymbol{c})|^2 = \|\boldsymbol{F}(\boldsymbol{Z})\boldsymbol{D}\boldsymbol{c}\|^2$, where $\boldsymbol{D} = \text{diag}\left([d_0, d_1, \dots d_K]\right)$ and

$$\boldsymbol{F}(\boldsymbol{Z}) = \begin{bmatrix} f_d(\boldsymbol{z}_1 \mid \boldsymbol{h}) & f_d(\boldsymbol{z}_1 \mid \boldsymbol{h}_1) & \cdots & f_d(\boldsymbol{z}_1 \mid \boldsymbol{h}_K) \\ f_d(\boldsymbol{z}_2 \mid \boldsymbol{h}) & f_d(\boldsymbol{z}_2 \mid \boldsymbol{h}_1) & \cdots & f_d(\boldsymbol{z}_2 \mid \boldsymbol{h}_K) \\ \vdots & \vdots & & \vdots \\ f_d(\boldsymbol{z}_n \mid \boldsymbol{h}) & f_d(\boldsymbol{z}_n \mid \boldsymbol{h}_1) & \cdots & f_d(\boldsymbol{z}_n \mid \boldsymbol{h}_K) \end{bmatrix}. \quad (11)$$

This results in a new solver for the corrector problem (5). This is described in Algorithm 2 (LSQ). We use interior point method to solve the constrained linear least squares problem in Line 3 of LSQ [10].

Remark 4 (Shape Degeneracy Check)**.** (10) *informs a shape degeneracy check, i.e., a check if the shape estimation is unique or not. See Appendix C for more details.*

5. Self-Training for Test-Time Adaptation

We now describe `CRISP-ST`, the self-training algorithm that self-trains `CRISP` at test-time with pesudo-labels. While self-training with pseudo-labels is not new [40, 43], to the best of our knowledge, this is the first time that it has been applied to simultaneous object pose and shape estimation. We now describe the three steps in `CRISP-ST`: *correction*, *certification*, and *self-training*.

Correction. We correct the outputs (6) using the corrector (Section 4.1) to produce refined estimates $\hat{\boldsymbol{Z}}$ and $\hat{\boldsymbol{h}}$.

Certification. We implement observable correctness certificate checks to assert the quality of the estimates from the corrector. We do so by testing geometric consistency of the corrected estimates with the observed depth. If $(\hat{\boldsymbol{R}}, \hat{\boldsymbol{t}})$ denote

the corrected pose (see (7)), then the value $|f_d(\hat{\boldsymbol{R}}\boldsymbol{x}_i + \hat{\boldsymbol{t}} \mid \boldsymbol{h})|$ must be low, barring outliers in the point cloud $\boldsymbol{X}$. Intuitively, this ensures that the pose-inverted depth is consistent with the implicit shape representation $f_d(\cdot|\boldsymbol{h})$.Therefore, we define the observably correctness certificate ($\texttt{oc}$) as:

$$\texttt{oc}(\hat{\boldsymbol{Z}}, \hat{\boldsymbol{h}}) = \mathbb{I}\left\{ \left[|f_d(\hat{\boldsymbol{z}}_i|\hat{\boldsymbol{h}}))| \right]_p < \epsilon \right\} \tag{12}$$

where $[\cdot]_p$ is the p-th quantile function w.r.t. i. If the corrected estimates $(\hat{\boldsymbol{Z}}, \hat{\boldsymbol{h}})$ pass the $\texttt{oc}$ check, then we self-annotate the image $\mathcal{I}$ with the corrected estimates.

Self-training. We train the model using the self-annotated images with the MSE training loss on both the latent shape code and the PNC:

$$L_h = \|\hat{\boldsymbol{h}} - \boldsymbol{h}\|^2 \qquad L_z = \sum_{i=1}^{n} \|\hat{\boldsymbol{z}}_i - \boldsymbol{z}_i\|^2. \tag{13}$$

As the self-training progresses, more and more corrected estimates pass the $\texttt{oc}$ check, resulting in more pseudo-labels. Note that the shape decoder is frozen during self-training.

Implementation Details. The self-training pipeline operates over a buffer of images, including different views for a scene. CRISP operates on every object detection on each image. We implement a multi-view corrector that inputs data associations of objects in different images. We also report the results of using a single-view corrector in the ablation study (Sec. 6.4).

6. Experiments

We evaluate our approach with three datasets ranging from household objects to satellites: (1) YCBV [14], where the test and train objects are the same; (2) SPE3R [31], where the test objects are unknown; (3) NOCS, a large-scale dataset with unknown test objects [47]. We implement our models in PyTorch [32]. We use DINOv2 [29] as the ViT backbone and keep it frozen during training. All supervised models are trained with 8 Nvidia V100 GPUs, while self-training experiments are conducted on one Nvidia A6000 GPU. For tables, we use bold fonts for the best results and color the top three results with , and , respectively. Additional details are provided in Appendix D.

6.1. YCBV Dataset

Setup. We evaluate CRISP and our self-training approach against baselines on the YCBV dataset [14, 51]. We train two models: CRISP-Real where we use the real-world images in the train set provided by the dataset, and CRISP-Syn where we train CRISP using 4200 (200 images per object) images rendered using BlenderProc [9] (see Appendix E for sample images). The reasoning behind using synthetic images is to evaluate test-time adaptation under a large sim-to-real gap. We train CRISP-Real with an Adam optimizer with a learning rate of 3×10^{-4} for 50 epochs, and CRISP-Syn with the same optimizer and learning rate for 500 epochs. In addition, we self-train CRISP-Syn for 5 epochs, using both BCD Alg. 1 (CRISP-Syn-ST (BCD)) and LSQ Alg. 2 (CRISP-Syn-ST (LSQ)).

Method	e_{shape} ↓ Mean	e_{shape} ↓ Median	e_{shape} (AUC) ↑ 3 cm	5 cm	10 cm
Shap-E [17]	0.099	0.052	0.05	0.17	0.43
CRISP-Syn	0.045	0.032	0.18	0.35	0.58
CRISP-Syn-ST (LSQ)	0.037	0.024	0.25	0.43	0.65
CRISP-Syn-ST (BCD)	0.039	0.028	0.19	0.39	0.62
CRISP-Real	**0.026**	**0.016**	**0.40**	**0.58**	**0.75**

Table 1. Evaluation results on the e_{shape} and e_{shape} (AUC) metrics for the YCBV dataset.

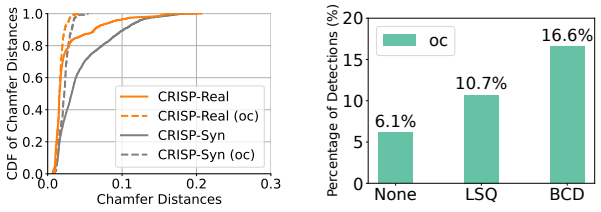
Method	ADD-S ↓ Mean	ADD-S ↓ Median	ADD-S (AUC) ↑ 1 cm	2 cm	3 cm
GSPose [3]	0.223	0.169	0.01	0.04	0.07
FoundationPose [50]	**0.006**	**0.005**	**0.46**	**0.72**	**0.81**
CosyPose [19]	0.010	0.007	0.30	0.56	0.68
BundleSDF [49]	0.014	0.012	0.14	0.37	0.55
GDRNet++ [26]	0.013	0.011	0.22	0.43	0.58
CRISP-Syn	0.020	0.015	0.12	0.30	0.45
CRISP-Syn-ST (LSQ)	0.016	0.010	0.24	0.44	0.56
CRISP-Syn-ST (BCD)	0.017	0.010	0.22	0.42	0.54
CRISP-Real	0.011	**0.005**	0.41	0.62	0.73

Table 2. Evaluation results on the ADD-S and ADD-S (AUC) metrics for the YCBV dataset.

Figure 4. Qualitative samples of CRISP on the YCBV dataset. *Top*: projection of transformed reconstructed mesh with our estimation. *Bottom:* reconstructed mesh. See Appendix for more examples.

For pose estimation, we compare against baselines including CosyPose [19], BundleSDF [49] and GDRNet++ [26, 46]. In addition, we compare against GSPose [3] and FoundationPose [50], which both require object onboarding. For shape reconstruction, we compare against Shap-E [17], a state-of-the-art conditional diffusion generative model. We report ADD-S metrics [51] and the Chamfer distance between ground truth model and reconstructed shape (labeled as e_{shape}). We also compute the area under curve (AUC) metric for both ADD-S and e_{shape} [51]. All methods use the same ground truth segmentation masks.

Results. Tab. 1 reports shape reconstruction performances. CRISP-Real performs the best, with the mean e_{shape} of 0.026, significantly outperforming Shap-E (0.099) by 73%. In addition, self-training improves CRISP-Syn's performance for all metrics. Compared to CRISP-Syn, CRISP-Syn-ST (LSQ) and CRISP-Syn-ST (BCD)



(a) (b)

Figure 5. (a) Cumulative distribution function of the Chamfer distances of the reconstructed shapes. Methods append with (oc) indicate the instances pass oc. (b) Comparison of percentages of test instances which pass oc (label: oc).

improve on mean e_{shape} by 17% and 12% and e_{shape} (AUC) (5 cm) by 22% and 11% respectively. CRISP-Syn-ST (LSQ) slightly outperforms CRISP-Syn-ST (BCD) on the e_{shape} (AUC) metric at 5 and 10 cm. Tab. 2 reports pose estimation performances, where we divide methods into two categories: using object onboarding (GSPose and FoundationPose) and without (CosyPose, BundleSDF, GDRNet++, and CRISP). CRISP-Real achieves the best performance at all ADD-S (AUC) metrics among methods without object onboarding, with CosyPose trailing behind. Our self-training approach improves CRISP-Syn's performance, with CRISP-Syn-ST (LSQ) and CRISP-Syn-ST (BCD) improve on mean ADD-S by 20% and 17% respectively and ADD-S (AUC) (2 cm) by 44% and 38% respectively. We show some qualitative examples of CRISP in Fig. 4.

To understand why self-training works, Fig. 5a shows the CDFs of e_{shape} of CRISP and CRISP-Syn with and without oc check applied. With oc, the CDF curves are shifted to the far left, indicating that the check filters out instances with high e_{shape} without access to ground truth. By collecting pseudo-labels using the oc check, we are collecting good samples that have low e_{shape}, enabling self-training. Fig. 5b shows the impact of correctors in self-training. Without corrector, CRISP-Syn only 6.1% of the outputs pass oc. With LSQ (Alg. 2) and BCD (Alg. 1), the percentage increases to 10.7% and 16.6%, respectively.

6.2. SPE3R Dataset

Setup. We evaluate CRISP on the SPE3R dataset [31], which contains photorealistic renderings of 64 real-world satellites. We use the official train set, which contains 57 satellites (800 images per satellite). Seven satellites are withheld for testing. We use an Adam optimizer with a learning rate of 10^{-4}, a batch size of 16 and for 100 epochs (label: CRISP). We also self-train CRISP for 10 epochs on the test set, using only the BCD corrector (Alg. 1) due to the unknown test objects (label: CRISP-ST). Following [31], we report the L_1 Chamfer distance between ground-truth and reconstructed shapes (label: $e^{L_1}_{shape}$) and between the ground-truth transformed model and the reconstructed model transformed using estimated pose (label: $e^{L_1}_{pose}$).

Results. As shown in Tab. 3, CRISP outperforms SQRECON and Shap-E in terms of mean and median pose error $e^{L_1}_{pose}$, whereas SQRECON achieves better $e^{L_1}_{shape}$. With self-training, CRISP-ST outperforms SQRECON on all metrics, improving mean $e^{L_1}_{shape}$ by 13% and $e^{L_1}_{pose}$ by 23%.

Method	$e^{L_1}_{shape} \downarrow$		$e^{L_1}_{pose} \downarrow$	
	Mean	Median	Mean	Median
Shap-E [17]	0.183	0.171	–	–
SQRECON [31]	0.145	0.134	0.304	0.258
CRISP	0.163	0.159	0.292	0.211
CRISP-ST	**0.141**	**0.125**	**0.224**	**0.168**

Table 3. Evaluation of CRISP and CRISP-ST on the SPE3R dataset.

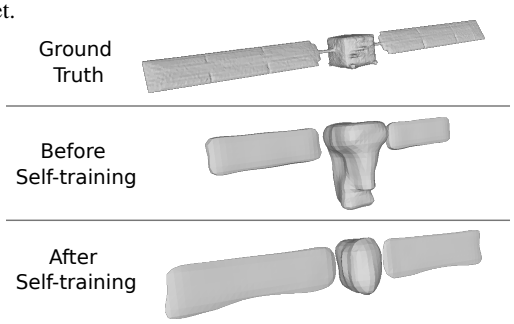


Figure 6. Qualitative example of self-training improving shape reconstruction.

Methods	e_{shape} (mm) $\downarrow$				
	Bottle	Camera	Laptop	Mug	Avg
SPD [44]	3.44	8.89	2.91	1.02	3.17
SGPA [7]	2.93	5.51	1.62	1.12	2.44
CASS [6]	0.75	0.77	3.73	0.32	1.06
RePoNet [53]	1.51	8.79	1.01	0.94	2.37
CRISP	**0.55**	**0.37**	**0.51**	**0.16**	**0.35**

Table 4. Shape reconstruction results on the REAL275 dataset. Tabulates average Chamfer distance (mm) for four categories and the average distance for all categories (full table in Appendix E).

Fig. 6 shows a qualitative example of self-training improving shape reconstruction performance. This satellite, smart-1, is unseen during training. Initially CRISP recalls the wrong shape, similar to the integral satellite in the train set (middle of Fig. 6). After self-training, the model is able to reconstruct a model (bottom of Fig. 6) much closer to the ground truth model (top of Fig. 6).

6.3. NOCS Dataset

Setup. We evaluate CRISP against baselines on the NOCS dataset [47]. The NOCS dataset is divided into two subsets: CAMERA, a synthetic dataset with 300K rendered images of ShapeNet objects, and REAL275 with 8K RGB-D frames of 18 different real-world scenes. We train CRISP on the train sets of CAMERA and REAL275 for 50 epochs, using an Adam optimizer with a learning rate of 10^{-4} and a cosine annealing scheduler with restarts per 4K iterations. We use the same segmentation masks as [7, 24], which are obtained from an off-the-shelf Mask R-CNN model [12]. For evaluation, we report results on the REAL275, which is the

	Methods	mAP ↑			
		IoU_{50}	IoU_{75}	5° 5 cm	10° 5 cm
Category-level	NOCS [47]	78.0	30.1	10.0	25.2
	Metric Scale [21]	68.1	–	5.3	24.7
	SPD [44]	77.3	53.2	21.4	54.1
	CASS [6]	77.7	–	23.5	58.0
	SGPA [7]	80.1	61.9	39.6	**70.7**
	RePoNet [53]	81.1	–	**40.4**	68.8
	SSC-6D [34]	73.0	–	19.6	54.5
Category-agnostic	DualPoseNet [24]	76.1	55.2	**31.3**	60.4
	FSD [27]	77.4	61.9	28.1	61.5
	CRISP	**83.5**	**70.5**	22.5	62.8

Table 5. Pose estimation results on the REAL275 dataset. We show here the average mAP at different thresholds for 3D IoU (IoU_{50} and IoU_{75}) and rotation and translation errors at 5° 5 cm and 10° 5 cm thresholds (full table in Appendix E).

Ablations	ADD-S (AUC) ↑ 2 cm	e_{shape} (AUC) ↑ 5 cm
No Projection Π (Line 3, BCD)	0.27	0.02
Single-View	0.37	0.29
No Corrector	0.30	0.35
Direct Self-training Loss	0.29	0.38
Proposed	**0.42**	**0.39**

Table 6. Ablation studies on test-time adaptation for CRISP

test set of REAL275 containing 2.75K images. We report e_{shape}, the average Chamfer distance between the reconstructed shape and the ground truth CAD model, and mean average precision (mAP) at different thresholds, following the metrics proposed by [47].

Results. Tab. 4 shows the shape reconstruction error for various baselines. CRISP outperforms the baselines in all categories, achieving a 67% improvement over the next best baseline, CASS. In the challenging object category of camera, CRISP achieves a 95% reduction comparing to SPD [44], and a 52% reduction comparing to CASS. Tab. 5 reports the pose estimation errors of methods tested. We separate out category-level methods from category-agnostic ones. The former either train one model per category, use explicit shape priors, or hard-code the number of categories in the network architecture. The latter train one model for all categories and do not hard code the number of categories in the architecture, which are more scalable. CRISP achieves the best performance in terms of mAP for IoU_{50} and IoU_{75}, while ranked third in terms of mAP at 10° 5 cm threshold. We note that CRISP's rotation error is higher than category-level baselines such as RePoNet [53] and SGPA [7].

6.4. Ablation Studies

We ablate a few design decisions to see their effects on self-training performance (Tab. 6). Without the projection in Line 3, in the BCD Algorithm 1, e_{shape} drops significantly (Row 1, Tab. 6). Restricting the corrector to single-view (Row 2, Tab. 6) and self-train without the corrector by directly certifying and self-training CRISP outputs (Row 3, Tab. 6) adversely affects both shape and pose estimation performance. Lastly, self-training without any pseudo-labels, but with a direct self-supervision loss similar to [34] achieves similar e_{shape} but worse ADD-S (AUC) metric (Row 4, Tab. 6). Tab. 7 shows the runtime of CRISP and correctors. LSQ (Alg. 2) achieves significantly faster runtime than BCD (Alg. 1).

Component	Runtime (ms) ↓ Mean	Median	SD
CRISP	125	127	161
LSQ	251	214	142
BCD (25)	3824	3483	1667
BCD (50)	7630	6942	3353

Table 7. Runtime comparisons of CRISP, LSQ (Alg. 2) and BCD (Alg. 1) with 25 and 50 maximum iterations.

7. Limitations and Future Work

We currently assume ground-truth object data association during test-time. A future direction is to investigate how tracking errors affect self-training and how to mitigate them. Another limitation is the use of latent shape codes for projection in the corrector, which scales linearly with the number of objects learned. One potential solution is to use dimensionality reduction techniques such as PCA. This will open up opportunities to self-train CRISP on larger datasets, potentially enabling it as a foundation model for object pose and shape estimation that can perform test-time adaptation.

8. Conclusion

We introduce CRISP, a category-agnostic object pose and shape estimation pipeline, and a self-training method CRISP-ST to bridge potential domain gaps. CRISP is fast at inference, amenable for multi-view pose and shape estimation and can be incorporated into real-time robotic perception pipeline for downstream tasks such as manipulation and object-SLAM. Experiments on the YCBV, SPE3R, and NOCS datasets show that CRISP is competitive in pose estimation and outperforms baselines in shape reconstruction.

Acknowledgements This work was partially funded by Army Research Laboratory Distributed and Collaborative Intelligent Systems and Technology Collaborative Research Alliance under Grant W911NF-17-2-0181, in part by the Office of Naval Research RAIDER project under Grant N00014-18-1-2828 and the NSF CAREER award "Certifiable Perception for Autonomous Cyber-Physical Systems,".

References

[1] K. Arun, T. Huang, and S. Blostein. Least-squares fitting of two 3-D point sets. *IEEE Trans. Pattern Anal. Machine Intell.*, 9(5):698–700, 1987. 3

[2] G. Brazil, A. Kumar, J. Straub, N. Ravi, J. Johnson, and G. Gkioxari. Omni3d: A large benchmark and model for 3d object detection in the wild. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 13154–13164, 2023. 1

[3] D. Cai, J. Heikkilä, and E. Rahtu. Gs-pose: Cascaded framework for generalizable segmentation-based 6d object pose estimation. In *Intl. Conf. on 3D Vision*, 2025. 2, 6

[4] E. R. Chan, M. Monteiro, P. Kellnhofer, J. Wu, and G. Wetzstein. pi-gan: Periodic implicit generative adversarial networks for 3d-aware image synthesis. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 5799–5809, 2021. 3

[5] B. Chen, J. Cao, A. Parra, and T.-J. Chin. Satellite pose estimation with deep landmark regression and nonlinear pose refinement. In *2019 IEEE/CVF International Conference on Computer Vision Workshop (ICCVW)*, pages 2816–2824. IEEE, 2019. 1

[6] D. Chen, J. Li, Z. Wang, and K. Xu. Learning canonical shape space for category-level 6d object pose and size estimation. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 11973–11982, 2020. 7, 8, 14

[7] K. Chen and Q. Dou. Sgpa: Structure-guided prior adaptation for category-level 6d object pose estimation. In *Intl. Conf. on Computer Vision (ICCV)*, pages 2773–2782, 2021. 7, 8, 14

[8] J. Deng, J. Lu, and T. Zhang. Unsupervised template-assisted point cloud shape correspondence network. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 5250–5259, 2024. 2

[9] M. Denninger, M. Sundermeyer, D. Winkelbauer, D. Olefir, T. Hodan, Y. Zidan, M. Elbadrawy, M. Knauer, H. Katam, and A. Lodhi. Blenderproc: Reducing the reality gap with photorealistic rendering. In *16th Robotics: Science and Systems, RSS 2020, Workshops*, 2020. 6, 12

[10] S. Diamond and S. Boyd. CVXPY: A Python-embedded modeling language for convex optimization. *Journal of Machine Learning Research*, 17(83):1–5, 2016. 5

[11] M. Hägele, K. Nilsson, J. N. Pires, and R. Bischoff. Industrial robotics. *Springer handbook of robotics*, pages 1385–1422, 2016. 1

[12] K. He, G. Gkioxari, P. Dollár, and R. Girshick. Mask R-CNN. In *Intl. Conf. on Computer Vision (ICCV)*, pages 2980–2988, 2017. 7

[13] X. He, J. Sun, Y. Wang, D. Huang, H. Bao, and X. Zhou. Onepose++: Keypoint-free one-shot object pose estimation without CAD models. In *Advances in Neural Information Processing Systems (NIPS)*, 2022. 1, 2

[14] T. Hodaň, M. Sundermeyer, B. Drost, Y. Labbé, E. Brachmann, F. Michel, C. Rother, and J. Matas. BOP challenge 2020 on 6D object localization. *European Conference on Computer Vision Workshops (ECCVW)*, 2020. 6

[15] T. Hodan, M. Sundermeyer, Y. Labbe, V. N. Nguyen, G. Wang, E. Brachmann, B. Drost, V. Lepetit, C. Rother, and J. Matas. Bop challenge 2023 on detection segmentation and pose estimation of seen and unseen rigid objects. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 5610–5619, 2024. 2

[16] M. Z. Irshad, S. Zakharov, R. Ambrus, T. Kollar, Z. Kira, and A. Gaidon. Shapo: Implicit representations for multi-object shape, appearance, and pose optimization. In *European Conf. on Computer Vision (ECCV)*, pages 275–292. Springer, 2022. 2

[17] H. Jun and A. Nichol. Shap-e: Generating conditional 3d implicit functions. *arXiv preprint arXiv:2305.02463*, 2023. 1, 6, 7

[18] H. Jun and A. Nichol. Shap-e: Generating conditional 3d implicit functions. *arXiv preprint arXiv:2305.02463*, 2023. 2

[19] Y. Labbe, J. Carpentier, M. Aubry, and J. Sivic. CosyPose: Consistent multi-view multi-object 6D pose estimation. In *European Conf. on Computer Vision (ECCV)*, 2020. 1, 2, 6

[20] Y. Labbé, L. Manuelli, A. Mousavian, S. Tyree, S. Birchfield, J. Tremblay, J. Carpentier, M. Aubry, D. Fox, and J. Sivic. Megapose: 6d pose estimation of novel objects via render & compare. In *Conference on Robot Learning (CoRL)*, 2022. 1, 2

[21] T. Lee, B.-U. Lee, M. Kim, and I. S. Kweon. Category-level metric scale object shape and pose estimation. *IEEE Robotics and Automation Letters*, 6(4):8575–8582, 2021. 8, 14

[22] T. Lee, J. Tremblay, V. Blukis, B. Wen, B.-U. Lee, I. Shin, S. Birchfield, I. S. Kweon, and K.-J. Yoon. Tta-cope: Test-time adaptation for category-level object pose estimation. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 21285–21295, 2023. 2

[23] Y. Li, G. Wang, X. Ji, Y. Xiang, and D. Fox. DeepIM: Deep Iterative Matching for 6D Pose Estimation. In *European Conf. on Computer Vision (ECCV)*, pages 683–698, 2018. 2

[24] J. Lin, Z. Wei, Z. Li, S. Xu, K. Jia, and Y. Li. Dualposenet: Category-level 6d object pose and size estimation using dual pose network with refined learning of pose consistency. In *Intl. Conf. on Computer Vision (ICCV)*, pages 3560–3569, 2021. 7, 8, 14

[25] M. Liu, C. Xu, H. Jin, L. Chen, M. Varma T, Z. Xu, and H. Su. One-2-3-45: Any single image to 3d mesh in 45 seconds without per-shape optimization. *Advances in Neural Information Processing Systems (NeurIPS)*, 36, 2024. 1, 2

[26] X. Liu, R. Zhang, C. Zhang, B. Fu, J. Tang, X. Liang, J. Tang, X. Cheng, Y. Zhang, G. Wang, and X. Ji. Gdrnpp. https://github.com/shanice-l/gdrnpp_bop2022, 2022. 6

[27] M. Lunayach, S. Zakharov, D. Chen, R. Ambrus, Z. Kira, and M. Z. Irshad. Fsd: Fast self-supervised single rgb-d to categorical 3d objects. In *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, pages 14630–14637. IEEE, 2024. 1, 2, 8, 14

[28] A. Mousavian, D. Anguelov, J. Flynn, and J. Kosecka. 3d bounding box estimation using deep learning and geometry. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 7074–7082, 2017. 1

[29] M. Oquab, T. Darcet, T. Moutakanni, H. Vo, M. Szafraniec, V. Khalidov, P. Fernandez, D. Haziza, F. Massa, A. El-Nouby, et al. Dinov2: Learning robust visual features without supervision. *arXiv preprint arXiv:2304.07193*, 2023. 3, 6, 11

[30] J. Park, P. Florence, J. Straub, R. Newcombe, and S. Lovegrove. DeepSDF: Learning continuous signed distance functions for shape representation. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2019. 2

[31] T. H. Park and S. D'Amico. Rapid abstraction of spacecraft 3d structure from single 2d image. In *AIAA SCITECH 2024 Forum*, page 2768, 2024. 1, 6, 7

[32] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32, 2019. 6

[33] G. Pavlakos, X. Zhou, A. Chan, K. Derpanis, and K. Daniilidis. 6-dof object pose from semantic keypoints. In *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, 2017. 2

[34] W. Peng, J. Yan, H. Wen, and Y. Sun. Self-supervised category-level 6d object pose estimation with deep implicit shape representation. In *Nat. Conf. on Artificial Intelligence (AAAI)*, pages 2082–2090, 2022. 2, 8, 14

[35] E. Perez, F. Strub, H. de Vries, V. Dumoulin, and A. Courville. FiLM: Visual Reasoning with a General Conditioning Layer. *Nat. Conf. on Artificial Intelligence (AAAI)*, 32(1), 2018. 2, 3, 11

[36] C. R. Qi, H. Su, K. Mo, and L. J. Guibas. Pointnet: Deep learning on point sets for 3D classification and segmentation. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 652–660, 2017. 2

[37] R. Ranftl, A. Bochkovskiy, and V. Koltun. Vision transformers for dense prediction. In *Intl. Conf. on Computer Vision (ICCV)*, pages 12179–12188, 2021. 2, 3, 11

[38] L. Sanneman, C. Fourie, J. A. Shah, et al. The state of industrial robotics: Emerging technologies, challenges, and key research directions. *Foundations and Trends® in Robotics*, 8 (3):225–306, 2021. 1

[39] S. Shalev-Shwartz, Y. Singer, K. P. Bennett, and E. Parrado-Hernández. Efficient learning of label ranking by soft projections onto polyhedra. *J. of Machine Learning Research*, 7(7), 2006. 4

[40] J. Shi, R. Talak, D. Maggio, and L. Carlone. A correct-and-certify approach to self-supervise object pose estimators via ensemble self-training. In *Robotics: Science and Systems (RSS)*, 2023. 2, 5

[41] V. Sitzmann, J. Martel, A. Bergman, D. Lindell, and G. Wetzstein. Implicit neural representations with periodic activation functions. *Advances in Neural Information Processing Systems (NeurIPS)*, 33:7462–7473, 2020. 3, 4, 11

[42] J. Sun, Z. Wang, S. Zhang, X. He, H. Zhao, G. Zhang, and X. Zhou. Onepose: One-shot object pose estimation without cad models. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 6825–6834, 2022. 1, 2

[43] R. Talak, L. Peng, and L. Carlone. Certifiable 3D object pose estimation: Foundations, learning models, and self-training. *IEEE Trans. Robotics*, 39(4):2805–2824, 2023. 2, 5

[44] M. Tian, M. H. Ang, and G. H. Lee. Shape prior deformation for categorical 6d object pose and size estimation. In *European Conf. on Computer Vision (ECCV)*, pages 530–546. Springer, 2020. 1, 2, 7, 8, 14

[45] G. Wang, F. Manhardt, J. Shao, X. Ji, N. Navab, and F. Tombari. Self6D: Self-supervised monocular 6D object pose estimation. In *European Conf. on Computer Vision (ECCV)*, pages 108–125, 2020. 2

[46] G. Wang, F. Manhardt, F. Tombari, and X. Ji. GDR-Net: Geometry-Guided Direct Regression Network for Monocular 6D Object Pose Estimation. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 16611–16621, 2021. 2, 6

[47] H. Wang, S. Sridhar, J. Huang, J. Valentin, S. Song, and L. Guibas. Normalized object coordinate space for category-level 6d object pose and size estimation. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 2642–2651, 2019. 1, 2, 3, 6, 7, 8, 11, 14, 16

[48] Y. Wang and J. M. Solomon. PRNet: Self-Supervised Learning for Partial-to-Partial Registration. In *Advances in Neural Information Processing Systems (NIPS)*, pages 8812–8824, 2019. 2

[49] B. Wen, J. Tremblay, V. Blukis, S. Tyree, T. Müller, A. Evans, D. Fox, J. Kautz, and S. Birchfield. Bundlesdf: Neural 6-dof tracking and 3d reconstruction of unknown objects. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 606–617, 2023. 6

[50] B. Wen, W. Yang, J. Kautz, and S. Birchfield. Foundationpose: Unified 6d pose estimation and tracking of novel objects. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 17868–17879, 2024. 1, 2, 6

[51] Y. Xiang, T. Schmidt, V. Narayanan, and D. Fox. PoseCNN: A convolutional neural network for 6D object pose estimation in cluttered scenes. In *Robotics: Science and Systems (RSS)*, 2018. 1, 2, 6

[52] Z. Xiao and C. G. Snoek. Beyond model adaptation at test time: A survey. *arXiv preprint arXiv:2411.03687*, 2024. 2

[53] Y. Ze and X. Wang. Category-level 6d object pose estimation in the wild: A semi-supervised learning approach and a new dataset. *Advances in Neural Information Processing Systems (NeurIPS)*, 35:27469–27483, 2022. 1, 2, 7, 8, 14

[54] Y. Zhao, T. Kwon, P. Streli, M. Pollefeys, and C. Holz. EgoPressure: A dataset for hand pressure and pose estimation in egocentric vision. In *ArXiv Preprint: 2409.02224*, 2024. 1