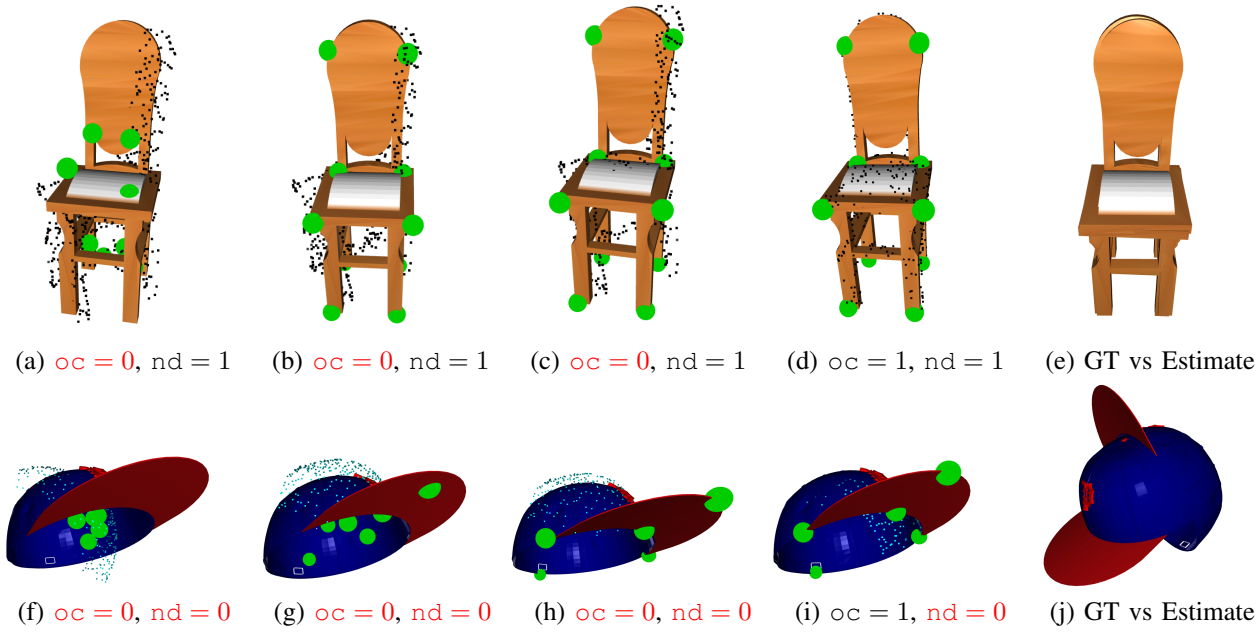
Certifiable 3D Object Pose Estimation: Foundations, Learning Models, and Self-Training

Rajat Talak, Lisa Peng, and Luca Carlone



(a) $\texttt{oc} = 0$, $\texttt{nd} = 1$ (b) $\texttt{oc} = 0$, $\texttt{nd} = 1$ (c) $\texttt{oc} = 0$, $\texttt{nd} = 1$ (d) $\texttt{oc} = 1$, $\texttt{nd} = 1$ (e) GT vs Estimate

(f) $\texttt{oc} = 0$, $\texttt{nd} = 0$ (g) $\texttt{oc} = 0$, $\texttt{nd} = 0$ (h) $\texttt{oc} = 0$, $\texttt{nd} = 0$ (i) $\texttt{oc} = 1$, $\texttt{nd} = 0$ (j) GT vs Estimate

Fig. 1: We propose C-3PO, a certifiable and self-supervised pose estimation model, that estimates the pose of an object from partial point cloud observations and computes a certificate of correctness for the resulting outputs. C-3PO detects semantic keypoints, corrects them using a differentiable optimization layer, and computes two certificates of observable correctness (oc) and non-degeneracy (nd). When $\texttt{oc} = 1$ and $\texttt{nd} = 1$, the C-3PO output is guaranteed to be correct. The first row shows C-3PO processing an input point cloud of a *chair* (each subfigure also shows the keypoints and the object CAD model arranged according to the pose computed at different iterations of the C-3PO corrector): (a) detected keypoints, (b)-(c) C-3PO corrector result at iterations 2, 10, and 300, and (e) ground-truth vs. final C-3PO estimate. The second row shows C-3PO processing the point cloud of a *cap*: (f) detected keypoints, (g)-(i) C-3PO corrector result at iterations 5, 10, 100, and (j) ground-truth vs. final estimate. The non-degeneracy certificate (nd) detects that in this case the input is degenerate, *i.e.,* while we can compute an estimate that fits the input point cloud, the latter does not contain sufficient information to uniquely estimate the object pose.

***Abstract*—We consider a *certifiable* object pose estimation problem, where —given a partial point cloud of an object— the goal is to not only estimate the object pose, but also to provide a certificate of correctness for the resulting estimate. Our first contribution is a general theory of certification for end-to-end perception models. In particular, we introduce the notion of ζ-*correctness*, which bounds the distance between an estimate and the ground truth. We then show that ζ-correctness can be assessed by implementing two certificates: (i) a certificate of *observable correctness*, that asserts if the model output is consistent with the input data and prior information, (ii) a certificate of *non-degeneracy*, that asserts whether the input data is sufficient to compute a unique estimate. Our second contribution is to apply this theory and design a new learning-based certifiable pose estimator. In particular, we propose C-3PO, a semantic-keypoint-based pose estimation model, augmented with the two certificates, to solve the certifiable pose estimation problem. C-3PO also includes a *keypoint corrector*, implemented as a differentiable optimization layer, that can correct large detection errors (*e.g.,* due to the sim-to-real gap). Our third contribution is a novel self-supervised training approach that uses our certificate of observable correctness to provide the supervisory signal to C-3PO during training. In it, the model trains only on the observably correct input-output pairs produced in each batch and at each iteration. As training progresses, we see that the observably correct input-output pairs grow, eventually reaching near 100% in many cases. We conduct extensive experiments to evaluate the performance of the corrector, the certification, and the proposed self-supervised training using the ShapeNet and YCB datasets. The experiments show that (i) standard semantic-keypoint-based methods (which constitute the backbone of C-3PO) outperform more recent alternatives in challenging problem instances, (ii) C-3PO further improves performance and significantly outperforms all the baselines, (iii) C-3PO's certificates are able to discern correct pose estimates. We release the implementation and an interactive visualization of all the results presented in this paper at: https://github.com/MIT-SPARK/C-3PO and https://github.com/MIT-SPARK/pose-baselines.**

The authors are with the Laboratory of Information and Decision Systems (LIDS), Massachusetts Institute of Technology, Cambridge, MA 02139, USA. Corresponding author: Rajat Talak (email: talak@mit.edu)

This work was partially funded by ARL DCIST CRA W911NF-17-2-0181, ONR RAIDER N00014-18-1-2828, and NSF CAREER award "Certifiable Perception for Autonomous Cyber-Physical Systems".

I. Introduction

Object pose estimation is a crucial prerequisite for robots to understand and interact with their surroundings. In this paper, we consider the problem of 3D object pose estimation using depth or partial point cloud data, which arises when the robot is equipped with depth sensors (*e.g.,* lidar, RBG-D cameras) or has to postprocess a 3D point cloud (*e.g.,* resulting from a 3D reconstruction pipeline). Local-feature-based registration methods can be re-purposed to solve the object pose estimation problem, see, *e.g.,* [1]. Recent work, however, argues that either a neural model that mimics the ICP algorithm [2], [3], or approaches based on learned global representations [4], [5] yield better performance. A known drawback of these methods is that they are not expressive enough to yield high pose estimation accuracy when the object point cloud suffers from occlusions. Object pose estimation from partial or depth point clouds, therefore, remains an open problem [3]–[5].

When using a learning-based approach for pose estimation, another major challenge is to guarantee —or at least assess— the trustworthiness of the predictions of neural-network-based models. Robot perception requires models that can be relied upon for the robot to act safely in the world. Perception failures can lead to catastrophic consequences including loss of life [6]. Recent progress has been made towards certifying geometric optimization problems [1], [7], [8], which form the back-end of common perception pipelines. However, these approaches only produce certificates of optimality for the back-end optimization, but might still fail without notice if the perception front-end (*e.g.,* the feature detector) produces largely incorrect results. Here we argue that we need to go beyond certifiable optimization and need to certify the *entire* perception pipeline, *i.e.,* we need to produce certificates that can determine if the output produced by an end-to-end (*e.g.,* opaque box) perception pipeline is correct or not. To date, there is no theory developed for this purpose for general perception tasks, let alone for the case of object pose estimation. The current literature only offers statistical guarantees for specific problems [9] or is restricted to data near the training regime [10]–[14].

Finally, most of the recent progress on pose estimation has focused on supervised learning and relies on large annotated datasets [15]–[19]. However, obtaining human-labeled data is not only costly, but also robot-dependent: a change in placement of the sensor/camera (*e.g.,* from the top of a self-driving car to the head of a quadruped robot) is likely to distort the predictions produced by the trained model. Though photo-realistic simulators provide a partial answer to this problem [20]–[22], a sim-to-real gap is likely going to remain and needs to be tackled. Here, we argue that a scalable pose estimation method must rely on self-supervision rather than human-labeled data. Very few works have tackled pose estimation in a self-supervised manner [5], [23]–[28].

A. Contributions

We address the twin challenges of certifiability and self-supervised training by noting that certification can provide a supervision signal for self-training (*i.e.,* if we can identify correct predictions, we can learn from them). More specifically, we establish the *foundations* of certifiable object pose estimation using learning-based models, by defining a notion of correctness and developing certificates that can assert correctness in practice. We then propose a new *learning-based model*, named C-3PO (Certifiable 3D POse), for certifiable pose estimation from depth or point cloud data. Finally, we show how to use the proposed certificates to enable *self-supervised training* of C-3PO. We unpack each contribution below.

(I) Theory of Certifiable Models and Certifiable Object Pose Estimation. We start by defining the notion of ζ-*correctness* that discerns whether an estimate is correct (*i.e.,* matches the ground truth object) or not (Definition 1). We then use this notion to introduce the problem of *certifiable object perception* (Problem 1), where —given a partial point cloud of an object— we aim to estimate the pose of the object in the point cloud and provide a certificate that guarantees that the pose estimate is ζ-correct (Section III).

After stating the problem, we develop a general theory of certifying learning-based models (Section IV). We show that by implementing two certificates, namely, a certificate of *observable correctness*, and a certificate of *non-degeneracy*, we can infer ζ-correctness (Theorem 4). The observable correctness certificate determines whether an output produced by a model is consistent with the sensor data and prior knowledge (*e.g.,* knowledge of the object shape). Non-degeneracy, on the other hand, ensures that for the provided input there exists a unique, correct solution. A case of degeneracy arises often, for instance, in object pose estimation when the partial point cloud is so occluded that there is more than one correct pose fitting the data; see Fig. 3. Therefore, a *certifiable model* is a learning-based model that implements these two certificates —of observable correctness and non-degeneracy— so that for every output produced by the model, its ζ-correctness may be checked.

(II) C-3PO: a Certifiable 3D POse Estimation Model. We propose C-3PO (Certifiable 3D POse) to solve the certifiable object pose estimation problem (Section V). C-3PO first detects semantic keypoints from the input, partial point cloud using a trainable regression model. To correct for potential keypoint-detection errors (*e.g.,* induced by a sim-to-real gap), C-3PO adds a keypoint *corrector* module that corrects some of the keypoint-detection errors. The corrector module is implemented as a differentiable optimization module [29], [30] (Section V-C): it takes in the detected keypoints and produces a correction, by solving a non-linear, non-convex optimization problem. Although the corrector optimization problem is hard to analyze, we show that it can be differentiated through very easily. We provide an exact analytical expression of the gradient of the output (the correction), given the input (detected keypoints) to the corrector (Theorem 9), which we use to implement back-propagation. We also implement a *batch gradient descent* that solves a batch of corrector optimization problems in parallel on GPU, thereby speeding up the forward pass. Experimental analysis shows that the corrector is able to correct high-variance perturbations to the semantic keypoints (Section VII-B).

Finally, using our theory of certifiable models, we derive two certificates, namely, a certificate of observable correctness and a certificate of non-degeneracy for C-3PO (Section V-D). We prove that this enables C-3PO to check ζ-correctness in

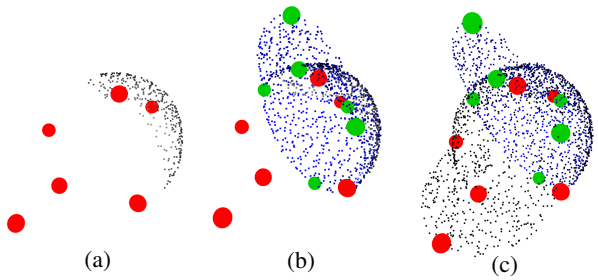


Fig. 3: Example of degenerate input for pose estimation. (a) Input: partial point cloud of an object (gray), namely the "cap" in ShapeNet [31]; (b) Estimated object pose shown as a posed object CAD model (blue); (c) Estimated (blue) and ground-truth (gray) pose overlaid as two posed point clouds. Ground truth semantic keypoints are in red, detected keypoints in green.

the forward pass (Theorem 16). We also empirically observe that the implemented certificates are able to identify correct predictions in practice (Sections VII-C and VII-D).

(III) Certificate-based Self-supervised Training. C-3PO allows addressing the twin problems of certification and self-supervision by noting the following: if at test time we can use the proposed certificates to distinguish ζ-correct outputs, we can use the latter for further training. This forms the basis of our new self-supervised training, which proceeds by training only on the observably correct instances (*i.e.,* input-output pairs), in each batch and in each training iteration. As the training progresses, the number of observably correct instances increases, eventually resulting in a fully self-trained model.

We test the self-supervised training and certification on depth-point-cloud dataset generated using ShapeNet objects [31] and on the depth point clouds extracted from RGB-D images in the YCB dataset [32] (Sections VII-C and VII-D). In both cases, we observe that while the simulation-trained C-3PO performs poorly (indicating a large sim-to-real gap), our self-supervised training is able to successfully train C-3PO with self-supervision and significantly outperforms existing baselines. As a further experiment, we show that the proposed self-supervised training can be even used to self-train C-3PO when the object category labels are not available, for instance C-3PO is able to learn to detect a "chair" from a large unlabeled dataset containing chairs, tables, cars, airplanes, etc. (Section VII-E).

An interactive visualization of all the results presented in this paper is made available at: https://github.com/MIT-SPARK/C-3PO and https://github.com/MIT-SPARK/pose-baselines.

II. Related Work

Here we provide a non-exhaustive review about object pose estimation, including recent image-based estimation methods.

Object Pose Estimation from RGB and RGB-D Inputs. Recent progress in object pose estimation has been fueled by the availability of pose-annotated datasets [33]–[36]. State-of-the-art methods have evolved from correspondence-based methods (which extract correspondences between the sensor data and the object CAD model, and use them to register the model to the data) and template-based methods (which attempt to match the sensor data to templates consisting of the CAD model of the object rendered at various poses) to direct regression, augmented with pose refinement. Xiang *et al.* [17] regress the object pose using a convolutional neural network (CNN) backbone. Wang *et al.* [37] propose GDR-Net, which instead extracts a 6-dimensional rotation representation, thus, avoiding discontinuity issues [38]. Nguyen *et al.* [39] revive template-based methods by showing their generalization power. Li *et al.* [40] propose DeepIM, which uses a pre-trained optical flow detection model to guide and train object pose refinement. Labbe *et al.* [41] propose CosyPose, which showed state-of-the-art performance on the BOP'20 pose estimation challenge [34], relying on the pose regression network in [17], followed by the pose refinement proposed in [40]. The state-of-the-art performance in the BOP'22 challenge was also attained by a model that uses a pose regression network [37], [42].

We note two gaps in the pose estimation literature. First, most of the approaches above are RGB or RGB-D based, and investigating object pose estimation from only depth or partial point clouds has received little attention. Second, these methods rely on large pose-annotated datasets, with only few attempts to develop self-supervised approaches [23], [24].

We next review different approaches for *point-cloud-only* pose estimation, including approaches based on local feature detection/description, global features, and semantic keypoints.

Local Feature Detectors and Descriptors. Local-feature-based methods extract local features (potentially with descriptors), determine point-to-point correspondences, and solve a robust registration problem, which estimates the object pose while filtering out incorrect correspondences. Significant progress has been made in recent years towards (i) better features to extract point-to-point correspondences [43]–[48], (ii) robust registration algorithms [1], [7], [49], as well as (iii) end-to-end registration pipelines that simultaneously learn correspondences and obtain the relative poses [50]–[56]. Modern robust registration methods have been shown to compute accurate estimates even in the face of 99% random outliers [1]. Several end-to-end learning-based registration approaches have been proposed, but only a few can be trained in a self-supervised manner. A weakly-supervised approach, using a triplet loss, is developed in [44]. Self-supervised partial-to-partial scene registration is considered in [57], [58]. In particular, Banani *et al.* [58] use the fact that two successive frames inherit some geometric and photometric consistency, and leverage differentiable rendering to implement the training loss. A teacher-student-verifier framework for joint feature learning and registration is considered in [28].

Global Features. Methods that extract and use global features have been proposed for point cloud alignment and object pose estimation. Sarode *et al.* [51] propose an architecture that uses PointNet encoding to extract global features. Huang *et al.* [59] extract global features from the point clouds and then estimate the pose by minimizing a feature-metric projection error. Li *et al.* [3] propose a similar approach, but uses Lucas-Kanade-like iterations for minimizing the feature-metric error. Yuan *et al.* [4] propose a probabilistic registration paradigm, which extracts pose-invariant point correspondences, as latent Gaussian mixture model. Zhu *et al.* [60] addresses point cloud registration by extracting $\mathrm{SO}(3)$-equivariant features. Sun *et al.* [61] use a capsule network encoder. Among these

approaches PointNetLK [3] and DeepGMR [4] show the best results for object pose estimation, albeit in different scenarios. Li *et al.* [5] tackle category-level object pose estimation from partial or complete object point clouds, by extracting an $\mathrm{SE}(3)$-invariant shape feature and a $\mathrm{SE}(3)$-equivariant pose feature.

Semantic Keypoints. Semantic keypoint detectors extract keypoints that correspond to specific points on the object to detect (*e.g.,* the wheels or headlights of a car), hence circumventing the needs to compute local descriptors. Semantic keypoint-based methods have been studied in the context of human pose estimation [62]–[64]. Pavlakos *et al.* [65], [66] use class-specific semantic keypoints, extracted from CNN features, to estimate object pose from an RGB image. Lin *et al.* [67] propose an end-to-end, differentiable pose estimation architecture using RGB images. Shi *et al.* [68] propose an active shape model using semantic keypoints, and solve the joint pose and shape estimation problem assuming detected semantic keypoints. Yang and Pavone [9] obtain performance guarantees on keypoint detections and pose estimates using conformal prediction. Zhou *et al.* [69] propose category-agnostic keypoint detection, and argue that using a fixed number of keypoints per-category can be limiting (*e.g.,* chair with many legs). Vecerik *et al.* [70] advocate semantic keypoints to be the right visual representation for object manipulation, and provide an efficient training approach to learn instance and category-level keypoints using a small number of annotated images.

Contrary to the works above, C-3PO does not rely on pose or keypoint annotations for training. Instead, it only assumes the availability of the object CAD model annotated with semantic keypoints. You *et al.* [71] provide a large-scale keypoint annotated dataset on ShapeNet CAD models. Suwajanakorn *et al.* [72] show that semantic keypoint annotations can be learned in a self-supervised manner, from just the object CAD models.

Self-Supervised Pose Estimation. Few works have tackled self-supervised pose estimation. In [23], [24], a pose estimation model is first trained on synthetic RGB-D data, and then refined further with self-supervised training on real, unannotated data; differentiable rendering provides the required supervision signal. Student-teacher iterative schemes are proposed in [27], [28] to bridge the domain gap. Another approach is to extract a pose-invariant feature, thereby canonizing only its shape [73], and using it for supervision. Zakharov *et al.* [25] utilize differentiable rendering of signed distance fields of objects, along with normalized object coordinate spaces [73], to learn 9D cuboids in a self-supervised manner. Li *et al.* [5] extract an $\mathrm{SE}(3)$-invariant feature, which works as a canonical object, and uses it to supervise training with a Chamfer loss. Sun *et al.* [61] tackles point cloud alignment in a self-supervised manner by extracting features using capsule network. Deng *et al.* [26] propose a way to self-supervise pose estimation by interacting with the objects in the environment; the model gets trained on the data collected autonomously by a manipulator.

Certification. Certifying correctness of the model output is crucial for safety-critical and high-integrity applications. Certifiably optimal algorithms return a solution and also provide a certificate of optimality [1], [7], [74] of the solution. Estimation contracts [**?**], [1], on the other hand, determine if the input is reasonable enough for the optimal solution to be indeed correct, *i.e.,* close to the ground-truth. Certifiably optimal algorithms have been devised for several geometric perception problems [1], [7], [8], where the duality gap serves as the certificate of optimality. These notions of certification can be applied to optimization problems, which form the back-end of any perception pipeline, but do not apply to the entire end-to-end pipeline or learning-based models. In this work, we extend the notion of certification to learning-based models.

Recent works have attempted to address the question of robustness and uncertainty-quantification of learning-based models. Neural network verification attempts to ensure that a learning-based model is robust to small deviations in the input space [12]–[14]. Conformal prediction, on the other hand, uses training data to safeguard a learning-based model [9]–[11], [75], [76]. Given a user-specified probability, it enables a trained model to predict uncertainty sets/intervals, instead of a point estimate. The predicted set is then provably correct, with the specified probability. Conformal prediction works under the assumption that the test set and the dataset used to derive thresholds for set-predictions have the same distribution. Providing probabilistic guarantees under sim-to-real gap or covariate shifts is an active research area [77], [78].

Differentiable Optimization. Learning-based models have traditionally relied on simple feedfoward functions (*e.g.,* linear, ReLU). Recent work has proposed to embed differentiable optimization as a layer in learning-based models [30], [79], [80]. A differentiable optimization layer is an optimization problem for which the gradient of the optimal solution, with respect to the input parameters, can be computed and back-propagated. Embedding an optimization problem as a differentiable optimization layer enables a learning-based model to explicitly take into account various geometric and physical constraints. Gould *et al.* [80] provide generic expressions to differentiate any non-linear optimization problem. Agrawal *et al.* [30] provide a way to differentiate through convex programs in standard form. Amos *et al.* [79] focus on quadratic optimization problems. Differentiating combinatorial optimization problems is considered in [81], [82], and a differentiable MAXSAT solver is proposed in [83]. Donti *et al.* [84] propose a way to differentiate a stochastic optimization problem. Teed *et al.* [85] implement differentiable bundle adjustment.

While developing generic techniques to implement derivatives of various classes of optimization problems is useful, there is much scope for efficiency if the problem structure can be exploited to implement simpler differentiation rules [79], [86], [87]. C-3PO implements a corrector module, which is a differentiable optimization layer. It solves a non-linear, non-convex optimization problem, but also exploits its specific structure efficiently to compute its derivative.

III. Problem Statement

Certifiable Pose Estimation. Let $\boldsymbol{B}$ be the CAD model of an object, represented as the set of all points (expressed in homogeneous coordinates) on the surface of the object. Let $\boldsymbol{X}^*$ be the corresponding *posed* model, *i.e.,* the CAD model transformed according to its ground-truth pose $\boldsymbol{T}^* \in \mathrm{SE}(3)$:

$$\boldsymbol{X}^* = \boldsymbol{T}^* \cdot \boldsymbol{B}. \tag{1}$$

Due to occlusions and sensor measurement noise, we only observe a partial and noisy point cloud $\boldsymbol{X}$. This can be written as a function of the ground-truth posed model $\boldsymbol{X}^*$:

$$\boldsymbol{X} = \Theta\left(\boldsymbol{X}^*\right) + \boldsymbol{n}_w, \tag{2}$$

where $\Theta\left(\boldsymbol{X}^*\right)$ denotes the sampling of n points on the object surface and deletion of occluded parts, and $\boldsymbol{n}_w$ is the measurement noise.

The goal of certifiable pose estimation is to estimate the pose of the object, given the partial point cloud $\boldsymbol{X}$, and also to provide a *certificate* on the correctness of the resulting pose estimate —in terms of its "closeness" to the ground-truth pose $\boldsymbol{T}^*$ (more precisely, we will assert closeness between the ground-truth posed model $\boldsymbol{X}^*$ and the estimated model). We define and use the following notion of correctness.

Definition 1 (ζ-correctness)**.** *We say that a pose estimate $\hat{\boldsymbol{T}}$, produced by a model, is ζ-correct if*

$$d_H(\hat{\boldsymbol{T}} \cdot \boldsymbol{B}, \boldsymbol{T}^* \cdot \boldsymbol{B}) \leq \zeta, \tag{3}$$

where $d_H(\cdot,\cdot)$ is the Hausdorff distance.

The Hausdorff distance between two (potentially non-finite) point sets $\boldsymbol{A}$ and $\boldsymbol{B}$ is defined as:

$$d_H(\boldsymbol{A}, \boldsymbol{B}) = \max\left\{\sup_{\boldsymbol{x}\in\boldsymbol{A}} D(\boldsymbol{x}, \boldsymbol{B}), \sup_{\boldsymbol{x}\in\boldsymbol{B}} D(\boldsymbol{x}, \boldsymbol{A})\right\}, \tag{4}$$

where $D(\boldsymbol{x}, \boldsymbol{A})$ denotes the minimum distance from point $\boldsymbol{x}$ to the set $\boldsymbol{A}$. Intuitively, ζ-correctness of the estimate $\hat{\boldsymbol{T}}$ ensures that every point on the surface of the posed model $\hat{\boldsymbol{T}} \cdot \boldsymbol{B}$ is at most ζ distance away from the surface of the ground-truth posed model $\boldsymbol{T}^* \cdot \boldsymbol{B}$. Unlike rotation and translation error, the metric $d_H(\hat{\boldsymbol{T}} \cdot \boldsymbol{B}, \boldsymbol{T}^* \cdot \boldsymbol{B})$ operates directly on the CAD model surfaces. This obviates the need to handle symmetric objects separately: for instance, while two poses corresponding to symmetries of an object would produce different rotation and translation errors, the posed model associated to both poses will be the same, leading to the same Hausdorff distance for both. We also use the fact that the Hausdorff distance is a valid distance metric over the space of all posed CAD models to derive and analyze the certificates in Section V-D. In Appendix A we discuss why we chose Hausdorff distance as opposed to other metrics like pose error or ADD-S metric [18] for our theoretical analysis.

The ζ-correctness of a pose estimate cannot be determined by directly computing (3), as we do not have access to the ground-truth pose in practice. A model solving the certifiable pose estimation problem should find a way to assert if (3) is satisfied, without a genie-access to the ground truth, leading to the following problem statement.

Problem 1 (Certifiable Object Pose Estimation)**.** *Propose a pose estimation model that, along with producing an estimate, can also provide a binary* $(0/1)$ certificate*, such that whenever a certificate is* 1*, the estimated pose is ζ-correct.*

Unannotated Data and Self-Supervision. It is easy to realize that certification is tightly coupled with self-supervision: if we are able to discern —at test time— correct outputs, we can then use them for further training. Therefore, in this paper we also study the twin problem of self-supervising pose estimation models. In particular, we investigate learning models that can self-train using an unannotated real-world dataset, which consists of partial point clouds of objects, segmented from a scene; no pose or any other annotation is assumed on the real-world dataset. We start by noting that annotated simulation data is broadly available and can be used to initialize weights of any learning-based pose estimation model. Therefore, we consider the practical case where we are given a sim-trained learning-based model, and our goal is to modify and further train the model on the unannotated real-world data.

Problem 2 (Self-supervised Pose Estimation)**.** *Propose a method to modify a sim-trained pose estimation model, and train it with self-supervision on unannotated real-world data.*

Note that the problem is particularly relevant when there is a significant sim-to-real gap, in which case the self-supervised training is used to bridge such gap.

IV. A Theory of Certifiable Models

Before delving into the details of our certifiable pose estimation model (Section V), we provide a general framework to develop certifiable models. We then tailor it to certifiable pose estimation in Section V-D. We start by considering a general problem setup, and discuss the certificates, their implementation, and the resulting certifiable models.

Setup. Let $\mathbb{X}$ and $\mathbb{Z}$ be the space of all inputs and outputs, for a problem set that we are solving. We assume the output space $\mathbb{Z}$ is a metric space with a distance metric $d_{\mathbb{Z}}(\cdot,\cdot)$. Moreover, we assume any input $\boldsymbol{X} \in \mathbb{X}$ is generated according to a given *generative model*, *i.e.,* $\boldsymbol{X} = \phi(\boldsymbol{Z}^*)$, for some unknown $\boldsymbol{Z}^* \in \mathbb{Z}$; hence, the goal is to estimate $\boldsymbol{Z}^*$ given $\boldsymbol{X}$. For instance, in our object pose estimation problem, $\boldsymbol{X}$ is the given depth or point cloud data, $\boldsymbol{Z}^*$ is the ground-truth posed model we want to estimate, and the generative model is the one in eq. (2). Note that since the output space is assumed to be a metric space, we can straightforwardly extend Definition 1 and say that an estimate $\boldsymbol{Z}$ is ζ-correct if $d_{\mathbb{Z}}(\boldsymbol{Z}, \boldsymbol{Z}^*) \leq \zeta$.

We denote a problem instantiated by an input $\boldsymbol{X} \in \mathbb{X}$ as $\mathcal{P}(\boldsymbol{X})$. A learning-based model $\mathcal{M}$, that is trained to solve problems $\mathcal{P}(\boldsymbol{X})$ (for $\boldsymbol{X} \in \mathbb{X}$), is nothing but a mapping $\mathcal{M} : \mathbb{X} \to \mathbb{Z}$. That is, given input $\boldsymbol{X}$, model $\mathcal{M}$ finds *a* (possibly incorrect) solution $\boldsymbol{Z} = \mathcal{M}(\boldsymbol{X})$ to the problem $\mathcal{P}(\boldsymbol{X})$.

The definition of the solution space $\mathcal{S}(\boldsymbol{X})$ below will play a pivotal role in the definition and analysis of our certificates.

Definition 2 (Solution Space)**.** *Given an input $\boldsymbol{X}$, the* solution space *is the set of all outputs $\boldsymbol{Z}$ that can generate the input $\boldsymbol{X}$ using the generative model $\phi(\cdot)$:*

$$\mathcal{S}(\boldsymbol{X}) \triangleq \{\boldsymbol{Z} \in \mathbb{Z} \mid \phi(\boldsymbol{Z}) = \boldsymbol{X}\}. \tag{5}$$

The intuition behind our certification approach is as follows: the ground truth $\boldsymbol{Z}^*$ is in the solution space since it generated the input data, hence if we can (i) produce an estimate in the solution space, and (ii) prove that the solution space is not large, we can conclude that the estimate must be close to the ground truth. Our *certificates* formalize these intuitions.

Certificates. We define two certificates to check (i) the *observable correctness* of the output (*i.e.,* whether the output $\boldsymbol{Z}$ is in the solution space $\mathcal{S}(\boldsymbol{X})$ or not) and (ii) the *non-degeneracy* (*i.e.,* whether $\mathcal{S}(\boldsymbol{X})$ is small, or not); then we prove that the two certificates imply correctness.

Definition 3 (Observable Correctness and Non-Degeneracy)**.** *For model $\mathcal{M}$ and input $\boldsymbol{X}$, the* certificate of observable correctness *is a Boolean condition defined as*

$$\texttt{ObsCorrect}[\mathcal{M}, \boldsymbol{X}] = \mathbb{I}\{\boldsymbol{Z} = \mathcal{M}(\boldsymbol{X}) \in \mathcal{S}(\boldsymbol{X})\}. \quad (6)$$

A certificate of non-degeneracy *is defined as*

$$\texttt{NonDegeneracy}[\mathcal{M}, \boldsymbol{X}] = \mathbb{I}\{\mathit{Diam}[\mathcal{S}(\boldsymbol{X})] < \delta\}, \quad (7)$$

where $\mathit{Diam}[\mathcal{S}(\boldsymbol{X})] = \max_{\boldsymbol{Z},\boldsymbol{Z}' \in \mathcal{S}(\boldsymbol{X})} d_{\mathbb{Z}}(\boldsymbol{Z}, \boldsymbol{Z}')$ denotes the diameter of the solution space $\mathcal{S}(\boldsymbol{X})$, and δ is a small constant.

We now show that the two certificates enable us to determine when a model produces an ζ-correct output.

Theorem 4. *For any input $\boldsymbol{X}$, the output produced by the model $\boldsymbol{Z} = \mathcal{M}(\boldsymbol{X})$ is ζ-certifiably correct,* i.e., *$d_{\mathbb{Z}}(\boldsymbol{Z}, \boldsymbol{Z}^*) < \zeta$, if the certificate of observable correctness (6) and non-degeneracy (7) are both 1, and $\delta \leq \zeta$ in (7).*

Implementing Certificates. Theorem 4 shows the importance of the certificates in (6)-(7). However, in general it may be hard to implement (6)-(7) as the solution space $\mathcal{S}(\boldsymbol{X})$ might be hard to characterize. We next prove that using an outer approximation of $\mathcal{S}(\boldsymbol{X})$ (*i.e.,* $\mathcal{S}(\boldsymbol{X}) \subseteq \overline{\mathcal{S}}(\boldsymbol{X})$), which is often easier to obtain in practice, retains the correctness guarantees.

Theorem 5. *Define the certificates $\texttt{oc}$ and $\texttt{nd}$ as:*

$$\texttt{oc}(\mathcal{M}, \boldsymbol{X}) = \mathbb{I}\left\{\boldsymbol{Z} = \mathcal{M}(\boldsymbol{X}) \in \overline{\mathcal{S}}(\boldsymbol{X})\right\}, \text{ and} \quad (8)$$

$$\texttt{nd}(\mathcal{M}, \boldsymbol{X}) = \mathbb{I}\left\{\mathit{Diam}[\overline{\mathcal{S}}(\boldsymbol{X})] < \delta\right\}, \quad (9)$$

where $\mathcal{S}(\boldsymbol{X}) \subseteq \overline{\mathcal{S}}(\boldsymbol{X})$. Then, the output $\boldsymbol{Z} = \mathcal{M}(\boldsymbol{X})$ is ζ-certifiably correct, i.e., *$d_{\mathbb{Z}}(\boldsymbol{Z}, \boldsymbol{Z}^*) < \zeta$, if $\texttt{oc}(\mathcal{M}, \boldsymbol{X}) = 1$ and $\texttt{nd}(\mathcal{M}, \boldsymbol{X}) = 1$, and $\delta \leq \zeta$ in (9).*

In Section V, we provide practical implementations of the certificates in Theorem 5 for the pose estimation problem.

Certifiable Model. We define a *certifiable model* to be a triplet $(\mathcal{M}, \texttt{oc}, \texttt{nd})$, where $\mathcal{M}$ is a learning-based model, and $\texttt{oc}$ and $\texttt{nd}$ are two certificates that (conservatively) approximate the certificate of observable correctness (6) and non-degeneracy (7), in order to establish ζ-correctness of every output produced. In other words, a certifiable model should always allow for a theorem, that is as follows:

Theorem 6 (Meta-Theorem for Certifiable Models)**.** *The output $\boldsymbol{Z} = \mathcal{M}(\boldsymbol{X})$ is ζ-certifiably correct,* i.e., *$d_{\mathbb{Z}}(\boldsymbol{Z}, \boldsymbol{Z}^*) < \zeta$, if $\texttt{oc}(\mathcal{M}, \boldsymbol{X}) = 1$ and $\texttt{nd}(\mathcal{M}, \boldsymbol{X}) = 1$.*

Remark 7 (Relation to Certifiable Algorithms)**.** *A parallel can be drawn between the two certificates in (6)-(7), and the* certificate of optimality *and the notion of* estimation contracts *in related work [?], [1], [7]. A certificate of optimality indicates whether the solver to optimal estimation problem returns the optimal solution or not. Estimation contracts, on the other hand, ensure that the input is "reasonable" enough for the optimal solution to be indeed correct,* i.e., *close to the ground-truth. While the former is analogous to our certificate of observable correctness (which indicates whether the output of a learning-based model is in the solution space or not), the latter is similar in spirit to the certificate of non-degeneracy. The difference is that our approach, unlike [?], [1], [7], aims at guaranteeing correctness of learning-based models, rather than optimization-based estimators.*

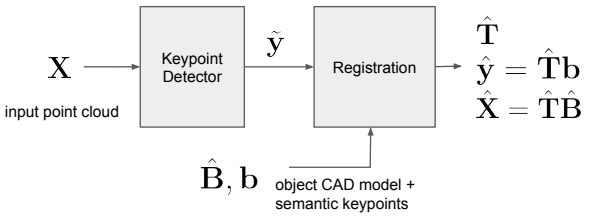


Fig. 4: KeyPO: Semantic keypoint-based pose estimation.

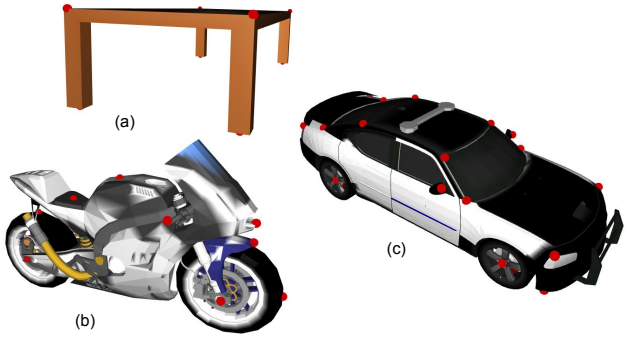


Fig. 5: Annotation of semantic keypoints (red) on CAD models of (a) table, (b) motorcycle, and (c) car in KeypointNet [71].

V. C-3PO: A Certifiable Pose Estimation Model

This section provides a certifiable pose estimation model, named C-3PO (Certifiable 3D POse), that includes a novel architecture (mixing learning-based and optimization-based modules), and a practical implementation of the certificates introduced in the previous section. Then, in Section VI we show that the use of the proposed certificates naturally enables C-3PO to self-train on unannotated data.

A. Preliminaries: Semantic-Keypoint-based Pose Estimation (KeyPO)

We start by briefly reviewing a standard pose estimation method based on semantic keypoints (Fig. 4), which constitutes the backbone of C-3PO. Semantic keypoints $\boldsymbol{b}[i], i = 1, \ldots, N$, are specific points annotated on the CAD model $\boldsymbol{B}$, and are fixed in number for a given object (see, *e.g.,* Fig. 5). A standard semantic-keypoint-based pose estimator (KeyPO) first detects semantic keypoints $\tilde{\boldsymbol{y}}[i], i = 1, \ldots, N$, from the input point cloud using a neural network. Then, it estimates the object pose by solving an outlier-free registration problem:

$$\hat{\boldsymbol{T}} = \underset{\boldsymbol{T} \in \mathrm{SE}(3)}{\arg\min} \quad \sum_{i=1}^{N} \|\tilde{\boldsymbol{y}}[i] - \boldsymbol{T} \cdot \boldsymbol{b}[i]\|_2^2, \quad (10)$$

that computes the object pose $\boldsymbol{T} \triangleq \begin{bmatrix} \boldsymbol{R} & \boldsymbol{t} \\ \boldsymbol{0} & 1 \end{bmatrix}$ (where $\boldsymbol{R}$ is the object rotation and $\boldsymbol{t}$ is its translation) by aligning the detected

keypoints $\tilde{\boldsymbol{y}}[i]$ and the model keypoints $\boldsymbol{b}[i]$, annotated on the CAD model. After computing the pose estimate $\hat{\boldsymbol{T}}$, we can also compute the *posed* CAD model and keypoints, namely

$$(\hat{\boldsymbol{y}}, \hat{\boldsymbol{X}}) = (\hat{\boldsymbol{T}} \cdot \boldsymbol{b}, \hat{\boldsymbol{T}} \cdot \hat{\boldsymbol{B}}) \tag{11}$$

where $\hat{\boldsymbol{B}}$ denotes a point cloud obtained by densely sampling the CAD model $\boldsymbol{B}$.

Remark 8 (Limitations of Standard Semantic-Keypoints-based Models and Novel Insights)**.** *In the presence of keypoint detection errors* (e.g., *caused by sim-to-real gap), the pose computed by KeyPO might be inaccurate. Moreover, using only a sparse number of keypoints (typically in the order of tens) for pose estimation might lead to less accurate results. For this reason, modern approaches either resort to neural models that mimic ICP [2], [3], or approaches based on learned global representations [4], [5]. In this paper we provide two novel insights: (i) despite common beliefs, standard semantic-keypoint-based methods still outperform newer alternatives in the presence of occlusions and large object displacements (Section VII-A); (ii) we can correct detection errors and factor dense information into the registration phase of semantic-keypoint methods by introducing a* corrector *module, as discussed in the following section.*

B. Overview of C-3PO

C-3PO takes a partial or occluded object point cloud $\boldsymbol{X}$ as input and estimates the object pose $\hat{\boldsymbol{T}}$, as well as a posed point cloud $\hat{\boldsymbol{X}}$ and keypoints $\hat{\boldsymbol{y}}$ as in (11). C-3PO uses a standard semantic keypoint-based pose estimation architecture as the backbone, which first uses a neural network to detect semantic keypoints, and then estimates the 3D pose via registration to the corresponding CAD model. However, contrary to standard semantic keypoint-based methods, C-3PO adds a keypoint *corrector module* that corrects some of the keypoint-detection errors (*cf.* Figs. 4 and 6). Finally, C-3PO implements two certificates to check ζ-correctness of the estimate (Fig. 7). Below, we provide an overview of the key components of C-3PO, while we postpone the details to the following sections.

Semantic Keypoint-based Pose Estimation. A semantic keypoint detection network first detects the semantic keypoints $\tilde{\boldsymbol{y}}[i], i = 1, \ldots, N$, from the input point cloud $\boldsymbol{X}$ (Fig. 6). We implement the keypoint detector in KeyPO as a trainable regression model; in our tests, we use PointNet++ [88] or point transformer [89] as a neural architecture that operates on point clouds. A regression model enables detecting semantic keypoints even in the occluded regions of the input $\boldsymbol{X}$.

Corrector. The semantic keypoint detector might produce perturbed keypoints when tested on real data (*e.g.,* if the detector is trained on a simulation dataset and there is a sim-to-real gap). In C-3PO, we add a *corrector* module that takes the estimated semantic keypoints $\tilde{\boldsymbol{y}}$ —produced by the keypoint detector— as input, and outputs a correction term $\Delta\boldsymbol{y}^*$ to the keypoints. The corrected keypoints become:

$$\boldsymbol{y} = \tilde{\boldsymbol{y}} + \Delta\boldsymbol{y}^*. \tag{12}$$

The resulting architecture is shown in Fig. 6. The correction term $\Delta\boldsymbol{y}^*$ is obtained as a solution to the *corrector optimization problem*, whose details are given in Section V-C. We implement the corrector as a differentiable optimization module, by explicitly computing its gradient. For faster training, we implement batch gradient descent that solves a batch of corrector optimization problems, in parallel, on GPU.

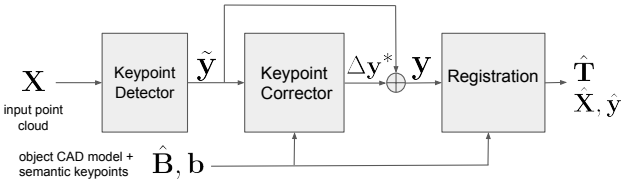


Fig. 6: C-3PO: Proposed semantic-keypoint-based pose estimation and model fitting architecture with *corrector*.

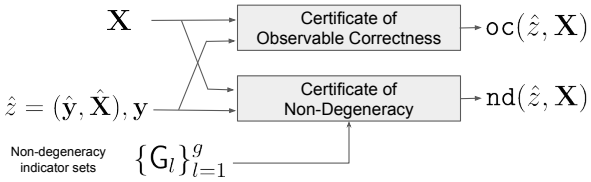


Fig. 7: C-3PO: Certificate of observable correctness and non-Degeneracy. Here, $\{\mathsf{G}_l\}_{l=1}^{g}$ denote the collection of *indicator sets* that are used to determine non-degeneracy; see (7).

Certification. C-3PO implements the two certificates introduced in Section IV, a certificate of observable correctness and a certificate of non-degeneracy, which together determine ζ-correctness of a pose estimate. The key idea is that the certificate of observable correctness assesses whether the output of C-3PO fits the input data. On the other hand, the non-degeneracy certificate ensures that the input point cloud $\boldsymbol{X}$ provides sufficient information on the pose of the object to allow us to unambiguously estimate it. Figure 3 shows an instance in which using just the input point cloud (shown in (a)), it is impossible to estimate the ground-truth pose. The key intuition in C-3PO is that we can use the *keypoints* to infer non-degeneracy: if we choose keypoints to cover the different parts of the object (*e.g.,* in Fig. 3, the keypoints capture the visor, the top button, and other parts of the cap), then as long as we detect specific subsets of keypoints on $\boldsymbol{X}$ (*e.g.,* including the ones on the visor for the cap in Fig. 3), we can unambiguously estimate the object pose. Since we design the two certificates to be conservative, we then use Theorem 5 to conclude that, when the certificates are 1, the resulting estimate is ζ-correct.

C. Keypoint Corrector

The keypoint corrector adds a correction $\Delta\boldsymbol{y}^*$ to the detected keypoints $\tilde{\boldsymbol{y}}$. This is intended to correct for any keypoint detection errors. This section describes how we compute $\Delta\boldsymbol{y}^*$, from the detected keypoints $\tilde{\boldsymbol{y}}$, and the implementation of the corrector as a differentiable optimization module.

First, we introduce some notation. Let $\hat{\boldsymbol{T}}(\Delta\boldsymbol{y})$ denote the optimal solution to the outlier-free registration problem in (10), but with the keypoints $\boldsymbol{y} = \tilde{\boldsymbol{y}} + \Delta\boldsymbol{y}$, instead of $\tilde{\boldsymbol{y}}$ in (10). For

a given $\hat{\boldsymbol{T}}(\Delta \boldsymbol{y})$, let

$$(\hat{\boldsymbol{y}}(\Delta \boldsymbol{y}), \hat{\boldsymbol{X}}(\Delta \boldsymbol{y})) \in \hat{\mathbb{Z}}, \tag{13}$$

be the posed keypoints and CAD model, computed using (11). Namely, $\hat{\boldsymbol{y}}(\Delta \boldsymbol{y})$ are the keypoints $\boldsymbol{b}$ transformed according to the pose $\hat{\boldsymbol{T}}(\Delta \boldsymbol{y})$ and $\hat{\boldsymbol{X}}(\Delta \boldsymbol{y})$ is the dense object point cloud $\hat{\boldsymbol{B}}$ transformed by the same transformation. In (13), we denoted with $\hat{\mathbb{Z}}$ the output space, *i.e.,* the set of all posed keypoints and sample CAD models.

Corrector Optimization Problem. The correction $\Delta \boldsymbol{y}$ is optimized so that the predicted object model $\hat{\boldsymbol{X}}(\Delta \boldsymbol{y})$ and model keypoints $\hat{\boldsymbol{y}}(\Delta \boldsymbol{y})$ are aligned to the input point cloud $\boldsymbol{X}$ and the corrected keypoints $\boldsymbol{y}$:

$$\underset{\Delta \boldsymbol{y} \in \mathbb{R}^{3\times N}}{\text{Minimize}} \quad \text{ch}_{1/2}(\boldsymbol{X}, \hat{\boldsymbol{X}}(\Delta \boldsymbol{y})) + \gamma ||\boldsymbol{y} - \hat{\boldsymbol{y}}(\Delta \boldsymbol{y})||_2^2, \tag{14}$$

where γ is a positive constant and $\text{ch}_{1/2}(\boldsymbol{X}, \hat{\boldsymbol{X}})$ is the half-Chamfer loss given by

$$\text{ch}_{1/2}(\boldsymbol{X}, \hat{\boldsymbol{X}}) = \frac{1}{n} \sum_{\boldsymbol{x} \in \boldsymbol{X}} \min_{\hat{\boldsymbol{x}} \in \hat{\boldsymbol{X}}} ||\boldsymbol{x} - \hat{\boldsymbol{x}}||_2^2. \tag{15}$$

The half-Chamfer loss measures the average squared distance between each point in the input $\boldsymbol{X}$ and the closest point in the estimated model $\hat{\boldsymbol{X}}$. The use of the half-Chamfer loss is motivated by the fact that the input point cloud $\boldsymbol{X}$ is typically occluded, hence we are only interested in assessing how well the visible part of the object fits the posed model.

Forward Pass: Solving (14). The corrector output $\Delta \boldsymbol{y}^*$ is the solution of the optimization problem (14). The optimization problem in (14) is non-linear and non-convex. A number of open-source solvers can be used to solve (14) in the forward pass. We tested different solvers and observed that two solvers work well, *i.e.,* they are able correct large errors in the keypoints. These are a simple constant-step-size gradient descent, which we implement in PyTorch, and a trust-region method [90], implemented in the SciPy library [91]. In implementing these solvers, we need to obtain gradients of the objective function in (14), which depends on the solution to the outlier-free registration problem (10). The solution to (10) involves linear operations and SVD computation [92]–[94], the derivative of which is computed using the autograd functionality in PyTorch. More specifically, we implement *batch gradient descent* – a simple, fixed-step-size gradient descent method implemented in PyTorch that solves (14) for a batch input on the GPU, and this leads to much faster compute times (Appendix E).

Gradient Computation for Back-Propagation. Implementing the corrector as a block in an end-to-end differentiable pipeline requires (14) to be differentiable, *i.e.,* there must be a way to compute $\partial \Delta \boldsymbol{y}^* / \partial \tilde{\boldsymbol{y}}$. We now show that the corrector optimization problem (14), although being non-linear and non-convex, heeds to a very simple derivative (proof in Appendix E). This allows us to implement the corrector as a differentiable optimization module in the C-3PO architecture.

Theorem 9. *The gradient of the correction $\Delta \boldsymbol{y}^*$ with respect to the estimated keypoints $\tilde{\boldsymbol{y}}$ is the negative identity, i.e.,*

$$\partial \Delta \boldsymbol{y}^* / \partial \tilde{\boldsymbol{y}} = -\mathbf{I}. \tag{16}$$

Remark 10 (Corrector vs. ICP)**.** *The corrector is similar in spirit but computationally different from an ICP-based pose refinement. ICP-based methods tend to correct/refine the pose by optimizing over the object rotation or translation. The corrector, on the other hand, does so by optimizing over the keypoint space; see (14). In our experiments, we observe that correcting for pose over the space of keypoints is more powerful (*i.e., *corrects larger errors) compared to doing so over the space of rotation and poses.*

D. Certificates for Pose Estimation

We now use the framework in Section IV to derive two certificates for C-3PO. C-3PO uses a dense point cloud representation $\hat{\boldsymbol{B}}$ of the object $\boldsymbol{B}$, to obtain the posed output $\hat{\boldsymbol{X}}$; see (11). While such a sampling is necessary for implementation (*e.g.,* to make it easy to compute distances between two posed models), the definition of ζ-correctness (Definition 3) requires closeness between the CAD models $\hat{\boldsymbol{T}} \cdot \boldsymbol{B}$ and $\boldsymbol{T}^* \cdot \boldsymbol{B}$.

To bridge this gap, we use two output spaces: (i) $\hat{\mathbb{Z}}$: the space of all outputs $(\hat{\boldsymbol{y}}, \hat{\boldsymbol{X}})$ produced by C-3PO; see (11). This space is nothing but the space of all posed tuple $(\boldsymbol{b}, \hat{\boldsymbol{B}})$ of model keypoints $\boldsymbol{b}$ and sampled object CAD model $\hat{\boldsymbol{B}}$. (ii) $\mathbb{Z}$: the space of all posed tuple $(\boldsymbol{b}, \boldsymbol{B})$:

$$\mathbb{Z} = \left\{ (\boldsymbol{T} \cdot \boldsymbol{b}, \boldsymbol{T} \cdot \boldsymbol{B}) \;\middle|\; \boldsymbol{T} \in \text{SE}(3) \right\}, \tag{17}$$

where now $\boldsymbol{B}$ is the full CAD model.

Remark 11. *The choice of $\mathbb{Z}$ (or $\hat{\mathbb{Z}}$) to be the space of all posed keypoints and models, and not just the space of all poses (*i.e., $\text{SE}(3)$*) circumvents the need to consider symmetric objects as a special case, and helps our analysis (Appendix D).*

Certificate of Observable Correctness. We conservatively approximate the observable correctness certificate by checking the geometric consistency between the input point cloud $\boldsymbol{X}$ and the output $\hat{\boldsymbol{Z}} = (\hat{\boldsymbol{y}}, \hat{\boldsymbol{X}})$ in (11). Since the input point cloud is partial, we only check if every point in the partial input $\boldsymbol{X}$ is close to a point in $\hat{\boldsymbol{X}}$:

$$\circ\text{c}(\hat{\boldsymbol{Z}}, \boldsymbol{X}) = \mathbb{I}\left\{ \max_{i \in [n]} \min_{j \in [m]} \|\boldsymbol{X}[i] - \hat{\boldsymbol{X}}[j]\|_2 < \epsilon_{\circ\text{c}} \right\}, \tag{18}$$

where $\epsilon_{\circ\text{c}}$ is a small positive constant. We show that this certificate can be derived as an outer approximation certificate when the noise $\boldsymbol{n}_w$ in (2) is bounded, and the point cloud representation $\hat{\boldsymbol{B}}$ of $\boldsymbol{B}$ is sufficiently dense (Appendix C).

We note that while the certificate (18) is written as a function of the output $\hat{\boldsymbol{Z}}$ and the input $\boldsymbol{X}$, it can repurposed to the form $\circ\text{c}(\mathcal{M}, \boldsymbol{X})$ (in Section IV) by setting $\hat{\boldsymbol{Z}} = \mathcal{M}(\boldsymbol{X})$.

Certificate of Non-degeneracy. Non-degeneracy depends on the size of the solution space $\mathcal{S}(\boldsymbol{X})$ (see (7)). Intuitively, the solution space is likely to be large if the input point cloud is highly occluded or is missing important features or parts of the object. Consider the case of an otherwise symmetric mug, with a handle. If the input point cloud $\boldsymbol{X}$ misses the handle entirely, there will be multiple ways to register the CAD model to the input – thereby, making $\mathcal{S}(\boldsymbol{X})$ large.

Motivated by this observation, we define the notion of a *cover set* of points on a CAD model.

Definition 12 (Cover Set)**.** *A subset of points $\boldsymbol{A}$ on the surface of a CAD model $\boldsymbol{B}$ is a* cover set *if, given the 3D positions of points in $\boldsymbol{A}$, we can unambiguously estimate (up to noise) the pose of $\boldsymbol{B}$:*

$$\sup_{\boldsymbol{x}\in\boldsymbol{T}'\cdot\boldsymbol{A}} D(\boldsymbol{x},\boldsymbol{T}\cdot\boldsymbol{B}) < \delta \implies d_H(\boldsymbol{T}'\cdot\boldsymbol{B},\boldsymbol{T}\cdot\boldsymbol{B}) < \delta, \quad (19)$$

for any $\boldsymbol{T},\boldsymbol{T}' \in \mathrm{SE}(3)$ and where δ is a positive constant.

Eq. (19) states that if the partial point cloud $\boldsymbol{A}$ of the object $\boldsymbol{B}$ is such that every point in $\boldsymbol{T}'\cdot\boldsymbol{A}$ is close in distance to the posed model $\boldsymbol{T}\cdot\boldsymbol{B}$, then it will necessitate that that two posed models $\boldsymbol{T}\cdot\boldsymbol{B}$ and $\boldsymbol{T}'\cdot\boldsymbol{B}$, are in fact very closely aligned. This implies that if our input point cloud $\boldsymbol{X}$ is large enough to encompass at least a cover set of the object, the size of the solution space must be small; implying non-degeneracy (7).

Our insight is that we can use semantic keypoints, intelligently annotated, to check for the size of the solution space, and if the observed $\boldsymbol{X}$ does indeed encompass a cover set. In our mug example, if we are able to detect a few keypoints on the mug handle, such that those points are also close to the input point cloud, then we can conclude that the input is non-degenerate. This leads to the notion of *indicator set*.

Definition 13 (Indicator Set)**.** *Let $\Theta(\boldsymbol{B})$ be a sampled and visible portion of an object $\boldsymbol{B}$ as in (2). A subset of keypoints $\mathsf{G} \subset [N]$ is an* indicator set *if when all keypoints $\{\boldsymbol{b}[i]\}_{i\in\mathsf{G}}$ are close to $\Theta(\boldsymbol{B})$ by distance δ_{ind},* i.e.,

$$\min_{j\in[n]} \|\boldsymbol{b}[i] - \Theta(\boldsymbol{B})[j]\|_2 < \delta_{ind} \;\forall\; i \in \mathsf{G}, \quad (20)$$

then $\Theta(\boldsymbol{B})$ is a cover set for $\boldsymbol{B}$.

Let $\{\mathsf{G}_l\}_{l=1}^{g}$ be a collection of such indicator sets for the object. Given an input point cloud $\boldsymbol{X}$ and an output $\hat{\boldsymbol{Z}} = (\hat{\boldsymbol{y}},\hat{\boldsymbol{X}}) \in \hat{\mathbb{Z}}$, we declare non-degeneracy if —for at least one set G_l, $l \in [g]$— every keypoint in $\hat{\boldsymbol{y}}[i]$ in G_l is close to a point in the input $\boldsymbol{X}$. Mathematically, this is given by:

$$\mathtt{nd}(\hat{\boldsymbol{Z}},\boldsymbol{X}) = \mathbb{I}\left\{\bigcup_{l=1}^{g}\bigcap_{i\in\mathsf{G}_l}\left\{\min_{j\in[n]}\|\hat{\boldsymbol{y}}[i]-\boldsymbol{X}[j]\|_2 < \delta_{\mathrm{nd}}\right\}\right\}, \quad (21)$$

where δ_{nd} is a small positive constant.

Certification Guarantees. We now prove that using the two certificates (18)-(21), we can determine whether an estimate produced by the model $\mathcal{M}$ is ζ-correct (Definition 1), under some reasonable assumptions. The proof is given in Appendix D.

Assumption 14 (Bounded Noise)**.** *The noise (2) is bounded, namely* $\max_i \|\boldsymbol{n}_w[i]\|_2 < \epsilon_w$.

Assumption 15 (Dense CAD Model Sampling)**.** *The sampled point cloud $\hat{\boldsymbol{B}}$ is such that every point on $\boldsymbol{B}$ is at most a distance ϵ_s away from a point in $\hat{\boldsymbol{B}}$, namely* $\min_{j\in[m]}\|\boldsymbol{x} - \hat{\boldsymbol{B}}[j]\|_2 < \epsilon_s$ *for all $\boldsymbol{x} \in \boldsymbol{B}$.*

Theorem 16. *Assume bounded noise, dense CAD model sampling (Assumptions 14-15), and that the sets $\{\mathsf{G}_l\}_{l=1}^{g}$ in (21) are indicator sets. Then, the output produced by a pose estimator $\mathcal{M}$: $\hat{\boldsymbol{Z}} = (\hat{\boldsymbol{y}},\hat{\boldsymbol{X}}) = \mathcal{M}(\boldsymbol{X})$ is ζ-correct (Definition 1), if $\mathtt{oc}(\hat{\boldsymbol{Z}},\boldsymbol{X}) = \mathtt{nd}(\hat{\boldsymbol{Z}},\boldsymbol{X}) = 1$ in (18), (21) for any $\zeta > \epsilon_{oc} + \epsilon_w$, provided $\epsilon_{oc} + \delta_{nd} + 2\epsilon_w < \delta_{ind}$.*

Remark 17 (Choosing Indicator Sets)**.** *The non-degeneracy certificate in (21) provably works (Theorem 16) when there exists indicator sets, among the annotated keypoints. Deriving indicator sets to provably meet its definition is hard and we do not choose that route. We, instead, hand-craft the indicator sets and show its usefulness towards correctly certifying* C-3PO*'s pose estimates. We show, in our experiments, that a simple choice of subsets (listed in Appendix F) suffices to determine non-degeneracy.*

VI. Self-Supervised Training

The certificates implemented in C-3PO enable a simple yet effective self-supervised training procedure. According to the setup described in Problem 2, we assume that the real-world, training dataset $\mathcal{D}$ consists of a collection of input point clouds $\boldsymbol{X}$, bearing no pose or keypoint annotation. The only trainable part in our C-3PO architecture is the keypoint detector, which is initialized to a sim-trained model. In the self-supervised training, it is the keypoint detector that gets trained to better detect semantic keypoint on real-world data. The other components in C-3PO (*i.e.,* corrector and certificates) enable this self-supervised training.

Certificate-based Self-Supervised Training. Our self-supervised training is the same as a standard supervised training using stochastic gradient descent (SGD), except from two changes. First, at each iteration, when we see a batch of inputs $\{\boldsymbol{X}^i\}$, we compute the output $\{(\hat{\boldsymbol{y}}^i,\hat{\boldsymbol{X}}^i)\}$ using the current model weights. We then compute the loss:

$$\mathcal{L}_i = \mathrm{ch}_{1/2}(\boldsymbol{X}^i,\hat{\boldsymbol{X}}^i) + \theta\sum_{j=1}^{N}\|\boldsymbol{y}^i[j] - \hat{\boldsymbol{y}}^i[j]\|_2^2, \quad (22)$$

for each input-output pair i in the batch; where $\mathrm{ch}_{1/2}$ is the half-Chamfer loss, $\{\boldsymbol{y}^i\}_i$ are the corrected keypoints, and θ is a positive constant.

Second, at each iteration, we also determine the observable correctness of each input-output pair, by computing certificates in (18): $\mathtt{oc}_i = \mathtt{oc}((\hat{\boldsymbol{y}}^i,\hat{\boldsymbol{X}}^i),\boldsymbol{X}^i)$.

We then compute the total training loss for the batch —that we use for back-propagation— as

$$\mathcal{L} = \sum_i \mathtt{oc}_i \cdot \mathcal{L}_i, \quad (23)$$

which is nothing but the sum-total of loss (22) computed only for the observably correct input-output pairs, in the batch.

Remark 18 (Role of Certification)**.** *Self-supervised training using the total loss $\mathcal{L} = \sum_i \mathcal{L}_i$,* i.e., *using all and not just the observably correct instances, would not work well in practice. This is because, some of the predicted models $\hat{\boldsymbol{X}}$ are not correctly registered to the input point clouds $\boldsymbol{X}$. Including them in the loss function (23) induces incorrect supervision causing the self-supervised training to fail. Certification, on the other hand, weeds out incorrectly registered object models during the self-supervised training and provides correct supervision to the keypoint detector during training. In all our experiments*

we observe that as our self-supervised training progresses, the fraction of observably correct instances increases, eventually converging to nearly 100% of the model outputs being observably correct (see Fig. 11, in Section VII-D).

VII. Experiments

We present five experiments. We first show that a standard semantic-keypoint-based architecture outperforms more recent alternatives in problems with partial point clouds and large object displacements (Section VII-A). We then analyze the ability of the corrector to correct errors in the keypoint detections (Section VII-B). We show the effectiveness of our self-supervised training and certification on a depth-point-cloud dataset generated using ShapeNet [31] objects (Section VII-C) and on the YCB dataset [32] (Section VII-D); here we observe that C-3PO significantly outperforms all the baselines and state-of-the-art approaches. Finally, we show that our self-supervised training method can work even when the training data has no object category labels on the input point clouds (Section VII-E).

A. Why Do We Use Semantic Keypoints?

We start by providing an experimental analysis that justifies our choice of KeyPO as the backbone for C-3PO. This choice seems to go against the commonplace belief that neural models that mimic the ICP algorithm [2], [3], or based on learned global representations [4], [5] outperform keypoint-based approaches. This section empirically shows that this conclusion is only true in relatively easy problem instances, while keypoint-based approaches still constitute the go-to solution for hard problems with occlusions and arbitrary object poses.

Setup. We consider the following approaches: (i) FPFH + TEASER++ [1], [43]: a local-feature-based pose estimation method, that extracts local features and computes a pose estimate via robust registration; (ii) PointNetLK [3]: a learning-based model that attempts to "neuralize" iterative closest point and is considered a state-of-the-art approach for point cloud alignment and pose estimation; (iii) DeepGMR [4]: a learning-based model that attempts to extract deep features, by modeling them as latent Gaussian variables; the method is also known to be competitive to PointNetLK; (iv) EquiPose [5]: a learning-based model that attempts to solve shape completion, pose estimation, and in-category generalization; and (v) KeyPO: a standard semantic keypoint-based pose estimation architecture (Fig. 4) with a point transformer [89] as a keypoint detector.

We analyze these models under two experimental settings, named *easy* and *hard*. In the *easy* case, we consider relatively small rotations and translations of the objects, while the *hard* case we induce larger rotations and translations. The cumulative distributions of the object rotations and translations are visualized in Fig. 9. Since several methods tend to perform well when the input $\boldsymbol{X}$ is a full point cloud, as opposed to a depth or a partial point cloud, we consider the following scenarios: (i) Easy + Full PC, where the object displacement is small and the input point cloud is the full point cloud, (ii) Hard + Full PC, where the object displacement is large and the input point cloud is the full point cloud, and (iii) Hard + Depth PC, where the object displacement is large, and the input point cloud is partial and computed from the rendered depth image of the object.

Results. Figure 8 shows the distribution of the ADD-S scores [18] for object pose estimation of a ShapeNet car object and for all the techniques above. From the figure, we note that the performance of all the approaches, except KeyPO, degrades as we go from easy to hard test setups and from full point clouds to depth point clouds. PointNetLK, while performing extremely well on the Easy + Full PC dataset degrades significantly when the object rotations and translations are large. DeepGMR, while still performing well with large pose magnitudes, shows significant performance degradation when the input point cloud is a depth point cloud. EquiPose, while generalizing better, shows consistent sub-optimal performance. In particular, EquiPose seems to trade off its pose estimation accuracy for the objective of in-category generalization, and we consistently see the object shape/size not exactly matching the input. Like EquiPose, the local feature-based method FPFH + TEASER++ also generalizes well, but are still outperformed by KeyPO on the Hard + DepthPC dataset.

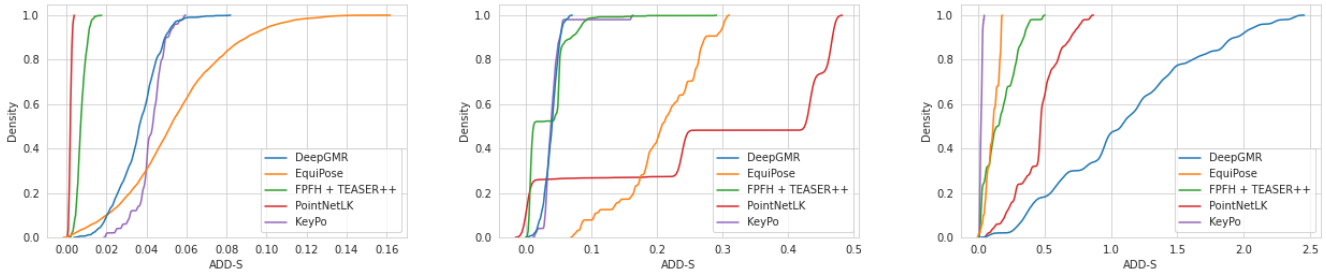
Insights. We gather the following insights from the above analysis. Methods attempting to "neuralize" ICP, for solving object pose estimation, can yield optimal performance under small rotations and translation, while yielding sub-optimal performance when the object displacement is large; this is not dissimilar from ICP, which works well when initialized close to the ground-truth pose, but is prone to converge to local minima otherwise. Methods like DeepGMR, that extract learned representations, while succeeding when the input is a full point cloud, fail when the input is occluded. The performance gap between KeyPO and DeepGMR (on Hard + Depth PC) indicates that we are better off using semantic keypoints as a learned representation for the task of pose estimation.

B. Keypoint Corrector Analysis

This section shows that the keypoint corrector module is able to correct large errors in the keypoint detections.

Setup. We consider ShapeNet objects [31] and use the semantic keypoints labeled by the KeypointNet dataset [71]. Give an object model and semantic keypoints $(\boldsymbol{b}, \boldsymbol{B})$, we extract depth point cloud of $\boldsymbol{B}$ from a certain viewing angle, transform $(\boldsymbol{b}, \boldsymbol{B})$ by a random pose, and add perturbations to the keypoints. The induced pose error is the same as in the *hard* case described in Section VII-A. For each keypoint $\boldsymbol{b}[i]$, with probability f, we add uniform noise distributed in $[-\sigma d/2, \sigma d/2]$, and with probability $1-f$, keep $\boldsymbol{b}[i]$ unperturbed. Here, d is the diameter of the object. We set $f = 0.8$. We study the performance of the corrector as a function of the noise variance parameter σ.

Results. Figure 10(left, mid) show the rotation and translation error as a function of the noise parameter σ; for the ShapeNet object chair. We compare the output of the corrector + registration with a naive method that applies only the registration block (eq. (10)) to the detected/perturbed keypoints $\tilde{\boldsymbol{y}}$. Not all outputs produced by these two models —naive and corrector— are observably correct per eq. (18). The figures also plot the rotation and translation errors for the observably correct instances. Figure 10(right) shows the



(a) Easy + Full PC (b) Hard + Full PC (c) Hard + Depth PC

Fig. 8: Shows the distribution of ADD-S scores [18] for object pose estimation of a ShapeNet object car. Shows various baselines on three datasets: (a) Easy + Full PC, (b) Hard + Full PC, and (c) Hard + Full PC.

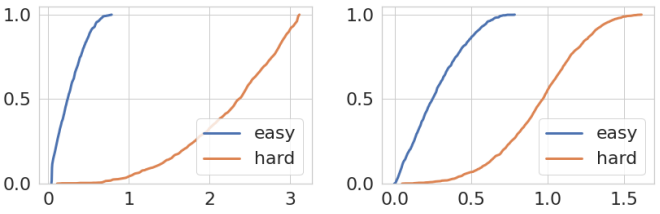
TABLE I: Evaluation of C-3PO and baselines for the ShapeNet experiment.

ADD-S ADD-S (AUC)	car		chair		helmet		laptop		skateboard		table	
KeyPo (sim)	0.00	0.00	0.00	12.01	0.00	2.00	0.00	0.00	4.00	11.30	0.00	3.13
KeyPo (sim) + ICP	0.00	0.00	0.00	11.56	0.00	2.00	2.00	2.00	2.00	9.80	0.00	0.00
KeyPo (sim) + Corr.	0.00	13.69	68.00	57.16	10.00	20.14	26.00	20.57	70.00	59.17	80.00	53.54
C-3PO	80.00	64.00	100.00	79.06	100.00	70.49	100.00	64.70	100.00	82.90	98.00	67.69
C-3PO (oc=1, nd=1)	100.00	84.59	100.00	90.13	100.00	71.53	100.00	64.70	100.00	82.90	100.00	69.07
DeepGMR	0.00	2.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
PointNetLK	0.00	2.76	6.00	7.06	0.00	0.00	0.00	0.00	2.00	5.14	4.00	5.27
FPFH + TEASER++	26.00	25.03	26.00	25.65	50.00	46.29	42.00	32.32	38.00	40.33	48.00	39.23
EquiPose	16.00	19.69	14.00	20.33	–	–	–	–	–	–	–	–
KeyPo (real)	100.00	79.08	100.00	79.92	100.00	80.28	100.00	83.11	100.00	84.02	54.00	44.67

fraction of *non*-observably correct outputs produced by the two methods. We conduct similar analysis for several other ShapeNet objects in [**?**].

Insights. We observe that the corrector significantly improves the rotation and translation error, viz-a-viz the keypoint registration used in standard keypoint-based methods. A further performance boost is seen when we inspect the observably correct, *i.e.,* $\circ\subset = 1$, instances produced by the corrector. This boost is not only in the absolute reduction of the two error metrics, but also in the reduction of the error variance. We observe a near-constant error, w.r.t. increasing keypoint noise σ, for the observably correct instances produced by the corrector. Finally, we note that the fraction observably correct remains significantly high, when using the corrector. For instance, more than 90% of the outputs produced by the corrector remain observably correct, for $\sigma < 1$, whereas the naive method fails to produce any observably correct output for $\sigma \geq 0.4$.

Note that the purpose of the corrector module in C-3PO is to ensure that there is a non-negligible fraction of input-output instances that are observably correct for self-supervised training, thus surmounting the sim-to-real gap. This analysis indicates that the corrector can indeed help bridge the sim-to-real gap.



(a) Rotations (b) Translations

Fig. 9: Cumulative distribution function of object rotations and translations for easy/hard test dataset. The object rotations are in radians (rotation angle around a randomly chosen rotation axis), while the translations is measured as the norm of the translation vector normalized by the object diameter.

C. The ShapeNet Experiment

This section demonstrates our self-supervised training on depth-point-cloud data generated using ShapeNet [31] objects.

Setup. ShapeNet [31] classifies objects into 16 categories. We select one object in each category and use the semantic keypoints labeled by the KeypointNet dataset [71]. We use uniformly sampled point clouds of these 16 objects as the simulation dataset, and a collection of depth point clouds, rendered using Open3D [95], as the real-world dataset. This choice ensures a large sim-to-real gap, and enables us to showcase the utility of our self-supervised training. We consider object displacements corresponding to the *hard* setup described in Section VII-A. We initially train the detector using the simulation dataset, and self-train it on the depth point cloud dataset, as discussed in Section VI. Hyper-parameter tuning and the implementation of the non-degeneracy certificates is discussed in Appendix F.

We report the following baselines in Table I: (i) KeyPO (sim): a simulation-trained keypoint detector KeyPO; (ii) KeyPO (sim) + ICP: the simulation-trained keypoint detector, with iterative closest point refinement using the input $\boldsymbol{X}$ and the predicted point cloud $\hat{\boldsymbol{X}}$; (iii) KeyPO (sim) + corrector, and (iv) KeyPO (real): the keypoint detector trained on the real-wold dataset of depth point clouds, with full supervision. KeyPO (real), therefore, marks an upper-bound on the performance of any self-supervised method. C-3PO denotes the proposed method, obtained after the self-supervised training on real-data.

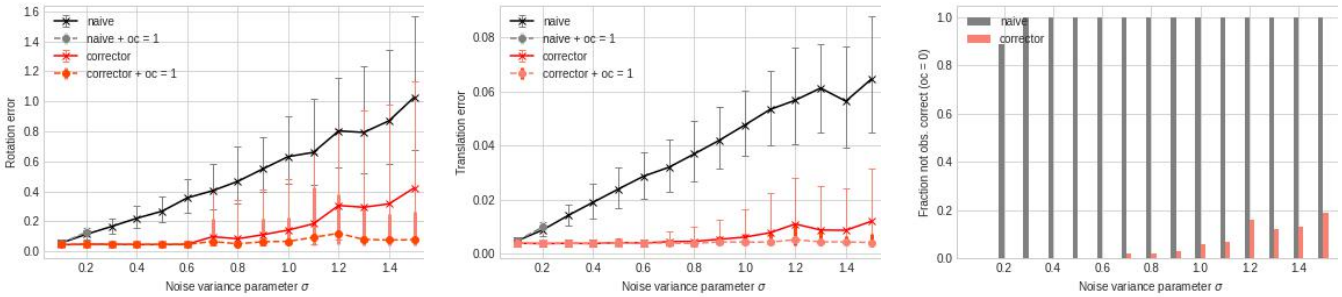


Fig. 10: Rotation error (left), translation error (mid), and fraction of *not* observably correct instances ($\texttt{oc} = 0$) (right), as a function of the noise parameter σ for the "chair"object in ShapeNet. The rotation error is in radians and the translation error is normalized by the object diameter.

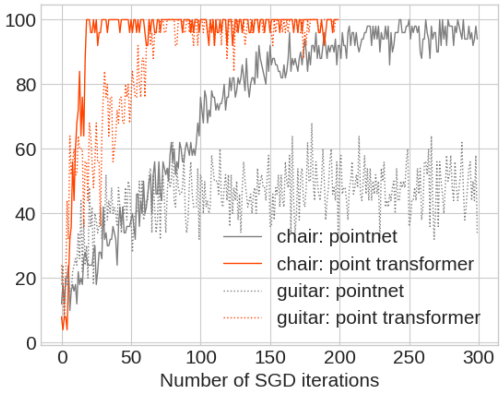


Fig. 11: Percentage of observably correct instances (18) vs. number of SGD iterations during the proposed self-supervised training (Section VI).

We use the point transformer regression model [89] for the keypoint detector in KeyPO and C-3PO. We also compare the performance of C-3PO against: (i) DeepGMR, (ii) PointNetLK, (iii) EquiPose: only in cases where the trained model is available, and (iv) FPFH + TEASER++. Table I shows the performance on 6 of the 16 objects. The remaining objects are reported in [**?**]. We evaluate the performance of each model using threshold ADD-S and ADD-S (AUC) score [18]. We choose thresholds for the ADD-S and ADD-S (AUC) to be 5% and 10% of the object diameter d.

Results. Figure 11 sheds light on the performance of the proposed self-supervised approach and shows how the number of observably correct instances in C-3PO increases with the number of SGD iterations during self-supervised training, reaching close to 100% in most cases. In Fig. 11, there is a noticeable difference in convergence time when the keypoint detector is modeled as a point transformer regression model vs. PointNet++ regression model. The point transformer regression model is consistently better than the PointNet++.

Table I shows that KeyPO (sim) performs very poorly and is not helped much by ICP; thereby, confirming the large sim-to-real gap in the experiment. KeyPO (sim) + Corr., on the other hand, shows a significant performance improvement in Table I; confirming that the corrector indeed helps bridge the sim-to-real gap, to an extent. We, therefore, deduce that correcting for pose over the space of keypoints is more powerful than doing so over the space of rotation and poses, which is what the ICP attempts, thus validating Remark 10.

We observe that C-3PO is able to significantly outperform all the baselines, and match the performance of the fully supervised KeyPO (real). A further performance boost is attained by evaluating only the observably correct and/or non-degenerate outputs produced. We see this in Table I, as well as in Figure 12, which shows the distribution of ADD-S scores, over the test dataset. The reasons for very low scores of the baselines such as DeepGMR, PointNetLK, FPFH + TEASER++, and EquiPose remain the same as discussed in Section VII-A.

Degeneracy. Degeneracy arises when the depth point cloud of an object is severely occluded, and there are multiple ways to fit the object model to it. We observe this specifically in the case of two objects in the ShapeNet dataset: mug and cap. For the mug, degeneracy occurs when the handle of the mug is not visible, whereas for the cap, when the visor of the cap is not visible (Fig. 3). Table II shows the ADD-S and ADD-S (AUC) scores attained by C-3PO on these two objects (cap and mug). We observe that, while the self-supervised training works correctly to reach 100% and 82% observably correct instances for cap and mug, respectively, the ADD-S and ADD-S (AUC) scores remain low. This is because the observably correct instances include degenerate cases. This causes the predicted output to deviate from the ground truth. Table II also shows the ADD-S and ADD-S AUC scores for $\texttt{oc} = 1$ and $\texttt{oc} = \texttt{nd} = 1$ instances produced by C-3PO; see Appendix F for implementation details of the non-degeneracy certificate. We observe that the non-degeneracy check (*i.e.,* $\texttt{nd} = 1$) on observably correct instances (*i.e.,* $\texttt{oc} = 1$) significantly improves the performance, empirically validating Theorem 16.

D. The YCB Experiment

This section shows that the proposed self-training method allows bridging the sim-to-real gap in the YCB dataset [32].

Setup. The YCB dataset [32] includes RGB-D images of household objects on a tabletop. The dataset provides Poisson-reconstructed meshes, which we use as object models $\boldsymbol{B}$. We manually annotate semantic keypoints on each model. For the simulation data, we use uniformly sampled point clouds on $\boldsymbol{B}$, and for the real-world data we use the segmented depth point clouds extracted from the RGB-D images in the dataset.

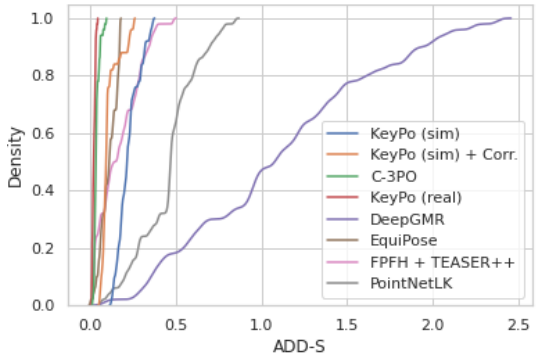


Fig. 12: Distribution of ADD-S score for C-3PO and a few baselines in the ShapeNet Experiment.

TABLE II: C-3PO w/wo observable correctness (18) and non-degeneracy certificates (21).

object	C-3PO	ADD-S	ADD-S AUC	percent
cap	all	78.00	63.87	100.00
	oc =1	78.00	63.87	100.00
	oc, nd =1	100.00	80.11	32.00
mug	all	68.00	53.55	100.00
	oc =1	82.93	60.63	82.00
	oc, nd =1	100.00	70.06	22.00
cracker box	all	68.00	54.24	100.00
	oc =1	69.39	55.35	98.00
	oc, nd =1	100.00	81.08	42.00

We use the same baselines as in the ShapeNet experiment, except EquiPose. This is because a trained EquiPose models are not available for the category-less YCB objects. We evaluate the performance of each model using threshold ADD-S and ADD-S (AUC) score [18]. We choose thresholds for the ADD-S and ADD-S (AUC) to be 2cm and 5cm.

Results and Insights. Table III shows the results for 6 of the 21 YCB objects. The remaining objects are reported in [**?**]. Even though the YCB dataset is generated using real objects and sensors (unlike the dataset in the ShapeNet experiment), the key results and insights in Section VII-C hold. We also see a large sim-to-real gap, and that the KeyPO (sim) and KeyPO (sim) + ICP do not yield good performance. The corrector helps to an extent to bridge the sim-to-real gap, and provide a respectable, initial fraction of observably correct instances for self-supervised training. We observe that the self-supervised training works, as the observable correct instances increase with training iterations. C-3PO significantly outperforms all the baseline approaches, and attains performance close to the supervised baseline KeyPO (real). We observe several degenerate cases (see cracker box in Table III), but our implemented non-degeneracy certificate (see Appendix F) is able to filter them and boost performance: see Table II. This, again, validates Theorem 16.

E. Learning without Object Category Labels

As a final result, we show that C-3PO's self-training works even when the data does not contain object category labels.

Setup. We use the depth point clouds from the ShapeNet and YCB experiments (see Sections VII-C and VII-D). We omit the object category labels and create two mixed datasets. The first one contains five ShapeNet objects (table, chair, bottle, laptop, skateboard) and the second contains five YCB objects (master chef can, mustard bottle, banana, scissors, extra large clamp). We train C-3PO for each object in the dataset. We set $\epsilon_{\text{oc}} = 0.0316d$ where d is the object diameter.

Results and Insights. A typical pose estimation method produces object poses without any certificates. The use of certificates in C-3PO helps it distinguish not only if the estimated pose is correct, but also if the input is a partial point cloud of the same object. Table IV shows C-3PO (A) (for instance, "C-3PO (table)"), trained to estimates pose of object A in sim, produces $\text{oc} = 0$ when it sees other objects (*i.e.,* chair, bottle, laptop, skateboard). Since we use observably correct instances to self-supervise (Section VI), this indicates that the training will ignore pose estimates produced on other objects.

Table V indeed shows that also in this case the self-supervised training in Section VI succeeds. In particular, Table V evaluates C-3PO (chair), trained to estimate the pose of a chair, on a mixed dataset containing depth point clouds of tables, chairs, bottles, laptop, and skateboard. We use the same ADD-S and ADD-S (AUC) thresholds as in Section VII-C. We see that C-3PO (chair) is able to attain performance close to C-3PO and the supervised baseline KeyPO (real) in Table I. We get similar success when training and evaluating C-3PO for other ShapeNet and YCB objects.

VIII. Conclusion

We introduced the problem of certifiable pose estimation from partial point clouds, and defined the notion of ζ-correctness. We developed a theory to certify end-to-end perception models, and showed that implementing two certificates —observable correctness and non-degeneracy— provides a way to ascertain ζ-correctness of the resulting pose estimates. We proposed C-3PO, a certifiable model, that implements the two certificates derived using our theory. C-3PO relies on a semantic keypoint-based architecture, and also implements a keypoint corrector module, which we saw to be more effective at correcting pose errors than traditional methods like ICP. Finally, we introduced a self-supervised training procedure, that leverages the certificate of observable correctness to provide a supervisory signal and enable self-training on unlabeled real data. Our experiments show that (i) standard semantic-keypoint-based methods (which constitute the backbone of C-3PO) outperform more recent alternatives in challenging problem instances, (ii) C-3PO further improves performance and significantly outperforms all the baselines, (iii) C-3PO's certificates are able to discern correct pose estimates.

This work opens many avenues for future investigation. The success of the semantic-keypoint-based architecture for pose estimation indicates the need to explore in-category generalization power of a semantic keypoint detector. We note here that self-supervised discovery of semantic keypoints on CAD models has been shown in [72], thus obviating the need for hand-annotation of semantic keypoints. Self-supervised, category-level keypoint discovery is a problem, which, once solved, can extend C-3PO to solve category-level object pose estimation. Secondly, the theory of certifiable models developed

TABLE III: Evaluation of C-3PO and baselines for the YCB experiment.

ADD-S ADD-S (AUC)	chips can		mustard bottle		banana		scissors		extra large clamp		cracker box	
KeyPo (sim)	0.00	3.84	0.00	15.15	2.00	27.29	34.00	53.22	0.00	11.73	0.00	2.00
KeyPo (sim) + ICP	0.00	4.47	0.00	13.97	4.00	26.09	36.00	52.55	2.00	10.79	0.00	2.00
KeyPo (sim) + Corr.	50.00	41.68	40.00	48.63	72.00	68.06	92.00	80.59	54.00	53.14	38.00	36.18
C-3PO	98.00	78.48	100.00	84.04	100.00	85.97	100.00	84.42	100.00	84.22	68.00	54.24
C-3PO (oc=1, nd=1)	100.00	78.78	100.00	85.05	100.00	87.34	100.00	86.77	100.00	84.80	100.00	81.08
DeepGMR	0.00	7.62	0.00	16.51	2.50	30.49	14.14	35.64	0.76	7.06	0.00	0.00
PointNetLK	0.00	0.00	0.00	3.72	22.17	28.37	7.07	8.08	0.00	0.00	0.00	0.00
FPFH + TEASER++	0.17	5.90	0.33	13.23	4.83	24.43	47.81	50.07	5.29	13.20	0.17	4.91
KeyPo (real)	100.00	83.67	100.00	74.12	100.00	89.92	100.00	88.51	98.00	82.06	98.00	79.34

TABLE IV: Percentage of observably correct instances (*i.e.,* $\%\mathtt{oc} = 1$) for the sim-trained C-3PO, for object A (row), evaluated on the depth point cloud dataset for object B (column)

% obs. correct	table	chair	bottle	laptop	skateboard
C-3PO (table)	74.00	00.00	00.00	00.00	00.00
C-3PO (chair)	00.00	58.00	00.00	00.00	00.00
C-3PO (bottle)	00.00	00.00	98.00	00.00	00.00
C-3PO (laptop)	00.00	00.00	00.00	06.00	00.00
C-3PO (skateboard)	00.00	00.00	00.00	00.00	52.00

TABLE V: Cross evaluation of C-3PO (chair) trained on the mixed ShapeNet dataset to estimate pose of the chair object.

	table	chair	bottle	laptop	skateboard
ADD-S	00.00	100.00	00.00	00.00	00.00
ADD-S (AUC)	00.00	83.73	00.00	00.00	00.00

in this work remains to be applied to other problems. We believe this can pave the way towards certifying end-to-end perception pipelines. Finally, we believe that two other ideas presented in this paper, namely the corrector and the self-supervised training approach, can be extended and applied to other perception problems and deserve further investigation.

APPENDIX

A. Appendix: Hausdorff Metric for Certification

The ζ-correctness of a pose estimate is defined in terms of the Hausdorff distance between the estimated $\hat{\boldsymbol{T}} \cdot \boldsymbol{B}$ and the ground-truth $\boldsymbol{T}^* \cdot \boldsymbol{B}$ posed CAD models. Here we address the question of why we chose the Hausdorff distance, as opposed to other metrics that are more common in the pose estimation literature, like the pose error or the ADD-S score [18].

Pose Error. A common choice of metrics is to use rotation and translation error between the estimated pose $\hat{\boldsymbol{T}}$ and the ground truth $\boldsymbol{T}^*$. The translation error is typically computed as the Euclidean distance between the estimated and ground-truth translation. Several alternatives exist for measuring the rotation error [96], [97]. Two popular options are (i) the angular distance, (ii) the chordal distance, which corresponds to the Frobenius norm of $\boldsymbol{I} - \hat{\boldsymbol{R}}^\mathsf{T}\boldsymbol{R}^*$. While these metrics are appealing, they cause problems for symmetric objects. For a symmetric object, there exist at least two different poses from which the object is identical, *i.e.,* $\boldsymbol{T} \cdot \boldsymbol{B} = \boldsymbol{B}$ for some $\boldsymbol{T} \in \mathrm{SE}(3)$. Using rotation and translation error, therefore, necessitates us to first address the question of such equivalent poses, for each object, thus adding complexity and getting in the way of our analysis.

ADD-S Metric. A popular metric for evaluating pose estimation is the ADD-S metric [18]. It is defined as the Chamfer loss between the estimated $\hat{\boldsymbol{T}} \cdot \hat{\boldsymbol{B}}$ and the ground-truth $\boldsymbol{T}^* \cdot \hat{\boldsymbol{B}}$ posed CAD models. Unlike the metric in (4), ADD-S is computed on sampled point clouds. The difference between such a metric and the Hausdorff distance metric in (4) is that the latter computes the worst-case distance from a point on $\hat{\boldsymbol{T}} \cdot \boldsymbol{B}$ to the surface $\boldsymbol{T}^* \cdot \boldsymbol{B}$, and vice-versa. The ADD-S metric, on the other hand, operates on averages, which can be problematic. An average metric, like ADD-S, tends to average out and potentially miss high errors in a few parts of the object, which gets in the way of certification. The Hausdorff distance metric, using "max" instead of "average", is able to single out such instances.

B. Appendix: Certifiable Models

Proof of Theorem 4. The result follows from the definitions. Recall that the ground truth $\boldsymbol{Z}^*$ generates the input data $\boldsymbol{X}$, hence —according to (5)— it belongs to the solution space: $\boldsymbol{Z}^* \in \mathcal{S}(\boldsymbol{X})$. Now assume that, for an estimate $\boldsymbol{Z} = \mathcal{M}(\boldsymbol{X})$, both the certificate of observable correctness (6) and non-degeneracy (7) are equal to 1. This implies (i) $\boldsymbol{Z} \in \mathcal{S}(\boldsymbol{X})$ and (ii) $\mathrm{Diam}[\mathcal{S}(\boldsymbol{X})] < \delta$. Now, (ii) implies that $d_{\mathbb{Z}}(\boldsymbol{Z}', \boldsymbol{Z}'') < \delta$ for any $\boldsymbol{Z}', \boldsymbol{Z}'' \in \mathcal{S}(\boldsymbol{X})$. Setting $\boldsymbol{Z}' = \boldsymbol{Z}$ and $\boldsymbol{Z}'' = \boldsymbol{Z}^*$, and using the assumption $\delta \leq \zeta$, we obtain the desired result.

Proof of Theorem 5. Since $\overline{\mathcal{S}}(\boldsymbol{X})$ is an outer approximation of the solution set (*i.e.,* $\mathcal{S}(\boldsymbol{X}) \subseteq \overline{\mathcal{S}}(\boldsymbol{X})$) and $\boldsymbol{Z}^* \in \mathcal{S}(\boldsymbol{X})$, it follows $\boldsymbol{Z}^* \in \overline{\mathcal{S}}(\boldsymbol{X})$. Moreover, since $\mathtt{oc}(\mathcal{M}, \boldsymbol{X}) = 1$, we also know that $\boldsymbol{Z} \in \overline{\mathcal{S}}(\boldsymbol{X})$. Now, $\mathtt{nd}(\mathcal{M}, \boldsymbol{X}) = 1$ implies that $d_{\mathbb{Z}}(\boldsymbol{Z}', \boldsymbol{Z}'') < \delta$ for all $\boldsymbol{Z}', \boldsymbol{Z}'' \in \overline{\mathcal{S}}(\boldsymbol{X})$. Putting these three conclusion together, and using the assumption $\delta \leq \zeta$, we obtain the desired result.

C. Appendix: Observable Correctness Certificate (18) as an Outer Approximation

The observable correctness certificate (18) is given by

$$\mathtt{oc}(\hat{\boldsymbol{Z}}, \boldsymbol{X}) = \mathbb{I}\left\{\max_{i \in [n]} \min_{j \in [m]} \|\boldsymbol{X}[i] - \hat{\boldsymbol{T}} \cdot \hat{\boldsymbol{B}}[j]\|_2 < \epsilon_{\mathtt{oc}}\right\}, \tag{24}$$

where we have used the fact that $\hat{\boldsymbol{X}} = \hat{\boldsymbol{T}} \cdot \hat{\boldsymbol{B}}$ in (18). Note that we use the densely sampled CAD model $\hat{\boldsymbol{B}}$ (instead of $\boldsymbol{B}$) to implement the certificate, while the definition of ζ-correctness is with respect to the posed CAD model $\boldsymbol{B}$. For mathematical

analysis, we also consider the following certificate

$$\texttt{oc}'(\hat{\boldsymbol{Z}}, \boldsymbol{X}) = \mathbb{I}\left\{\max_{i\in[n]} D(\boldsymbol{X}[i], \hat{\boldsymbol{T}}\cdot\boldsymbol{B}) < \epsilon_{\texttt{oc}'}\right\}. \tag{25}$$

Note that (25) is an "idealized" version of the certificate in (18), as it computes the exact distance $D(\boldsymbol{X}[i], \boldsymbol{T}\cdot\boldsymbol{B})$ from any point in the point cloud $\boldsymbol{X}$ to the posed CAD model $\boldsymbol{T}\cdot\boldsymbol{B}$. We prove the following.

Theorem 19 (Outer Approximation)**.** *Let $\overline{\mathcal{S}}(\boldsymbol{X})$ be given by*

$$\overline{\mathcal{S}}(\boldsymbol{X}) = \left\{(\boldsymbol{T}\cdot\boldsymbol{b}, \boldsymbol{T}\cdot\boldsymbol{B}) \;\middle|\; \max_{i\in[n]} D(\boldsymbol{X}[i], \boldsymbol{T}\cdot\boldsymbol{B}) < \epsilon_{\texttt{oc}'}\right\}.$$

Assuming bounded noise (Assumption 14) and dense CAD model sampling (Assumption 15), we have

(i) $\overline{\mathcal{S}}(\boldsymbol{X})$ is an outer approximation of the solution space $\mathcal{S}(\boldsymbol{X})$, provided $\epsilon_w < \epsilon_{\texttt{oc}'}$.

(ii) $\texttt{oc}'(\boldsymbol{Z}, \boldsymbol{X}) \leq \texttt{oc}(\boldsymbol{Z}, \boldsymbol{X})$, provided $\epsilon_w + \epsilon_s < \epsilon_{\texttt{oc}}$.

Proof: **(i)** We first show that $\overline{\mathcal{S}}(\boldsymbol{X})$ is an outer approximation to the solution space $\mathcal{S}(\boldsymbol{X})$. Recall that, given a generative model $\phi : \mathbb{Z} \to \mathbb{X}$, the solution space $\mathcal{S}(\boldsymbol{X})$ is given by

$$\mathcal{S}(\boldsymbol{X}) = \{\boldsymbol{Z} \in \mathbb{Z} \mid \phi(\boldsymbol{Z}) = \boldsymbol{X}\}. \tag{26}$$

For the certifiable pose estimation problem, we have the generative model (11):

$$\boldsymbol{X} = \Theta\left(\boldsymbol{T}^*\cdot\boldsymbol{B}\right) + \boldsymbol{n}_w. \tag{27}$$

Let a tuple $(\boldsymbol{T}\cdot\boldsymbol{b}, \boldsymbol{T}\cdot\boldsymbol{B}) \in \mathbb{Z}$ be in the solution space $\mathcal{S}(\boldsymbol{X})$. This implies that there exists a realization of the noise $\boldsymbol{n}_w^{'}$ such that $\boldsymbol{X} - \boldsymbol{n}_w^{'} = \Theta\left(\boldsymbol{T}\cdot\boldsymbol{B}\right)$, which is equivalent to saying that $\boldsymbol{X} - \boldsymbol{n}_w^{'} \subseteq \boldsymbol{T}\cdot\boldsymbol{B}$. We can write this as

$$D(\boldsymbol{X}[i] - \boldsymbol{n}_w^{'}[i], \boldsymbol{T}\cdot\boldsymbol{B}) = 0, \;\forall\; i \in [n] \tag{28}$$

The equation simply states that the output, up to noise, is a rigid transformation of points in the CAD model. Since we assume bounded noise (Assumption 14), we can write (28) as

$$\max_{i\in[n]} D(\boldsymbol{X}[i], \boldsymbol{T}\cdot\boldsymbol{B}) < \epsilon_w, \tag{29}$$

after making use of the triangle inequality. From this, we know that whenever $(\boldsymbol{T}\cdot\boldsymbol{b}, \boldsymbol{T}\cdot\boldsymbol{B}) \in \mathcal{S}(\boldsymbol{X})$ we must have (29), which is equivalent to saying $(\boldsymbol{T}\cdot\boldsymbol{b}, \boldsymbol{T}\cdot\boldsymbol{B}) \in \overline{\mathcal{S}}(\boldsymbol{X})$ with $\epsilon_w < \epsilon_{\texttt{oc}'}$.

(ii) We now show $\texttt{oc}'(\boldsymbol{Z}, \boldsymbol{X}) \leq \texttt{oc}(\boldsymbol{Z}, \boldsymbol{X})$, provided $\epsilon_w + \epsilon_s < \epsilon_{\texttt{oc}}$. It suffices to prove that whenever $\texttt{oc}'(\boldsymbol{Z}, \boldsymbol{X}) = 1$ we have $\texttt{oc}(\boldsymbol{Z}, \boldsymbol{X}) = 1$. We do this by showing

$$\max_{i\in[n]} D(\boldsymbol{X}[i], \boldsymbol{T}\cdot\boldsymbol{B}) < \epsilon_w \implies \max_{i\in[n]} \min_{j\in[m]} \|\boldsymbol{X}[i] - \hat{\boldsymbol{X}}[j]\|_2 < \epsilon_{\texttt{oc}}, \tag{30}$$

provided $\epsilon_w + \epsilon_s < \epsilon_{\texttt{oc}}$.

Let the left-hand side (LHS) of (30) hold for a $\boldsymbol{T}$. We note that

$$\|\boldsymbol{X}[i] - \hat{\boldsymbol{X}}[j]\|_2 \leq \|\boldsymbol{X}[i] - \boldsymbol{x}\|_2 + \|\boldsymbol{x} - \hat{\boldsymbol{X}}[j]\|_2, \tag{31}$$

for a $\boldsymbol{x} \in \boldsymbol{T}\cdot\boldsymbol{B}$ such that $\|\boldsymbol{X}[i] - \boldsymbol{x}\|_2 < \epsilon_w$; which is possible due the LHS of (30). Taking minimum over $j \in [m]$, we obtain

$$\min_{j\in[m]} \|\boldsymbol{X}[i] - \hat{\boldsymbol{X}}[j]\|_2 < \epsilon_w + \epsilon_s, \tag{32}$$

after applying the CAD model sampling Assumption 15. This is nothing but the right-hand side of (30), provided $\epsilon_w + \epsilon_s < \epsilon_{\texttt{oc}}$. ■

Remark 20. *We remark here that Theorems 19 and 5 imply that the implemented observable correctness certificate is an upper-bound,* i.e.,

$$\texttt{ObsCorrect}[\mathcal{M}, \boldsymbol{X}] \leq \texttt{oc}(\boldsymbol{Z} = \mathcal{M}(\boldsymbol{X}), \boldsymbol{X}), \tag{33}$$

under Assumptions 14-15 and $\epsilon_{\texttt{oc}} > \epsilon_w + \epsilon_s$.

D. Appendix: Proof of Theorem 16

We prove the result in three parts: (i) we first prove that if the certificate of non-degeneracy holds, then the posed keypoints cannot be far from the posed ground-truth CAD model; (ii) if the certificate of observable correctness holds, then the posed CAD model cannot be far away from the posed and occluded ground-truth CAD model; and finally (iii) we use these two to show that the estimated pose $\hat{\boldsymbol{T}}$ is ζ-correct.

(i): Let $\hat{\boldsymbol{Z}} = (\hat{\boldsymbol{y}}, \hat{\boldsymbol{X}}) = \mathcal{M}(\boldsymbol{X})$ be the model output as in (11). Let $\texttt{nd}(\hat{\boldsymbol{Z}}, \boldsymbol{X}) = 1$. Then, there exists a G_l such that

$$\min_{j\in[n]} \|\hat{\boldsymbol{y}}[i] - \boldsymbol{X}[j]\|_2 < \delta_{\texttt{nd}}, \;\forall\; i \in \mathsf{G}_l. \tag{34}$$

Now, $\boldsymbol{X} = \Theta\left(\boldsymbol{T}^*\cdot\boldsymbol{B}\right) + \boldsymbol{n}_w$. This implies

$$\|\hat{\boldsymbol{y}}[i] - \Theta\left(\boldsymbol{T}^*\cdot\boldsymbol{B}\right)[j]\|_2 \leq \|\hat{\boldsymbol{y}}[i] - \boldsymbol{X}[j]\|_2 + \|\boldsymbol{n}_w[j]\|_2, \tag{35}$$

$$\leq \|\hat{\boldsymbol{y}}[i] - \boldsymbol{X}[j]\|_2 + \epsilon_w, \tag{36}$$

where we used the bounded noise Assumption 14. Now, eqs. (36) and (34) imply

$$\min_{j\in[n]} \|\hat{\boldsymbol{y}}[i] - \Theta\left(\boldsymbol{T}^*\cdot\boldsymbol{B}\right)[j]\|_2 < \delta_{\texttt{nd}} + \epsilon_w, \;\forall\; i \in \mathsf{G}_l. \tag{37}$$

This proves that the posed keypoints $\hat{\boldsymbol{y}} = \hat{\boldsymbol{T}}\cdot\boldsymbol{b}$, *cf.* (11), are not far away from the posed ground-truth CAD model. We use $\hat{\boldsymbol{y}} = \hat{\boldsymbol{T}}\cdot\boldsymbol{b}$, *cf.* (11), to write (37) as

$$\min_{j\in[n]} \|\boldsymbol{b}[i] - \boldsymbol{C}[j]\|_2 < \delta_{\texttt{nd}} + \epsilon_w, \;\forall\; i \in \mathsf{G}_l, \tag{38}$$

where $\boldsymbol{C} = \hat{\boldsymbol{T}}^{-1}\cdot\Theta\left(\boldsymbol{T}^*\cdot\boldsymbol{B}\right)$.

(ii): We also have $\texttt{oc}(\hat{\boldsymbol{Z}}, \boldsymbol{X}) = 1$. This implies

$$\min_{i\in[m]} \|\boldsymbol{X}[j] - \hat{\boldsymbol{X}}[i]\|_2 < \epsilon_{\texttt{oc}}, \;\forall\; j \in [n]. \tag{39}$$

Again, using the fact that $\boldsymbol{X} = \Theta\left(\boldsymbol{T}^*\cdot\boldsymbol{B}\right) + \boldsymbol{n}_w$, and that the noise $\boldsymbol{n}_w$ is bounded (Assumption 14), we can show

$$\min_{i\in[m]} \|\Theta\left(\boldsymbol{T}^*\cdot\boldsymbol{B}\right)[j] - \hat{\boldsymbol{X}}[i]\|_2 < \epsilon_{\texttt{oc}} + \epsilon_w, \;\forall\; j \in [n]. \tag{40}$$

Recall that $\hat{\boldsymbol{X}} = \hat{\boldsymbol{T}}\cdot\hat{\boldsymbol{B}}$ and consider the posed CAD model $\hat{\boldsymbol{T}}\cdot\boldsymbol{B}$. Using (40), and the fact that every point in $\hat{\boldsymbol{X}}$ is in the set $\hat{\boldsymbol{T}}\cdot\boldsymbol{B}$, we get

$$D(\Theta\left(\boldsymbol{T}^*\cdot\boldsymbol{B}\right)[j], \hat{\boldsymbol{T}}\cdot\boldsymbol{B}) < \epsilon_{\texttt{oc}} + \epsilon_w, \;\forall\; j \in [n]. \tag{41}$$

Substituting $\boldsymbol{C} = \hat{\boldsymbol{T}}^{-1}\cdot\Theta\left(\boldsymbol{T}^*\cdot\boldsymbol{B}\right)$ in (41) we get

$$D(\boldsymbol{C}, \boldsymbol{B}) < \epsilon_{\texttt{oc}} + \epsilon_w, \;\forall\; j \in [n]. \tag{42}$$

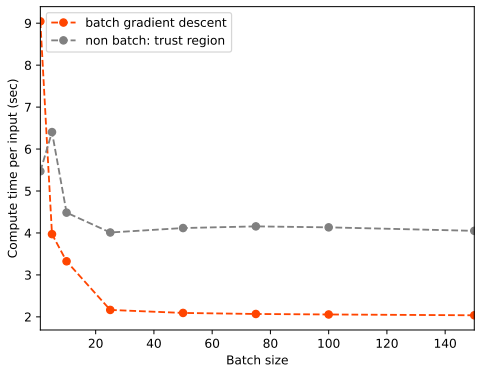


Fig. 13: Compute time (ms) for solving (14) per data point in a batched input, as a function of the batch size.

This implies that there exists an occlusion and sampling function Θ' such that

$$d_H(\boldsymbol{C}, \Theta'(\boldsymbol{B})) < \epsilon_{\text{oc}} + \epsilon_w. \tag{43}$$

Such a point cloud $\Theta'(\boldsymbol{B})$ can be obtained by only sampling points on $\boldsymbol{B}$ that are closest to every point in $\boldsymbol{C}$.

(iii): From (38) and (43), we obtain

$$\min_j \|\boldsymbol{b}[i] - \Theta'(\boldsymbol{B})[j]\|_2 < \epsilon_{\text{oc}} + \delta_{\text{nd}} + 2\epsilon_w, \tag{44}$$

for all $i \in \mathsf{G}_l$, using the triangle inequality. This means that all the keypoints in the indicator set G_l are close to the non-occluded part of $\Theta'(\boldsymbol{B})$. Since G_l is an indicator set (Definition 13), we have that $\Theta'(\boldsymbol{B})$ is a cover set of $\boldsymbol{B}$, provided

$$\epsilon_{\text{oc}} + \delta_{\text{nd}} + 2\epsilon_w < \delta_{ind}. \tag{45}$$

Now, (43) also implies that $D(\Theta'(\boldsymbol{B})[j], \boldsymbol{C}) < \epsilon_{\text{oc}} + \epsilon_w$, from which we can deduce

$$D(\Theta'(\boldsymbol{B})[j], \hat{\boldsymbol{T}}^{-1} \cdot \boldsymbol{T}^* \cdot \boldsymbol{B}) < \epsilon_{\text{oc}} + \epsilon_w, \tag{46}$$

for all j. Knowing that $\Theta'(\boldsymbol{B})$ is a cover set, this gives us $d_H(\boldsymbol{B}, \hat{\boldsymbol{T}}^{-1} \cdot \boldsymbol{T}^* \cdot \boldsymbol{B}) < \epsilon_{\text{oc}} + \epsilon_w$, and therefore,

$$d_H(\hat{\boldsymbol{T}} \cdot \boldsymbol{B}, \boldsymbol{T}^* \cdot \boldsymbol{B}) < \epsilon_{\text{oc}} + \epsilon_w, \tag{47}$$

which proves the result. This proves that the estimate produced $\hat{\boldsymbol{T}}$ is ζ-correct for any $\zeta > \epsilon_{\text{oc}} + \epsilon_w$.

E. Appendix: Corrector

Proof of Theorem 9. Note that the only places where $\Delta\boldsymbol{y}$ appears in the corrector is in conjunction with $\tilde{\boldsymbol{y}}$, and that too, in the form $\tilde{\boldsymbol{y}} + \Delta\boldsymbol{y}$ (see (10), (14)). The corrector optimization problem can therefore be written as:

$$\min_{\Delta\boldsymbol{y}} \quad f(\tilde{\boldsymbol{y}} + \Delta\boldsymbol{y}), \tag{48}$$

for some function f, obtained by subsuming the constraints in (14) into the objective function and realizing that $\boldsymbol{R}(\Delta\boldsymbol{y})$ and $\boldsymbol{t}(\Delta\boldsymbol{y})$, can in fact be written as $\boldsymbol{R}(\tilde{\boldsymbol{y}} + \Delta\boldsymbol{y})$ and $\boldsymbol{t}(\tilde{\boldsymbol{y}} + \Delta\boldsymbol{y})$.

Let $\boldsymbol{y}^*$ be a solution of the optimization problem $\min_{\boldsymbol{y}} f(\boldsymbol{y})$, for f in (48). Then, the optimal correction will be $\Delta\boldsymbol{y}^* = \boldsymbol{y}^* - \tilde{\boldsymbol{y}}$. Taking derivative with respect to $\tilde{\boldsymbol{y}}$ gives the result.

Batch Gradient Descent. To enable batch processing in the corrector forward propagation, we implement *batch gradient descent* – a simple, fixed-step-size gradient descent implemented in PyTorch that solves (14) for a batch input on the GPU, in parallel, without iterating over data points in the batch. This leads to much faster compute times, while solving (14) quite accurately.

Figure 13 shows a comparison of the compute time advantage viz-a-viz a SciPy optimization solver. The figure plots the time taken by a solver to output the solution to (14), per data point in the batch, and a function of the total batch size. We see that with increasing batch size, the implemented batch gradient descent yields a compute time advantage over the SciPy solver and over using batch size of one.

Remark 21 (Batch Solvers for Differentiable Optimization Layers)**.** *We remark here that any differentiable optimization layer require batch solvers to enable faster training. We observe that implementations using existing solvers, and iterating over data points in the batch, results in slower training times. While PyTorch has many optimization algorithms (*e.g.*, SDG, ADAM), they are not suited to solve non-linear optimization problems by design.*[1]

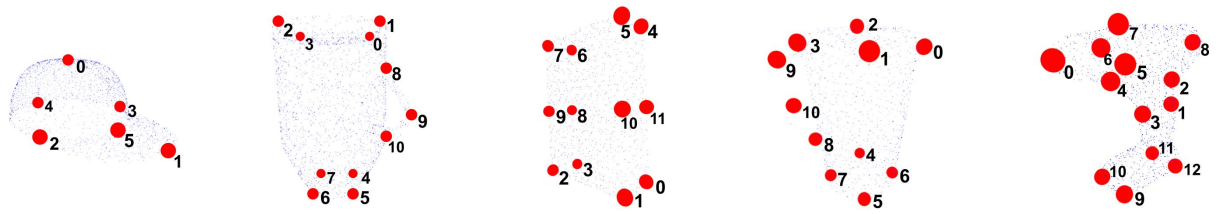
F. Appendix: The ShapeNet and YCB Experiment

Non-degeneracy Certificate. In this appendix, we discuss the implementation of the non-degeneracy certificates and the choice of indicator sets, used in the experiments. We note that deriving indicator sets to provably meet the specifications in Definition 13 is hard and we do not choose that route. We, instead, hand-craft the indicator sets and show its usefulness towards correctly certifying C-3PO's pose estimates.

For most of the ShapeNet and YCB objects, we choose the indicator set to be empty, *i.e.*, $\mathsf{G}_l = \emptyset$, implying that $\texttt{nd}(\hat{\boldsymbol{Z}}, \boldsymbol{X}) = 1$ always. This is because any depth point cloud of the object turns out to be a valid cover set, and therefore, can uniquely identify the object pose (see (19)). However, for some objects, this turns out not to be the case (*e.g.*, cap in Figure 3). We implement non-trivial non-degeneracy certificates for ShapeNet object's cap and mug, and for YCB object's boxes (of any kind), pitcher base, and power drill.

Let $\mathcal{G} = \{\mathsf{G}_l\}_{l=1}^g$ denote the collection of all the indicator sets. We describe our implementation of $\mathcal{G}$ for the few cases in which we implement a non-trivial non-degeneracy certificate. Figure 14 shows a list of five ShapeNet and YCB objets, along with annotated semantic keypoints and their indices. For these objects we implement a non-trivial, non-degeneracy certificate. For the cap and mug, we choose $\mathcal{G} = \{\text{set}([1])\}$ and $\mathcal{G} = \{\text{set}([9])\}$, respectively. Note that [1] is the keypoint on the flap of the cap, while [9] is the keypoint on the handle of the mug. This choice requires that the input point cloud $\boldsymbol{X}$ have points near the flap of the cap and the handle of the mug, to be deemed as a cover set, and thereby, declare the input $\boldsymbol{X}$ as non-degenerate. For 019 pitcher base we choose $\mathcal{G} = \{$ set([8]), set([10])$\}$ as non-visibility of the handle causes degeneracy, and for 035 power drill we choose $\mathcal{G} = \{$set([9]), set([10]), set([11]), set([12])$\}$ as non-visibility of the base of the power drill leads to degeneracy.

[1] SGD and ADAM are designed to solve stochastic optimization problems.



(a) cap (b) mug (c) 004 sugar box (d) 019 pitcher base (e) 035 power drill

Fig. 14: Annotated keypoints and keypoint indices for ShapeNet objects (a) cap, (b) mug and for YCB objects (c) 004 sugar box, (d) 019 pitcher base, and (e) 035 power drill.

TABLE VI: Comparing the keypoint detector architectures used in C-3PO: PointNet++ vs Point Transformer.

Object	ADD-S	ADD-S AUC	% oc=1	keypoint detector
airplane	100.00	87.76	100.00	Point Transformer
	58.00	63.35	54.00	PointNet++
bathtub	72.00	57.71	60.00	Point Transformer
	54.00	50.40	50.00	PointNet++
bed	56.00	52.68	0.00	Point Transformer
	34.00	38.14	0.00	PointNet++
bottle	100.00	70.35	100.00	Point Transformer
	100.00	68.41	100.00	PointNet++
cap	78.00	63.87	100.00	Point Transformer
	60.00	47.55	82.00	PointNet++
car	80.00	64.00	6.00	Point Transformer
	46.00	44.53	0.00	PointNet++
chair	100.00	79.06	12.00	Point Transformer
	100.00	73.44	2.00	PointNet++
guitar	100.00	83.77	28.00	Point Transformer
	100.00	80.03	0.00	PointNet++
helmet	100.00	70.49	92.00	Point Transformer
	60.00	47.78	32.00	PointNet++
knife	100.00	86.94	86.00	Point Transformer
	100.00	80.28	20.00	PointNet++
laptop	100.00	64.70	100.00	Point Transformer
	82.00	56.67	86.00	PointNet++
motorcycle	100.00	75.02	18.00	Point Transformer
	98.00	71.33	0.00	PointNet++
mug	68.00	53.55	82.00	Point Transformer
	32.00	37.38	52.00	PointNet++
skateboard	100.00	82.90	100.00	Point Transformer
	96.00	79.66	96.00	PointNet++
table	98.00	67.69	98.00	Point Transformer
	100.00	64.61	100.00	PointNet++
vessel	100.00	78.21	80.00	Point Transformer
	80.00	63.23	20.00	PointNet++

We implement non-trivial non-degeneracy certificate for all the boxes in the YCB dataset. This is because it is impossible to determine the correct pose of the box by seeing the depth point cloud of only one side. We therefore implement $\mathcal{G}$ = {set([0,1,3,4]), set([0,1,2,5]), set([1,2,3,7]), set([0,2,3,6])set([4,5,6,0]), set([4,5,7,1]), set([5,6,7,2]), set([4,6,7,3])}, which requires visibility of at least three sides of the box to be declared non-degenerate in (7). We choose $\delta_{\texttt{nd}}$ in (21) to be 1.5% of the object diameter for ShapeNet objects and 1.5cm for the YCB objects.

TABLE VII: C-3PO w/wo observable correctness and non-degeneracy certificates.

object	C-3PO	ADD-S	ADD-S AUC	percent
sugar box	all	58.00	52.03	100.00
	oc =1	58.33	52.57	96.00
	oc, nd =1	100.00	87.43	50.00
pudding box	all	50.00	53.82	100.00
	oc =1	26.09	36.52	46.00
	oc, nd =1	100.00	89.73	8.00
gelatin box	all	82.00	78.07	100.00
	oc =1	73.53	74.41	68.00
	oc, nd =1	100.00	89.26	42.00
potted meat can	all	54.00	72.17	100.00
	oc =1	61.90	75.86	84.00
	oc, nd =1	100.00	88.81	36.00
power drill	all	86.00	81.42	100.00
	oc =1	97.22	85.53	72.00
	oc, nd =1	100.00	87.13	56.00
wood block	all	62.00	50.17	100.00
	oc =1	48.28	39.85	58.00
	oc, nd =1	100.00	83.11	22.00
foam brick	all	98.00	85.33	100.00
	oc =1	96.67	85.74	60.00
	oc, nd =1	100.00	89.68	42.00

Remark 22 (Learning Indicator Sets). *While the implementation of the non-degeneracy certificate is hand-crafted, we only use it at test time and on a few objects. For most objects, evaluations show that observable correctness imply certifiable correctness. We leave it to future work to implement a more exact and systematic non-degeneracy certificate –probably a learning-based model, that does not require hand-crafting.*

Hyper-parameters. We describe our choice of learning rate, number of epochs, the $\epsilon_{\texttt{oc}}$ used in certification (Definition 3), and other parameters used during the self-supervised training. We set the maximum number of epochs to be 20 for ShapeNet and 50 for YCB objects. We made two exceptions in the case of ShapeNet, where we observed that the models for "car" and "helmet" were still training (*i.e.,* observed continuing decrease in the train and validation loss, and improvement in the percent certifiable). For these two models we set the maximum number of epochs to 40. For the learning rate, we used 0.02, momentum to be 0.9, while the training batch size was 50. We did not optimize much over these parameters, but only found ones that worked. We believe some improvement can be expected by exhaustively optimizing these training hyper-parameters.

We observe that correctly setting up the certification param-

eter $\epsilon_{\circ\subset}$ in (18) plays a crucial role in efficiently training the model in a self-supervised manner. Setting $\epsilon_{\circ\subset}$ too high allows for the model to make and accept errors in training, while setting it to too tight leads to increased training time, as a very small number of input-output pairs remain certifiable, for the simulation-trained model with the corrector, at the beginning of the self-supervised training. A good $\epsilon_{\circ\subset}$ can be obtained either by grid search or by visually inspecting the simulation-trained KeyPO, applied on the real-world data, with the corrector. For instance, if all the declared $\circ\subset = 1$ instances are also visually correct (*i.e.,* posed CAD model $\hat{\boldsymbol{X}}$ is correctly aligned with the input point cloud $\boldsymbol{X}$), and the $\circ\subset = 0$ instances are not, then the chosen $\epsilon_{\circ\subset}$ is correct. Table VIII lists the $\epsilon_{\circ\subset}$ used for various objects in the ShapeNet and YCB dataset.

Remark 23 (Automatic Tuning of $\epsilon_{\circ\subset}$)**.** *The parameter $\epsilon_{\circ\subset}$ in (18) (i) determines the observable correctness of the input-output pairs and (ii) influences the proposed self-supervised training (see Section VI). This dual purpose makes $\epsilon_{\circ\subset}$ amenable for automatic tuning during self-supervised training. We think that it is possible to use the number of observably correct instances, during training, as an indicator to automatically adjust $\epsilon_{\circ\subset}$. However, we leave this thread of inquiry for future work.*

PointNet++ vs Point Transformer. In Section VII-C, we noted that the C-3PO works better when we use point transformer as a keypoint detector, as opposed to PointNet++, *i.e.,* the former converges to 100% certifiability much sooner during training (see Fig. 11). Table VI reports the ADD-S, ADD-S (AUC), and percentage certifiable for models trained using the point transformer and the PointNet++ architecture. We observe that the point transformer performs better in all the cases.

REFERENCES

[1] H. Yang, J. Shi, and L. Carlone, "TEASER: Fast and Certifiable Point Cloud Registration," *IEEE Trans. Robotics*, vol. 37, no. 2, pp. 314–333, 2020. extended arXiv version 2001.07715 (pdf).

[2] Y. Aoki, H. Goforth, R. A. Srivatsan, and S. Lucey, "PointNetLK: robust & efficient point cloud registration using PointNet," in *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pp. 7163–7172, 2019.

[3] X. Li, J. K. Pontes, and S. Lucey, "Pointnetlk revisited," in *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pp. 12758–12767, 2021.

[4] W. Yuan, B. Eckart, K. Kim, V. Jampani, D. Fox, and J. Kautz, "Deepgmr: Learning latent gaussian mixture models for registration," in *European Conf. on Computer Vision (ECCV)*, p. 733–750, Aug. 2020.

[5] X. Li, Y. Weng, L. Yi, L. Guibas, A. L. Abbott, S. Song, and H. Wang, "Leveraging SE(3) Equivariance for Self-Supervised Category-Level Object Pose Estimation," in *Advances in Neural Information Processing Systems (NIPS)*, Dec. 2021.

[6] The New York Times, "Self-Driving Uber Car Kills Pedestrian in Arizona, Where Robots Roam." https://www.nytimes.com/2018/03/19/technology/uber-driverless-fatality.html, 2018.

[7] H. Yang and L. Carlone, "Certifiably optimal outlier-robust geometric perception: Semidefinite relaxations and scalable global optimization," *IEEE Trans. Pattern Anal. Machine Intell.*, 2022. (pdf).

[8] M. Garcia-Salguero, J. Briales, and J. Gonzalez-Jimenez, "Certifiable relative pose estimation," *Image and Vision Computing*, vol. 109, p. 104142, 2021.

[9] H. Yang and M. Pavone, "Conformal Semantic Keypoint Detection with Statistical Guarantees," in *NeurIPS Workshop on Robot Learning: Trustworthy Robotics*, Nov. 2022.

[10] G. Shafer and V. Vovk, "A Tutorial on Conformal Prediction," *J. of Machine Learning Research*, p. 51, 2008.

[11] A. N. Angelopoulos and S. Bates, "A Gentle Introduction to Conformal Prediction and Distribution-Free Uncertainty Quantification," *arXiv*, Sep. 2022.

[12] K. Xu, H. Zhang, S. Wang, Y. Wang, S. Jana, X. Lin, and C.-J. Hsieh, "Fast and complete: Enabling complete neural network verification with rapid and massively parallel incomplete verifiers," in *Intl. Conf. on Learning Representations (ICLR)*, Sep. 2021.

[13] S. Wang, H. Zhang, K. Xu, X. Lin, S. Jana, C.-J. Hsieh, and J. Z. Kolter, "Beta-CROWN: Efficient bound propagation with per-neuron split constraints for neural network robustness verification," in *Advances in Neural Information Processing Systems (NIPS)*, Nov. 2021.

[14] A. Albarghouthi, "Introduction to Neural Network Verification," *arXiv*, Oct. 2021.

[15] C. Gümeli, A. Dai, and M. Nießner, "ROCA: Robust CAD Model Retrieval and alignment from a single image," in *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2022.

[16] K. Park, T. Patten, and M. Vincze, "Pix2Pose: Pixel-Wise Coordinate Regression of Objects for 6D Pose Estimation," *Intl. Conf. on Computer Vision (ICCV)*, pp. 7667–7676, Oct. 2019.

[17] 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.

[18] C. Wang, D. Xu, Y. Zhu, R. Martin-Martin, C. Lu, L. Fei-Fei, and S. Savarese, "Densefusion: 6d object pose estimation by iterative dense fusion," in *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pp. 3338–3347, Jun. 2019.

[19] A. Kundu, Y. Li, and J. M. Rehg, "3d-rcnn: Instance-level 3d object reconstruction via render-and-compare," in *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pp. 3559–3568, 2018.

[20] J. Tremblay, T. To, B. Sundaralingam, Y. Xiang, D. Fox, and S. Birchfield, "Deep object pose estimation for semantic robotic grasping of household objects," in *Conference on Robot Learning (CoRL)*, pp. 306–316, Oct. 2018.

[21] S. Höfer, K. Bekris, A. Handa, J. C. Gamboa, M. Mozifian, F. Golemo, C. Atkeson, D. Fox, K. Goldberg, J. Leonard, C. Karen Liu, J. Peters, S. Song, P. Welinder, and M. White, "Sim2Real in Robotics and Automation: Applications and Challenges," *IEEE Trans. Autom. Sci. Eng.*, vol. 18, pp. 398–400, Apr. 2021.

[22] X. Li, R. Cao, Y. Feng, K. Chen, B. Yang, C.-W. Fu, Y. Li, Q. Dou, Y.-H. Liu, and P.-A. Heng, "A Sim-to-Real Object Recognition and Localization Framework for Industrial Robotic Bin Picking," *IEEE Robotics and Automation Letters*, vol. 7, pp. 3961–3968, Apr. 2022.

[23] 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)* (A. Vedaldi, H. Bischof, T. Brox, and J.-M. Frahm, eds.), pp. 108–125, Nov. 2020.

[24] G. Wang, F. Manhardt, X. Liu, X. Ji, and F. Tombari, "Occlusion-Aware Self-Supervised Monocular 6D Object Pose Estimation," *IEEE Trans. Pattern Anal. Machine Intell.*, pp. 1–1, 2022.

[25] S. Zakharov, W. Kehl, A. Bhargava, and A. Gaidon, "Autolabeling 3D Objects with Differentiable Rendering of SDF Shape Priors," in *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pp. 12224–12233, Jun. 2020.

[26] X. Deng, Y. Xiang, A. Mousavian, C. Eppner, T. Bretl, and D. Fox, "Self-supervised 6D Object Pose Estimation for Robot Manipulation," in *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, pp. 3665–3671, May 2020.

[27] K. Chen, R. Cao, S. James, Y. Li, Y.-H. Liu, P. Abbeel, and Q. Dou, "Sim-to-Real 6D Object Pose Estimation via Iterative Self-training for Robotic Bin-picking," *arXiv:2204.07049 [cs]*, Apr. 2022.

[28] H. Yang, W. Dong, L. Carlone, and V. Koltun, "Self-supervised geometric perception," in *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2021. Oral Presentation, arXiv preprint: 2103.03114, (pdf), (code).

[29] S. Gould, R. Hartley, and D. Campbell, "Deep declarative networks: A new hope," *arXiv preprint arXiv:1909.04866*, 2019.

[30] A. Agrawal, B. Amos, S. Barratt, S. Boyd, S. Diamond, and J. Z. Kolter, "Differentiable convex optimization layers," in *Advances in Neural Information Processing Systems (NIPS)*, pp. 9558–9570, 2019.

[31] A. X. Chang, T. Funkhouser, L. Guibas, P. Hanrahan, Q. Huang, Z. Li, S. Savarese, M. Savva, S. Song, H. Su, *et al.*, "Shapenet: An information-rich 3d model repository," *arXiv preprint arXiv:1512.03012*, 2015.

[32] B. Calli, A. Singh, A. Walsman, S. Srinivasa, P. Abbeel, and A. M. Dollar, "The YCB object and Model set: Towards common benchmarks for manipulation research," in *Intl. Conf. on Advanced Robotics (ICAR)*, pp. 510–517, Jul. 2015.

TABLE VIII: ϵ_{OC} for ShapeNet (in % of object diameter d) and YCB objects (in cm).

object	ϵ_{OC}	object	ϵ_{OC}	object	ϵ_{OC}	object	ϵ_{OC}	object	ϵ_{OC}	object	ϵ_{OC}
airplane	3.16	bathtub	4.47	bed	4.47	bottle	10.03	cap	10.03	car	3.16
chair	3.16	guitar	2.24	helmet	4.47	knife	2.24	laptop	7.08	motorcycle	3.16
mug	7.08	skateboard	3.16	table	10.03	vessel	3.16				

object	ϵ_{OC}	object	ϵ_{OC}	object	ϵ_{OC}	object	ϵ_{OC}	object	ϵ_{OC}
chips can	1.00	master chef can	1.00	cracker box	1.41	sugar box	1.00	tomato soup can	0.71
mustard bottle	1.00	tuna fish can	0.77	pudding box	0.71	gelatin box	0.71	potted meat can	1.00
banana	0.89	pitcher base	1.58	bleach cleanser	1.14	power drill	0.89	wood block	1.00
scissors	0.95	large marker	1.00	large clamp	0.71	extra large clamp	0.89	foam brick	0.71

[33] T. Hodaň, F. Michel, E. Brachmann, W. Kehl, A. G. Buch, D. Kraft, B. Drost, J. Vidal, S. Ihrke, X. Zabulis, C. Sahin, F. Manhardt, F. Tombari, T.-K. Kim, J. Matas, and C. Rother, "BOP: Benchmark for 6D Object Pose Estimation," in *European Conf. on Computer Vision (ECCV)*, pp. 19–35, 2018.

[34] 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.

[35] B. Calli, A. Walsman, A. Singh, S. Srinivasa, P. Abbeel, and A. M. Dollar, "Benchmarking in Manipulation Research: Using the Yale-CMU-Berkeley Object and Model Set," *IEEE Robotics & Automation Magazine*, vol. 22, pp. 36–52, Sep. 2015.

[36] T. Hodan, P. Haluza, S. Obdrzalek, J. Matas, M. Lourakis, and X. Zabulis, "T-LESS: An RGB-D Dataset for 6D Pose Estimation of Texture-less Objects," in *IEEE Winter Conf. Appl. Computer Vision (WACV)*, Mar. 2017.

[37] 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)*, pp. 16611–16621, 2021.

[38] Y. Zhou, C. Barnes, J. Lu, J. Yang, and H. Li, "On the Continuity of Rotation Representations in Neural Networks," in *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pp. 5738–5746, Jin. 2019.

[39] V. N. Nguyen, Y. Hu, Y. Xiao, M. Salzmann, and V. Lepetit, "Templates for 3d object pose estimation revisited: Generalization to new objects and robustness to occlusions," in *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, Jun. 2022.

[40] 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)*, pp. 683–698, 2018.

[41] 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.

[42] 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.

[43] R. Rusu, N. Blodow, and M. Beetz, "Fast point feature histograms (fpfh) for 3d registration," in *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, pp. 3212–3217, Citeseer, 2009.

[44] Z. Yew and G. Lee, "3dfeat-net: Weakly supervised local 3d features for point cloud registration," in *European Conf. on Computer Vision (ECCV)*, 2018.

[45] H. Deng, T. Birdal, and S. Ilic, "PPFNet: Global Context Aware Local Features for Robust 3D Point Matching," in *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pp. 195–205, Jun. 2018.

[46] H. Deng, T. Birdal, and S. Ilic, "PPF-FoldNet: Unsupervised Learning of Rotation Invariant 3D Local Descriptors," in *European Conf. on Computer Vision (ECCV)* (V. Ferrari, M. Hebert, C. Sminchisescu, and Y. Weiss, eds.), pp. 620–638, 2018.

[47] C. Choy, J. Park, and V. Koltun, "Fully convolutional geometric features," in *Intl. Conf. on Computer Vision (ICCV)*, pp. 8958–8966, 2019.

[48] A. Zeng, S. Song, M. Nießner, M. Fisher, J. Xiao, and T. Funkhouser, "3dmatch: Learning the matching of local 3d geometry in range scans," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, vol. 1, p. 4, 2017.

[49] J. Yang, H. Li, D. Campbell, and Y. Jia, "Go-ICP: A globally optimal solution to 3D ICP point-set registration," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 38, pp. 2241–2254, Nov. 2016.

[50] Q. Zhou, J. Park, and V. Koltun, "Fast global registration," in *European Conf. on Computer Vision (ECCV)*, pp. 766–782, Springer, 2016.

[51] V. Sarode, X. Li, H. Goforth, Y. Aoki, R. A. Srivatsan, S. Lucey, and H. Choset, "PCRNet: Point Cloud Registration Network using PointNet Encoding," *arXiv:1908.07906 [cs]*, Nov. 2019.

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

[53] C. Choy, W. Dong, and V. Koltun, "Deep global registration," in *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2020.

[54] S. Huang, Z. Gojcic, M. Usvyatsov, A. Wieser, and K. Schindler, "PREDATOR: Registration of 3D Point Clouds with Low Overlap," in *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pp. 4265–4274, Jun. 2021.

[55] Z. J. Yew and G. H. Lee, "RPM-Net: Robust Point Matching using Learned Features," in *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pp. 11821–11830, Jun. 2020.

[56] Z. Gojcic, C. Zhou, J. D. Wegner, and A. Wieser, "The perfect match: 3d point cloud matching with smoothed densities," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 5545–5554, 2019.

[57] Y. Yuan, D. Borrmann, J. Hou, Y. Ma, A. Nüchter, and S. Schwertfeger, "Self-supervised point set local descriptors for point cloud registration," *Sensors*, vol. 21, no. 2, 2021.

[58] M. E. Banani, L. Gao, and J. Johnson, "UnsupervisedR&R: Unsupervised Point Cloud Registration via Differentiable Rendering," in *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pp. 7125–7135, Jun. 2021.

[59] X. Huang, G. Mei, and J. Zhang, "Feature-Metric Registration: A Fast Semi-Supervised Approach for Robust Point Cloud Registration Without Correspondences," in *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pp. 11363–11371, Jun. 2020.

[60] M. Zhu, M. Ghaffari, and H. Peng, "Correspondence-Free Point Cloud Registration with SO(3)-Equivariant Implicit Shape Representations," in *Conference on Robot Learning (CoRL)*, pp. 1412–1422, Jan. 2022.

[61] W. Sun, A. Tagliasacchi, B. Deng, S. Sabour, S. Yazdani, G. E. Hinton, and K. M. Yi, "Canonical Capsules: Self-Supervised Capsules in Canonical Pose," in *Advances in Neural Information Processing Systems (NIPS)*, vol. 34, pp. 24993–25005, 2021.

[62] S. Yang, Z. Quan, M. Nie, and W. Yang, "TransPose: Keypoint Localization via Transformer," in *Intl. Conf. on Computer Vision (ICCV)*, pp. 11782–11792, Oct. 2021.

[63] W. Liu, Q. Bao, Y. Sun, and T. Mei, "Recent Advances of Monocular 2D and 3D Human Pose Estimation: A Deep Learning Perspective," *ACM Computing Surveys*, vol. 55, pp. 80:1–80:41, Nov. 2022.

[64] G. Pavlakos, L. Zhu, X. Zhou, and K. Daniilidis, "Learning to estimate 3d human pose and shape from a single color image," *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pp. 459–468, 2018.

[65] 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.

[66] K. Schmeckpeper, P. Osteen, Y. Wang, G. Pavlakos, K. Chaney, W. Jordan, X. Zhou, K. Derpanis, and K. Daniilidis, "Semantic keypoint-based pose estimation from single rgb frames," *arXiv preprint arXiv:2204.05864*, 2022.

[67] S. Lin, Z. Wang, Y. Ling, Y.-D. Tao, and C. Yang, "E2EK: End-to-End Regression Network Based on Keypoint for 6D Pose Estimation," *IEEE Robotics and Automation Letters*, 2022.

[68] J. Shi, H. Yang, and L. Carlone, "Optimal pose and shape estimation for

category-level 3D object perception," in *Robotics: Science and Systems (RSS)*, 2021. arXiv preprint: 2104.08383, (pdf), (video).

[69] X. Zhou, A. Karpur, L. Luo, and Q. Huang, "Starmap for category-agnostic keypoint and viewpoint estimation," in *European Conf. on Computer Vision (ECCV)*, pp. 328–345, 2018.

[70] M. Vecerik, J.-B. Regli, O. Sushkov, D. Barker, R. Pevceviciute, T. Rothörl, R. Hadsell, L. Agapito, and J. Scholz, "S3K: Self-Supervised Semantic Keypoints for Robotic Manipulation via Multi-View Consistency," in *Conference on Robot Learning (CoRL)*, pp. 449–460, Oct. 2021.

[71] Y. You, Y. Lou, C. Li, Z. Cheng, L. Li, L. Ma, W. Wang, and C. Lu, "KeypointNet: A Large-scale 3D Keypoint Dataset Aggregated from Numerous Human Annotations," in *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pp. 13647–13656, Jun. 2020.

[72] S. Suwajanakorn, N. Snavely, J. J. Tompson, and M. Norouzi, "Discovery of Latent 3D Keypoints via End-to-end Geometric Reasoning," in *Advances in Neural Information Processing Systems (NIPS)*, vol. 31, Dec. 2018.

[73] 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)*, pp. 2642–2651, 2019.

[74] A. Bandeira, "A note on probably certifiably correct algorithms," *arXiv:1509.00824*, 2015.

[75] H. Papadopoulos, K. Proedrou, V. Vovk, and A. Gammerman, "Inductive Confidence Machines for Regression," in *European Conf. on Machine Learning* (T. Elomaa, H. Mannila, and H. Toivonen, eds.), (Berlin, Heidelberg), pp. 345–356, Springer, 2002.

[76] J. Lei and L. Wasserman, "Distribution-free prediction bands for non-parametric regression," *J. Royal Stat. Soc.: Series B (Stat. Methodology)*, vol. 76, pp. 71–96, Jan. 2014.

[77] R. J. Tibshirani, R. Foygel Barber, E. Candes, and A. Ramdas, "Conformal Prediction Under Covariate Shift," in *Advances in Neural Information Processing Systems (NIPS)*, vol. 32, Nov. 2019.

[78] C. Fannjiang, S. Bates, A. N. Angelopoulos, J. Listgarten, and M. I. Jordan, "Conformal prediction under feedback covariate shift for biomolecular design," *Proceedings of the National Academy of Sciences*, vol. 119, Oct. 2022.

[79] B. Amos and J. Z. Kolter, "Optnet: Differentiable optimization as a layer in neural networks," in *Intl. Conf. on Machine Learning (ICML)*, pp. 136–145, JMLR. org, 2017.

[80] S. Gould, R. Hartley, and D. Campbell, "Deep Declarative Networks," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 44, pp. 3988–4004, Aug. 2022.

[81] M. V. Pogančić, A. Paulus, V. Musil, G. Martius, and M. Rolinek, "Differentiation of Blackbox Combinatorial Solvers," in *Intl. Conf. on Learning Representations (ICLR)*, Mar. 2020.

[82] A. Paulus, M. Rolinek, V. Musil, B. Amos, and G. Martius, "CombOptNet: Fit the Right NP-Hard Problem by Learning Integer Programming Constraints," in *Intl. Conf. on Machine Learning (ICML)*, pp. 8443–8453, Jul. 2021.

[83] P.-W. Wang, P. Donti, B. Wilder, and Z. Kolter, "SATNet: Bridging deep learning and logical reasoning using a differentiable satisfiability solver," in *Proceedings of the 36th International Conference on Machine Learning*, pp. 6545–6554, May 2019.

[84] P. Donti, B. Amos, and J. Z. Kolter, "Task-based End-to-end Model Learning in Stochastic Optimization," in *Advances in Neural Information Processing Systems*, vol. 30, Nov. 2017.

[85] Z. Teed and J. Deng, "DROID-SLAM: Deep visual SLAM for monocular, stereo, and RGB-d cameras," in *Advances in Neural Information Processing Systems (NIPS)* (A. Beygelzimer, Y. Dauphin, P. Liang, and J. W. Vaughan, eds.), 2021.

[86] S. Jiang, D. Campbell, M. Liu, S. Gould, and R. Hartley, "Joint Unsupervised Learning of Optical Flow and Egomotion with Bi-Level optimization," in *Int. Conf. 3D Vision*, pp. 682–691, Nov. 2020.

[87] D. Campbell, L. Liu, and S. Gould, "Solving the blind perspective-n-point problem end-to-end with robust differentiable geometric optimization," in *European Conf. on Computer Vision (ECCV)*, p. 244–261, Aug. 2020.

[88] C. R. Qi, L. Yi, H. Su, and L. J. Guibas, "Pointnet++: Deep hierarchical feature learning on point sets in a metric space," in *Advances in neural information processing systems*, pp. 5099–5108, 2017.

[89] H. Zhao, L. Jiang, J. Jia, P. Torr, and V. Koltun, "Point Transformer," *arXiv:2012.09164 [cs]*, Dec. 2020.

[90] A. R. Conn, N. I. M. Gould, and P. L. Toint, *Trust Region Methods*. Society for Industrial and Applied Mathematics, 2000.

[91] P. Virtanen, R. Gommers, T. E. Oliphant, M. Haberland, T. Reddy, D. Cournapeau, E. Burovski, P. Peterson, W. Weckesser, J. Bright, S. J. van der Walt, M. Brett, J. Wilson, K. J. Millman, N. Mayorov, A. R. J. Nelson, E. Jones, R. Kern, E. Larson, C. J. Carey, İ. Polat, Y. Feng, E. W. Moore, J. VanderPlas, D. Laxalde, J. Perktold, R. Cimrman, I. Henriksen, E. A. Quintero, C. R. Harris, A. M. Archibald, A. H. Ribeiro, F. Pedregosa, P. van Mulbregt, and SciPy 1.0 Contributors, "SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python," *Nature Methods*, vol. 17, pp. 261–272, 2020.

[92] B. k P Horn and E. J. Weldon, "Direct methods for recovering motion," *Intl. J. of Computer Vision*, vol. 1, no. 2, pp. 51–76, 1988.

[93] K. Arun, T. Huang, and S. Blostein, "Least-squares fitting of two 3-D point sets," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 9, pp. 698–700, sept. 1987.

[94] G. Wahba, "A least squares estimate of satellite attitude," *SIAM review*, vol. 7, no. 3, pp. 409–409, 1965.

[95] Q.-Y. Zhou, J. Park, and V. Koltun, "Open3D: A modern library for 3D data processing," *arXiv:1801.09847*, 2018.

[96] R. Hartley, J. Trumpf, Y. Dai, and H. Li, "Rotation averaging," *IJCV*, vol. 103, no. 3, pp. 267–305, 2013.

[97] L. Carlone, R. Tron, K. Daniilidis, and F. Dellaert, "Initialization techniques for 3D SLAM: a survey on rotation estimation and its use in pose graph optimization. supplemental material," 2015.

Luca Carlone is the Leonardo Career Development Associate Professor in the Department of Aeronautics and Astronautics at the Massachusetts Institute of Technology, and a Principal Investigator in the Laboratory for Information & Decision Systems (LIDS). He has obtained a B.S. degree in mechatronics from the Polytechnic University of Turin, Italy, in 2006; an S.M. degree in mechatronics from the Polytechnic University of Turin, Italy, in 2008; an S.M. degree in automation engineering from the Polytechnic University of Milan, Italy, in 2008; and a Ph.D. degree in robotics also from the Polytechnic University of Turin in 2012. He joined LIDS as a postdoctoral associate (2015) and later as a Research Scientist (2016), after spending two years as a postdoctoral fellow at the Georgia Institute of Technology (2013-2015). His research interests include nonlinear estimation, numerical and distributed optimization, and probabilistic inference, applied to sensing, perception, and decision-making in single and multi-robot systems. His work includes seminal results on certifiably correct algorithms for localization and mapping, as well as approaches for visual-inertial navigation and distributed mapping. He is a recipient of the Best Student Paper Award at IROS 2021, the Best Paper Award in Robot Vision at ICRA 2020, a 2020 Honorable Mention from the IEEE Robotics and Automation Letters, a Track Best Paper award at the 2021 IEEE Aerospace Conference, the 2017 and 2022 Transactions on Robotics King-Sun Fu Memorial Best Paper Award, the Best Paper Award at WAFR 2016, the Best Student Paper Award at the 2018 Symposium on VLSI Circuits, and he was best paper finalist at RSS 2015, RSS 2021, and WACV 2023. He is also a recipient of the AIAA Aeronautics and Astronautics Advising Award (2022), the NSF CAREER Award (2021), the RSS Early Career Award (2020), the Google Daydream Award (2019), the Amazon Research Award (2020, 2022), and the MIT AeroAstro Vickie Kerrebrock Faculty Award (2020). He is an IEEE senior member and an AIAA associate fellow.