Abstract—In this article, we consider a category-level perception problem, where one is given 2-D or 3-D sensor data picturing an object of a given category (e.g., a car) and has to reconstruct the 3-D pose and shape of the object despite intraclass variability (i.e., different car models have different shapes). We consider an active shape model, where—for an object category—we are given a library of potential computer-aided design models describing objects in that category, and we adopt a standard formulation where pose and shape are estimated from 2-D or 3-D keypoints via nonconvex optimization. Our first contribution is to develop PACE3D $^\star$ and PACE2D $^\star$, the first certifiably optimal solvers for pose and shape estimation using 3-D and 2-D keypoints, respectively. Both the solvers rely on the design of tight (i.e., exact) semidefinite relaxations. Our second contribution is to develop outlier-robust versions of both the solvers, named PACE3D $\#$ and PACE2D $\#$. Toward this goal, we propose ROBIN (Reject Outliers Based on INvariants), a general graph-theoretic framework to prune outliers, which uses compatibility hypergraphs to model measurements’ compatibility. We show that in category-level perception problems, these hypergraphs can be built from the winding orders of the keypoints (in 2-D) or their convex hulls (in 3-D), and many outliers can be filtered out via maximum hyperclicque computation. The last contribution is an extensive experimental evaluation. Besides providing an ablation study on simulated datasets and on the PASCAL3D+ dataset, we combine our solver with a deep keypoint detector and show that PACE3D $\#$ improves over the state of the art in vehicle pose estimation in the ApolloScape datasets, and its runtime is compatible with practical applications. We release our code at https://github.com/MIT-SPARK/PACE.

Index Terms—Category-level object perception, certifiable algorithms, outlier-robust estimation, RGB-D perception.

Manuscript received 28 March 2023; accepted 9 May 2023. Date of publication 31 August 2023; date of current version 4 October 2023. This work was supported in part by the Army Research Laboratory’s Distributed and Collaborative Intelligent Systems and Technology Collaborative Research Alliance under Grant W911NF-17-2-0181, in part by the Office of Naval Research’s RAIDER program under Grant N00014-18-1-2828, in part by Lincoln Laboratory’s “Resilient Perception in Degraded Environments” program, and in part by an Amazon Research Award. The work of Luca Carlone was also supported by the National Science Foundation’s Faculty Early Career Development Program. This paper was recommended for publication by Associate Editor J. Civera and Editor S. Behnke upon evaluation of the reviewers’ comments. (Corresponding author: Jingnan Shi.)

Jingnan Shi and Luca Carlone are with the Laboratory for Information and Decision Systems, Massachusetts Institute of Technology, Cambridge, MA 02139 USA (e-mail: jnshi@mit.edu; lcarlone@mit.edu).

Heng Yang is with the School of Engineering and Applied Sciences, Harvard University, Cambridge, MA 02139 USA (e-mail: hanyang@seas.harvard.edu).

Color versions of one or more figures in this article are available at https://doi.org/10.1109/TRO.2023.3277273.

Digital Object Identifier 10.1109/TRO.2023.3277273.

Optimal and Robust Category-Level Perception: Object Pose and Shape Estimation From 2-D and 3-D Semantic Keypoints

Jingnan Shi $^\diamond$, Member, IEEE, Heng Yang $^\diamond$, and Luca Carlone $^\circ$, Senior Member, IEEE

( Evolved Paper)
an object belonging to a given category (e.g., detections of the wheels, rear-view mirrors, and other interest points of a car) and has to reconstruct the pose and shape of the object. We assume the availability of a library of CAD models of objects in that category; such a library is typically available, since CAD models are extensively used in the design, manufacturing, and simulation of 3-D objects. Note that the definition of category-level perception is ambiguous in the literature: we follow the setup in [1], [2], [22], [23], [24], [25], while some authors use the same name for problems where the objects seen at test time are different from the ones seen during training [26], while being in the same category.

Our first contribution is to develop the first certifiably optimal solvers for pose and shape estimation using 3-D and 2-D keypoints. In the 3-D case, we show that—despite the nonconvexity of the problem—rotation estimation can be decoupled from the estimation of object translation and shape, and we demonstrate that 1) the optimal object rotation can be computed via a tight (small-size) semidefinite relaxation and 2) the translation and shape parameters can be computed in closed form given the rotation. We call the resulting solver PACE3 Đ∗ (Pose and shApes estimation for 3-D–3-D Category-level pErception). In the 2-D case, we formulate pose and shape estimation using an algebraic point-to-line cost, and leverage Lasserre’s hierarchy of semidefinite relaxations [27] to solve the problem to certifiable global optimality. We call the resulting solver PACE2 Đ∗ (Pose and shApes estimation for 2-D–3-D Category-level pErception). Contrarily to PACE3 Đ∗, PACE2 Đ∗ leads to semidefinite relaxations whose size increases with the number of CAD models in the active shape model.

Our second contribution is to develop an outlier rejection scheme applicable to both PACE3 Đ∗ and PACE2 Đ∗. Toward this goal, we introduce a general framework for graph-theoretic outlier pruning, named ROBIN(Reject Outliers Based on INvariants), which generalizes our previous work [7], [28] to use hypergraphs. ROBIN models compatibility between subsets of measurements using a compatibility hypergraph. We show that the compatibility hypergraph can be efficiently constructed by inspecting the winding orders of the keypoints in 2-D or the convex hulls of the keypoints in 3-D. We then prove that all the inliers are contained in a single hyperclicque of the compatibility hypergraph and can be typically found within the maximum hyperclicque. ROBIN is able to remove a large fraction of outliers. The resulting measurements are then passed to our optimal solvers (PACE3 Đ∗; see Section VI-A) wrapped in a graduated nonconvexity (GNC) scheme (see Section VII), which estimates the object pose and shape coefficients. (b) PACE2 Đ# performs 2-D–3-D category-level perception and works on RGB inputs. The images are passed through a neural network to obtain 2-D keypoints. We then use ROBIN with 2-D–3-D compatibility hypergraphs (see Section V) to prune outliers. We finally pass the resulting measurements to our optimal solver (PACE2 Đ∗; see Section VI-B) with GNC to obtain a pose and shape estimate.

Our last contribution is an extensive experimental evaluation in both synthetic experiments and real datasets [30]. We provide an ablation study on simulated datasets and on the PASCAL3D+ dataset and show that: 1) PACE3 Đ∗ is more accurate than state-of-the-art iterative solvers; 2) PACE2 Đ∗ is more accurate than baseline local solvers and convex relaxations based on the weak perspective projection model [14], [15]; 3) PACE3 Đ# dominates other robust solvers and is robust to 70–90% outliers; and 4) PACE2 Đ# is robust to 10% outliers. Finally, we integrate our solvers in a realistic system—including a deep keypoint detector—and apply it to vehicle pose and shape estimation in the ApolloScape [30] driving datasets. While PACE2 Đ# is currently slow and suffers from the low quality of the deep keypoint detections, PACE3 Đ# largely outperforms the state of the art and a nonoptimized implementation runs in a fraction of a second. We also show that ROBIN is even able to detect mislabeled keypoints used to train the keypoint detector in the ApolloScape dataset [30].

Novelty with Respect to [28] and [31]: In our previous works, we introduced ROBIN [28], a graph-theoretic outlier rejection framework, and two solvers [31] (PACE3 Đ∗ and PACE3 Đ#) for pose and shape estimation from 3-D keypoints. The present article 1) allows ROBIN to handle more general compatibility tests and 2) extends the concept of inlier selection to maximum hyperclicques on compatibility hypergraphs. This article also develops PACE2 Đ∗ and PACE2 Đ#, which estimate pose and shape from only 2-D (instead of 3-D) keypoints. In addition, we report a more comprehensive experimental evaluation in simulation and on the ApolloScape dataset.

Article Structure: Section II formulates the category-level perception problem. Section III provides a brief overview of the proposed approaches (also summarized in Fig. 1). Sections IV and V present our graph-theoretic outlier pruning (ROBIN)
and its application to category-level perception. Section VI introduces our certifiably optimal solvers for category-level perception, and Section VII recalls how to wrap the solvers in a GNC scheme. Section VIII discusses experimental results. Section IX provides an in-depth review of related work. Finally, Section X concludes this article.

II. PROBLEM STATEMENT: 3-D–3-D AND 2-D–3-D CATEGORY-LEVEL PERCEPTION

In this section, we formulate the 3-D–3-D and 2-D–3-D category-level perception problems. The goal is to compute the 3-D pose and shape of an object, given 3-D or 2-D sensor data. We focus on a multistage setup (cf. Section IX), where a front end is used to extract 2-D or 3-D semantic keypoints from the sensor data, which are then used by a back-end solver to estimate the object’s 3-D pose \((\mathbf{R}, \mathbf{t})\) and shape, where \(\mathbf{R} \in \mathbb{SO}(3)\) and \(\mathbf{t} \in \mathbb{R}^3\) are the 3-D rotation and translation, respectively. The front end is typically implemented using standard deep networks [3], while our goal here is to design more accurate and robust back ends.

In the following, we first introduce a standard parameterization of the object shape (see Section II-A) and then formalize the 3-D–3-D category-level perception problem (see Section II-B) and its 2-D–3-D counterpart (see Section II-C).

A. Active Shape Model

We assume the object shape to be partially specified: we are given a library of 3-D CAD models \(B_k\), \(k = 1, \ldots, K\), and assume that the unknown object shape \(S\) (modeled as a collection of 3-D points) can be written as a combination of the points on the given CAD models. More formally, each point \(s_i\) of the shape \(S\) can be written as

\[
 s_i = \sum_{k=1}^{K} c_k b_i^k
\]

where \(b_i^k\) is a given point on the surface of the CAD model \(B_k\); the shape parameters \(c \equiv [c_1 \ldots c_K]^T\) are unknown, and the entries of \(c\) are assumed to be nonnegative and sum up to 1, i.e., \(c\) belongs to the \(K\)-dimensional probability simplex \(\Delta_K := \{c \in \mathbb{R}^K | c \geq 0, \sum_{k=1}^{K} c_k = 1\}\). For instance, if—upon estimation—the vector \(c\) has the \(l\)th entry equal to 1 and the remaining entries equal to zero in (1), then the estimated shape matches the \(l\)th CAD model in the library; therefore, the estimation of the shape parameters \(c\) can be understood as a fine-grained classification of the object among the instances in the library. However, the model is even more expressive, since it allows the object shape to be a convex combination of CAD models, which enables the active shape model (1) to interpolate between different shapes in the library (see Fig. 2).

B. 3-D–3-D Category-Level Perception

In the 3-D–3-D category-level perception problem, the goal is to estimate an object’s pose and shape, given a set of 3-D keypoint detections, typically obtained using learning-based keypoint detectors. Such detectors are trained to detect semantic features of the 3-D object (e.g., wheels of a car) and can be applied to RGB-D or RGB+LiDAR data (e.g., [3]).

We assume that each 3-D measurement \(p_{3D,i}\) \((i = 1, \ldots, N)\) is a noisy measurement of a keypoint \(s_i\) of our target object in the coordinate frame of the sensor. More formally, each \(p_{3D,i}\) is described by the following generative model:

\[
 p_{3D,i} = R \sum_{k=1}^{K} c_k b_i^k + t + \epsilon_{3D,i}, \quad i = 1, \ldots, N \tag{2}
\]

where the measurement \(p_{3D,i}\) pictures a 3-D point on the object [written as a linear combination \(\sum_{k=1}^{K} c_k b_i^k\) of the shapes in the library as in (1)], after the point is rotated and translated according to the 3-D pose \((\mathbf{R}, \mathbf{t})\) of the object, and where \(\epsilon_{3D,i}\) represents measurement noise. Intuitively, each measurement corresponds to a noisy measurement of a semantic feature of the object (e.g., wheel center or rear-view mirrors of a car), and each \(b_i^k\) corresponds to the corresponding feature (e.g., wheel or mirror) location for a specific CAD model.

Problem 1 (Robust 3-D–3-D Category-Level Perception): Compute the 3-D pose \((\mathbf{R}, \mathbf{t})\) and shape \((c)\) of an object given \(N\) 3-D keypoint measurements in the form \(p_{3D,i}\) (2), possibly corrupted by outliers, i.e., measurements with large error \(\epsilon_{3D,i}\).

C. 2-D–3-D Category-Level Perception

In the 2-D–3-D category-level perception problem, we want to estimate an object’s 3-D pose and shape, given only 2-D projections of keypoints. In this case, we describe each 2-D measurement using the following generative model:

\[
 p_{2D,i} = \pi \left( R \sum_{k=1}^{K} c_k b_i^k + t \right) + \epsilon_{2D,i}, \quad i = 1, \ldots, N \tag{3}
\]

where \(p_{2D,i}\) represents a 2-D (pixel) measurement, \(\pi(\cdot)\) is the canonical perspective projection, \(\epsilon_{2D,i}\) is the measurement noise. Intuitively, the measurements in (3) correspond to pixel projections of the object keypoints onto an image.

Problem 2 (Robust 2-D–3-D Category-Level Perception): Compute the 3-D pose \((\mathbf{R}, \mathbf{t})\) and shape \((c)\) of an object given \(N\) 2-D keypoint measurements in the form (3), possibly corrupted by outliers, i.e., measurements with large error \(\epsilon_{2D,i}\).

\[1\text{For a 3-D vector } p = \begin{bmatrix} p_x \\ p_y \\ p_z \end{bmatrix}, \text{ the canonical projection is } \pi(p) = \begin{bmatrix} p_x/p_z \\ p_y/p_z \end{bmatrix}.\]
III. OVERVIEW OF PACE#: POSE AND SHAPE ESTIMATION FOR ROBUST CATEGORY-LEVEL PERCEPTION

Our approach, named PACE#, is summarized in Fig. 1, for both the 3-D–3-D case [PACE3D#, Fig. 1(a)] and the 2-D–3-D case [PACE2D#, Fig. 1(b)]. We assume access to a perception front end that detects semantic keypoints given sensor data. Our work forms the back end and consists of two stages. In the first stage, we employ a graph-theoretic framework, named ROBIN, to preprocess the keypoints and prune gross outliers without explicitly solving the underlying estimation problem. We then pass the filtered measurements to the second stage, where an optimal solver (wrapped in a GNC scheme) computes a pose and shape estimate.

In the following, we introduce Stage 1 by first presenting ROBIN, a general framework for outlier pruning (see Section IV) and then discussing its application to category-level perception (see Section V); we then discuss Stage 2 by presenting our optimal solvers for 3-D–3-D (PACE3D*; see Section VI-A) and 2-D–3-D (PACE2D*; see Section VI-B) category-level perception, and a brief review of GNC [29] (see Section VII).

IV. STAGE 1: GRAPH-THEORETIC OUTLIER PRUNING WITH ROBIN

This section develops a general framework to prune gross outliers from a set of measurements without explicitly computing an estimate for the variables of interest. In particular, we introduce the notion of n-invariant to check if a subset of measurements contains outliers. We then use these checks to construct compatibility hypergraphs that describe mutually compatible measurements, and show how to reject outliers by computing maximum hypercliques of these graphs. Combining these insights, we obtain ROBIN, our graph-theoretic algorithm for pruning outliers.

This section presents our framework in full generality, and then, we tailor it to category-level perception in Section V. In particular, here, we consider a more general measurement model that relates measurements \( y_i \) to the to-be-estimated variable \( x \) (where \( X \) is the domain of \( x \), e.g., the set of 3-D poses) and gives a model \( \theta \) (e.g., our CAD models)

\[
y_i = h(x, \theta_i, \epsilon_i), \quad i = 1, \ldots, N
\]

where \( \epsilon_i \) denotes the measurement noise. Clearly, (2) and (3) can be understood as special instances of (4), where \( x \) includes the unknown pose and shape of the object, and the (given) model \( \theta \) corresponds to the CAD models.

A. FROM MEASUREMENTS TO INVARIANTS

This section formalizes the concepts of n-invariant and generalized n-invariant, which are the building blocks of our outlier pruning framework. The main motivation is to use invariance to establish checks on the (inlier) measurements that hold true regardless of the state under estimation; we are later going to use these checks to detect outliers.

Let us consider the measurements in (4) and denote the indices of the measurements as \( \mathcal{Y} = \{1, \ldots, N\} \). For a given integer \( n \leq N \), let \( \mathcal{M} \subset \mathcal{Y} \) be a subset of \( n \) indices in \( \mathcal{Y} \) and denote with \( \mathcal{M}_j \) the \( j \)th element of this subset (with \( j = 1, \ldots, n \)).

\[
y_M = [y_{M_1}, y_{M_2}, \ldots, y_{M_n}], \quad \theta_M = [\theta_{M_1}, \theta_{M_2}, \ldots, \theta_{M_n}], \quad \epsilon_M = [\epsilon_{M_1}, \epsilon_{M_2}, \ldots, \epsilon_{M_n}]
\]

which is simply stacking together measurements \( y_i \), parameters \( \theta_i \), and noise \( \epsilon_i \) for the subset of measurements \( i \in \mathcal{M} \).

Let us now formalize the notion of noiseless invariance.

**Definition 1 (Noiseless n-invariant):** Consider the generative model (4) and assume that there is no measurement noise (i.e., \( y_i = h(x, \theta_i) \)). Then, a function \( f \) is called a noiseless n-invariant if for any arbitrary \( \mathcal{M} \subset \mathcal{Y} \) of size \( n \), the following relation holds, regardless of the choice of \( x \):

\[
f(y_M) = f(\theta_M).
\]

Equation (7) is also known as the point cloud registration problem [8], [33] and consists in finding a rigid body transformation \((R, t)\) (where \( R \in SO(3) \) and \( t \in \mathbb{R}^3 \)) that aligns two sets of 3-D points \( y_i \in \mathbb{R}^3 \) and \( \theta_i \in \mathbb{R}^3 \), with \( i = 1, \ldots, N \). The corresponding measurement model can again be seen to be an instance of the general model (4).

In the noiseless case \( (\epsilon_i = 0) \), it follows from (7) that

\[
\|y_j - y_i\| = \|R(\theta_j + t) - (R\theta_i + t)\| = \|R(\theta_j - \theta_i)\| = \|\theta_j - \theta_i\|
\]

for any pair of measurements \( i, j \), where \( \| \cdot \| \) is the 2-norm, and we used the fact that the 2-norm is invariant under rotation. This is an example of noiseless 2-invariant, \( f(y_j, y_i) \neq f(\theta_j, \theta_i) \), which relates measurements and model regardless of the choice of \( x \), and formalizes the intuition that the distance between pairs of points in a point cloud is invariant under rigid transformations.

While Definition 1 provides a general definition of noiseless invariance, practical problems always involve noise. Therefore, we need to generalize the notion of invariance as follows.

**Definition 2 (Generalized n-invariant):** Given (4) and assuming that the measurement noise is bounded \( \|\epsilon_i\| \leq \beta \) \( (i = 1, \ldots, N) \), a pair of functions \((f, F)\) is called a generalized n-invariant if for any arbitrary \( \mathcal{M} \subset \mathcal{Y} \) of size \( n \), the following relation holds regardless of the choice of \( x \):

\[
f(y_M) \in F(\theta_M, \beta)
\]

where \( F(\theta_M, \beta) \) is a set-valued function (independent on \( x \)).

Intuitively, because of the noise, now, the measurements can produce a number of different invariants \( f(y_M) \), but we can still define a set \( F(\theta_M, \beta) \) that contains all potential invariants produced by the measurements. An example is in order.

**Example:** Going back to the example of point cloud registration in (7), with \( \|\epsilon_i\| \leq \beta \) and \( \|\epsilon_j\| \leq \beta \), we have

\[
\|y_j - y_i\| = \|R(\theta_j - \theta_i) + \epsilon_j - \epsilon_i\|.
\]

Authorized licensed use limited to the terms of the applicable license agreement with IEEE. Restrictions apply.
If we apply the triangle inequality to the right-hand side of (13), we obtain
\[ \|\theta_j - \theta_i\| - \|\epsilon_j - \epsilon_i\| \leq \|R(\theta_j - \theta_i) + \epsilon_j - \epsilon_i\| \]
\[ \leq \|\theta_j - \theta_i\| + \|\epsilon_j - \epsilon_i\|. \quad (11) \]
Now, \(\|\epsilon_i\| \leq \beta\) and \(\|\epsilon_j\| \leq \beta\) imply \(\|\epsilon_j - \epsilon_i\| \leq 2\beta\). Substituting into (11), we obtain
\[ \|\theta_j - \theta_i\| - 2\beta \leq \|y_j - y_i\| \leq \|\theta_j - \theta_i\| + 2\beta \] or, in other words,
\[ \frac{\|y_j - y_i\|}{\|\theta_j - \theta_i\| - 2\beta} \leq \frac{\|\theta_j - \theta_i\| + 2\beta}{\|\theta_j - \theta_i\| - 2\beta} \quad (13) \]
which corresponds to our definition of generalized \(n\)-invariant (with \(n = 2\)). Geometrically, (13) states that the distances between pairs of measured points \((y_j, y_i)\) must match corresponding distances between points in our model \((\theta_j, \theta_i)\) up to noise. Note that when \(\beta = 0\) (noiseless case), (13) reduces back to (8), as the set \(\mathcal{F}(\theta_{ij}, \beta) = \{\|\theta_j - \theta_i\| - 2\beta, \|\theta_j - \theta_i\| + 2\beta\}\) reduced to the singleton \(\{\|\theta_j - \theta_i\|\}\).

Definition 2 generalizes Definition 1 to account for the presence of noise. Moreover, as we will see in Section V, we can develop generalized \(n\)-invariants for category-level perception even when noiseless invariants are difficult to find. In other words, while it may be difficult to pinpoint a noiseless invariant function, it is often easier to find a set of values that a suitable function \(f(y_{M_n})\) must belong to. In the following sections, unless otherwise specified, we use the term \(n\)-invariants to refer to generalized \(n\)-invariants.

B. From Invariants to Compatibility Tests for Outlier Pruning

While the previous section developed invariants without distinguishing inliers from outliers, this section shows that the resulting invariants can be directly used to check if a subset of measurements contains an outlier. Toward this goal, we formalize the notion of inlier and outlier.

Definition 3 (Inliers and Outliers): Given measurements (4) and a threshold \(\beta \geq 0\), a measurement \(i\) is an inlier if the corresponding noise satisfies \(\|\epsilon_i\| \leq \beta\) and is an outlier otherwise.

The notion of invariants introduced in the previous section allowed us to obtain relations that depend on the measurements and a noise bound, but are independent on \(x\) [see (9)]. Therefore, we can directly use these relations to check if a subset of measurements contains outliers: if (9) is not satisfied by a subset of measurements \(y_{M_n}\), then the corresponding subset of measurements must contain an outlier. We call the corresponding check a compatibility test. In the following, we provide an example of compatibility test for point cloud registration. The reader can find more examples of compatibility tests for other applications in [28].

Example of compatibility test: For our point cloud registration example, (13) states that any pair of measurements \(y_i\) and \(y_j\) with noise \(\|\epsilon_i\| \leq \beta\) and \(\|\epsilon_j\| \leq \beta\) must satisfy
\[ \|y_j - y_i\| \leq \|\theta_j - \theta_i\| - 2\beta, \|\theta_j - \theta_i\| + 2\beta \]. \quad (14) \]
If the relation is satisfied, we say that \(y_i\) and \(y_j\) are compatible with each other (i.e., they can potentially be both inliers); however, if the relation is not satisfied, then one of the measurements must be an outlier. Generalizing this example, we obtain the following general definition of compatibility test.

Definition 4 (Compatibility test): Given a subset of \(n\) measurements and the corresponding \(n\)-invariant, a compatibility test is a binary condition (computed using the invariant), such that if the condition fails, then the set of measurements must contain at least one outlier.

Note that we require the test to be sound (i.e., it does not detect outliers when testing a set of inliers), but may not be complete (i.e., the test might pass even in the presence of outliers). This property is important since our goal is to prune as many outliers as we can, while preserving the inliers. Also note that the test detects if the set contains outliers, but does not provide information on which measurements are outliers. We are going to fill in this gap below.

C. From Compatibility Tests to Compatibility Hypergraph

For a problem with an \(n\)-invariant, we describe the results of the compatibility tests for all subsets of \(n\) measurements using a compatibility hypergraph.

Definition 5 (Compatibility Hypergraph): Given a compatibility test with \(n\) measurements, define the compatibility hypergraph \(\mathcal{G}(V, E)\) as an \(n\)-uniform undirected hypergraph, where each node \(v\) in the node set \(V\) is associated with a measurement in (4), and an hyperedge \(e\) (connecting a subset of \(n\) measurements) belongs to the edge set \(E\) if and only if its subset of measurements passes the compatibility test.

Note that in the case where \(n = 2\), the above definition reduces to a regular undirected graph. Building the compatibility graph requires looping over all subsets of \(n\) measurements and, whenever the subset passes the compatibility test, adding a hyperedge between the corresponding \(n\) nodes in the graph. Note that these checks are very fast and easy to parallelize since they only involve checking Boolean conditions [e.g., (12)] without computing an estimate (as opposed to RANSAC).

Inlier Structures in Compatibility Hypergraphs: Here, we show that the inliers in the set of measurements (4) are contained in a single hyperclique of the compatibility hypergraph. Let us start with some definitions.

Definition 6 (Hypercliques and Maximum Hyperclique in Hypergraphs): A hyperclique of an \(n\)-uniform hypergraph \(\mathcal{G}\) is a set of vertices such that any subset of \(n\) vertices is connected by an hyperedge in \(\mathcal{G}\). The maximum hyperclique of \(\mathcal{G}\) is the hyperclique with the largest number of vertices.

Again, in the case where \(n = 2\), the above definition reduces to the usual clique and maximum clique definition. Given a compatibility hypergraph \(\mathcal{G}\), the following result relates the set of inliers in (4) with hypercliques in \(\mathcal{G}\) (proof in Appendix A in [34]).

Theorem 7 (Inliers and Hypercliques): Assume that we are given measurements (4) with inlier noise bound \(\beta\) and the corresponding \(n\)-invariants; call \(\mathcal{G}\) the corresponding compatibility hypergraph. Then, assuming that there are at least \(n\) inliers, all inliers belong to a single hyperclique in \(\mathcal{G}\).

Theorem 7 implies that we can look for inliers by computing hypercliques in the compatibility graph. Since we expect to have more compatible inliers than outliers, here, we propose to compute the maximum hyperclique; this approach is shown to work extremely well in practice in Section VIII. Appendix C in [34] describes an algorithm to find the maximum hyperclique.

Comparison with [28]: In our previous work [28]—where we first proposed ROBIN—we defined compatibility graphs as ordinary graphs, and inliers structures as maximum cliques. In the present article, we define compatibility graphs as hypergraphs (see Definition 5) and inlier structures as maximum

---

3In an \(n\)-uniform hypergraph, each hyperedge involves exactly \(n\) nodes.
Algorithm 1: ROBIN.

1. **Input:** set of measurements \( \mathcal{Y} \) and model (4); \( n \)-invariant functions \( f_\varphi, F_\varphi \) (for some \( n \)); inter noise bound \( \beta \).
2. **Output:** subset \( \mathcal{Y}^* \subset \mathcal{Y} \).
3. Initialize compatibility graph \( \mathcal{Y} = \mathcal{Y} \); \% each node is a measurement
4. \( \mathcal{E} = \emptyset \); \% start with empty hyperedge set
5. **Perform compatibility tests**
6. for all subsets \( \mathcal{M} \subset \mathcal{Y} \) of size \( n \) do
7. \( \text{if} \ test\text{Compatibility}(\mathcal{M}, f_\varphi, F_\varphi) = \text{pass} \) then
8. \( \text{add a hyperedge } e = \mathcal{M} \) to \( \mathcal{E} \);
9. end
10. end
11. **Find compatible measurements**
12. \( \mathcal{Y}^* = \max\text{hyperclique}(\mathcal{V}, \mathcal{E}) \)
13. **Return:** \( \mathcal{Y}^* \).

Hypercliques (see Theorem 7). This new formulation is equivalent to [28] for 2-invariants, but different otherwise. Topologically, the compatibility graphs in [28] can be seen as clique-expanded compatibility hypergraphs, where each hyperedge on a subset of \( n \) nodes is substituted with pairwise edges between all nodes in the subset (i.e., a clique in the graph). Compared to [28], our new formulation leads to pruning a larger number of outliers (see Appendix B in [34] for a concrete example).

D. ROBIN: Graph-Theoretic Outlier Rejection

This section summarizes all the findings above into a single algorithm for graph-theoretic outlier pruning, named ROBIN. ROBIN’s pseudocode is given in Algorithm 1. The algorithm takes a set of measurements \( \mathcal{Y} \) in input and outputs a subset \( \mathcal{Y}^* \subset \mathcal{Y} \), from which many outliers have been pruned. Given an \( n \)-invariant, ROBIN first performs compatibility tests on all subsets of \( n \) measurements and builds the corresponding compatibility hypergraph (lines 3–11). Then, it uses a maximum hyperclique solver to compute the subset of measurements surviving outlier pruning (lines 12 and 14). We have implemented the maximum hyperclique solver described in Appendix C in [34] using CVXPY [35]; in the special case where \( n = 2 \), i.e., the compatibility graph is an ordinary graph, we use the parallel maximum clique solver from [36]. We remark that ROBIN is not guaranteed to reject all outliers, i.e., some outliers may still pass the compatibility tests and end up in the maximum hyperclique. Indeed, as mentioned in the introduction, ROBIN is designed to be a preprocessing step to prune gross outliers and enhance the robustness of existing robust estimators. Note that Algorithm 1 can be applied to various estimation problems, as long as suitable compatibility tests are available (i.e., test\text{Compatibility} function).

We derive a pairwise invariant (i.e., a 2-invariant) according to Definition 2. The challenge is to develop a set-valued function \( F \) that does not depend on the pose and shape parameters, which are unknown. Toward this goal, we show how to manipulate (2) to obtain a function \( F \) that does not depend on \( R, t, \) and \( c \).

Taking the difference between measurement \( i \) and \( j \) in (2) leads to

\[
p_{3D,i} - p_{3D,j} = R \sum_{k=1}^{K} c^k(b^k_j - b^k_i) + (\epsilon_{3D,i} - \epsilon_{3D,j})
\]

where the translation cancels out in the subtraction. Now, taking the \( \ell_2 \) norm of both members, we obtain

\[
\|p_{3D,i} - p_{3D,j}\| = \left\| \sum_{k=1}^{K} c^k(b^k_j - b^k_i) + (\epsilon_{3D,i} - \epsilon_{3D,j}) \right\|
\]

Using the triangle inequality, we have

\[
-\|\epsilon_{3D,i} - \epsilon_{3D,j}\| \leq \|p_{3D,i} - p_{3D,j}\| - \left\| \sum_{k=1}^{K} c^k(b^k_j - b^k_i) \right\| \\
\leq \|\epsilon_{3D,i} - \epsilon_{3D,j}\|.
\]

Now, observing that the \( \ell_2 \) norm is invariant to rotation and rearranging the terms, we have

\[
\left\| \sum_{k=1}^{K} c^k(b^k_j - b^k_i) \right\| - \|\epsilon_{3D,i} - \epsilon_{3D,j}\| \leq \|p_{3D,i} - p_{3D,j}\| \\
\leq \left\| \sum_{k=1}^{K} c^k(b^k_j - b^k_i) \right\| + \|\epsilon_{3D,i} - \epsilon_{3D,j}\|.
\]

Taking the extreme cases over the possible shape coefficients, we obtain

\[
\min_{c^k \geq 0, k=1}^{\ell_2 min} \left\| \sum_{k=1}^{K} c^k(b^k_j - b^k_i) \right\| - \|\epsilon_{3D,i} - \epsilon_{3D,j}\| \leq \|p_{3D,i} - p_{3D,j}\| \\
\leq \left\| \sum_{k=1}^{K} c^k(b^k_j - b^k_i) \right\| + \|\epsilon_{3D,i} - \epsilon_{3D,j}\|.
\]

Since \( \sum_{k=1}^{K} c^k(b^k_j - b^k_i) \) is a convex combinations of the points \( b^k_j \) \((k = 1, \ldots, K)\) and hence lies in the convex hull of such points, the term \( \|\sum_{k=1}^{K} c^k(b^k_j - b^k_i)\| \) represents the distance between two (unknown) points in the two convex hulls defined by the set of points \( b^k_j \) and \( b^k_i \) \((k = 1, \ldots, K)\) (see Fig. 3). The minimum \( b_{ij}^{\text{min}} \) and the maximum \( b_{ij}^{\text{max}} \) over the convex hulls can be easily computed, either in closed form or via small convex programs (see details in Appendix D in [34]). Accordingly

\[
\|p_{3D,i} - p_{3D,j}\| \\
\in \left[ b_{ij}^{\text{min}} - \|\epsilon_{3D,i} - \epsilon_{3D,j}\|, b_{ij}^{\text{max}} + \|\epsilon_{3D,i} - \epsilon_{3D,j}\| \right].
\]

Note that \( b_{ij}^{\text{min}} \) and \( b_{ij}^{\text{max}} \) only depend on the given CAD library and are independent on \( \{R, t, c\} \). Therefore, they can be precomputed. We can now define the pairwise invariant for Problem 1 with generative model defined in (2).

Proposition 8 (3-D–3-D category-level pairwise invariant and compatibility test): Assume bounded noise \( \|\epsilon_{3D,i}\| \leq \beta_{3D} \)
Fig. 3. Example of compatibility test with three CAD models of cars (red, dark green, blue, indexed from 1 to 3). (Noiseless) inliers (e.g., the detection of the back wheel positions across CAD models) must fall in the convex hull of the corresponding points on the CAD models (e.g., triangle $b_1 - b_2 - b_3$ encompassing the back wheel positions across CAD models). This restricts the relative distance between two inliers and allows filtering out outliers. For instance, the dashed black line shows a distance that is compatible with the location of the convex hulls, while the solid black line is too short compared to the relative position of the wheels (for any car model) and allows pointing out that there is an outlier (i.e., $p_{3D,j}$ in the figure).

for $i = 1, \ldots, N$. The function $f_{3D}(p_{3D,i}, p_{3D,j}) \doteq \| p_{3D,j} - p_{3D,i} \|$ is a pairwise invariant for (2), with

$$F_{3D}(\theta_i, \theta_j, \beta_3) = [b_{ij}^{\min} - 2\beta_3, b_{ij}^{\max} + 2\beta_3]$$

where $\theta_i = \{b_i^k \mid k = 1, \ldots, K\}$ and $\theta_j = \{b_j^k \mid k = 1, \ldots, K\}$. Therefore, two measurements $p_{3D,i}$ and $p_{3D,j}$ are compatible if $f_{3D}(p_{3D,j}, p_{3D,i}) \in F_{3D}(\theta_i, \theta_j, \beta_3)$.

The proof of the proposition trivially follows from (18) and from the observation that $\| \epsilon_{3D,i} \| \leq \beta_3$ and $\| \epsilon_{3D,j} \| \leq \beta_3$ imply $\| \epsilon_{3D,j} - \epsilon_{3D,i} \| \leq 2\beta_3$.

Proposition 8 provides a compatibility test according to Definition 4. In words, a pair of measurements is mutually compatible if their distance $\| p_{3D,j} - p_{3D,i} \|$ matches the corresponding distances in the CAD models (lower-bounded by $b_{ij}^{\min}$ and upper-bounded by $b_{ij}^{\max}$) up to measurement noise $\beta_3$. If a pair of measurements fails the compatibility test, then one of them must be an outlier. A geometric interpretation of the compatibility test (for $\beta_3 = 0$) is given in Fig. 3.

B. 2-D–3-D Category-Level Compatibility Test

The pairwise invariant presented in the previous section was inspired by the fact that the distance between pairs of points is (a noiseless) invariant to rigid transformations. When it comes to 2-D–3-D problems, it is known that there is no (noiseless) invariant for 3-D points in generic configurations under perspective projection [38]. One option would be to use invariants for special configurations of points, such as cross ratios for collinear points [28]; however, this would not be generally applicable to our problem, where 3-D keypoints are arbitrarily distributed on the CAD models. Here, we take an alternative route, and we directly design a generalized 3-invariant for generic 3-D point projections.

Our 2-D–3-D category-level compatibility test draws inspiration from back-face culling in computer graphics [39]. The key idea is that when observing an object, the winding order of triplet of keypoints seen in the image (roughly speaking: if the points are arranged in clockwise or counterclockwise order) must be consistent with the arrangement of the corresponding triplet of keypoints in the CAD models. Therefore, we formulate a test by checking whether the observed winding order matches our expectation from the CAD models. In this section, we first define the notion of 2-D and 3-D winding orders, as well as the visibility and covisibility regions (camera locations where triplets of points are covisible); then we introduce a 3-invariant involving winding orders; finally, we combine winding orders and covisibility regions to develop a 2-D–3-D category-level compatibility test.

2-D Winding Orders: Winding orders refer to whether an ordered triplet of projected points is arranged in a clockwise or counterclockwise order. For example, if we enumerate the points in Fig. 4 in ascending order of their indices (i.e., 1, 2, 3), camera 1 sees points in counterclockwise order, while camera 2 sees them in clockwise order.

Definition 9 (2-D winding order): Given three 2-D image points $p_{2D,i}$, $p_{2D,j}$, and $p_{2D,m}$, where $i < j < m$, their winding order is the orientation (clockwise, counterclockwise) of points when enumerating them in the order $i \rightarrow j \rightarrow m$.

The following proposition allows computing the 2-D winding order algebraically (proof in Appendix E in [34]).

Proposition 10 (2-D winding order computation): Assume three noncollinear 2-D image points $p_{2D,i}$, $p_{2D,j}$, and $p_{2D,m}$, then the winding order $W$ can be computed as

$$W = \begin{cases} 
\text{clockwise}, & \text{if } V > 0 \\
\text{counterclockwise}, & \text{otherwise}
\end{cases}$$

where $V = \det([p_{2D,j} - p_{2D,i}, p_{2D,m} - p_{2D,i}])$ and $\det(\cdot)$ denotes the matrix determinant.

In words, Proposition 10 states that the 2-D winding order can be calculated from the signed area of the parallelogram formed by $p_{2D,j} - p_{2D,i}$ and $p_{2D,m} - p_{2D,i}$.

Half-spaces and 3-D winding orders: While Proposition 10 provides a simple way to compute the winding order for a triplet of 2-D image points, toward our 2-D–3-D category-level invariant, we need to establish a notion of winding order also for the 3-D shape keypoints on a CAD model. In the following, we show that the winding order for a triplet of 3-D points
can be uniquely determined given the location of the camera.
To develop some intuition, consider Fig. 4, where we have three 3-D keypoints on the faces of a cube. The plane passing across the triplet of 3-D keypoints divides the space into two half-spaces. Theorem 11 shows that, whenever the camera lies within one of the half-spaces, only one winding order is possible. Therefore, if the keypoints are only covisible by camera locations in one of the half-spaces, their winding order is uniquely determined.

Let us formally define the half-spaces induced by the triplet plane. Let \( b_i^k, b_j^k, \text{ and } b_m^k \) \((i < j < m)\) be three model keypoints on the \( k \)th CAD model. Define the triplet normal vector in the model’s frame as follows (cf. \( n \) in Fig. 4): \[
\mathbf{n}_{i,j,m}^k = (\mathbf{b}_j^k - \mathbf{b}_i^k) \times (\mathbf{b}_m^k - \mathbf{b}_i^k). \tag{20}
\]
Therefore, the triplet plane equation is \((\mathbf{o} - \mathbf{b}_i^k) \cdot \mathbf{n}_{i,j,m}^k = 0\) for any \( \mathbf{o} \in \mathbb{R}^3 \). If \((\mathbf{o} - \mathbf{b}_i^k) \cdot \mathbf{n}_{i,j,m}^k > 0 \) (respectively, \((\mathbf{o} - \mathbf{b}_i^k) \cdot \mathbf{n}_{i,j,m}^k < 0 \), \( \mathbf{o} \) lies in the positive (respectively, negative) half-space. In this section, one can think about \( \mathbf{o} \) as the optical center of the camera in the CAD model’s frame; hence, the inequalities above capture which half-space the camera is located in.

The following theorem connects winding order with the two half-spaces created by the triplet plane (proof in Appendix F in [34]).

**Theorem 11 (Half-spaces and 3-D winding orders):** Under perspective projection per (3) with zero noise \((\mathbf{e}_{2D,i} = \mathbf{e}_{2D,j} = \mathbf{e}_{2D,m} = 0)\), the following equality holds:

\[
\text{sgn}((\mathbf{o} - \mathbf{b}_i^k) \cdot \mathbf{n}_{i,j,m}^k) = -\text{sgn}(\det[\mathbf{P}_{2D,j} - \mathbf{P}_{2D,i} \mathbf{P}_{2D,m} - \mathbf{P}_{2D,i}]) \tag{21}
\]

where \(\text{sgn}(\cdot)\) is the signum function.

The theorem states that the half-space the camera is located in (identified by \(\text{sgn}((\mathbf{o} - \mathbf{b}_i^k) \cdot \mathbf{n}_{i,j,m}^k)\)) uniquely determines the 2-D winding order of the projection of the 3-D points. This is not informative if the camera can be anywhere, since both winding orders are possible. In the following, we use the idea of covisibility regions to restrict potential locations of the camera, such that we can associate a possibly unique winding order to triplets of 3-D keypoints.

**Visibility and Covisibility Regions:** Visibility and covisibility regions describe the set of camera locations from which keypoints are visible. The 3-D object observed by the camera is assumed to be opaque; hence, a keypoint visible from one viewpoint might not be visible from another due to self-occlusions. Fig. 4 demonstrates this concept: due to self-occlusions, the three keypoints on the cube are visible to Camera 1, but occluded (i.e., not visible) in Camera 2.

We now define the visibility region of keypoints on a shape, following the standard definition [40].

**Definition 12 (Visibility region):** The visibility region of a keypoint \( b_i^k \) is the set of 3-D points \( \{o\} \) such that line segments connecting \( o \) and \( b_i^k \) do not intersect the \( k \)th shape model.

We also define covisibility regions for keypoints triplets, which are the 3-D space in which all three keypoints are visible.

**Definition 13 (Covisibility region):** The covisibility region of a triplet of keypoints \( b_i^k, b_j^k, \text{ and } b_m^k \) is the intersection of the visibility region of each keypoint.

Fig. 5(b) shows visibility and covisibility regions surrounding the Q7-SUV model from the ApolloScape dataset for selected triplets of keypoints. For the triplet \((0, 1, 8)\), their covisibility regions are to the front of the car, whereas for \((12, 14, 17)\), their covisibility regions are to the left of the car. Notice how the relative positions between triplet half-spaces and covisibility regions affect the visible winding orders [see Fig. 5(c)]. The plane formed by \((0, 1, 8)\) cuts through their covisibility regions; hence, both winding orders are visible. For \((12, 14, 17)\), their covisibility regions are on one side of the keypoint plane, so the only visible winding order is clockwise. This observation is formalized as follows.

**Corollary 14 (Covisibility-constrained 3-D winding orders):** The projection of a triplet of 3-D keypoints \( b_i^k, b_j^k, \text{ and } b_m^k \) is arranged in counterclockwise winding order if and only if

\[
\{\mathbf{o} \in \mathbb{R}^3 \mid (\mathbf{o} - \mathbf{b}_i^k) \cdot \mathbf{n}_{i,j,m}^k > 0, \mathbf{o} \in \mathcal{C}\} \neq \emptyset \tag{22}
\]

where \( \mathcal{C} \) is the covisibility region of \( b_i^k, b_j^k, \text{ and } b_m^k \) on the \( k \)th shape. Similarly, the keypoints can be viewed in clockwise winding order if and only if

\[
\{\mathbf{o} \in \mathbb{R}^3 \mid (\mathbf{o} - \mathbf{b}_i^k) \cdot \mathbf{n}_{i,j,m}^k < 0, \mathbf{o} \in \mathcal{C}\} \neq \emptyset. \tag{23}
\]
This corollary follows directly from Proposition 10, Theorem 11, and Definition 13. Remarkably, the two feasibility problems in (22) and (23) can be solved a priori, since they only depend on the triplets of 3-D keypoints and their normal. For polyhedral shapes, the constraint $o \in C$ can be expressed via linear constraints; in addition, if we replace the strict inequalities in (22) and (23) with nonstrict ones and replace zeros with a small positive constant, the problems can be solved via linear programming. For complex shapes, we can check the conditions in (22) and (23) by splitting the space around the CAD models into voxels (i.e., we discretize the set of possible $o$) and ray tracing the keypoint to check visibility (see Appendix H in [34]). In summary, given a CAD model and for each triplet of 3-D keypoints, using (22) and (23), we are able to predict if a covisible triplet will produce clockwise or counterclockwise keypoint projections.

**Definition 15 (Feasible winding order dictionary):** For shape $k$, its feasible winding order dictionary $W_k : \{1, \ldots, N\}^3 \to \mathcal{P}(-1,1)$ (where $\mathcal{P}(-1,1)$ denotes the power set, i.e., $\{+1,-1,0\}$) is defined as

$$W_k(i, j, m) = \{\text{sgn}((o - b_k^i) \cdot n_{i,j,m}^k) | \forall o \in C\}$$

where $\{\text{sgn}((o - b_k^i) \cdot n_{i,j,m}^k) | \forall o \in C\}$ is empty if both (22) and (23) are false (i.e., when the triplet is never covisible); it contains $+1$ if (22) is true; it contains $-1$ if (23) is true; it contains both $+1$ and $-1$ if both (22) and (23) are true.

We are now ready to define our generalized 3-invariant and the corresponding compatibility test.

**2-D-3-D Invariant and Compatibility Test:** We solve the two feasibility problems (22) and (23) for all triplets and CAD models and obtain a dictionary of feasible winding orders. In essence, each dictionary serves as a compatibility test for a single shape: for observed keypoints $p_{2D,i}$, $p_{2D,j}$, and $p_{2D,m}$, if the observed winding orders are not in $W_k(i, j, m)$, then the triplets are not compatible. However, this dictionary is only for a single shape. To formulate a 2-D-3-D category-level invariant, we need to create a dictionary of feasible winding orders for all $K$ shapes. We address this in Proposition 16.

**Proposition 16 (2-D-3-D category-level invariant and compatibility test):** Assume that the keypoints in (3) are generated by one of the shapes $\{1, \ldots, K\}$, that the reprojection noise is bounded by $\beta$ (i.e., $|\varepsilon_{2D,i}| \leq \beta$, $|\varepsilon_{2D,j}| \leq \beta$, $|\varepsilon_{2D,m}| \leq \beta$), and that $\beta$ is small enough for Theorem 11 to hold true; then, the functions $f_{2D}(p_{2D,i}, p_{2D,j}, p_{2D,m})$ is a 3-invariant for (3), with

$$f_{2D}(p_{2D,i}, p_{2D,j}, p_{2D,m}) = \det((p_{2D,j} - p_{2D,i}), p_{2D,m} - p_{2D,i})$$

(25)

$$F_{2D}(\theta_i, \theta_j, \theta_m) = \sum_{k=1}^{K} W_k(i, j, m)$$

(26)

where $\theta = \{b_k^i | k = 1, \ldots, K\}$ and $W_k$ is the winding order dictionary for shape $k$, as per Definition 15. Therefore, a triplet of measurements $p_{2D,i}, p_{2D,j},$ and $p_{2D,m}$ is compatible if $f_{2D}(p_{2D,i}, p_{2D,j}, p_{2D,m}) \in F_{2D}(\theta_i, \theta_j, \theta_m)$.

Intuitively, the proposition states that the observed winding order must match at least one of the observed winding orders contained in the feasible winding order dictionary of all shapes. The non-compatible triplets from Proposition 16 are points with noise so large that the measured winding orders become inconsistent with the models. Under technical conditions (discussed in Appendix G in [34]), Proposition 16 holds true even when keypoints are generated by convex combinations of $b_k^i$. 

**VI. Stage II: Certifiably Optimal Solvers for Category-Level Perception**

While Stage 1 serves the purpose of filtering out a large fraction of gross outliers (without even computing an estimate), Stage 2 aims to use the remaining measurements to estimate the pose and shape parameters. In this section, we develop certifiably optimal solvers for Problems 1 and 2 in the outlier-free case (i.e., assuming that ROBIN removed all the outliers). In Section VII, we further improve robustness by incorporating a GNC scheme to handle potential leftover outliers in the measurements after ROBIN.

**A. Certifiably Optimal Solver for Outlier-Free 3-D–3-D Category-Level Perception**

We show how to solve Problem 1 in the outlier-free case, where the noise $\varepsilon_{3D,i}$ is assumed to follow a zero-mean Gaussian distribution. In the outlier-free case, a standard formulation for the pose and shape estimation problem leads to a regularized nonlinear least squares problem (3D-3D)

$$\min_{t \in \mathbb{R}^3, \lambda \geq 0} \sum_{i=1}^{N} \left( \|p_{3D,i} - R \sum_{k=1}^{K} c_k b_k^{i,t} - t\|^2 + \lambda \|\varepsilon\|^2 \right)$$

(3D-3D)

where the first summand in the objective minimizes the residual error w.r.t. the generative model (2) ($w_i \geq 0, i = 1, \ldots, N$ are given weights), and the second term provides an $\ell_2$ regularization (a.k.a. Tikhonov regularization [41]) of the shape coefficients $c$ (controlled by the user-specified parameter $\lambda \geq 0$). Note that for mathematical convenience, we replaced the constraint $c \in \Delta_K$ with the constraint $1^T c = 1$, where $1$ is a vector with all entries equal to 1; in other words, we force the shape coefficients to sum up to 1 but allow them to be negative. Note that the regularization term serves the dual purpose of penalizing the occurrence of large negative entries in $c$ and keeping the problem well-posed regardless of the number of shapes in the library. From the probabilistic standpoint, problem (3D-3D) is a maximum a posteriori (MAP) estimator assuming that the keypoints measurement noise follows a zero-mean Gaussian with covariance $\frac{1}{\lambda} I_3$ (where $I_3$ is the 3-by-3 identity matrix), and we have a zero-mean Gaussian prior with covariance $\frac{1}{\lambda}$ over the shape parameters $c$ (see Appendix I in [34]). Problem (3D-3D) is nonconvex due to the product between $R$ and $c$ in the objective, and due to the nonconvexity of the constraint set $SO(3)$, the rotation $R$ is required to belong to $SO(3)$. Therefore, existing approaches based on local search [43], [44], [45] are prone to converge to local minima corresponding to incorrect estimates.

**3-D-3-D Solver Overview:** We develop the first certifiably optimal algorithm to solve (3D-3D). Toward this goal, we show that 1) the translation $t$ in (3D-3D) can be solved in closed form given the rotation and shape parameters (see Section VI-A1); 2) the shape parameters $c$ can be solved in closed form given the rotation (see Section VI-A2); and 3) the rotation can be estimated (independently on shape and translation) using a tight semidefinite relaxation (see Section VI-A3).

1) Closed-Form Translation Estimation: By inspection of (3D-3D), we observe that $t$ is unconstrained and appears quadratically in the cost. Hence, for any choice of $R$ and $c$, the optimal translation can be computed in closed form

$$t^*(R, c) = y_w - \frac{R}{\sum_{k=1}^{K} c_k b_{k,w}}$$

(27)
where 

\[
y_w \triangleq \frac{1}{\sum_{i=1}^{N} w_i} \sum_{i=1}^{N} w_i p_{3D,i}, \quad b_{k,w} \triangleq \frac{1}{\sum_{i=1}^{N} w_i} \sum_{i=1}^{N} w_i b_{k,i}^T
\]

are the weighted centroids of \( p_{3D,i} \) and \( b_{k,i}^T \)'s. This manipulation is common in related work; see, e.g., [14] and [15].

2) Closed-Form Shape Estimation: Substituting the optimal translation (27) (as a function of \( R \) and \( \epsilon \)) back into (3D-3D), we obtain an optimization that only depends on \( R \) and \( \epsilon \)

\[
\min_{R \in SO(3)} \sum_{i=1}^{N} \| y_i - R \sum_{k=1}^{K} c_k b_{k,i} \|^2 + \lambda \| \epsilon \|^2
\]

(28)

where

\[
y_i \triangleq \sqrt{u_i}(p_{3D,i} - y_w), \quad b_{k,i}^T \triangleq \sqrt{u_i}(b_{k,i}^T - b_{k,w})
\]

(29)

are the (weighted) relative positions of \( p_{3D,i} \) and \( b_{k,i}^T \) w.r.t. their corresponding weighted centroids. Using the fact that the \( \ell_2 \) norm is invariant to rotation, problem (28) is equivalent to

\[
\min_{R \in SO(3)} \sum_{i=1}^{N} \| R^T y_i - R \sum_{k=1}^{K} c_k b_{k,i}^T \|^2 + \lambda \| \epsilon \|^2.
\]

(30)

We can further simplify the problem by adopting the following matrix notations:

\[
y = (y_1^T, \ldots, y_N^T)^T \in \mathbb{R}^{3N}
\]

(31)

\[
B = \begin{bmatrix}
   b_1 & \cdots & b_K
   \vdots & \ddots & \vdots
   b_1 & \cdots & b_K
\end{bmatrix} \in \mathbb{R}^{3N \times K}
\]

(32)

which allows rewriting (30) in the following compact form:

\[
\min_{R \in SO(3)} \| B c - (I_N \otimes R^T) y \|^2 + \lambda \| \epsilon \|^2.
\]

(33)

Now, the reader can again recognize that—for any choice of \( R \)—problem (33) is a linearly constrained linear least squares problem in \( c \), which admits a closed-form solution.

Proposition 17 (Optimal shape): For any choice of rotation \( R \), the optimal shape parameters that solve (33) can be computed in closed form as

\[
c^*(R) = 2GB^T(I_N \otimes R^T)y + g
\]

(34)

where we defined the following constant matrices and vectors:

\[
H \triangleq 2(BB^T + \lambda I_K)
\]

(35)

\[
G \triangleq H^{-1} - H^{-1}11^T H^{-1}, \quad g \triangleq H^{-1}1^T H^{-1}.
\]

(36)

3) Certifiably Optimal Rotation Estimation: Substituting the optimal shape parameters (34) (as a function of \( R \)) back into (33), we obtain an optimization that only depends on \( R \)

\[
\min_{R \in SO(3)} \| M(I_N \otimes R^T)y + h \|^2
\]

(37)

where the matrix \( M \in \mathbb{R}^{(3N+K) \times 3N} \) and vector \( h \in \mathbb{R}^{3N+K} \) are defined as

\[
M \triangleq \begin{bmatrix} 2GBG^T & -1_{3N} \end{bmatrix}, \quad h \triangleq \begin{bmatrix} \bar{B}g \\ g \end{bmatrix}.
\]

(38)

Problem (37) is a quadratic optimization over the nonconvex set \( SO(3) \). It is known that \( SO(3) \) can be described as a set of quadratic equality constraints; see, e.g., [42], [46] or [15, Lemma 5]. Therefore, we can succinctly rewrite (37) as a quadratically constrained quadratic program (QCQP).

Proposition 18 (Optimal rotation): The category-level rotation estimation problem (37) can be equivalently written as a QCQP

\[
\min_{\tilde{r} \in \mathbb{R}^{10}} \tilde{r}^T Q \tilde{r}
\]

s.t. \( \tilde{r}^T A_i \tilde{r} = 0, \forall i = 1, \ldots, 15 \)

(39)

where \( \tilde{r} \triangleq [1, vec(R^T)]^T \in \mathbb{R}^{10} \) is a vector stacking all the entries of the unknown rotation \( R \) in (37) (with an additional unit element), \( Q \in S^{10} \) is a symmetric constant matrix (expression given in Appendix J in [34]), and \( A_i \in S^{10}, i = 1, \ldots, 15 \), are the constant matrices that define the quadratic constraints describing the set \( SO(3) \) [15, Lemma 5].

While a QCQP is still a nonconvex problem, it admits a standard semidefinite relaxation, described as follows.

Corollary 19 (Shor’s semidefinite relaxation): The following semidefinite program (SDP) is a convex relaxation of (39):

\[
\min_{X \in S^{10}} tr(QX)
\]

s.t. \( tr(A_i X) = 1, \forall i = 1, \ldots, 15 \)

\( X \succeq 0. \)

(40)

Moreover, when the optimal solution \( X^* \) of (40) has rank 1, it can be factorized as \( X^* = [vec(R^T)] [1 vec(R^T)]^T \), where \( R^* \) is the optimal rotation minimizing (37).

The relaxation entails solving a small SDP (10 x 10 matrix size, and 16 linear equality constraints); hence, it can be solved in milliseconds using standard interior-point methods (e.g., MOSEK [47]). Similar to related quadratic problems over \( SO(3) \) [7], [42], [48], [49], [50], the relaxation (40) empirically produces rank-1—and hence optimal—solutions in common problems. Even when the relaxation is not tight, the relaxation allows computing an estimate and a bound on its suboptimality (see Appendix L in [34]). The proposed solution falls in the class of certifiable algorithms (see [51] and [7, Appendix A]), since it allows solving a hard (nonconvex) problem efficiently and with provable a posteriori guarantees.

4) Summary: The results in this section suggest a simple algorithm to compute a certifiably optimal solution to the pose and shape estimation problem (3D-3D): 1) we first estimate the rotation \( R^* \) using the semidefinite relaxation (40) (and compute the corresponding suboptimality gap \( \eta \), as described in Appendix L in [34]); 2) we retrieve the optimal shape \( c^*(R^*) \) given the optimal rotation using (34); and 3) we retrieve the optimal translation \( t^*(R^*, c^*) \) using (27). If \( \eta = 0 \), we certify that \( (R^*, t^*, c^*) \) is a globally optimal solution to (3D-3D).

We call the resulting algorithm \texttt{PAGE3D} (Pose and shApes estimation for 3-D–3-D Category-level Perception).

B. Certifiably Optimal Solver for Outlier-Free 2-D–3-D Category-Level Perception

We now show how to solve Problem 2 in the outlier-free case, where the noise \( \epsilon_{2D,i} \) in the generative model (3) follows a zero-mean isotropic Gaussian distribution. In such a case, the MAP
estimator for Problem 2 becomes

\[
\min_{\mathbf{t} \in \mathbb{R}^3, \mathbf{c} \in \Delta_K} \sum_{i=1}^{N} w_i \left\| \mathbf{p}_{2D,i} - \mathbf{R} \sum_{k=1}^{K} c_k \mathbf{b}_k^i + \mathbf{t} \right\|^2 + \lambda \| \mathbf{c} \|^2
\]

(41)

where \(\lambda \| \mathbf{c} \|^2\) with \(\lambda \geq 0\) is an \(\ell_2\) regularization on the shape parameters, and \(w_i\) are given nonnegative weights. Equation (41) minimizes the geometric reprojection error and belongs to a class of optimization problems known as fractional programming because the objective in (41) is a sum of rational functions. Unfortunately, it is generally intractable to obtain a globally optimal solution for fractional programming [52]. In fact, even if \(c\) is known in (41), searching for the optimal \((\mathbf{R}, \mathbf{t})\) typically resorts to branch-and-bound (BnB) [53], [54], which runs in worst case exponential time.

To circumvent the difficulty of fractional programming caused by the geometric reprojection error, we adopt an algebraic reprojection error that minimizes the point-to-line distance between each 3-D keypoint (i.e., \(\mathbf{R} \sum_{k=1}^{K} c_k \mathbf{b}_k^i + \mathbf{t}\)) and the bearing vector emanating from the camera center to the 2-D keypoint (i.e., the bearing vector \(\mathbf{v}_i = \mathbf{p}_{2D,i}/\|\mathbf{p}_{2D,i}\|\)), where

\[
\hat{\mathbf{p}}_{2D,i} = [\hat{\mathbf{p}}_{2D,i}]^T := \mathbf{I}_3 - \mathbf{v}_i \mathbf{v}_i^T
\]

computes the distance from a given 3-D point \(\mathbf{p}\) to a bearing vector \(\mathbf{v}_i\). The point-to-line objective in (2D-3D) has been adopted in other works, including [12], [55]. In the following, we omit the weights \(w_i\), since they can be directly included in the matrix \(\mathbf{W}_i\).

2-D–3-D solver overview: We develop the first certifiably optimal algorithm to solve problem (2D-3D). We show that

1) the translation \(\mathbf{t}\) in (2D-3D) can be solved in closed form given the rotation and shape parameters (see Section VI-B1) and
2) the shape parameters \(c\) and the rotation \(\hat{\mathbf{R}}\) can be estimated using a tight semidefinite relaxation (see Section VI-B2).

1) Closed-Form Translation Estimation: Similar to Section VI-A1, our first step is to algebraically eliminate the translation \(\mathbf{t}\) in (2D-3D) By deriving the gradient of the objective of (2D-3D) and setting it to zero, we obtain that

\[
\mathbf{t}^* = -\frac{1}{N} \mathbf{W}_i \mathbf{R} \left( \sum_{k=1}^{K} c_k \mathbf{b}_k^i \right)
\]

(42)

where \(\mathbf{W} := \sum_{i=1}^{N} \mathbf{W}_i\), and \(\bar{\mathbf{W}}_i := \mathbf{W}_i^{-1} \mathbf{W}_i\). Inserting (42) back into (2D-3D), we get the following translation-free problem:

\[
\min_{\mathbf{c} \in \mathbb{S}(3)} \sum_{i=1}^{N} \left\| \mathbf{R} \mathbf{s}_i(c) - \bar{\mathbf{W}}_j \mathbf{R} \mathbf{s}_j(c) \right\|^2 + \lambda \| \mathbf{c} \|^2
\]

(43)

where we compactly wrote \(\mathbf{s}_i(c) = \sum_{k=1}^{K} c_k \mathbf{b}_k^i\).

2) Certifiably Optimal Shape and Rotation Estimation: In this case, it is not easy to decouple rotation and shape parameters as we did in Section VI-A2. Therefore, we adopt a more advanced machinery to globally optimize \(\hat{\mathbf{R}}\) and \(\mathbf{c}\) together. Toward this goal, we observe that problem (43) is in the form of a polynomial optimization problem (POP), i.e., an optimization problem where both objective and constraints can be written using polynomials:

\[
\min_{\mathbf{x} \in \mathbb{R}^d} \quad p(\mathbf{x})
\]

s.t. \(h_i(\mathbf{x}) = 0, \quad i = 1, \ldots, l_h\)
\(g_j(\mathbf{x}) \geq 0, \quad j = 1, \ldots, l_g\)

(44)

where \(\mathbf{x} = [\mathbf{vec}(\mathbf{R})^T, \mathbf{c}^T]^T \in \mathbb{R}^d, d = K + 9\), denotes the vector of unknowns, \(p(\mathbf{x})\) is a degree-4 objective polynomial corresponding to the objective in (43), \(h_i(\mathbf{x}), i = 1, \ldots, l_h = 16\) are polynomial equality constraints (including \(\mathbf{R} \in \mathbb{S}(3)\) and \(\mathbf{1}^T \mathbf{c} = 1\)), and \(g_j(\mathbf{x}), j = 1, \ldots, l_g = K + 1\) are polynomial inequality constraints (e.g., \(c_k \geq 0\) for all \(k\)). \(p(\mathbf{x})\) in (44) has degree 4, which prevents us from applying Shor’s semidefinite relaxation as in Corollary 19. However, writing (43) in the form (44) allows us to leverage a more general semidefinite relaxation technique called Lasserre’s hierarchy of moment and sum-of-squares relaxations [27], [56] to solve (43) to certifiable global optimality.

Corollary 20 (Order-2 Lasserre’s moment relaxation): The following multiblock SDP is a convex relaxation of (39):

\[
\mathbf{x} = (\mathbf{X}_0, \mathbf{X}_1, \ldots, \mathbf{X}_{K+1})\]

s.t. \(A(\mathbf{x}) = \mathbf{b}, \quad \mathbf{X} \succeq 0\)

(45)

where \(\mathbf{X}_0\) is the moment matrix of size \(n_0 = (K+1)^2\), \(\mathbf{X}_1, \ldots, \mathbf{X}_{K+1}\) are the so-called localizing matrices (of size \(n_i < n_{i+1}\) for \(i \geq 1\)) arising from the inequality constraints \(g_j\) in (44) and the additional constraint \(\mathbf{c}^T \mathbf{c} \leq 1, \quad \mathbf{X} \succeq 0\) indicates each element of \(\mathbf{X}\), i.e., \(\mathbf{X}_{ij}, i = 0, \ldots, K, j = 1, \ldots, m_i\), is positive semidefinite, \(C = (\mathbf{C}_0, \mathbf{C}_1, \ldots, \mathbf{C}_{K+1})\) are known matrices with \(\mathbf{C}_i = 0\) for \(i \geq 1\), and \(A(\mathbf{x}) = \mathbf{b}\) collects all linear equality constraints on \(\mathbf{X}\) (each scalar constraint is written as \(\sum_{i=0}^{K+1} \langle \mathbf{A}_{ij}, \mathbf{X}_i \rangle = b_j\) for \(j = 1, \ldots, m\)).

Moreover, when the optimal solution \(\mathbf{X}^*\) of (45) is such that \(\mathbf{X}^*_{0,0}\) has rank 1, then \(\mathbf{X}^*_0\) can be factorized as \(\mathbf{X}^*_0 = [\mathbf{x}^*]^2 [\mathbf{x}^*]^T\), where \(\mathbf{x}^* = [\mathbf{vec}(\mathbf{R}^*)^T, (\mathbf{c}^*)^T]^T\) is a globally optimal for (43), and \([\mathbf{x}^*]^2\) denotes the vector of monomials in the entries of \(\mathbf{x}^*\) of degree up to 2.

We refer the interested reader to [57, Sec. 2] for details about how to generate \((\mathbf{A}, \mathbf{b}, \mathbf{C})\) from the POP formulation (44). In practice, there exists an efficient MATLAB implementation4 that automatically generates the SDP relaxation (45) given a POP (44). The reader can observe that the SDP (45) is conceptually the same as the SDP (40) except that (45) has more than one positive-semidefinite matrix decision variable. Appendix N in [34] contains more details about Lasserre’s hierarchy optimality certificates and the rounding procedure to extract an estimate from the solution of the SDP (45).

3) Summary:

1) We solve problem (2D-3D) by solving the SDP (45) using MOSEK [47] and obtaining an estimate \((\hat{\mathbf{R}}, \hat{\mathbf{c}})\) and the corresponding suboptimality gap \(\eta\), using the rounding procedure in Appendix N in [34].

2) Then, we compute \(\mathbf{t}\) from (42). If \(\eta = 0\), we certify that \((\hat{\mathbf{R}}, \hat{\mathbf{t}}, \hat{\mathbf{c}})\) is a globally optimal solution to (2D-3D). We call this approach PACE2D+.

Remark 21 (Distribution of \(c\)): The active shape model allows the entries of \(c\) to be scalars in \([0,1]\), which enables the model to interpolate between shapes (see Fig. 2). We note that if the vector \(c\) is assumed to be a one-hot vector (i.e., it has a single entry

[4] Online: Available: https://github.com/MIT-SPARK/CertifiablyRobustPerception
equal to 1 and all other entries equal to zero), then there exist trivial solvers for (2D-3D) and (2D-3D). Namely, one can run $K$ times an instance-based solver (e.g., PnP or 3-D registration), once for each shape in the CAD library, and then pick the solution attaining the lowest cost. Such an approach sacrifices the interpolation power of the active shape model, and in the experiments, we show that it leads to large errors when $c$ is not a one-hot vector.

VII. STAGE II (CONTINUED): FURTHER ROBUSTNESS THROUGH GNC

While, in principle, we can use PACE3D* and PACE2D* directly after Stage 1, the measurements selected by ROBIN potentially still contain a few outliers. These outliers could still hinder the quality of the pose and shape estimates.

To compute accurate estimates in the face of those remaining outliers, we add a robust loss function—in particular, a truncated least squares (TLS) loss—to problems (3D-3D) and (2D-3D) and solve the resulting optimization using a standard GNC [29], [58] approach. At each iteration, GNC alternates between solving a weighted least squares problem in the forms (3D-3D) and (2D-3D) (these can be solved to certifiable optimality using PACE3D* and PACE2D*) and updating the weights for each measurement (which can be computed in closed form [29]).

The interested reader can find more details in Appendix O in [34]. We release the proposed solvers at https://github.com/MIT-SPARK/PACE.

VIII. EXPERIMENTS

This section presents a comprehensive evaluations of the proposed approaches. First, we showcase the optimality of PACE3D* and robustness of PACE3D# through experiments on synthetic data, PASCAL3D+ [59], and KeypointNet [60] (see Section VIII-A). Then, we demonstrate the optimality of PACE2D* and robustness of PACE2D# through experiments on synthetic datasets (see Section VIII-B). Finally, we test both PACE3D# and PACE2D# on ApolloScape, a self-driving dataset [61]. Experiments in Sections VIII-A and VIII-C are run with an Intel i9-9920X CPU at 3.5 GHz with 128-GB RAM. Section VIII-B experiments are run on MIT SuperCloud [62] Xeon Platinum 8260 cluster, using six threads and 24-GB RAM per run.

A. Optimality and Robustness of PACE3D* and PACE3D#

1) Optimality of PACE3D*: To evaluate the performance of PACE3D* in solving the outlier-free problem (3D-3D), we randomly simulate $K$ shape models $B_k$ whose points $b_k^i$ are drawn from an independent identically distributed (i.i.d.) Gaussian distribution $N(0, I_3)$. We sample shape parameters $c$ uniformly at random in $[0, 1]^K$ and normalize $c$ such that $1^T c = 1$. Then, we uniformly sample $R$ from $SO(3)$ and $t$ from $N(0, I_3)$ and generate the measurements $p_{3D,i}$ according to the model (2), where the noise $e_{3D,i}$ follows $N(0, \sigma^2 I_3)$ with standard deviation $\sigma = 0.01$. We fix $N = 100$ and increase $K$ from 1 up to 2000 to stress-test the algorithms. We set the regularization factor $\lambda = \sqrt{K/N}$ so that larger regularization is imposed when the problem becomes more ill-posed. We compare PACE3D* with a baseline approach based on alternating minimization [43], [44], [45] (details in Appendix M in [34]) that offers no optimality guarantees (label: Altern).

Fig. 6(a) plots the statistics of rotation error (angular distance between estimated and ground-truth rotations), translation error, shape parameters error ($\ell_2$ distance between estimated and ground-truth translation/shape parameters), as well as average runtime and relative duality gap (see also Appendix K in [34] for a formal definition). We make the following observations.

1) PACE3D* returns accurate pose and shape estimates up to $K = 2000$, while Altern starts failing at $K = 500$.
2) Although Altern is faster than PACE3D* for small $K$ (e.g., $K < 200$), PACE3D* is orders of magnitude faster than Altern for large $K$. PACE3D* solves a fixed-size SDP regardless of $K$ and the slight runtime increase is due to inversion of the dense matrix in (35).
3) The relaxation (40) is empirically tight (relative duality gap < $10^{-5}$), certifying global optimality of the solution returned by PACE3D*.

In Appendix P-A in [34], we show extra results with different noise levels and $N$.

2) Robustness of PACE3D#: To test the robustness of PACE3D# on outlier-contaminated data, we follow the same data generation protocol as before, except that: 1) when generating the CAD models, we follow a more realistic active shape model [13] where we first generate a mean shape $B$ whose points $b_i$ are i.i.d. Gaussian $N(0, I_3)$, and then, each CAD model is generated from the mean shape by $b_i^* = b_i + \nu_i$, where $\nu_i$ follows $N(0, \sigma^2 I_3)$ and represents the intraclass variation of semantic keypoints with variation radius $r$; and 2) we replace a fraction of the measurements $p_{3D,i}$ with arbitrary 3-D points sampled according to $N(0, I_3)$ and violating the generative model (2). We compare PACE3D# with two variants: Clique-PACE3D* (where, after pruning outliers using maximum clique, PACE3D* is applied without GNC) and GNC-PACE3D* (where GNC is applied without any outlier pruning), as well as two variants of the popular iterative reweighted least squares method: IRLS-TLS and IRLS-GM, where TLS and GM denote the TLS cost function and the Geman–McClure cost function [63]. Moreover, we compare against RANSAC-PACE3D*, a five-point RANSAC scheme. We use PACE3D* in the inner iterations of PACE3D#, GNC-PACE3D*, IRLS-TLS, IRLS-GM, and RANSAC-PACE3D*. We set $\beta_D = 0.05$ for outlier pruning in GNC, IRLS-TLS, and IRLS-GM. RANSAC-PACE3D* uses a maximum of 5000 iterations, whereas both IRLS-TLS and IRLS-GM use a maximum of $10^3$ iterations.

Fig. 6(b) plots the results under increasing outlier rates up to 93% when $N = 100, K = 10$, and $r = 0.1$. We make the following observations.

1) IRLS-TLS quickly fails (at 10% outlier rate) due to the highly nonconvex nature of the TLS cost, while IRLS-GM starts breaking at 60% outliers.
2) GNC-PACE3D* alone already outperforms IRLS-TLS and IRLS-GM and is robust to 60% outliers.
3) RANSAC-PACE3D* is also robust to 60% outliers.
4) With our maximum clique outlier pruning, the robustness of PACE3D# is boosted to 93%, a level that is comparable to cases when the shapes are known (see, e.g., [7]). In addition, outlier pruning speeds up the convergence of GNC-PACE3D* [cf. the runtime plot for GNC and PACE3D# in Fig. 6(b)].
5) Even without GNC, the outlier pruning is so effective that PACE3D* alone is able to succeed with up to 92% outliers, albeit the estimates are typically less accurate than PACE3D#.

In fact, looking at the clique inlier rate plot [yellow lineplot in Fig. 6(b)], the reader sees that the set of measurements after maximum clique pruning is almost free of outliers, explaining
Fig. 6. Performance of PACE3D* and PACE3D# compared with baselines in simulated experiments. (a) PACE3D* compared with alternating minimization (Altern) on synthetic outlier-free data with $N = 100$ and $K$ increasing from 10 to 2000. (b) PACE3D# and variants (Clique-PACE3D* and GNC), compared with two variants of iterative reweighted least squares (IRLS-GM and IRLS-TLS) [63] on synthetic outlier-contaminated data with $N = 100$, $K = 10$, and outlier rates up to 93%. Each boxplot and lineplot summarizes 50 Monte Carlo random runs. (c) Same as (b) but using the car category CAD models from the PASCAL3D+ dataset [59], with $N = 12$, $K = 9$, and outlier rates up to 80%. (d) Qualitative results of IRLS-GM, IRLS-TLS, GNC, and PACE3D# on a PASCAL3D+ instance with 70% outlier rate. Blue meshes represent the ground-truth shape, and yellow meshes represent the pose and shape estimated by each model. Red points represent outliers. In this case, both GNC and PACE3D# succeeded, while IRLS-GM and IRLS-TLS failed. (a) Performance of PACE3D* on outlier-free synthetic data: $N = 100$. (b) Robustness of PACE3D# against increasing outliers on synthetic data: $N = 100$, $K = 10$, $r = 0.1$. (c) Robustness of PACE3D# against increasing outliers on the car category in the PASCAL3D+ dataset [59]: $N = 12$, $K = 9$. (d) Qualitative results of IRLS-GM, IRLS-TLS, GNC, and PACE3D# on a PASCAL3D+ instance with 70% outlier rate.

3) Robustness on PASCAL3D+: For a simulation setup that is closer to realistic scenarios, we use the CAD models from the car category in the PASCAL3D+ dataset [59], which contains $K = 9$ CAD models with $N = 12$ semantic keypoints. We randomly sample ($R$, $t$, $c$) and add noise and outliers as before and compare the performance of PACE3D# with other baselines, as shown in Fig. 6(c). The dominance of PACE3D# over other baselines and the effectiveness of outlier pruning are clearly seen across the plots. PACE3D# is robust to 70% outliers (see Fig. 6(d) for a qualitative example), while other baselines break at a much lower outlier rate. Note that at 80% outlier rate, there are only two inlier semantic keypoints, making it pathological to estimate shape and pose.

4) Performance on KeypointNet: We conduct experiments on KeypointNet, a large-scale 3-D keypoint dataset built from ShapeNetCore [66], containing 8329 3-D models from 16 object categories [60]. For each object category, we select one object and render depth point clouds using Open3D [67]. We apply a random translation bounded within $[0, 1]^3$, normalized by the objects’ diameters, and apply a uniformly sampled rotation to the object. We generate 5000 samples as the training set, and 50 samples for test and validation sets each. We compare the following methods: 1) PACE3D#, where we first use a network based on PointTransformer [68] to detect keypoints and then run PACE3D#; 2) EquiPose [26]; 3) PointNetLK [64]; and 4) DeepGMR [65]. Pretrained models provided by the authors (and trained on the same KeypointNet dataset) are used for EquiPose, PointNetLK, and DeepGMR. Fig. 7 shows the cumulative distribution of ADD-S score [69] over four

Authorized licensed use limited to the terms of the applicable license agreement with IEEE. Restrictions apply.
categories (airplane, bottle, car, and chair), for which pretrained models are available for EquiPose. PACE3D# outperforms DeepGMR and PointNetLK by a large margin. Only EquiPose achieves comparable performance with PACE3D#. Additional experimental details and results on all KeypointNet categories are given in Appendix P-B in [34].

B. Optimality and Robustness of PACE2D* and PACE2D#

1) Optimality of PACE2D*: To evaluate the performance of PACE2D* in solving the outlier-free problem (41), we randomly simulate $K$ shapes $B_i$ whose points $b_i^k$’s are drawn from an i.i.d. Gaussian distribution $\mathcal{N}(0, I_3)$. We sample $c$ uniformly at random in the probability simplex $\Delta_K$. We draw random poses $(R, t)$ such that the camera lies on a sphere with radius equal to 3 centered at origin and generate the measurements $p_{2D,i}$ according to the model (3), where the noise $\epsilon_{2D,i}$ follows $\mathcal{N}(0, \sigma^2 I_2)$ with standard deviation $\sigma = 0.01$. We compare PACE2D* against: 1) a baseline approach based on a local solver optimizing (41), starting from an initial guess obtained by running EPnP [70] on the mean shape (label: MS-PnP); 2) a solver based on a convex relaxation using the weak perspective camera model [71] (label: Zhou); 3) a solver based on a tighter relaxation with the weak perspective model [15] (label: Shape*); and 4) a solver that solves a PnP problem for each of the $K$ shapes using SQPnP [72] and picks the one with the lowest cost (label: KPnP) (see Remark 21).

In Fig. 8(a), we report statistics for the rotation, translation, and shape errors. We also show the average runtime and relative duality gap. We observe that PACE2D* consistently outperforms all baselines. Predictably, KPnP fails when $K \geq 2$, as it assumes a one-hot $c$, while the ground-truth shape is a generic point in the probability simplex. In Appendix P-C in [34], we show that in the case of one-hot $c$, KPnP can indeed obtain good performance. While Zhou and Shape* perform similarly at $K = 1$, Shape* has lower errors at higher shape counts. Both Zhou and Shape* perform significantly worse than PACE2D*, as they use the weak perspective projection model to approximate the actual (fully perspective) camera. The relative duality gap for PACE2D* stays below $10^{-3}$, indicating an empirically tight relaxation.

2) Robustness of PACE2D#: To test the robustness of PACE2D#, we use a different data generation procedure to enable the use of ROBIN. We first generate $K$ octahedra, center-aligned at the origin, with vertices sampled component-wise in $[0.5, 2]$ m. The use of octahedra (convex shapes with known face planar equations) allows us to solve for sets of feasible winding orders—following Corollary 14—using linear programs (see Appendix H in [34]). We sample shape parameters $c$ uniformly at random in $[0, 1]^K$. Then, we draw random poses $(R, t)$ such that the resulting camera locations lie on a sphere centered at the origin with radius of 3 m. For each camera location, we randomly sample $b_i^k$, $i = 1, \ldots, N$, from each octahedron such that $\langle r \cdot N \rangle$ of them lie on the weighted octahedron’s visible faces where $r$ is the outlier ratio. For the remaining $b_i^k$, we sample them from the nonvisible faces of the octahedron. We generate the measurements $p_{2D,i}$ according to (3), where the noise $\epsilon_{2D,i}$ follows $\mathcal{N}(0, \sigma^2 I_2)$ with $\sigma = 0.01$. For the $p_{2D,i}$ generated by nonvisible $b_i^k$’s, we replace their noise term $\epsilon_{2D,i}$ with arbitrary 2-D points violating the generative model (3). The regularization factor $\lambda$ is set to 0.01.

We consider PACE2D#, as well as two variants: GNC-pace2d* (where GNC is applied to PACE2D* without ROBIN), and Clique-pace2d* (where PACE2D* is applied after ROBIN without GNC). For PACE2D# and GNC-pace2d*, inlier threshold $\beta_{2D}$ is set to be 0.05. We also compare against: 1) Zhou-Robust, which is a robust version of Zhou’s solver from [71]; 2) MS-PnP; 3) RANSAC-MS-PnP, where we wrap MS-PnP in a four-point RANSAC loop with inlier threshold of 0.05; 4) Clique-MS-PnP, where we apply ROBIN before MS-PnP; and 5) KPnP-Robust, where we apply ROBIN and GNC before KPnP. Fig. 8(b) plots the results under increasing outlier rates. PACE2D# is robust to 10% of outliers and achieves low median errors for outlier rates below 30%, while GNC-pace2d*, MS-PnP, RANSAC-MS-PnP, and Zhou-Robust already exhibit large median errors at 10% outlier rates. Interestingly, at around 10%, Clique-pace2d* remains robust, while GNC-pace2d* starts to show failures. This remarks the effectiveness of ROBIN in filtering out outliers, also shown in the clique inlier rate plot in Fig. 8(c). Similar to PACE3D#, ROBIN improves the convergence rate of GNC: see runtime curves of PACE2D# and GNC-pace2d* in Fig. 8(c). KPnP-Robust fails to obtain low median errors due to its assumption of one-hot $c$. RANSAC-MS-PnP, while degrading more gracefully at high outlier rates, is unable to achieve low median errors comparable to PACE2D#. Its inner four-point MS-PnP solver produces estimates that have large residuals for inliers (note the discrepancy between MS-PnP and RANSAC-MS-PnP at 0% outlier rate), hence affecting its inlier set estimation. Fig. 8(d) reports qualitative results on an simulated instance with 30% outlier rate. These results underline that 2-D–3-D category-level perception is a much harder problem compared with its 3-D–3-D counterpart. Our algorithms, while being competitive against baselines, are still slow and only robust to a small fraction of outliers. One may consider using RANSAC with PACE2D*; unfortunately, the slow runtime of PACE2D* prohibits such implementation: a mere 100 iterations of PACE2D* at $K = 5$ and $N = 8$ takes around 50 min. This highlights another benefit of ROBIN: it does not require solving the underlying problem; hence, it can improve robustness at a significant runtime advantage, more so if the underlying solver is slow. We also report extra results in Appendix P-C in [34].

C. Vehicle Pose and Shape Estimation on ApolloScape

1) Setup and Baselines: We evaluate PACE3D# and PACE2D# on the ApolloScape dataset [30], [61]. The ApolloScape self-driving dataset is a large collection of multimodal data collected in four different cities in China under
Fig. 8. Performance of PACE2D* and PACE2D# compared with baselines in simulated experiments. (a) PACE2D* compared with KPnP, MS-PnP, Zhou, and Shape* on synthetic outlier-free data with varying number of shapes, where $c$ is sampled uniformly at random from $\Delta_K$. (b) PACE2D# and variants compared with KPnP-Robust, MS-PnP, RANSAC-MS-PnP, Clique-MS-PnP, and Zhou-Robust on synthetic outlier-contaminated data with varying outlier rates. (c) Qualitative results of MS-PnP, Zhou-Robust, GNC-PACE2D*, and PACE2D# on an instance with 30% outlier rate. Blue meshes represent the ground-truth shape, and yellow meshes represent the pose and shape estimated by each model. Red rays indicate outliers (bearing vectors originated from the camera center). In this case, PACE2D# succeeds, while the other methods fail. (a) Performance of PACE2D*(OH) and PACE2D* on outlier-free synthetic data: $N = 8$, $c$ sampled from $\Delta_K$ uniformly at random. (b) Robustness of PACE2D# against increasing outliers on synthetic data: $N = 10$, $K = 3$; $c$ sampled from $\Delta_K$ uniformly at random. (c) Qualitative results of MS-PnP, Zhou-Robust, GNC-PACE2D*, and PACE2D# on a test instance with 30% outlier rate.

Fig. 9. We pass ground-truth-annotated keypoints to ROBIN, using a winding order dictionary generated from ray tracing. Green dots represent inliers accepted by ROBIN. Red dots represent outliers rejected by ROBIN. In these four examples, ROBIN correctly rejects mislabeled keypoints. Left: #8 and #49 are switched (with respect to the CAD models). Right: #61 and #60 are switched. See more examples in Appendix Q in [34].

We compare PACE3D# and PACE2D# against DeepMANTA [17], 3-D-RCNN [20], and GSNet [18], three recent state of the art methods for 3-D vehicle pose estimation. For our experiments, we use the official splits of the ApolloCar3D dataset. Namely, we use the validation split (200 images) for all the quantitative experiments shown below, consistent with the evaluation setups reported in other baseline methods. We use the 2-D semantic keypoints extracted by GSNet [18] as measurements for PACE2D#. We use the pretrained weights from [18] and reject keypoints with confidence less than 0.05. For PACE3D# and IRLS-GM, we additionally retrieve the corresponding depth from the depth images provided by ApolloScape for each 2-D semantic keypoint. For PACE2D#, we construct the dictionary of feasible winding orders by recording the observed winding orders for all keypoint triplets in the training set, as well as by performing ray tracing to keypoints in a volume surrounding each CAD model (see Appendix H in [34]). Notably, when we applied ROBIN on the ground-truth annotations, we discovered multiple mislabeled keypoints (see Fig. 9 and Appendix Q in [34]). This suggests that ROBIN may also be helpful in terms of verifying datasets.

While ApolloCar3D provides 2-D semantic keypoint annotations, it does not provide the corresponding 3-D keypoint annotations on the CAD models. Hence, we manually labeled the 66 3-D semantic keypoints on the 79 models and provide them as the shape library to PACE3D#. For PACE2D#, we instead select three random models, including the ground-truth model,
as a shape library, since using the entire set leads to prohibitive runtime. We use $\lambda = 0.5$ and $\beta_{D} = 0.15$ in PACE3D#, and $\lambda = 0.001$ and $\beta_{D} = 0.01$ in PACE2D#. IRLS-GM utilizes an inlier threshold of 0.15, same as PACE3D#.

2) Results: Table I shows the performance of PACE3D# and PACE2D# against various baselines (qualitative results can be found in Appendix P-D in [34]). We use two metrics called A3DP-Rel and A3DP-Abs (for both, the higher the better) following the same definitions in [61]. They are measures of precision with thresholds jointly considering translation, rotation, and 3-D shape similarity between estimated cars and ground truth. A3DP-Abs uses absolute translation thresholds, whereas A3DP-Rel uses relative ones. A total of ten thresholds are used, of which c-l represents a loose criterion (2.8 m, $\pi/6$ rad, and 0.5 for translation, rotation, and shape) and c-s represents a strict criterion (1.4 m, $\pi/12$ rad, and 0.75 for translation, rotation, and shape). The mean column represents the average A3DP-Abs/Rel over all thresholds.

Overall, PACE2D# achieves performance comparable but slightly inferior to learning-based approaches in A3DP-Rel, while PACE3D# excels in both A3DP-Rel and A3DP-Abs. The main failure mode of PACE2D# is in its translation estimation: over 98% of the failures do not meet the translation threshold only, and the translation estimation accuracy degrades with the increase in distance between the vehicle and the camera. PACE3D# outperforms the baselines in terms of the mean and c-s criteria; this is partially expected since we use depth information, which is not available to the other methods at test time. In terms of the strict criterion c-s, PACE3D# outperforms competitors by a large amount, confirming that it can retrieve highly accurate estimates. When using the loose criterion c-l, GSNNet is slightly better than PACE3D#, suggesting that learning-based techniques may have more graceful failure modes.

3) Runtime: Table II shows the timing breakdown for PACE3D# and PACE2D#. We also report the timing for the GSNNet keypoint detection from [18] for completeness. For PACE3D#, the max-clique pruning is written in C++ and has negligible runtime, while GNC is implemented in Python. For PACE2D#, both the maximum hyperclique estimation and GNC are written in Python. PACE3D# is significantly faster than PACE2D# thanks to PACE3D#‘s compact semidefinite relaxation. While PACE2D# is currently slow for real-world applications, an optimized implementation of PACE3D# is amenable to practical applications.

IX. RELATED WORK

This section reviews related work on category-level perception and outlier-robust estimation.

A. Category-Level Perception

Early approaches for category-level perception focus on 2-D problems, where one has to locate objects—from human faces [73] to resistors [13]—in images. Classical approaches include active contour models [74], [75] and active shape models [13], [76]. These works use techniques like principal component analysis to build a library of 2-D landmarks from training data and then use iterative optimization algorithms to estimate the 2-D object locations in the images, rather than estimating 3-D poses.

The landscape of category-level perception has been recently reshaped by the rapid adoption of convolutional networks [77], [78], [79]. Pipelines using deep learning have seen great successes in areas such as human pose estimation [6], [80], [81], [82], [83] and pose estimation of household objects [1], [2], [3]. With the growing interest in self-driving vehicles, research has also focused on jointly estimating vehicle pose and shape [17], [18], [19], [20], [21].

For methods that aim to recover both the pose and shape of objects, a common paradigm is to use end-to-end methods. Usually, an encoder–decoder network is used to first convert input images to some latent representations and then map the latent representation back to 3-D estimates (e.g., 3-D bounding boxes, or pose and shape estimates). For example, Richter and Roth [84] predict 3-D shapes through an efficient 2-D encoding. Groueix et al. [85] represent shapes as collections of parametric surface elements. Tatarchenko et al. [86] generate 3-D shapes through an octree representation. Burchfiel and Konidaris [87] train CNNs with generative representations of 3-D objects to predict probabilistic distributions of object poses. An additional alignment loss can also be incorporated into the network to directly regress a pose [88], [89], [90]. Wen et al. [91] design a network with a loss function over SE(3) to regress relative poses. One drawback of such approaches is that it is difficult for neural networks to learn the necessary 3-D structure of the object on a per-pixel basis; moreover, end-to-end approaches typically require 3-D pose labels that might be difficult (or expensive) to obtain for real data. As shown by Tatarchenko et al. [92], such networks can be outperformed by methods trained on model recognition and retrieval only. Alternative methods circumvent pose and shape estimation and directly regress 3-D semantic keypoints for manipulation [1] or dense visual descriptors [93].

Multistage methods constitute another major paradigm for category-level perception. Such approaches commonly include a neural-network-based front end that extracts features from input data (such as RGB or RGB-D images) [3], [94] and an optimization-based back end that recovers the 3-D pose of the object given the features [3], [95], [96], [97], [98]. The front end may predict positions of semantic keypoints [3] or feature embeddings [94] and generate correspondences from those features. In early works, Lim et al. [99] establish 2-D–3-D correspondences between images and textureless CAD models by using histogram of oriented gradients descriptors on images and rendered edgemaps of the CAD models. Chabot et al. [17]
regress a set of 2-D part coordinates and then choose the best corresponding 3-D template. Pavlakos et al. [3] use a stacked hourglass neural network [81] for 2-D semantic keypoint detection. In other works, a canonical category-level coordinate space is predicted for each detection, from which correspondences are generated [23], [24], [100], [101]. Our work belongs to the class of multistage methods. In particular, we use [18] as our front end and develop optimal and robust back-end solvers.

Research effort has also been devoted to developing more robust and efficient back-end solvers given 2-D or 3-D features extracted by the front end. The back-end solvers recover the 3-D pose (and possibly the shape) of the object by solving an optimization problem [3], [95], [96], [97], [98]. Depending on the input modalities, back-end solvers can be roughly divided into 2-D–3-D or 3-D–3-D solvers, where the former use 2-D inputs only, and the latter incorporate additional depth information. A number of related works investigate 2-D–3-D back-end solvers [3], [14], [32], [71], [81], [102], [103]. Hou et al. [98] defer the task of shape estimation to a neural network and use EPnP [70] to solve for the object bounding box’s pose only. Zhou et al. [14], [71] propose a convex relaxation to jointly optimize 3-D shape parameters and object pose from 2-D keypoints. However, the relaxation assumes a weak perspective camera model, which might lead to poor results if the object is close to the camera. Yang and Carlone [15] apply the moment/sums-of-squares hierarchy [27], [104] to develop tighter relaxations than [14], still under the assumption of a weak perspective model. Schmackpeper et al. [32] use a local solver with a full perspective camera model. Our work differs from [14], [15], [32], and [71] since we propose a certifiably optimal solver for the full perspective case, using an algebraic point-to-line cost.

3-D–3-D back-end solvers have been investigated in the robotics literature [105], [106], [107]. In robotics applications, such as manipulation and self-driving, depth information is readily available either via direct sensing (e.g., RGB-D or stereo) or algorithms (e.g., mono depth techniques [108], [109]), so the requirements of depth is not too constraining. Wang et al. [110] decouple shape from pose estimation by predicting category-specific keypoints and use Arun’s method [33] for estimating frame-by-frame relative pose. Wen and Belkis [105] view category-level object detection and tracking as a pose graph optimization problem, solving 3-D registration of keypoints across frames and then jointly optimizing the pose graph online. Deng et al. [94] use nonlinear optimization and alternate between optimizing shape size and pose. In this and our previous work [31], we propose the first 3-D–3-D certifiably optimal solver that runs in a fraction of a second even in the presence of thousands of CAD models.

B. Robust Estimation

We review three robust estimation paradigms: M-estimation, consensus maximization, and graph-based outlier pruning.

M-estimation performs estimation by optimizing a robust cost function that reduces the influence of outliers. The resulting problems are typically optimized using iterative local solvers. Tavish and Barfoot [63] compare several robust costs using iterative reweighted least squares solvers. The downside of local solvers is that they need a good initial guess, which is often unavailable in practice. A popular approach to circumvent the need for an initial guess is GNC [58], [111]. Zhou et al. [112] use GNC for point cloud registration. Yang et al. [29] and Antonante et al. [113] combine GNC with global nonminimal solvers and show their general applicability to problems with up to 80% outliers.

For certain low-dimensional geometric problems, fast global solvers exist. Enqvist et al. [114] use a TLS cost to solve triangulation in polynomial time in the number of measurements, but exponential time in the dimension of the to-be-estimated state $x$. Ask et al. [115] use a TLS cost for image registration. Recently, certifiable algorithms have been developed to solve outlier-robust estimation problems with a posteriori optimality guarantees [7], [48], [57], [116], [117]. They rely on Lasserre’s hierarchy of moment relaxations to obtain convex relaxations of robust estimation problems [57], [116]. They compute an estimate together with a certificate of optimality (or a bound on the suboptimality gap), based on the rank of the SDP solution or the duality gap. Brynte et al. [118] categorize cases where such relaxations are always tight and study the failure cases. Unfortunately, current SDP solvers have poor scalability, and such methods are mostly viable to check optimality [7], [116].

Consensus maximization is a framework for robust estimation that aims to find the largest set of measurements with errors below a user-defined threshold. Consensus maximization is NP-hard [113], [119]; hence, the community has resorted to randomized approaches, such as RANSAC [120]. RANSAC repeatedly draws a minimal subset of measurements from which a rough estimate is computed, and the algorithm stops after finding an estimate that agrees with a large set of measurements. While RANSAC works well for problems where the minimal set is small and there are not many outliers, the average number of iterations it requires increases exponentially with the percentage of outliers [121], making it impractical for problems with many outliers. On the other hand, global solvers, such as BNb [122] and tree search [123], exist but scale poorly with the problem size, with BNb being exponential in the size of $x$, and tree search being exponential in the number of outliers [123].

Graph-based outlier pruning methods aim at discarding gross outliers from the set of measurements. These methods do not necessarily reject all the outliers; hence, they are often used as a preprocessing for M-estimation or maximum consensus [7], [28]. Outlier pruning methods detect outliers by analyzing a compatibility graph, where vertices represent data points and edges represent predefined compatibility measures between data points [28]. Bailey et al. [124] propose a maximum common subgraph algorithm for feature matching in LiDAR scans. Segundo and Artieda [125] build an association graph and find the maximum clique for 2-D image feature matching. Perera and Barnes [126] segment objects under rigid body motion with a clique formulation. Leordeanu and Hebert [127] establish image matches by finding strongly connected clusters in the correspondence graph with an approximate spectral method. Enqvist et al. [37] develop an outlier rejection algorithm for 3-D–3-D and 2-D–3-D registration based on approximate vertex cover. Recent progress in graph algorithms (see, e.g., [36] and [128]) has led to fast graph-theoretic outlier pruning algorithms that are robust to extreme outlier rates; see, e.g., TEASER+++ [7].

In this work, we generalize graph-based methods to use hypergraphs: while a standard graph only contains edges connecting pairs of nodes (which represent compatibility tests in our context), each edge in a hypergraph may connect an arbitrary subset of vertices. Hypergraphs have been studied in the context of network learning, robotics, and computer vision. Torres-Jimenez et al. [129] develop an exact algorithm for finding maximum cliques in uniform hypergraphs. Shun [130] develops a collection of fast parallel hypergraph algorithms for
large-scale networks. Srinivasan et al. [131] develop a framework for hypergraph representation learning. Rube and Wong [132] formulate free space as a hypergraph for robot path planning. Du et al. [133] represent humans as a hypergraph for visual tracking. Yu et al. [134] treat image classification as a hypergraph edge weight optimization problem. In our work, we build upon [28] by extending the definition of compatibility graphs from simple graphs to hypergraphs.

X. CONCLUSION

In this article, we proposed PACE2D* and PACE3D*, the first certifiably optimal solvers for the estimation of the pose and shape of 3-D objects from 2-D and 3-D keypoint detections, respectively. While existing iterative methods get stuck in local minima corresponding to poor estimates, PACE2D* and PACE3D* leverage tight SDP relaxations to compute certifiably optimal estimates. We then designed a general framework for graph-theoretic outlier pruning, named ROBIN, that extends our original proposal in [28] to operate on compatibility hypergraphs. We showed that ROBIN can be effectively applied to 2-D and 3-D category-level perception and is able to prune a large fraction of outliers. The combination of ROBIN and our optimal solvers (PACE2D* and PACE3D*) leads to PACE2D** and PACE3D**, which are outlier-robust algorithms for 3-D–3-D and 2-D–3-D pose and shape estimation. While PACE2D** is currently slow and is sensitive to the quality of the keypoint detections, PACE3D** largely outperforms the state of the art and a nonoptimized implementation runs in a fraction of a second.

ACKNOWLEDGMENT

The authors would like to thank the editor and the anonymous reviewers for their constructive feedback, Rajat Talak for discussion about pose estimation networks, and Charleen Tan for labeling keypoints.

REFERENCES

[1] L. Manuelli, W. Gao, P. Florence, and R. Tedrake, “KPAM: Keypoint affordances for category-level robotic manipulation,” in Proc. Int. Symp. Robot. Res., 2019, pp. 132–157.

[2] W. Gao and R. Tedrake, “KPAM 2.0: Feedback control for category-level robotic manipulation,” IEEE Robot. Autom. Lett., vol. 6, no. 2, pp. 2962–2969, Apr. 2021.

[3] G. Pavlakos, X. Zhou, A. Chan, K. Derpanis, and K. Daniilidis, “6-DoF object pose from semantic keypoints,” in IEEE Int. Conf. Robot. Autom., 2017, pp. 2011–2018.

[4] P. Mc Causland, Self-Driving Uber Car that Hit and Killed Woman did not Recognize that Pedestrians Jaywalk, NBC News, New York, NY, USA, Nov. 2019.

[5] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, “You only look once: Unified, real-time object detection,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit., 2016, pp. 779–788.

[6] K. He, G. Gkioxari, P. Dollár, and R. Girshick, “Mask R-CNN,” in Proc. Int. Conf. Comput. Vis., 2017, pp. 2980–2988.

[7] H. Yang, J. Shi, and L. Carlone, “TEASER: Fast and certifiable point cloud registration,” IEEE Trans. Robot., vol. 37, no. 2, pp. 314–333, Apr. 2021.

[8] B. K. P. Horn, “Closed-form solution of absolute orientation using unit quaternions,” J. Opt. Soc. Amer., vol. 4, no. 4, pp. 629–642, 1987.

[9] A. P. Huston and T. J. Chin, “Guaranteed outlier removal for point cloud registration with correspondences,” IEEE Trans. Pattern Anal. Mach. Intell., vol. 40, no. 12, pp. 2868–2882, Dec. 2018.

[10] Y. Zheng, Y. Kuang, S. Sugimoto, K. Astrom, and M. Okutomi, “Revisiting the PnP problem: A fast, general and optimal solution,” in Proc. Int. Conf. Comput. Vis., 2013, pp. 2344–2351.

[11] L. Kneip, H. Li, and Y. See, “UPnP: An optimal O(n) solution to the absolute pose problem with universal applicability,” in Proc. Eur. Conf. Comput. Vis., 2014, pp. 127–142.

[12] G. Schweighofer and A. Pinz, “Globally optimal O(n) solution to the PnP problem for general camera models,” in Proc. Brit. Mach. Vis. Conf., 2008, pp. 1–10.

[13] T. F. Cootes, C. J. Taylor, D. H. Cooper, and J. Graham, “Active shape models—Their training and application,” Comput. Vis. Image Understanding, vol. 61, no. 1, pp. 38–59, 1995.

[14] X. Zhou, S. Leonards, X. Hu, and K. Daniilidis, “3D shape reconstruction from 2D landmarks: A convex formulation,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit., 2015, pp. 4447–4455.

[15] H. Yang and L. Carlone, “In perfect shape: Certifiably optimal 3D shape reconstruction from 2D landmarks,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit., 2020, pp. 621–630.

[16] N. Kolotouros, G. Pavlakos, and K. Daniilidis, “Convolutional mesh regression for single-image human shape reconstruction,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit., 2019, pp. 4501–4510.

[17] F. Chabot, M. Chauve, J. Rabarisoa, C. Teuliere, and T. Chateau, “Deep MANTA: A coarse-to-fine many-task network for joint 2D and 3D vehicle analysis via monocular image,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit., 2017, pp. 2040–2049.

[18] L. Ke, S. Li, Y. Sun, Y.-W. Tai, and C.-K. Tang, “GSNet: Joint vehicle pose and shape reconstruction with geometrical and scene-aware supervision,” in Proc. Eur. Conf. Comput. Vis., 2020, pp. 515–532.

[19] J. G. López, A. Agudo, and F. Moreno-Noguer, “Vehicle pose estimation via regression of semantic points of interest,” in Proc. Int. Symp. Image Signal Process. Anal., 2019, pp. 209–214.

[20] A. Kundu, Y. Li, and J. M. Rehg, “3D-RCNN: Instance-level 3D object reconstruction via render-and-compare,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit., 2018, pp. 3559–3568.

[21] S. Suwajanakorn, N. Snavely, J. Tompson, and M. Norouzi, “Discovery of latent 3D keypoints via end-to-end geometric reasoning,” in Proc. 32nd Int. Conf. Neural Inf. Process. Syst., 2018, pp. 2063–2074.

[22] C. Sahin and T.-K. Kim, “Category-level 6D object pose recovery in depth images,” in Proc. Eur. Conf. Comput. Vis., 2018, pp. 665–681.

[23] 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 Proc. IEEE Conf. Comput. Vis. Pattern Recognit., 2019, pp. 2642–2651.

[24] X. Li, H. Wang, L. Yi, L. Guibas, A. L. Abbott, and S. Song, “Category-level articulated object pose estimation,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit., 2020, pp. 3706–3715.

[25] J. Wang, K. Chen, and Q. Dou, “Category-level 6D object pose estimation via cascaded relation and recurrent reconstruction networks,” in Proc. IEEE/RSJ Int. Conf. Intell. Robots Sci. Syst., 2021, pp. 4807–4814.

[26] X. Li et al., “Leveraging SE(3) equivariance for self-supervised category-level 6D pose estimation,” in Proc. Int. Conf. Neural Inf. Process. Syst., 2021, pp. 1–12.

[27] J. B. Lasserre, “Global optimization with polynomials and the problem of moments,” SIAM J. Optim., vol. 11, no. 3, pp. 796–817, 2001.

[28] J. Shi, H. Yang, and L. Carlone, “ROBIN: A graph-theoretic approach to reject outliers in robust estimation using invariants,” in Proc. IEEE Int. Conf. Robot. Autom., 2021, pp. 13820–13827.

[29] H. Yang, P. Antonante, V. Tzoumas, and L. Carlone, “Graduated non-convexity for robust spatial perception: From non-minimal solvers to global outlier rejection,” IEEE Robot. Autom. Lett., vol. 5, no. 2, pp. 1127–1134, Apr. 2020.

[30] P. Wang, X. Huang, X. Cheng, D. Zhou, Q. Geng, and R. Yang, “The ApolloScape open dataset for autonomous driving and its application,” IEEE Trans. Pattern Anal. Mach. Intell., vol. 42, no. 10, pp. 2702–2719, Oct. 2020.

[31] J. Shi, H. Yang, and L. Carlone, “Optimal pose and shape estimation for category-level 3D object perception,” in Proc. Robot. Sci. Syst., Jul. 2021, doi: 10.15607/RSS.2021.XVII.025.

[32] K. Schmeckpeper et al., “Semantic keypoint-based pose estimation from single RGB frames,” Field Robot., vol. 2, pp. 147–171, 2022, doi: 10.55417/fr.2022006.

[33] K. Arun, T. Huang, and S. Blostein, “Least-squares fitting of two 3-D point sets,” IEEE Trans. Pattern Anal. Mach. Intell., vol. PAMI-9, no. 5, pp. 698–700, Sep. 1987.
[34] J. Shi, H. Yang, and L. Carlone, “Optimal and robust category-level perception: Object pose and shape estimation from 2D and 3D semantic keypointsetss,” 2022, arXiv:2206.12498.

[35] S. Diamond and S. Boyd, “CVXPY: A Python-embedded modeling language for convex optimization,” J. Mach. Learn. Res., vol. 17, no. 83, pp. 1–5, 2016.

[36] R. A. Rossi, D. F. Gleich, and A. H. Gebremedhin, “Parallel maximum clique algorithms with applications to network analysis,” SIAM J. Sci. Comput., vol. 37, no. 5, pp. C589–C616, 2015.

[37] O. Enqvist, K. Josephson, and F. Kahl, “Optimal correspondences from pairwise constraints,” in Proc. Int. Conf. Comput. Vis., 2009, pp. 1295–1302.

[38] J. Mundy and A. Zisserman, Geometric Invariance in Computer Vision. Cambridge, MA, USA: MIT Press, 1992.

[39] D. Eberly, 3D Game Engine Design: A Practical Approach to Real-Time Computer Graphics. Boca Raton, FL, USA: CRC, 2006.

[40] F. Thomas et al., Computer Graphics: Principles and Practice. London, U.K.: Pearson Educ., 2014.

[41] A. Tikhonov, A. Goncharsky, V. Stepanov, and A. Yagola, Numerical Methods for the Solution of Ill-Posed Problems, 2nd ed. New York, NY, USA: Springer, 2013.

[42] D. Rosen, L. Carlone, A. Bandeira, and J. Leonard, “SE-Sync: A certifiably correct algorithm for synchronization over the special Euclidean group,” Int. J. Robot. Res., vol. 38, pp. 95–125, 2018.

[43] Y.-L. Lin, V. I. Morariu, W. H. Hsu, and L. S. Davis, “Jointly optimizing 3D model fitting and fine-grained classification,” in Proc. Eur. Conf. Comput. Vis., 2014, pp. 466–480.

[44] L. Gu and T. Kanade, “3D alignment of face in a single image,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit., vol. 1, 2006, pp. 1305–1312.

[45] V. Ramakrishna, T. Kanade, and Y. Sheikh, “Reconstructing 3D human pose from 2D image landmarks,” in Proc. Eur. Conf. Comput. Vis., 2012, pp. 573–586.

[46] R. Tron, D. Rosen, and L. Carlone, “On the inclusion of determinant constraints in Lagrangian duality for 3D SLAM,” in Proc. Robot.: Sci. Syst., Workshop, 2015, vol. 4.

[47] The MOSEK Optimization Toolbox for MATLAB Manual Version 8.1, Moskow Aps, Copenhagen, Denmark, 2017. [Online]. Available: http://docs.mosek.com/8.1/toolbox/index.html

[48] H. Yang and L. Carlone, “A qaternion-based certifiably optimal solution to the Wahba problem with outliers,” in Proc. Int. Conf. Comput. Vis., 2019, pp. 1665–1674.

[49] J. Biale and J. Gonzalez-Jimenez, “Fast global optimality verification in 3D SLAM,” in Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst., 2016, pp. 4630–4636.

[50] E. Eriksson, C. Olsson, F. Kahl, and T.-J. Chin, “Rotation averaging and strong duality,” Proc. IEEE Conf. Comput. Vis. Pattern Recognit., 2018, pp. 127–135.

[51] B. Babanezhad, “A note on probably certifiably correct algorithms,” Comptes Rendus Mathematique, vol. 354, no. 3, pp. 329–333, 2015, doi: 10.1016/j.crma.2015.11.009.

[52] S. Schaible and J. Shi, “Fractional programming: The sum-of-ratios case,” Optim. Methods Softw., vol. 18, no. 2, pp. 219–229, 2003.

[53] R. Hartley and F. Kahl, “Global optimization through rotation space search,” Int. J. Comput. Vis., vol. 82, no. 1, pp. 64–79, 2009.

[54] C. Olsson, F. Kahl, and M. Oskarsson, “Optimal estimation of perspective camera pose,” in Proc. Int. Conf. Comput. Vis. Pattern Recognit., 2006, pp. 5–8.

[55] H. Yang, C. Doran, and T. J. Slone, “Dynamical pose estimation,” in Proc. Int. Conf. Comput. Vis., 2021, pp. 5926–5935.

[56] P. A. Parrilo, “Semidefinite programming relaxations for semialgebraic problems,” Math. Comput., vol. 76, no. 2, pp. 929–320, 2003.

[57] H. Yang and L. Carlone, “Certifiably optimal outlier-robust geometric perception: Semidefinite relaxations and scalable global optimization,” IEEE Trans. Pattern Anal. Mach. Intell., vol. 45, no. 3, pp. 2816–2834, Mar. 2023.

[58] A. Blake and A. Zisserman, Visual Reconstruction. Cambridge, MA, USA: MIT Press, 1987.

[59] Y. Xiang, R. Motaghi, and S. Savarese, “Beyond PASCAL: A benchmark for 3D object detection in the wild,” in Proc. IEEE Winter Conf. Appl. Comput. Vis., 2014, pp. 75–82.

[60] Y. You et al., “KeypointNet: A large-scale 3D keypoint dataset aggregated from numerous human annotations,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit., 2020, pp. 13647–13656.

[61] X. Song et al., “ApolloCar3D: A large 3D car instance understanding benchmark for autonomous driving,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit., 2019, pp. 5452–5462.
Jingnan Shi (Member, IEEE) received the B.S. degree in aeronautics and astronautics from Harvey Mudd College, Claremont, CA, USA, in 2019, and the M.S. degree in aeronautics and astronautics in 2021 from the Massachusetts Institute of Technology, Cambridge, MA, USA, where he is currently working toward the Ph.D. degree in aeronautics and astronautics with the Department of Aeronautics and Astronautics.

His research interests include robust perception and self-supervised learning with applications to robotics.

Mr. Shi was a Best Paper Finalist at 2021 Robotics: Science and Systems Conference and a recipient of the MathWorks Fellowship.

Heng Yang received the B.Eng. degree in automotive engineering from Tsinghua University, Beijing, China, in 2015, and the M.S. and Ph.D. degrees in mechanical engineering from the Massachusetts Institute of Technology, Cambridge, MA, USA, in 2017 and 2022, respectively.

He is currently an Assistant Professor of Electrical Engineering with the School of Engineering and Applied Sciences, Harvard University, Cambridge, MA. His research interests include algorithmic foundations of robot perception, decision making, and learning, with focus on bringing large-scale convex optimization, semidefinite relaxation, statistics, and machine learning to safe and trustworthy autonomy.

Dr. Yang was a recipient of the Best Paper Award at 2020 International Conference on Robotics and Automation, a Best Paper Award Honorable Mention from IEEE ROBOTICS AND AUTOMATION LETTERS in 2020, and a Best Paper Finalist at 2021 Robotics: Science and Systems Conference (RSS). He is a Class of 2021 RSS Pioneer.

Luca Carlone (Senior Member, IEEE) received the B.S. and S.M. degrees in mechatronics in 2006 and 2008, respectively, and the Ph.D. degree in robotics in 2012, all from the Polytechnic University of Turin, Turin, Italy, and the S.M. degree in automation engineering from the Polytechnic University of Milan, Milan, Italy, in 2008.

He is currently the Boeing Career Development Associate Professor with the Department of Aeronautics and Astronautics, Massachusetts Institute of Technology, Cambridge, MA, USA, and a Principal Investigator with the Laboratory for Information and Decision Systems (LIDS). In 2015, he joined LIDS as a Postdoctoral Associate and later as a Research Scientist in 2016, after two years as a Postdoctoral Fellow with the Georgia Institute of Technology, Atlanta, GA, USA, from 2013 to 2015. His work includes seminal results on certifiably correct algorithms for localization and mapping, as well as approaches for visual-inertial navigation and distributed mapping. His research interests include nonlinear estimation, numerical and distributed optimization, and probabilistic inference, applied to sensing, perception, and decision making in single- and multirobot systems.

Dr. Carlone was a recipient of the 2017 and 2022 IEEE TRANSACTIONS ON ROBOTICS King-Sun Fu Memorial Best Paper Award, the Best Student Paper Award at 2021 IEEE/RSJ International Conference on Intelligent Robots and Systems, the Best Paper Award in Robot Vision at 2020 International Conference on Robotics and Automation, a 2020 Honorable Mention from IEEE ROBOTICS AND AUTOMATION LETTERS, a Track Best Paper Award at 2021 IEEE Aerospace Conference, the Best Paper Award at 2016 Workshop on the Algorithmic Foundations of Robotics, and the Best Student Paper Award at 2018 Symposium on VLSI Circuits. He was the Best Paper Finalist at 2015 Robotics: Science and Systems Conference (RSS), RSS 2021, and 2023 IEEE CVF Winter Conference on Applications of Computer Vision. He was also a recipient of the AIAA Aeronautics and Astronautics Advising Award in 2022, the NSF CAREER Award in 2021, the RSS Early Career Award in 2020, the Google Daydream Award in 2019, the Amazon Research Award in 2020 and 2022, and the MIT AeroAstro Vickie Kerrebrock Faculty Award in 2020. He is an Associate Fellow of the American Institute of Aeronautics and Astronautics.