Certifiable Outlier-Robust Geometric Perception: Exact Semidefinite Relaxations and Scalable Global Optimization

Heng Yang, *Student Member, IEEE,* and Luca Carlone, *Senior Member, IEEE*

Abstract—We propose the first general and scalable framework to design *certifiable* algorithms for robust geometric perception in the presence of outliers. Our first contribution is to show that estimation using common robust costs, such as truncated least squares (TLS), maximum consensus, Geman-McClure, Tukey's biweight, among others, can be reformulated as *polynomial optimization problems* (POPs). By focusing on the TLS cost, our second contribution is to exploit *sparsity* in the POP and propose a sparse *semidefinite programming* (SDP) relaxation that is much smaller than the standard Lasserre's hierarchy while preserving *exactness*, *i.e.*, the SDP recovers the optimizer of the nonconvex POP with an *optimality certificate*. Our third contribution is to solve the SDP relaxations at an unprecedented scale and accuracy by presenting STRIDE, a solver that blends *global descent* on the convex SDP with fast *local search* on the nonconvex POP. Our fourth contribution is an evaluation of the proposed framework on six geometric perception problems including single and multiple rotation averaging, point cloud and mesh registration, absolute pose estimation, and category-level object pose and shape estimation. Our experiments demonstrate that (i) our sparse SDP relaxation is exact with up to 60%–90% outliers across applications; (ii) while still being far from real-time, STRIDE is up to 100 times faster than existing SDP solvers on medium-scale problems, and is the only solver that can solve large-scale SDPs with hundreds of thousands of constraints to high accuracy; (iii) STRIDE provides a safeguard to existing fast heuristics for robust estimation (*e.g.*, RANSAC or Graduated Non-Convexity), *i.e.*, it certifies global optimality if the heuristic estimates are optimal, or detects and allows escaping local optima when the heuristic estimates are suboptimal.

Index Terms—certifiable algorithms, outlier-robust estimation, robust fitting, robust estimation, polynomial optimization, semidefinite programming, global optimization, moment/sums-of-squares relaxation, large-scale convex optimization

1 Introduction

Geometric perception, the task of estimating unknown geometric models (*e.g.*, object poses, rotations, 3D structure, robot trajectory) from sensor measurements (*e.g.*, images, point clouds, relative poses), is a fundamental problem in computer vision and robotics. It finds extensive applications to object detection and localization [94], motion estimation and 3D reconstruction [32], simultaneous localization and mapping (SLAM) [73] and structure from motion (SfM) [75], virtual and augmented reality [55], and medical imaging [9], to name a few.

A modern machine perception pipeline includes a *perception front-end* that extracts, describes, and matches relevant features from raw sensor data, and a *perception back-end* that estimates the geometric models of interest given the putative feature matches. In practice, due to various sources of imperfections and uncertainties (*e.g.*, sensor failures, incorrect detections and matchings by hand-crafted or deep-learned features), a large amount of *outliers* —measurements that tell no or little information about the underlying geometric models— are generated by the front-end. Therefore, designing an *outlier-robust* back-end that can tolerate large amounts of outliers, also known as *robust fitting* [31] in computer vision and *robust state estimation* [11] in robotics, has been a longstanding quest in both communities.

Unfortunately, from a theoretical standpoint, performing robust estimation by discerning *inliers* (*i.e.*, the correct and useful measurements) from outliers, is known to be NP-hard and *inapproximable* due to the combinatorial nature of the problem [7], [31]. Consequently, existing algorithms for outlier-robust geometric perception are mostly divided into *fast heuristics*, *e.g.*, RANSAC [41] and graduated non-convexity (GNC) [16], [17], [93], that are efficient but offer no optimality guarantees, and *global solvers*, *e.g.*, Branch-and-Bound [102] and mixed-integer programming [50], that guarantee optimality but run in worst-case exponential time. Although in some cases it is acceptable to trade off *optimality* (hence robustness) for *efficiency*, real-time safety-critical applications —such as autonomous driving and space robotics— pose high demands for *efficient global optimality*.

The conflict between the fundamental intractability of robust estimation and the demand for computational efficiency calls for a paradigm shift: since it is impossible to solve all robust estimation problems in polynomial time, we argue that a useful goal is to design algorithms that perform well in typical instances and are able to *certify* optimality of the resulting estimates, but at the same time can declare "failure" on worst-case instances rather than blindly returning an incorrect estimate. Inspired by related works [10], [101], we formalize the notion of a *certifiable algorithm* below.

Definition 1 (Certifiable Algorithm)**.** *Given an optimization problem $\mathbb{P}(\mathbb{D})$ with input data $\mathbb{D}$, an algorithm $\mathbb{A}$ is said to be certifiable if (i) $\mathbb{A}$ runs in polynomial time; (ii) after solving $\mathbb{P}(\mathbb{D})$, $\mathbb{A}$ either returns the global optimizer of $\mathbb{P}$ together with a certificate of optimality, or fails to do so but provides a measure of suboptimality* (e.g., *a bound on the objective value, or the distance to the global optimizer); (iii) $\mathbb{A}$ can certifiably optimally solve $\mathbb{P}(\mathbb{D})$ for common instances of $\mathbb{D}$ (empirically or theoretically).*

Our notion of a certifiable algorithm is stricter than that of [101], as it requires $\mathbb{A}$ to solve $\mathbb{P}(\mathbb{D})$ to global optimality

- *H. Yang and L. Carlone are with the Laboratory for Information and Decision Systems (LIDS), Massachusetts Institute of Technology, Cambridge, MA, 02139. E-mail:* {hankyang, lcarlone}@mit.edu

Code: https://github.com/MIT-SPARK/CertifiablyRobustPerception

for common $\mathbb{D}$. This requirement rules out algorithms that seldomly attain global optimality but provide suboptimality guarantees (*e.g.*, approximation algorithms [84]).

Semidefinite relaxations are a natural choice for designing certifiable algorithms. If the problem $\mathbb{P}$ is a *polynomial optimization problem* (POP, *i.e.*, both its objective and constraints are polynomials), then there exists a standard semidefinite relaxation *hierarchy*, known as Lasserre's hierarchy [59], that relaxes $\mathbb{P}$ into a hierarchy of convex semidefinite programs (SDPs) of increasing size. Each relaxation in this hierarchy can be solved in polynomial time [81] and provides a measure of suboptimality for the resulting estimate. Moreover, under mild technical conditions, the suboptimality of these relaxations becomes zero when their size is large enough, in which case we say the relaxation is *exact*, or *tight*. Therefore, Lasserre's hierarchy provides a hierarchy of certifiable algorithms per Definition 1. We provide an accessible introduction to POPs and their relaxations in Section 2.

Semidefinite relaxations have been successfully used to design certifiable algorithms for many geometric perception problems. The pioneering work by Kahl and Henrion [52] applies Lasserre's hierarchy to solve several early perception problems including camera resectioning, homography estimation, and fundamental matrix estimation. Since then, certifiable algorithms have been designed for modern applications such as pose graph optimization [27], [73], rotation averaging [39], [42], triangulation [4], [33], 3D registration [22], [29], [49], [65], absolute pose estimation [3], relative pose estimation [23], [43], [107], hand-eye calibration [44], [46], [90], and category-level object perception [77], [96]. Although the original formulations of the problems mentioned above are nonconvex, semidefinite relaxations at the *lowest* relaxation order in the hierarchy are shown to be exact in practical applications. Since the SDP resulting from the lowest relaxation order can usually be solved efficiently (*e.g.*, below one second) by off-the-shelf SDP solvers (*e.g.*, SDPT3 [83], MOSEK [8]) or the Burer-Monteiro (B-M) low-rank factorization method [20], [24], [72], both efficiency and (certifiable) optimality can be obtained.

However, these successful examples of certifiable algorithms are underpinned by the restrictive assumption that *the measurements are free of outliers*, which seldomly holds in practice. Heuristics like RANSAC and GNC are typically used to filter out outliers, but it is precisely the use of such heuristics that breaks the optimality guarantee and makes the system prone to undetected failures. Although several works have attempted to design certifiable algorithms for *outlier-robust* geometric perception [26], [58], [88], [95], most approaches (i) are problem-specific, (ii) cannot tolerate high outlier rates (*e.g.*, above 70%) [26], [58], [88], or (iii) become too large to be solved by existing SDP solvers [95].

Contributions. In this paper, we propose a general and scalable framework for designing certifiable outlier-robust estimation algorithms that are empirically *exact* with up to 90% outliers, and present a fast SDP solver that can solve the tight relaxations at an unprecedented scale. We now describe our four contributions in detail.

(I) Robust estimation as polynomial optimization (Section 3). We investigate outlier-robust estimation with common robust cost functions, including truncated least squares (TLS), maximum consensus, Geman-McClure, Tukey's Biweight, L1, Huber, and Barron's adaptive kernel [12]. Our first contribution is to show that robust estimation using these costs can be equivalently reformulated as POPs, even though the robust costs themselves are not polynomials. This result is established by introducing additional variables and manipulating the original costs to polynomials.

(II) A sparse, but exact, semidefinite relaxation (Section 4). With the POP reformulation, it is tempting to apply the standard Lasserre's hierarchy to develop certifiable algorithms for robust estimation. Nevertheless, due to the additional variables (one or two variables per measurement), even for small estimation problems with fewer than 20 measurements, the *lowest-order relaxation* can already lead to SDPs that are too large for existing SDP solvers. Therefore, our second contribution is to focus on the TLS cost and show that it allows us to exploit *term sparsity* of the polynomials in the POP and design a much smaller semidefinite relaxation using *basis reduction*. Compared to the standard Lasserre's hierarchy, our sparse semidefinite relaxation leads to 100 times reduction in the size of the SDP. Unfortunately, even with our sparse relaxation, solving the SDP using off-the-shelf SDP solvers (*e.g.*, MOSEK) is still too slow, and we can only demonstrate empirical exactness of our relaxation on small estimation problems (*e.g.*, 30 measurements).

(III) A scalable and robust SDP solver (Section 5). The limitations of existing SDP solvers lead to our third contribution, a scalable SDP solver that can *certifiably optimally* solve robust estimation problems of moderate but realistic sizes (*e.g.*, 100 measurements). Our solver, called *SpecTrahedral pRoximal gradIent Descent along vErtices* (STRIDE), blends fast *local search* on the nonconvex POP with *global descent* on the convex SDP. Specifically, STRIDE follows a globally convergent trajectory driven by a *proximal gradient method* for solving the SDP, while simultaneously probing long, but *safeguarded*, *rank-one* "strides", generated by fast nonlinear programming algorithms on the POP, to seek rapid descent. Notably, fast heuristics such as RANSAC and GNC can be readily used to bootstrap STRIDE. Particularly, when RANSAC and GNC succeed in finding the globally optimal solution (which happens frequently in the low-outlier regime), STRIDE serves to certify global optimality. Otherwise, when fast heuristics converge to local minima, STRIDE detects suboptimality and escapes such minima.

(IV) Evaluation on six geometric perception problems (Section 6). Our last contribution is to apply our framework and solver to six perception problems: single and multiple rotation averaging, point cloud and mesh registration, absolute pose estimation, and category-level object perception. With extensive experiments on synthetic and real datasets, we demonstrate (i) our sparse SDP relaxation is exact in the presence of up to 60%–90% outliers, (ii) while still being far from real-time, STRIDE is up to 100 times faster than existing SDP solvers on medium-scale problems, and is the only solver than can solve large-scale SDPs with hundreds of thousands of constraints to high accuracy, (iii) STRIDE safeguards existing fast heuristics, *i.e.*, it certifies global optimality if the heuristic estimates are already optimal, or detects and escapes local minima otherwise. We showcase real examples of STRIDE *certifiably* performing scan matching on 3DMatch [105], mesh registration on HomebrewedDB [53], satellite pose estimation on SPEED [76], and vehicle pose and shape estimation on ApolloScape [89].

Novelty with respect to [97], [100]. This paper extends and unifies the contributions presented in our previous conference papers [97], [100]. More in detail, we expand on [97] by (i) showing that other robust costs (beyond TLS) can be rephrased as POPs, (ii) providing a more extensive comparison between (and discussion about) Lasserre's hierarchy and the proposed sparse relaxations, (iii) going beyond certification (in this paper we propose a *solver*, rather than a certification approach), (iv) considering a broader set of applications. We also extend [100], which introduced STRIDE, by (i) generalizing STRIDE to work on multi-block SDPs arising from the proposed relaxations, (ii) tailoring STRIDE to use fast heuristics (*e.g.,* RANSAC or GNC) as a warmstart, and (iii) testing STRIDE on a broader range of problems.

We remark that the main goal of this paper is *not* to produce a method that outperforms problem-specific state-of-the-art algorithms in terms of robustness or efficiency. Our key contribution is instead to show that a broad class of robust estimation problems in geometric perception can be solved to certifiable optimality in polynomial time (despite their hardness), and lay out a scalable framework to build SDP relaxations, that we believe —with further advancement of SDP solvers— will eventually run in real time.

Notation

Scalars, vectors, matrices. We use lowercase characters (*e.g.,* a) to denote real scalars, bold lowercase characters (*e.g.,* $\boldsymbol{a}$) for real (column) vectors, and bold uppercase characters (*e.g.,* $\boldsymbol{A}$) for real matrices. $\mathbf{I}_d$ denotes the identity matrix of size $d \times d$, and $\mathbf{0}$ denotes the all-zero vector or matrix. Given $\boldsymbol{A} \in \mathbb{R}^{m \times n}$, a_{ij} denotes the (i,j)-th entry of $\boldsymbol{A}$, and $[\boldsymbol{A}]_{\mathcal{I},\mathcal{J}}$ denotes the submatrix of $\boldsymbol{A}$ formed by indexing rows $\mathcal{I} \subseteq [m]$ and columns $\mathcal{J} \subseteq [n]$, where $[n] \triangleq \{1,\dots,n\}$ is the set of positive integers up to n. For a vector $\boldsymbol{v} \in \mathbb{R}^n$, we shorthand v_i for its i-th entry and $\boldsymbol{v}_{\mathcal{I}}$ for its entries indexed by $\mathcal{I} \subseteq [n]$. For $\boldsymbol{A}, \boldsymbol{B} \in \mathbb{R}^{m \times n}$, $\langle \boldsymbol{A}, \boldsymbol{B} \rangle \triangleq \sum_{i=1}^m \sum_{j=1}^n a_{ij} b_{ij}$ denotes the usual inner product between real matrices. $\operatorname{tr}(\boldsymbol{A}) \triangleq \sum_{i=1}^n a_{ii}$ denotes the trace of a square matrix $\boldsymbol{A} \in \mathbb{R}^{n \times n}$. We use $\|\cdot\|$ to denote the ℓ_2 norm of a vector and the Frobenius norm of a matrix, *i.e.,* $\|\boldsymbol{a}\| \triangleq \sqrt{\langle \boldsymbol{a}, \boldsymbol{a} \rangle}$ for any $\boldsymbol{a} \in \mathbb{R}^n$ and $\|\boldsymbol{A}\| \triangleq \sqrt{\langle \boldsymbol{A}, \boldsymbol{A} \rangle}$ for any $\boldsymbol{A} \in \mathbb{R}^{m \times n}$. $\|\boldsymbol{a}\|_1 \triangleq \sum_{i=1}^n |a_i|$ denotes the ℓ_1 norm of a vector. $[\boldsymbol{A}, \boldsymbol{B}]$ denotes the *horizontal* concatenation, while $[\boldsymbol{A}\,;\,\boldsymbol{B}]$ denotes the *vertical* concatenation, for proper $\boldsymbol{A}, \boldsymbol{B}$. For $a \in \mathbb{R}$, the symbol $\lceil a \rceil$ returns the smallest integer above a.

Sets. We use $\mathbb{S}^n$ to denote the space of $n \times n$ real symmetric matrices, and $\mathbb{S}^n_+$ (resp. $\mathbb{S}^n_{++}$) to denote the set of matrices in $\mathbb{S}^n$ that are *positive semidefinite* (resp. definite). We also write $\boldsymbol{X} \succeq 0$ (resp. $\boldsymbol{X} \succ 0$) to indicate $\boldsymbol{X}$ is positive semidefinite (resp. definite). $\mathcal{S}^{d-1} \triangleq \{\boldsymbol{v} \in \mathbb{R}^d \mid \|\boldsymbol{v}\| = 1\}$ denotes the d-dimensional unit sphere. We denote by $\mathrm{SO}(d) \triangleq \{\boldsymbol{R} \in \mathbb{R}^d \mid \boldsymbol{R}^{\mathsf{T}}\boldsymbol{R} = \mathbf{I}_d, \det(\boldsymbol{R}) = +1\}$ the d-dimensional *special orthogonal group* (rotation matrices). $|\mathcal{A}|$ denotes the cardinality of a finite set $\mathcal{A}$. $\mathbb{Z}_+$ (resp. $\mathbb{Z}_{++}$) denotes the set of nonnegative (resp. positive) integers, and $\mathbb{Q}$ denotes the set of rational numbers.

2 Preliminaries

This section reviews key facts about multi-block semidefinite programming [83] (Section 2.1), and provides an introduction to polynomial optimization and Lasserre's semidefinite relaxation hierarchy [59] (Section 2.2). While somewhat mathematically dense, these preliminaries are designed as a pragmatic introduction for the non-expert reader.

2.1 Semidefinite Programming

A *multi-block* semidefinite programming (SDP) problem is an optimization problem in the following *primal* form [83]:

$$\min_{\boldsymbol{X} \in \mathbb{X}} \{\langle \boldsymbol{C}, \boldsymbol{X} \rangle \mid \mathcal{A}(\boldsymbol{X}) = \boldsymbol{b},\ \boldsymbol{X} \in \mathcal{K}\}\,. \tag{P}$$

where the variable $\boldsymbol{X} = (\boldsymbol{X}_1, \dots, \boldsymbol{X}_l)$ is a collection of l square matrices (the "blocks") with $\boldsymbol{X}_i \in \mathbb{R}^{n_i \times n_i}$ for $i = 1, \dots, l$ (conveniently ordered such that $n_1 \geq \dots \geq n_l$); the domain $\mathbb{X} \triangleq \mathbb{S}^{n_1} \times \dots \times \mathbb{S}^{n_l}$ restricts the matrices to be symmetric. The objective is a linear combination of the matrices in $\boldsymbol{X}$, *i.e.,* $\langle \boldsymbol{C}, \boldsymbol{X} \rangle \triangleq \sum_{i=1}^l \langle \boldsymbol{C}_i, \boldsymbol{X}_i \rangle$ (for given matrices $\boldsymbol{C}_i \in \mathbb{S}^{n_i}, i = 1, \dots, l$). The problem includes independent linear constraints $\mathcal{A}(\boldsymbol{X}) = \boldsymbol{b}$ on $\boldsymbol{X}$, where:

$$\mathcal{A}(\boldsymbol{X}) \triangleq \left[\sum_{i=1}^l \langle \boldsymbol{A}_{i1}, \boldsymbol{X}_i \rangle \ ; \ \dots \ ; \ \sum_{i=1}^l \langle \boldsymbol{A}_{im}, \boldsymbol{X}_i \rangle \right] \in \mathbb{R}^m \tag{1}$$

for given matrices $\boldsymbol{A}_{ij} \in \mathbb{S}^{n_i}, i = 1, \dots, l$ and $j = 1, \dots, m$, and where $\boldsymbol{b} \in \mathbb{R}^m$ is a given vector. Finally, the constraint $\boldsymbol{X} \in \mathcal{K}$ enforces that each matrix in $\boldsymbol{X}$ is positive semidefinite (*i.e.,* $\mathcal{K} \triangleq \mathbb{S}^{n_1}_+ \times \dots \times \mathbb{S}^{n_l}_+$ is a product of l positive semidefinite cones). The feasible set of (P), denoted by $\mathcal{F}_{\mathrm{P}} \triangleq \{\boldsymbol{X} \in \mathbb{X} \mid \mathcal{A}(\boldsymbol{X}) = \boldsymbol{b}, \boldsymbol{X} \in \mathcal{K}\}$, is called a *spectrahedron* [18].

The Lagrangian *dual* of (P) is another multi-block SDP:

$$\max_{\boldsymbol{y} \in \mathbb{R}^m, \boldsymbol{S} \in \mathbb{X}} \{\langle \boldsymbol{b}, \boldsymbol{y} \rangle \mid \mathcal{A}^*(\boldsymbol{y}) + \boldsymbol{S} = \boldsymbol{C},\ \boldsymbol{S} \in \mathcal{K}\} \tag{D}$$

where $\mathcal{A}^* : \mathbb{R}^m \to \mathbb{X}$ is the adjoint of $\mathcal{A}$ and is defined as:

$$\mathcal{A}^*(\boldsymbol{y}) \triangleq \left(\sum_{j=1}^m y_j \boldsymbol{A}_{1j}, \dots, \sum_{j=1}^m y_j \boldsymbol{A}_{lj} \right) \in \mathbb{X} \tag{2}$$

and the equality $\mathcal{A}^*(\boldsymbol{y}) + \boldsymbol{S} = \boldsymbol{C}$ is enforced block-wise.

Under mild assumptions (*e.g.,* Slater's condition [21]), *strong duality* holds between (P) and (D) (*i.e.,* the minimum of (P) equals the maximum of (D)). In this case, $(\boldsymbol{X}^\star, \boldsymbol{y}^\star, \boldsymbol{S}^\star) \in \mathbb{X} \times \mathbb{R}^m \times \mathbb{X}$ is simultaneously *optimal* for (P)-(D) if and only if the following KKT conditions hold

$$\begin{array}{ll} \text{primal feasibility :} & \mathcal{A}(\boldsymbol{X}^\star) = \boldsymbol{b}, \boldsymbol{X}^\star \in \mathcal{K}, \\ \text{dual feasibility :} & \mathcal{A}^*(\boldsymbol{y}^\star) + \boldsymbol{S}^\star = \boldsymbol{C}, \boldsymbol{S}^\star \in \mathcal{K}, \\ \text{complementarity :} & \langle \boldsymbol{X}^\star, \boldsymbol{S}^\star \rangle = 0. \end{array} \tag{3}$$

The KKT conditions (3) imply strong duality because

$$\begin{aligned} 0 &= \langle \boldsymbol{X}^\star, \boldsymbol{S}^\star \rangle = \langle \boldsymbol{X}^\star, \boldsymbol{C} - \mathcal{A}^*(\boldsymbol{y}^\star) \rangle \\ &= \langle \boldsymbol{C}, \boldsymbol{X}^\star \rangle - \langle \mathcal{A}(\boldsymbol{X}^\star), \boldsymbol{y}^\star \rangle = \langle \boldsymbol{C}, \boldsymbol{X}^\star \rangle - \langle \boldsymbol{b}, \boldsymbol{y}^\star \rangle\,. \end{aligned} \tag{4}$$

Given $(\boldsymbol{X}, \boldsymbol{y}, \boldsymbol{S}) \in \mathcal{K} \times \mathbb{R}^m \times \mathcal{K}$, we measure its feasibility and optimality using the standard relative KKT residuals

$$\begin{aligned} \eta_p &\triangleq \|\mathcal{A}(\boldsymbol{X}) - \boldsymbol{b}\| / (1 + \|\boldsymbol{b}\|), \\ \eta_d &\triangleq \|\mathcal{A}^*(\boldsymbol{y}) + \boldsymbol{S} - \boldsymbol{C}\| / (1 + \|\boldsymbol{C}\|), \\ \eta_g &\triangleq |\langle \boldsymbol{C}, \boldsymbol{X} \rangle - \langle \boldsymbol{b}, \boldsymbol{y} \rangle| / (1 + |\langle \boldsymbol{C}, \boldsymbol{X} \rangle| + |\langle \boldsymbol{b}, \boldsymbol{y} \rangle|), \end{aligned} \tag{5}$$

where $\|\boldsymbol{X}\| = \sum_{i=1}^{l} \|\boldsymbol{X}_i\|$ for any $\boldsymbol{X} \in \mathbb{X}$. We define $\eta_{\max} \triangleq \max\{\eta_p, \eta_d, \eta_g\}$ as the *maximum KKT residual*.

SDP solvers. The most robust approach for solving SDP (P) (and (D)) is based on *primal-dual interior point methods* (IPM) [6], [81], *e.g.*, SDPT3 [83] and MOSEK [8]. For problems of small to medium size (*e.g.*, $n_1 \leq 5000, m \leq 50,000$), IPMs can solve the SDP to arbitrary accuracy, *i.e.*, $\eta_{\max} < \varepsilon$ for ε arbitrarily small, with a typical per-iteration complexity $\mathcal{O}(n_1^3 + m^2 n_1^2 + m^3)$.[1] If each linear constraint only involves a small number of blocks (*i.e.*, for each $j = 1, \ldots, m$, $\boldsymbol{A}_{ij} = \boldsymbol{0}$ for many blocks $i = 1, \ldots, l$), then IPMs can be made much more efficient using *dualization* [106]. Nevertheless, such sparsity is not always present and generally IPMs cannot solve large-scale problems on an ordinary workstation.

First-order methods based on ADMM and Augmented Lagrangian, *e.g.*, CDCS [109], and SDPNAL+ [104], can handle large-scale problems but exhibit slow convergence, and hence can only obtain solutions of moderate accuracy.

For single-block problems ($l = 1$) with low-rank solutions (*i.e.*, $\operatorname{rank}(\boldsymbol{X}^\star) \ll n_1$) and $m = \mathcal{O}(n_1)$, the Burer-Monteiro (B-M) low-rank factorization method [20], [24] is preferable. Section 1 mentioned the success of SDP relaxations in solving *outlier-free* perception problems. This success is attributed to the following facts: (a) most of the SDPs arising in outlier-free estimation have $n_1 < 100$ and $m < 1000$, and can be solved by IPMs in less than one second; (b) although some SDPs (*e.g.*, [73]) can have $n_1 > 10,000$, they can be efficiently solved by B-M because the optimal solution is low-rank and $m \approx n_1$ [72].

Challenges. Unfortunately, *none* of the existing solvers can solve the SDPs presented in this paper to a desired accuracy. In particular, our SDPs have $n_1 < 5000$ but $m = \mathcal{O}(n_1^2)$ as large as a few millions, rendering IPMs and B-M factorization inapplicable. Moreover, our SDPs admit rank-one optimal solutions and are necessarily degenerate [5] (loosely speaking, degeneracy is a property that often leads to slower convergence in SDP solvers and prevents the application of B-M). Our previous work [100] shows that first-order methods perform poorly on degenerate problems.

2.2 Polynomial Optimization and Lasserre's Hierarchy

Polynomial optimization. Given $\boldsymbol{x} = [x_1\,;\, x_2\,;\, \ldots\,;\, x_d] \in \mathbb{R}^d$, a *monomial* in $\boldsymbol{x}$ is a product of x_i's with *nonnegative* integer exponents, *i.e.*, $\boldsymbol{x}^{\boldsymbol{\alpha}} \triangleq x_1^{\alpha_1} \cdots x_d^{\alpha_d}$ for $\boldsymbol{\alpha} \in \mathbb{Z}_+^d$ (for instance $x_1^2 x_5 x_6^3$ is a monomial). The sum of the exponents, $\|\boldsymbol{\alpha}\|_1$, is called the *degree* of the monomial (*e.g.*, the monomial $x_1^2 x_5 x_6^3$ has degree 6).

A real *polynomial* $p(\boldsymbol{x})$ is a finite sum of monomials with real coefficients. The degree of a polynomial $p(\boldsymbol{x})$, denoted by $\deg(p)$, is the *maximum* degree of its monomials. The ring of polynomials with real coefficients is denoted by $\mathbb{R}[\boldsymbol{x}]$. A standard polynomial optimization problem (POP) reads

$$p^\star \triangleq \min_{\boldsymbol{x} \in \mathbb{R}^d} \left\{ p(\boldsymbol{x}) \;\middle|\; \begin{array}{l} h_i(\boldsymbol{x}) = 0, i = 1, \ldots, l_h \\ g_j(\boldsymbol{x}) \geq 0, j = 1, \ldots, l_g \end{array} \right\}, \qquad \text{(POP)}$$

where $p, h_i, g_j \in \mathbb{R}[\boldsymbol{x}]$. Problem (POP) is easily seen to be NP-hard [61], *e.g.*, it can model combinatorial binary constraints $x_i \in \{+1, -1\}$ via $x_i^2 - 1 = 0, i = 1, \ldots, d$.

Lasserre's hierarchy. We now give a simplified (and somewhat less conventional) introduction to Lasserre's hierarchy that is sufficient for understanding our paper. For a comprehensive treatment, we refer the reader to [61].

We define $[\boldsymbol{x}]_\kappa \triangleq \{\boldsymbol{x}^{\boldsymbol{\alpha}} \mid \|\boldsymbol{\alpha}\|_1 \leq \kappa, \boldsymbol{\alpha} \in \mathbb{Z}_+^d\}$ to be the set of monomials of degree up to κ. For example, if $\boldsymbol{x} = [x_1\,;\, x_2]$ and $\kappa = 2$, then $[\boldsymbol{x}]_2 = [1\,;\, x_1\,;\, x_2\,;\, x_1^2\,;\, x_1 x_2\,;\, x_2^2]$. The dimension of $[\boldsymbol{x}]_\kappa$ is $\underline{d}_\kappa \triangleq \binom{d+\kappa}{\kappa}$. With $[\boldsymbol{x}]_\kappa$, we form the so-called *moment matrix* $\boldsymbol{X}_\kappa \triangleq [\boldsymbol{x}]_\kappa [\boldsymbol{x}]_\kappa^\mathsf{T}$. For instance, for $\boldsymbol{x} = [x_1\,;\, x_2]$ and $\kappa = 2$ (*cf.* with $[\boldsymbol{x}]_2$ above):

$$\boldsymbol{X}_\kappa \triangleq [\boldsymbol{x}]_2 [\boldsymbol{x}]_2^\mathsf{T} = \left[\begin{array}{cccccc} 1 & x_1 & x_2 & x_1^2 & x_1 x_2 & x_2^2 \\ x_1 & x_1^2 & x_1 x_2 & x_1^3 & x_1^2 x_2 & x_1 x_2^2 \\ x_2 & x_1 x_2 & x_2^2 & x_1^2 x_2 & x_1 x_2^2 & x_2^3 \\ x_1^2 & x_1^3 & x_1^2 x_2 & x_1^4 & x_1^3 x_2 & x_1^2 x_2^2 \\ x_1 x_2 & x_1^2 x_2 & x_1 x_2^2 & x_1^3 x_2 & x_1^2 x_2^2 & x_1 x_2^3 \\ x_2^2 & x_1 x_2^2 & x_2^3 & x_1^2 x_2^2 & x_1 x_2^3 & x_2^4 \end{array} \right]. \qquad (6)$$

By construction, $\boldsymbol{X}_\kappa \in \mathbb{S}_+^{\underline{d}_\kappa}$ is positive semidefinite and has $\operatorname{rank}(\boldsymbol{X}_\kappa) = 1$. Moreover, the set of *unique* entries in $\boldsymbol{X}_\kappa$ is simply $[\boldsymbol{x}]_{2\kappa}$, *i.e.*, the set of monomials of degree up to 2κ (these monomials typically appear multiple times in $\boldsymbol{X}_\kappa$, *e.g.*, see $x_1 x_2$ in eq. (6)). Therefore, a key fact is that *—for a suitable matrix $\boldsymbol{A}$— the linear function $\langle \boldsymbol{A}, \boldsymbol{X}_\kappa \rangle$ can express any polynomial in $\boldsymbol{x}$ of degree up to 2κ.*

The key idea of Lasserre's hierarchy is to (i) rewrite (POP) using the moment matrix $\boldsymbol{X}_\kappa$, (ii) relax the (non-convex) rank-1 constraint on $\boldsymbol{X}_\kappa$, and (iii) add redundant constraints that are trivially satisfied in (POP); as we show below, this leads to a *convex* semidefinite program.

(i) Rewriting (POP) *using* $\boldsymbol{X}_\kappa$. We pick a positive integer $\kappa \in \mathbb{Z}_{++}$ (the *order* of the relaxation) such that $2\kappa \geq \max\{\deg(p), \deg(h_1), \ldots, \deg(h_{l_h}), \deg(g_1), \ldots, \deg(g_{l_g})\}$. (this way we can express both objective function and constraints using $\boldsymbol{X}_\kappa$). For instance, we can rewrite the objective and the equality constraints as:

$$\text{objective}: \quad \langle \boldsymbol{C}_1, \boldsymbol{X}_\kappa \rangle \qquad (7)$$

$$\text{equality constraints}: \; \langle \boldsymbol{A}_{\mathrm{eq},j}, \boldsymbol{X}_\kappa \rangle = 0, \; j = 1, \ldots, l_h \qquad (8)$$

for suitable matrices $\boldsymbol{C}_1$ and $\boldsymbol{A}_{\mathrm{eq},j}$. Note that using $\boldsymbol{X}_\kappa$ is already a relaxation since we are no longer enforcing the entries of $\boldsymbol{X}_\kappa$ to be monomials (*e.g.*, we do not enforce the entry $x_1 x_2$ in (6) to be the product of the entries x_1 and x_2, which would be a non-convex constraint).

*(ii) Relaxing the (non-convex) rank-*1 *constraint on* $\boldsymbol{X}_\kappa$. At the previous point we noticed we can rewrite objective and constraints in (POP) as linear (hence convex) functions of $\boldsymbol{X}_\kappa$. However, $\boldsymbol{X}_\kappa$ still belongs to the set of positive-semidefinite rank-1 matrices, which is a non-convex set due to the rank constraint. Therefore, we simply relax the rank constraint and only enforce:

$$\text{moment matrix}: \quad \boldsymbol{X}_\kappa \succeq 0. \qquad (9)$$

(iii) Adding redundant constraints. Since we have relaxed (POP) by re-parametrizing it in $\boldsymbol{X}_\kappa$ and dropping the rank constraint, the final step to obtain Lasserre's relaxation consists in adding extra constraints to make the relaxation tighter. First of all, we observe that there are multiple repeated entries in the moment matrix (*e.g.*, in (6), the entry $x_1 x_2$ appears 4 times in the matrix). Therefore, we can enforce these entries to be the same. In general, this

[1] $\mathcal{O}(n_1^3)$ for spectral decomposition of dense primal and dual iterates $(\boldsymbol{X}, \boldsymbol{S})$, $\mathcal{O}(m^2 n_1^2)$ for forming the Schur complement system, and $\mathcal{O}(m^3)$ for factorizing and solving the Schur complement system.

leads to $m_{\text{mom}} = \mathfrak{t}(\underline{d}_\kappa) - \underline{d}_{2\kappa} + 1$ linear constraints, where $\mathfrak{t}(n) \triangleq \frac{n(n+1)}{2}$ is the dimension of $\mathbb{S}^n$. These constraints are typically called *moment constraints*:

$$\text{moment constraints}: \quad \begin{array}{l} \langle \boldsymbol{A}_{\text{mom},j}, \boldsymbol{X}_\kappa \rangle = 0, \\ j = 1, \dots, \mathfrak{t}(\underline{d}_\kappa) - \underline{d}_{2\kappa} + 1. \end{array} \tag{10}$$

These constraints also include the constraint $[\boldsymbol{X}_k]_{11} = 1$, which follows from the definition of the moment matrix (see eq. (6)) and the fact that 1 is the monomial of order zero.

Second, we can also add *redundant* equality constraints. Simply put, if $h_i = 0$, then also $h_i \cdot x_1 = 0$, $h_i \cdot x_2 = 0$, and so on, for any monomial we multiply by h_i. Since via $\boldsymbol{X}_\kappa$ we can represent any polynomial of degree up to 2κ, we can write as linear constraints any polynomial equality in the form $h_i \cdot [\boldsymbol{x}]_{2\kappa - \deg(h_i)} = \boldsymbol{0}$ (the order of the monomials is chosen such that the product does not exceed order 2κ). These new equalities can again be written linearly as:

$$\begin{array}{l} \text{(redundant) equality constraints}: \langle \boldsymbol{A}_{\text{req},ij}, \boldsymbol{X}_\kappa \rangle = 0, \\ \qquad i = 1, \dots, l_h, \ \ j = 1, \dots, \underline{d}_{2\kappa - \deg(h_i)} \end{array} \tag{11}$$

for suitable $\boldsymbol{A}_{\text{req},ij}$. Since the first entry of $[\boldsymbol{x}]_{2\kappa - \deg(h_i)}$ is always 1 (*i.e.*, the monomial of order zero), eq. (11) already includes the original equality constraints in (8).

Finally, we observe that if $g_j \geq 0$, then for any positive semidefinite matrix $\boldsymbol{M}$, it holds $g_j \cdot \boldsymbol{M} \succeq 0$. Since we can represent any polynomial of order up to 2κ as a linear function of $\boldsymbol{X}_\kappa$, we can add redundant constraints in the form $g_j \cdot \boldsymbol{X}_{\kappa - \lceil \deg(g_j)/2 \rceil} \succeq 0$ (by construction $g_j \cdot \boldsymbol{X}_{\kappa - \lceil \deg(g_j)/2 \rceil}$ only contains polynomials of degree up to 2κ). To phrase the resulting relaxation in the standard form (P), it is common to add extra matrix variables $\boldsymbol{X}_{g_j} = g_j \cdot \boldsymbol{X}_{\kappa - \lceil \deg(g_j)/2 \rceil}$ for $j = 1, \dots, l_g$ (the so called *localizing matrices* [61, §3.2.1]) and then force these matrices to be a linear function of $\boldsymbol{X}_\kappa$:

$$\text{localizing matrices}: \quad \boldsymbol{X}_{g_j} \succeq 0, \quad j = 1, \dots, l_g \tag{12}$$

$$\begin{array}{l} \text{localizing constraints}: \langle \boldsymbol{A}_{\text{loc},jkh}, \boldsymbol{X}_\kappa \rangle = [\boldsymbol{X}_{g_j}]_{hk} \\ \qquad j = 1, \dots, l_g, \ \ 1 \leq h \leq k \leq \underline{d}_{\kappa - \lceil \deg(g_j)/2 \rceil} \end{array} \tag{13}$$

where the linear constraints (for some $\boldsymbol{A}_{\text{loc},jkh}$) enforce each entry of $\boldsymbol{X}_{g_j}$ to be a linear combination of entries in $\boldsymbol{X}_\kappa$.

Following steps (i)-(iii) above, it is straightforward to obtain the following (convex) semidefinite program:

$$f^\star_\kappa = \min_{\boldsymbol{X} = (\boldsymbol{X}_\kappa, \boldsymbol{X}_1, \dots, \boldsymbol{X}_{l_g})} \{ \langle \boldsymbol{C}_1, \boldsymbol{X}_\kappa \rangle \mid \mathcal{A}(\boldsymbol{X}) = \boldsymbol{b}, \boldsymbol{X} \succeq 0 \}, \tag{LAS}$$

where the variable $\boldsymbol{X} = (\boldsymbol{X}_\kappa, \boldsymbol{X}_1, \dots, \boldsymbol{X}_{l_g})$ is a collection of positive-semidefinite matrices (*cf.* (9) and (12), and we shorthand $\boldsymbol{X}_j = \boldsymbol{X}_{g_j}$ for notation convenience), the objective is the one given in (7), and the linear constraints $\mathcal{A}(\boldsymbol{X}) = \boldsymbol{b}$ collect all the constraints in (10), (11), and (13). Problem (LAS) can be readily formulated as a multi-block SDP in the primal form (P), which matches the data format used by common SDP solvers. Problem (LAS) is commonly known as the *dense* Lasserre's relaxation because a fully dense monomial basis $[\boldsymbol{x}]_\kappa$ is used to build the moment matrix. One can solve the relaxation for different choices of κ, leading to a *hierarchy* of convex relaxations.

While we presented Lasserre's hierarchy in a somewhat procedural way, the importance of the hierarchy lies in its stunning theoretical properties, that we review below.

Theorem 2 (Lasserre's Hierarchy [59], [61], [68])**.** *Let $-\infty < p^\star < \infty$ be the optimum of* (POP) *and $f^\star_\kappa$ be the optimum of* (LAS)*, assume* (POP) *satisfies the Archimedeanness condition (a stronger form of compactness,* cf. *[18, Definition 3.137]), then*

(i) $f^\star_\kappa$ converges to $p^\star$ from below as $\kappa \to \infty$, and convergence occurs at a finite κ under suitable technical conditions [68];

(ii) if $\operatorname{rank}(\boldsymbol{X}^\star_\kappa) = 1$ *at some finite κ, then $f^\star_\kappa = p^\star$;*

(iii) if $f^\star_\kappa = p^\star$ at some finite κ, then for every global minimizer $\boldsymbol{x}^\star$ of (POP)*, $\boldsymbol{X}^\star_\kappa \triangleq [\boldsymbol{x}^\star]_\kappa [\boldsymbol{x}^\star]_\kappa^\top$ is optimal for* (LAS)*, and every rank-one optimal solution $\boldsymbol{X}^\star_\kappa$ of* (LAS) *can be written as $[\boldsymbol{x}^\star]_\kappa [\boldsymbol{x}^\star]_\kappa^\top$ for some $\boldsymbol{x}^\star$ that is optimal for* (POP)*;*

Theorem 2 states that (LAS) provides a hierarchy of lower bounds for (POP). When the relaxation is exact ($p^\star = f^\star_\kappa$), global minimizers of (POP) correspond to rank-one solutions of (LAS).

3 Outlier-Robust Estimation as POP

In this section, we consider a general formulation of estimation with robust cost functions. We show that, for seven popular robust costs, this formulation can be recast as a (POP). We conclude the section by showcasing the resulting formulation on six perception problems.

Outlier-robust estimation. Given a set of N measurements $\mathcal{Z} = \{\boldsymbol{z}_i\}_{i=1}^N$ (*e.g.*, 2D image keypoints, 3D point clouds, relative poses), we consider the problem of using $\mathcal{Z}$ to estimate an unknown geometric model $\boldsymbol{x} \in \mathcal{X} \subseteq \mathbb{R}^d$ (*e.g.*, camera poses, rigid transformations, 3D shapes, robot trajectory), despite the fact that the measurement set $\mathcal{Z}$ may contain a large amount of *outliers*. Building on standard M-estimation [66], [80], we perform outlier-robust estimation by solving the following optimization problem

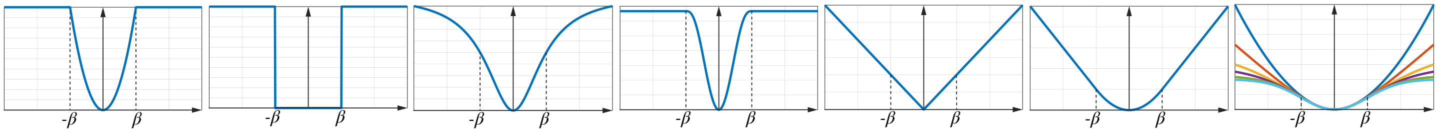
$$\min_{\boldsymbol{x} \in \mathcal{X} \subseteq \mathbb{R}^d} \sum_{i=1}^N \rho(r(\boldsymbol{x}, \boldsymbol{z}_i), \beta_i) + \psi(\boldsymbol{x}), \tag{Robust}$$

where $r(\boldsymbol{x}, \boldsymbol{z}_i)$ is a (scalar) *residual* function that measures the mismatch between $\boldsymbol{x}$ and $\boldsymbol{z}_i$ (*e.g.*, Euclidean distances, pose errors), $\beta_i > 0$ (set by the user) is the *maximum admissible residual* for a measurement to be considered as an *inlier* (or minimum residual to be an outlier), $\rho(r, \beta_i)$ is a *robust* cost function that penalizes outliers much *less* than inliers to prevent outliers from contaminating the estimate. We include a *regularization* term $\psi(\boldsymbol{x})$ in (Robust), to keep full generality: as we will see in the examples below, a regularizer is often added to high-dimensional estimation problems to ensure the solution is unique and well-behaved. We make the following assumption on problem (Robust).

Assumption 3 (Polynomial Residual, Constraint, and Regularization)**.** *In* (Robust)*, assume (i) r^2, ψ are polynomials; (ii) the constraint $\boldsymbol{x} \in \mathcal{X}$ can be described by finitely many polynomial equalities and inequalities,* i.e., $\mathcal{X} = \{\boldsymbol{x} \in \mathbb{R}^d \mid h_i(\boldsymbol{x}) = 0, i \in 1, \dots, l_h, g_j(\boldsymbol{x}) \geq 0, j = 1, \dots, l_g\}$.

Assumption 3 is the prerequisite for applying the machinery of semidefinite relaxation for POP in Section 2.2. These assumptions are often mild in geometric perception problems, a point that will become clearer when we introduce the six examples later in this section (*cf.* Proposition 7).

Now the only component of (Robust) that may prevent it from being a POP is the robust cost $\rho(r, \beta_i)$. In *outlier-free*



(a) TLS (b) MC (c) GM (d) TB (e) L1 (f) Huber (g) Adaptive

Fig. 1. Estimation using common robust cost functions can be equivalently reformulated as polynomial optimization problems (*cf.* Proposition 4).

estimation, $\rho = r^2/\beta_i^2$ is chosen as the *least squares* cost and (Robust) is immediately in the form of (POP). However, in outlier-robust estimation, ρ is typically not a polynomial. For instance, let us consider the *truncated least squares* (TLS) cost, which will be extensively used in this paper:

$$\rho_{\mathrm{TLS}}(r,\beta_i) \triangleq \min\left\{\frac{r^2}{\beta_i^2},1\right\} = \begin{cases}\frac{r^2}{\beta_i^2} & |r| \leq \beta_i \\ 1 & \text{otherwise}\end{cases}, \tag{14}$$

The TLS cost (14) is apparently not a polynomial, and it is not even a *smooth* function (*cf.* Fig. 1(a)).

Reformulation as POP. To build intuition, we now show that (Robust) with the TLS cost (14) can be reformulated as a POP; then we generalize this conclusion to other cost functions in Proposition 4. The key observation is that, for any $a, b \in \mathbb{R}$, $\min\{a,b\} \equiv \min_{\theta\in\{+1,-1\}} \frac{1+\theta}{2}a + \frac{1-\theta}{2}b$ which allows recasting (Robust) with $\rho = \rho_{\mathrm{TLS}}$ as

$$\min_{\substack{\boldsymbol{x}\in\mathcal{X}\subseteq\mathbb{R}^d,\\ \boldsymbol{\theta}\in\{\pm1\}^N}} \sum_{i=1}^{N} \frac{1+\theta_i}{2}\frac{r^2(\boldsymbol{x},\boldsymbol{z}_i)}{\beta_i^2} + \frac{1-\theta_i}{2} + \psi(\boldsymbol{x}), \tag{TLS}$$

where each binary variable $\theta_i \in \{+1,-1\}$ decides whether the i-th measurement $\boldsymbol{z}_i$ is an inlier ($\theta_i = +1$) or an outlier ($\theta_i = -1$). By recalling that $\theta_i \in \{+1,-1\} \Leftrightarrow \theta_i^2 - 1 = 0$, we see that problem (TLS) is an instance of (POP), with the decision variables now being $(\boldsymbol{x},\boldsymbol{\theta}) \in \mathbb{R}^{d+N}$. The next proposition states that, the reformulation above can be generalized to a broader set of robust cost functions.

Proposition 4 (Robust Estimation as POP)**.** *Under Assumption 3, if the cost function ρ in (Robust) is one of the following:*

(i) truncated least squares (TLS): $\rho_{\mathrm{TLS}} \triangleq \min\left\{\frac{r^2}{\beta_i^2},1\right\}$;

(ii) maximum consensus: $\rho_{\mathrm{MC}} \triangleq \begin{cases}0 & |r| \leq \beta_i \\ 1 & \textit{otherwise}\end{cases}$;

(iii) Geman-McClure: $\rho_{\mathrm{GM}} \triangleq \frac{r^2/\beta_i^2}{1+r^2/\beta_i^2}$;

(iv) Tukey's Biweight: $\rho_{\mathrm{TB}} \triangleq \begin{cases}\frac{r^2}{\beta_i^2} - \frac{r^4}{\beta_i^4} + \frac{r^6}{3\beta_i^6} & |r| \leq \beta_i \\ \frac{1}{3} & \textit{otherwise}\end{cases}$,

then (Robust) can be recast as a (POP) with $d+N$ variables, where each of the additional N variables indicates the confidence of the corresponding measurement being an inlier. Moreover, (Robust) with the following costs can also be written as a (POP)

(v) L1: $\rho_{\mathrm{L1}} \triangleq |r|/\beta_i$;

(vi) Huber: $\rho_{\mathrm{HB}} \triangleq \begin{cases}\frac{r^2}{2\beta_i^2} & |r| \leq \beta_i \\ \frac{|r|}{\beta_i} - \frac{1}{2} & \textit{otherwise}\end{cases}$;

(vii) Adaptive [12]: $\rho_{\mathrm{ADT},s} \triangleq \frac{|s-2|}{s}\left(\left(\frac{r^2/\beta_i^2}{|s-2|}+1\right)^{\frac{s}{2}} - 1\right)$,

for a given scale parameter $s \in \mathbb{Q}\backslash\{0,2\}$,

by adding slack variable(s) for each measurement.

Fig. 1 plots the seven robust costs (Fig. 1(g) shows $\rho_{\mathrm{ADT},s}$ for six different values of s). While we postpone the proof to Supplementary Material, the key insight is that for common robust cost functions we can either (a) use Black-Rangarajan duality [16] to convert them into polynomials by introducing additional slack variables – one for each measurement (we use this approach for (i)-(iv)), or (b) directly manipulate them into polynomials by change of variables (for (v)-(vii)).

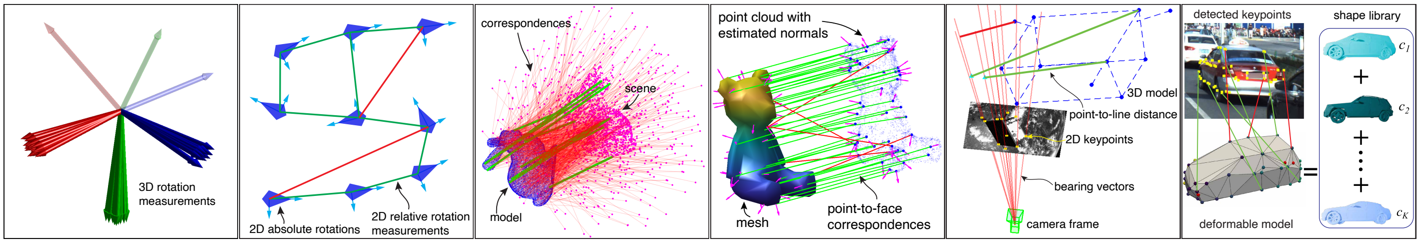
Perception examples. We now shed some light on the generality of the formulation (Robust) and Assumption 3 by considering six outlier-robust geometric perception problems. We first present the examples and then conclude they all satisfy Assumption 3 in Proposition 7. We assume $\psi(\boldsymbol{x}) = 0$ unless otherwise mentioned.

Example 1 (Single Rotation Averaging [45])**.** *Given N measurements of an unknown q-dimensional rotation $\{\boldsymbol{z}_i = \widetilde{\boldsymbol{R}}_i \in \mathrm{SO}(q)\}_{i=1}^N$, single rotation averaging seeks to find the best average rotation $\boldsymbol{x} = \boldsymbol{R} \in \mathrm{SO}(q)$. The residual function is chosen as the chordal distance between $\boldsymbol{R}$ and $\widetilde{\boldsymbol{R}}_i$: $r(\boldsymbol{x},\boldsymbol{z}_i) = \|\boldsymbol{R} - \widetilde{\boldsymbol{R}}_i\|$. Fig. 2(a) plots an instance of 3D single rotation averaging with 20 measurements (rotations are plotted as 3D coordinate frames), among which there is a single outlier (shown as transparent).*

Example 2 (Multiple Rotation Averaging [27], [39], [58])**.** *Let $\mathcal{G} = (\mathcal{V},\mathcal{E})$ be an undirected graph with vertex set $\mathcal{V} = [n]$ and edge set $\mathcal{E}$. Each vertex $i \in \mathcal{V}$ is associated with an unknown rotation $\boldsymbol{R}_i \in \mathrm{SO}(q)$ (typically $q = 2$ or $q = 3$), while each edge $(i,j) \in \mathcal{E}$ gives a relative rotation measurement $\widetilde{\boldsymbol{R}}_{ij} \in \mathrm{SO}(q)$ between the unknown rotations at vertex i and j. Multiple rotation averaging estimates the set of absolute rotations on the vertices $\boldsymbol{x} = \{\boldsymbol{R}_i\}_{i\in\mathcal{V}} \in \mathrm{SO}(q)^n$ from relative measurements over $\mathcal{E}$. The residual function is chosen as the chordal distance between $\boldsymbol{R}_i\widetilde{\boldsymbol{R}}_{ij}$ and $\boldsymbol{R}_j$ for $(i,j) \in \mathcal{E}$: $r(\boldsymbol{x},\boldsymbol{z}_{ij}) = \|\boldsymbol{R}_i\widetilde{\boldsymbol{R}}_{ij} - \boldsymbol{R}_j\|$. Fig. 2(b) plots an instance of 2D multiple rotation averaging with 9 (unknown) absolute rotations and 11 (measured) relative measurements, two of which are outliers (shown in red).*

Example 3 (Point Cloud Registration [101])**.** *Given two sets of 3D points with putative correspondences $\{\boldsymbol{z}_i = (\boldsymbol{p}_i,\boldsymbol{q}_i)\}_{i=1}^N$ (e.g., matched by deep-learned features [98]), point cloud registration seeks the best rigid transformation $\boldsymbol{x} = (\boldsymbol{R},\boldsymbol{t}) \in \mathrm{SO}(3)\times\mathbb{R}^3$ to align them. The residual function is chosen as the Euclidean distance between pairs of points after applying the rigid transformation: $r(\boldsymbol{x},\boldsymbol{z}_i) = \|\boldsymbol{q}_i - \boldsymbol{R}\boldsymbol{p}_i - \boldsymbol{t}\|$. For mathematical convenience* (i.e., *to satisfy the Archimedeanness condition in Theorem 2), we assume the translation to be bounded: $\boldsymbol{t} \in \mathcal{B}_T^3$, where $\mathcal{B}_T^q \triangleq \{\boldsymbol{t} \in \mathbb{R}^q \mid \|\boldsymbol{t}\| \leq T\}$ defines a q-dimensional ball centered at the origin with radius T. Fig. 2(c) plots an instance of point cloud registration using the* Bunny *dataset [36] (outlier correspondences are shown in red).*

Example 4 (Mesh Registration [22], [78])**.** *Given a set of N putative correspondences from a 3D point cloud to a 3D mesh, where the point cloud $\{(\boldsymbol{p}_i,\boldsymbol{u}_i)\}_{i=1}^N$ is represented as a collection of points ($\boldsymbol{p}_i \in \mathbb{R}^3$) with estimated normals ($\boldsymbol{u}_i \in \mathcal{S}^2$), and the mesh $\{(\boldsymbol{q}_i,\boldsymbol{v}_i)\}_{i=1}^N$ is represented as a collection of*



(a) Single rotation averaging (b) Multiple rotation averaging (c) Point cloud registration (d) Mesh registration (e) Absolute pose estimation (f) Category-level perception

Fig. 2. Perception examples considered in this paper that can be modeled as polynomial optimization problems (*cf.* Examples 1-6).

faces with unit normals ($\boldsymbol{v}_i \in \mathcal{S}^2$) and arbitrary points that belong to them ($\boldsymbol{q}_i \in \mathbb{R}^3$), mesh registration seeks the best rigid transformation $\boldsymbol{x} = (\boldsymbol{R}, \boldsymbol{t}) \in \mathrm{SO}(3) \times \mathbb{R}^3$ to align the point cloud with the mesh. The residual function is chosen as: $r(\boldsymbol{x}, \boldsymbol{z}_i) = \sqrt{\|\langle \boldsymbol{v}_i, \boldsymbol{q}_i - \boldsymbol{R}\boldsymbol{p}_i - \boldsymbol{t}\rangle\|^2 + \|\boldsymbol{v}_i - \boldsymbol{R}\boldsymbol{u}_i\|^2}$, where $\|\langle \boldsymbol{v}_i, \boldsymbol{q}_i - \boldsymbol{R}\boldsymbol{p}_i - \boldsymbol{t}\rangle\|$ is the point-to-plane distance, and $\|\boldsymbol{v}_i - \boldsymbol{R}\boldsymbol{u}_i\|$ is the normal-to-normal distance. Similar to Example 3, we enforce $\boldsymbol{t} \in \mathcal{B}^3_T$. Fig. 2(d) visualizes an instance of mesh registration using the TeddyBear *model from the* HomebrewedDB *dataset [53] (outlier correspondences shown in red).*

Example 5 (Absolute Pose Estimation [56], [99])**.** *Consider a camera with field of view (FOV) $\alpha \in (0, \pi)$ picturing a 3D object (conventionally centered at zero). Given a set of N putative correspondences between 3D keypoints $\{\boldsymbol{p}_i \in \mathbb{R}^3\}_{i=1}^N$ on the object and 2D image keypoint detections $\{\boldsymbol{u}_i \in \mathcal{S}^2\}_{i=1}^N$, where $\boldsymbol{u}_i$ denotes the unit bearing vector corresponding to the i-th 2D keypoint, absolute pose estimation (also known as* Perspective-n-Points*) seeks to estimate the absolute camera pose $\boldsymbol{x} = (\boldsymbol{R}, \boldsymbol{t}) \in \mathrm{SO}(3) \times \mathbb{R}^3$ from the 2D-3D correspondences. The residual function is chosen as: $r(\boldsymbol{x}, \boldsymbol{z}_i) = \sqrt{\langle \boldsymbol{R}\boldsymbol{p}_i + \boldsymbol{t}, (\mathbf{I}_3 - \boldsymbol{u}_i\boldsymbol{u}_i^{\mathsf{T}})(\boldsymbol{R}\boldsymbol{p}_i + \boldsymbol{t})\rangle}$,* i.e., *the point-to-line distance from the transformed 3D keypoint $\boldsymbol{R}\boldsymbol{p}_i + \boldsymbol{t}$ (in camera frame) to the bearing vector $\boldsymbol{u}_i$.*[2] *In this paper, we enforce $\boldsymbol{t} \in \mathcal{B}^3_T \cap \mathcal{C}_\alpha$, where $\mathcal{C}_\alpha \triangleq \{\boldsymbol{t} \in \mathbb{R}^3 \mid \tan(\frac{\alpha}{2})t_3 \geq \sqrt{t_1^2 + t_2^2}\}$ defines the 3D cone corresponding to the camera FOV; the constraint $\boldsymbol{t} \in \mathcal{C}_\alpha$ enforces the center of the 3D object* (i.e., *$\boldsymbol{R} \cdot \mathbf{0} + \boldsymbol{t} = \boldsymbol{t}$ in camera frame) to lie inside the FOV. Fig. 2(e) shows an instance of absolute pose estimation using a satellite image from the* SPEED *dataset [76] (outliers in red).*

Example 6 (Category-Level Object Pose and Shape Estimation [77])**.** *Given N 3D semantic keypoint observations $\{\boldsymbol{p}_i\}_{i=1}^N$ of an object of a certain category* (e.g., *car, chair), category-level perception estimates the object pose and shape. We consider the standard* active shape model, *where the unknown shape of the object is described as a nonnegative combination of K shapes in a library $\{\{\boldsymbol{q}_{k,i}\}_{i=1}^N\}_{k=1}^K$ (the* bases, *which intuitively correspond to examples of objects in that category). Hence, category-level perception estimates the pose $(\boldsymbol{R}, \boldsymbol{t}) \in \mathrm{SO}(3) \times \mathbb{R}^3$ and shape coefficients $\boldsymbol{c} \in \mathbb{R}^K_+$ describing the object. The residual function is chosen as: $r(\boldsymbol{x}, \boldsymbol{z}_i) = \|\boldsymbol{R}\boldsymbol{p}_i + \boldsymbol{t} - \sum_{k=1}^K c_k \boldsymbol{q}_{k,i}\|$,* i.e., *the Euclidean distance between the transformed 3D keypoint detections and the nonnegative combination of the shape bases. We include $\psi(\boldsymbol{x}, \lambda) = \lambda \|\boldsymbol{c}\|^2$ as a regularization for the shape parameters $\boldsymbol{c}$, as in [77]. Again, we enforce $\boldsymbol{t} \in \mathcal{B}^3_T, \boldsymbol{c} \in \mathcal{B}^K_T$ to be both bounded. Fig. 2(f) pictures an example of category-level perception from the* ApolloScape *dataset [89], where one estimates the pose and shape of a vehicle given 2D semantic keypoint detections with associated depth values (outliers shown in red).*

Proposition 7 (Polynomial Expressibility)**.** *Examples 1-6 satisfy Assumption 3. Precisely, (i) r^2 and ψ (if $\psi \neq 0$) are quadratic polynomials* (i.e., $\deg(r^2) = \deg(\psi) = 2$)*; (ii) the constraint set $\mathcal{X}$ can be described by polynomial equalities h_i's and inequalities g_j's with degree up to 2* (i.e., $\deg(h_i), \deg(g_j) \leq 2$)*.*

While we postpone the proof to Supplementary Material, we observe that the key insights behind the proof are simple but powerful: (i) rigid body transformations can be expressed as linear functions (*e.g.*, $\boldsymbol{R}\boldsymbol{p}_i + \boldsymbol{t}$ for a given point $\boldsymbol{p}_i$), (ii) squared residuals r^2 (and our regularizer ψ) are commonly squared L2 norms, that can be written as quadratic functions, and (iii) the set of poses and rotations can be described by quadratic (in-)equality constraints, a fact already used in, *e.g.*, [23], [28], [82], [96].

Proposition 4 and 7 together establish that outlier-robust geometric perception (Robust) with TLS, MC, GM, TB, L1, Huber and Adaptive costs (Fig. 1), when applied to Examples 1-6 (Fig. 2), are instances of (POP). The expert reader will also recognize other geometric perception problems that satisfy Assumption 3, including 2D-2D relative pose estimation [23], triangulation [4], rotation search (Wahba problem) [95], pose graph optimization [73], among others.

For the rest of the paper, we will focus on designing certifiable algorithms and semidefinite relaxations for (Robust) with the (TLS) cost function. However, semidefinite relaxations proposed in Section 4 can be extended to the other costs in Proposition 4, and we leave that exercise to the interested reader. We end this section with a remark about why we prefer the TLS cost over the others in Proposition 4.

Remark 8 (Preference for TLS)**.** *(i) Compared to GM, TB, L1 and Huber, which still penalize outliers, TLS completely discards outliers. Consequently, TLS can often achieve better robustness to outliers [80], [93]. (ii) MC also completely discards outliers, but it does not select a model to minimize the inlier residuals. Therefore, there can be an infinite number of solutions to problem (Robust) with equal cost (number of outliers). (iii) The adaptive cost typically leads to POPs with high-degree polynomials, which requires a large κ from the relaxation hierarchy and results in SDPs that are intractable. (iv) TLS can be shown as a maximum likelihood estimator, when the inliers have a Gaussian distribution and the outliers are uniformly distributed, see [7, Proposition 5].*

4 Sparse Semidefinite Relaxation

In the previous section, we showed how to rephrase the TLS cost as a nonconvex polynomial optimization in $\widetilde{\boldsymbol{x}} \triangleq$

[2] Instead of using the geometric reprojection error as the residual (a rational function), we follow [99] and choose the point-to-line distance as the residual so that r^2 is a polynomial per Assumption 3.

$[\boldsymbol{x}\,;\,\boldsymbol{\theta}] \in \mathbb{R}^{d+N}$. The goal of this section is to design algorithms that can solve (TLS) to certifiable global optimality.

Can we just use Lasserre's hierarchy? Before introducing our sparse semidefinite relaxation, let us attempt to apply the dense Lasserre's hierarchy (LAS) to (TLS). We know that the objective in (TLS) has degree 3,[3] thus $\kappa \geq 2$ is needed for (LAS). In fact, as we have shown in [97], (LAS) at $\kappa = 2$ is empirically exact (on small problem instances). However, as we can see from Examples 1-6, the problems we care about have minimum $d = 9$ (a 3D rotation in Example 1) and maximum $d = 9n$ (n 3D rotations in Example 2) with n being as large as a few hundreds, and meanwhile, it is desirable to be able to handle $N = 100$ measurements. Choosing $d = 10, N = 100, \kappa = 2$, the SDP resulting from the dense relaxation (LAS) has $n_1 = 6216, m_{\text{mom}} = 12,649,561$; when $d = 100, N = 100, \kappa = 2$, such SDP would have $n_1 \approx 2 \times 10^4, m_{\text{mom}} \approx 1.4 \times 10^8$. In both cases, it is hopeless to solve the resulting SDPs using existing solvers.

Sparse semidefinite relaxation (SSR). Now we present a semidefinite relaxation that is much more scalable than (LAS). Note that the fundamental reason why (LAS) leads to an intractable SDP is the use of the *dense* monomial basis $[\widetilde{\boldsymbol{x}}]_\kappa$ for building the moment matrix $\boldsymbol{X}_\kappa$. Although the full set of monomials $[\widetilde{\boldsymbol{x}}]_\kappa$ is necessary when the polynomials p, h_i, g_j contain all monomials up to degree 2κ, in practice p, h_i, g_j are almost always *sparse* (*i.e.*, include a small set of monomials). Therefore, the crux of our semidefinite relaxation is to construct a sparse set of monomials that result in a much smaller moment matrix. Towards this, we analyze the sparsity of the objective and constraint polynomials in (TLS) and observe they only contain three types of monomials:

(i) $[\boldsymbol{x}]_2$, coming from r^2 and ψ in the objective, and polynomials defining the feasible set $\mathcal{X}$ (*cf.* Proposition 7);

(ii) $\theta_i \cdot [\boldsymbol{x}]_2, i = 1 \ldots, N$, coming from $\theta_i r^2$ and θ_i in the objective for $i = 1, \ldots, N$; and

(iii) $\theta_i^2, i = 1, \ldots, N$, coming from the equality constraints $\theta_i^2 - 1 = 0$ for $i = 1, \ldots, N$.

Therefore, it is easy to see that, with the Kronecker product denoted by "$\otimes$", choosing the sparse basis

$$\boldsymbol{v}(\widetilde{\boldsymbol{x}}) \triangleq [1\,;\,\boldsymbol{x}\,;\,\boldsymbol{\theta}\,;\,\boldsymbol{\theta}\otimes\boldsymbol{x}] \in \mathbb{R}^{n_1},\ n_1 \triangleq (1+d)(1+N) \quad (15)$$

leads to the following moment matrix

$$\boldsymbol{X}_v \triangleq \boldsymbol{v}\boldsymbol{v}^{\mathsf{T}} = \begin{bmatrix} 1 & \boldsymbol{x}^{\mathsf{T}} & \boldsymbol{\theta}^{\mathsf{T}} & \boldsymbol{\theta}^{\mathsf{T}}\otimes\boldsymbol{x}^{\mathsf{T}} \\ \boldsymbol{x} & \boldsymbol{x}\boldsymbol{x}^{\mathsf{T}} & \boldsymbol{x}\boldsymbol{\theta}^{\mathsf{T}} & \boldsymbol{x}(\boldsymbol{\theta}^{\mathsf{T}}\otimes\boldsymbol{x}^{\mathsf{T}}) \\ \boldsymbol{\theta} & \boldsymbol{\theta}\boldsymbol{x}^{\mathsf{T}} & \boldsymbol{\theta}\boldsymbol{\theta}^{\mathsf{T}} & \boldsymbol{\theta}(\boldsymbol{\theta}^{\mathsf{T}}\otimes\boldsymbol{x}^{\mathsf{T}}) \\ \boldsymbol{\theta}\otimes\boldsymbol{x} & (\boldsymbol{\theta}\otimes\boldsymbol{x})\boldsymbol{x}^{\mathsf{T}} & (\boldsymbol{\theta}\otimes\boldsymbol{x})\boldsymbol{\theta}^{\mathsf{T}} & \boldsymbol{\theta}\boldsymbol{\theta}^{\mathsf{T}}\otimes\boldsymbol{x}\boldsymbol{x}^{\mathsf{T}} \end{bmatrix} \quad (16)$$

that contains all the three types of monomials ($[\boldsymbol{x}]_2$, $\theta_i \cdot [\boldsymbol{x}]_2$, and θ_i^2) in (i)-(iii). Therefore, *we can write the objective and constraint polynomials in* (TLS) *as linear functions of the smaller moment matrix* (16). Clearly, the advantage is that the size of the moment matrix is now $(1+d)(1+N)$, which is much smaller than $\binom{d+N+\kappa}{\kappa}$ (for $\kappa = 2$) from Lasserre's hierarchy.

Now we can formulate our sparse relaxation using $\boldsymbol{X}_v$ in (16), by following the same procedure as in Section 2.2.

(i) Rewriting (TLS) *using the sparse moment matrix* $\boldsymbol{X}_v$. Because the sparse moment matrix $\boldsymbol{X}_v$ contains all monomials in the objective and constraint polynomials of (TLS), we can write them as linear functions of $\boldsymbol{X}_v$. For example, the objective can be written as $\langle \boldsymbol{C}_1, \boldsymbol{X}_v \rangle$.

(ii) Relaxing the rank-1 constraint on $\boldsymbol{X}_v$. By construction, $\boldsymbol{X}_v$ belongs to the set of rank-one positive semidefinite matrices. Since the rank constraint is non-convex, we drop it and only enforce $\boldsymbol{X}_v$ to be positive semidefinite: $\boldsymbol{X}_v \succeq 0$.

(iii) Adding redundant constraints. First, similar to the dense relaxation case, some monomials can repeat themselves at multiple entries of $\boldsymbol{X}_v$. For example, in (16), the "$\boldsymbol{\theta}\otimes\boldsymbol{x}$" block is the same as the "$\boldsymbol{\theta}\boldsymbol{x}^{\mathsf{T}}$" block up to rearrangement of entries. In fact, the number of *unique* monomials in $\boldsymbol{X}_v$ is $m_{2v} = \mathfrak{t}(d+1)\mathfrak{t}(N+1)$, while the dimension of $\boldsymbol{X}_v$ (in terms of a symmetric matrix) is $\mathfrak{t}((1+d)(1+N))$. Therefore, we can add a total number of $m_{\text{mom}} = \mathfrak{t}((1+d)(1+N)) - m_{2v} + 1$ *moment constraints*:

$$\text{moment constraints}: \quad \langle \boldsymbol{A}_{\text{mom},j}, \boldsymbol{X}_v \rangle = 0, \quad j = 1, \ldots, m_{\text{mom}}, \quad (17)$$

to enforce the repeating monomials in $\boldsymbol{X}_v$ to be equal to each other, as well as the leading entry $[\boldsymbol{X}_v]_{11} = 1$.

Second, we add redundant equality constraints. For each equality constraint h_i in (TLS), we denote $[\widetilde{\boldsymbol{x}}]_{h_i}$ as the largest set of unique monomials such that $h_i \cdot [\widetilde{\boldsymbol{x}}]_{h_i}$ only contains monomials in $\boldsymbol{X}_v$. Formally,

$$[\widetilde{\boldsymbol{x}}]_{h_i} \triangleq \{\widetilde{\boldsymbol{x}}^{\boldsymbol{\alpha}} \mid \text{mono}\,(h_i \cdot \widetilde{\boldsymbol{x}}^{\boldsymbol{\alpha}}) \subseteq \text{mono}\,(\boldsymbol{X}_v)\}, \quad (18)$$

where $\text{mono}\,(\cdot)$ returns the set of unique monomials of a polynomial (or of a matrix of polynomials). Consequently, we can write $h_i \cdot [\widetilde{\boldsymbol{x}}]_{h_i} = \boldsymbol{0}$ as linear equalities in $\boldsymbol{X}_v$:

$$\text{(redundant) equality constraints}: \langle \boldsymbol{A}_{\text{req},ij}, \boldsymbol{X}_v \rangle = 0, \quad i = 1, \ldots, l_h, \quad j = 1, \ldots, |[\widetilde{\boldsymbol{x}}]_{h_i}|\,. \quad (19)$$

Note that since each $[\widetilde{\boldsymbol{x}}]_{h_i}$ must include the monomial "1", eq. (19) includes the original equality constraints h_i in (TLS).

Finally, for each inequality constraint g_j (recall $\deg\,(g_j) \leq 2$ by Proposition 7), we denote by $[\boldsymbol{X}_1]_{\mathcal{I}_j,\mathcal{I}_j}$ the maximum principal submatrix of $\boldsymbol{X}_1$ (*i.e.*, order-one full moment matrix) such that $g_j \cdot [\boldsymbol{X}_1]_{\mathcal{I}_j,\mathcal{I}_j}$ only contains monomials in $\boldsymbol{X}_v$. Formally, the indices $\mathcal{I}_j$ are selected as:

$$\mathcal{I}_j = \arg\max_{\mathcal{J}}\{|\mathcal{J}| \mid \text{mono}\,(g_j \cdot [\boldsymbol{X}_1]_{\mathcal{J},\mathcal{J}}) \subseteq \text{mono}\,(\boldsymbol{X}_v)\}. \quad (20)$$

As a result, calling $\boldsymbol{X}_{g_j} = g_j \cdot [\boldsymbol{X}_1]_{\mathcal{I}_j,\mathcal{I}_j}$, which is positive semidefinite by construction, we can write down the following localizing matrices and constraints:

$$\text{localizing matrices}: \quad \boldsymbol{X}_{g_j} \succeq 0, \quad j = 1, \ldots, l_g \quad (21)$$

$$\text{localizing constraints}: \langle \boldsymbol{A}_{\text{loc},jkh}, \boldsymbol{X}_v \rangle = [\boldsymbol{X}_{g_j}]_{hk} \quad j = 1, \ldots, l_g, \quad 1 \leq h \leq k \leq |\mathcal{I}_j|\,, \quad (22)$$

where the linear constraints in (22) simply enforce each entry of $\boldsymbol{X}_{g_j}$ to be a linear combination of entries in $\boldsymbol{X}_v$.

Steps (i)-(iii) above lead to the following SDP:

$$f^\star = \min_{\boldsymbol{X}=(\boldsymbol{X}_v,\boldsymbol{X}_1,\ldots,\boldsymbol{X}_{l_g})} \{\langle \boldsymbol{C}_1, \boldsymbol{X}_v \rangle \mid \mathcal{A}(\boldsymbol{X}) = \boldsymbol{b}, \boldsymbol{X} \succeq 0\}, \quad \text{(SSR)}$$

where we have shorthanded $\boldsymbol{X}_j = \boldsymbol{X}_{g_j}$ for notation convenience, and $\mathcal{A}(\boldsymbol{X}) = \boldsymbol{b}$ collects all the linear equality constraints in (17), (19) and (22).

Similar to Theorem 2 for (LAS), we have the following result for (SSR) about certifiable global optimality.

[3] The residuals $r^2(\boldsymbol{x}, \boldsymbol{z}_i)$ are quadratic from Proposition 7, hence the terms $\theta_i r^2(\boldsymbol{x}, \boldsymbol{z}_i)$ in the objective of (TLS) become cubic.

Theorem 9 (Sparse Semidefinite Relaxation for (TLS)). *Denote by $p(\boldsymbol{x}, \boldsymbol{\theta})$ the objective function of (TLS), by $p^\star$ the optimum of (TLS), and by $f^\star$ the optimum of (SSR), we have*

(i) (lower bound) $f^\star \leq p^\star$, and

(ii) (rank-one solutions) if $f^\star = p^\star$, then for each global minimizer $\widetilde{\boldsymbol{x}}^\star = (\boldsymbol{x}^\star, \boldsymbol{\theta}^\star)$ of (TLS), its rank-one lifting $\boldsymbol{X}_v = \boldsymbol{v}(\widetilde{\boldsymbol{x}}^\star)\boldsymbol{v}(\widetilde{\boldsymbol{x}}^\star)^\mathsf{T}$ is optimal for (SSR).

Further, assume $\boldsymbol{X}_v^\star$ is an optimal solution of (SSR) and let $\boldsymbol{v}$ be its eigenvector corresponding to the maximum eigenvalue; define the rounding function $(\widehat{\boldsymbol{x}}, \widehat{\boldsymbol{\theta}}) = \texttt{rounding}(\boldsymbol{v})$ as

$$\boldsymbol{v} \leftarrow \boldsymbol{v}/\boldsymbol{v}_1, \; \widehat{\boldsymbol{x}} = \Pi_{\mathcal{X}}(\boldsymbol{v}_x), \; \widehat{\boldsymbol{\theta}} = \operatorname{sgn}(\boldsymbol{v}_\theta), \tag{23}$$

where $\boldsymbol{v}_x$ (resp. $\boldsymbol{v}_1, \boldsymbol{v}_\theta$) takes the entries of $\boldsymbol{v}$ corresponding to monomials $\boldsymbol{x}$ (resp. $1, \boldsymbol{\theta}$) in (15), $\operatorname{sgn}(a)$ returns the sign of a scalar "a", and $\Pi_{\mathcal{X}}$ denotes the projection onto set $\mathcal{X}$. We have

(iii) (optimality certification) if $\widehat{p} \triangleq p(\widehat{\boldsymbol{x}}, \widehat{\boldsymbol{\theta}}) = f^\star$, then the relaxation (SSR) is exact, $f^\star = p^\star = \widehat{p}$, and $(\widehat{\boldsymbol{x}}, \widehat{\boldsymbol{\theta}})$ is a global optimizer to the original (TLS) problem.

Theorem 9 states that (SSR) is a convex relaxation for (TLS) and provides a numerical procedure for *certifying* the exactness of the relaxation. In particular, after solving (SSR) and obtaining an optimizer $\boldsymbol{X}_v^\star$ with cost $f^\star$, one can *round* a feasible solution to (TLS) from the leading eigenvector of $\boldsymbol{X}_v^\star$ using (23). If the rounded solution $(\widehat{\boldsymbol{x}}, \widehat{\boldsymbol{\theta}})$ attains the same cost as $f^\star$, then the relaxation is exact and $(\widehat{\boldsymbol{x}}, \widehat{\boldsymbol{\theta}})$ is guaranteed to be a global minimizer of (TLS). Note that for our Examples 1-6, the feasible set $\mathcal{X}$ includes $\mathrm{SO}(q)$, whose projections can be performed in closed form, and $\mathcal{B}_T^3$, $\mathcal{C}_\alpha$, $\mathcal{B}_T^3 \cap \mathcal{C}_\alpha$, $\mathcal{B}_T^K \cap \mathbb{R}_+^K$, all of which are *low-dimensional convex* sets whose projections can be computed to arbitrary accuracy using standard convex solvers. Therefore, the `rounding` procedure (23) can be done efficiently.

Relative suboptimality. Even if $\widehat{p} \neq f^\star$ in Theorem 9(iii), we can compute a *relative suboptimality* of the rounded solution $(\widehat{\boldsymbol{x}}, \widehat{\boldsymbol{\theta}})$. Precisely, since $f^\star \leq p^\star \leq \widehat{p}$, we compute

$$\eta_s \triangleq |f^\star - \widehat{p}| / (1 + |f^\star| + |\widehat{p}|) \tag{24}$$

as a measure of suboptimality ($\eta_s = 0$ implies $(\widehat{\boldsymbol{x}}, \widehat{\boldsymbol{\theta}})$ is optimal and (SSR) is exact). As an advanced reading, in Supplementary Material, we discuss how to compute a relative suboptimality measure that is not sensitive to potential numerical inaccuracies in the computation of $f^\star$.

Scalability improvement. Table 1 compares the size of the SDP from our sparse relaxation (SSR) with that from the standard Lasserre's hierarchy (LAS), in terms of the size of the maximum positive semidefinite block n_1 and the number of moment constraints m_{mom} (in our problems, over 60% of the equality constraints are moment constraints, hence m_{mom} is representative of the size of the SDP). For illustration purpose, Fig. 3 plots n_1 and m_{mom} as N increases from 20 to 200, when applying (LAS) and (SSR) to Example 1 ($d = 9$). We can observe a drastic reduction in both n_1 and m_{mom} when using (SSR). Notably, when $N = 200$, $n_1 > 20,000$ and $m_{\text{mom}} > 100,000,000$ if using (LAS), while $n_1 \approx 2,000$ and $m_{\text{mom}} \approx 1,000,000$ if using (SSR). This is about 10 times reduction in n_1 and 100 times reduction in m_{mom}. Certainly, such scalability improvement would be meaningless if (SSR) is *inexact* and fails to solve the original (TLS) problem to global optimality. However, as we will show in Section 6, (SSR) is empirically exact across all Examples 1-6, even in the presence of many outliers.

TABLE 1
Comparison of the sizes of two semidefinite relaxations applied to problem (TLS). n_1: size of the largest positive semidefinite block (*i.e.*, the moment matrix), m_{mom}: number of moment constraints.

Relaxation	n_1	m_{mom}
(LAS), $\kappa = 2$	$(\underline{d+N})_2$	$\mathfrak{t}((\underline{d+N})_2) - (\underline{d+N})_4 + 1$
(SSR)	$(1+d)(1+N)$	$\mathfrak{t}((1+d)(1+N)) - \mathfrak{t}(1+d)\mathfrak{t}(1+N) + 1$

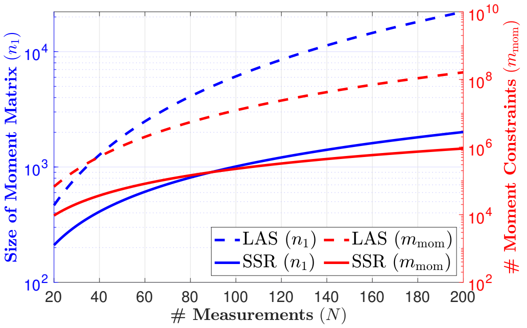


Fig. 3. Comparison of SDP sizes when applying (LAS) and (SSR) to Example 1 ($d = 9$) with N increased from 20 to 200.

Further reduction on Example 2. For multiple rotation averaging, the dimension of the geometric model is $d = 2n$ (2D) or $d = 9n$ (3D), where n is the number of nodes of a graph. Practical rotation averaging problems in structure from motion and SLAM can have n and N being a few hundreds to a few thousands [39], [73]. Taking $d = 400, N = 20$ leads to $m_{\text{mom}} = 16,842,001$ that is still too large. In Supplementary Material, we present a method to further reduce the size of the sparse monomial basis in (15).

We end this section with a remark about how to exploit sparsity while preserving exactness of the relaxation.

Remark 10 (Exploiting Sparsity). *(i) A sparse relaxation can be exact only when the dense relaxation (LAS) is tight. Therefore, we believe it is good practice to first obtain a tight relaxation using (LAS) at certain κ (as we have done in [97]), and then try to find a sparse monomial basis at that κ. (ii) When the dense relaxation is exact, it is nontrivial to decide if a sparse relaxation will be tight without empirical evaluation. For example, replacing (15) with $\boldsymbol{v}(\widetilde{\boldsymbol{x}}) = [[\boldsymbol{x}]_2 \,;\, \boldsymbol{\theta}]$ is also a valid sparse relaxation — the corresponding moment matrix includes all monomials in (i)-(iii)— but it is far from being exact. (iii) In parallel to our previous work [97], recently [87] has presented a methodology,* TSSOS*, to systematically exploit term sparsity for general POPs. However,* TSSOS *tends to find a larger monomial basis when compared to problem-specific techniques such as (SSR) in this paper. For Example 5 with $N = 10$ measurements, the dense monomial basis has dimension 276, our sparse basis (15) has dimension 143, but* TSSOS *with "maximal chordal extension" finds a sparse basis that has dimension 246 and is a strict superset of (15).*

5 STRIDE: Scalable SDP Solver

The sparse relaxation (SSR) leads to an SDP that can still have m as large as hundreds of thousands when N is large (*cf.* Fig. 3). Therefore, with IPMs such as MOSEK, the scale at which (SSR) can be solved is still quite limited (recall IPMs can typically handle m up to 50,000). This section presents STRIDE (*SpecTrahedral pRoximal gradIent Descent along vErtices*), an SDP solver that goes far beyond IPMs and enables solving (SSR) on problems of moderate but realistic size.

Intuition. The key insight behind STRIDE comes from Theorem 9(ii): assuming the relaxation (SSR) is exact, then the SDP (P) admits *rank-one* optimal solutions $\boldsymbol{X}_v^\star = \boldsymbol{v}(\widetilde{\boldsymbol{x}}^\star)\boldsymbol{v}(\widetilde{\boldsymbol{x}}^\star)^\mathsf{T}$, where $\widetilde{\boldsymbol{x}}^\star = (\boldsymbol{x}^\star, \boldsymbol{\theta}^\star)$ corresponds to the global minimizer of (TLS). Therefore, STRIDE tries to move between rank-one matrices in the feasible set of the SDP (these are the vertices of the spectrahedron [18]), searching for a globally optimal solution. More in detail, STRIDE employs a globally convergent *proximal gradient method* (PGM) as the backbone for solving the convex SDP (SSR), but blends *short* PGM steps with *long* rank-one steps generated by fast NLP algorithms on the POP (TLS). Intuitively, the long rank-one steps circumvent the slow convergence of PGM, while the PGM backbone allows escaping local minima where the NLP algorithm can be stuck in.

With this insight, we now develop the details of STRIDE.

Short PGM step. The backbone of STRIDE implements a PGM for solving the primal SDP (P). Given an initial point $\boldsymbol{X}^0 \in \mathbb{X}$, the k-th ($k \geq 0$) iteration of PGM performs

$$\boldsymbol{X}^{k+1} = \Pi_{\mathcal{F}_\mathrm{P}}\left(\boldsymbol{X}^k - \sigma_k \boldsymbol{C}\right), \tag{PGM}$$

for a given constant $\sigma_k > 0$, where $\Pi_{\mathcal{F}_\mathrm{P}}$ denotes the metric projection onto the spectrahedron $\mathcal{F}_\mathrm{P} \triangleq \{\boldsymbol{X} \in \mathbb{X} \mid \mathcal{A}(\boldsymbol{X}) = \boldsymbol{b}, \boldsymbol{X} \succeq 0\}$ (*i.e.*, the feasible set of (P)). In words, the (PGM) step first moves along the direction of the negative gradient for some step size σ_k (recall the objective of (P) is $\langle \boldsymbol{C}, \boldsymbol{X} \rangle$ with a constant gradient $\boldsymbol{C}$), and then projects the new point $\boldsymbol{X}^k - \sigma_k \boldsymbol{C}$ onto the feasible set $\mathcal{F}_\mathrm{P}$. For this reason, this particular version of PGM is also called *projected gradient descent*. It is well known that (PGM) guarantees to converge to an optimal solution of (P), provided that $\sigma_{k+1} \geq \sigma_k, \forall k \geq 0$ (see [14], [15], [51]). In Supplementary Material, we show the Lagrangian dual of the projection subproblem in (PGM) can be reformulated as a *smooth unconstrained optimization*, which allows solving (PGM) for large-scale problems using a limited-memory BFGS (L-BFGS) algorithm.

Long rank-one step. The issue with (PGM) is that the convergence can be slow, particularly when the optimal $\boldsymbol{X}^\star$ is rank one and degenerate (as in (SSR)). Here we propose to exploit the low-rankness of $\boldsymbol{X}^\star$ and accelerate the convergence by generating long rank-one steps. Towards this goal, calling $\overline{\boldsymbol{X}}^{k+1} := \Pi_{\mathcal{F}_\mathrm{P}}(\boldsymbol{X}^k - \sigma_k \boldsymbol{C})$, and $\overline{\boldsymbol{X}}_v^{k+1} \in \mathbb{S}_+^{n_1}$ as the first block in $\overline{\boldsymbol{X}}^{k+1}$ (*i.e.*, the moment matrix), we compute a potentially better rank-one iterate via three steps:

(i) **(Rounding)**. Let $\overline{\boldsymbol{X}}_v^{k+1} = \sum_{i=1}^{n_1} \lambda_i \boldsymbol{v}_i \boldsymbol{v}_i^\mathsf{T}$ be the spectral decomposition of $\overline{\boldsymbol{X}}_v^{k+1}$ with $\lambda_1 \geq \ldots \geq \lambda_{n_1}$ in nonincreasing order. Compute r hypotheses from the leading r eigenvectors

$$(\bar{\boldsymbol{x}}_i^{k+1}, \bar{\boldsymbol{\theta}}_i^{k+1}) = \texttt{rounding}(\boldsymbol{v}_i), \quad i = 1, \ldots, r, \tag{25}$$

where the function `rounding` is defined as in (23).

(ii) **(Local search)**. Apply a local search method for (TLS) using NLP with initial point chosen as $(\bar{\boldsymbol{x}}_i^{k+1}, \bar{\boldsymbol{\theta}}_i^{k+1})$ for each $i = 1, \ldots, r$. Denoting the solution of each local search as $(\widehat{\boldsymbol{x}}_i^{k+1}, \widehat{\boldsymbol{\theta}}_i^{k+1})$, with associated objective value $p(\widehat{\boldsymbol{x}}_i^{k+1}, \widehat{\boldsymbol{\theta}}_i^{k+1})$, choose the best local solution with *minimum* objective value. Formally,

$$\begin{aligned}(\widehat{\boldsymbol{x}}_i^{k+1}, \widehat{\boldsymbol{\theta}}_i^{k+1}) &= \texttt{nlp}(\bar{\boldsymbol{x}}_i^{k+1}, \bar{\boldsymbol{\theta}}_i^{k+1}), \quad i = 1, \ldots, r, && (26a)\\ (\widehat{\boldsymbol{x}}^{k+1}, \widehat{\boldsymbol{\theta}}^{k+1}) &= \operatorname*{arg\,min}_{(\widehat{\boldsymbol{x}}_i^{k+1}, \widehat{\boldsymbol{\theta}}_i^{k+1}), i=1\ldots,r} p(\widehat{\boldsymbol{x}}_i^{k+1}, \widehat{\boldsymbol{\theta}}_i^{k+1}). && (26b)\end{aligned}$$

(iii) **(Lifting)**. Perform a rank-one lifting of the best local solution $\widetilde{\boldsymbol{x}}^{k+1} \triangleq (\widehat{\boldsymbol{x}}^{k+1}, \widehat{\boldsymbol{\theta}}^{k+1})$

$$\begin{aligned}\widehat{\boldsymbol{X}}_v^{k+1} &= \boldsymbol{v}(\widetilde{\boldsymbol{x}}^{k+1})\boldsymbol{v}(\widetilde{\boldsymbol{x}}^{k+1})^\mathsf{T}, \quad (\textit{cf.}\ (15)) && (27a)\\ \widehat{\boldsymbol{X}}_{g_j}^{k+1} &= \boldsymbol{X}_{g_j}^{k+1}(\widetilde{\boldsymbol{x}}^{k+1}), j = 1, \ldots, l_g, \quad (\textit{cf.}\ (20)) && (27b)\\ \widehat{\boldsymbol{X}}^{k+1} &= (\widehat{\boldsymbol{X}}_v^{k+1}, \ldots, \widehat{\boldsymbol{X}}_{g_j}^{k+1}, \ldots)_{j=1}^{l_g}, && (27c)\end{aligned}$$

where $\widehat{\boldsymbol{X}}^{k+1}, \widehat{\boldsymbol{X}}_{g_j}^{k+1}, j = 1, \ldots, l_g$ are computed by *evaluating* the moment and localizing matrices at $\widetilde{\boldsymbol{x}}^{k+1}$.

Taking the right step. Now we are given two candidates for the next iteration, namely the short PGM step $\overline{\boldsymbol{X}}^{k+1}$ (generated by computing the projection of $\boldsymbol{X}^k - \sigma_k \boldsymbol{C}$ onto $\mathcal{F}_\mathrm{P}$) and the long rank-one step $\widehat{\boldsymbol{X}}^{k+1}$ (obtained by rounding, local search, and lifting). Which one should we choose to be the next iterate $\boldsymbol{X}^{k+1}$ such that the entire sequence $\{\boldsymbol{X}^k\}$ is globally convergent? The answer to this question is quite natural –we accept $\widehat{\boldsymbol{X}}^{k+1}$ if and only if it attains a strictly lower cost than $\overline{\boldsymbol{X}}^{k+1}$ (*cf.* eq. (29)).

Algorithm 1: STRIDE

```
Given $(\boldsymbol{X}^0, \boldsymbol{S}^0, \boldsymbol{y}^0) \in \mathcal{K} \times \mathcal{K} \times \mathbb{R}^m$, a tolerance
  $\texttt{tol} > 0$, an integer $r \in [1, n]$, a constant $\epsilon > 0$, a
  nondecreasing sequence $\{\sigma_k > 0\}$. Perform the
  following steps for $k = 0, 1, \ldots$.
1 (Short PGM step). Solve the projection problem
  with initial point $(\boldsymbol{X}^k, \boldsymbol{S}^k, \boldsymbol{y}^k)$
      $(\overline{\boldsymbol{X}}^{k+1}, \boldsymbol{S}^{k+1}, \boldsymbol{y}^{k+1}) = \Pi_{\mathcal{F}_\mathrm{P}}\left(\boldsymbol{X}^k - \sigma_k \boldsymbol{C}\right),$   (28)
  using the L-BFGS algorithm in [100, Section 4].
2 (Long rank-one step). Compute a candidate
  rank-one iterate $\widehat{\boldsymbol{X}}^{k+1}$ from (25)-(27).
3 (Update primal variable). Update $\boldsymbol{X}^{k+1}$ as
      $\boldsymbol{X}^{k+1} = \begin{cases} \widehat{\boldsymbol{X}}^{k+1} & \text{if } f(\widehat{\boldsymbol{X}}^{k+1}) \leq f(\overline{\boldsymbol{X}}^{k+1}) - \epsilon \\ & \phantom{\text{if }} \widehat{\boldsymbol{X}}^{k+1} \in \mathcal{F}_\mathrm{P} \\ \overline{\boldsymbol{X}}^{k+1} & \text{otherwise} \end{cases}.$   (29)
4 (Check convergence). Compute KKT residuals at
  $(\boldsymbol{X}^{k+1}, \boldsymbol{S}^{k+1}, \boldsymbol{y}^{k+1})$ from (5), if $\eta_{\max} < \texttt{tol}$, then
  output $(\boldsymbol{X}^{k+1}, \boldsymbol{S}^{k+1}, \boldsymbol{y}^{k+1})$. Otherwise, go to Step 1.
```

The full STRIDE algorithm is presented in Algorithm 1.

Theorem 11 (Global Convergence)**.** *Suppose the Slater condition for (P) holds and $\{(\boldsymbol{X}^k, \boldsymbol{y}^k, \boldsymbol{S}^k)\}$ is generated by* STRIDE*, then $\{f(\boldsymbol{X}^k)\}$ converges to $f^\star$, where $f^\star$ is the optimum of (P).*

While we provide the proof in Supplementary Material, the intuition is that eq. (29) ensures the rank-one "strides" are accepted only if they strictly decrease the objective value. Therefore, either the last rank-one point is already optimal, or —if it is suboptimal— it still provides an improved reinitialization for (PGM) to globally converge to the optimal $\boldsymbol{X}^\star$. Note that the PGM backbone allows STRIDE to converge even when the optimal solution has rank higher than one.

Although STRIDE is a globally convergent algorithm for solving the primal SDP (P), the initial guess $(\boldsymbol{X}^0, \boldsymbol{S}^0, \boldsymbol{y}^0)$ can have a significant impact on its convergence speed. The next remark states that existing fast heuristics for robust perception can be readily incorporated into STRIDE.

Remark 12 (Fast Heuristics and Certification)**.** *Existing fast heuristics for robust estimation, such as graduated non-convexity* (GNC) *[16], [93] and* RANSAC *[41], can typically return the* globally optimal *solution to* (TLS) *when the measurement set $\mathcal{Z}$ contains a low or medium portion of outliers* (e.g., *below* 70%)*. Therefore, we use* GNC *or* RANSAC *to generate an initial guess for the SDP relaxation* (SSR)*. Formally, calling* $(\widehat{\boldsymbol{x}}, \widehat{\boldsymbol{\theta}})$ *the candidate solution obtained by solving* (TLS) *using* GNC *or* RANSAC*, we generate* $\boldsymbol{X}^0$ *(for* STRIDE*) by applying the lifting procedure in* (27) *to* $(\widehat{\boldsymbol{x}}, \widehat{\boldsymbol{\theta}})$*. Notably, when* $(\widehat{\boldsymbol{x}}, \widehat{\boldsymbol{\theta}})$ *is already globally optimal to* (TLS) *(hence* $\boldsymbol{X}^0$ *is an optimizer of* (SSR) *as long as the relaxation is exact),* STRIDE *only finds a* certificate of optimality *for* $(\widehat{\boldsymbol{x}}, \widehat{\boldsymbol{\theta}})$ *by performing one step of* (PGM) (cf. (28) *in Algorithm 1).*

Fast heuristics provide a good *primal* initialization for STRIDE. However, little information is known about how to obtain a good *dual* initialization. In Supplementary Material, we describe a dual initialization procedure that exploits *correlative sparsity* [86] and leverages a fast first-order algorithm called *semi-proximal ADMM* (also known as ADMM+) [79]. We also give more implementation details about how to use Riemannian optimization to perform local search.

6 Experiments

In this section, we test the sparse relaxation (SSR) and the SDP solver STRIDE on Examples 1-6 using both synthetic and real data. The goal of our experiments is not to claim state-of-the-art efficiency or robustness (*e.g.*, against problem-specific implementations), but rather to show that (SSR) and STRIDE, for the first time, provide a general framework to solve large-scale nonconvex outlier-robust perception problems to certifiable global optimality within reasonable computation time. We believe with the advancement of SDP solvers, our framework will eventually run in real time.

Baselines. We use two state-of-the-art SDP solvers, MOSEK [8] and SDPNAL+ [104], as baseline solvers to compare against STRIDE. We omit MOSEK whenever the SDP becomes too large to be solved by MOSEK (*i.e.*, when $m > 50,000$). We use default settings for both MOSEK and SDPNAL+.

STRIDE's settings. In Algorithm 1, we choose $\texttt{tol} = 1\text{e}{-6}$, $r = 3$, $\epsilon = 1\text{e}{-12}$, $\sigma_k = 10, \forall k$, and run it for a maximum of 5 iterations. As described in Remark 12, we use GNC or RANSAC to initialize the primal variable, and ADMM+ to initialize the dual variable. The local search is performed using Manopt with a trust region solver. Details about local search and ADMM+ can be found in Supplementary Material.

Evaluation metrics. Let $(\widehat{\boldsymbol{X}}, \widehat{\boldsymbol{y}}, \widehat{\boldsymbol{S}})$ be the solution for (SSR) returned by an SDP solver and $(\widehat{\boldsymbol{x}}, \widehat{\boldsymbol{\theta}})$ be the corresponding rounded solution for (TLS). We evaluate the performance of the solver using four metrics: (i) the estimation errors of $\widehat{\boldsymbol{x}}$ compared to the groundtruth, whenever the groundtruth is available; (ii) SDP solution quality, using the maximum KKT residual $\eta_{\max}$ from (5); (iii) certified suboptimality, using the $\texttt{rounding}$ procedure in (23) and the relative suboptimality measure η_s in (24) (we deem a rounded solution as globally optimal if $\eta_s < 1\text{e}{-3}$); (iv) solver CPU time in seconds. For simulation experiments, statistics are computed over 20 Monte Carlo runs per setup.

Hardware. Experiments are performed on a Linux PC with 12-core Intel i9-7920X CPU@2.90GHz and 128GB RAM.

6.1 Single Rotation Averaging

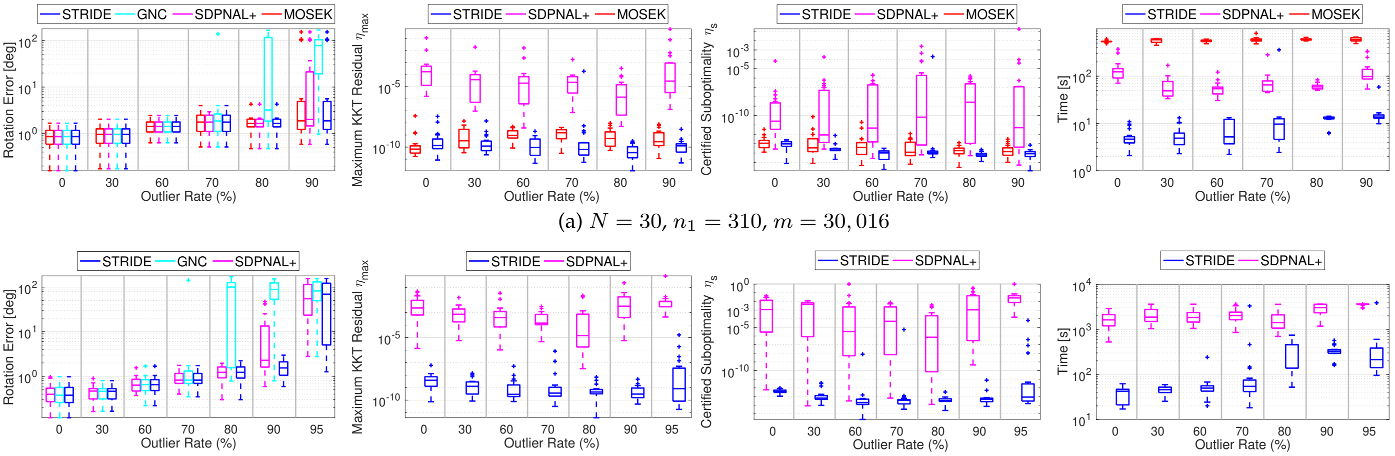
Setup. At each Monte Carlo run, we first randomly generate a groundtruth 3D rotation $\boldsymbol{R}^{\circ}$; then inliers are generated by $\boldsymbol{R}_{\text{in}} = \boldsymbol{R}^{\circ}\boldsymbol{R}_{\varepsilon}$, where the inlier noise $\boldsymbol{R}_{\varepsilon}$ is generated by randomly sampling a rotation axis and a rotation angle $\varepsilon \sim \mathcal{N}(0, \sigma^2)$ with $\sigma = 5°$; outliers are arbitrary random rotations. We test two setups with $N = 30$ and $N = 100$. At $N = 30$, we sweep the outlier ratio from 0% to 90%, while at $N = 100$, we sweep the outlier ratio up to 95%.

Results. Fig. 4(a)-(b) plot the evaluation metrics for $N = 30$ and $N = 100$, respectively. We make the following observations. (i) Our sparse relaxation (SSR) is exact with up to 90% outliers when $N = 30$ and up to 95% outliers when $N = 100$ (the suboptimality η_s is below $1\text{e}{-3}$ for all test runs). (ii) For $N = 30$, STRIDE solves the SDP to an accuracy that is comparable to MOSEK (*cf.* the $\eta_{\max}$ plot), but is about 100 (and up to 270) times faster (*cf.* the time plot). (iii) For $N = 100$, MOSEK cannot run. While SDPNAL+ still runs, its accuracy is at least five orders of magnitude worse than STRIDE (*cf.* the $\eta_{\max}$ plot, where STRIDE attains $1\text{e}{-8}$ accuracy, but SDPNAL+ only attains $1\text{e}{-3}$ accuracy), and its runtime is about 40 times slower than STRIDE. (iv) STRIDE safeguards GNC. While GNC is used to initialize STRIDE, STRIDE can *certify* the global optimality of GNC and escapes the local minima of GNC (*e.g.*, at 80% outlier rate in the rotation error plot, while GNC fails many times, the solution of STRIDE is always correct and optimal). (v) When the outlier rate is too high, global optimality not necessarily implies a correct estimation (in the sense of being close to the groundtruth). For example, at 90% outlier rate when $N = 30$, STRIDE and MOSEK both obtain certifiable optimality ($\eta_s = 0$), but the rotation error can be quite large (about 100°). Similarly, at 95% outlier rate when $N = 100$, the optimal estimate obtained by STRIDE also has large rotation errors. For further discussion about this point, we refer the reader to the notion of *estimation contract* in [101], which ties the number of inliers to the accuracy of the optimal solution of (TLS) w.r.t. the groundtruth, and reports estimation contracts for a 3D registration problem.

6.2 Multiple Rotation Averaging

Setup. We test 2D multiple rotation averaging in a SLAM setting, where a robot traverses a trajectory following a 2D grid pattern (*e.g.*, Fig. 2(b) shows a 3 × 3 grid) with both odometry measurements (between consecutive nodes) and loop closures. We assume the odometry measurements are outlier-free (*i.e.*, we include them in the function $\psi(\boldsymbol{x})$ in Example 2) and only the loop closures could be corrupted by outliers as in [93]. Inlier relative rotations are generated by $\boldsymbol{R}_{\text{in}} = \boldsymbol{R}^{\circ}\boldsymbol{R}_{\varepsilon}$ where $\boldsymbol{R}^{\circ} = \boldsymbol{R}_i^{\mathsf{T}}\boldsymbol{R}_j$ is the groundtruth relative rotation between nodes (i, j) and $\boldsymbol{R}_{\varepsilon}$ is a random 2D rotation with angle $\varepsilon \sim \mathcal{N}(0, \sigma^2)$ ($\sigma = 0.6°$). Outlier relative rotations are arbitrary 2D rotations. We test two cases with increasing outlier rates: a 10×10 grid with $N = 10$ loop closures, and a 20×20 grid with $N = 20$ loop closures.

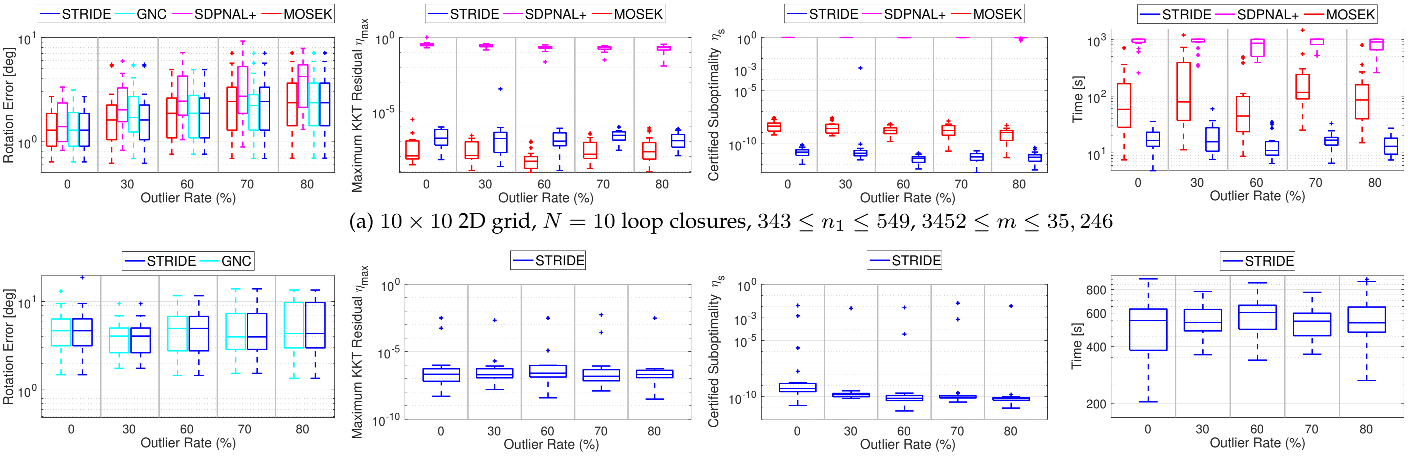
Results. Fig. 5(a)-(b) plot the evaluation metrics for both cases. We make the following observations. (i) For 10×10 grid with $N = 10$, our relaxation is always exact, with up to 80% outliers. In this case, STRIDE can solve the SDP to an accuracy that is comparable to MOSEK, while being about 20



(a) $N = 30, n_1 = 310, m = 30,016$

(b) $N = 100, n_1 = 1010, m = 310,016$

Fig. 4. Single Rotation Averaging (Example 1).



(a) 10×10 2D grid, $N = 10$ loop closures, $343 \leq n_1 \leq 549$, $3452 \leq m \leq 35,246$

(b) 20×20 2D grid, $N = 20$ loop closures, $1483 \leq n_1 \leq 1761$, $64,384 \leq m \leq 189,198$

Fig. 5. 2D Multiple Rotation Averaging (Example 2).

times faster (up to 40 times faster). SDPNAL+, unfortunately, completely fails in this problem. Therefore, we did not run SDPNAL+ for the more challenging 20×20 grid. (ii) For 20×20 grid with $N = 20$, our relaxation is also almost always exact, with up to 80% outliers. However, there exist 1-2 runs per outlier rate where STRIDE fails to obtain an $\eta_s < 1\text{e}{-3}$. In such cases, we suspect the relaxation is inexact.

6.3 Point Cloud Registration

Setup. We first sample a random set of 3D points $\{\boldsymbol{p}_i\}_{i=1}^N$, where each $\boldsymbol{p}_i \sim \mathcal{N}(\mathbf{0}, \mathbf{I}_3)$. Then we generate a random rotation and translation $(\boldsymbol{R}^\circ, \boldsymbol{t}^\circ)$ such that $\|\boldsymbol{t}^\circ\| \leq T = 10$. Using $(\boldsymbol{R}^\circ, \boldsymbol{t}^\circ)$, we generate $\{\boldsymbol{q}_i\}_{i=1}^N$ by $\boldsymbol{q}_i = \boldsymbol{R}^\circ \boldsymbol{p}_i + \boldsymbol{t}^\circ + \boldsymbol{\varepsilon}_i$ ($\boldsymbol{\varepsilon}_i \sim \mathcal{N}(\mathbf{0}, 0.01^2 \mathbf{I}_3)$) if $\boldsymbol{q}_i$ is an inlier, or by $\boldsymbol{q}_i \sim \mathcal{N}(\mathbf{0}, \mathbf{I}_3)$ if $\boldsymbol{q}_i$ is an outlier. We test $N = 20$ and $N = 100$.

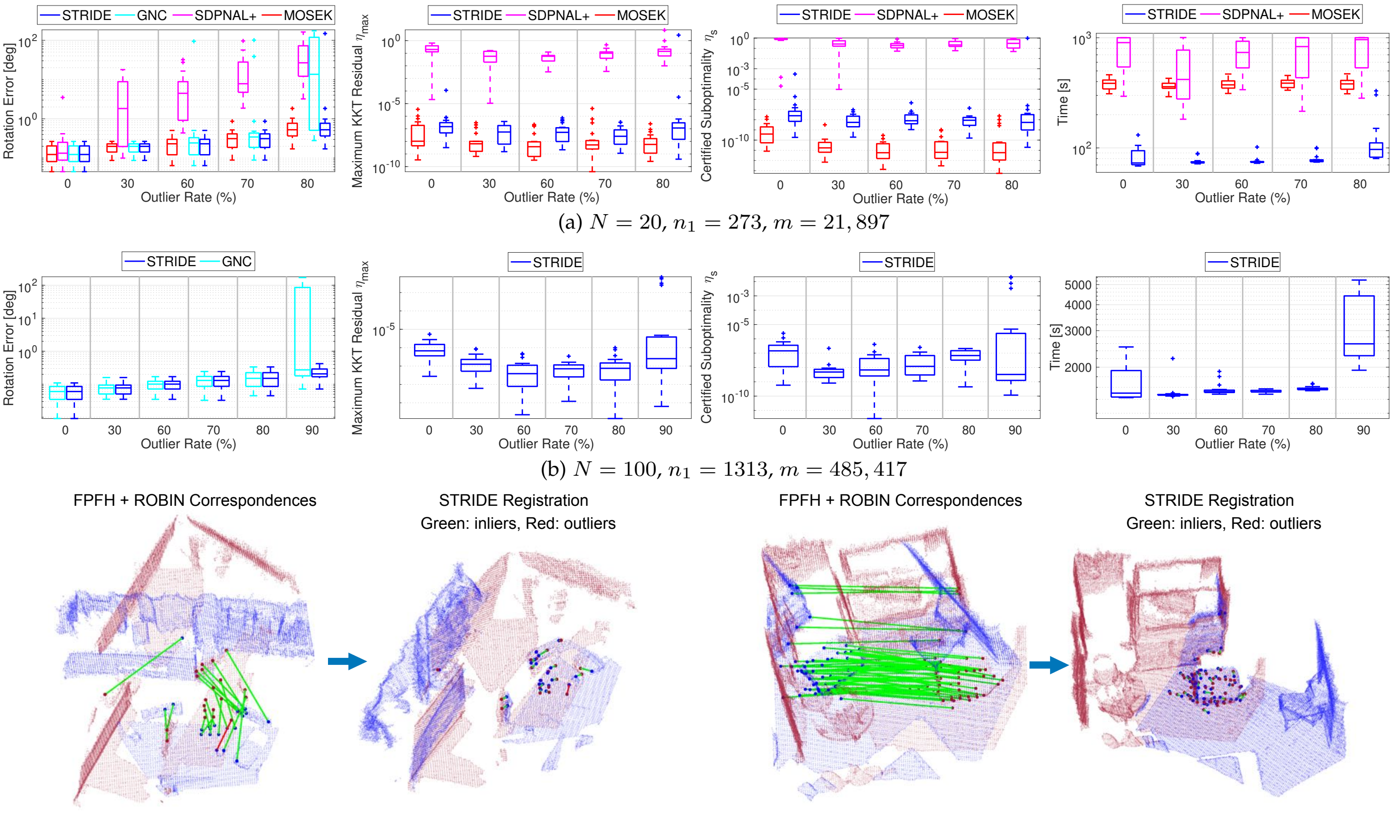
Results. Fig. 6(a)-(b) plot the evaluation metrics for $N = 20$ and $N = 100$, respectively. We make the following observations. (i) When $N = 20$, our relaxation is tight with up to 80% outlier correspondences. Both MOSEK and STRIDE can obtain a certifiably optimal solution, except that STRIDE failed once to attain sufficient accuracy (within 5 iterations) at 80% outlier rate. [4] However, STRIDE is about 5 times faster than MOSEK. SDPNAL+ completely fails in this problem. (ii) When $N = 100$, our relaxation is exact with up to 90% outliers and STRIDE is the only solver that can certify exactness. At 90% outlier rate, STRIDE certified the global optimality for 17 runs, while failed to do so for 3 runs. (iii) STRIDE can certify the success of GNC and escape local minima when GNC fails (*e.g.*, at 60 − 80% when $N = 20$ and at 90% when $N = 100$).

Scan matching on 3DMatch. To showcase the practical value of STRIDE, we perform scan matching using the 3DMatch test data [105]. We use FPFH [74] to generate putative feature matches, followed by using ROBIN [78] to filter out gross outliers. The result of FPFH and ROBIN is typically a set of sparse keypoint matches with only a few outliers. We then use STRIDE to *certifiably* estimate the rigid transformation. Fig. 6(c)-(d) visualize two examples where STRIDE returns certified globally optimal estimates ($\eta_s < 1\text{e}{-6}$). More examples are provided in Supplementary Material.

6.4 Mesh Registration

Setup. We first simulate a random mesh by sampling a set of N 3D planes $\{\boldsymbol{q}_i, \boldsymbol{v}_i\}_{i=1}^N$, where $\boldsymbol{v}_i$ is the unit normal of the plane (by sampling a random 3D direction) and $\boldsymbol{q}_i \sim \mathcal{N}(\mathbf{0}, \mathbf{I}_3)$ is an arbitrary point on the plane. We then generate a random point on each plane via $\boldsymbol{q}'_i =$

[4] Consistent with [97], we empirically noticed that the relaxation breaks earlier when fewer measurements are available. We remark that the formulation considered in this paper is more challenging that the rotation-only version in [95], which remains tight at 90% outliers.



(a) $N = 20, n_1 = 273, m = 21,897$

(b) $N = 100, n_1 = 1313, m = 485,417$

(c) $\boldsymbol{R}$ error: 3.5°, $\boldsymbol{t}$ error: 9.5e−2, $\eta_s = 1.1\text{e}{-10}$, time: 75 [s]

(d) $\boldsymbol{R}$ error: 3.3°, $\boldsymbol{t}$ error: 2.0e−1, $\eta_s = 1.1\text{e}{-7}$, time: 558 [s]

Fig. 6. Point Cloud Registration (Example 3).

$\boldsymbol{q}_i + \boldsymbol{w}_i \times \boldsymbol{v}_i$, where $\boldsymbol{w}_i \sim \mathcal{N}(\mathbf{0}, \mathbf{I}_3)$ is a random point and "×" denotes the vector cross product. After this, we generate a random groundtruth transformation $(\boldsymbol{R}^\circ, \boldsymbol{t}^\circ)$, and transform $(\boldsymbol{q}_i', \boldsymbol{v}_i)$ to be $\boldsymbol{p}_i = \boldsymbol{R}^\circ \boldsymbol{q}_i' + \boldsymbol{t}^\circ + \boldsymbol{\varepsilon}_{pi}$ and $\boldsymbol{u}_i = \texttt{normalize}(\boldsymbol{R}^\circ \boldsymbol{v}_i + \boldsymbol{\varepsilon}_{ni})$, if $(\boldsymbol{p}_i, \boldsymbol{u}_i)$ is an inlier, where $\boldsymbol{\varepsilon}_{pi}, \boldsymbol{\varepsilon}_{ni} \sim \mathcal{N}(\mathbf{0}, 0.01^2 \mathbf{I}_3)$ are random Gaussian noise, and $\texttt{normalize}(\boldsymbol{v}) \triangleq \boldsymbol{v}/\|\boldsymbol{v}\|$ normalizes a vector to have unit norm. If $(\boldsymbol{p}_i, \boldsymbol{u}_i)$ is an outlier, then $\boldsymbol{p}_i$ is a random 3D point and $\boldsymbol{u}_i$ is a random 3D direction. Given the mesh $\{\boldsymbol{q}_i, \boldsymbol{v}_i\}_{i=1}^N$ and the noisy point cloud with normals $\{\boldsymbol{p}_i, \boldsymbol{u}_i\}_{i=1}^N$, we seek the best transformation $(\boldsymbol{R}^\star, \boldsymbol{t}^\star)$ to *align the point cloud to the mesh* using the residual defined in Example 4. After $(\boldsymbol{R}^\star, \boldsymbol{t}^\star)$ is found, its *inverse* transformation is used to compute the estimation errors w.r.t. $(\boldsymbol{R}^\circ, \boldsymbol{t}^\circ)$ (recall $(\boldsymbol{R}^\circ, \boldsymbol{t}^\circ)$ is generated to align the mesh to the point cloud). We test $N = 20$ and $N = 100$ with increasing outlier rates.

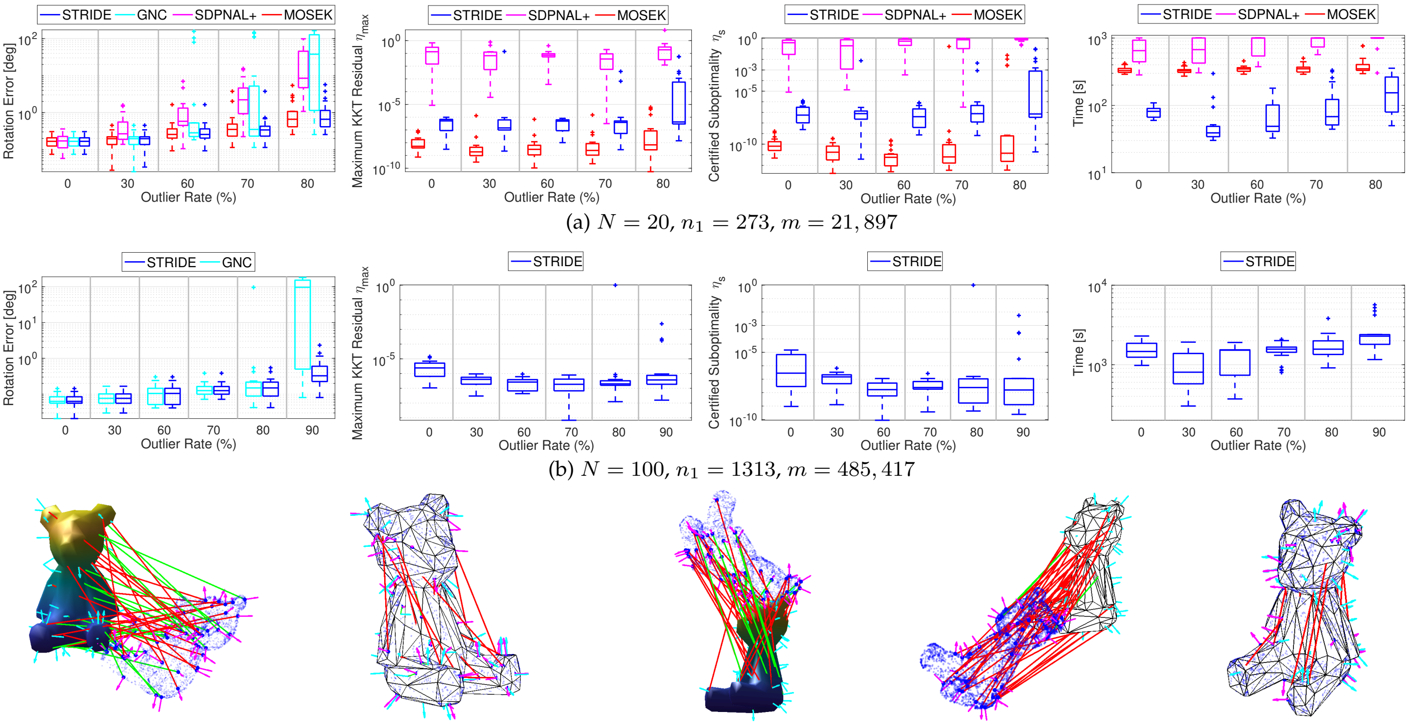
Results. Fig. 7(a)-(b) plots the evaluation metrics for $N = 20$ and $N = 100$, respectively. The results are mostly the same as point cloud registration in Fig. 6(a)-(b), except that when $N = 20$, the relaxation is not always tight at 70% and 80% outlier rates (from the η_s plot of MOSEK we see one inexact run at 70% and three inexact runs at 80%).

Mesh registration with TeddyBear. We perform mesh registration using the TeddyBear mesh model from the HomebrewedDB dataset [53]. From the TeddyBear mesh, we generate a noisy point cloud by densely sampling points on each face of the mesh with additive Gaussian noise, and transform the point cloud using a random rigid transformation. We use the `pcnormals` function in Matlab to estimate surface normals for each point in the point cloud. We then randomly sample $N = 50$ point-to-face correspondences with outliers, and use STRIDE to estimate the rigid transformation. Fig. 7(c-1) shows an instance with 50% outliers, where GNC successfully returns the globally optimal solution and STRIDE computes a certificate of optimality ($\eta_s = 2.5\text{e}{-8}$). Fig. 7(c-2) shows an instance with 70% outliers, where GNC converges to a suboptimal solution but STRIDE escapes the local minimum and finds the globally optimal solution with a certificate of optimality ($\eta_s = 1.1\text{e}{-7}$).

6.5 Absolute Pose Estimation

Setup. We first generate a set of random 3D points $\{\boldsymbol{p}_i\}_{i=1}^N$ that are centered at zero. We then generate a random pose $(\boldsymbol{R}^\circ, \boldsymbol{t}^\circ)$ such that $\|\boldsymbol{t}^\circ\| \leq T = 10$ and $\boldsymbol{t}^\circ$ lies inside the camera FOV cone $\mathcal{C}_\alpha$ with $\alpha = \frac{\pi}{2}$. Using $(\boldsymbol{R}^\circ, \boldsymbol{t}^\circ)$ and $\{\boldsymbol{p}_i\}_{i=1}^N$, we generate 2D keypoints by projecting the transformed 3D points onto the imaging plane, *i.e.*, $\boldsymbol{v}_i = \mathcal{P}(\boldsymbol{R}^\circ \boldsymbol{p}_i + \boldsymbol{t}^\circ)$, where $\mathcal{P} : \mathbb{R}^3 \to \mathbb{R}^2$ is defined as $\mathcal{P}(\boldsymbol{a}) = [a_1/a_3 \,;\, a_2/a_3]$. We then generate the inlier bearing vectors from the 2D keypoints by $\boldsymbol{u}_i = \texttt{normalize}([\boldsymbol{v}_i + \boldsymbol{\varepsilon}_i \,;\, 1])$, where $\boldsymbol{\varepsilon}_i \sim \mathcal{N}(\mathbf{0}, 0.001^2 \mathbf{I}_2)$ is a random 2D Gaussian noise. For outliers, we generate $\boldsymbol{u}_i$ as random unit vectors inside the FOV cone. We test $N = 20$ and $N = 100$ with increasing outlier rates. We use RANSAC implemented in the Matlab function `estimateWorldCameraPose` to initialize STRIDE.

Results. Fig. 8(a)-(b) plot the evaluation metrics. We make the following observations. (i) When $N = 20$, our relaxation is exact with up to 60% outliers. At 70% outlier rate, even if MOSEK solves the SDP to high accuracy, since the solution in not rank one, the rounding procedure obtains a pose estimation that is far from the groundtruth. (ii) When



(a) $N = 20$, $n_1 = 273$, $m = 21,897$

(b) $N = 100$, $n_1 = 1313$, $m = 485,417$

(c-1) Registration using TeddyBear [53]. An example where the globally optimal estimate returned by GNC is certified by STRIDE. $N = 50$, 50% outliers.

(c-2) An example where GNC converges to a local minimum (middle column), but STRIDE escapes it and obtains a certifiably optimal estimate (right column). $N = 50$, 70% outliers.

Fig. 7. Mesh Registration (Example 4).

$N = 100$, our relaxation becomes mostly tight at 70% outlier rate, which suggests that increasing the total number of matches could lead to a tighter relaxation.

Satellite pose estimation on SPEED. We showcase STRIDE on satellite pose estimation using the SPEED dataset [76]. We use the 3D satellite model provided in [30] (with $N = 11$ keypoints) and spoil the groundtruth 2D keypoints with outliers. Fig. 8(c) shows four examples with 2-5 outliers, where STRIDE obtains accurate pose estimates with certified global optimality in less than one minute. More examples are provided in Supplementary Material.

6.6 Category-Level Object Perception

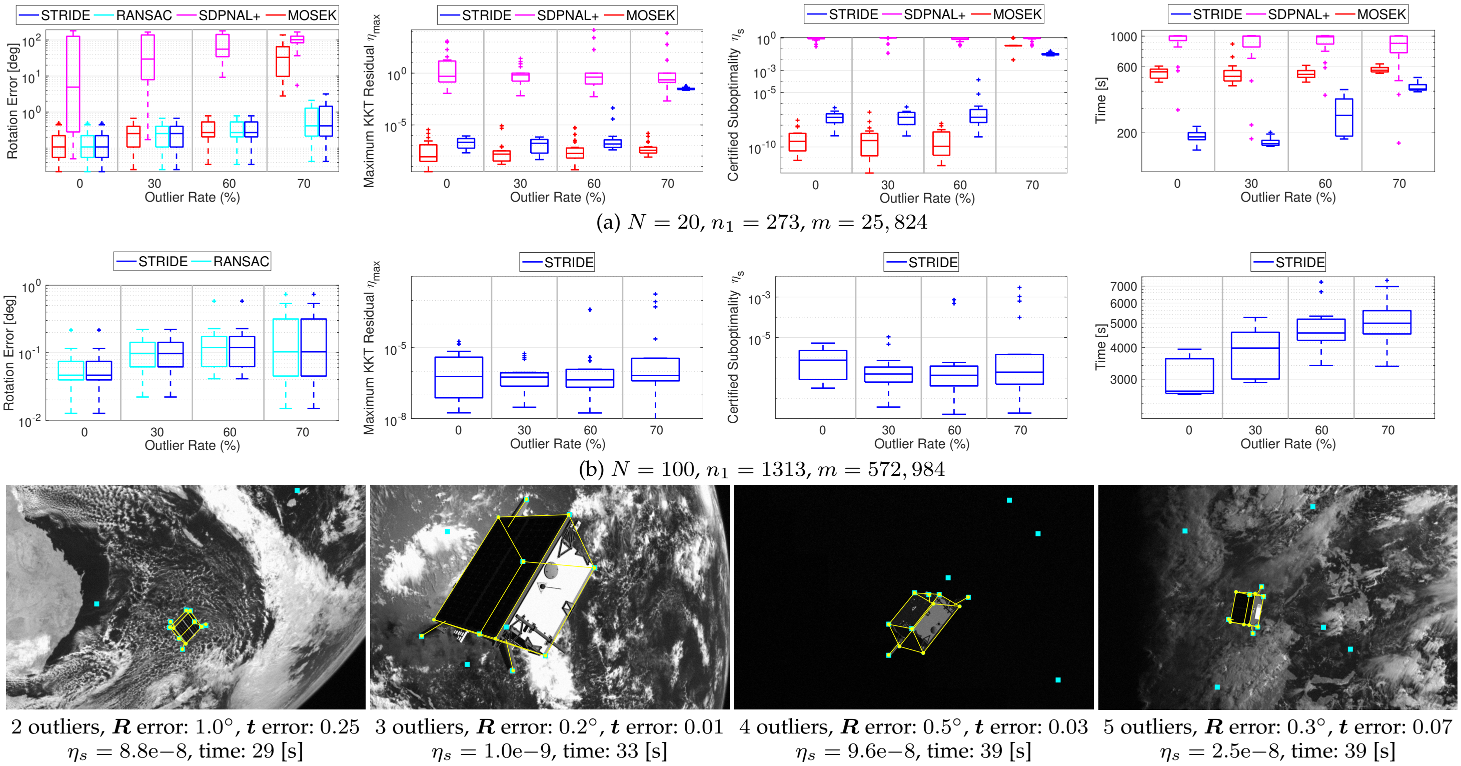
Setup. We use the *"car"* category from PASCAL3D+ dataset [92] for simulation experiments, which contains $N = 12$ keypoints with $K = 9$ basis shapes. We generate an unknown instance of the category by sampling a random vector of shape coefficients $\boldsymbol{c}^{\circ} \in \mathbb{R}^K_+$ such that $\sum_{k=1}^K c_i^{\circ} = 1$ and using $\boldsymbol{c}^{\circ}$ to linearly combine the K basis shapes. We then add random Gaussian noise (with standard deviation 0.01) to the new instance and transform it with a random rigid transformation $(\boldsymbol{R}^{\circ}, \boldsymbol{t}^{\circ})$ with $\|\boldsymbol{t}^{\circ}\| \leq T = 10$. We test increasing outlier rates up to 60% with 20 runs per outlier rate. We use a regularization parameter $\lambda = 1$.

Results. Fig. 9(a) plots the evaluation metrics: (i) our relaxation is exact with up to 60% outliers; (ii) STRIDE can can certify the global optimality of GNC and escapes its local minima; (iii) STRIDE is about 10 times faster than MOSEK.

Vehicle pose and shape estimation on ApolloScape. We use STRIDE to jointly estimate the pose and shape of an unknown vehicle from the ApolloScape self-driving dataset [89]. We use a set of $K = 5$ basis shapes, each with $N = 66$ annotated 3D keypoints. Given a 2D image depicting an unknown vehicle, we use the pretrained GSNet [54] to detect 2D keypoints of the unknown vehicle with groundtruth depth (same setup as one of the tests in [77]). Fig. 9(b-1) shows four examples where STRIDE certified the global optimality of solutions returned by GNC ($\eta_s = 1.5\text{e}{-7}, 1.3\text{e}{-9}, 1.4\text{e}{-10}, 1.6\text{e}{-9}$) , and Fig. 9(b-2) shows two example where STRIDE escapes the suboptimal solutions returned by GNC and finds the certified globally optimal solutions ($\eta_s = 3.2\text{e}{-4}, 4.6\text{e}{-4}$). More examples are provided in Supplementary Material.

6.7 Summary and Discussion

Table 2 summarizes the timing results of STRIDE, compared with MOSEK, for all six problems. We make a few comments. (i) STRIDE is able to solve problems far beyond the reach of MOSEK (in fact, the SDPs solved in this paper are among the largest in all semidefinite programming literature). (ii) When fast heuristics converge to the globally optimal solution, STRIDE just needs to perform optimality certification and can be 2-5 times faster (*cf.* STRIDE (Certify) vs. STRIDE (Escape)). (iii) For problems of similar sizes (in terms of n_1 and m), the speed of STRIDE can be *application-dependent* (*e.g.*, STRIDE is much faster in single rotation averaging than in other applications). This suggests that relaxations of different applications lead to SDP problems of *drastically different geometry*. Understanding the geometry and leveraging new tools to further speedup the computation is an exciting research venue. For example, it could be promising to use data-driven methods to "learn" the geometry of different problems to generate high-quality initializations.



(a) $N = 20$, $n_1 = 273$, $m = 25,824$

(b) $N = 100$, $n_1 = 1313$, $m = 572,984$

2 outliers, $\boldsymbol{R}$ error: 1.0°, $\boldsymbol{t}$ error: 0.25
$\eta_s = 8.8\text{e}{-8}$, time: 29 [s]

3 outliers, $\boldsymbol{R}$ error: 0.2°, $\boldsymbol{t}$ error: 0.01
$\eta_s = 1.0\text{e}{-9}$, time: 33 [s]

4 outliers, $\boldsymbol{R}$ error: 0.5°, $\boldsymbol{t}$ error: 0.03
$\eta_s = 9.6\text{e}{-8}$, time: 39 [s]

5 outliers, $\boldsymbol{R}$ error: 0.3°, $\boldsymbol{t}$ error: 0.07
$\eta_s = 2.5\text{e}{-8}$, time: 39 [s]

(c) Satellite pose estimation on SPEED [76]. Cyan square: detected 2D keypoints; Yellow circle: reprojected 3D keypoints.

Fig. 8. Absolute Pose Estimation (Example 5).

TABLE 2

Average timing of STRIDE and MOSEK (in seconds). We omit SDPNAL+ because it failed to solve most of the problems. "$**$" indicates MOSEK out of memory. "N/A" indicates GNC or RANSAC succeeded for all instances and there was no data for STRIDE escaping local minima.

	Single Rotation Averaging		Multiple Rotation Averaging		Point Cloud Registration		Mesh Registration		Absolute Pose Estimation		Category-level Perception
	$N = 30$	$N = 100$	$10 \times 10, N = 10$	$20 \times 20, N = 20$	$N = 20$	$N = 100$	$N = 20$	$N = 100$	$N = 20$	$N = 100$	$N = 12, K = 9$
(SSR) size (n_1)	310	1010	343 – 549	1483 – 1761	273	1313	273	1313	273	1313	286
(SSR) size (m)	$30,016$	$310,016$	$3452 – 35,246$	$64,384 – 189,198$	$21,897$	$485,417$	$21,897$	$485,417$	$25,824$	$572,984$	$23,241$
MOSEK	573	$**$	174	$**$	377	$**$	372	$**$	547	$**$	454
STRIDE (Certify)	11	99	17	564	78	1675	66	1381	211	4168	88
STRIDE (Escape)	15	506	N/A	N/A	127	4274	163	3136	N/A	N/A	196

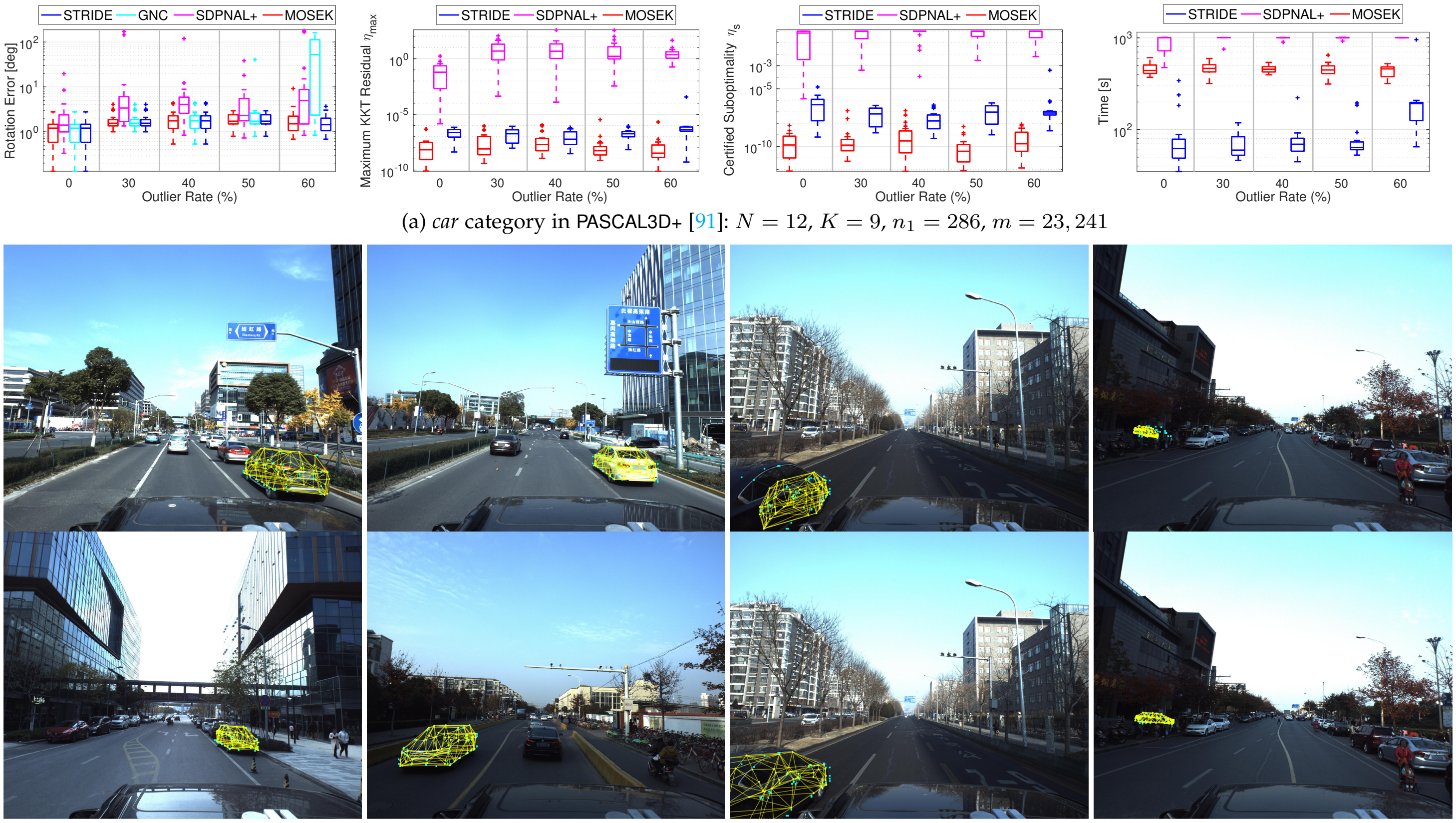
7 Related Work

We review related works on outlier-free and outlier-robust geometric perception, while we refer the interested reader to [87], [104] for recent progress in semidefinite programming and semidefinite relaxations.

Outlier-free geometric perception algorithms can be divided into *minimal solvers* and *non-minimal solvers*. Minimal solvers assume *noiseless* measurements (*i.e.*, $r(\boldsymbol{x}, \boldsymbol{z}_i) = 0, \forall\, i$ in (Robust)) and use the minimum number of measurements necessary to estimate $\boldsymbol{x}$, which typically leads to solving a system of polynomial equations [57]. Non-minimal solvers account for measurement noise and estimate $\boldsymbol{x}$ via nonlinear least squares (NLS), *i.e.*, $\rho(r) = r^2/\beta_i^2$ in (Robust). While in rare cases NLS can be solved in closed form [47] or by solving the polynomial equations arising from the first-order optimality conditions [56], in general they lead to nonconvex problems and are attacked using local solvers [75] or exponential-time methods (*e.g.*, *Branch and Bound* [70]).

Certifiable algorithms for outlier-free perception have recently emerged as an approach to compute globally optimal NLS solutions in polynomial time. These algorithms relax the NLS minimization into a convex optimization, using Lasserre's hierarchy of semidefinite relaxations for *polynomial optimizations* [52], [61]. By solving the SDP resulting from the convex relaxations, certifiable algorithms compute global solutions to NLS problems and provide a certificate of optimality, which usually depends on the rank of the SDP solution or the duality gap. Empirically tight convex relaxations have been discovered in pose graph optimization [27], [73], rotation averaging [39], [42], triangulation [4], 3D registration [22], [29], [65], absolute pose estimation [3], relative pose estimation [23], [107], hand-eye calibration [46] and shape and pose estimation from 2D or 3D landmarks [77], [96]. More recently, theoretical analysis of when and why the relaxations are tight is also emerging [4], [28], [29], [34], [38], [39], [40], [49], [73], [107]. Tight relaxations also enable optimality certification (*i.e.*, checking if a given solution is optimal), which —in outlier-free perception— can sometimes be performed in closed form [20], [24], [27], [39], [43], [49], [72]. Despite being certifiably optimal, these solvers assume all measurements are inliers (*i.e.*, have small noise), which rarely occurs in practice, and hence give poor estimates even in the presence of a single outlier.

Outlier-robust geometric perception algorithms can be divided into *fast heuristics* and *globally optimal solvers*. Two general frameworks for designing fast heuristics are RANSAC [41] and *graduated non-convexity* (GNC) [7], [16], [93]. RANSAC robustifies minimal solvers and acts as a fast heuristics to solve *consensus maximization* [31], while GNC robustifies non-minimal solvers and acts as a fast heuristics to solve *M-estimation* (*i.e.*, using a robust cost function ρ



(a) *car* category in PASCAL3D+ [91]: $N = 12$, $K = 9$, $n_1 = 286$, $m = 23,241$

(b-1) Vehicle pose and shape estimation on ApolloScape [89]. Four examples where globally optimal estimates returned by GNC are *certified* by STRIDE.

(b-2) Two examples where GNC converges to suboptimal estimates (top row), but STRIDE escapes the local minima and obtains *certifiably optimal* estimates.

Fig. 9. Category-level Object Perception (Example 6).

in (Robust)). Local optimization is also a popular fast heuristics [2], [75] for the case where an initial guess is available. Approximate but deterministic algorithms have also been designed to solve consensus maximization [62]. On the other hand, globally optimal solvers are typically designed using Branch and Bound [13], [50], [71], [103]. Graph-theoretic approaches are popular for outlier pruning, in order to boost the robustness of estimation algorithms [63], [71], [78], [94].

Certifiable outlier-robust algorithms relax problem (Robust) with a robust cost into a tight convex optimization. While certain robust costs, such as L1 [88] and Huber [25], are already convex, they have low breakdown points (*i.e.*, they can be compromised by a single outlier [66]). Problem-specific certifiably robust algorithms have been proposed to deal with high-breakdown-point formulations, such as the TLS cost [58], [94], [95].

8 Conclusions

We presented the first general and scalable framework to design certifiable algorithms for outlier-robust geometric perception. We first showed that estimation with several common robust cost functions can be reformulated as polynomial optimization problems. We then designed a semidefinite relaxation scheme that exploits the sparsity of outlier-robust estimation to generate SDPs of much smaller sizes while maintaining empirical exactness. Finally, we proposed a robust and scalable SDP solver, STRIDE, that can solve the sparse relaxations to unprecedented scale and accuracy. We tested our framework on six geometric perception applications using both synthetic and real data, demonstrating its robustness, scalability, and capability to safeguard existing fast heuristics for robust estimation.

Acknowledgments

The authors would like to thank Jie Wang, Victor Magron, and Jean B. Lasserre for the discussion about Lasserre's hierarchy and TSSOS; Ling Liang and Kim-Chuan Toh for the discussion about SDP solvers; Bo Chen and Tat-Jun Chin for the SPEED data; and Jingnan Shi for the ApolloScape data.

This work was funded by ARL DCIST CRA W911NF-17-2-0181, ONR RAIDER N00014-18-1-2828, MathWorks, NSF CAREER award "Certifiable Perception for Autonomous Cyber-Physical Systems", and Lincoln Laboratory's Resilient Perception in Degraded Environments program.

Supplementary Material

A1 Proof of Proposition 4

Proof. We first prove (i)-(iv) using Black-Rangarajan duality [16], and then (v)-(vii) by manipulating the cost functions.

Proof of (i)-(iv). The TLS proof has been given in (TLS) of the main text. We start with (ii) MC. With a similar strategy of introducing a binary variable as in (TLS), we can write the MC cost function as

$$\rho_{\mathrm{MC}} \equiv \min_{\theta \in \{+1,-1\}} \left\{ \frac{1-\theta}{2} \;\middle|\; -\theta(r^2 - \beta_i^2) \geq 0 \right\}, \tag{A1}$$

where the constraint $-\theta(r^2 - \beta_i^2) \geq 0$ enforces $\theta = -1$ if $r^2 > \beta_i^2$ (hence $\rho_{\mathrm{MC}} = 1$), and $\theta = +1$ if $r^2 < \beta_i^2$ (hence $\rho_{\mathrm{MC}} = 0$). If $r^2 = \beta_i^2$, then the minimization selects $\theta = +1$ as it minimizes the objective to be zero. Using the identity in (A1), problem (Robust) with $\rho = \rho_{\mathrm{MC}}$ is equivalent to

$$\min_{\substack{\boldsymbol{x} \in \mathcal{X} \subseteq \mathbb{R}^d, \\ \boldsymbol{\theta} \in \{\pm 1\}^N}} \left\{ \sum_{i=1}^{N} \frac{1-\theta_i}{2} + \psi(\boldsymbol{x}) \;\middle|\; \begin{array}{c} -\theta_i(r^2(\boldsymbol{x}, \boldsymbol{z}_i) - \beta_i^2) \geq 0, \\ \forall i = 1, \dots, N \end{array} \right\}, \tag{MC}$$

which is an instance of (POP) in $(\boldsymbol{x}, \boldsymbol{\theta}) \in \mathbb{R}^{d+N}$.

To prove (iii), we leverage Black-Rangarajan duality [16, Fig. 28] and write ρ_{GM} as a minimization problem by introducing a confidence variable $w \in [0,1]$

$$\rho_{\mathrm{GM}} \equiv \min_{w \in [0,1]} w \frac{r^2}{\beta_i^2} + (\sqrt{w} - 1)^2. \tag{A2}$$

One can check the correctness of (A2) by setting the derivative of the objective in (A2) w.r.t. w to zero and obtain w as a function of r in closed form. Eq. (A2), however, does not directly lead to a POP due to the existence of $\sqrt{w}$. Nevertheless, with a change of variable $\theta := \sqrt{w} \in [0,1]$, we can write problem (Robust) with $\rho = \rho_{\mathrm{GM}}$ as the following POP

$$\min_{\substack{\boldsymbol{x} \in \mathcal{X} \subseteq \mathbb{R}^d, \\ \boldsymbol{\theta} \in [0,1]^N}} \sum_{i=1}^{N} \frac{\theta_i^2 r^2(\boldsymbol{x}, \boldsymbol{z}_i)}{\beta_i^2} + (\theta_i - 1)^2 + \psi(\boldsymbol{x}). \tag{GM}$$

Similarly, we can use Black-Rangarajan duality [16, Fig. 25] to prove (iv) by introducing a confidence variable w and writing ρ_{TB} as the solution of the following minimization

$$\rho_{\mathrm{TB}} \equiv \min_{w \in [0,1]} w \frac{r^2}{\beta_i^2} + \frac{1}{3} - w + \frac{2}{3} w^{\frac{3}{2}}. \tag{A3}$$

Then, with a change of variable $\theta := \sqrt{w}$, we conclude that (Robust) with $\rho = \rho_{\mathrm{TB}}$ can be written as the following POP

$$\min_{\substack{\boldsymbol{x} \in \mathcal{X} \subseteq \mathbb{R}^d, \\ \boldsymbol{\theta} \in [0,1]^N}} \sum_{i=1}^{N} \frac{\theta_i^2 r^2(\boldsymbol{x}, \boldsymbol{z}_i)}{\beta_i^2} + \frac{1}{3} - \theta_i^2 + \frac{2}{3}\theta_i^3 + \psi(\boldsymbol{x}). \tag{TB}$$

In (TLS) and (MC), θ_i is binary and discrete, with $\theta_i = +1$ (resp. $\theta_i = -1$) indicating the i-the measurement $\boldsymbol{z}_i$ is an inlier (resp. outlier). While in (GM) and (TB), θ_i is continuous, with $\theta_i \uparrow 1$ (resp. $\theta_i \downarrow 0$) indicating the i-the measurement $\boldsymbol{z}_i$ is an inlier (resp. outlier).

Proof of (v)-(vii). The L1 cost function can be simply rewritten as

$$\rho_{\mathrm{L1}} \equiv \left\{ \frac{\gamma}{\beta_i} \;\middle|\; \gamma \geq 0, \gamma^2 = r^2 \right\}, \tag{A4}$$

where $\gamma \geq 0$ and $\gamma^2 = r^2$ implies that $\gamma = |r|$. Therefore, (Robust) with $\rho = \rho_{\mathrm{L1}}$ is equivalent to the following POP:

$$\min_{\substack{\boldsymbol{x} \in \mathcal{X} \subseteq \mathbb{R}^d, \\ \boldsymbol{\gamma} \in \mathbb{R}^N}} \left\{ \sum_{i=1}^{N} \frac{\gamma_i}{\beta_i} \;\middle|\; \gamma_i \geq 0, \gamma_i^2 = r^2(\boldsymbol{x}, \boldsymbol{z}_i), i = 1, \dots, N \right\}. \tag{L1}$$

Now we prove (Robust) with the Huber loss [48] can also be written as a POP. We first perform a change of variable and let $\gamma = |r|$ (which is equal to $\gamma \geq 0, \gamma^2 = r^2$):

$$\rho_{\mathrm{HB}} = \begin{cases} \frac{\gamma^2}{2\beta_i^2} & \gamma \leq \beta_i \\ \frac{\gamma}{\beta_i} - \frac{1}{2} & \text{otherwise} \end{cases}. \tag{A5}$$

Then we introduce a binary variable $\theta \in \{+1, -1\}$, and equivalently write (A5) as

$$\rho_{\mathrm{HB}} = \left\{ \frac{1+\theta}{2} \frac{\gamma^2}{2\beta_i^2} + \frac{1-\theta}{2} \left(\frac{\gamma}{\beta_i} - \frac{1}{2} \right) \middle| \theta(\gamma - \beta_i) \leq 0 \right\}, \tag{A6}$$

where the constraint $\theta(\gamma - \beta_i) \leq 0$ enforces $\theta = -1$ when $\gamma > \beta_i$ (hence $\rho_{\mathrm{HB}} = \frac{\gamma}{\beta_i} - \frac{1}{2}$), $\theta = +1$ when $\gamma < \beta_i$ (hence $\rho_{\mathrm{HB}} = \frac{\gamma^2}{2\beta_i^2}$). Therefore, we can write (Robust) with $\rho = \rho_{\mathrm{HB}}$ as the following POP:

$$\min_{\substack{\boldsymbol{x} \in \mathcal{X}, \boldsymbol{\gamma} \in \mathbb{R}^N, \\ \boldsymbol{\theta} \in \{\pm 1\}^N}} \left\{ \sum_{i=1}^{N} \frac{1+\theta_i}{2} \frac{\gamma_i^2}{2\beta_i^2} + \frac{1-\theta_i}{2} \left(\frac{\gamma_i}{\beta_i} - \frac{1}{2} \right) \middle| \begin{array}{c} \gamma_i \geq 0, \\ \gamma_i^2 = r^2(\boldsymbol{x}, \boldsymbol{z}_i^2), \\ \theta_i(\gamma_i - \beta_i) \leq 0, \\ i = 1, \dots, N \end{array} \right\}. \tag{HB}$$

Finally, we prove (Robust) with the adaptive cost function $\rho_{\mathrm{ADT},s}$, proposed by Barron [12], can also be written as a POP, when we restrict the scale parameter s to be rational numbers (we avoid $s = 0$ and $s = 2$ because the cost function is not defined at those two parameters, and [12] augments the cost by taking its limits at $s = 0$ and $s = 2$). Note that restricting s to rational numbers preserves the expressiveness of the original adaptive cost in [12], because the set of rational numbers is *dense* in the set of real numbers. Because s is a rational number, we can let $s = \frac{p}{q}$ with p, q as integers, and write the adaptive cost as

$$\rho_{\mathrm{ADT},s} = \frac{|s-2|}{s} \left(\left(\frac{r^2/\beta_i^2}{|s-2|} + 1 \right)^{\frac{p}{2q}} - 1 \right). \tag{A7}$$

Now we perform a change of variable and let $\gamma = \left(\frac{r^2/\beta_i^2}{|s-2|} + 1 \right)^{\frac{p}{2q}}$. This change of variable is equivalent to the following polynomial equality constraint:

$$0 = h(\gamma, r^2) := \begin{cases} \gamma^{2q} - \left(\frac{r^2/\beta_i^2}{|s-2|} + 1 \right)^{p} & p > 0 \\ \gamma^{2q} \left(\frac{r^2/\beta_i^2}{|s-2|} + 1 \right)^{|p|} - 1 & p < 0 \end{cases}. \tag{A8}$$

Therefore, we conclude that (Robust) with $\rho = \rho_{\mathrm{ADT},s}$ can be written as the following POP:

$$\min_{\substack{\boldsymbol{x} \in \mathcal{X}, \\ \boldsymbol{\gamma} \in \mathbb{R}^N}} \left\{ \sum_{i=1}^{N} \frac{|s-2|}{s} (\gamma_i - 1) \;\middle|\; h(\gamma_i, r^2(\boldsymbol{x}, \boldsymbol{z}_i)) = 0, i = 1, \dots, N \right\}. \tag{ADT}$$

This concludes the proof for all seven cost functions. ■

A2 Proof of Proposition 7

Proof. (i) can be proved by inspection: all the r^2 and ψ in Examples 1-6 are squared norms of *affine* (degree-one) polynomials in $\boldsymbol{x}$, and are naturally quadratic.

To show (ii), we note that the q-dimensional ball $\mathcal{B}_T^q$ can be described by a single quadratic inequality $\mathcal{B}_T^q = \{\boldsymbol{t} \in \mathbb{R}^q \mid T^2 - \langle \boldsymbol{t}, \boldsymbol{t} \rangle \geq 0\}$, the 3D FOV cone $\mathcal{C}_\alpha$ can be described by two inequalities $\mathcal{C}_\alpha = \{\boldsymbol{t} \in \mathbb{R}^3 \mid \tan^2(\frac{\alpha}{2}) t_3^2 - t_1^2 - t_2^2 \geq 0, t_3 \geq 0\}$, where the first inequality is quadratic and the second is affine. Now it remains to show that 2D and 3D rotations can be described by polynomial equalities. First, any 2D rotation $\boldsymbol{R} \in \mathrm{SO}(2)$ can be equivalently parametrized by

$$\boldsymbol{R} = \left\{ \left[\begin{array}{cc} r_1 & -r_2 \\ r_2 & r_1 \end{array} \right] \middle| \boldsymbol{r} \in \mathbb{R}^2, \langle \boldsymbol{r}, \boldsymbol{r} \rangle = 1 \right\}, \tag{A9}$$

and hence described by a single quadratic equality. For a 3D rotation $\boldsymbol{R} \in \mathrm{SO}(3)$, we shorthand $\boldsymbol{r}_i$ as its i-th column, and $\boldsymbol{r} = [\boldsymbol{r}_1\, ;\, \boldsymbol{r}_2\, ;\, \boldsymbol{r}_3] \in \mathbb{R}^9$ as its vectorization. Using the results from [22], [82], [96], we know that $\boldsymbol{R} \in \mathrm{SO}(3)$ can be equivalently described by the following set of 15 quadratic equality constraints

$$\text{Unit norm}: \quad h_i = 1 - \langle \boldsymbol{r}_i, \boldsymbol{r}_i \rangle, i = 1, 2, 3, \tag{A10a}$$

$$\text{Orthogonal}: \quad h_{i,j} = \langle \boldsymbol{r}_i, \boldsymbol{r}_j \rangle, \tbinom{i}{j} \in \{\tbinom{1}{2}, \tbinom{2}{3}, \tbinom{3}{1}\}, \tag{A10b}$$

$$\text{Right-hand}: h_{i,j,k} = \boldsymbol{r}_i \times \boldsymbol{r}_j - \boldsymbol{r}_k, \begin{pmatrix} i \\ j \\ k \end{pmatrix} \in \left\{ \begin{pmatrix} 1 \\ 2 \\ 3 \end{pmatrix}, \begin{pmatrix} 2 \\ 3 \\ 1 \end{pmatrix}, \begin{pmatrix} 3 \\ 1 \\ 2 \end{pmatrix} \right\}, \tag{A10c}$$

where "×" denotes the vector cross product, and each $h_{i,j,k}$ defines a vector of 3 equality constraints. Though the set of 15 equalities is redundant (*e.g.*, (A10a) and (A10c) are sufficient for $\boldsymbol{R} \in \mathrm{SO}(3)$), we use all of them to enhance robustness and tightness of the relaxation in Section 4. ■

A3 Proof of Theorem 9

Proof. (i): Every $\widetilde{\boldsymbol{x}} = (\boldsymbol{x}, \boldsymbol{\theta}) \in \mathcal{X} \times \{\pm 1\}^N$ of (TLS) leads to a rank-one lifting $\boldsymbol{v}(\widetilde{\boldsymbol{x}}) \boldsymbol{v}(\widetilde{\boldsymbol{x}})^\mathsf{T}$ that is feasible for (SSR). Therefore, the feasible set of (TLS) is a subset of the feasible set of (SSR), and hence $f^\star \leq p^\star$.

(ii) $p^\star = p(\widetilde{\boldsymbol{x}}^\star) = f(\boldsymbol{X}_v) = f^\star$, $\boldsymbol{X}_v$ is optimal for (SSR).

(iii) $f^\star \leq p^\star \leq \widehat{p}$ by construction. If $\widehat{p} = f^\star$, then equality holds: $f^\star = p^\star = \widehat{p}$, and $(\widehat{\boldsymbol{x}}, \widehat{\boldsymbol{\theta}})$ is optimal for (SSR). ■

A4 Relative Suboptimality

This section is concerned with the computation of a formally correct suboptimality gap η_s for a given estimate (which we use as a performance metric in our experiments), whose validity is not hindered by potential numerical inaccuracies in the solution of the SDP relaxation (SSR).

In Theorem 9 and (24), we stated that, by solving the sparse SDP relaxation (SSR) to global optimality with optimizer $\boldsymbol{X}^\star$ and associated optimum $f^\star$, one can round from $\boldsymbol{X}^\star$ a feasible solution $(\widehat{\boldsymbol{x}}_1, \widehat{\boldsymbol{\theta}}_1)$ to the original (TLS) problem with associated cost $\widehat{p} = p(\widehat{\boldsymbol{x}}_1, \widehat{\boldsymbol{\theta}}_1)$. Then, a measure of suboptimality for the rounded solution $(\widehat{\boldsymbol{x}}_1, \widehat{\boldsymbol{\theta}}_1)$ can be computed as follows (also in (24)):

$$\eta_s \triangleq \frac{|f^\star - \widehat{p}|}{1 + |f^\star| + |\widehat{p}|}. \tag{A11}$$

It is apparent that $\eta_s = 0$ implies the relaxation (SSR) is exact and the rounded solution $(\widehat{\boldsymbol{x}}_1, \widehat{\boldsymbol{\theta}}_1)$ is indeed globally optimal for (TLS) (recall $f^\star \leq p^\star \leq \widehat{p}$ by construction, where $p^\star$ is the unknown optimum of the nonconvex (TLS) problem).

However, the caveat in computing the relative suboptimality as in (A11) is that, although it is almost always possible to compute a rounded solution $(\widehat{\boldsymbol{x}}_1, \widehat{\boldsymbol{\theta}}_1)$ with cost $\widehat{p}$ (provided that the feasible set $\mathcal{X}$ of (TLS) is simple to project, as in our examples), it can be quite challenging to obtain $f^\star$ (which acts as a valid lower bound for $p^\star$) *to machine precision*, since $f^\star$ is computed by numerically solving the SDP (SSR), which may still lead to small inaccuracies. Moreover, as shown in the experimental section of the main text, first-order solvers such as SDPNAL+ typically cannot solve the SDP to even moderate accuracy (with reasonable amount of iterations), in which case $f^\star$ in not attained.

Here we describe a procedure to compute a valid lower bound for $p^\star$, from any approximate solution $(\widehat{\boldsymbol{X}}, \widehat{\boldsymbol{y}}, \widehat{\boldsymbol{S}}) \in \mathbb{X} \times \mathbb{R}^m \times \mathbb{X}$ of the SDP (SSR), without requiring it to be an optimal solution satisfying the KKT conditions (3). In fact, as we will show soon, only $\widehat{\boldsymbol{y}} \in \mathbb{R}^m$ is needed to compute a valid lower bound.

Bounded trace. Let us first shows that, each block of the primal variable $\boldsymbol{X}$ in (SSR) has a bounded trace, when (SSR) is applied to all six Examples 1-6. Towards this goal, let us first observe that the variable $\boldsymbol{x} \in \mathcal{X}$ has a bounded norm, *i.e.*, there exists $M_0 > 0$ such that $\|\boldsymbol{x}\| \leq M_0, \forall \boldsymbol{x} \in \mathcal{X}$. For example, in Example 1, $\boldsymbol{x} = \boldsymbol{R} \in \mathrm{SO}(3)$ has $\|\boldsymbol{x}\| = \sqrt{3} = M_0$; in Example 6, $\boldsymbol{x} = (\boldsymbol{R}, \boldsymbol{t}, \boldsymbol{c})$ with $\boldsymbol{R} \in \mathrm{SO}(3)$, $\boldsymbol{t} \in \mathcal{B}_{T_t}^3$, and $\boldsymbol{c} \in \mathcal{B}_{T_c}^K$ has $\|\boldsymbol{x}\| \leq \sqrt{3 + T_t^2 + T_c^2} = M_0$, where T_t and T_c are the upper bounds for the norm of the translation and the norm of the shape parameters, respectively.

Now recall that the primal variable $\boldsymbol{X}$ has $1 + l_g$ blocks, with the first block being the moment matrix and the other l_g blocks being localizing matrices. With the observation that $\|\boldsymbol{x}\| \leq M_0, \forall \boldsymbol{x} \in \mathcal{X}$, we can bound the trace of the moment matrix $\boldsymbol{X}_v$ (*cf.* (16)) as

$$\begin{aligned} \operatorname{tr}(\boldsymbol{X}_v) = & \quad \operatorname{tr}\left(\boldsymbol{v}(\widetilde{\boldsymbol{x}}) \boldsymbol{v}(\widetilde{\boldsymbol{x}})^\mathsf{T}\right) = \boldsymbol{v}(\widetilde{\boldsymbol{x}})^\mathsf{T} \boldsymbol{v}(\widetilde{\boldsymbol{x}}) \\ = & \quad 1 + \|\boldsymbol{x}\|^2 + \textstyle\sum_{i=1}^N \theta_i^2 + \sum_{i=1}^N \theta_i^2 \|\boldsymbol{x}\|^2 \\ = & \quad (1+N)(1 + \|\boldsymbol{x}\|^2) \\ \leq & \quad (1+N)(1 + M_0^2) =: M_1, \\ & \quad \forall \boldsymbol{x} \in \mathcal{X}, \boldsymbol{\theta} \in \{\pm 1\}^N. \end{aligned} \tag{A12}$$

Regarding the localizing matrices $\boldsymbol{X}_{g_j} = g_j \cdot [\boldsymbol{X}_1]_{\mathcal{I}_j}, j = 1, \dots, l_g$ (where $\boldsymbol{X}_1$ is the order-one moment matrix), we have that (recall $g_j \geq 0$ by definition)

$$\begin{aligned} \operatorname{tr}(\boldsymbol{X}_{g_j}) = & \quad g_j \cdot \operatorname{tr}\left([\boldsymbol{X}_1]_{\mathcal{I}_j}\right) \\ \leq & \quad g_j \cdot \operatorname{tr}(\boldsymbol{X}_1) \\ = & \quad g_j \cdot (1 + \|\boldsymbol{x}\|^2 + \textstyle\sum_{i=1}^N \theta_i^2) \\ \leq & \quad g_j \cdot (1 + N + M_0^2), \\ & \quad \forall \boldsymbol{x} \in \mathcal{X}, \boldsymbol{\theta} \in \{\pm 1\}^N. \end{aligned} \tag{A13}$$

Therefore, it suffices to show that g_j is upper bounded for any $\boldsymbol{x} \in \mathcal{X}$. This is obvious for all the examples in this paper. Particularly, there are only two types of inequality constraints among Examples 1-6. (i) The ball constraint $\boldsymbol{t} \in \mathcal{B}_T^K$ (bounded translation and bounded shape parameters), which reads $g = T^2 - \|\boldsymbol{t}\|^2 \geq 0$, which certainly satisfies $g \leq T^2$ and is upper bounded. (ii) The camera FOV cone constraint $\boldsymbol{t} \in \mathcal{C}_\alpha$ that induces two inequality constraints $g_1 = t_3 \geq 0$ and $g_2 = \tan^2(\alpha/2)t_3^2 - t_1^2 - t_2^2 \geq 0$. However, since the translation also lies in the bounded ball $\mathcal{B}_T^3$, we have $g_1 = t_3 \leq \|\boldsymbol{t}\| \leq T$, and $g_2 = \tan^2(\alpha/2)t_3^2 - t_1^2 - t_2^2 \leq \tan^2(\alpha/2)t_3^2 \leq \tan^2(\alpha/2)\left\|\boldsymbol{t}\right\|^2 \leq \tan^2(\alpha/2)T^2$ are both upper bounded. Therefore, we have shown that each localizing matrix also has bounded trace.

A valid lower bound. Now suppose we are given a $\widehat{\boldsymbol{y}} \in \mathbb{R}^m$, then for any $\boldsymbol{x} \in \mathcal{X}, \boldsymbol{\theta} \in \{\pm 1\}^N$, we have

$$\begin{aligned}
p(\boldsymbol{x}, \boldsymbol{\theta}) &= \langle \boldsymbol{C}, \boldsymbol{X} \rangle \\
&= \langle \boldsymbol{C} - \mathcal{A}^*(\widehat{\boldsymbol{y}}), \boldsymbol{X} \rangle + \langle \mathcal{A}^*(\widehat{\boldsymbol{y}}), \boldsymbol{X} \rangle \\
&= \langle \boldsymbol{C} - \mathcal{A}^*(\widehat{\boldsymbol{y}}), \boldsymbol{X} \rangle + \langle \mathcal{A}(\boldsymbol{X}), \widehat{\boldsymbol{y}} \rangle \\
&= \langle \boldsymbol{C} - \mathcal{A}^*(\widehat{\boldsymbol{y}}), \boldsymbol{X} \rangle + \langle \boldsymbol{b}, \widehat{\boldsymbol{y}} \rangle \qquad \text{(A14)} \\
&\geq \underbrace{\langle \boldsymbol{b}, \widehat{\boldsymbol{y}} \rangle + \sum_{i=1}^{1+l_g} M_i \cdot \min\{\lambda_{\min}\left([\boldsymbol{C} - \mathcal{A}^*(\widehat{\boldsymbol{y}})]_i\right), 0\}}_{\underline{p}(\widehat{\boldsymbol{y}})}, \text{(A15)}
\end{aligned}$$

where $M_i, i = 1, \dots, 1 + l_g$ are the upper bounds for the traces of the moment matrix and the localizing matrices (shown in previous paragraphs and (A12)-(A13)), $[\boldsymbol{C} - \mathcal{A}^*(\widehat{\boldsymbol{y}})]_i$ denotes the i-th block of $[\boldsymbol{C} - \mathcal{A}^*(\widehat{\boldsymbol{y}})]$ (recall that both $\boldsymbol{C}$ and $\mathcal{A}^*(\widehat{\boldsymbol{y}})$ are multi-block symmetric matrices, *cf.* (2)), and $\lambda_{\min}(\cdot)$ denotes the minimum eigenvalue of a symmetric matrix. In (A14), we used that any $\boldsymbol{X}$ that comes from moment matrix and localizing matrices must be primal feasible and hence $\mathcal{A}(\boldsymbol{X}) = \boldsymbol{b}$. In (A15), we used that $\langle \boldsymbol{A}, \boldsymbol{B} \rangle \geq \lambda_{\min}(\boldsymbol{A})\mathrm{tr}\left(\boldsymbol{B}\right)$ for any $\boldsymbol{A} \in \mathbb{S}^n$ and $\boldsymbol{B} \in \mathbb{S}^n_+$. With this lower bound $\underline{p}(\widehat{\boldsymbol{y}})$, we can compute the relative suboptimality from any $\widehat{\boldsymbol{y}} \in \mathbb{R}^m$:

$$\eta_s \triangleq \frac{\left|\underline{p}(\widehat{\boldsymbol{y}}) - \widehat{p}\right|}{1 + \left|\underline{p}(\widehat{\boldsymbol{y}})\right| + \left|\widehat{p}\right|}. \qquad \text{(A16)}$$

A5 Further Reduction on Multiple Rotation Averaging (Example 2)

Recall that in multiple rotation averaging we are given a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ with vertex set $\mathcal{V} = [n]$ and edge set $\mathcal{E}$. Each vertex $i \in \mathcal{V}$ is associated with an unknown rotation $\boldsymbol{R}_i \in \mathrm{SO}(q)$, and each edge $(i,j) \in \mathcal{E}$ provides a relative measurement $\widetilde{\boldsymbol{R}}_{ij}$ between the unknown rotations $\boldsymbol{R}_i$ and $\boldsymbol{R}_j$ at vertices i and j. Let $\mathcal{R}$ be the set of edges whose relative measurements are known to be free of outliers (*e.g.*, odometry measurements in SLAM), and let $\mathcal{Z} = \mathcal{E}/\mathcal{R}$ be the set of edges whose measurements are corrupted by outliers (*e.g.*, loop closures in SLAM). If no edge set is known to be free of outliers, then we set $\mathcal{R} = \emptyset$.

We now present a further reduction for multiple rotation averaging. Let us denote by $\mathcal{V}_{\mathcal{Z}} \triangleq \{i \in \mathcal{V} \mid \exists j \in \mathcal{V}, (i,j) \in \mathcal{Z}\} \subseteq \mathcal{V}$ the subset of nodes that are attached to at least one edge in $\mathcal{Z}$. Note that typically $|\mathcal{V}_{\mathcal{Z}}| \ll n$ for SLAM applications (*i.e.*, these are the nodes at which loop closures occur). For each edge $(i,j) \in \mathcal{Z}$, we define its *depth-ζ neighbor set* for $\zeta \in \mathbb{Z}_+$, in the following recursive manner:

$$\mathcal{V}^0_{(i,j)} \triangleq \{i, j\}, \mathcal{V}^\zeta_{(i,j)} \triangleq \{i \in \mathcal{V} \mid \exists j \in \mathcal{V}^{\zeta-1}_{(i,j)}, (i,j) \in \mathcal{E}\}, \text{(A17)}$$

where one can see that $\mathcal{V}^\zeta_{(i,j)}$ (for $\zeta \geq 1$) is essentially the union of the ζ-hop neighbor set of node i with the ζ-hop neighbor set of node j. It is easy to see that $\mathcal{V}^\zeta_{(i,j)} = \mathcal{V}, \forall (i,j) \in \mathcal{Z}$ when ζ is sufficiently large, as long as the graph $\mathcal{G}$ is connected. With $\mathcal{V}_{\mathcal{Z}}$ and $\mathcal{V}^\zeta_{(i,j)}$, for each edge $(i,j) \in \mathcal{Z}$, we define

$$\boldsymbol{x}^\zeta_{(i,j)} \triangleq \{\boldsymbol{R}_k \mid k \in \mathcal{V}^\zeta_{(i,j)} \cap \mathcal{V}_{\mathcal{Z}}\} \supseteq \{\boldsymbol{R}_i, \boldsymbol{R}_j\}, \qquad \text{(A18)}$$

as the set of node-wise rotations in $\mathcal{V}_{\mathcal{Z}}$ that are attached to (i,j) within depth ζ. By definition, $\mathcal{V}^\zeta_{(i,j)} \cap \mathcal{V}_{\mathcal{Z}}$ must contain node i and node j, and hence $\boldsymbol{x}^\zeta_{(i,j)}$ contains at least two rotations (attached to the edge (i,j)). We now replace the sparse basis in (15) as

$$\begin{aligned}
&\boldsymbol{v}(\widetilde{\boldsymbol{x}}) = [1\,;\, \boldsymbol{x}\,;\, \boldsymbol{\theta}\,;\, \dots \theta_{ij}\boldsymbol{x}^\zeta_{(i,j)} \dots]_{(i,j)\in\mathcal{Z}} \in \mathbb{R}^{n_1}, \\
&\quad 1 + 2n + 5N \leq n_1 \leq 1 + 2n + N(1 + 2\left|\mathcal{V}_{\mathcal{Z}}\right|), \quad \text{(A19)}
\end{aligned}$$

and use it to generate the semidefinite relaxation (SSR). It is worth noting that our relaxation recovers the hand-crafted SDP relaxation in [58] with the choice of $\zeta = 0$, which is shown to be *inexact* when the outlier rate is around 50%. In Section 6, we show that, with a larger ζ, we can achieve exact relaxation in the presence of over 70% outliers.

A6 Solving the Projection Subproblem

In this section, we describe how to solve the projection subproblem in STRIDE (*cf.* (PGM) and (28) in Algorithm 1). In particular, we show that the dual problem of (PGM) admits an unconstrained formulation, which allows developing a scalable algorithm based on limited-memory BFGS.

Recall that the projection step (28) of STRIDE seeks to compute the projection of a given point onto the spectrahedron $\mathcal{F}_{\mathrm{P}} = \{\boldsymbol{X} \in \mathbb{X} \mid \mathcal{A}(\boldsymbol{X}) = \boldsymbol{b}, \boldsymbol{X} \in \mathcal{K}\}$. Formally, given a point $\boldsymbol{Z} \in \mathbb{X}$, the projection problem seeks the closest point in $\mathcal{F}_{\mathrm{P}}$ w.r.t. $\boldsymbol{Z}$

$$\min_{\boldsymbol{X} \in \mathbb{X}} \left\{ \frac{1}{2}\left\|\boldsymbol{X} - \boldsymbol{Z}\right\|^2 \;\middle|\; \boldsymbol{X} \in \mathcal{F}_{\mathrm{P}} \right\}. \qquad \text{(A20)}$$

Since $\mathcal{F}_{\mathrm{P}}$ is the intersection of two convex sets, namely the hyperplane defined by $\mathcal{A}(\boldsymbol{X}) = \boldsymbol{b}$ and the (product of) positive semidefinite cone $\mathcal{K}$, a natural idea is to apply Dykstra's projection algorithm (see *e.g.*, [35]) to generate an approximate solution by alternating the projection onto the hyperplane and the projection onto the semidefinite cone, both of which are easy to compute. However, Dykstra's projection is known to have slow convergence and it may take too many iterations until a satisfactory projection is found. As a result, instead of solving (A20) directly, we consider its dual problem

$$\min_{\boldsymbol{y} \in \mathbb{R}^m, \boldsymbol{S} \in \mathbb{X}} \left\{ \frac{1}{2}\left\|\boldsymbol{S} + \mathcal{A}^*(\boldsymbol{y}) + \boldsymbol{Z}\right\|^2 - \langle \boldsymbol{b}, \boldsymbol{y} \rangle \;\middle|\; \boldsymbol{S} \in \mathcal{K} \right\}, \qquad \text{(A21)}$$

where we have ignored the constant term $-\frac{1}{2}\left\|\boldsymbol{Z}\right\|^2$ and converted "max" to "min" by changing the sign of the objective. The KKT conditions for the pair (A20) and (A21) are:

$$\mathcal{A}(\boldsymbol{X}) = \boldsymbol{b},\ \mathcal{A}^*(\boldsymbol{y}) + \boldsymbol{S} = \boldsymbol{X} - \boldsymbol{Z},\ \ \boldsymbol{X}, \boldsymbol{S} \in \mathcal{K},\ \langle \boldsymbol{X}, \boldsymbol{S} \rangle = 0. \qquad \text{(A22)}$$

An unconstrained formulation. Now we introduce a key observation that allows us to further simplify the dual (A21). Fixing the unconstrained $\boldsymbol{y}$, problem (A21) can be seen as finding the closest $\boldsymbol{S} \in \mathcal{K}$ w.r.t. the matrix $-\mathcal{A}^*(\boldsymbol{y}) - \boldsymbol{Z}$, and hence admits a closed-form solution

$$\boldsymbol{S} = \Pi_{\mathcal{K}} \left(-\mathcal{A}^*(\boldsymbol{y}) - \boldsymbol{Z} \right). \tag{A23}$$

As a result, after inserting (A23), problem (A21) is equivalent to

$$\min_{\boldsymbol{y} \in \mathbb{R}^m} \quad \phi(\boldsymbol{y}) := \frac{1}{2} \left\| \Pi_{\mathcal{K}}(\mathcal{A}^*(\boldsymbol{y}) + \boldsymbol{Z}) \right\|^2 - \langle \boldsymbol{b}, \boldsymbol{y} \rangle , \tag{A24}$$

with the gradient of $\phi(\boldsymbol{y})$ given as

$$\nabla \phi(\boldsymbol{y}) = \mathcal{A} \Pi_{\mathcal{K}}(\mathcal{A}^*(\boldsymbol{y}) + \boldsymbol{Z}) - \boldsymbol{b}. \tag{A25}$$

Thus, if $\boldsymbol{y}^\star$ is an optimal solution for problem (A24), we can recover $\boldsymbol{S}^\star$ from (A23), and $\boldsymbol{X}^\star$ from the KKT conditions (A22):

$$\boldsymbol{X}^\star = \mathcal{A}^*(\boldsymbol{y}^\star) + \boldsymbol{S}^\star + \boldsymbol{Z}. \tag{A26}$$

Formulating the dual problem as the unconstrained problem (A24) has appeared multiple times in [64], [108].

Now that (A24) is a *smooth unconstrained convex* problem in $\boldsymbol{y} \in \mathbb{R}^m$, plenty of efficient algorithms are available, such as (accelerated) gradient descend [67], nonlinear conjugate gradient [37], quasi-Newton methods [69] and the semismooth Newton method [108]. In this paper, we apply the celebrated limited-memory BFGS (L-BFGS) method, see for example [69, Algorithm 7.5]. L-BFGS is easy to implement, can handle very large unconstrained optimization problems due to its low memory consumption, and is typically "the algorithm of choice" for large-scale problems [69, Chapter 7]. Empirically, we observed that L-BFGS is efficient and robust for various applications. To the best of our knowledge, this is the first work that demonstrates the effectiveness of L-BFGS, or in general quasi-Newton methods, in solving large-scale and degenerate SDPs.

A7 Proof of Theorem 11

Proof. Let $\mathcal{V} = \{\widehat{\boldsymbol{X}}^{k(i)}\}$ be the sequence of all the $\widehat{\boldsymbol{X}}$ that have been accepted due to (29), where $k(i)$ returns the iteration index of the i-th element in $\mathcal{V}$. If $\mathcal{V} = \emptyset$, then STRIDE reduces to (PGM) and is globally convergent. If $\mathcal{V} \neq \emptyset$, then we claim that $\mathcal{V}$ must be finite. Note that, for any two consecutive elements $\widehat{\boldsymbol{X}}^{k(i)}$ and $\widehat{\boldsymbol{X}}^{k(i+1)}$ in $\mathcal{V}$, we have

$$\begin{aligned} f(\widehat{\boldsymbol{X}}^{k(i+1)}) \leq & \qquad\qquad f(\overline{\boldsymbol{X}}^{k(i+1)}) - \epsilon \\ < & \quad f(\boldsymbol{X}^{k(i+1)-1}) - \epsilon \leq f(\widehat{\boldsymbol{X}}^{k(i)}) - \epsilon, \end{aligned} \tag{A27}$$

where the first inequality is due to (29), the second inequality is due to (28) and the fact that projected gradient descent must strictly decrease the objective value when optimality has not been achieved [15, Proposition 3.4.1], and the last inequality holds because $k(i+1) - 1 \geq k(i)$. Eq. (A27) states that the objective value must decrease by at least ϵ along each element of $\mathcal{V}$. Therefore, we have $f_{\min}(\mathcal{V}) \leq f_{\max}(\mathcal{V}) - (|\mathcal{V}|-1)\epsilon$, where $f_{\min}$ and $f_{\max}$ are the minimum and maximum objective values along $\mathcal{V}$. Hence $|\mathcal{V}|$ must be finite, otherwise $f^\star$ is unbounded below, contradicting Slater's condition and strong duality. Let $\widehat{\boldsymbol{X}}^{k(|\mathcal{V}|)}$ be the last element of $\mathcal{V}$, then STRIDE reduces to (PGM) with a new initial point at $\widehat{\boldsymbol{X}}^{k(|\mathcal{V}|)}$ and is globally convergent. ■

A8 Implementation Details for STRIDE

In Section 5, we presented the STRIDE algorithm and proved its global convergence. We noted that the initial point $(\boldsymbol{X}^0, \boldsymbol{y}^0, \boldsymbol{S}^0)$ could have a significant impact on the convergence speed of STRIDE. Therefore, in STRIDE we use existing fast heuristics (GNC, RANSAC) to generate a *primal* initial guess (*cf.* Remark 12). In this section, we describe how to generate a *dual* initial guess (Section A8.1), and how to use Riemannian optimization for local search (Section A8.2).

A8.1 Dual Warmstart

We propose to use a combination of two techniques to generate a good dual initial point $(\boldsymbol{y}^0, \boldsymbol{S}^0)$. Section A8.1.1 describes a method to relax the (TLS) problem by exploiting correlative sparsity. Although such a relaxation is not tight, we show that its solution can be used to warmstart STRIDE. In Section A8.1.2, we present a fast first-order algorithm to refine both the primal and the dual initializations.

A8.1.1 Bootstrapping via Correlative Sparsity

The (TLS) problem has another special property called *correlative sparsity* [60], [85], [86], which, loosely speaking, refers to the property that there exists a partition of the variables $(\boldsymbol{x}, \boldsymbol{\theta})$ into a union of smaller groups, such that (i) each constraint of (TLS) involves only one group of the variables, and (ii) the objective of (TLS) can be decomposed into terms that each involves only one group of the variables (*cf.* [60, Assumption 2]). Particularly, we observe that the objective polynomial, denoted by $p(\boldsymbol{x}, \boldsymbol{\theta})$, can be expressed as a sum of N polynomials:

$$p(\boldsymbol{x}, \boldsymbol{\theta}) = \sum_{i=1}^{N} \underbrace{\left(\frac{1+\theta_i}{2} \frac{r^2(\boldsymbol{x}, \boldsymbol{z}_i)}{\beta_i^2} + \frac{1-\theta_i}{2} + \frac{1}{N}\psi(\boldsymbol{x}) \right)}_{p_i(\boldsymbol{x}, \theta_i)}, \tag{A28}$$

where each p_i is a polynomial that only involves $\widetilde{\boldsymbol{x}}_i \triangleq [\boldsymbol{x}\,;\, \theta_i] \in \mathbb{R}^{d+1}$. The constraint polynomials can also be partitioned into N groups where the i-th group of constraints only involves $\widetilde{\boldsymbol{x}}_i$. To see this, note that there are two types of constraints in (TLS), the ones that constrain $\boldsymbol{x}$ (to be proper rotations and translations), denoted by $\mathcal{H}[\boldsymbol{x}]$, and the ones that constrain each θ_i to be a binary variable, denoted by $\mathcal{H}[\theta_i] = \{\theta_i^2 - 1 = 0\}, i = 1, \dots, N$. Therefore, defining $\mathcal{H}_i \triangleq \{\mathcal{H}[\boldsymbol{x}], \mathcal{H}[\theta_i]\}$, then each $\mathcal{H}_i$ only contains polynomials in $\widetilde{\boldsymbol{x}}_i$, and the union of $\mathcal{H}_i$ for $i = 1, \dots, N$ is the full constraint set of (TLS). This correlative sparsity allows us to design an SDP relaxation for (TLS) using N moment matrices $\boldsymbol{X}_{v_i}, i = 1, \dots, N$, where each $\boldsymbol{X}_{v_i}$ is defined as

$$\boldsymbol{v}_i(\widetilde{\boldsymbol{x}}_i) \triangleq [1\,;\, \boldsymbol{x}\,;\, \theta_i\,;\, \theta_i \boldsymbol{x}] \in \mathbb{R}^{2d+2}, \tag{A29}$$

$$\boldsymbol{X}_{v_i} \triangleq \boldsymbol{v}_i(\widetilde{\boldsymbol{x}}_i)\boldsymbol{v}_i(\widetilde{\boldsymbol{x}}_i)^\mathsf{T} = \left[\begin{array}{cccc} 1 & \boldsymbol{x}^\mathsf{T} & \theta_i & \theta_i \boldsymbol{x}^\mathsf{T} \\ \boldsymbol{x} & \boldsymbol{x}\boldsymbol{x}^\mathsf{T} & \theta_i \boldsymbol{x} & \theta_i \boldsymbol{x}\boldsymbol{x}^\mathsf{T} \\ \theta_i & \theta_i \boldsymbol{x}^\mathsf{T} & \theta_i^2 & \theta_i^2 \boldsymbol{x}^\mathsf{T} \\ \theta_i \boldsymbol{x} & \theta_i \boldsymbol{x}\boldsymbol{x}^\mathsf{T} & \theta_i^2 \boldsymbol{x} & \theta_i^2 \boldsymbol{x}\boldsymbol{x}^\mathsf{T} \end{array} \right] \tag{A30}$$

and has a *constant* size $2d+2$. It is easy to verify that $\boldsymbol{X}_{v_i}$ contains all the monomials in $p_i(\widetilde{\boldsymbol{x}}_i)$ and $\mathcal{H}_i$. Therefore, by following similar steps as in the main text, we can derive an SDP relaxation that exploits correlative sparsity.

(i) Rewriting (TLS) *using the moment matrices* $\{\boldsymbol{X}_{v_i}\}_{i=1}^N$. Because the sparse moment matrix $\boldsymbol{X}_{v_i}$ contains all monomials in p_i, and the (TLS) cost is a sum of p_i's, we can write the objective of (TLS) as a linear functions of $\{\boldsymbol{X}_{v_i}\}_{i=1}^N$:

$$\text{objective}: \quad \textstyle\sum_{i=1}^N \langle \boldsymbol{C}_i, \boldsymbol{X}_{v_i} \rangle \,. \tag{A31}$$

(ii) Relaxing the rank-1 constraint on $\{\boldsymbol{X}_{v_i}\}_{i=1}^N$. By construction, $\boldsymbol{X}_{v_i}$ belongs to the set of rank-one positive semidefinite matrices. Since the rank constraint is non-convex, we drop it and only enforce each $\boldsymbol{X}_{v_i}$ to be positive semidefinite:

$$\text{moment matrices}: \quad \boldsymbol{X}_{v_i} \succeq 0, i = 1, \dots, N. \tag{A32}$$

(iii) Adding redundant constraints. Now we add moment constraints to each moment matrix $\boldsymbol{X}_{v_i}$ and use the set of constraints $\mathcal{H}_i$ to add redundant equality and localizing constraints for $\boldsymbol{X}_{v_i}$. Because this procedure is the same for each moment matrix $\boldsymbol{X}_{v_i}$, we will only describe it once for a fixed i. First, some monomials can repeat themselves at multiple entries of $\boldsymbol{X}_{v_i}$. For example, in (A30) the "$\theta_i \boldsymbol{x}$" block is the same as the "$\theta_i \boldsymbol{x}^{\mathsf{T}}$" block up to rearrangement of entries. In fact, the number of *unique* monomials in $\boldsymbol{X}_{v_i}$ is $m_{2v_i} = 3\mathfrak{t}(d+1)$, while the dimension of $\boldsymbol{X}_{v_i}$ (in terms of a symmetric matrix) is $\mathfrak{t}(2d+2)$. Therefore, we can add a total number of $m_{\text{mom}_i} = \mathfrak{t}(2d+2) - m_{2v_i} + 1$ *moment constraints*:

$$\begin{aligned}\text{moment constraints}: \quad & \langle \boldsymbol{A}_{\text{mom},j}, \boldsymbol{X}_{v_i} \rangle = 0, \\ & j = 1, \dots, m_{\text{mom}_i},\end{aligned} \tag{A33}$$

to enforce the repeating monomials in $\boldsymbol{X}_{v_i}$ to be equal to each other, as well as the leading entry $[\boldsymbol{X}_{v_i}]_{11} = 1$.

Second, we add redundant equality constraints. For each equality constraint h_k in $\mathcal{H}_i$, we denote $[\widetilde{\boldsymbol{x}}_i]_{h_k}$ as the maximum set of unique monomials such that $h_k \cdot [\widetilde{\boldsymbol{x}}_i]_{h_k}$ only contains monomials in $\boldsymbol{X}_{v_i}$. Formally,

$$[\widetilde{\boldsymbol{x}}_i]_{h_k} \triangleq \{\widetilde{\boldsymbol{x}}_i^{\boldsymbol{\alpha}} \mid \text{mono}\,(h_k \cdot \widetilde{\boldsymbol{x}}_i^{\boldsymbol{\alpha}}) \subseteq \text{mono}\,(\boldsymbol{X}_{v_i})\}. \tag{A34}$$

Consequently, we can write $h_k \cdot [\widetilde{\boldsymbol{x}}_i]_{h_k} = \boldsymbol{0}$ as linear equalities in $\boldsymbol{X}_{v_i}$:

$$\begin{aligned}\text{(redundant) equality constraints}: \; & \langle \boldsymbol{A}_{\text{req},kj}, \boldsymbol{X}_{v_i} \rangle = 0, \\ & k = 1, \dots, l_{h_i} \\ & j = 1, \dots, |[\widetilde{\boldsymbol{x}}_i]_{h_k}| \,,\end{aligned} \tag{A35}$$

where l_{h_i} is the number of equality constraints in $\mathcal{H}_i$.

Finally, for each inequality constraint g_j in $\mathcal{H}_i$ ($\deg\,(g_j) \le 2$ by Proposition 7), we denote by $[\boldsymbol{X}_1]_{\mathcal{I}_j}$ the maximum principal submatrix of $\boldsymbol{X}_1$ (*i.e.*, order-one full moment matrix) such that $g_j \cdot [\boldsymbol{X}_1]_{\mathcal{I}_j}$ only contains monomials in $\boldsymbol{X}_{v_i}$. Formally,

$$\begin{gathered}[\boldsymbol{X}_1]_{\mathcal{I}_j} \triangleq [\boldsymbol{X}_1]_{\mathcal{I}_j, \mathcal{I}_j}, \text{ with} \\ \mathcal{I}_j = \arg\max_{\mathcal{J}} \{|\mathcal{J}| \mid \text{mono}\,(g_j \cdot [\boldsymbol{X}_1]_{\mathcal{J},\mathcal{J}}) \subseteq \text{mono}\,(\boldsymbol{X}_{v_i})\}.\end{gathered} \tag{A36}$$

As a result, calling $\boldsymbol{X}_{g_j} = g_j \cdot [\boldsymbol{X}_1]_{\mathcal{I}_j}$, which is positive semidefinite by construction, we can write down the following localizing matrices and constraints:

$$\text{localizing matrices}: \quad \boldsymbol{X}_{g_j} \succeq 0, \quad j = 1, \dots, l_{g_i} \tag{A37}$$

$$\begin{aligned}\text{localizing constraints}: \; & \langle \boldsymbol{A}_{\text{loc},jkh}, \boldsymbol{X}_{v_i} \rangle = [\boldsymbol{X}_{g_j}]_{hk} \\ & j = 1, \dots, l_{g_i}, \\ & 1 \le h \le k \le |\mathcal{I}_j| \,,\end{aligned} \tag{A38}$$

where the linear constraints simply enforce each entry of $\boldsymbol{X}_{g_j}$ to be a linear combination of entries in $\boldsymbol{X}_{v_i}$, and l_{g_i} is the number of inequality constraints in $\mathcal{H}_i$.

(iv) Adding overlapping constraints. The extra step that needs to be performed when there are multiple moment matrices is to add constraints that enforce *overlapping entries* to be the same. Clearly, from (A30), one can see that the top left 2×2 blocks, *i.e.*, $[1\,;\, \boldsymbol{x}][1, \boldsymbol{x}^{\mathsf{T}}]$ is shared among $\boldsymbol{X}_{v_i}$ for all $i = 1, \dots, N$. Therefore, we add the following overlapping constraints

$$\begin{aligned}\text{overlapping constraints}: \; & [\boldsymbol{X}_{v_i}]_{\text{ovlp}} = [\boldsymbol{X}_{v_1}]_{\text{ovlp}}, \\ & i = 2, \dots, N,\end{aligned} \tag{A39}$$

where $[\boldsymbol{X}_{v_i}]_{\text{ovlp}}$ refers to the top-left 2×2 blocks of $\boldsymbol{X}_{v_i}$.

Steps (i)-(iv) above lead to the following SDP:

$$\begin{gathered}\min_{\boldsymbol{X}} \left\{ \sum_{i=1}^N \langle \boldsymbol{C}_i, \boldsymbol{X}_{v_i} \rangle \;\middle|\; \mathcal{A}(\boldsymbol{X}) = \boldsymbol{b}, \boldsymbol{X} \succeq 0 \right\} \\ \text{with } \boldsymbol{X} = \begin{pmatrix} \boldsymbol{X}_{v_1}, \boldsymbol{X}_{1,1}, \dots, \boldsymbol{X}_{1,l_{g_1}} \\ \boldsymbol{X}_{v_2}, \boldsymbol{X}_{2,1}, \dots, \boldsymbol{X}_{2,l_{g_2}} \\ \vdots \\ \boldsymbol{X}_{v_N}, \boldsymbol{X}_{N,1}, \dots, \boldsymbol{X}_{N,l_{g_N}} \end{pmatrix},\end{gathered} \tag{CSSR}$$

where we have shorthanded $\boldsymbol{X}_{i,j}$ as the j-th localizing matrix for the i-th moment matrix for notation convenience (*cf.* (A37)), and $\mathcal{A}(\boldsymbol{X}) = \boldsymbol{b}$ collects all the linear equality constraints from (A33), (A35), (A38), and (A39).

Comparing (CSSR) with (SSR), we see that, although (CSSR) has more positive semidefinite blocks than (SSR), the size of the blocks become much smaller, especially when N is large ((CSSR) has $n_1 = 2d + 2$, while (SSR) has $n_1 = (1+d)(1+N)$). Therefore, (CSSR) can be solved much more efficiently using off-the-shelf interior point methods such as MOSEK [8]. However, the caveat is that (CSSR) is not tight and cannot provide a certifiably optimal solution to the original (TLS) problem.

Assembling a dual initialization for STRIDE. Although the (CSSR) relaxation is inexact, it is still useful to solve it because we can use its solution to warmstart STRIDE. To do this, let us recall the block structure of (SSR) for the primal variable:

$$\boldsymbol{X} = (\boldsymbol{X}_v, \boldsymbol{X}_1, \dots, \boldsymbol{X}_{l_g}). \tag{A40}$$

The dual variable $\boldsymbol{S}$ has the same block structure:

$$\boldsymbol{S} = (\boldsymbol{S}_v, \boldsymbol{S}_1, \dots, \boldsymbol{S}_{l_g}), \tag{A41}$$

where each block of $\boldsymbol{S}$ has the same size as the corresponding block of $\boldsymbol{X}$. With a slight change of notation, let us rewrite the block structure of (CSSR) as:

$$\boldsymbol{X}_c = \begin{pmatrix} \boldsymbol{X}_{v_1}, \boldsymbol{X}_{1,1}, \dots, \boldsymbol{X}_{1,l_g} \\ \boldsymbol{X}_{v_2}, \boldsymbol{X}_{2,1}, \dots, \boldsymbol{X}_{2,l_g} \\ \vdots \\ \boldsymbol{X}_{v_N}, \boldsymbol{X}_{N,1}, \dots, \boldsymbol{X}_{N,l_g} \end{pmatrix}, \tag{A42}$$

$$\boldsymbol{S}_c = \begin{pmatrix} \boldsymbol{S}_{v_1}, \boldsymbol{S}_{1,1}, \dots, \boldsymbol{S}_{1,l_g} \\ \boldsymbol{S}_{v_2}, \boldsymbol{S}_{2,1}, \dots, \boldsymbol{S}_{2,l_g} \\ \vdots \\ \boldsymbol{S}_{v_N}, \boldsymbol{S}_{N,1}, \dots, \boldsymbol{S}_{N,l_g} \end{pmatrix}, \tag{A43}$$

where the subscript "c" indicates correlative, and we have used the fact that $l_{g_i} = l_g$ for all $i = 1, \dots, N$ because the only inequality constraints in (TLS) come from $\boldsymbol{x} \in \mathcal{X}$ and each $\mathcal{H}_i$ has an equal number of l_g inequality constraints. Our goal is to generate $\boldsymbol{S}$, given $\boldsymbol{S}_c$, for STRIDE. Note that the matrices $\boldsymbol{S}_v$ ($\boldsymbol{X}_v$) and $\boldsymbol{S}_{v_i}$ ($\boldsymbol{X}_{v_i}$) have different dimensions, so that it is inappropriate to just sum up all $\{\boldsymbol{S}_{v_i}\}_{i=1}^N$ to get $\boldsymbol{S}_v$. The correct way to "assemble" $\{\boldsymbol{S}_{v_i}\}_{i=1}^N$ is as follows. For each $\boldsymbol{S}_{v_i}$, we define $\overline{\boldsymbol{S}}_{v_i}$ so that it satisfies the following polynomial equality

$$\left\langle \overline{\boldsymbol{S}}_{v_i}, \boldsymbol{X}_v \right\rangle \equiv \left\langle \boldsymbol{S}_{v_i}, \boldsymbol{X}_{v_i} \right\rangle \tag{A44}$$

for any $\boldsymbol{X}_v$ and $\boldsymbol{X}_{v_i}$ that are *proper* moment matrices (note that both sides of (A44) are polynomials and the equality implies that the coefficients of both polynomials must be equal). This is essentially creating $\overline{\boldsymbol{S}}_{v_i}$ to be an all-zero matrix except that the principal submatrix of $\overline{\boldsymbol{S}}_{v_i}$ indexed by the monomials $\boldsymbol{v}_i(\widetilde{\boldsymbol{x}}_i)$ is equal to $\boldsymbol{S}_{v_i}$. Now that $\overline{\boldsymbol{S}}_{v_i}$ has the same size as $\boldsymbol{X}_v$ and $\boldsymbol{S}_v$, we can assemble $\boldsymbol{S}_v$ as

$$\boldsymbol{S}_v = \sum_{i=1}^N \overline{\boldsymbol{S}}_{v_i}, \tag{A45}$$

where the rationale for the sum can be partially understood from the complementarity condition of (3). By the same token, for each $\boldsymbol{S}_{i,j}$, we create $\overline{\boldsymbol{S}}_{i,j}$ such that

$$\left\langle \overline{\boldsymbol{S}}_{i,j}, \boldsymbol{X}_j \right\rangle \equiv \left\langle \boldsymbol{S}_{i,j}, \boldsymbol{X}_{i,j} \right\rangle, i = 1, \dots, N, j = 1, \dots, l_g, \tag{A46}$$

for any $\boldsymbol{X}_j$ and $\boldsymbol{X}_{i,j}$ that are proper localizing matrices. Then we assemble $\boldsymbol{S}_j$ as

$$\boldsymbol{S}_j = \sum_{i=1}^N \overline{\boldsymbol{S}}_{i,j}, \quad j = 1, \dots, l_g. \tag{A47}$$

The rationale for (A44) and (A46) can be understood from the complementarity condition of the KKT system (3), and more deeply from the dual perspective of sums-of-squares (SOS) polynomials [18] (precisely, we are assembling a SOS polynomial in $(\boldsymbol{x}, \boldsymbol{\theta})$ from N SOS polynomials, each only involves the variables $(\boldsymbol{x}, \theta_i)$). Since this is less relevant for the purpose of this paper (and it is only used for warmstart), we only state the assembling procedure as in (A45) and (A47) without diving too deep into the theory of sums of squares. The interested reader is encouraged to refer to the dual SOS perspective in [61].

A8.1.2 Semi-proximal ADMM

After obtaining $\boldsymbol{X}^0$ from primal heuristics such as GNC [93] or RANSAC [41], and $\boldsymbol{S}^0$ from solving (CSSR) and performing the assembly procedure in Section A8.1.1, we use the semi-proximal alternating direction method of multipliers (ADMM+) proposed in [79] to refine both the primal and the dual initializations $(\boldsymbol{X}^0, \boldsymbol{S}^0)$. The full ADMM+ algorithm, for solving a standard SDP (P)-(D), is presented in Algorithm A2. As we can see, at each iteration of ADMM+, the major computation involves solving a linear system (*cf.* (A48) and (A50)) and performing a projection onto the product of positive semidefinite cones $\mathcal{K}$ (*cf.* (A49b)). Since $\mathcal{A}$ is typically sparse in our examples, Cholesky factorization of $\mathcal{A}\mathcal{A}^*$ can be done efficiently and needs to be performed only once. ADMM+ is a globally convergent algorithm for solving the SDP (P)-(D) and the interested reader can refer to [79] for a detailed study. Notably, [79] shows that ADMM+ is typically 2 to 3 times faster than a conventional ADMM. In our implementation, we use the function `admmplus` in SDPNAL+ [104] to refine $(\boldsymbol{X}^0, \boldsymbol{S}^0)$ and warmstart STRIDE. Although one can directly pass $(\boldsymbol{X}^0, \boldsymbol{S}^0)$ to STRIDE, empirically we found it is beneficial to refine $(\boldsymbol{X}^0, \boldsymbol{S}^0)$ using ADMM+ because the refined initial points will have higher quality that promotes the convergence of STRIDE. In our experiments, we run ADMM+ for a maximum of $20,000$ iterations, or until $\max\{\eta_p, \eta_d\}$ is below a threshold (*e.g.*, $1\mathrm{e}{-6}$).

Algorithm A2: ADMM+

```
Given $\boldsymbol{X}^0 = \boldsymbol{S}^0 \in \mathbb{X}$, $\sigma > 0$, and $\gamma \in (0, 2)$. Iterate
  the following steps for $k = 0, 1, \dots$.
Compute
    $$\widehat{\boldsymbol{y}}^{k+1} = (\mathcal{A}\mathcal{A}^*)^{-1} \left( \frac{1}{\sigma}\boldsymbol{b} - \mathcal{A}\left( \frac{1}{\sigma}\boldsymbol{X}^k + \boldsymbol{S}^k - \boldsymbol{C} \right) \right). \tag{A48}$$
Compute
    $$\boldsymbol{M}^{k+1} = \boldsymbol{X}^k + \sigma \left( \mathcal{A}^*(\widehat{\boldsymbol{y}}^{k+1}) - \boldsymbol{C} \right), \tag{A49a}$$
    $$\boldsymbol{S}^{k+1} = \frac{1}{\sigma} \left( \Pi_{\mathcal{K}}(\boldsymbol{M}^{k+1}) - \boldsymbol{M}^{k+1} \right). \tag{A49b}$$
Compute
    $$\boldsymbol{y}^{k+1} = (\mathcal{A}\mathcal{A}^*)^{-1} \left( \frac{1}{\sigma}\boldsymbol{b} - \mathcal{A}\left( \frac{1}{\sigma}\boldsymbol{X}^k + \boldsymbol{S}^{k+1} - \boldsymbol{C} \right) \right). \tag{A50}$$
Compute
    $$\boldsymbol{X}^{k+1} = \boldsymbol{X}^k + \gamma\sigma \left( \boldsymbol{S}^{k+1} + \mathcal{A}^*(\boldsymbol{y}^{k+1}) - \boldsymbol{C} \right). \tag{A51}$$
```

A8.2 Local Search and Nonlinear Programming

Recall the local search step (26a) applies a nonlinear programming (NLP) algorithm to solve the (TLS) problem given an initial point. Since (TLS) is a polynomial optimization, it is straightforward to implement NLP using `fmincon` in Matlab. However, here we show that it is possible to exploit the smooth manifold structure of (TLS) and solve it more efficiently with Riemannian optimization [1] (*e.g.*, using Manopt [19]). First, we can model the vector of binary variables $\boldsymbol{\theta}$ as an *oblique manifold* of size $N \times 1$ (an oblique manifold contains matrices with unit-norm rows). Second, from Examples 1-6, we know the geometric model $\boldsymbol{x}$ contains 2D and 3D rotations, which are both smooth manifolds. However, $\boldsymbol{x}$ can also contain translation $\boldsymbol{t}$ and shape parameters $\boldsymbol{c}$ that do not live on smooth manifolds. Fortunately, we can drop some constraints so that they both live on smooth manifolds. For example, in Examples 3, 4, and 6, we can relax $\boldsymbol{t} \in \mathcal{B}_T^3$ to $\boldsymbol{t} \in \mathbb{R}^3$, with the rationale that when the SDP iterate $\boldsymbol{X}^k$ is close to optimal, $\|\boldsymbol{t}\| \leq T$ should be naturally satisfied (from rounding (25)) even without explicit constraint. Similarly, we relax $\boldsymbol{t} \in \mathcal{B}_T^3 \cap \mathcal{C}_\alpha$ in Example 5 to $\boldsymbol{t} \in \mathbb{R}^3$, and relax $\boldsymbol{c} \in \mathbb{R}_+^K \cap \mathcal{B}_T^K$ in Example 6 to $\boldsymbol{c} \in \mathbb{R}_{++}^K$ (matrices with strictly positive entries live on a smooth manifold). Note that these modifications will not affect the global convergence of STRIDE because (29) will reject the NLP solution if it violates the constraints that have been dropped.

A9 Extra Experimental Results

In this section, we report extra results on real datasets.

A9.1 Point Cloud Registration on 3DMatch

We provide 10 extra scan matching results by STRIDE on the 3DMatch dataset [105] in Fig. A1. STRIDE returned the globally optimal transformation estimates in all cases.

A9.2 Absolute Pose Estimation on SPEED

We provide extra satellite pose estimation results by STRIDE on the SPEED dataset [76] in Fig. A2. In all six image instances with 2-5 outliers, STRIDE returned accurate pose estimates with global optimality certificates.

A9.3 Vehicle Pose and Shape Estimation on ApolloScape

We provide vehicle pose and shape estimation results by STRIDE on the ApolloScape dataset [89] in Fig. A3, whose first row also includes the four examples presented in Fig. 9(c-1). We provide details of each problem instance such as N, n_1 and m, as well as evaluation metrics such as $(\boldsymbol{R}, \boldsymbol{t})$ errors, relative suboptimality η_s, and STRIDE's computation time. In all cases, STRIDE returned accurate pose and shape estimates with global optimality certificates.

References

[1] P.-A. Absil, R. Mahony, and R. Sepulchre. *Optimization Algorithms on Matrix Manifolds*. Princeton University Press, Princeton, NJ, USA, 2007. 22

[2] P. Agarwal, G. D. Tipaldi, L. Spinello, C. Stachniss, and W. Burgard. Robust map optimization using dynamic covariance scaling. In *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, 2013. 16

[3] S. Agostinho, J. Gomes, and A. Del Bue. CvxPnPL: A unified convex solution to the absolute pose estimation problem from point and line correspondences. *arXiv preprint arXiv:1907.10545*, 2019. 2, 15

[4] C. Aholt, S. Agarwal, and R. Thomas. A qcqp approach to triangulation. In *European Conference on Computer Vision*, pages 654–667. Springer, 2012. 2, 7, 15

[5] F. Alizadeh, J. Haeberly, and M. Overton. Complementarity and nondegeneracy in semidefinite programming. *Mathematical Programming*, 77:111–128, 1997. 4

[6] F. Alizadeh, J.-P. A. Haeberly, and M. L. Overton. Primal-dual interior-point methods for semidefinite programming: Convergence rates, stability and numerical results. *SIAM Journal on Optimization*, 8(3):746–768, 1998. 4

[7] P. Antonante, V. Tzoumas, H. Yang, and L. Carlone. Outlier-robust estimation: Hardness, minimally-tuned algorithms, and applications. *IEEE Trans. Robotics*, 2021. 1, 7, 15

[8] M. ApS. *The MOSEK optimization toolbox for MATLAB manual. Version 8.1.*, 2017. 2, 4, 11, 21

[9] M. A. Audette, F. P. Ferrie, and T. M. Peters. An algorithmic overview of surface registration techniques for medical imaging. *Med. Image Anal.*, 4(3):201–217, 2000. 1

[10] A. Bandeira. A note on probably certifiably correct algorithms. *Comptes Rendus Mathematique*, 354(3):329–333, 2016. 1

[11] T. Barfoot. *State Estimation for Robotics*. Cambridge University Press, 2017. 1

[12] J. T. Barron. A general and adaptive robust loss function. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4331–4339, 2019. 2, 6, 17

[13] J. C. Bazin, Y. Seo, and M. Pollefeys. Globally optimal consensus set maximization through rotation search. In *Asian Conference on Computer Vision*, pages 539–551. Springer, 2012. 16

[14] A. Beck and M. Teboulle. A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *SIAM journal on imaging sciences*, 2(1):183–202, 2009. 10

[15] D. Bertsekas. *Nonlinear Programming*. Athena Scientific, 1999. 10, 20

[16] M. J. Black and A. Rangarajan. On the unification of line processes, outlier rejection, and robust statistics with applications in early vision. *Intl. J. of Computer Vision*, 19(1):57–91, 1996. 1, 6, 11, 15, 17

[17] A. Blake and A. Zisserman. *Visual reconstruction*. MIT Press, 1987. 1

[18] G. Blekherman, P. A. Parrilo, and R. R. Thomas. *Semidefinite optimization and convex algebraic geometry*. SIAM, 2012. 3, 5, 10, 22

[19] N. Boumal, B. Mishra, P.-A. Absil, and R. Sepulchre. Manopt, a Matlab toolbox for optimization on manifolds. *Journal of Machine Learning Research*, 15(42):1455–1459, 2014. 22

[20] N. Boumal, V. Voroninski, and A. Bandeira. The non-convex Burer–Monteiro approach works on smooth semidefinite programs. In *Advances in Neural Information Processing Systems (NIPS)*, pages 2757–2765. 2016. 2, 4, 15

[21] S. Boyd and L. Vandenberghe. *Convex optimization*. Cambridge University Press, 2004. 3

[22] J. Briales and J. Gonzalez-Jimenez. Convex Global 3D Registration with Lagrangian Duality. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2017. 2, 6, 15, 18

[23] J. Briales, L. Kneip, and J. Gonzalez-Jimenez. A certifiably globally optimal solution to the non-minimal relative pose problem. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2018. 2, 7, 15

[24] Burer, Samuel and Monteiro, Renato D C. A nonlinear programming algorithm for solving semidefinite programs via low-rank factorization. *Mathematical Programming*, 95(2):329–357, 2003. 2, 4, 15

[25] L. Carlone and G. Calafiore. Convex relaxations for pose graph optimization with outliers. *IEEE Robotics and Automation Letters*, 3(2):1160–1167, 2018. 16

[26] L. Carlone and G. C. Calafiore. Convex relaxations for pose graph optimization with outliers. *IEEE Robotics and Automation Letters*, 3(2):1160–1167, 2018. 2

[27] L. Carlone, G. C. Calafiore, C. Tommolillo, and F. Dellaert. Planar pose graph optimization: Duality, optimal solutions, and verification. *IEEE Transactions on Robotics*, 32(3):545–565, 2016. 2, 6, 15

[28] L. Carlone and F. Dellaert. Duality-based verification techniques for 2D SLAM. In *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, pages 4589–4596, 2015. 7, 15

[29] K. N. Chaudhury, Y. Khoo, and A. Singer. Global registration of multiple point clouds using semidefinite programming. *SIAM Journal on Optimization*, 25(1):468–501, 2015. 2, 15

[30] B. Chen, J. Cao, A. Parra, and T.-J. Chin. Satellite pose estimation with deep landmark regression and nonlinear pose refinement. In *2019 IEEE/CVF International Conference on Computer Vision Workshop (ICCVW)*, pages 2816–2824. IEEE, 2019. 14

[31] T. J. Chin and D. Suter. The maximum consensus problem: recent algorithmic advances. *Synthesis Lectures on Computer Vision*, 7(2):1–194, 2017. 1, 15

[32] S. Choi, Q. Y. Zhou, and V. Koltun. Robust reconstruction of indoor scenes. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 5556–5565, 2015. 1

[33] D. Cifuentes. A convex relaxation to compute the nearest structured rank deficient matrix. *SIAM Journal on Matrix Analysis and Applications*, 42(2):708–729, 2021. 2

[34] D. Cifuentes, S. Agarwal, P. Parrilo, and R. Thomas. On the local stability of semidefinite relaxations. *ArXiv preprint: 1710.04287v1*, 2017. 15

[35] P. L. Combettes and J.-C. Pesquet. Proximal splitting methods in signal processing. In *Fixed-point algorithms for inverse problems in science and engineering*, pages 185–212. Springer, 2011. 19

[36] B. Curless and M. Levoy. A volumetric method for building complex models from range images. In *SIGGRAPH*, pages 303–312, 1996. 6

[37] Y.-H. Dai and Y. Yuan. A nonlinear conjugate gradient method with a strong global convergence property. *SIAM Journal on optimization*, 10(1):177–182, 1999. 20

[38] N. Dym and Y. Lipman. Exact recovery with symmetries for procrustes matching. *SIAM Journal on Optimization*, 27(3):1513–1530, 2017. 15

[39] A. Eriksson, C. Olsson, F. Kahl, and T.-J. Chin. Rotation averaging and strong duality. *IEEE Conf. on Computer Vision and Pattern*

Recognition (CVPR), 2018. 2, 6, 9, 15

[40] A. Eriksson, C. Olsson, F. Kahl, and T.-J. Chin. Rotation averaging with the chordal distance: Global minimizers and strong duality. *IEEE Trans. Pattern Anal. Machine Intell.*, 43(1):256–268, 2019. 15

[41] M. Fischler and R. Bolles. Random sample consensus: a paradigm for model fitting with application to image analysis and automated cartography. *Commun. ACM*, 24:381–395, 1981. 1, 11, 15, 22

[42] J. Fredriksson and C. Olsson. Simultaneous multiple rotation averaging using lagrangian duality. In *Asian Conf. on Computer Vision (ACCV)*, 2012. 2, 15

[43] M. Garcia-Salguero, J. Briales, and J. Gonzalez-Jimenez. Certifiable relative pose estimation. *Image and Vision Computing*, 109:104142, 2021. 2, 15

[44] M. Giamou, Z. Ma, V. Peretroukhin, and J. Kelly. Certifiably globally optimal extrinsic calibration from per-sensor egomotion. *IEEE Robotics and Automation Letters*, 4(2):367–374, 2019. 2

[45] R. Hartley, J. Trumpf, Y. Dai, and H. Li. Rotation averaging. *IJCV*, 103(3):267–305, 2013. 6

[46] J. Heller, D. Henrion, and T. Pajdla. Hand-eye and robot-world calibration by global polynomial optimization. In *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, pages 3157–3164. IEEE, 2014. 2, 15

[47] B. K. P. Horn. Closed-form solution of absolute orientation using unit quaternions. *J. Opt. Soc. Amer.*, 4(4):629–642, Apr 1987. 15

[48] P. Huber. *Robust Statistics*. John Wiley & Sons, New York, NY, 1981. 17

[49] J. P. Iglesias, C. Olsson, and F. Kahl. Global optimality for point set registration using semidefinite programming. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2020. 2, 15

[50] G. Izatt, H. Dai, and R. Tedrake. Globally optimal object pose estimation in point clouds with mixed-integer programming. In *Proc. of the Intl. Symp. of Robotics Research (ISRR)*, 2017. 1, 16

[51] K. Jiang, D. Sun, and K.-C. Toh. An inexact accelerated proximal gradient method for large scale linearly constrained convex sdp. *SIAM Journal on Optimization*, 22(3):1042–1064, 2012. 10

[52] F. Kahl and D. Henrion. Globally optimal estimates for geometric reconstruction problems. *Intl. J. of Computer Vision*, 74(1):3–15, 2007. 2, 15

[53] R. Kaskman, S. Zakharov, I. Shugurov, and S. Ilic. Homebreweddb: Rgb-d dataset for 6d pose estimation of 3d objects. In *Proceedings of the IEEE International Conference on Computer Vision Workshops*, pages 0–0, 2019. 2, 7, 13, 14

[54] 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 *European Conference on Computer Vision*, pages 515–532. Springer, 2020. 14

[55] G. Klein and D. Murray. Parallel tracking and mapping for small ar workspaces. In *2007 6th IEEE and ACM international symposium on mixed and augmented reality*, pages 225–234. IEEE, 2007. 1

[56] L. Kneip, H. Li, and Y. Seo. UPnP: An optimal o(n) solution to the absolute pose problem with universal applicability. In *European Conf. on Computer Vision (ECCV)*, pages 127–142. Springer, 2014. 7, 15

[57] Z. Kukelova, M. Bujnak, and T. Pajdla. Automatic generator of minimal problem solvers. In *European Conf. on Computer Vision (ECCV)*, pages 302–315. Springer, 2008. 15

[58] P.-Y. Lajoie, S. Hu, G. Beltrame, and L. Carlone. Modeling perceptual aliasing in slam via discrete–continuous graphical models. *IEEE Robotics and Automation Letters*, 4(2):1232–1239, 2019. 2, 6, 16, 19

[59] J. B. Lasserre. Global optimization with polynomials and the problem of moments. *SIAM J. Optim.*, 11(3):796–817, 2001. 2, 3, 5

[60] J. B. Lasserre. Convergent sdp-relaxations in polynomial optimization with sparsity. *SIAM Journal on Optimization*, 17(3):822–843, 2006. 20

[61] J.-B. Lasserre. *Moments, positive polynomials and their applications*, volume 1. World Scientific, 2010. 4, 5, 15, 22

[62] H. M. Le, T.-J. Chin, A. Eriksson, T.-T. Do, and D. Suter. Deterministic approximate methods for maximum consensus robust fitting. *IEEE Trans. Pattern Anal. Machine Intell.*, 2019. 16

[63] P. C. Lusk, K. Fathian, and J. P. How. CLIPPER: A Graph-Theoretic Framework for Robust Data Association. In *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, 2021. 16

[64] J. Malick, J. Povh, F. Rendl, and A. Wiegele. Regularization methods for semidefinite programming. *SIAM Journal on Optimization*, 20(1):336–356, 2009. 20

[65] H. Maron, N. Dym, I. Kezurer, S. Kovalsky, and Y. Lipman. Point registration via efficient convex relaxation. *ACM Transactions on Graphics (TOG)*, 35(4):1–12, 2016. 2, 15

[66] R. A. Maronna, R. D. Martin, V. J. Yohai, and M. Salibián-Barrera. *Robust statistics: theory and methods (with R)*. John Wiley & Sons, 2019. 5, 16

[67] Y. Nesterov. *Lectures on convex optimization*, volume 137. Springer, 2018. 20

[68] J. Nie. Optimality conditions and finite convergence of lasserre's hierarchy. *Mathematical programming*, 146(1-2):97–121, 2014. 5

[69] J. Nocedal and S. Wright. *Numerical optimization*. Springer Science & Business Media, 2006. 20

[70] C. Olsson, F. Kahl, and M. Oskarsson. Branch-and-bound methods for euclidean registration problems. *IEEE Trans. Pattern Anal. Machine Intell.*, 31(5):783–794, 2009. 15

[71] Á. Parra Bustos and T. J. Chin. Guaranteed outlier removal for point cloud registration with correspondences. *IEEE Trans. Pattern Anal. Machine Intell.*, 40(12):2868–2882, 2018. 16

[72] D. M. Rosen. Scalable low-rank semidefinite programming for certifiably correct machine perception. In *Intl. Workshop on the Algorithmic Foundations of Robotics (WAFR)*, 2020. 2, 4, 15

[73] D. M. Rosen, L. Carlone, A. S. Bandeira, and J. J. Leonard. SE-Sync: A certifiably correct algorithm for synchronization over the special euclidean group. *The International Journal of Robotics Research*, 38(2-3):95–125, 2019. 1, 2, 4, 7, 9, 15

[74] R. Rusu, N. Blodow, and M. Beetz. Fast point feature histograms (fpfh) for 3d registration. In *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, pages 3212–3217. Citeseer, 2009. 12

[75] J. L. Schonberger and J.-M. Frahm. Structure-from-motion revisited. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 4104–4113, 2016. 1, 15, 16

[76] S. Sharma and S. D'Amico. Pose estimation for non-cooperative rendezvous using neural networks. *arXiv preprint arXiv:1906.09868*, 2019. 2, 7, 14, 15, 23, 27

[77] J. Shi, H. Yang, and L. Carlone. Optimal pose and shape estimation for category-level 3D object perception. In *Robotics: Science and Systems (RSS)*, 2021. 2, 7, 14, 15

[78] J. Shi, H. Yang, and L. Carlone. ROBIN: a graph-theoretic approach to reject outliers in robust estimation using invariants. In *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, 2021. 6, 12, 16

[79] D. Sun, K.-C. Toh, and L. Yang. A convergent 3-block semiproximal alternating direction method of multipliers for conic programming with 4-type constraints. *SIAM journal on Optimization*, 25(2):882–915, 2015. 11, 22

[80] K. M. Tavish and T. D. Barfoot. At all costs: A comparison of robust cost functions for camera correspondence outliers. In *Computer and Robot Vision (CRV), 2015 12th Conference on*, pages 62–69. IEEE, 2015. 5, 7

[81] M. J. Todd, K.-C. Toh, and R. H. Tütüncü. On the nesterov–todd direction in semidefinite programming. *SIAM Journal on Optimization*, 8(3):769–796, 1998. 2, 4

[82] R. Tron, D. M. Rosen, and L. Carlone. On the inclusion of determinant constraints in lagrangian duality for 3d slam. In *Robotics: Science and Systems (RSS), Workshop "The problem of mobile sensors: Setting future goals and indicators of progress for SLAM*, volume 4, 2015. 7, 18

[83] R. H. Tütüncü, K.-C. Toh, and M. J. Todd. Solving semidefinite-quadratic-linear programs using SDPT3. *Mathematical programming*, 95(2):189–217, 2003. 2, 3, 4

[84] V. V. Vazirani. *Approximation algorithms*. Springer Science & Business Media, 2013. 2

[85] H. Waki, S. Kim, M. Kojima, and M. Muramatsu. Sums of squares and semidefinite program relaxations for polynomial optimization problems with structured sparsity. *SIAM J. Optim.*, 17(1):218–242, 2006. 20

[86] J. Wang, V. Magron, and J.-B. Lasserre. Chordal-TSSOS: a moment-SOS hierarchy that exploits term sparsity with chordal extension. *SIAM Journal on Optimization*, 31(1):114–141, 2021. 11, 20

[87] J. Wang, V. Magron, and J.-B. Lasserre. TSSOS: A Moment-SOS hierarchy that exploits term sparsity. *SIAM Journal on Optimization*, 31(1):30–58, 2021. 9, 15

[88] L. Wang and A. Singer. Exact and stable recovery of rotations for robust synchronization. *Information and Inference: A Journal of the IMA*, 30, 2013. 2, 16

[89] 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. Machine Intell.*, 2019. 3, 7, 14, 16, 23, 28

[90] E. Wise, M. Giamou, S. Khoubyarian, A. Grover, and J. Kelly. Certifiably optimal monocular hand-eye calibration. In *IEEE International Conference on Multisensor Fusion and Integration for*

Intelligent Systems (MFI), pages 271–278. IEEE, 2020. 2

[91] Y. Xiang, R. Mottaghi, and S. Savarese. Beyond pascal: A benchmark for 3d object detection in the wild. In *IEEE winter conference on applications of computer vision*, pages 75–82. IEEE, 2014. 16

[92] Y. Xiang, R. Mottaghi, and S. Savarese. Beyond PASCAL: A benchmark for 3d object detection in the wild. In *IEEE Winter Conference on Applications of Computer Vision*, pages 75–82. IEEE, 2014. 14

[93] 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 Robotics and Automation Letters*, 5(2):1127–1134, 2020. 1, 7, 11, 15, 22

[94] H. Yang and L. Carlone. A polynomial-time solution for robust registration with extreme outlier rates. In *Robotics: Science and Systems (RSS)*, 2019. 1, 16

[95] H. Yang and L. Carlone. A quaternion-based certifiably optimal solution to the wahba problem with outliers. In *Intl. Conf. on Computer Vision (ICCV)*, pages 1665–1674, 2019. 2, 7, 12, 16

[96] H. Yang and L. Carlone. In perfect shape: Certifiably optimal 3d shape reconstruction from 2d landmarks. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 621–630, 2020. 2, 7, 15, 18

[97] H. Yang and L. Carlone. One ring to rule them all: Certifiably robust geometric perception with outliers. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020. 3, 8, 9, 12

[98] H. Yang, W. Dong, L. Carlone, and V. Koltun. Self-supervised geometric perception. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 14350–14361, 2021. 6

[99] H. Yang, C. Doran, and J.-J. Slotine. Dynamical pose estimation. In *Intl. Conf. on Computer Vision (ICCV)*, 2021. 7

[100] H. Yang, L. Liang, K.-C. Toh, and L. Carlone. STRIDE along Spectrahedral Vertices for Solving Large-Scale Rank-One Semidefinite Relaxations. *arXiv preprint arXiv:2105.14033*, 2021. 3, 4, 10

[101] H. Yang, J. Shi, and L. Carlone. TEASER: Fast and Certifiable Point Cloud Registration. *IEEE Trans. Robotics*, 2020. 1, 6, 11

[102] J. Yang, H. Li, D. Campbell, and Y. Jia. Go-ICP: A globally optimal solution to 3D ICP point-set registration. *IEEE Trans. Pattern Anal. Machine Intell.*, 38(11):2241–2254, Nov. 2016. 1

[103] J. Yang, H. Li, and Y. Jia. Optimal essential matrix estimation via inlier-set maximization. In *European Conf. on Computer Vision (ECCV)*, pages 111–126. Springer, 2014. 16

[104] L. Yang, D. Sun, and K.-C. Toh. SDPNAL+: a majorized semismooth newton-cg augmented lagrangian method for semidefinite programming with nonnegative constraints. *Math. Program. Comput.*, 7(3):331–366, 2015. 4, 11, 15, 22

[105] A. Zeng, S. Song, M. Nießner, M. Fisher, J. Xiao, and T. Funkhouser. 3dmatch: Learning the matching of local 3d geometry in range scans. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, volume 1, page 4, 2017. 2, 12, 23, 26

[106] R. Y. Zhang and J. Lavaei. Sparse semidefinite programs with guaranteed near-linear time complexity via dualized clique tree conversion. *Mathematical programming*, pages 1–43, 2020. 4

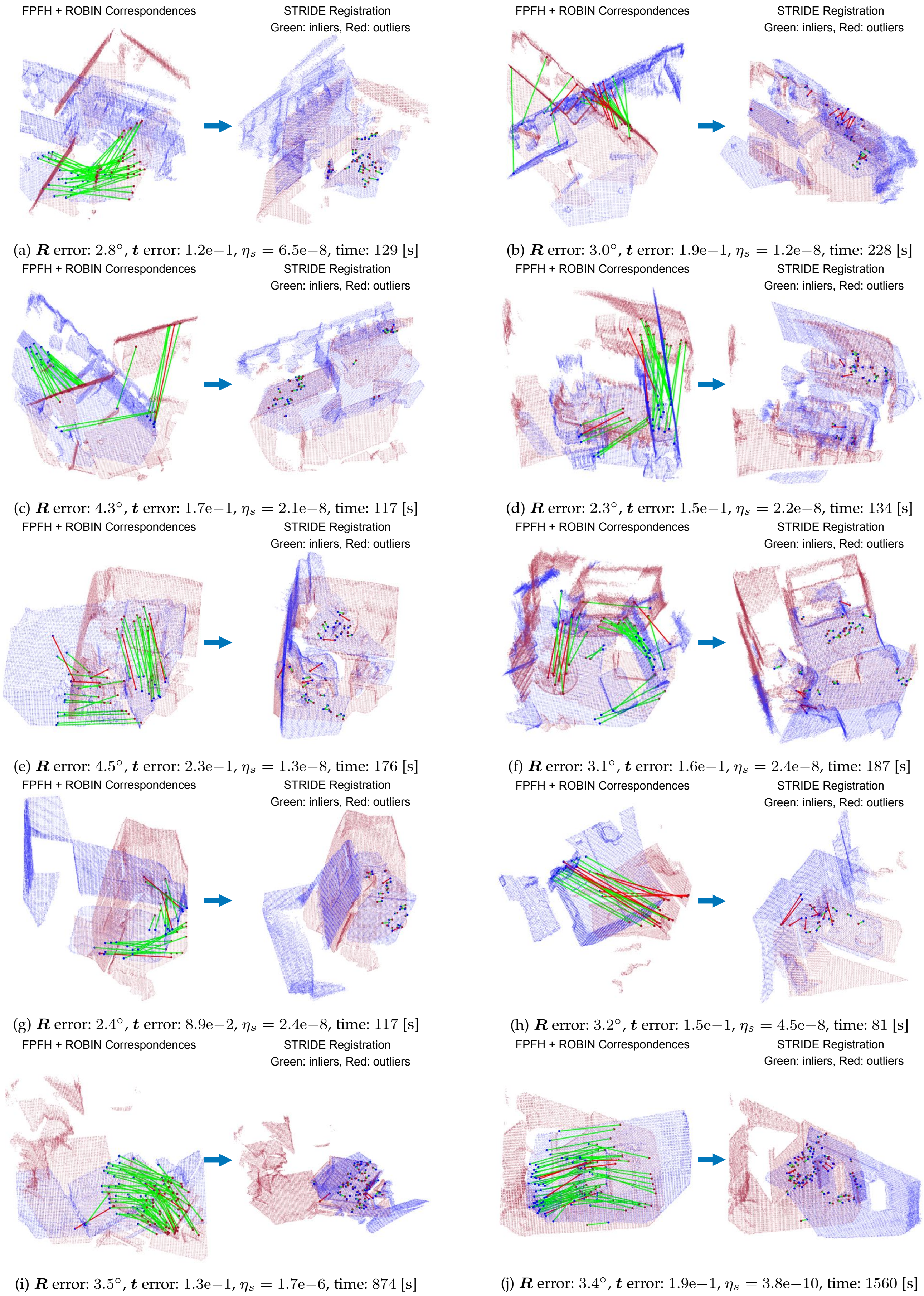
[107] J. Zhao. An efficient solution to non-minimal case essential matrix estimation. *IEEE Trans. Pattern Anal. Machine Intell.*, 2020. 2, 15

[108] X.-Y. Zhao, D. Sun, and K.-C. Toh. A newton-cg augmented lagrangian method for semidefinite programming. *SIAM Journal on Optimization*, 20(4):1737–1765, 2010. 20

[109] Y. Zheng, G. Fantuzzi, A. Papachristodoulou, P. Goulart, and A. Wynn. Chordal decomposition in operator-splitting methods for sparse semidefinite programs. *Mathematical Programming*, 180(1):489–532, 2020. 4

Heng Yang is a Ph.D. candidate in the Department of Mechanical Engineering and the Laboratory for Information & Decision Systems at the Massachusetts Institute of Technology. He has obtained a B.S. degree in Mechanical Engineering (with honors) from the Tsinghua University, Beijing, China, in 2015; and an S.M. degree in Mechanical Engineering from MIT in 2017. His research interests include large-scale convex optimization, semidefinite relaxation, robust estimation, and machine learning, applied to computer vision. His work includes developing certifiably optimal outlier-robust machine perception algorithms, large-scale semidefinite programming solvers, and self-supervised geometric perception frameworks. Heng Yang is a recipient of the Best Paper Award in Robot Vision at the 2020 IEEE International Conference on Robotics and Automation (ICRA), a Best Paper Award Honorable Mention from the 2020 IEEE Robotics and Automation Letters (RA-L), and a Best Paper Award Finalist at the 2021 Robotics: Science and Systems (RSS) conference. He is a Class of 2021 RSS Pioneer.

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



(a) $\boldsymbol{R}$ error: $2.8°$, $\boldsymbol{t}$ error: $1.2\text{e}{-1}$, $\eta_s = 6.5\text{e}{-8}$, time: 129 [s]

(b) $\boldsymbol{R}$ error: $3.0°$, $\boldsymbol{t}$ error: $1.9\text{e}{-1}$, $\eta_s = 1.2\text{e}{-8}$, time: 228 [s]

(c) $\boldsymbol{R}$ error: $4.3°$, $\boldsymbol{t}$ error: $1.7\text{e}{-1}$, $\eta_s = 2.1\text{e}{-8}$, time: 117 [s]

(d) $\boldsymbol{R}$ error: $2.3°$, $\boldsymbol{t}$ error: $1.5\text{e}{-1}$, $\eta_s = 2.2\text{e}{-8}$, time: 134 [s]

(e) $\boldsymbol{R}$ error: $4.5°$, $\boldsymbol{t}$ error: $2.3\text{e}{-1}$, $\eta_s = 1.3\text{e}{-8}$, time: 176 [s]

(f) $\boldsymbol{R}$ error: $3.1°$, $\boldsymbol{t}$ error: $1.6\text{e}{-1}$, $\eta_s = 2.4\text{e}{-8}$, time: 187 [s]

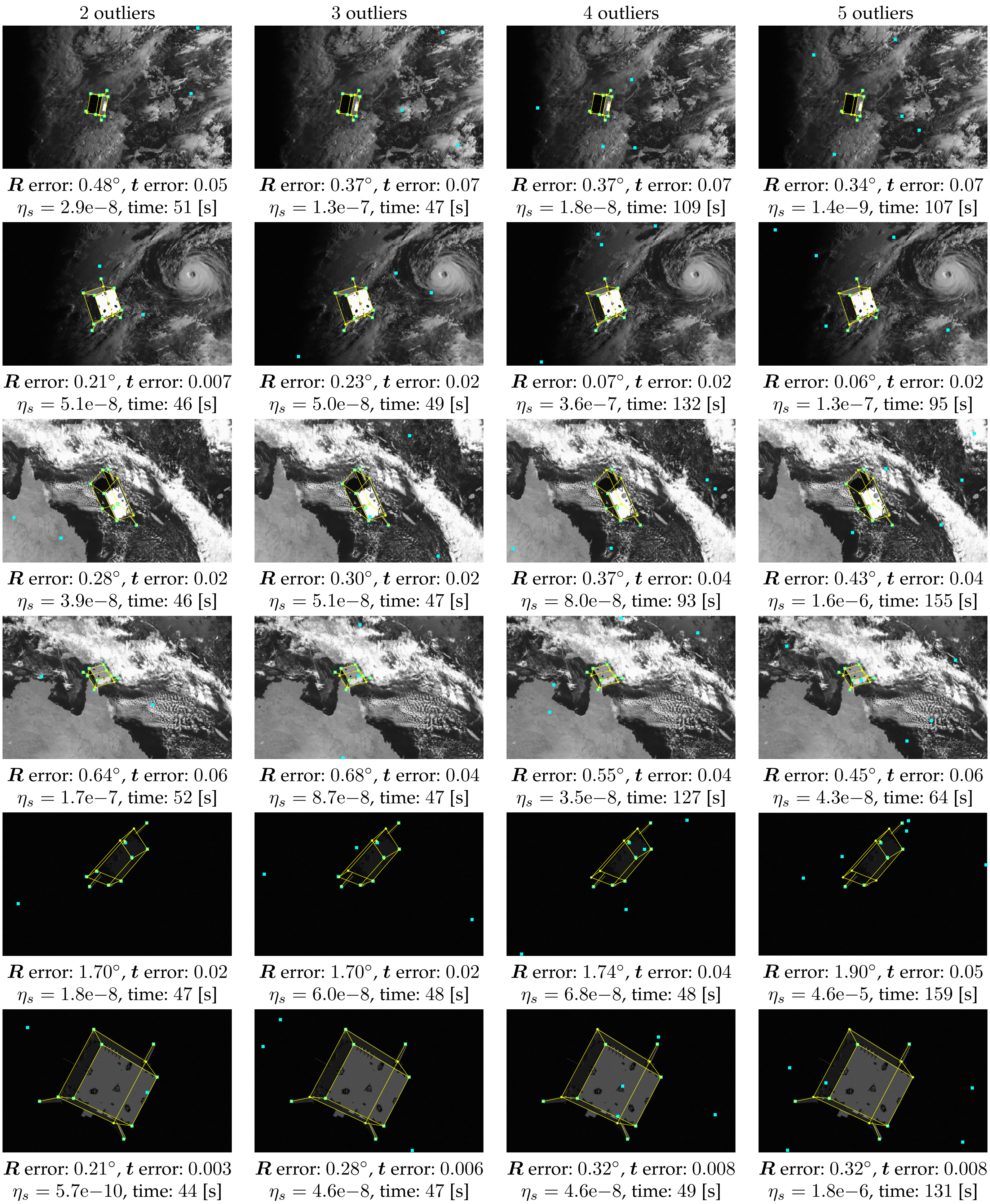
(g) $\boldsymbol{R}$ error: $2.4°$, $\boldsymbol{t}$ error: $8.9\text{e}{-2}$, $\eta_s = 2.4\text{e}{-8}$, time: 117 [s]

(h) $\boldsymbol{R}$ error: $3.2°$, $\boldsymbol{t}$ error: $1.5\text{e}{-1}$, $\eta_s = 4.5\text{e}{-8}$, time: 81 [s]

(i) $\boldsymbol{R}$ error: $3.5°$, $\boldsymbol{t}$ error: $1.3\text{e}{-1}$, $\eta_s = 1.7\text{e}{-6}$, time: 874 [s]

(j) $\boldsymbol{R}$ error: $3.4°$, $\boldsymbol{t}$ error: $1.9\text{e}{-1}$, $\eta_s = 3.8\text{e}{-10}$, time: 1560 [s]

Fig. A1. Extra scan matching results on 3DMatch [105].



2 outliers | 3 outliers | 4 outliers | 5 outliers

$\boldsymbol{R}$ error: 0.48°, $\boldsymbol{t}$ error: 0.05
$\eta_s = 2.9\text{e}{-8}$, time: 51 [s]

$\boldsymbol{R}$ error: 0.37°, $\boldsymbol{t}$ error: 0.07
$\eta_s = 1.3\text{e}{-7}$, time: 47 [s]

$\boldsymbol{R}$ error: 0.37°, $\boldsymbol{t}$ error: 0.07
$\eta_s = 1.8\text{e}{-8}$, time: 109 [s]

$\boldsymbol{R}$ error: 0.34°, $\boldsymbol{t}$ error: 0.07
$\eta_s = 1.4\text{e}{-9}$, time: 107 [s]

$\boldsymbol{R}$ error: 0.21°, $\boldsymbol{t}$ error: 0.007
$\eta_s = 5.1\text{e}{-8}$, time: 46 [s]

$\boldsymbol{R}$ error: 0.23°, $\boldsymbol{t}$ error: 0.02
$\eta_s = 5.0\text{e}{-8}$, time: 49 [s]

$\boldsymbol{R}$ error: 0.07°, $\boldsymbol{t}$ error: 0.02
$\eta_s = 3.6\text{e}{-7}$, time: 132 [s]

$\boldsymbol{R}$ error: 0.06°, $\boldsymbol{t}$ error: 0.02
$\eta_s = 1.3\text{e}{-7}$, time: 95 [s]

$\boldsymbol{R}$ error: 0.28°, $\boldsymbol{t}$ error: 0.02
$\eta_s = 3.9\text{e}{-8}$, time: 46 [s]

$\boldsymbol{R}$ error: 0.30°, $\boldsymbol{t}$ error: 0.02
$\eta_s = 5.1\text{e}{-8}$, time: 47 [s]

$\boldsymbol{R}$ error: 0.37°, $\boldsymbol{t}$ error: 0.04
$\eta_s = 8.0\text{e}{-8}$, time: 93 [s]

$\boldsymbol{R}$ error: 0.43°, $\boldsymbol{t}$ error: 0.04
$\eta_s = 1.6\text{e}{-6}$, time: 155 [s]

$\boldsymbol{R}$ error: 0.64°, $\boldsymbol{t}$ error: 0.06
$\eta_s = 1.7\text{e}{-7}$, time: 52 [s]

$\boldsymbol{R}$ error: 0.68°, $\boldsymbol{t}$ error: 0.04
$\eta_s = 8.7\text{e}{-8}$, time: 47 [s]

$\boldsymbol{R}$ error: 0.55°, $\boldsymbol{t}$ error: 0.04
$\eta_s = 3.5\text{e}{-8}$, time: 127 [s]

$\boldsymbol{R}$ error: 0.45°, $\boldsymbol{t}$ error: 0.06
$\eta_s = 4.3\text{e}{-8}$, time: 64 [s]

$\boldsymbol{R}$ error: 1.70°, $\boldsymbol{t}$ error: 0.02
$\eta_s = 1.8\text{e}{-8}$, time: 47 [s]

$\boldsymbol{R}$ error: 1.70°, $\boldsymbol{t}$ error: 0.02
$\eta_s = 6.0\text{e}{-8}$, time: 48 [s]

$\boldsymbol{R}$ error: 1.74°, $\boldsymbol{t}$ error: 0.04
$\eta_s = 6.8\text{e}{-8}$, time: 48 [s]

$\boldsymbol{R}$ error: 1.90°, $\boldsymbol{t}$ error: 0.05
$\eta_s = 4.6\text{e}{-5}$, time: 159 [s]

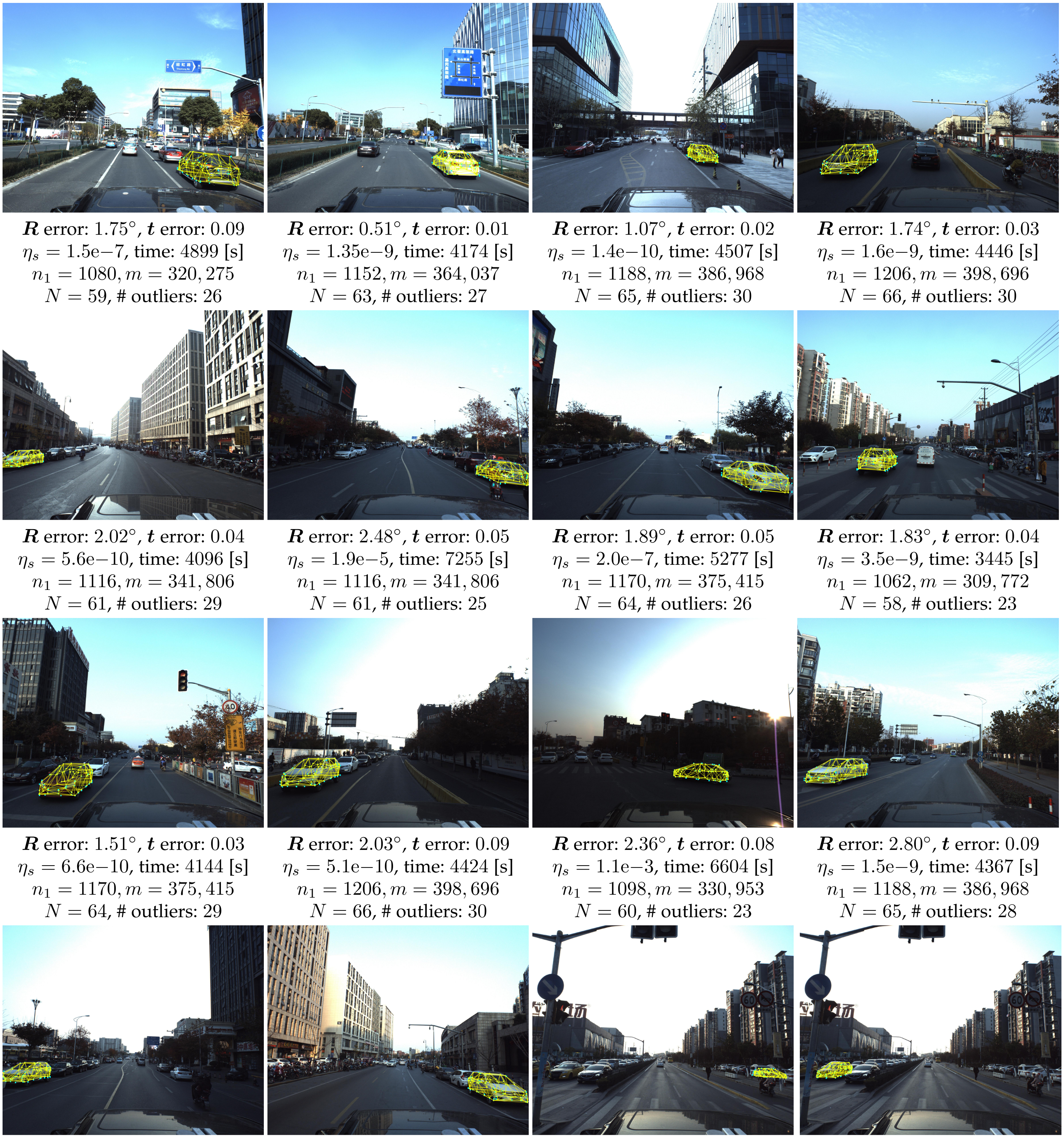
$\boldsymbol{R}$ error: 0.21°, $\boldsymbol{t}$ error: 0.003
$\eta_s = 5.7\text{e}{-10}$, time: 44 [s]

$\boldsymbol{R}$ error: 0.28°, $\boldsymbol{t}$ error: 0.006
$\eta_s = 4.6\text{e}{-8}$, time: 47 [s]

$\boldsymbol{R}$ error: 0.32°, $\boldsymbol{t}$ error: 0.008
$\eta_s = 4.6\text{e}{-8}$, time: 49 [s]

$\boldsymbol{R}$ error: 0.32°, $\boldsymbol{t}$ error: 0.008
$\eta_s = 1.8\text{e}{-6}$, time: 131 [s]

Fig. A2. Extra satellite pose estimation results on SPEED [76].



$\boldsymbol{R}$ error: 1.75°, $\boldsymbol{t}$ error: 0.09
$\eta_s = 1.5\text{e}{-7}$, time: 4899 [s]
$n_1 = 1080, m = 320,275$
$N = 59$, # outliers: 26

$\boldsymbol{R}$ error: 0.51°, $\boldsymbol{t}$ error: 0.01
$\eta_s = 1.35\text{e}{-9}$, time: 4174 [s]
$n_1 = 1152, m = 364,037$
$N = 63$, # outliers: 27

$\boldsymbol{R}$ error: 1.07°, $\boldsymbol{t}$ error: 0.02
$\eta_s = 1.4\text{e}{-10}$, time: 4507 [s]
$n_1 = 1188, m = 386,968$
$N = 65$, # outliers: 30

$\boldsymbol{R}$ error: 1.74°, $\boldsymbol{t}$ error: 0.03
$\eta_s = 1.6\text{e}{-9}$, time: 4446 [s]
$n_1 = 1206, m = 398,696$
$N = 66$, # outliers: 30

$\boldsymbol{R}$ error: 2.02°, $\boldsymbol{t}$ error: 0.04
$\eta_s = 5.6\text{e}{-10}$, time: 4096 [s]
$n_1 = 1116, m = 341,806$
$N = 61$, # outliers: 29

$\boldsymbol{R}$ error: 2.48°, $\boldsymbol{t}$ error: 0.05
$\eta_s = 1.9\text{e}{-5}$, time: 7255 [s]
$n_1 = 1116, m = 341,806$
$N = 61$, # outliers: 25

$\boldsymbol{R}$ error: 1.89°, $\boldsymbol{t}$ error: 0.05
$\eta_s = 2.0\text{e}{-7}$, time: 5277 [s]
$n_1 = 1170, m = 375,415$
$N = 64$, # outliers: 26

$\boldsymbol{R}$ error: 1.83°, $\boldsymbol{t}$ error: 0.04
$\eta_s = 3.5\text{e}{-9}$, time: 3445 [s]
$n_1 = 1062, m = 309,772$
$N = 58$, # outliers: 23

$\boldsymbol{R}$ error: 1.51°, $\boldsymbol{t}$ error: 0.03
$\eta_s = 6.6\text{e}{-10}$, time: 4144 [s]
$n_1 = 1170, m = 375,415$
$N = 64$, # outliers: 29

$\boldsymbol{R}$ error: 2.03°, $\boldsymbol{t}$ error: 0.09
$\eta_s = 5.1\text{e}{-10}$, time: 4424 [s]
$n_1 = 1206, m = 398,696$
$N = 66$, # outliers: 30

$\boldsymbol{R}$ error: 2.36°, $\boldsymbol{t}$ error: 0.08
$\eta_s = 1.1\text{e}{-3}$, time: 6604 [s]
$n_1 = 1098, m = 330,953$
$N = 60$, # outliers: 23

$\boldsymbol{R}$ error: 2.80°, $\boldsymbol{t}$ error: 0.09
$\eta_s = 1.5\text{e}{-9}$, time: 4367 [s]
$n_1 = 1188, m = 386,968$
$N = 65$, # outliers: 28

$\boldsymbol{R}$ error: 2.12°, $\boldsymbol{t}$ error: 0.10
$\eta_s = 3.2\text{e}{-10}$, time: 4482 [s]
$n_1 = 1152, m = 364,037$
$N = 63$, # outliers: 28

$\boldsymbol{R}$ error: 2.62°, $\boldsymbol{t}$ error: 0.04
$\eta_s = 1.2\text{e}{-10}$, time: 4524 [s]
$n_1 = 1152, m = 364,037$
$N = 63$, # outliers: 29

$\boldsymbol{R}$ error: 2.62°, $\boldsymbol{t}$ error: 0.07
$\eta_s = 1.8\text{e}{-7}$, time: 2344 [s]
$n_1 = 684, m = 129,634$
$N = 37$, # outliers: 16

$\boldsymbol{R}$ error: 2.19°, $\boldsymbol{t}$ error: 0.09
$\eta_s = 3.1\text{e}{-10}$, time: 5090 [s]
$n_1 = 1170, m = 375,415$
$N = 64$, # outliers: 28

Fig. A3. Extra vehicle pose and shape estimation on ApolloScape [89].