Scalable Low-Rank Semidefinite Programming for Certifiably Correct Machine Perception

David M. Rosen

Laboratory for Information and Decision Systems, Massachusetts Institute of Technology, Cambridge, MA 02139, USA

Abstract. Semidefinite relaxation has recently emerged as a powerful technique for machine perception, in many cases enabling the recovery of *certifiably globally optimal* solutions of generally-intractable nonconvex estimation problems. However, the high computational cost of standard interior-point methods for semidefinite optimization prevents these algorithms from scaling effectively to the high-dimensional problems frequently encountered in machine perception tasks. To address this challenge, in this paper we present an efficient algorithm for solving the large-scale but *low-rank* semidefinite relaxations that underpin current certifiably correct machine perception methods. Our algorithm preserves the scalability of current state-of-the-art low-rank semidefinite optimizers that are *custom-built* for the geometry of specific machine perception problems, but generalizes to a broad class of convex programs over spectrahedral sets *without* the need for detailed manual analysis or design. The result is an easy-to-use, general-purpose computational tool capable of supporting the development and deployment of a broad class of novel certifiably correct machine perception methods.

1 Introduction

Motivation. Over the coming decades, the increasingly widespread adoption of robotic technology in fields such as transportation [26], medicine [39], and disaster response [17, 22] has enormous potential to increase productivity, alleviate suffering, and preserve life. At the same time, however, the applications for which robotics is poised to realize the greatest societal benefit typically carry a correspondingly high cost of poor performance. For example, the failure of an autonomous vehicle to function correctly may lead to destruction of property, severe injury, or even loss of life [43]. This presents a serious barrier to the widespread adoption of robotic technology in high-impact but safety-critical applications, absent some guarantee of "good behavior" on the part of autonomous agents. While such guarantees (to wit, of correctness, feasibility, optimality, bounded suboptimality, etc.), have long been a feature of algorithms for planning [37] and control [1, 38], to date the development of practical algorithms with guaranteed performance for the high-dimensional and nonconvex estimation problems typical of machine *perception* has largely remained an open problem [10].

Convex relaxation approaches based upon semidefinite programming have recently emerged as one very promising paradigm for addressing the challenge of provably reliable machine perception [32]. A nascent but growing body of work indicates that this approach can be remarkably effective in practice, in many instances enabling the efficient recovery of *certifiably globally optimal* solutions of generally-intractable nonconvex estimation problems [3, 6, 7, 16, 18, 29, 35, 36]. However, despite these encouraging initial results, the high computational cost of standard interior-point algorithms for semidefinite optimization [40] remains a major obstacle to the development of practical semidefinite-relaxation-based certifiably correct algorithms for the high-dimensional estimation problems frequently encountered in machine perception.

Related work. Semidefinite optimization [40, 42] remains a very active research area, with a particular focus in recent years on the development of structure-exploiting scalable algorithms [27]. In machine perception, prior work on semidefinite relaxation has primarily employed *superlinear* optimization methods, both to address the poor numerical conditioning that is a common feature of perception problems (for example, SLAM [20] and bundle adjustment [41]), as well as to maintain competitiveness (in terms of computational speed and solution accuracy) with the highly-optimized Gauss-Newton-based *local* search methods that comprise current state-of-the-art alternatives [24, 25, 34]. This prior work can be broadly categorized into two classes. For low-dimensional applications (such as two-view registration [7]), standard central-path-following Newton methods suffice [42]. However, applications involving more than a few thousand variables (such as SLAM or bundle adjustment) require the use of specialized, structure-exploiting semidefinite optimizers [11, 40].

In previous work [33, 35, 36], we demonstrated that Burer-Monteiro factorization [8, 9] together with superlinear Riemannian optimization [2] provides an effective means of exploiting the favorable low-rank and geometric structure of a semidefinite relaxation of pose-graph SLAM to produce a fast certifiably correct estimation method [3]. Subsequent work has similarly applied this approach, *mutatis mutandis*, to produce fast certifiably correct algorithms for several other large-scale machine perception tasks [6, 18, 21]. However, while this prescription has proven to be effective, it is also quite cumbersome: it requires a manual analysis of the feasible set of the Burer-Monteiro-factored semidefinite relaxation, the recognition of this set as a Riemannian manifold whose geometry is already known,[1] and then the design and implementation of a Riemannian optimization algorithm on *that specific* manifold [2]. Moreover, it is also necessarily limited to problems whose Burer-Monteiro-factored feasible sets can be recognized as a "standard" Riemannian manifold (e.g. a sphere or a Stiefel manifold) whose geometry is already well-understood.

Contribution. We present an efficient *general-purpose* algorithm for large-scale low-rank semidefinite programming. Our algorithm leverages many of the same computational insights and design principles eludicated in our previous work on certifiably correct pose-graph SLAM [33, 35, 36], including the use of low-rank (Burer-Monteiro) factorization in conjunction with fast (superlinear)

[1] More precisely, it requires knowledge of an efficiently-computable *retraction* on that manifold [2].

optimization methods. However, it both generalizes and improves upon this prior work in several important ways:

- It admits the use of *any* convex, twice-continuously-differentiable objective.
- It admits the use of linear *inequality* constraints in addition to the linear *equality* constraints considered in previous work [6, 16, 18, 35, 36].
- Most importantly, it does *not* require any manual analysis of the problem's feasible set. Instead, it accepts, and directly operates on, the problem's description in the standard form of a conic program (Problem 1).

Our algorithm thus preserves the scalability of existing *custom-built* low-rank semidefinite optimization methods for certifiably correct machine perception, but applies to a broad class of convex programs over spectrahedral sets *without* the need for manual analysis or design. The result is an easy-to-use, general-purpose computational tool capable of supporting the development of a broad class of novel certifiably correct machine perception methods.

Notation. We use $\mathbb{R}$ and $\mathbb{R}_+$ to denote the real and nonnegative real numbers, $\mathbb{S}^n$ and $\mathbb{S}^n_+$ for the symmetric and positive semidefinite matrices of order n, respectively, and equip $\mathbb{R}^n$ and $\mathbb{S}^n$ with the standard Euclidean and Frobenius inner products (respectively). We write $[n] \triangleq \{1, \dots, n\}$ for $n > 0$ as a shorthand for sets of indexing integers. For $f \colon \mathbb{R}^n \to \mathbb{R}$ a twice-continuously-differentiable function and $x \in \mathbb{R}^n$, we write $\mathrm{D}\, f(x)[\dot{x}]$ to denote the directional derivative of f at x along the tangent direction $\dot{x} \in \mathbb{R}^n$, $\nabla f(x)$ for the gradient of f at x, and $\nabla^2 f(x)$ for the Hessian of f at x, which we will regard as the symmetric linear operator that computes the directional derivative of $\nabla f(x)$ along $\dot{x}$ [2]:

$$\begin{aligned} \nabla^2 f(x) &\colon \mathbb{R}^n \to \mathbb{R}^n \\ \nabla^2 f(x)[\dot{x}] &\triangleq \mathrm{D}\left(\nabla f(x)\right)[\dot{x}]. \end{aligned} \tag{1}$$

Finally, for $x, y \in \mathbb{R}^n$, $[x]_+$ denotes the orthogonal projection of x onto the nonnegative orthant[2] $\mathbb{R}^n_+$, and $x \odot y$ the Hadamard (elementwise) product.

2 Problem formulation

In this paper we are interested in finding *low-rank* solutions of large-scale instances of the following problem:

Problem 1 (Positive-semidefinite matrix program).

$$f^* = \min_{X \in \mathbb{S}^n} f(X) \quad \text{s.t.} \;\; \mathcal{A}(X) = b, \; \mathcal{B}(X) \leq u, \; X \succeq 0 \tag{2}$$

where $f \colon \mathbb{S}^n \to \mathbb{R}$ is a convex and twice-continuously-differentiable function, $\mathcal{A} \colon \mathbb{S}^n \to \mathbb{R}^{m_1}$ and $\mathcal{B} \colon \mathbb{S}^n \to \mathbb{R}^{m_2}$ are linear operators defined by:

$$\mathcal{A}(X)_i = \langle A_i, X \rangle \quad \forall i \in [m_1], \qquad \mathcal{B}(X)_j = \langle B_j, X \rangle \quad \forall j \in [m_2], \tag{3}$$

for fixed sets of symmetric matrices $\{A_i\}_{i=1}^{m_1}$, $\{B_j\}_{j=1}^{m_2} \subset \mathbb{S}^n$, and $b \in \mathbb{R}^{m_1}$ and $u \in \mathbb{R}^{m_2}$ are fixed right-hand-side vectors for the linear constraints in (2).

[2] In coordinates, $([x]_+)_i = \max(x_i, 0)$ for all $i \in [n]$.

As a convex optimization problem, Problem 1 can in principle be solved efficiently using general-purpose optimization methods [5, 30]. In practice, however, the high computational cost of storing and manipulating the $O(n^2)$ elements of the decision variable X appearing in (2) prevents such straightforward approaches from scaling effectively to problems in which n is larger than a few thousand [40]. Unfortunately, typical instances of many important machine perception problems (for example, SLAM [20] and bundle adjustment [41]) are several orders of magnitude larger than this maximum effective problem size, placing them far beyond the reach of such general-purpose approaches [11].

3 Low-rank semidefinite programming

While Problem 1 is computationally expensive to solve *in general*, the specific instances of (1) underpinning certifiably correct machine perception methods generically admit *low-rank* solutions X^* under realistic operating conditions (cf. [6, 16, 18, 32, 35, 36] and the references therein); such solutions can be represented much more concisely in the factored form $X^* = Y^* {Y^*}^\mathsf{T}$, where $Y^* \in \mathbb{R}^{n \times r}$ and $r \ll n$. In a pair of papers, Burer and Monteiro [8, 9] proposed to exploit the existence of such low-rank solutions by replacing the high-dimensional decision variable X in (2) with a much more concise rank-r product of the form YY^T, producing the following *rank-restricted* version of problem (2):

Problem 2 (Rank-restricted positive semidefinite matrix program).

$$f_r^* = \min_{Y \in \mathbb{R}^{n \times r}} f(YY^\mathsf{T}) \quad \text{s.t.} \;\; \mathcal{A}(YY^\mathsf{T}) = b, \; \mathcal{B}(YY^\mathsf{T}) \leq u. \tag{4}$$

This substitution has the two-fold effect of dramatically reducing the size of the search space of (4) versus (2), as well as rendering the positive-semidefiniteness constraint in (2) redundant. Moreover, it is clear by construction that if Problem 1 admits a minimizer X^* with $\operatorname{rank}(X^*) \leq r$, then we can recover such a minimizer from a solution $Y^* \in \mathbb{R}^{n \times r}$ of Problem 2 according to $X^* = Y^* {Y^*}^\mathsf{T}$. In light of these observations, Burer and Monteiro [8, 9] proposed to search for minimizers X^* of Problem 1 by attempting to recover their low-rank factors Y^* from instances of the (nonconvex) low-rank factorization Problem 2.

3.1 Ensuring optimality

In this subsection, we establish the following remarkable property of the (nonconvex) Problem 2: if a first-order critical point $Y \in \mathbb{R}^{n \times r}$ does *not* correspond to a global minimizer $X = YY^\mathsf{T}$ of Problem 1, it is possible to *detect* that this is so, and to produce a second-order *direction of descent* from Y, embedded in a higher-dimensional instance of (4). Our derivation unifies (and slightly generalizes) several closely-related results originally presented in [4, 13, 23].

Throughout this subsection (Sec. 3.1), we make the following assumption (a slight relaxation of the usual Slater constraint qualification [5, Sec. 5.2.3]):

Relaxed Slater CQ. Problem 1 admits a positive-definite feasible point.

Theorem 1 (Optimality conditions for Problem 1) *A matrix* $X \in \mathbb{S}^n$ *is a minimizer of* (2) *if and only if there exist Lagrange multipliers* $\lambda \in \mathbb{R}^{m_1}$ *and* $\gamma \in \mathbb{R}^{m_2}_+$ *such that:*

$$\mathcal{A}(X) = b \tag{5a}$$

$$\mathcal{B}(X) \leq u \tag{5b}$$

$$X \succeq 0 \tag{5c}$$

$$(\mathcal{B}(X) - u) \odot \gamma = 0 \tag{5d}$$

$$S \succeq 0 \tag{5e}$$

$$SX = 0 \tag{5f}$$

where

$$S \triangleq \nabla f(X) + \mathcal{A}^*(\lambda) + \mathcal{B}^*(\gamma) \tag{6}$$

and $\mathcal{A}^* \colon \mathbb{R}^{m_1} \to \mathbb{S}^n$ *and* $\mathcal{B}^* \colon \mathbb{R}^{m_2} \to \mathbb{S}^n$ *are the adjoint operators of the linear maps* $\mathcal{A}$ *and* $\mathcal{B}$ *defined in* (3)*:*

$$\mathcal{A}^*(\lambda) \triangleq \sum_{i=1}^{m_1} \lambda_i A_i, \quad \mathcal{B}^*(\gamma) \triangleq \sum_{j=1}^{m_2} \gamma_j B_j. \tag{7}$$

We provide a proof of Theorem 1 in the supplementary material, using some elementary machinery from variational analysis [31]. Similarly, let us now consider the first-order necessary conditions for Problem 2:

Theorem 2 (First-order necessary optimality conditions for Problem 2) *Let* $Y \in \mathbb{R}^{n \times r}$ *be a local minimizer of* (4) *that satisfies the linear independence constraint qualification. Then there exist Lagrange multipliers* $\lambda \in \mathbb{R}^{m_1}$ *and* $\gamma \in \mathbb{R}^{m_2}_+$ *such that:*

$$\mathcal{A}(YY^\mathsf{T}) = b, \tag{8a}$$

$$\mathcal{B}(YY^\mathsf{T}) \leq u, \tag{8b}$$

$$(\mathcal{B}(YY^\mathsf{T}) - u) \odot \gamma = 0 \tag{8c}$$

$$SY = 0 \tag{8d}$$

where

$$S \triangleq \nabla f(YY^\mathsf{T}) + \mathcal{A}^*(\lambda) + \mathcal{B}^*(\gamma). \tag{9}$$

Proof. Following [30, Sec. 12.3], the Lagrangian of (4) is:

$$\begin{gathered} \mathcal{L} \colon \mathbb{R}^{n \times r} \times \mathbb{R}^{m_1} \times \mathbb{R}^{m_2}_+ \to \mathbb{R} \\ \mathcal{L}(Y, \lambda, \gamma) \triangleq f(YY^\mathsf{T}) + \left\langle \mathcal{A}(YY^\mathsf{T}) - b, \lambda \right\rangle + \left\langle \mathcal{B}(YY^\mathsf{T}) - u, \gamma \right\rangle. \end{gathered} \tag{10}$$

We may rewrite this more compactly using the adjoint operators $\mathcal{A}^*$ and $\mathcal{B}^*$:

$$\mathcal{L}(Y, \lambda, \gamma) = f(YY^\mathsf{T}) + \left\langle \mathcal{A}^*(\lambda) + \mathcal{B}^*(\gamma), YY^\mathsf{T} \right\rangle - \langle \lambda, b \rangle - \langle \gamma, u \rangle. \tag{11}$$

Theorem 12.1 of [30] then shows that if $Y \in \mathbb{R}^{n\times r}$ is a local minimizer of (4), there exist Lagrange multipliers $\lambda \in \mathbb{R}^{m_1}$ and $\gamma \in \mathbb{R}_+^{m_2}$ satisfying:

$$\mathcal{A}(YY^\mathsf{T}) = b \tag{12a}$$

$$\mathcal{B}(YY^\mathsf{T}) \leq u \tag{12b}$$

$$(\mathcal{B}(YY^\mathsf{T}) - u) \odot \gamma = 0 \tag{12c}$$

$$\nabla_Y \mathcal{L}(Y, \lambda, \gamma) = 0. \tag{12d}$$

Differentiating (11), we compute:

$$\nabla_Y \mathcal{L}(Y, \lambda, \gamma) = 2\nabla f(YY^\mathsf{T})Y + 2\mathcal{A}^*(\lambda)Y + 2\mathcal{B}^*(\gamma)Y = 2SY, \tag{13}$$

where S is defined as in (9); condition (12d) is thus equivalent to (8d). The remainder of conditions (8) are identical to conditions (12). □

Recall that our goal is to obtain minimizers of Problem 1 by searching for their low-rank factors using the (nonconvex) Problem 2. To that end, let us assume that we have identified a KKT point $Y \in \mathbb{R}^{n\times r}$ of Problem 2, with associated Lagrange multipliers $\lambda \in \mathbb{R}^{m_1}$ and $\gamma \in \mathbb{R}_+^{m_2}$ (as described in Theorem 2), and consider the corresponding optimality conditions (5) of Problem 1 at $X = YY^\mathsf{T}$. Observe that the factorization $X = YY^\mathsf{T}$ automatically ensures (5c); similarly, (5a), (5b), (5d), and (5f) follow directly from (8a)–(8d). Thus, the only condition that may fail to hold is the positive-semidefiniteness condition (5e). We summarize this observation as the following:

Corollary 3 (Recovering minimizers of Problem 1 from Problem 2) *Let $Y \in \mathbb{R}^{n\times r}$ be a KKT point of (4), with Lagrange multipliers $\lambda \in \mathbb{R}^{m_1}$ and $\gamma \in \mathbb{R}_+^{m_2}$. If the matrix S defined in (9) is positive semidefinite, then $X = YY^\mathsf{T}$ is a global minimizer of (2), with corresponding Lagrange multipliers λ and γ.*

On the other hand, if Y is a KKT point of Problem 2 at which the linear independence constraint qualification (LICQ) holds, and the matrix S defined in (9) is *not* positive semidefinite, then $X = YY^\mathsf{T}$ is *not* a minimizer of Problem 1.[3] This implies that there is a direction of descent $\dot{X} \in \mathbb{S}^n$ from X in (2) that is not reflected in the low-rank factorization used in (4), *at least to first order*. This could be because Y is a saddle point for (4) (in which case there may be a *second*-order direction of descent), or because the descent direction $\dot{X}$ is towards a set of higher-rank matrices than the rank-r factorization used in (4) is able to capture. The key result that we derive in this section is that it is possible to

[3] Here the LICQ is necessary to ensure that the Lagrange multipliers (λ, γ) associated with Y are *unique* [44]. Without this condition, there could conceivably exist some *other* set of multipliers $(\lambda', \gamma') \in \mathbb{R}^{m_1} \times \mathbb{R}_+^{m_2}$ (that we do not have in hand) satisfying (8) and *also* (5), in which case X would be optimal for Problem 1. However, any multipliers satisfying (5) for $X = YY^\mathsf{T}$ satisfy (8) *a fortiori*. Therefore, the uniqueness of (λ, γ) implies that there *cannot* exist alternative multipliers (λ', γ') satisfying (5), and therefore that X is *not* optimal (by Theorem 1).

account for *both* of these potential obstructions simultaneously by leveraging a negative eigenvector of S to generate a *second*-order direction of descent from Y, embedded within a higher-rank instance of (4)

To that end, consider the *second*-order behavior of the Lagrangian $\mathcal{L}(Y,\lambda,\gamma)$ at Y. Computing the directional derivative of $\nabla_Y \mathcal{L}(Y,\lambda,\gamma)$ along the tangent direction $\dot{Y} \in \mathbb{R}^{n\times r}$, we obtain (cf. (1)):

$$\nabla_Y^2 \mathcal{L}(Y,\lambda,\gamma)[\dot{Y}] = 2\nabla^2 f(YY^\mathsf{T})\left[Y\dot{Y}^\mathsf{T} + \dot{Y}Y^\mathsf{T}\right] + 2\nabla f(YY^\mathsf{T})\dot{Y} + 2\left(\mathcal{A}^*(\lambda) + \mathcal{B}^*(\gamma)\right)\dot{Y}. \tag{14}$$

We will now exploit (14) to generate a second-order direction of descent from Y, embedded within a higher-rank instance of the relaxation (4). Since $S \not\succeq 0$, let $v \in \mathbb{R}^n$ be any vector such that $v^\mathsf{T} S v < 0$, and consider the matrices:

$$Y_+ \triangleq \begin{pmatrix} Y & 0 \end{pmatrix} \in \mathbb{R}^{n\times(r+1)}, \qquad \dot{Y}_+ \triangleq \begin{pmatrix} 0 & v \end{pmatrix} \in \mathbb{R}^{n\times(r+1)}. \tag{15}$$

Observe that $Y_+Y_+^\mathsf{T} = YY^\mathsf{T}$, so that the first-order optimality conditions (8) continue to hold at Y_+ (i.e., Y_+ is still first-order critical for (4)). Moreover,

$$Y_+\dot{Y}_+^\mathsf{T} = 0 \tag{16}$$

by construction, and therefore

$$\mathrm{D}\left(\mathcal{A}(Y_+Y_+^\mathsf{T})\right)[\dot{Y}_+] = \mathcal{A}\left(\mathrm{D}\left(Y_+Y_+^\mathsf{T}\right)[\dot{Y}]\right) = \mathcal{A}\left(Y_+\dot{Y}_+^\mathsf{T} + \dot{Y}_+Y_+^\mathsf{T}\right) = 0 \tag{17}$$

since $\mathcal{A}\colon \mathbb{S}^n \to \mathbb{R}^{m_1}$ is a linear map (and similarly for $\mathcal{B}$), so that $\dot{Y}_+$ is a first-order feasible direction for (4) at Y_+. Finally, (14) and (16) imply that:

$$\left\langle \dot{Y}_+, \nabla_Y^2 \mathcal{L}(Y_+,\lambda,\gamma)[\dot{Y}_+]\right\rangle = \left\langle \dot{Y}_+,\, 2S\dot{Y}_+\right\rangle = 2v^\mathsf{T} S v < 0 \tag{18}$$

so that $\dot{Y}_+$ is a (*second*-order) direction of descent for (4) at Y_+ [30, Thm. 12.5]. We summarize these results as the following theorem of the alternative:

Theorem 4 *Let $Y \in \mathbb{R}^{n\times r}$ be a KKT point of* (4) (*with corresponding Lagrange multipliers $\lambda \in \mathbb{R}^{m_1}$ and $\gamma \in \mathbb{R}_+^{m_2}$*) *that satisfies the linear independence constraint qualification, and S be the matrix defined in* (9). *Then exactly one of the following two cases holds:*

(*a*) *$S \succeq 0$ and $X = YY^\mathsf{T}$ is a global minimizer of* (2).
(*b*) *There exists $v \in \mathbb{R}^n$ such that $v^\mathsf{T} S v < 0$, and in that case, $Y_+ = \begin{pmatrix} Y & 0 \end{pmatrix}$ is a KKT point of* (4) *attaining the same objective value as Y, and $\dot{Y}_+ = \begin{pmatrix} 0 & v \end{pmatrix}$ is a feasible second-order direction of descent from Y_+.*

Theorem 4 immediately suggests an algorithm for recovering global minimizers of Problem 1 from Problem 2: starting at some initial rank $r > 0$, apply a *local* search technique to recover a KKT point $Y \in \mathbb{R}^{n\times r}$ of Problem 2, and construct the corresponding certificate matrix S defined in (9). If $S \succeq 0$, then Theorem 4(a) guarantees that $X = YY^\mathsf{T}$ is a global minimizer of Problem 1; otherwise, Theorem 4(b) provides a direction of descent from Y that enables us to continue the search after increasing the rank parameter r of Problem 2 [4, 23].

3.2 Restoring feasibility

A novel complication arising from the constrained optimization approach we propose in this paper is the issue of *feasibility*: in contrast to the low-rank Riemannian approaches employed in previous work [6, 18, 35, 36], we do *not* assume that the feasible set of the low-rank factorization (4) is a Riemannian manifold whose geometry is known *a priori*. In particular, we do *not* assume access to a feasible point $Y \in \mathbb{R}^{n \times r}$ for initializing the local search in (4). Instead, in this subsection we develop a procedure for producing either an initial feasible point Y for some instance of Problem 2, or a *certificate of infeasibility* for Problem 1.

Our approach is based upon the observation (due to [14]) that we can formulate the search for a feasible point of Problem 1 in terms of the following:[4]

Problem 3 (Positive-semidefinite feasibility problem).

$$\varphi^* = \min_{X \in \mathbb{S}^n} \underbrace{\|\mathcal{A}(X) - b\|^2 + \left\|[\mathcal{B}(X) - u]_+\right\|^2}_{\varphi(X)} \quad \text{s.t.} \quad X \succeq 0. \tag{19}$$

It is clear from a direct comparison of Problems 1 and 3 that the objective φ of the latter measures the sum of the squared violations of the linear constraints appearing in the former (2). It follows that $X \in \mathbb{S}^n$ is feasible for Problem 1 if and only if $\varphi(X) = 0$, and therefore that Problem 1 is feasible if and only if $\varphi^* = 0$. Furthermore, we observe that Problem 3 is itself an always-feasible convex program over the positive semidefinite cone; this suggests that we may be able to leverage a Burer-Monteiro-factored version of Problem 3 to search for low-rank feasible points Y of Problem 2. Therefore, let us consider the following:

Problem 4 (Rank-restricted positive-semidefinite feasibility problem).

$$\varphi_r^* = \min_{Y \in \mathbb{R}^{n \times r}} \varphi(YY^\mathsf{T}) \tag{20}$$

Using an argument similar to the derivation of Theorem 4, one can derive the following analogue for Problems 3 and 4:[5]

Theorem 5 *Let $Y \in \mathbb{R}^{n \times r}$ be a first-order critical point of* (20) *and*

$$\Sigma \triangleq 2\mathcal{A}^*\left(\mathcal{A}(YY^\mathsf{T}) - b\right) + 2\mathcal{B}^*\left(\left[\mathcal{B}(YY^\mathsf{T}) - u\right]_+\right). \tag{21}$$

Then exactly one of the following two cases holds:

(*a*) $\Sigma \succeq 0$ *and* $X = YY^\mathsf{T}$ *is a global minimizer of* (20).

(*b*) *There exists* $v \in \mathbb{R}^n$ *such that* $v^\mathsf{T} \Sigma v < 0$, *and in that case,* $Y_+ = \begin{pmatrix} Y & 0 \end{pmatrix}$ *is a stationary point of* (20) *attaining the same objective value as* Y, *and* $\dot{Y}_+ = \begin{pmatrix} 0 & v \end{pmatrix}$ *is a second-order direction of descent from* Y_+.

[4] More precisely, reference [14] studied (19) and (20) for the case of equality constraints ($\varphi(X) = \|A(X) - b\|^2$), and derived the corresponding specialization of Thm. 5(a).

[5] Note that the argument used to prove Thm. 4 does not *directly* apply to (19) and (20) because the inequality term $\|[\mathcal{B}(X) - u]_+\|^2$ renders φ only C^1. In the supplementary material, we show how to derive a C^∞ reformulation of (19) and (20).

3.3 The complete algorithm

The results of Sections 3.1 and 3.2 enable the implementation of an algorithm for recovering a *global minimizer* $X \in \mathbb{S}^n$ of the positive-semidefinite matrix program Problem 1 by applying a *local* constrained optimization algorithm to identify *first-order critical points* $Y \in \mathbb{R}^{n \times r}$ of a sequence of instances of the rank-restricted Problems 2 and 4 of increasing dimension. Our proposed method is shown as Algorithm 1.

Algorithm 1 Low-rank semidefinite programming

Input: Initial estimate $Y_0 \in \mathbb{R}^{n \times r_0}$ of a KKT point of Problem 2.
Output: Symmetric factor $Y \in \mathbb{R}^{n \times r}$ and Lagrange multipliers $\lambda \in \mathbb{R}^{m_1}$, $\gamma \in \mathbb{R}_+^{m_2}$ for a KKT point $X = YY^\mathsf{T}$ of Problem 1.

Initialization: $Y \leftarrow Y_0$.
loop
    $(Y, \lambda, \gamma) \leftarrow$ LocalOptimization(Y). ▷ Search for KKT point of Problem 2
    if $\varphi(Y) > 0$ **then** ▷ Y is an infeasible stationary point
        Construct certificate matrix Σ in (21).
        $(\theta, v) \leftarrow$ MinimumEigenpair(Σ).
        if $\theta \geq 0$ **then** ▷ Problem 1 is infeasible (Thm. 5(a))
            return (Y, λ, γ)
        else
            Set $Y \leftarrow \begin{pmatrix} Y & 0 \end{pmatrix}$ and $\dot{Y} \leftarrow \begin{pmatrix} 0 & v \end{pmatrix}$.
            $Y \leftarrow$ LineSearch$(\varphi, Y, \dot{Y})$. ▷ Descend from Y (Thm. 5(b))
        end if
    else ▷ Y is a KKT point of Problem 2
        Construct certificate matrix S in (9).
        $(\theta, v) \leftarrow$ MinimumEigenpair(S).
        if $\theta \geq 0$ **then**
            return (Y, λ, γ) ▷ $X = YY^\mathsf{T}$ is a KKT point of Problem 1 (Cor. 4)
        else
            Set $Y \leftarrow \begin{pmatrix} Y & 0 \end{pmatrix}$ and $\dot{Y} \leftarrow \begin{pmatrix} 0 & v \end{pmatrix}$.
            $Y \leftarrow$ LineSearch$(f, Y, \dot{Y})$. ▷ Descend from Y (Thm. 4(b))
        end if
    end if
end loop

---

4 Experimental evaluation

In this section we evaluate the numerical performance of our general-purpose low-rank semidefinite programming method (Algorithm 1). For the purposes of this evaluation, we adopt as our test suite the large-scale pose-graph SLAM benchmarks previously considered in [36, Sec. 6.2], and SE-Sync as our baseline for comparison. SE-Sync is representative of many current state-of-the-art semidefinite-relaxation-based machine perception algorithms [6, 16, 18, 29, 35, 36] in that the estimation problem it is designed to solve requires the co-registration of local geometric data under rigid transformations of Euclidean space. Moreover, SE-Sync is a mature, highly-optimized algorithm that is specifically designed to solve the pose-graph SLAM problem efficiently; indeed, to the best of our knowledge, it is the fastest available pose-graph SLAM method of *any* kind [33, 36], and is therefore a very challenging baseline for comparison.

4.1 Experimental problem formulation

In this subsection we provide a brief review of the pose-graph SLAM problem and its semidefinite relaxations employed by SE-Sync; readers are encouraged to consult [36] for additional details.

Pose-graph SLAM requires the estimation of a collection of p unknown poses $x_i \in \mathrm{SE}(d) \cong \mathbb{R}^d \rtimes \mathrm{SO}(d)$ given noisy observations $\tilde{x}_{ij} \approx x_i^{-1} x_j$ of a subset of their pairwise relative transforms. Under a suitable noise model, this problem can be formalized as a maximum-likelihood estimation over a directed graph $G = (\mathcal{V}, \overrightarrow{\mathcal{E}})$ whose nodes are in one-to-one correspondence with the unknown states $x_i = (t_i, R_i) \in \mathrm{SE}(d)$, and whose directed edges $(i,j) \in \overrightarrow{\mathcal{E}}$ are in one-to-one correspondence with the available measurements $\tilde{x}_{ij} = (\tilde{t}_{ij}, \tilde{R}_{ij}) \in \mathrm{SE}(d)$:

$$p^*_{\mathrm{MLE}} = \min_{\substack{t_i \in \mathbb{R}^d \\ R_i \in \mathrm{SO}(d)}} \sum_{(i,j) \in \overrightarrow{\mathcal{E}}} \kappa_{ij} \|R_j - R_i \tilde{R}_{ij}\|_F^2 + \tau_{ij} \left\|t_j - t_i - R_i \tilde{t}_{ij}\right\|_2^2. \tag{22}$$

Using the fact that the minimization over translations $t = (t_1, \dots, t_p) \in \mathbb{R}^{dp}$ in (22) is an unconstrained linear least-squares problem, it is possible to solve for an optimal assignment $t^*(R)$ of these states as a (linear) function of the rotations $R = (R_1, \dots, R_p) \in \mathrm{SO}(d)^p \subset \mathbb{R}^{d \times dp}$. Substituting $t^*(R)$ into (22) then reduces it to the minimization of a convex quadratic function of the rotations *only*, which can be expressed in the following matricized form:

$$p^*_{\mathrm{MLE}} = \min_{R \in \mathrm{SO}(d)^p} \operatorname{tr}(\tilde{Q} R^\mathsf{T} R) \tag{23}$$

where $\tilde{Q} \in \mathbb{S}^{dp}_+$ is a (constant) data matrix constructed from the measurements $\tilde{x}_{ij}$ [36, Sec. 4.1]. The semidefinite relaxation employed by SE-Sync [35, 36] is obtained from (23) by replacing the rank-d symmetric product $R^\mathsf{T} R \in \mathbb{S}^{dp}_+$ with a *generic* positive-semidefinite matrix $Z \in \mathbb{S}^{dp}_+$ whose $(d \times d)$-block-diagonal is comprised of identity matrices [36, Sec. 4.2]:

$$\begin{aligned} p^*_{\mathrm{SDP}} = \min_{Z \in \mathbb{S}^{dp}_+} & \operatorname{tr}(\tilde{Q} Z) \\ \text{s.t.} \quad & \operatorname{BlockDiag}_{d \times d}(Z) = (I_d, \dots, I_d). \end{aligned} \tag{24}$$

Alternatively, one can express (22) directly in matricized form:

$$p^*_{\mathrm{MLE}} = \min_{X \in \mathbb{R}^{d \times p} \times \mathrm{SO}(d)^p} \operatorname{tr}(\tilde{M} X^\mathsf{T} X) \tag{25}$$

where again $\tilde{M} \in \mathbb{S}_+^{(d+1)p}$ is a (constant) data matrix constructed from the measurements $\tilde{x}_{ij}$, and $X = (t_1, \ldots, t_p, R_1, \ldots, R_p) \in \mathbb{R}^{d \times p} \times \mathrm{SO}(d)^p \subset \mathbb{R}^{d \times (d+1)p}$ is a block matrix that aggregates the translational and rotational states of (22) [36, Sec. 4.1]. As before, we can produce a semidefinite relaxation of (25) by replacing the rank-d symmetric product $X^\mathsf{T} X \in \mathbb{S}_+^{(d+1)p}$ with a generic positive-semidefinite matrix $Z \in \mathbb{S}_+^{(d+1)p}$ whose lower-right $dp \times dp$ block Z_{22} has a $(d \times d)$-block-diagonal comprised of identity matrices:

$$
\begin{aligned}
p^*_{\mathrm{SDP}} = & \min_{Z \in \mathbb{S}_+^{(d+1)p}} \operatorname{tr}(\tilde{M} Z) \\
\text{s.t.} \quad & \operatorname{BlockDiag}_{d \times d}(Z_{22}) = (I_d, \ldots, I_d).
\end{aligned}
\tag{26}
$$

Relaxations of the form (26) were previously studied in [6].

In the following experiments, we make use of both relaxations (24) and (26). Relaxation (24) has the advantages of a lower-dimensional, compact search space and a better-conditioned objective (since $\tilde{Q}$ is obtained as a generalized Schur complement of $\tilde{M}$ [19, 28]), and can therefore be solved more efficiently. However, the data matrix $\tilde{Q}$ parameterizing it is generically dense, and therefore in order to realize an efficient optimization algorithm, it is necessary to develop custom procedures that enable the computation of products with $\tilde{Q}$ *without* the need to explicitly instantiate it as a matrix [36, Sec. 5.1.1]. In contrast, relaxation (26) has a higher-dimensional, noncompact search space and a more poorly-conditioned objective, but has the advantage of a sparse parameter matrix $\tilde{M}$, which does not require the use of specialized linear-algebraic subroutines.

4.2 Implementation and results

In the following experiments we compare the performance of three algorithms for solving the pose-graph SLAM problem (22) by means of semidefinite relaxation. The first two (SE-Sync-S and SE-Sync-T) solve the *simplified* and *translation-explict* relaxations (24) and (26), respectively, by applying *custom-built* Riemannian optimization methods to their low-rank factorizations (4) (as described in [33, 35, 36] and [6], respectively). The third approach (Alg1-T) solves the translation-explicit relaxation (26) using our general-purpose low-rank semidefinite programming method Algorithm 1.[6]

All experiments are performed on a Lenovo T480 laptop with an Intel Core i7-8650U 1.90 GHz processor and 16 Gb of RAM running Ubuntu 18.04. Our experimental implementation of Algorithm 1[7] is written in C++, using the IPOPT library[8] [45] to perform the local optimization required in line 3, and the Spectra[9]

[6] We do not apply Algorithm 1 to (24) because (as described in Sec. 4.1) the data matrix $\tilde{Q}$ is dense, and Algorithm 1 does not implement the specialized linear-algebraic subroutines developed in [33, 35, 36] that enable efficient operations with $\tilde{Q}$.

[7] Available at https://github.com/david-m-rosen/LowRankSDP

[8] Version 3.13.1, available at https://github.com/coin-or/Ipopt.

[9] Available at https://spectralib.org/.

	# Poses	# Measurements	SE-Sync-S			SE-Sync-T			Alg1-T		
			Opt [s]	Eig [s]	Total [s]	Opt [s]	Eig [s]	Total [s]	Opt [s]	Eig [s]	Total [s]
manhattan	3500	5453	0.034	2.387	2.531	0.226	18.659	18.975	0.757	0.166	0.940
city	10000	20687	0.702	1.076	2.332	1.576	4.593	6.737	2.795	0.596	3.401
csail	1045	1172	0.003	0.34	0.355	0.019	4.815	4.844	0.072	0.023	0.101
intel	1728	2512	0.023	0.223	0.276	0.127	0.64	0.797	0.155	0.049	0.209
ais2klinik	15115	16727	2.152	20.151	22.514	148.648	115.764	264.613	3.683	0.431	4.122

Table 1. Results for the 2D SLAM benchmark datasets.

and Pardiso[10] libraries to perform the minimum-eigenpair computations in lines 6 and 15 using spectrally-shifted inverse Lanczos iterations. The line searches in lines 11 and 20 are performed via backtracking [30, Chp. 5], and second-order corrections are employed (as necessary) in line 20 to ensure that the accepted iterate Y maintains second-order feasibility while decreasing the objective f [15, Sec. 15.3.2.3]. The C++ implementations of SE-Sync-S and SE-Sync-T used in these experiments are provided by the SE-Sync library.[11]

We adopt as our test set a collection of 10 pose-graph SLAM benchmarks previously considered in [36]. Four of these (the manhattan, city, sphere, and torus datasets) are synthetic, while the remainder (the csail, intel, ais2klinik, garage, cubicle, and rim datasets) are real-world examples. Following [36, Sec. 6], we initialize each algorithm using the *chordal initialization* [12], and set the initial relaxation rank to $r_0 = d$, where $d \in \{2, 3\}$ is the dimension of the Euclidean space in which the pose-graph SLAM problem (22) is defined. Each algorithm is limited to a maximum of 500 iterations, and convergence is declared whenever the norm of the feasibility violation[12] is less than 10^{-4} *and* the norm of the (Riemannian) gradient is less than 10^{-2} or the relative decrease in function value between two subsequent (accepted) iterations is less than 10^{-7}. Results of the experiments are shown in Tables 1 and 2: these report the number of poses and measurements appearing in the original pose-graph SLAM problem (22), and the elapsed computation time spent in local optimization, minimum-eigenvalue computation, and in total, for each of the three algorithms considered.[13]

The results reveal several interesting trends. First, we observe that SE-Sync-S (solving (24)) consistently outperforms SE-Sync-T (solving (26)) by a considerable margin across all problem instances, as predicted by our analysis in Section 4.1. Second, we observe that our general-purpose method Alg1-T appears to be very roughly comparable (within an order of magnitude) in terms of computational speed with the purpose-built solvers SE-Sync-S and SE-Sync-T on these examples. Indeed, Alg1-T (somewhat surprisingly) outperforms both SE-Sync-S

[10] Version 6.2, available at https://www.pardiso-project.org/

[11] Available at https://github.com/david-m-rosen/SE-Sync.

[12] This termination criterion only affects Alg1-T, since SE-Sync-S and SE-Sync-T employ Riemannian optimization methods that automatically enforce feasibility.

[13] Note that the sum of the optimization and minimum-eigenvalue computation times is *not* equal to the total elapsed time for SE-Sync-S and SE-Sync-T because these algorithms also construct and cache certain sparse matrix factorizations [33, 36].

	# Poses	# Measurements	SE-Sync-S			SE-Sync-T			Alg1-T		
			Opt [s]	Eig [s]	Total [s]	Opt [s]	Eig [s]	Total [s]	Opt [s]	Eig [s]	Total [s]
sphere	2500	4949	0.44	0.199	1.252	0.751	0.584	1.926	11.46	0.206	11.68
torus	5000	9048	0.371	0.411	2.185	0.585	0.313	2.239	28.56	0.484	29.06
garage	1661	6275	0.743	0.533	1.503	10.171	0.475	10.85	6.715	0.120	6.840
cubicle	5750	16869	0.742	1.171	3.806	4.401	4.037	10.137	26.16	0.754	26.93
rim	10195	29743	10.500	32.525	47.229	98.225	386.798	488.983	241.2	1.063	242.3

Table 2. Results for the 3D SLAM benchmark datasets.

and SE-Sync-T on most of the 2D examples in Table 1. Alg1-T's superior performance in the 2D cases is primarily due to its improved minimum-eigenvalue computation, which can take advantage of the sparsity of the data matrix $\tilde{M}$ in (26) (and hence S in (9)) to efficiently perform *inverse* Lanczos iterations using sparse linear system solves. In contrast, SE-Sync is restricted to employing less-efficient *forward* Lanczos iterations due to the density of $\tilde{Q}$ [33, Sec. III-C]; this computation generally dominates the running times of both SE-Sync-S and SE-Sync-T. The situation is reversed in the (higher-dimensional) 3D cases, where the specialized Riemannian optimizers implemented in SE-Sync-S and SE-Sync-T substantially outperform IPOPT, which dominates Alg1-T's running time.

Most importantly, in *all* cases we observe that Alg1-T preserves the tractable scaling that originally motivated the development of the specialized Riemannian low-rank semidefinite programming methods implemented in SE-Sync. Indeed, Alg1-T is efficient enough to process even the largest of these examples in real-time. This is in marked contrast to off-the-shelf interior-point semidefinite optimization methods, which require *hours* to process even toy instances of (26) involving only a few hundred poses [11]. Alg1-T thus affords the ease-of-use of existing general-purpose SDP solvers, while preserving the scalability of current state-of-the-art, purpose-built Riemannian low-rank methods.

5 Conclusion

In this paper we presented an efficient algorithm to solve the large-scale but *low-rank* semidefinite relaxations that underpin current certifiably correct machine perception methods [6, 7, 16, 18, 29, 35, 36]. Our algorithm preserves the scalability of current state-of-the-art, *problem-specific* Riemannian low-rank semidefinite optimizers, but applies to a broad class of convex programs over spectrahedral sets *without* the need for detailed manual analysis or design. The result is an easy-to-use, general-purpose computational tool capable of supporting the development and deployment of a broad class of novel certifiably correct machine perception methods.

Bibliography

[1] K.J. Åström and R.M. Murray. *Feedback Systems: An Introduction for Scientists and Engineers.* Princeton University Press, Princeton, NJ, USA, 2008.

[2] P.-A. Absil, R. Mahony, and R. Sepulchre. *Optimization Algorithms on Matrix Manifolds.* Princeton University Press, 2008.

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

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

[5] S. Boyd and L. Vandenberghe. *Convex Optimization.* Cambridge University Press, 2004.

[6] J. Briales and J. Gonzalez-Jimenez. Cartan-Sync: Fast and global SE(d)-synchronization. *Robotics and Automation Letters*, 2(4):2127–2134, October 2017.

[7] J. Briales, L. Kneip, and J. Gonzalez-Jimenez. A certifiably globally optimal solution to the non-minimal relative pose problem. In *IEEE Intl. Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 145–154, 2018.

[8] S. Burer and R.D.C. Monteiro. A nonlinear programming algorithm for solving semidefinite programs via low-rank factorization. *Math. Program.*, 95:329–357, 2003.

[9] S. Burer and R.D.C. Monteiro. Local minima and convergence in low-rank semidefinite programming. *Math. Program.*, 103:427–444, 2005.

[10] C. Cadena, L. Carlone, H. Carrillo, Y. Latif, D. Scaramuzza, J. Neira, I. Reid, and J.J. Leonard. Past, present, and future of simultaneous localization and mapping: Toward the robust-perception age. *IEEE Trans. on Robotics*, 32(6):1309–1332, December 2016.

[11] L. Carlone, D.M. Rosen, G. Calafiore, J.J. Leonard, and F. Dellaert. Lagrangian duality in 3D SLAM: Verification techniques and optimal solutions. In *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, Hamburg, Germany, September 2015.

[12] L. Carlone, R. Tron, K. Daniilidis, and F. Dellaert. Initialization techniques for 3D SLAM: A survey on rotation estimation and its use in pose graph optimization. In *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, pages 4597–4605, Seattle, WA, May 2015.

[13] D. Cifuentes. On the Burer-Monteiro method for general semidefinite programs. *arXiv preprint arXiv:1904.07147*, 2019.

[14] D. Cifuentes and A. Moitra. Polynomial time guarantees for the Burer-Monteiro method. *arXiv preprint arXiv:1912.01745*, 2019.

[15] A.R. Conn, N.I.M. Gould, and Ph.L. Toint. *Trust-Region Methods.* The Society for Industrial and Applied Mathematics (SIAM), 2000.

[16] A. Eriksson, C. Olsson, F. Kahl, and T.-J. Chin. Rotation averaging and strong duality. In *IEEE Intl. Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 127–135, Salt Lake City, UT, USA, June 2018.

[17] M. Fallon, S. Kuindersma, S. Karumanchi, M. Antone, T. Schneider, H. Dai, C.P. D'Arpino, R. Deits, M. DiCicco, D. Fourie, T. Koolen, P. Marion, M. Posa, A. Valenzuela, K.-T. Yu, J. Shah, K. Iagnemma, R. Tedrake, and S. Teller. An architecture for online affordance-based perception and whole-body planning. *Intl. J. of Robotics Research*, 32(2):229–254, 2015.

[18] T. Fan, H. Wang, M. Rubenstein, and T. Murphey. Efficient and guaranteed planar pose graph optimization using the complex number representation. In *IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS)*, pages 1904–1911, Macau, China, November 2019.

[19] J. Gallier. The Schur complement and symmetric positive semidefinite (and definite) matrices. unpublished note, available online: http://www.cis.upenn.edu/ jean/schur-comp.pdf, December 2010.

[20] G. Grisetti, R. Kümmerle, C. Stachniss, and W. Burgard. A tutorial on graph-based SLAM. *IEEE Intelligent Transportation Systems Magazine*, 2 (4):31–43, 2010.

[21] S. Hu and L. Carlone. Accelerated inference in Markov random fields via smooth Riemannian optimization. *Robotics and Automation Letters*, 4(2): 1295–1302, 2019.

[22] M. Johnson, B. Shrewsbury, S. Bertrand, T. Wu, D. Duran, M. Floyd, P. Abeles, D. Stephen, N. Mertins, A. Lesman, J. Carff, W. Rifenburgh, P. Kaveti, W. Straatman, J. Smith, M. Griffioen, B. Layton, T. de Boer, T. Koolen, P. Neuhaus, and J. Pratt. Team IHMC's lessons learned from the DARPA Robotics Challenge Trials. *Intl. J. of Robotics Research*, 32(2): 192–208, 2015.

[23] M. Journée, F. Bach, P.-A. Absil, and R. Sepulchre. Low-rank optimization on the cone of positive semidefinite matrices. *SIAM J. Optim.*, 20(5):2327–2351, 2010.

[24] M. Kaess, H. Johannsson, R. Roberts, V. Ila, J.J. Leonard, and F. Dellaert. iSAM2: Incremental smoothing and mapping using the Bayes tree. *Intl. J. of Robotics Research*, 31(2):216–235, February 2012.

[25] R. Kümmerle, G. Grisetti, H. Strasdat, K. Konolige, and W. Burgard. g2o: A general framework for graph optimization. In *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, pages 3607–3613, Shanghai, China, May 2011.

[26] J. Leonard, J. How, S. Teller, M. Berger, S. Campbell, G. Fiore, L. Fletcher, E. Frazzoli, A. Huang, S. Karaman, O. Koch, Y. Kuwata, D. Moore, E. Olson, S. Peters, J. Teo, R. Truax, M. Walter, D. Barrett, A. Epstein, K. Maheloni, K. Moyer, T. Jones, R. Buckley, M. Antone, R. Galejs, S. Krishnamurthy, and J. Williams. A perception-driven autonomous urban vehicle. *Intl. J. of Robotics Research*, 25(10):727–774, 2008.

[27] A. Majumdar, G. Hall, and A.A. Ahmadi. Recent scalability improvements for semidefinite programming with applications in machine learning, control and robotics. *Annu. Rev. Control Robot. Auton. Syst.*, 3:331–360, 2020.

[28] J. Mandel. On block diagonal and Schur complement preconditioning. *Numerische Mathematik*, 58(1):79–93, December 1990.

[29] J.G. Mangelson, J. Liu, R.M. Eustice, and R. Vasudevan. Guaranteed globally optimal planar pose graph and landmark SLAM via sparse-bounded sums-of-squares programming. In *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, pages 9306–9312, Montreal, Canada, May 2019.

[30] J. Nocedal and S.J. Wright. *Numerical Optimization*. Springer Science+Business Media, New York, 2nd edition, 2006.

[31] J.T. Rockafellar and R.J.-B. Wets. *Variational Analysis*. Springer Science+Business Media, 2009.

[32] D.M. Rosen. Towards provably robust machine perception. Presented at Robotics: Science and Systems (RSS) in the workshop "RSS Pioneers", June 2019.

[33] D.M. Rosen and L. Carlone. Computational enhancements for certifiably correct SLAM. Presented at the International Conference on Intelligent Robots and Systems (IROS) in the workshop "Introspective Methods for Reliable Autonomy", September 2017.

[34] D.M. Rosen, M. Kaess, and J.J. Leonard. RISE: An incremental trust-region method for robust online sparse least-squares estimation. *IEEE Trans. on Robotics*, 30(5):1091–1108, October 2014.

[35] D.M. Rosen, L. Carlone, A.S. Bandeira, and J.J. Leonard. A certifiably correct algorithm for synchronization over the special Euclidean group. In *Intl. Workshop on the Algorithmic Foundations of Robotics (WAFR)*, San Francisco, CA, December 2016.

[36] 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. *Intl. J. of Robotics Research*, 38(2–3):95–125, March 2019.

[37] S. Russel and P. Norvig. *Artifical Intelligence: A Modern Approach*. Prentice Hall, Upper Saddle River, NJ, USA, 3rd edition, 2010.

[38] R.F. Stengel. *Optimal Control and Estimation*. Dover Publications, 1994.

[39] R.H. Taylor. A perspective on medical robotics. *Proc. of the IEEE*, 94(9): 1652–1664, September 2006.

[40] M.J. Todd. Semidefinite optimization. *Acta Numerica*, 10:515–560, 2001.

[41] B. Triggs, P. McLauchlan, R. Hartley, and A. Fitzgibbon. Bundle adjustment – a modern synthesis. In W. Triggs, A. Zisserman, and R. Szeliski, editors, *Vision Algorithms: Theory and Practice*, volume 1883 of *Lecture Notes in Computer Science*, pages 298–372. Springer Verlag, 2000.

[42] L. Vandenberghe and S. Boyd. Semidefinite programming. *SIAM Review*, 38(1):49–95, March 1996.

[43] B. Vlasic and N.E. Boudette. As U.S. investigates fatal Tesla crash, company defends Autopilot system. *The New York Times*, July 12, 2016.

[44] W. Wachsmuth. On LICQ and the uniqueness of Lagrange multipliers. *Operations Research Letters*, 41:78–80, 2013.

[45] A. Wächter and L.T. Biegler. On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming. *Math. Program.*, 106:25–57, 2006.