Polynomial optimization in geometric modeling

Soodeh Habibi, Michal Kočvara, Bernard Mourrain

Polynomial optimization in geometric modeling

Soodeh Habibi, Michal Kočvara and Bernard Mourrain

Abstract In this chapter, we review applications of Polynomial Optimization techniques to Geometric Modeling problems. We present examples of topical problems in Geometric Modeling, illustrate their solution using Polynomial Optimization Tools, report some experimental results and analyse the behavior of the methods, showing what are their strengths and their limitations.

1 Geometric Modeling and Polynomials

Geometric modeling aims at describing shapes of the real world by digital models. These digital models are used in many domains: visualization in scientific computing, rendering and animation, design and conception, manufacturing, numerical simulation, analysis of physics phenomena, mechanics, performance optimization . . .

Most of the digital models used nowadays in Computer Aided Design (CAD) involve models based on spline representations, which are piecewise polynomial functions with global regularity properties [5]. Among them, the representation or approximation of shapes by meshes is certainly the most popular and corresponds to piecewise linear splines. Higher order splines allow to increase the accuracy of

Soodeh Habibi
School of Mathematics, University of Birmingham, Birmingham, B15 2TT, UK,
e-mail: s.habibi@bham.ac.uk

Michal Kočvara
School of Mathematics, University of Birmingham, Birmingham, B15 2TT, UK, and
Institute of Information Theory and Automation, Czech Academy of Sciences, Pod vodárenskou věží 4, 18208 Praha 8, Czech Republic
e-mail: m.kocvara@bham.ac.uk

Bernard Mourrain
Inria at Université Côte d'Azur, 2004 route des Lucioles, 06902 Sophia Antipolis, France, e-mail: Bernard.Mourrain@inria.fr

the representation and the efficiency to describe complex shapes. By increasing the degree of the polynomials of the representation, one needs less pieces and thus less data for the same level of approximation. The “unreasonable" power of polynomial approximation makes piecewise polynomial representations ubiquitous in Geometric Modeling. Standard digital representations of shapes correspond to the image of simple domains like triangles, squares, cubes by piecewise polynomial maps, which define so called patches. These patches are trimmed and assembled together to define the boundary surface or the volume of an object.

Although piecewise polynomial models are very effective in representing complex shapes, they also require advanced methods and tools in practice. Many operations on shapes, such as patch intersections, distance computation, boundary volumes, etc., involve solving nonlinear and difficult problems [4]. In what follows, we illustrate how polynomial optimization methods can help solve these issues and discuss their practical performance. See also [9] for other examples of applications of polynomial optimization in geometric modeling.

2 Polynomial Optimization Problems and Convex Relaxations

Polynomial Optimization Problems (POP) are problems of the form

$$\min_{\mathbf{x}\in S} f(\mathbf{x}) \tag{1}$$

where

$$S = \{\mathbf{x} \in \mathbb{R}^n \text{ s.t.} \quad g_j(\mathbf{x}) \geq 0,\ j = 1,\ldots,m_I, \quad h_k(\mathbf{x}) = 0,\ k = 1,\ldots,m_E\}$$

is the *semi-algebraic set* defined by the sign constraints $\mathbf{g} = (g_1,\ldots,g_{m_I})$ and the equality constraints $\mathbf{h} = (h_1,\ldots,h_{m_E})$, for polynomial functions $f, g_j, h_k : \mathbb{R}^n \to \mathbb{R}$, $j = 1,\ldots,m_I$, $k = 1,\ldots,m_E$. Problem (1) is a special instance of nonlinear nonconvex optimization. In the following, we will discuss how these problems are translated into Semidefinite Programming (SDP) problems.

There are various approaches to solving polynomial optimization problems, such as using a hierarchy of convex (semidefinite) relaxations to approximate (1). Such relaxations can be built using two methods: the SoS representation of nonnegative polynomials and the dual theory of moments. The general approach started with the work of Shor [14] and Nesterov [11]. Then, it was further developed by Lasserre [8] and Parrilo [12]. Here, we describe the two dual approaches to this development. Each will give additional complementary information about the problem.

2.1 Sum of Squares Relaxations

Sum-of-Squares (SoS) relaxation is a type of convex relaxation in which polynomial non-negativity constraints are replaced by SoS constraints which are more computationally tractable, and can be represented by semidefinite programs. This relaxation transforms nonlinear polynomial optimization problems into sequences of convex problems whose complexity is captured by a single degree parameter. To approximate the solutions of (1), it uses the following finite dimensional convex cones, also called *truncated quadratic modules*,

$$Q_\ell = \{p = s_0 + s_1 g_1 + s_{m_I} g_{m_I} + t_1 h_1 + \cdots + t_{m_E} h_{m_E}, s_j \in \Sigma^2_{2\,\ell - d_j}, t_k \in \mathbb{R}[\mathbf{x}]_{2\,\ell - d'_k}\},$$

where

- g_j are the non-negativity polynomials of degree d_j and h_k are the equality polynomials of degree d'_k (we take $g_0 = 1$ and $d_0 = 0$ for notation convenience),
- $\Sigma^2_\ell = \{p = \sum_i q_i^2,\ q_i \in \mathbb{R}[\mathbf{x}]_{\frac{\ell}{2}}\}$ is the convex cone of polynomials of degree $\leq \ell$, which are sums of squares,
- $\mathbb{R}[\mathbf{x}]_\ell$ is the vector space of polynomials of degree $\leq \ell$ in the variables $\mathbf{x} = (x_1, \ldots, x_n)$. It is of dimension $s(\ell) = \binom{n+\lfloor \ell \rfloor}{n}$.We will denote the dual to $\mathbb{R}[\mathbf{x}]_\ell$ by $\mathbb{R}[\mathbf{x}]^*_\ell$.

We verify that the truncated quadratic module Q_ℓ is a convex cone since it is stable by scaling by a positive scalar and by addition. By construction, the polynomials in Q_ℓ are non-negative on S.

We approximate the solution of (1) by the solution of the following convex optimization problem:

$$\begin{aligned} f^{\wedge,\ell} = \sup\ & \lambda \\ \text{s.t.}\quad & \lambda \in \mathbb{R} \\ & f(\mathbf{x}) - \lambda \in Q_\ell\,. \end{aligned} \tag{2}$$

For ℓ big enough, this problem is feasible. We check that if $f(\mathbf{x}) - \lambda \in Q_\ell$ then $\forall \mathbf{x} \in S, f(\mathbf{x}) - \lambda \geq 0$ and $f^* \geq \lambda$. This shows that $f^{\wedge,\ell} \leq f^*$ for all $\ell \in \mathbb{N}$. Under some conditions on $\mathbf{g}, \mathbf{h}$ (see e.g. [8]), we have $\lim_{\ell\to\infty} f^{\wedge,\ell} = f^*$.

The convex cones Q_ℓ are tractable, since they involve sums of multiples of sum-of-squares cones $\Sigma^2_{2\ell}$ and multiples of linear spaces $\mathbb{R}[\mathbf{x}]_{2\ell-d'_k}$. Problem (2) is a tractable semidefinite program that can be solved by classical convex optimization techniques, such as interior point methods [3, 16].

Let $\mathbf{v}_j$ denote a basis of $\mathbb{R}[\mathbf{x}]_{\frac{2\ell-d_j}{2}}$ and $\mathbf{w}_k$ a basis of $\mathbb{R}[\mathbf{x}]_{2\,\ell-d'_k}$. Then (2) is implemented as a semidefinite program of the form

$$
\begin{aligned}
f^{\wedge,\ell} = \sup\ & \lambda \\
\text{s.t.}\quad & \lambda \in \mathbb{R},\ A_j \succcurlyeq 0,\ B_k \in \mathbb{R}^{N_K} \\
& f(\mathbf{x}) - \lambda - \sum_{j=1}^{m_I} g_j(\mathbf{x}) \mathbf{v}_j^T A_j \mathbf{v}_j - \sum_{k=1}^{m_E} h_k(\mathbf{x}) \mathbf{w}_k^T B_k = 0\,,
\end{aligned}
$$

where $A_j \in \mathbb{R}^{s_j \times s_j}$ for $s_j = s(\ell - \frac{d_j}{2})$ is *positive semidefinite*, i.e. $A_j \succcurlyeq 0$, if for any vector $\mathbf{v} \in \mathbb{R}^{s \times s}$ $\mathbf{v}^T A_j \mathbf{v} \geq 0$. The last polynomial constraint can be written as

$$
\sum_{|\alpha| \leq 2\,\ell} c_\alpha(\lambda, A_1, \ldots, A_{m_I}, B_1, \ldots, B_{m_E}) \mathbf{x}^\alpha = 0
$$

where $(\mathbf{x}^\alpha)_{|\alpha| \leq 2\,\ell}$ is the monomial basis of $\mathbb{R}[\mathbf{x}]_\ell$. It corresponds to a sequence of coefficient constraints $c_\alpha(\lambda, \mathbf{A}, \mathbf{B}) = 0$, which are linear in λ, $\mathbf{A} = (A_1, \ldots, A_{m_I})$, $\mathbf{B} = (B_1, \ldots, B_{m_E})$.

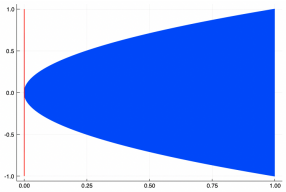
We verify that $Q_\ell \subset Q_{\ell+1}$ so that (Q_ℓ) is a hierarchy of nested convex finite dimensional cones.

Example of a Sum-of-Squares relaxation

We consider the semi-algebraic set S defined by

$$
\begin{aligned}
\mathbf{g} &= \{1 - x, x - y^2\} \\
\mathbf{h} &= \{\}
\end{aligned}
$$

corresponding to the blue domain on the adjacent figure. The objective function is $f = x$, corresponding to the vertical red line.



We construct a SoS relaxation at order $\ell = 2$. Here is how it can be constructed with the Julia packages `MomentTools`[1] and `DynamicPolynomials`.

```
> using MomentTools, DynamicPolynomials
> X = @polyvar x y
> M = SOS.Model(:inf, x, [], [1-x, x-y^2], X, 2)
```

This gives a semidefinite program with three matrix variables $A_0 \in \mathbb{R}^{6\times 6}$ for $s_0 \in \Sigma_4^2$, $A_1 \in \mathbb{R}^{3\times 3}$ for $s_1 \in \Sigma_2^2$, $A_2 \in \mathbb{R}^{3\times 3}$ for $s_2 \in \Sigma_1^2$ and no B_k since $\mathbf{h} = \{\}$. There are 15 linear constraints corresponding to the coefficients of the 15 monomials of degree ≤ 4 in the variables x, y.

[1] https://gitlab.inria.fr/AlgebraicGeometricModeling/MomentTools.jl

2.2 Moment Relaxations

The dual formulation of (2) is

$$\begin{aligned} f^{\vee,\ell} = \inf\ & \Lambda(f) \\ \text{s.t.}\quad & \Lambda(1) - 1 \in \mathbb{R}^{\vee} = \{0\} \\ & \Lambda \in \mathcal{L}_\ell := (Q_\ell)^{\vee} \end{aligned} \tag{3}$$

where

$$\mathcal{L}_\ell = (Q_\ell)^{\vee} = \{\Lambda \in \mathbb{R}[\mathbf{x}]^*_{2\ell} \mid \forall p \in Q_\ell, \Lambda(p) \geq 0\}$$

is the dual cone of Q_ℓ and $\mathbb{R}^{\vee} = \{\sigma : \mathbb{R} \to \mathbb{R} \mid \forall x \in \mathbb{R}, \sigma(x) \geq 0\} = \{0\}$. The elements $\Lambda \in \mathcal{L}_\ell$ are linear functionals $\Lambda : \mathbb{R}[\mathbf{x}]_{2\,\ell} \to \mathbb{R}$, represented in the dual basis $\mathbf{x}^\alpha$ of $\mathbb{R}[\mathbf{x}]_{2\,\ell}$ by the coefficient vector $(\Lambda_\alpha)_{|\alpha|\leq 2\,\ell}$. The coefficients $\Lambda_\alpha := \Lambda(\mathbf{x}^\alpha)$ are called the *pseudo-moments* of Λ. These coefficients correspond to the *slack variables* associated to the linear constraints $c_\alpha(\lambda, \mathbf{A}, \mathbf{B})$ in (2).

The constraint $\Lambda \in \mathcal{L}_\ell$ translates into the conditions

$$\forall s_0 \in \Sigma^2_{2\,\ell}, \quad \Lambda(s_0) \geq 0, \tag{4}$$

$$\forall s_j \in \Sigma^2_{2\,\ell - d_j}, \quad \Lambda(g_j\, s_j) \geq 0 \text{ for } j = 1, \ldots, m_I, \tag{5}$$

$$\forall t_k \in \mathbb{R}[\mathbf{x}]_{2\,\ell - d'_k}, \quad \Lambda(h_k\, t_k) = 0 \text{ for } j = 1, \ldots, m_E. \tag{6}$$

The first two types of constraints (4), (5) correspond to semidefinite constraints

$$H_0(\Lambda) \succcurlyeq 0, \quad H_{g_j}(\Lambda) \succcurlyeq 0, \text{ for } j = 1, \ldots, m_I,$$

where

- $H_0(\Lambda) = (\Lambda(\mathbf{x}^{\alpha+\beta}))_{|\alpha|,|\beta|\leq\ell}$ is called the *moment matrix* of Λ in degree (ℓ, ℓ);
- $H_{g_j}(\Lambda) = (\Lambda(g_j\, \mathbf{x}^{\alpha+\beta}))_{|\alpha|,|\beta|\leq\ell-\frac{d_j}{2}}$ is called the *localizing moment matrix* of Λ at g_j in degree $\ell - \frac{d_j}{2}$.

We denote by $H^{\ell,\ell'}(\Lambda) = (\mathbf{x}^{\alpha+\beta}))_{|\alpha|\leq\ell,|\beta|\leq\ell'}$ the moment matrix of Λ in degree (ℓ, ℓ').

The third type of constraints (6) corresponds to linear constraints on Λ of the form

$$\Lambda(h_k \mathbf{x}^\alpha) = 0 \quad \text{for } |\alpha| \leq 2\,\ell - d'_k, \quad k = 1, \ldots, m_E\,.$$

Therefore, (3) is also a tractable semidefinite program.

As the evaluation $\mathbf{e}_{\boldsymbol{\xi}} : p \in \mathbb{R}[\mathbf{x}]_{2\ell} \mapsto p(\boldsymbol{\xi})$ at a point $\boldsymbol{\xi} \in S$ is an element in $\mathcal{L}_\ell$ such that $\mathbf{e}_{\boldsymbol{\xi}}(1) = 1$, we have $f^{\vee,\ell} \leq f(\boldsymbol{\xi})$ for $\boldsymbol{\xi} \in S$. This implies that

$$f^{\vee,\ell} \leq \inf_{\boldsymbol{\xi}\in S} f(\boldsymbol{\xi}) = f^*.$$

We verify that for $\Lambda \in \mathcal{L}_\ell$ with $\Lambda(1) = 1$ and $\lambda \in \mathbb{R}$ such that $f - \lambda \in Q_\ell$, we have $\Lambda(f) - \lambda \geq 0$ since $\mathcal{L}_\ell = (Q_\ell)^\vee$, so that $f^{\wedge,\ell} \leq f^{\vee,\ell} \leq f^*$. Under certain conditions on $\mathbf{g}, \mathbf{h}$ (see [8]), we also have $\lim_{\ell\to\infty} f^{\vee,\ell} = f^*$.

Example of a Moment Relaxation

The moment relaxation at order $\ell = 2$ for the previous example where $S = \{(x, y) \in \mathbb{R}^2 \mid 1 - x \geq 0, x - y^2 \geq 0\}$ and the objective function is $f = x$ is built as follows:

```
> using MomentTools, DynamicPolynomials
> X = @polyvar x y
> M = MOM.Model(:inf, x, [], [1-x, x-y^2], X, 2)
```

It is a convex optimization program on the moment sequence $\Lambda \in \mathbb{R}^{15}$ with a linear objective function $\Lambda(f)$ (as a function of Λ) and with SDP constraints on moment matrices in $\mathbb{R}^{6\times6}, \mathbb{R}^{3\times3}, \mathbb{R}^{3\times3}$.

2.3 Computing the minimizers

The solution of the dual convex optimization problem (3) provides an optimal sequence of pseudo-moments $\Lambda^* = (\Lambda^*_\alpha)$, from which approximations of the optimizers of the non-linear optimization problem (1) can be recovered under some conditions. This can be done as follows. Assume that the set $\{\xi_1, \ldots, \xi_r\}$ of minimizers of (1) is finite and that Λ^* is numerically close to the moment sequence of a weighted sum $\mu = \sum_{i=1}^r \omega_i \delta_{\xi_i}$ of Dirac measures δ_{ξ_i} at the minimizers, with $\omega_i > 0$ and $\sum_{i=1}^r \omega_i = 1$. This is the case for a sufficiently large order ℓ of the moment relaxation, by the convergence properties of the moment hierarchy [8, 13, 1].

1. We form the moment matrix H^* of Λ^* in degree $(\ell - 1, \ell)$.
2. We compute a Singular Value Decomposition (SVD) of $H^* = USV^T$, where U and V are orthogonal matrices and S is diagonal, and deduce the numerical rank r of H^*. Let $U^{[r]}$ denote the first r columns of U.
3. We extract from $U^{[r]}$ an invertible block U_0 of r rows corresponding to a monomial set $\mathbf{b} = \{b_1, \ldots, b_r\}$ of low degree. We compute the matrices U_i corresponding to the rows associated to the monomials $x_i \cdot \mathbf{b}$ in $U^{[r]}$.
4. We compute the common eigenvectors of $M_i = U_0^{-1} U_i$. We deduce the points $\xi_1, \ldots, \xi_r$, whose j^{th} coordinate is the eigenvalue of M_j for the eigenvector associated to ξ_i.

For more details on the algorithm, see [7, 10]. Notice that in the construction of H^* in step 1, we use the pseudo-moments of Λ^* up to degree $2\,\ell - 1$.

We illustrate the behavior of this approach on different geometric problems, using the package `MomentTools`[2].

[2] https://gitlab.inria.fr/AlgebraicGeometricModeling/MomentTools.jl

Example of minimizer computation

We continue with the previous example, and solve the moment relaxation at order $\ell = 2$, using convex optimization tools from the `Mosek`[3] library,

```
> using MomentTools, DynamicPolynomials, MosekTools
> X = @polyvar x y
> v, M = minimize(x, [], [1-x, x-y^2], X, 2, Mosek.Optimizer)
```

we obtain $v = -6.371168130759666 \cdot 10^{-10}$. The moment matrix H^* of the optimal pseudo-moment sequence Λ^* computed as

```
> s = get_series(M)[1]; L1 = monomials(X, 0:2); L2 = monomials(X, 0:1)
> H = MultivariateSeries.hankel(s, L1, L2);
```

is the following matrix (rounded with 2 decimal digits).

$$H^* = \begin{bmatrix} 1.0 & 0.0 & -0.0 & 0.0 & -0.0 & 0.0 \\ 0.0 & 0.0 & -0.0 & 0.0 & -0.0 & 0.0 \\ -0.0 & -0.0 & 0.0 & -0.0 & 0.0 & -0.0 \end{bmatrix}^T .$$

The singular values of H^* give a numerical rank $r = 1$ and the extracted matrix $U^{[1]}$ is

$$U^{[1]} = \begin{bmatrix} -1.0 & -0.0 & 0.0 & -0.0 & 0.0 & -0.0 \end{bmatrix}^T$$

with entries indexed by the monomials $[1, x, y, x^2, x\,y, y^2]$. We have $U_0 = [1]$ indexed by $\mathbf{b} = \{1\}$ and $U_1 = [0.0]$ indexed by $x \cdot \mathbf{b} = \{x\}$, $U_2 = [0.0]$ indexed by $y \cdot \mathbf{b} = \{y\}$. The eigenvector and eigenvalue computation of $M_1 = U_0^{-1} U_1 = [0.0]$ and $M_2 = U_0^{-1} U_1 = [0.0]$ gives the unique minimizer $\xi = (0.0, 0.0)$. This minimizer computation can be done directly as follows:

```
w, Xi = get_measure(M)
```

which yields the following weight ω and point Ξ for the approximation of Λ^* as a weighted sum of Dirac measures:

$$\omega = \begin{bmatrix} 1.0 \end{bmatrix}, \qquad \Xi = \begin{bmatrix} 0.0 \\ -0.0 \end{bmatrix}.$$

Solution of linear semidefinite optimization problems resulting from SoS or Moment relaxations is often the bottleneck of the approach. With increasing order of the relaxation, the size of the SDP problem grows very quickly. While the "small" problems can be solved by general-purpose SDP solvers such as `Mosek`, larger problems are beyond their reach. It is therefore necessary to use an algorithm and software exploiting the particular structure of SoS relaxations. One of the features of these SDP problems is the very low rank of the solution (moment matrix). `Loraine` [6] is a new general-purpose interior-point SDP solver targeted to problems with *low-rank data* and *low-rank solutions*. It employs special treatment of low-rank data matrices and, in particular, an option to use a preconditioned iterative Krylov type

[3] https://github.com/MOSEK/Mosek.jl

method for the solution of the linear system. The used preconditioner is tailored to problems with low-rank solutions and proved to be rather efficient for these problems; for more details, see [6].

3 Minimal enclosing ellipsoids of semi-algebraic sets

In this section, we consider another type of optimization problems, which appears in geometric modeling, namely computing the minimal ellipsoid enclosing a semi-algebraic set. Let S be a semi-algebraic set defined as in Section 2:

$$S = \{\mathbf{x} \in \mathbb{R}^n \text{ s.t. } \quad g_j(\mathbf{x}) \geq 0,\ j = 1, \ldots, m_I, \quad h_k(\mathbf{x}) = 0,\ k = 1, \ldots, m_E\}\,.$$

Let $\xi_i(\mathbf{x}),\ i = 1, \ldots, \nu$, be given polynomial functions. Define the set

$$\hat{S} = \{\boldsymbol{\eta} \in \mathbb{R}^\nu \text{ s.t. } \quad \eta_i = \xi_i(\mathbf{x}),\ i = 1, \ldots, \nu, \quad \mathbf{x} \in S\}\,.$$

This set typically represents a parametric subset of $\mathbb{R}^n$ described by parameters x; see the next section.

Assume that $\hat{S}$ is bounded in $\mathbb{R}^\nu$. We are looking for a smallest-volume ("minimal") ellipsoid E that is enclosing the set $\hat{S}$, i.e., all feasible positions of the ν-dimensional point $\boldsymbol{\xi}(\mathbf{x}) = (\xi_1(\mathbf{x}), \ldots, \xi_\nu(\mathbf{x}))$.

It is well-known that finding a minimal enclosing ellipsoid of a set of points $S_P = \{\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(m)}\} \subset \mathbb{R}^\nu$ amounts to solving a semidefinite optimization problem; see, e.g., [2, 15]. Let us first recall this formulation.

Assume that the convex hull of S_P has a nonempty interior. We consider an $n-$dimensional ellipsoid represented by a strictly convex quadratic inequality with $D \in \mathbb{S}^\nu, D \succ 0$:

$$E = \{\mathbf{y} \in \mathbb{R}^\nu \mid (\mathbf{y} - \mathbf{c})^T D(\mathbf{y} - \mathbf{c}) \leq 1\}\,.$$

The volume of such an ellipsoid is

$$\mathrm{Vol}(E) = (\det(D))^{-\frac{1}{2}}\,,$$

so minimizing it amounts to maximizing $\det(D)$ and, further, to maximizing $\log\det(D)$, as D is assumed to be positive definite.

The minimal ellipsoid enclosing the set S_P can be computed from the solution of the following convex semidefinite optimization problem

$$\begin{aligned} \sup_{Z \in \mathbb{S}^\nu,\, \mathbf{z} \in \mathbb{R}^\nu,\, \gamma \in \mathbb{R}} \quad & \log\det(Z) \\ \text{s.t.} \quad & 1 - (\mathbf{x}^{(i)})^T Z \mathbf{x}^{(i)} + 2\mathbf{z}^T \mathbf{x}^{(i)} - \gamma \geq 0, \quad i = 1, \ldots, m\,, \\ & \begin{bmatrix} \gamma & \mathbf{z}^T \\ \mathbf{z} & Z \end{bmatrix} \succcurlyeq 0\,, \end{aligned} \tag{7}$$

where $\gamma = \mathbf{c}^T D\mathbf{c}$. Let $(Z^*, \mathbf{z}^*, \gamma^*)$ be the solution of the above problem, then the minimal ellipsoid containing S_P is given by

$$E = \{\mathbf{y} \in \mathbb{R}^n \mid (\mathbf{y} - \mathbf{c})^T D(\mathbf{y} - \mathbf{c}) \leq 1\}$$

where $D = Z^*$ and $\mathbf{c} = D^{-1}\mathbf{z}^*$ (see, e.g., [2, Prop. 4.9.2]).

Now, to find the minimal enclosing ellipsoid of the semi-algebraic set $\hat{S}$, we generalize the above problem and solve, to global optimality, the following *polynomial* SDP problem

$$\begin{aligned} \sup_{Z\in\mathbb{S}^\nu,\, \mathbf{z}\in\mathbb{R}^\nu,\, \gamma\in\mathbb{R},\, \mathbf{x}\in\mathbb{R}^n} \quad & \log\det(Z) \\ \text{s.t.} \quad & 1 - \boldsymbol{\xi}(\mathbf{x})^T Z \boldsymbol{\xi}(\mathbf{x}) + 2\mathbf{z}^T \boldsymbol{\xi}(\mathbf{x}) - \gamma \geq 0\,, \\ & \begin{bmatrix} \gamma & \mathbf{z}^T \\ \mathbf{z} & Z \end{bmatrix} \succcurlyeq 0\,, \\ & g_j(\mathbf{x}) \geq 0, \quad j = 1, \ldots, m_I\,, \\ & h_k(\mathbf{x}) = 0, \quad k = 1, \ldots, m_E\,. \end{aligned} \tag{8}$$

The relaxation of order ℓ reads as

$$\begin{aligned} \sup_{\substack{Z\in\mathbb{S}^\nu,\, \mathbf{z}\in\mathbb{R}^\nu,\, \gamma\in\mathbb{R} \\ \{\sigma_j\},\, \{\psi_k\}}} \quad & \log\det(Z) \\ \text{s.t.} \quad & 1 - \boldsymbol{\xi}(\mathbf{x})^T Z \boldsymbol{\xi}(\mathbf{x}) + 2\mathbf{z}^T \boldsymbol{\xi}(\mathbf{x}) - \gamma \\ & \qquad - \sum_{j=1}^{m_I} \sigma_j(\mathbf{x}) g_j(\mathbf{x}) - \sum_{k=1}^{m_E} \psi_k(\mathbf{x}) h_k(\mathbf{x}) \in \Sigma^2_\ell\,, \\ & \begin{bmatrix} \gamma & \mathbf{z}^T \\ \mathbf{z} & Z \end{bmatrix} \succcurlyeq 0\,, \\ & \sigma_j \in \Sigma^2_{\frac{\ell - d_j}{2}}, \quad j = 1, \ldots, m_I\,, \\ & \psi_k \in \mathbb{R}[\mathbf{x}]_{\ell - d'_k}, \quad k = 1, \ldots, m_E\,. \end{aligned} \tag{9}$$

To implement this convex program, we replace the objective function $\log\det(Z)$ by t for a point $(t, 1, Z)$ in the *log-det cone*:

$$\mathcal{K} = \{(t, s, Z) \in \mathbb{R} \times \mathbb{R} \times \mathbb{R}^{\nu\times\nu} \mid s \log\det(Z/s) \geq t, Z \succcurlyeq 0, s > 0\}.$$

Using the fact that every SoS polynomial can be represented by a positive semidefinite matrix, we can rewrite (9) as the following convex SDP with linear matrix inequalities:

$$\max_{\substack{Z\in\mathbb{S}^\nu,\,\mathbf{z}\in\mathbb{R}^\nu,\,\gamma\in\mathbb{R}\\ X_j\in\mathbb{S}_+^{s\left(\frac{\ell-d_j}{2}\right)},\,Y_k\in\mathbb{S}^{s\left(\ell-d'_k\right)}}} \quad \log\det(Z) \tag{10}$$

$$\begin{aligned}
\text{s.t.}\quad & 1-\left(\boldsymbol{\xi}(\mathbf{x})^T Z\boldsymbol{\xi}(\mathbf{x})\right)_\alpha + 2\left(\mathbf{z}^T\boldsymbol{\xi}(\mathbf{x})\right)_\alpha - \gamma \\
&\qquad = \sum_{j=0}^{m_I}\left\langle X_j, C_\alpha^j\right\rangle + \sum_{k=1}^{m_E}\left\langle Y_k, D_\alpha^k\right\rangle, \qquad \text{for } \alpha = 0,\\
& -\left(\boldsymbol{\xi}(\mathbf{x})^T Z\boldsymbol{\xi}(\mathbf{x})\right)_\alpha + 2\left(\mathbf{z}^T\boldsymbol{\xi}(\mathbf{x})\right)_\alpha \\
&\qquad = \sum_{j=0}^{m_I}\left\langle X_j, C_\alpha^j\right\rangle + \sum_{k=1}^{m_E}\left\langle Y_k, D_\alpha^k\right\rangle, \qquad \text{for } |\alpha|\le\ell, \alpha\neq 0,\\
& \begin{bmatrix}\gamma & \mathbf{z}^T\\ \mathbf{z} & Z\end{bmatrix} \succcurlyeq 0,\\
& X_j \succcurlyeq 0, \quad j=0,\dots,m_I,
\end{aligned}$$

where $(\pi(\mathbf{x}))_\alpha$ is the α coefficient of the polynomial $\pi(\mathbf{x})$, $\langle A, B\rangle = \text{trace}(A^T B)$ for any matrices $A, B \in \mathbb{R}^{s\times s}$, C_α^j and D_α^k are matrices associated, respectively, with polynomials $g_j(\mathbf{x})$ ($g_0(\mathbf{x}) = 1$) and $h_k(\mathbf{x})$ via, $g_j(\mathbf{x})v_\ell(\mathbf{x})v_\ell(\mathbf{x})^T = \sum_\alpha C_\alpha^j \mathbf{x}^\alpha$ and $h_k(\mathbf{x})v_\ell(\mathbf{x})v_\ell(\mathbf{x})^T = \sum_\alpha D_\alpha^k \mathbf{x}^\alpha$ for some basis v_ℓ of $\mathbb{R}[\mathbf{x}]_\ell$, of size $s(\ell) = \binom{n+\ell}{n}$.

Denote by $\boldsymbol{\xi}_\alpha \in \mathbb{R}^\nu$, $\alpha : |\alpha| \le \ell$, vectors containing $\alpha-$coefficients of polynomials $\xi_i(\mathbf{x})$, $i = 1,\dots,\nu$. Then $(\boldsymbol{\xi}(\mathbf{x})\boldsymbol{\xi}(\mathbf{x})^T)_\alpha = \sum_{\beta+\gamma=\alpha} \boldsymbol{\xi}_\beta\boldsymbol{\xi}_\gamma^T$.

Theorem 0.1 *The dual problem to (10) reads as*

$$\inf_{\Psi\in\mathbb{S}^\nu,\,\boldsymbol{\lambda}\in\mathbb{R}^{d[\ell]}} \lambda_0 - \log\det(\Psi) - \nu \tag{11}$$

$$\begin{aligned}
\text{s.t.}\quad & \begin{bmatrix} \lambda_0 & \sum_{\alpha:|\alpha|\le 2\ell}\lambda_\alpha\boldsymbol{\xi}_\alpha^T \\ \sum_{\alpha:|\alpha|\le 2\ell}\lambda_\alpha\boldsymbol{\xi}_\alpha & \Psi - \sum_{\alpha:|\alpha|\le 2\ell}\lambda_\alpha\left(\sum_{\beta+\gamma=\alpha}\boldsymbol{\xi}_\beta\boldsymbol{\xi}_\gamma^T\right)\end{bmatrix} \succcurlyeq 0,\\
& \Psi \succcurlyeq 0,\\
& \sum_{\alpha:|\alpha|\le 2\ell}\lambda_\alpha C_\alpha^j \succcurlyeq 0, \quad j=0,\dots,m_I,\\
& \sum_{\alpha:|\alpha|\le 2\ell}\lambda_\alpha D_\alpha^k = 0, \quad k=1,\dots,m_E.
\end{aligned}$$

Proof The Lagrangian function for (10) can be written as

$$\mathcal{L}(Z,\mathbf{z},\gamma,X,Y;\boldsymbol{\lambda},\Xi,\Theta) = \log\det(Z)$$
$$+\sum_{\alpha:|\alpha|\le 2\ell}\lambda_\alpha\left(1_{|\alpha=0} - (\xi^T Z\xi)_\alpha + 2(\mathbf{z}^T\xi)_\alpha - \gamma_{|\alpha=0} - \sum_{j=0}^{m_I}\left\langle X_j, C_\alpha^j\right\rangle - \sum_{k=1}^{m_E}\left\langle Y_k, D_\alpha^k\right\rangle\right)$$
$$+\left\langle \Xi, \begin{bmatrix}\gamma & \mathbf{z}^T\\ \mathbf{z} & Z\end{bmatrix}\right\rangle + \sum_{j=1}^{m_I}\left\langle \Theta_j, X_j\right\rangle$$

with $\boldsymbol{\lambda} \in \mathbb{R}^{d[\ell]}$, $\Xi \in \mathbb{S}^{\nu+1}$, $\Xi \succcurlyeq 0$ and $\Theta \in \mathbb{S}^{\nu}$, $\Theta_j \succcurlyeq 0$, leading to the following system of optimality conditions:

$$\begin{aligned} Z^{-1} - \sum_{\alpha:|\alpha|\le 2\ell}\lambda_\alpha\left(\sum_{\beta+\gamma=\alpha}\boldsymbol{\xi}_\beta\boldsymbol{\xi}_\gamma^T\right) + \Xi_{2:\nu+1,2:\nu+1} &= 0 \\ \sum_{\alpha:|\alpha|\le 2\ell}\lambda_\alpha\boldsymbol{\xi}_\alpha + \Xi_{2:\nu+1,1} &= 0 \\ \lambda_0 - \Xi_{1,1} &= 0 \\ \sum_{\alpha:|\alpha|\le 2\ell}\lambda_\alpha C_\alpha^j - \Theta_j = 0, \quad j &= 0,\dots,m_I \\ \sum_{\alpha:|\alpha|\le 2\ell}\lambda_\alpha D_\alpha^k = 0, \quad k &= 1,\dots,m_E\,. \end{aligned} \tag{12}$$

The Lagrangian dual to (10) reads as

$$\inf_{\boldsymbol{\lambda},\Xi,\Theta}\ \sup_{Z,\mathbf{z},\gamma,X,Y}\ \mathcal{L}(Z,\mathbf{z},\gamma,X,Y;\boldsymbol{\lambda},\Xi,\Theta)$$

and, using (12), we get

$$\sup_{Z,\mathbf{z},\gamma}\mathcal{L}(Z,\mathbf{z},\gamma,X,Y;\boldsymbol{\lambda},\Xi,\Theta) = \log\det(Z)$$
$$+\lambda_0 - \sum_{\alpha:|\alpha|\le 2\ell}\lambda_\alpha\left\langle Z, \sum_{\beta+\gamma=\alpha}\boldsymbol{\xi}_\beta\boldsymbol{\xi}_\gamma^T\right\rangle - \sum_{\alpha:|\alpha|\le 2\ell}\lambda_\alpha\left(\sum_{j=0}^{m_I}\left\langle X_j, C_\alpha^j\right\rangle + \sum_{k=1}^{m_E}\left\langle Y_k, D_\alpha^k\right\rangle\right)$$
$$+\left\langle \Xi_{2:\nu+1,s:\nu+1}, Z\right\rangle + \sum_{j=1}^{m_I}\left\langle \Theta_j, X_j\right\rangle$$
$$= \log\det(Z) + \lambda_0 - \left\langle \sum_{\alpha:|\alpha|\le 2\ell}\lambda_\alpha\left(\sum_{\beta+\gamma=\alpha}\boldsymbol{\xi}_\beta\boldsymbol{\xi}_\gamma^T\right) - \Xi_{2:\nu+1,2:\nu+1}, Z\right\rangle$$
$$= \log\det(Z) + \lambda_0 - \left\langle Z^{-1}, Z\right\rangle = \log\det(Z) + \lambda_0 - \nu\,.$$

Setting $\Psi := Z^{-1} = \sum_{\alpha:|\alpha|\le 2\,\ell} \lambda_\alpha \left(\sum_{\beta+\gamma=\alpha} \boldsymbol{\xi}_\beta \boldsymbol{\xi}_\gamma^T \right) - \Xi_{2:\nu+1,2:\nu+1}$, we arrive at the objective function of the dual problem. The condition $\Xi \succcurlyeq 0$, together with the equalities (12), then leads to the constraints in the dual problem (11). □

The SDP formulation (11) corresponds to the following moment problem:

$$\begin{aligned} \inf_{\Psi\in\mathbb{S}^\nu,\, \Lambda\in\mathbb{R}[\mathbf{x}]_\ell^*} \quad & \Lambda_0 - \log\det(\Psi) - \nu \\ \text{s.t.} \quad & \begin{bmatrix} \Lambda_0 & \Lambda(\boldsymbol{\xi})^T \\ \Lambda(\boldsymbol{\xi}) & \Psi - \Lambda(\boldsymbol{\xi}\boldsymbol{\xi}^T) \end{bmatrix} \succcurlyeq 0 \\ & \Psi \succcurlyeq 0\,, \\ & H_{g_j}(\Lambda) \succcurlyeq 0, \quad j = 0,\ldots,m_I, \\ & \Lambda(\mathbf{x}^\alpha h_k) = 0, \quad |\alpha| \le 2\,\ell - d'_k,\ k = 1,\ldots,m_E\,, \end{aligned} \tag{13}$$

where $\Lambda(\boldsymbol{\xi}) = [\Lambda(\xi_1(\mathbf{x})),\ldots,\Lambda(\xi_\nu(\mathbf{x}))]$, $\Lambda(\boldsymbol{\xi}\boldsymbol{\xi}^T) = (\Lambda(\xi_i(\mathbf{x})\xi_j(\mathbf{x})))_{1\le i,j\le\nu}$, $g_0 = 1$, $d_0 = 0$ and $H_{g_j}(\Lambda) = (\Lambda(g_j\mathbf{x}^{\alpha+\beta}))_{|\alpha|,|\beta|\le\frac{2\,\ell-d_j}{2}}$.

4 Parameterized surfaces

Most of the representations of shapes used in CAD are based on piecewise polynomial or rational parametrizations, namely the image of functions of the form $\sigma : \mathbf{u} \in D \mapsto (p_1(\mathbf{u}),\ldots,p_3(\mathbf{u})) \in \mathbb{R}^3$, where D is typically an interval, the unit box in $\mathbb{R}^2$ or the unit cube in $\mathbb{R}^3$ and p_i are spline or piecewise polynomial functions or a ratio of two spline functions. For the sake of simplicity, hereafter the functions p_i will be polynomial functions. We illustrate the use of optimization tools on two types of problems involving surfaces parameterized by polynomials over the domain $D = [0,1]^2$. These examples generalize easily to parametric volumes.

4.1 Closest point and surface-surface intersection

Given a point $A = (a_1,a_2,a_3) \in \mathbb{R}^3$, finding the closest point to A on a parameterized surface $\sigma : (u_1,u_2) \in D = [0,1]^2 \mapsto (p_1(u_1,u_2),\ldots,p_3(u_1,u_2))$ can easily be stated as a minimization problem. This minimization problem is the problem (1) with

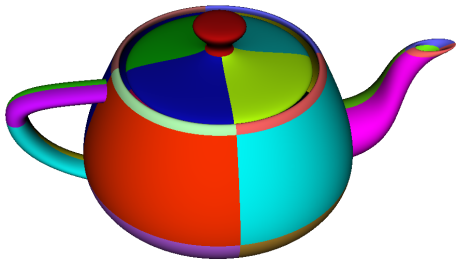


Fig. 1: A teapot model composed on 20 patches of bicubic polynomial parametrizations.

$$\begin{aligned} f &= \sum_{i=1}^{3} (a_i - x_i)^2, \\ \mathbf{g} &= \left\{ u_j,\ 1 - u_j, \quad j = 1,2 \right\}, \\ \mathbf{h} &= \left\{ x_i - p_i(u_1,u_2), \quad i = 1,\ldots,3 \right\}. \end{aligned} \tag{14}$$

The objective function is a polynomial of degree 2. We have 3 equality constraints $x_i - p_i(u_1,u_2) = 0, i = 1,\ldots,3$ and 4 sign constraints $0 \leq u_j \leq 1$. This usually gives a single closest point, except for points on the medial axis of the surface as illustrated in Figure 2.

To detect if two surfaces given by the parametrizations $\sigma_1 : \mathbf{u}_1 \in D \mapsto (p_{1,1}(\mathbf{u}_1),\ldots,p_{1,3}(\mathbf{u}_1)) \in \mathbb{R}^3$, $\sigma_2 : \mathbf{u}_2 \in D \mapsto (p_{2,1}(\mathbf{u}_2),\ldots,p_{2,3}(\mathbf{u}_2)) \in \mathbb{R}^3$ intersect, we use a slightly different formulation in order to get generically a single minimizer: We solve the optimization problem (1) with

$$\begin{aligned} f &= \|\mathbf{u}_1 - \mathbf{u}_2\|^2, \\ \mathbf{g} &= \left\{ u_{k,j},\ 1 - u_{k,j}, \quad j = 1,2,\ k = 1,2 \right\}, \\ \mathbf{h} &= \left\{ p_{1,i}(\mathbf{u}_1) - p_{2,i}(\mathbf{u}_2), \quad i = 1,\ldots,3 \right\}. \end{aligned} \tag{15}$$

Example of closest point and intersection point computation

We illustrate in Figure 2 the computation of the closest point based on the moment formulation (3) for the polynomial objective and constraints (14) (the two yellow points, which are the closest to the red point on the pink patch) for patches of degree 3 in u_1 and 3 in u_2 (called bi-cubic patches). We also show an intersection point (the green point on the intersection of the pink and blue patches) corresponding to the solution of (15). The orders of relaxation used for these computations are respectively $\ell = 3$ and $\ell = 4$.

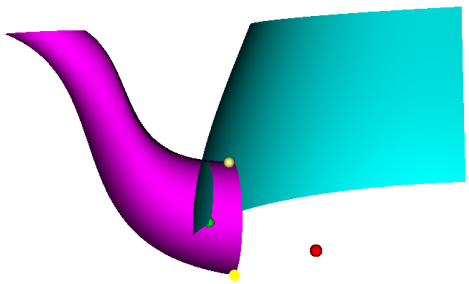


Fig. 2: Closest points and intersection point for bicubic parametric surfaces.

4.2 Bounding box and enclosing ellipsoid of parametric surfaces

Computing simple minimal enclosing solids of a given shape is an important problem with many applications in animation, collision detection, simulation, robotics, etc.

For minimal enclosing axis-aligned bounding boxes of parametric surfaces, this reduces to solving problems of the form (14), where the objective function is replaced by $\pm x_i$.

For minimal enclosing ellipsoids, we need to solve the polynomial SDP (8) using SoS relaxations (10). Notice that (10) is not a standard (linear) SDP, due to the determinant function. It can be solved, for instance, by software `Hypatia`[4] or `Mosek` using so called *LogDetTriangular* cones.

Example of minimal enclosing ellipsoid computation

In Figure 3, we present an example of a minimal enclosing ellipsoid for $p_1(u,v) = u+v, p_2(u,v) = 2u+u^2-uv+v^2, p_3(u,v) = v+\frac{1}{2}u^3+\frac{1}{2}v^3$ solved by `MomentTools` at relaxation level $\ell = 2$, with `Mosek` convex optimizer:

```
> s = [u+v, 2*u+u^2-u*v+v^2, v+1/2*u^3+ 1/2*v^3]
> H = [x1-s[1], x2-s[2], x2-s[3]]
> G = [u-u^2, v-v^2]
> P = [x1, x2, x3]
> c, U, M = min_ellipsoid(P, H, G, X, 2, Mosek.Optimizer)
```

It returns the center `c` and the matrix `U`, which columns are the principal axes of the ellipsoid.

[4] https://github.com/chriscoey/Hypatia.jl

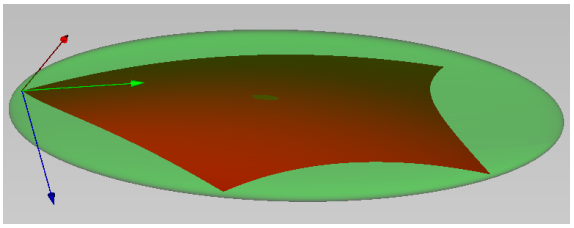


Fig. 3: Closest points, intersection point and minimal enclosing ellipsoid for bicubic parametric surfaces.

5 Robots and mechanisms

A parallel robot is defined by a fixed platform

$$A = [A_1, \ldots, A_6] \subset \mathbb{R}^{3\times 6},$$

and a moving platform with initial position

$$B = [B_1, \ldots, B_6] \subset \mathbb{R}^{3\times 6},$$

connected by extensible arms A_i–B_i. Figure 4 shows two examples of parallel robots. The robot on the right is known as the Stewart platform, and it is the geometry that we are considering in our numerical experiments.

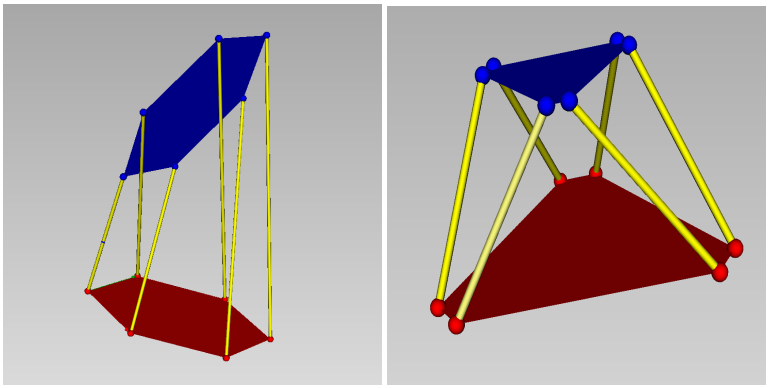


Fig. 4: Parallel robots with a fixed platform (red platform), moving platform (blue platform) and extensible arms (yellow arms).

We represent the displacement of the moving platform by a rotation R of the orthogonal group $O(\mathbb{R}^3)$ and a translation $T \in \mathbb{R}^3$ so that the position of the points of the moving platform is

$$\widehat{B}_i = RB_i + T, \quad i = 1, \ldots, 6$$

with the length of the arms

$$d_i = \|RB_i + T - A_i\|_2, \; i = 1, \ldots, 6.$$

The problem consists in analyzing the position of the moving platform under the constraints that the lengths of the arms are within some intervals:

$$m_i \leq d_i \leq M_i, \quad i = 1, \ldots, 6\,.$$

Here is the parameterization of the rotation by unit quaternion:

$$R = \begin{bmatrix} u_1^2 + u_2^2 - u_3^2 - u_4^2 & -2u_1u_4 + 2u_2u_3 & 2u_1u_3 + 2u_2u_4 \\ 2u_1u_4 + 2u_2u_3 & u_1^2 - u_2^2 + u_3^2 - u_4^2 & -2u_1u_2 + 2u_3u_4 \\ -2u_1u_3 + 2u_2u_4 & 2u_1u_2 + 2u_3u_4 & u_1^2 - u_2^2 - u_3^2 + u_4^2 \end{bmatrix} \tag{16}$$

with $u_1^2 + u_2^2 + u_3^2 + u_4^2 = 1$ and the translation $T = [x, y, z]$. Then

$$\delta_i(u_1, u_2, u_3, u_4, x, y, z) = d_i^2 = \|B_i\|^2 + \|A_i\|^2 + \|T\|^2 + 2RB_i \cdot T - 2RB_i \cdot A_i - 2T \cdot A_i \tag{17}$$

is a polynomial function of $\mathbf{u} = (u_1, \ldots, u_4)$ and x, y, z of degree 2 in $\mathbf{u}$ and total degree 3 (here $\mathbf{v} \cdot \mathbf{w}$ stands for the standard inner product of vectors $\mathbf{v}, \mathbf{w} \in \mathbb{R}^p$).

5.1 Direct kinematic problem

The direct kinematic problem consists in finding the position of the moving platform such that a point C attached to the platform is at given point X_0 in the space. To solve this problem, we search for a position of the platform, which minimizes the norm of the translation T, solving the optimization problem (1) with

$$\begin{aligned} f &= \|T\|^2, \\ \mathbf{g} &= \left\{\delta_i - m_i^2, M_i^2 - \delta_i, \quad i = 1, \ldots, 6\right\}, \\ \mathbf{h} &= \left\{(X_0)_j - (RC + T)_j, \quad j = 1, \ldots, 3\,, \right. \\ &\qquad \left. \|u\|^2 - 1\right\}. \end{aligned} \tag{18}$$

This is when we know the problem is feasible. If we are uncertain about the existence of a feasible solution, we reformulate the objective functions and the constraints to

$$
\begin{aligned}
f &= \|T\|^2 + \rho\|X_0 - (RC + T)\|^2, \\
\mathbf{g} &= \left\{\delta_i - m_i^2, M_i^2 - \delta_i, \quad i = 1, \ldots, 6\right\}, \\
\mathbf{h} &= \left\{\|u\|^2 - 1\right\},
\end{aligned} \tag{19}
$$

where ρ is a penalty parameter (we choose $\rho = 1000$ in our experiments). The minimizer of this problem yields the translation(s) T^{new} and rotation(s) R^{new} required to locate the new position of the platform. This position is calculated by $B_i^{\text{new}} = R^{\text{new}} B_i + T^{\text{new}}$ for $i = 1, \ldots, 6$.

Example of a direct kinematic problem

In Figure 5, we illustrate the computational result of solving problem (1) for finding the new position of a Stewart platform. The values of the fixed platform A_0 and the initial position of the moving platform B_0 of this geometry are produced in Julia by

```
> function pl(x,r)  [r * cos(x), r * sin(x), 0] end
> A0    = [pl(0,1), pl(pi/12,1),pl(2pi/3,1), pl(2pi/3+pi/12,1),
           pl(4pi/3,1), pl(4pi/3+pi/12,1)]
> B0pre = [pl(5pi/3+pi/12,1/2), pl(pi,1/2), pl(pi/12+pi/3,1/2),
           pl(pi/3,1/2), pl(pi+pi/12,1/2), pl(5pi/3,1/2)]
> Xi0 = [1, 0, 0, 0, 0, 0, 1]
> R0  = Rotation(Xi0[1:4]...); T0 = Xi0[5:7]
> B0  = [R0*a + T0 for a in B0pre]
```

where the function `Rotation` is the quaternion parametrization of rotations given in (16). In this numerical experiment, we consider the point C to be the centre of the moving platform and the point X_0 reached by the moving platform (shown in green in Figure 5), to be

```
> C = sum(b for b in B0)/length(B0)
> X0 = [0.6,0.4,1]
```

In addition, the upper bound and lower bound on the lengths are

```
> D0 = norm.(B0-A0)
> ub = D0 .+ 2.5
> lb = D0 .- 2.5
```

where `D0` is the vector of the lengths of the robot arms in the initial position. We consider the objective function and constraints to be

```
> f = nrm2(T)
> g = []; for i in 1:6 g = [g; D1[i] - lb[i]^2; ub[i]^2 - D1[i]] end
> h = [u1^2+u2^2+u3^2+u4^2-1]
> h = [h; X0-(R*C+T)]
```

We solve the moment relaxation of order $\ell = 3$ by using `Mosek` convex optimizer:

```
> using MomentTools, DynamicPolynomials, MosekTools
> X = @polyvar u1 u2 u3 u4 x y z
> v, M = minimize(f, h, g, X, 3, Mosek.Optimizer)
> w, Xi = get_measure(M)
```

By selecting the minimizers `Xi[:,i]` with a big enough weight `w[i]`, we obtain T^{new} and R^{new} needed to find the platform's new position:

```
> Sol = []
> for i in 1:length(w)
     if abs(w[i]) > 0.1  push!(Sol,Xi[:,i])  end
  end
> R_new = Rotation(Sol[1][1:4]...)
> T_new = Sol[1][5:7]
```

and the new platform will be

```
> B_new = [R_new *a + T_new for a in B0]
```

which is demonstrated in violet in Figure 5.

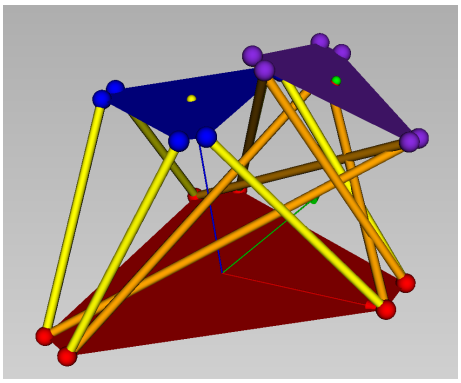


Fig. 5: The new position of the platform is shown in violet such that the middle of the blue platform is at a given point in the space (the green point).

5.2 Bounding box of robot workspace

Let C be a point attached to the moving platform B; in our numerical experiments it is the center of the platform but it can be any other point. We will be looking for the axis-aligned bounding box problem of this point after all possible rotations and translations, i.e., of the point $\widehat{C} = RC + T$. The bounds can be obtained by solving the optimization problem (1) with

$$\begin{aligned} f &= \pm\, \widehat{C}_k = (RC+T)_k, \quad k = 1,2,3, \\ \mathbf{g} &= \left\{\delta_i - m_i^2,\ M_i^2 - \delta_i, \quad i = 1,\ldots,6\right\}, \\ \mathbf{h} &= \left\{\|\mathbf{u}\|^2 - 1\right\}. \end{aligned} \tag{20}$$

One needs to solve six optimization problems, three with objective $\widehat{C}_k$ and three with $-\widehat{C}_k$, $k = 1,2,3$, to find an interval for each coordinate of $\widehat{C}$. By using these values, we obtain the extreme positions for different faces of the intended bounding box.

Example of computing the bounding box

In Figure 6–left, we present the computational result of solving problem (1) to find the bounding box of a Stewart platform. The coordinates of the platform points for this robot are as mentioned in the previous example. We choose the upper bound and lower bound of the lengths to be

```
> ub = D0 .+ 0.2
> lb = D0 .- 0.2
```

The constraints of this problem are

```
> g = []; for i in 1:6 g = [g; D1[i]-lb[i]^2; ub[i]^2-D1[i]; B1[i][3]] end
> h = [u1^2+u2^2+u3^2+u4^2-1]
```

where

```
> B1 = [R*a+T for a in B0]
```

As we mentioned, to find the bounding box, we need to solve six optimization problems. If we consider

```
> C_hat = R*C+T
```

where `C` is the centre of the moving platform, the objective functions of these problems for $i = 1,\ldots,3$ will be

```
> f[i] = C_hat[i]
```

We solve six moment relaxations by using `Mosek` software

```
> for i in 1:3
    b[i], Mb[i] = minimize(f[i],h,g,X,l,opt)
    B[i], MB[i] = maximize(f[i],h,g,X,l,opt)
end
```

from which we deduce the bounding box shown in Figure 6–left. These moment relaxations are of order $\ell = 3$. Only to find `b[3]`, we needed a higher order relaxation and we considered $\ell = 4$. In order to find out if the order of relaxation is sufficient, we check whether $u_1^2 + u_2^2 + u_3^2 + u_4^2$ is close enough to one at the solutions. In our numerical experiments to compute the minimum `b[3]` of the z-coordinate, with the relaxation order $\ell = 3$, we find 4 approximate minimizers with the following norm for $\mathbf{u}$:

```
>  norm sol u: 0.9345868913061018
>  norm sol u: 0.9796236439932896
>  norm sol u: 0.6612425286297167
>  norm sol u: 0.8916846004102044
```

Since these values are not close enough to 1, we increase the order of the relaxation and for $\ell = 4$, we obtain 4 approximate minimizers with

```
>  norm sol u: 0.9973888649655355
>  norm sol u: 0.9968397474090084
>  norm sol u: 0.9971812563313878
>  norm sol u: 0.9985517657681842
```

The relaxation at order $\ell = 4$ gives a good enough approximation of the minimizers and of the minimum of z.

5.3 Enclosing ellipsoid of robot workspace

The minimal enclosing ellipsoid for all positions of point C from the above section can now be found by the same technique as in Section 4.2. The points $\xi(\mathbf{x})$ in (8) will now be replaced by the point $\widehat{C}$, as a function of $u_1, u_2, u_3, u_4, x, y, z$ as in problem (20). Figure 6–left presents a comparison of the bounding box and the minimal ellipsoid for the center point of the moving platform. The order of relaxation for computing the minimal ellipsoid is $\ell = 4$ in this example.

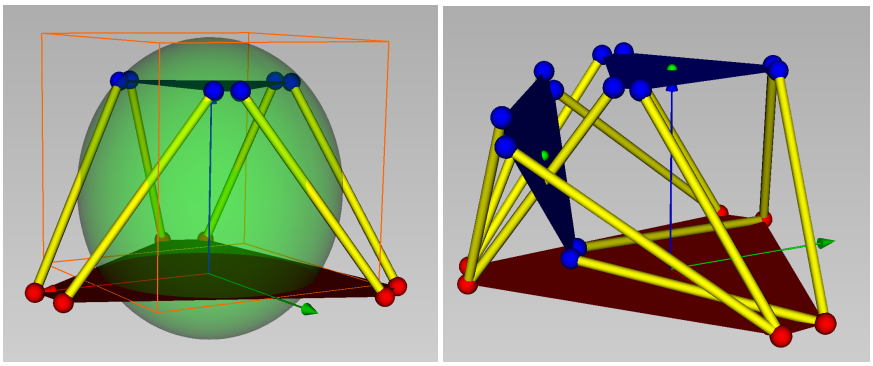


Fig. 6: On the left, we show a comparison of minimal enclosing ellipsoid and bounding box. The initial position and extreme position of the platform minimizing y is presented in the right.

5.4 Trajectories of a parallel robot

In this section, we analyse the trajectory of the point C attached to the platform B, first when the first interval constraint is satisfied and the other five lengths have a fixed value in their length interval; secondly when the first two interval constraints are satisfied and the other four lengths have a fixed value.

Case 1: Five fixed values

In this case, the trajectory of the point C is the curve consisting of the possible positions of the C in the space. To obtain different positions of the platform under these constraints, we minimize and maximize $\widehat{C}_1$, so we will have an interval for extreme positions corresponding to the first coordinate. This means that we need to solve the problem (20) with objective $\pm\widehat{C}_1$ and the additional constraints

$$\delta_i = \delta_i^0, \quad i = 2, \ldots, 6,$$

in which $\delta^0 = (d^0)^2 = \|B_0 - A_0\|$. Next, we sample the interval between these extreme values at intermediate values μ_k, $k = 1, \ldots, m$ and solve feasibility problems with an additional equality constraint $\widehat{C}_1 - \mu_k = 0$ for $k = 1, \ldots, m$. This minimization is done by solving problem (1) with

$$\begin{aligned} f &= 1, \\ \mathbf{g} &= \left\{\delta_1 - m_1^2,\ M_1^2 - \delta_1\right\}, \\ \mathbf{h} &= \Big\{\widehat{C}_1 - \mu_k, \quad k = 1, \ldots, m, \\ &\qquad \delta_i - \delta_i^0, \quad i = 2, \ldots, 6, \\ &\qquad \|\mathbf{u}\|^2 - 1\Big\}. \end{aligned} \tag{21}$$

The minimizers of these problems yield the displacements (R, T) and the positions of C in the space. The same computation is repeated for the other coordinates $\widehat{C}_2, \widehat{C}_3$ to get a better sampling of the trajectory.

Case 2: Four fixed values

We compute positions of C attached to the platform when the first two interval constraints are satisfied and the other four lengths have a fixed value. This traces out a surface of the positions of the point C. The problem is analogous to Case 1; only,we need to compute the interval for extreme positions corresponding to the first and second coordinates. Thus, we need to solve problem (1) with

$$\begin{aligned} f &= \pm\,\widehat{C}_k, \quad k = 1, 2, \\ \mathbf{g} &= \left\{\delta_i - m_i^2,\ M_i^2 - \delta_i, \quad i = 1, 2\right\}, \\ \mathbf{h} &= \Big\{\delta_i - \delta_i^0, \quad i = 3, \ldots, 6, \\ &\qquad \|\mathbf{u}\|^2 - 1\Big\}. \end{aligned} \tag{22}$$

If we consider μ and γ to be values in the intervals obtained for the coordinates $\widehat{C}_1$ and $\widehat{C}_2$, respectively, the final feasibility problem to solve should satisfy all the previous constraints and the additional constraints $\widehat{C}_1 = \mu$, $\widehat{C}_2 = \gamma$. The final problem to solve is problem (1) with

$$
\begin{aligned}
f &= 1\,, \\
\mathbf{g} &= \left\{\delta_i - m_i^2,\ M_i^2 - \delta_i, \quad i = 1,2\right\}, \\
\mathbf{h} &= \Big\{\widehat{C}_1 - \mu,\ \widehat{C}_2 - \gamma,\ \|\mathbf{u}\|^2 - 1\,, \\
&\qquad \delta_i - \delta_i^0, \quad i = 2,\ldots,6\Big\}\,.
\end{aligned}
\tag{23}
$$

As a result, we will have a set of points approximating a surface. We obtain these points by changing the constraints, choosing different values for $\widehat{C}_1$ and $\widehat{C}_2$ and solving the corresponding optimization problem.

Figure 7 shows the computational result of solving problem (1) to find the trajectories of a Stewart platform for these two cases. The order of relaxation for these problems is $\ell = 3$.

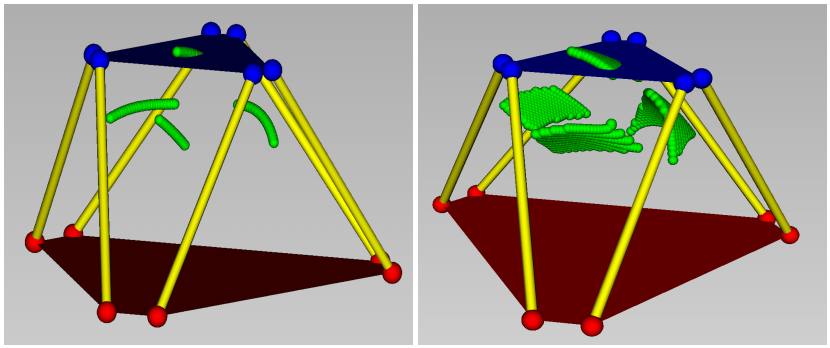


Fig. 7: On the left, the trajectory of the center of the platform with 5 arm lengths fixed is shown. The trajectory of the center of the platform with 4 arm lengths fixed is shown in the right.

6 Conclusion

We have demonstrated how techniques of polynomial optimization can be used in the important area of geometric modeling. We focused on practical examples such as finding the bounding box or minimum enclosing ellipsoid for parametric surfaces or for a workspace of parallel robots. The limitations of our approach are typical for polynomial optimization, in particular, reliability of the numerical solutions of the resulting SDP problems and, sometimes, the necessity for higher order relaxations resulting in very large SDPs. The latter is, however, not too restrictive, given the small dimensions of the respective polynomial optimization problems. We thus believe that this technique will find further use, in particular, in robotics.

Acknowledgement. This work has been supported by European Union's Horizon 2020 research and innovation programme under the Marie Skłodowska-Curie Actions, grant agreement 813211 (POEMA). The Julia package `MomentTools.jl` and Matlab package `Loraine` used in this chapter have been developed in the context of the POEMA project.

References

1. Baldi, L., Mourrain, B.: On the Effective Putinar's Positivstellensatz and Moment Approximation. Mathematical Programming, Series A (2022). DOI 10.1007/s10107-022-01877-6. URL https://hal.science/hal-03437328
2. Ben-Tal, A., Nemirovski, A.: Lectures on modern convex optimization: analysis, algorithms, and engineering applications. Society for Industrial and Applied Mathematics, Philadelphia, PA (2001)
3. De Klerk, E.: Aspects of semidefinite programming: interior point algorithms and selected applications. Kluwer Academic Publishers (2002)
4. Farin, G., Hoschek, J., Kim, M.S.: Handbook of Computer Aided Geometric Design. Elsevier (2002)
5. Farin, G.E.: Curves and Surfaces for CAGD: A Practical Guide, 5th ed edn. Morgan Kaufmann Series in Computer Graphics and Geometric Modeling. Morgan Kaufmann, San Francisco, CA (2001)
6. Habibi, S., Kočvara, M., Stingl, M.: Loraine—an interior-point solver for low-rank semidefinite programming. Optmization Methods and Software (2022). Submitted.
7. Harmouch, J., Khalil, H., Mourrain, B.: Structured low rank decomposition of multivariate Hankel matrices. Linear Algebra and its Applications (2017). DOI 10.1016/j.laa.2017.04.015. URL https://hal.inria.fr/hal-01440063
8. Lasserre, J.B.: Global optimization with polynomials and the problem of moments. SIAM Journal on Optimization **11**(3), 796–817 (2001)
9. Marschner, Z., Zhang, P., Palmer, D., Solomon, J.: Sum-of-squares geometry processing. ACM Transactions on Graphics **40**(6), 1–13 (2021). DOI 10.1145/3478513.3480551
10. Mourrain, B.: Polynomial-exponential decomposition from moments. Foundations of Computational Mathematics **18**(6), 1435–1492 (2018). DOI 10.1007/s10208-017-9372-x. URL https://hal.inria.fr/hal-01367730
11. Nesterov, Y.: Squared functional systems and optimization problems. In: H. Frenk, K. Roos, T. Terlaky, S. Zhang (eds.) High Performance Optimization, pp. 405–440. Springer, Boston, MA (2000)
12. Parrilo, P.A.: Structured semidefinite programs and semialgebraic geometry methods in robustness and optimization. Ph.D. thesis, California Institute of Technology (2000)
13. Schweighofer, M.: Optimization of Polynomials on Compact Semialgebraic Sets **15**(3), 805–825. DOI 10.1137/S1052623403431779. URL http://epubs.siam.org/doi/10.1137/S1052623403431779
14. Shor, N.Z.: Quadratic optimization problems. Soviet Journal of Computer and Systems Sciences **25**, 1–11 (1987)
15. Todd, M.J.: Minimum-Volume Ellipsoids. Society for Industrial and Applied Mathematics, Philadelphia, PA (2016)
16. Todd, M.J., Toh, K.C., Tütüncü, R.H.: On the Nesterov–Todd direction in semidefinite programming. SIAM Journal on Optimization **8**(3), 769–796 (1998)