Synthesis and Optimization of Force Closure Grasps via Sequential Semidefinite Programming

Hongkai Dai, Anirudha Majumdar and Russ Tedrake[1]*

Abstract In this paper we present a novel approach for synthesizing and optimizing both positions and forces in force closure grasps. This problem is a non-convex optimization problem in general since it involves constraints that are *bilinear*; in particular, computing wrenches involves a bilinear product between grasp contact points and contact forces. Thus, conventional approaches to this problem typically employ general purpose gradient-based nonlinear optimization. The key observation of this paper is that the force closure grasp synthesis problem can be posed as a *Bilinear Matrix Inequality* (BMI), for which there exist efficient solution techniques based on semidefinite programming. We show that we can synthesize force closure grasps on different geometric objects, and by maximizing a lower bound of a grasp metric, we can improve the quality of the grasp. While this approach is not guaranteed to find a solution, it has a few distinct advantages. First, we can handle non-smooth but convex *positive semidefinite* constraints, which can often be important. Second, in contrast to gradient-based approaches we can prove *infeasibility* of problems. We demonstrate our method on a 15 joint robot model grasping objects with various geometries. The code is included in https://github.com/RobotLocomotion/drake

1 Introduction

Force closure, which measures the ability of a grasp to resist wrench disturbances, is an important property in grasping and has an extensive literature [18, 17]. A commonly observed fact is that synthesis of force closure grasps is a non-convex optimization problem, mostly due to the fact that computing the torque on an object involves a bilinear product between contact locations and contact forces. As a result, most approaches resort to gradient-based non-convex nonlinear optimization to synthesize a force closure grasp [5]. On the other hand, when fixing the contact locations, checking if the given contact achieve force closure becomes a convex optimization problem over contact forces only [2, 10]. Moreover, several grasp metrics

* [1] Computer Science and Artificial Intelligence Lab, MIT, daih,anirudha,russt@csail.mit.edu. This work is supported by David S.Y & Harold Wong Fellowship, and ONR MURI grant N00014-09-1-1051.

have been introduced to measure the quality of a force closure grasp. These involve computing the smallest wrench that the grasp cannot resist with bounded contact forces [12, 9, 15]. Liu et al. optimized the contact locations based on such a metric with nonlinear optimization [13].

In this paper we exploit the observation that although the force-closure grasp synthesis problem involves non-convex constraints, the bilinear structure of the constraints makes it special. In particular we pose the problem of finding (and optimizing) a force closure grasp as a *Bilinear Matrix Inequality* (BMI). There exist powerful tools to solve BMIs via sequences of semidefinite programming problems (SDP), which is a special form of convex optimization.

Besides satisfying the force closure constraint, the contact locations should also be reachable by the hand, subject to robot kinematics constraints. Traditionally finding robot posture is solved by the Jacobian transpose method [4] or nonlinear optimization [7] on robot minimal coordinates, which involve non-polynomial (e.g. trigonometric) functions. Recently Rosales et al. searched for hand posture by solving linear and bilinear equations of robot maximal coordinates, through an iterative linear optimization scheme [20]. We will adopt the similar idea here to formulate the inverse kinematics problem as BMIs, and solve them through sequential semidefinite programming.

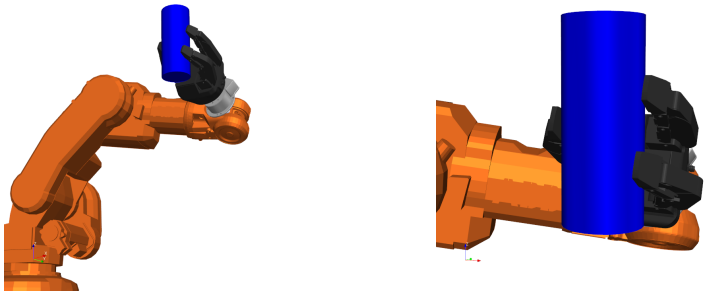

Fig. 1 Optimized force closure grasp of a 15 joints robot on a cylinder, from two perspectives

Sequential semidefinite programming is a common technique in solving bilinear matrix inequalities (BMI) [11, 8]. It relaxes the original non-convex problem to a convex SDP, and in each iteration solves a convex relaxation until convergence. The advantage of this approach over gradient based nonlinear-optimization include

1. The ability to incorporate non-smooth *positive semidefinite* (psd) constraints, which appear frequently in grasp planning. The gradient-based nonlinear optimization cannot handle such constraints gracefully due to non-smoothness.
2. Proof of infeasibility. If the relaxed convex problem is infeasible, then the original non-convex problem is also infeasible. Nonlinear optimization cannot guarantee that a problem is globally infeasible, when locally it fails to find a solution.

We will introduce the background on solving BMI with sequential SDP in Section 2, and elaborate how this technique can be applied to synthesis and optimization of force closure grasps in Section 3. Our results are shown in Section 4.

2 Background

As we will see in Section 3, finding force closure grasps for a broad class of object geometries can be posed as a bilinear matrix inequality (BMI), which is a particular kind of optimization problem. In this section we provide an introduction to BMIs along with methods to find feasible solutions to them.

2.1 Bilinear Matrix Inequalities

Bilinear matrix inequalities (BMIs) are problems of the following form:

$$\text{Find} \qquad x \in \mathbb{R}^n \tag{1}$$

$$\text{s.t.} \qquad F_0 + \sum_{i=1}^{N} x_i F_i + \sum_{i=1}^{N}\sum_{j=1}^{N} x_i x_j F_{ij} \succeq 0, \tag{2}$$

where F_0, F_i, F_{ij} are constant $m \times m$ symmetric matrices. $\succeq 0$ means the matrix on the left hand-side is positive semidefinite (psd), i.e. all the eigenvalues are non-negative; the special case is when the matrix is just a scalar, then $\succeq 0$ is the same as ≥ 0. We also note that BMIs include constraints that are both bilinear ($i \neq j$) as well as quadratic ($i = j$).

2.2 Finding Feasible Solutions to BMIs

While it is well known that BMIs are NP-hard in general [11], there exist very good heuristic methods based on semidefinite programming (SDP) for solving them. Here we review the method presented in [11] for finding feasible solutions to BMIs.

The first step is to write the BMI (1) as a rank-constrained *Linear Matrix Inequality* (LMI) with an additional variable $X \in \mathbb{R}^{N\times N}$:

$$\text{Find:} \qquad x \in \mathbb{R}^N, X \in \mathbb{R}^{N\times N} \tag{3}$$

$$\text{s.t.} \qquad F_0 + \sum_{i=1}^{N} x_i F_i + \sum_{i=1}^{N}\sum_{j=1}^{N} X_{ij} F_{ij} \succeq 0, \tag{4}$$

$$M := \begin{bmatrix} X & x \\ x^T & 1 \end{bmatrix} \succeq 0, \tag{5}$$

$$\text{rank}(M) = 1. \tag{6}$$

Here, each occurrence of bilinear terms $x_i x_j$ in (1) has been replaced by the (i, j) element of the decision matrix X. Constraints (5)(6) have been introduced to ensure that $X = xx^T$, resulting in the problems (3) and (1) having equivalent constraints. We note that without the rank constraint (6), problem (3) is a semidefinite program, which is a particular kind of *convex* optimization problem and can be solved efficiently (e.g., using interior point methods) [3].

The key idea in [11] is to drop the rank constraint in (3) and solve a sequence of SDPs that attempt to minimize the rank of M, as shown in Algorithm 1.

Algorithm 1 Finding feasible solutions to BMIs

Minimize trace(X) subject to constraints (4) and (5). If problem is infeasible, then problem (1) is infeasible. If problem is feasible, initialize $x^{(0)}$ and $X^{(0)}$ with the solution. Initialize $k = 1$.
while ¬converged **do**
1. Minimize trace($X^{(k)}$) - $2x^{(k-1)T}x^{(k)}$ subject to the constraints (4) and (5).
2. Set $k \leftarrow k+1$
end while

Note that the first step in Algorithm 1 is the standard trace heuristic for minimizing the rank of a positive semidefinite matrix [3, 8]. The justification for the proceeding steps in the algorithm is based on the observation that the constraint (5) implies (by the Schur complement lemma) that $X \succeq 0$ and $X - xx^T \succeq 0$. This in turn implies that $\text{trace}(X) - x^T x \geq 0$ with equality holding if and only if $X = xx^T$ (i.e. when we have a feasible solution to (1)). Thus, Algorithm 1 proceeds by linearizing the function $\text{trace}(X) - x^T x$ and minimizing this linearization at every iteration. A termination criterion for Algorithm 1 is provided by the following Lemma in [11].

Lemma 1. *[11] The following sequence is bounded below by 0 and non-increasing for $k = 1, 2, \ldots$:*

$$t_k := trace(X^{(k)}) - 2x^{(k-1)T}x^{(k)} + x^{(k-1)T}x^{(k-1)}.$$

Hence, this sequence converges to a value $t_{opt} \geq 0$. Equality holds if and only if $X^{(k)} = x^{(k)}x^{(k)T}$ as $k \to \infty$.

Lemma 1 provides us with a convergence criterion for Algorithm 1. Assuming that the first step in Algorithm 1 is feasible (if this is not the case, the original BMI is infeasible), then convergence of the value of t_k to 0 implies that we have found a feasible solution to the BMI. In the case where t_{opt} is not 0, nothing can be inferred.

2.3 Implementation Details

An important detail in implementing Algorithm 1 is that the SDP constraint (5) can be quite large if one has many decision variables x. However, it is typically the case that a large number of variables do not multiply with each other as bilinear products. Formally, consider a graph whose vertices are the variables in x. Two vertices are connected by an edge if the corresponding variables appear in a bilinear product in some constraint. Then we can partition the variables x into subsets $x_{I_1}, x_{I_2}, \ldots, x_{I_k}, \ldots, x_{I_K}$ corresponding to the connected components of the graph. We can then replace the constraints (5) and (6) by the following constraints:

$$M_k := \begin{bmatrix} X_{I_k,I_k} & x_{I_k} \\ x_{I_k}^T & 1 \end{bmatrix} \succeq 0, \quad \text{rank}(M_k) = 1, \ \forall k = 1, \ldots, K. \tag{7}$$

The cost function in Algorithm 1 is then replaced by the sum of the traces of the matrices X_{I_k,I_k}. While we end up with more psd constraints in general, each constraint

involves a smaller matrix. Since SDP solve times typically scale poorly with the size of the largest psd constraint, we observe large computational gains in practice.

Another important implementation detail is to employ a randomization step in Algorithm 1, as described in [11]. In each iteration k of the algorithm, we sample a point $x_{rand}^{(k)}$ from the Gaussian distribution with mean $x^{(k)}$ and covariance $X^{(k)} - x^{(k)}x^{(k)T}$, where $(x^{(k)}, X^{(k)})$ is a solution to the SDP at the k-th iteration, and use cost function $\text{trace}(X^{(k+1)}) - 2x_{rand}^{(k)T}x^{(k+1)}$ in $k+1^{th}$ iteration. In practice, the randomization step prevents the algorithm from getting stuck in local minima.

Finally, we note that while we have restricted ourselves so far to *feasibility* problems, it is also possible to optimize cost functions subject to BMI constraints. We will discuss this in the context of optimizing grasp metrics in Section 3.3.

3 Approach

3.1 Force Closure

The force closure property for n grasp points $x_i \in \mathbb{R}^3, i = 1, ..., n$, is achieved when these grasp points can resist arbitrary external wrenches with contact forces f_i at point x_i lying within the friction cone. Mathematically, force closure is formulated as the existence of x_i and f_i satisfying the following constraints:

$$GG' \succeq \varepsilon I_{6\times 6} \tag{8a}$$

$$Gf = 0 \tag{8b}$$

$$f_i \in \text{int}(\mathcal{FC}_i) \tag{8c}$$

$$x_i \in \mathcal{S}_i \tag{8d}$$

where

$$G = \begin{bmatrix} I_{3\times 3} & I_{3\times 3} & \dots & I_{3\times 3} \\ \lfloor x_1 \rfloor_\times & \lfloor x_2 \rfloor_\times & \dots & \lfloor x_n \rfloor_\times \end{bmatrix}, \ \lfloor x_i \rfloor_\times = \begin{bmatrix} 0 & -x_i^{(3)} & x_i^{(2)} \\ x_i^{(3)} & 0 & -x_i^{(1)} \\ -x_i^{(2)} & x_i^{(1)} & 0 \end{bmatrix} \in \mathbb{R}^{3\times 3} \tag{9}$$

$\lfloor x_i \rfloor_\times$ is the skew-symmetric matrix representing the cross product $\lfloor x_i \rfloor_\times f_i = x_i \times f_i$, ε is a small given positive scalar, constraint (8a) is the same as G being full rank; $f = [f_1^T \ f_2^T \ \dots f_n^T]^T \in \mathbb{R}^{3n}$; $\text{int}(\mathcal{FC}_i)$ is the interior of the friction cone $\mathcal{FC}_i$ at grasp point x_i, and $\mathcal{S}_i$ is the admissible contact region of grasp point x_i (for example, the surface of the object being grasped).

We note that condition (8a) is quadratic on x_i, and (8b) is bilinear on x_i and f_i. Unlike some existing approaches that fix the contact points x_i and search only contact force f_i through convex optimization, we can search both x_i and f_i simultaneously by solving these BMIs through sequential SDP, as introduced in Section 2.2. In the following two subsections, we will show that friction cone constraint (8c) and contact region constraint (8d) can also be formulated as BMIs.

3.1.1 Friction cones

We consider the Coulomb friction cones as depicted in Fig 2. For the i^{th} friction cone

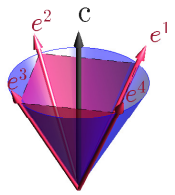

Fig. 2 A nonlinear friction cone (blue), and the 4-edge linearized friction cone (red). The red arrows $e^1,\dots,e^4$ are the edges of the linearized friction cone. The axis c is along the direction of the normal force, pointing outward from the contact surface.

$\mathcal{FC}_i$, the axis of the cone is denoted by a vector $c_i \in \mathbb{R}^3$, which is a normal vector originating at the grasp point x_i and pointing outward and perpendicular to the contact surface. We introduce c_i as a decision variable in our problem, to parameterize the friction cone $\mathcal{FC}_i$. For the benefit of the later discussion, we will constrain c_i to have unit length:

$$c_i^T c_i = 1. \tag{10}$$

If we use the nonlinear friction cone, then $f_i \in \text{int}(\mathcal{FC}_i)$ is equivalent to

$$f_i^T c_i > \frac{1}{\sqrt{\mu^2+1}}|f_i|, \tag{11a}$$

where μ is the fixed friction coefficient.

If c_i was fixed, constraint (11a) would be a *Second-order cone constraint* on f_i, which is a special type of psd constraint [1]. Thus by searching both c_i and f_i, constraints (10)(11a) are both BMIs on variables c_i and f_i.

If $\mathcal{FC}_i$ is a linearized friction cone with n_e edges, to compute its edges, we can first construct a cone $\mathcal{FC}_0$ that has unit axis $c_0 = [0\,0\,1]^T$, with edges $e_0^1, e_0^2, \dots, e_0^{n_e}$. Without loss of generality we suppose all the edges of the cone have unit length, $|e_0^j| = 1, j = 1,\dots,n_e$. The edge e_0^j can be computed using the friction coefficient and c_0, thus they are fixed. The linearized friction cone at x_i with cone axis c_i , can be obtained by appropriately rotating cone $\mathcal{FC}_0$ such that cone axis c_0 is aligned with c_i. Such rotation is parameterized with a unit quaternion z_i, satisfying constraints:

$$z_i \otimes z_i^* = 1 \tag{12a}$$

$$c_i = R(z_i)c_0 \tag{12b}$$

where $\otimes$ is the Hamiltonian product between quaternions. z_i^* is the conjugate of z_i, and $R(z_i) \in \mathbb{R}^{3\times3}$ is the rotation matrix corresponding to z_i, each entry in $R(z_i)$ is a second-order polynomial of z_i [21]. Applying the same rotation to the friction cone edges e_0^j generates the friction cone edges e_i^j at x_i.

$$e_i^j = R(z_i)e_0^j, \quad j = 1,\dots,n_e. \tag{13}$$

The contact force f_i is a positive weighted sum of the edges of the friction cone:

$$f_i = \sum_{j=1}^{n_e} w_i^j e_i^j, \quad w_i^j > 0 \tag{14}$$

Constraints (12a)-(14) involve only second order terms of the decision variables z_i, w_i^j, e_i^j, and thus can be posed as a BMI.

3.1.2 Contact geometries

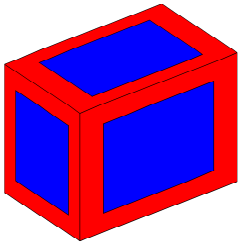

Fig. 3 The polyhedron $\mathcal{P}$ to be grasped. The admissible contact regions are the shrunk regions on each facets (blue region).

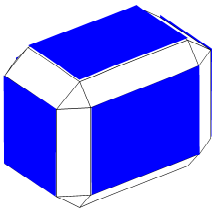

Fig. 4 The shrunk polyhedron $\mathcal{P}_s$ obtained as the convex hull of the blue regions, which are the shrunk regions on each facets as in Fig.3.

Fig. 5 The cylinder to be grasped, the blue surface is the grasp region.

In this section, we consider four types of objects to be grasped, including convex polyhedra (Fig.3,4), spheres, ellipsoids and cylinders (Fig.5). The constraints on contact point x_i and contact normal c_i are straight-forward for the sphere, ellipsoid and cylinder, since the contact surfaces for these geometries are all parameterized by quadratic functions. Thus the constraints on x_i and c_i are also quadratic, and can be solved as BMIs. When the object is a polyhedron, and the grasp is free to choose any facets, the problem becomes trickier to handle, and we will discuss it below.

For a convex polyhedron $\mathcal{P} = \text{ConvexHull}(v_p^1, \ldots, v_p^{N_p})$ (The red box in Fig.3), where v_p^i is the i^{th} vertex of the polyhedron, we want to avoid contacts lying at edges or corners of the polyhedron, since such a grasp can be unstable and the object can slide out of the grasp. Thus the admissible contact regions are given as the shrunk surface regions (blue shades). We then construct a shrunk polyhedron (Fig.4) as the convex hull of the shrunk surface regions (blue shades). The shrunk polyhedron is given as $\mathcal{P}_s = \{x | A_s x \leq b_s\}$; this *H-representation* of a polyhedron can be readily computed from its vertices [24]. To constrain x_i lying on one of the shrunk surface regions, we use the fact that a point is on the surface of a convex object, if and only if a supporting hyperplane intersects the object at that point. Thus we introduce a supporting hyperplane $\mathcal{H}_i = \{x | c_i^T x + d_i = 0\}$, where c_i is the axis of the friction cone, and the constraints:

- The grasp point x_i is on the hyperplane

$$c_i^T x_i + d_i = 0 \tag{15}$$

- All vertices of the original polyhedron $\mathcal{P}$ lie on one side of the hyperplane, and the normal vector c_i points outward from the polyhedron

$$c_i^T v_p^j + d_i \leq 0 \quad \forall j = 1, \ldots, N_p \tag{16}$$

- The grasp point lies within the shrunk polyhedron $\mathcal{P}_s$

$$A_s x_i \leq b_s \tag{17}$$

Geometrically, constraints (15)-(17) state that $c_i^T x + d = 0$ is a supporting hyperplane of the polyhedron $\mathcal{P}$, and the supporting point x_i is not at edges or corners of the polyhedron, so c_i has to coincide with one of the face normals. We want to highlight that we do not specify on which facet the contact lies; by searching over c_i, x_i and d_i, the optimization program will determine the contact facets by itself. Constraints (16)(17) are linear on x_i, c_i and d_i. Constraint (15) is a BMI on x_i and c_i.

3.2 Kinematics

The contact points are meaningful only if they are reachable by the hand , subject to the kinematic constraints. As we will show in this section, such kinematic constraints can also be formulated as BMIs, using robot maximal coordinates.

We illustrate the kinematic chain between two links, welded by a revolute joint as in Fig.6. The orientation of the link frame $i-1, i$ are represented by unit quaternions q_{i-1}, q_i, and the position of frame origins are $p_{i-1}, p_i \in \mathbb{R}^3$ respectively. The transformation from the axis frame i to the link frame $i-1$ is fixed, with a given unit quaternion $z_{i-1,i}$ for the rotation, and a given vector $p_{i-1,i}$ for the translation, both expressed in link frame $i-1$. We introduce two additional variables $\hat{c}_i, \hat{s}_i$, to represent $\cos\frac{\theta_i}{2}, \sin\frac{\theta_i}{2}$ respectively. The rotation of the axis i is thus expressed by the unit quaternion $z_{\theta_i} = \hat{c}_i + \hat{s}_i \mathbf{k}$. The relationship between link frame $i-1$ and i are

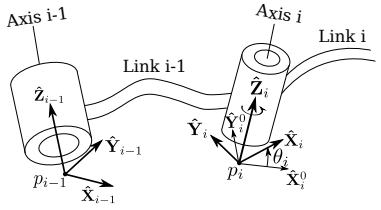

Fig. 6 [6] A link frame $\hat{\mathbf{X}}_{i-1}, \hat{\mathbf{Y}}_{i-1}, \hat{\mathbf{Z}}_{i-1}$ is attached to link $i-1$, link frame $\hat{\mathbf{X}}_i, \hat{\mathbf{Y}}_i, \hat{\mathbf{Z}}_i$ is attached to link i. $\hat{\mathbf{Z}}_{i-1}, \hat{\mathbf{Z}}_i$ are the revolute axes of the joints. The axis frame $\hat{\mathbf{X}}_i^0, \hat{\mathbf{Y}}_i^0, \hat{\mathbf{Z}}_0$ is attached to the joint i, and is fixed in the link frame $i-1$. The axis frame i and the link frame i share the frame origin and $\hat{\mathbf{Z}}_i$ axis, the latter frame is obtained by rotating the former by angle θ_i around the $\hat{\mathbf{Z}}_i$ axis.

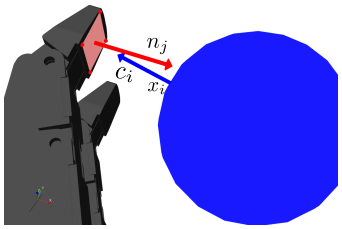

Fig. 7 To grasp the object with a given finger face (the red shaded region with the red dots as vertices) on the link j, the face normal vector n_j must be in the opposite direction of the object surface normal vector c_i, and the face must be in contact with the object at x_i.

$$p_i = R(q_{i-1})p_{i-1,i} + p_{i-1} \tag{18a}$$
$$q_i = q_{i-1} \otimes z_{i-1,i} \otimes z_{\theta_i} \tag{18b}$$
$$q_i \otimes q_i^* = 1, q_{i-1} \otimes q_{i-1}^* = 1 \tag{18c}$$

where $R(q_{i-1})$ is the rotation matrix for unit quaternion q_{i-1}.

The constraints on $\hat{c}_i, \hat{s}_i$ are

$$\hat{c}_i^2 + \hat{s}_i^2 = 1 \tag{19a}$$
$$\hat{c}_i \in \text{range}\left(\cos\frac{\theta_i}{2}\right), \hat{s}_i \in \text{range}\left(\sin\frac{\theta_i}{2}\right),\ \theta_i \in [\underline{\theta}_i,\ \bar{\theta}_i] \tag{19b}$$

where constraint (19a) guarantees that $\hat{c}_i, \hat{s}_i$ are the values of cosine and sine functions of a certain angle, and constraint (19b) encodes the joint limits $[\underline{\theta}_i,\ \bar{\theta}_i]$ for axis i. Constraints (18a)-(19b) encode kinematics chain that welds the adjacent links.

We constrain the hand grasping the object with a given face on link j, as shown in Fig 7. Suppose the vertices of the contact face on link j's are given as $v_1^l, \ldots, v_{n_l}^l$ (the superscript l denotes the point measured in **link** frame), and the finger face touches the object at point x_i on the object (introduced in Section 3.1). By introducing extra variables α_k as convex weights, and v_k^w as the position of the k^{th} vertex in the world frame (the superscript w for **world** frame), we can express x_i as a convex combination of the finger face vertices in the world frame:

$$v_k^w = p_j + R(q_j)v_k^l \tag{20a}$$
$$\sum_{k=1}^{n_l} \alpha_k v_k^w = x_i,\ \sum_{k=1}^{n_l} \alpha_k = 1, \alpha_k \geq 0. \tag{20b}$$

Suppose the unit length face normal vector in link j's link frame is given as n_j^l. Then the face normal vector in the world frame must be the opposite to the object surface normal vector c_i, as below:

$$R(q_j)n_j^l + c_i = 0. \tag{21}$$

With kinematic constraints formulated as linear and bilinear equations in this section, we can solve the inverse kinematics problem by solving BMIs. Furthermore, we can combine the kinematic constraints and the force closure constraints in Section 3.1, to find a force closure grasping posture through sequential SDP.

3.3 Grasp quality optimization

The Q_1 metric proposed by Kirkpatrick [12, 22] measures the smallest magnitude of wrench disturbance that cannot be resisted, given an upper bound on the total contact forces. For contact point $x_i, i = 1, \ldots, n$, and linearized friction cone, whose unit length edges are $e_i^j, j = 1, \ldots, n_e$, we define the wrench set $\mathcal{W}$ as the set of wrench that can be resisted by those contact points, when the total contact forces on all contact points are bounded by 1.

$$\mathcal{W} = \text{ConvexHull}(V_i^j), i = 1,\ldots,n, j = 1,\ldots,n_e, \text{ where } V_i^j = \begin{bmatrix} e_i^j \\ x_i \times e_i^j \end{bmatrix}. \quad (22)$$

If $\mathcal{W}$ contains the origin in the wrench space, then force closure is achieved.

The Q_1 metric is defined as the radius of the largest L_2 ball centered at the origin and being contained in the wrench set $\mathcal{W}$. An L_2 ball is formulated as $\mathcal{B} = \{ w \in \mathbb{R}^6 \big| w^T Q_w w \leq r^2 \}$, where $Q_w \in \mathbb{R}^{6\times 6} \succeq 0$ is a given matrix (usually diagonal), which weights the relative importance between the force disturbance and the torque disturbance. We illustrate the geometric interpretation of Q_1 metric in Fig.8. This cartoon depicts the ball and convex hull in 2D. In the real problem the wrench space has 6 dimensions.

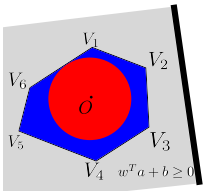

Fig. 8 The largest ball centered at the origin O, being contained in the convex hull spanned by vertices. A necessary and sufficient condition for the ball being contained in the convex hull, is that for any half-space parameterized as $w^T a + b \geq 0$ (the shaded region) that contains all the vertices of the convex hull, such half-space will also contain the ball.

To find the contact points and friction cones, such that their wrench set contains the largest L_2 ball, we employ an iterative procedure. In each iteration for a given L_2 ball radius r, we search contact point x_i and edges of friction cone e_i^j such that $\mathcal{B} \subset \mathcal{W}$ for that r; and then increment r in the next iteration.

To derive the condition on x_i and e_i^j such that $\mathcal{B} \subset \mathcal{W}$ for a given r, we define two cones by appending an extra dimension to $\mathcal{W}$ and $\mathcal{B}$

$$\mathcal{K}_\mathcal{B} = \left\{ \begin{bmatrix} w \\ t \end{bmatrix} \middle| w^T Q_w w \leq r^2 t^2, t \geq 0 \right\}, \quad (23a)$$

$$\mathcal{K}_\mathcal{W} = \sum_{i,j} \lambda_i^j \begin{bmatrix} V_i^j \\ 1 \end{bmatrix}, \lambda_i^j \geq 0, i = 1,\ldots,n, j = 1,\ldots,n_e. \quad (23b)$$

The cross section of cones $\mathcal{K}_\mathcal{B}$ and $\mathcal{K}_\mathcal{W}$ with plane $\left\{ \begin{bmatrix} w \\ t \end{bmatrix} \middle| t = 1 \right\}$ are $\mathcal{B}$ and $\mathcal{W}$ respectively. So the subset relation between $\mathcal{B}, \mathcal{W}$ is equivalent to the subset relation between $\mathcal{K}_\mathcal{B}, \mathcal{K}_\mathcal{W}$

$$\mathcal{B} \subset \mathcal{W} \Leftrightarrow \mathcal{K}_\mathcal{B} \subset \mathcal{K}_\mathcal{W} \Leftrightarrow \mathcal{K}_\mathcal{W}^* \subset \mathcal{K}_\mathcal{B}^*. \quad (24)$$

The second equivalence is based on the fact that for any arbitrary cones $\mathcal{K}_1, \mathcal{K}_2$, $\mathcal{K}_1 \subset \mathcal{K}_2 \Leftrightarrow \mathcal{K}_2^* \subset \mathcal{K}_1^*$, where $\mathcal{K}_1^*$ and $\mathcal{K}_2^*$ are the dual cones of $\mathcal{K}_1, \mathcal{K}_2$ respectively[3].

The formulation of the dual cones $\mathcal{K}_\mathcal{W}^*$ and $\mathcal{K}_\mathcal{B}^*$ are

$$\mathcal{K}^*_{\mathcal{W}} = \left\{ \begin{bmatrix} a \\ b \end{bmatrix} \middle| (V_i^j)^T a + b \geq 0, \forall i = 1,\ldots,n, j = 1,\ldots,n_e \right\} \tag{25a}$$

$$\mathcal{K}^*_{\mathcal{B}} = \left\{ \begin{bmatrix} a \\ b \end{bmatrix} \middle| b^2 \geq r^2 a^T Q_w^{-1} a,\ b \geq 0 \right\}. \tag{25b}$$

The geometric interpretation for $K^*_{\mathcal{W}} \subset K^*_{\mathcal{B}}$ is that for any half-space $\{w | w^T a + b \geq 0\}$ which contains all the vertices $V_i^j, i = 1,\ldots,n, j = 1,\ldots,n_e$, that half-space will also contain the L_2 ball $\mathcal{B}$, as shown in the cartoon Fig.8. So the necessary and sufficient condition on $\mathcal{B} \subset \mathcal{W}$ for a given r is that

$$(V_i^j)^T a + b \geq 0, \forall i, j \Rightarrow \begin{cases} b^2 \geq r^2 a^T Q_w^{-1} a \\ b \geq 0 \end{cases} \tag{26}$$

where $\Rightarrow$ means that the conditions on a, b on the left hand-side implies the relations on a, b on the right hand-side. We simplify this condition further to remove the quadratic term on b. Condition (26) is equivalent to

$$\begin{cases} (V_i^j)^T a + b \geq 0, \forall i, j \\ a^T Q_w^{-1} a = 1 \end{cases} \Rightarrow b \geq r. \tag{27}$$

Condition (27) is a necessary and sufficient condition for $\mathcal{B} \subset \mathcal{W}$. Using the *S-procedure* [19], we write the sufficient condition for (27) as the following algebraic constraints on polynomials, with a, b being indeterminates

$$b - r - L_1(a,b)(a^T Q_w^{-1} a - 1) - \sum_{1 \leq i \leq n} \sum_{1 \leq j \leq n_e} L_2^{i,j}(a,b)((V_i^j)^T a + b) \text{ is SOS} \tag{28a}$$

$$L_2^{i,j}(a,b) \text{ is SOS } \forall i, j \tag{28b}$$

where $L_1(a,b), L_2^{i,j}(a,b)$ are polynomials on indeterminates a, b. SOS stands for *Sum of Squares*, and is a sufficient condition for a polynomial being non-negative. For a polynomial $\alpha(y)$ on indeterminates $y \in \mathbb{R}^k$ with highest order $2d$

$$\alpha(y) \text{ is SOS} \Leftrightarrow \alpha(y) = \Phi(y)^T H \Phi(y), H \succeq 0 \tag{29}$$

where $\Phi(y)$ is the vector containing all monomials of y up to degree d. Thus to search for a non-negative polynomial, it is sufficient to search for the psd matrix H, which ends up with a semidefnite problem on the coefficients of the polynomial. The reader can refer to [19, 23] for more details on *SOS*.

Since constraint (28a),(28b) are sufficient conditions of (27), for x_i and e_i^j satisfying constraints (28a)(28b), r is a lower bound of its Q_1 metric. To maximize r, we can use either bilinear alternation (Algorithm 2) or binary search (Algorithm 3).

In the bilinear alternation, the k^{th} iteration is guaranteed to obtain an objective r that is at least as good as the previous iteration, since a solution to step 2 in iteration k is also feasible for step 1 in both iteration k and $k+1$; also r cannot increase to

Algorithm 2 Bilinear alternation

Start with a force closure grasp x_i, e_i^j, c_i and V_i^j found using approach described in sections 3.1
while $r \neg$ converged **do**

1. At iteration k, fix V_i^j in constraint (28a), search for $L_1(a,b), L_2^{i,j}(a,b)$ and r to maximize r, subject to constraints (28a),(28b). This optimization is a semi-definite programming problem. It finds an L_2 ball contained in the convex hull of V_i^j.
2. Fix $L_2^{i,j}(a,b)$ to the solution in step 1, find feasible $V_i^j, x_i, e_i^j, c_i, L_1(a,b)$ that satisfy (28a) and constraints on x_i, c_i, e_i^j in section 3.1, 3.2. This is a BMI problem. It finds grasp points x_i and friction cone edges e_i^j, such that the grasp quality is no worse than that in the previous iteration. The solution V_i^j will be used in step 1 in the next iteration.

end while

infinity. Hence the bilinear alternation will terminate with convergence of the cost. It is a common strategy in the SDP literature [19, 23, 14].

Algorithm 3 Binary search

Start with $\underline{r} = 0$, and $\bar{r}$ to be some big value, $r = \frac{\bar{r}+\underline{r}}{2}$.
while $r \neg$ converged **do**

1. Fix r, search for the coefficients of $L_1(a,b), L_2^{i,j}(a,b), V_i^j, x_i, e_i^j, c_i$ together, subject to constraints (28a),(28b) and the constraints on x_i, e_i^j, c_i in section 3.1. This is a BMI problem. If the problem converges, set the lower-bound $\underline{r} = r$; otherwise set the upper-bound $\bar{r} = r$.
2. $r = \frac{\bar{r}+\underline{r}}{2}$, go to step 1.

end while

The binary search algorithm needs to deal with psd constraints of larger size than that in bilinear alternation, since it involves the product of $L_2^{i,j}(a,b)$ and V_i^j. Thus the binary search algorithm takes longer time to solve each SDP. Experimentally, we find that the binary search algorithm is less susceptible to local minima than the bilinear alternation alone.

4 Results

4.1 Force closure contact

We show the results of finding force closure contact locations on different geometries in Fig.9. We also show the time scalability w.r.t number of contacts in Fig.10, 11, and number of polyhedron facets in Fig.12. When we increase the number of contacts (Fig.10, 11), the size of the largest psd constraints remains the same, and the number of psd constraints increases linearly. As expected, the computation time in each SDP scales linearly (Fig.11); and empirically we observe that the number of SDP calls remains almost constant (Fig.10). As a result, the total computation time scales linearly w.r.t number of contacts. On the other hand, the number of polyhedron facets does not affect the size or the number of the psd constraints, so the total computation time remains almost constant (Fig.12).

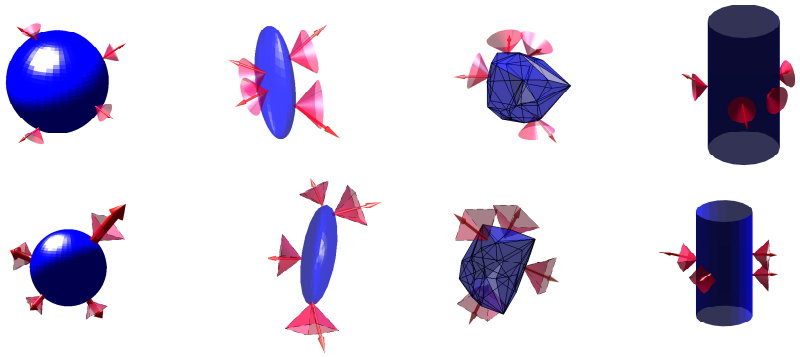

Fig. 9 Force closure contacts on different geometries. The upper row uses nonlinear friction cone, the lower row uses linearized friction cone. For the polyhedron (column 3), the contact facets are not specified by the user beforehand.

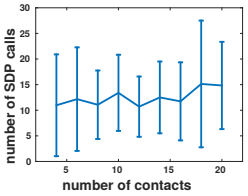

Fig. 10 Scalability w.r.t number of contacts on a 30 facets polyhedron.

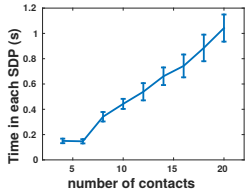

Fig. 11 Scalability w.r.t number of contacts on a 30 facets polyhedron.

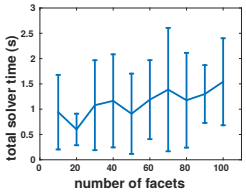

Fig. 12 Scalability w.r.t number of facets, test with 4 contacts.

4.2 Kinematics

The inverse kinematics problem is solved for an ABB IRB140 arm with a Robotiq hand, with 15 joints in total. To evaluate how effective the algorithm is as solving this inverse kinematics problem, in Fig.13 we take 10000 samples within the 0.6m x 0.6m x 0.6m box in the shaded region, and require the center of the palm to reach the sample point. There are three possible outcomes from the sequential SDP.

- Green dot: sequential SDP converges to $t_{opt} = 0 \Rightarrow$ feasible BMIs, thus reachable.
- Red dot: the rank-relaxed SDP reports infeasibility, thus proved unreachable.
- Blue dot: the rank-relaxed SDP is feasible, but the sequential SDP does not converge to $t_{opt} = 0$.

As shown in Fig.13, the blue dot layer is thin, showing that in most cases the sequential SDP algorithm either solves the problem or proves that the problem is infeasible. The histogram in Fig.14 shows that when the sequential SDP can solve the problem, in most (81.36%) cases it converges within 5 SDP calls. The average time to solve the BMI is 0.25 seconds using MOSEK [16] on an Intel i7 machine.

4.3 Grasp optimization

We first show the result of using bilinear alternation to optimize a 3-point force closure grasp on a sphere. The initial contacts and linearized friction cones are plotted in Fig.15, the optimized contacts become more evenly distributed (Fig.16), as is known to be the better 3-point grasp on the sphere [13]. In Fig.17 we draw the Q_1

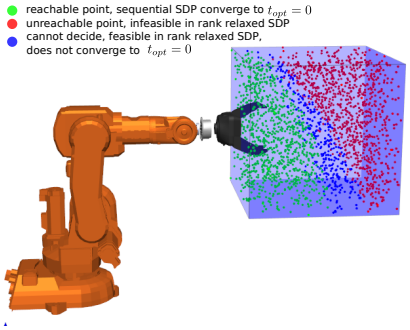

Fig. 13 Robot arm reachability.

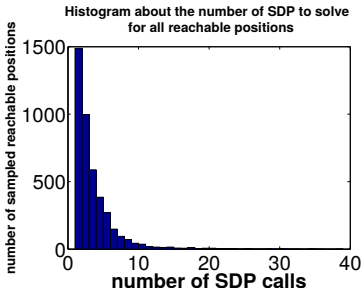

Fig. 14 Histogram on number of SDP calls.

metric in each iteration. The SOS program (28a),(28b) finds a lower bound of the Q_1 metric. The true Q_1 metric is computed as in Appendix. We can see that the gap between the SOS verified lower bound and true Q_1 metric is small. The computation time is 172 seconds using MOSEK solver [16] on an Intel i7 machine.

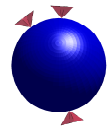

Fig. 15 Initial contacts and friction cones.

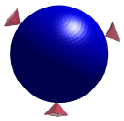

Fig. 16 Optimized contacts and friction cones.

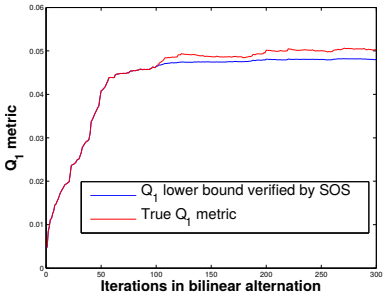

Fig. 17 The change of Q_1 metric in each iteration of bilinear alternation.

We also show the result of optimizing force closure contacts on a diamond shaped polyhedron, through binary search. The optimized contacts (Fig.19) are more evenly distributed than the initial contacts (Fig.18). Also we want to highlight that the facets on which the contacts lie are changed through optimization, this again demonstrates that the optimization program can search over *all* facets by itself. The computation time is around an hour using MOSEK solver on an Intel i7 machine.

We show the result of optimizing the force closure grasp with Robotiq hand and ABB arm on a cylinder. The initial posture grasps the tip of the cylinder (Fig.21), the optimized posture gets improved by grasping the center of the cylinder (Fig.22). The computation time is around 20 minutes using MOSEK on an Intel i7 machine.

5 Conclusion and Discussion

In this paper we exploit the bilinear structure in the force closure and kinematic constraints to synthesize and optimize force closure grasping postures. We do this by formulating the problem as bilinear matrix inequalities (BMIs) and applying the sequential semidefinite programming technique commonly employed in the BMI

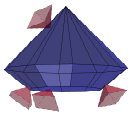

Fig. 18 Initial contacts and friction cones.

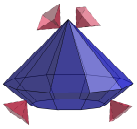

Fig. 19 Optimized contacts and friction cones.

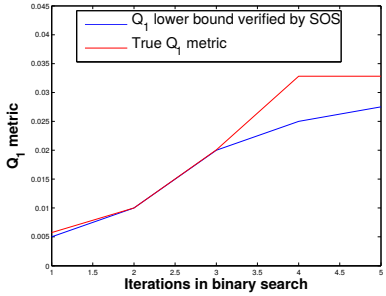

Fig. 20 The change of Q_1 metric in each iteration of binary search.

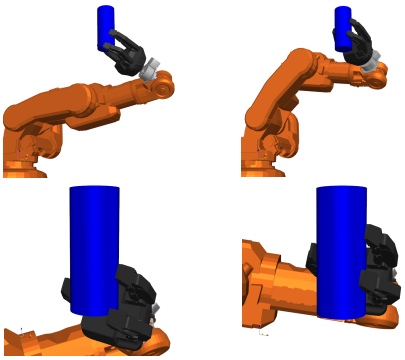

Fig. 21 initial force closure grasp from two views.

Fig. 22 optimized force closure grasp from two views.

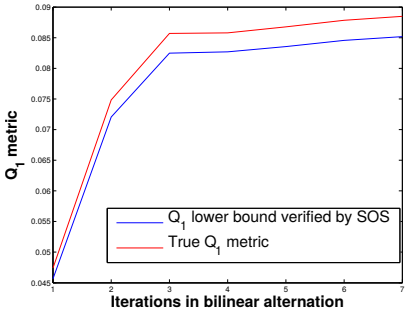

Fig. 23 The change of Q_1 metric in each iteration of bilinear alternation, for robotiq hand grasping the cylinder.

literature. In contrast to more conventional approaches to the problem that rely on gradient based nonlinear optimization, our approach is able to handle non-smooth (such as psd) constraints along with being able to prove *infeasibility* of problems. We demonstrate our results on a 15-joint robot and several types of object geometries.

Some tangible improvements include using the nonlinear friction cone when optimizing force closure grasps, dealing with non-convex polyhedron object, etc.

6 Acknowledgements

We would like to thank Amir Ali Ahmadi for introducing BMI and reference [11]; Alberto Rodriguez for the discussion, and ABB Inc for loaning the IRB140 arm.

7 Appendix

When all contact points x_i and friction cone edges e_i^j are given, we can compute the exact value of Q_1 metric. First we transform representation of the wrench set $\mathcal{W}$ from using vertices V_i^j (*V-representation*) to using half-spaces (*H-representation*) $\mathcal{W} = \left\{w | w^T a_{\mathcal{W}}^i \leq b_{\mathcal{W}}^i,\ i = 1,\ldots,m\right\}$, where m is the number of facets for $\mathcal{W}$. The Q_1 metric is computed as $\min_{i=1,\ldots,m} b_{\mathcal{W}}^i / \sqrt{\left(a_{\mathcal{W}}^i\right)^T Q_w^{-1} a_{\mathcal{W}}}$. Note that we cannot

optimize the Q_1 metric while searching for x_i and e_i^j, since it is nontrivial to transform from *V-representation* to *H-representation* when the vertices are not fixed [24].

References

1. F. Alizadeh and D. Goldfarb. Second-order cone programming. *Mathematical programming*, 95(1):3–51, 2003.
2. Boyd, S.P., Wegbreit, and B. Fast computation of optimal contact forces. *IEEE Transactions on Robotics*, 23(6):1117–1132, 2007.
3. S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004.
4. S. R. Buss. Introduction to inverse kinematics with jacobian transpose, pseudoinverse and damped least squares methods. April 2004.
5. I.-M. Chen and J. W. Burdick. Finding antipodal point grasps on irregularly shaped objects. *Robotics and Automation, IEEE Transactions on*, 9(4):507–512, 1993.
6. J. J. Craig. *Introduction to Robotics: Mechanics and Control*. Pearson Education, Inc, third edition, 2005.
7. H. Dai, A. Valenzuela, and R. Tedrake. Whole-body motion planning with centroidal dynamics and full kinematics. *IEEE-RAS International Conference on Humanoid Robots*, 2014.
8. M. Fazel. *Matrix rank minimization with applications*. PhD thesis, 2002.
9. C. Ferrari and J. Canny. Planning optimal grasps. In *Robotics and Automation, 1992. Proceedings., 1992 IEEE International Conference on*, pages 2290–2295. IEEE, 1992.
10. L. Han, J. C. Trinkle, and Z. X. Li. Grasp analysis as linear matrix inequality problems. *Robotics and Automation, IEEE Transactions on*, 16(6):663–674, 2000.
11. S. Ibaraki and M. Tomizuka. Rank minimization approach for solving bmi problems with random search. In *American Control Conference, 2001. Proceedings of the 2001*, volume 3, pages 1870–1875. IEEE, 2001.
12. D. Kirkpatrick, B. Mishra, and C.-K. Yap. Quantitative steinitz's theorems with applications to multifingered grasping. *Discrete & Computational Geometry*, 7(1):295–318, 1992.
13. G. Liu, J. Xu, X. Wang, and Z. Li. On quality functions for grasp synthesis, fixture planning, and coordinated manipulation. *Automation Science and Engineering, IEEE Transactions on*, 1(2):146–162, 2004.
14. A. Majumdar, A. A. Ahmadi, and R. Tedrake. Control design along trajectories with sums of squares programming. In *Proceedings of the 2013 IEEE International Conference on Robotics and Automation (ICRA)*, 2013.
15. B. Mishra. Grasp metrics: Optimality and complexity. In *Proceedings of the workshop on Algorithmic foundations of robotics*, pages 137–165. AK Peters, Ltd., 1995.
16. A. Mosek. The mosek optimization software. *Online at http://www.mosek.com*, 54, 2010.
17. R. M. Murray, Z. Li, and S. S. Sastry. *A Mathematical Introduction to Robotic Manipulation*. CRC Press, Inc., 1994.
18. V.-D. Nguyen. Constructing force-closure grasps. *The International Journal of Robotics Research*, 7(3):3–16, 1988.
19. P. A. Parrilo. *Structured Semidefinite Programs and Semialgebraic Geometry Methods in Robustness and Optimization*. PhD thesis, California Institute of Technology, May 18 2000.
20. C. Rosales, L. Ros, J. M. Porta, and R. Suárez. Synthesizing grasp configurations with specified contact regions. *The International Journal of Robotics Research*, 30(4):431–443, 2011.
21. E. Salamin. Application of quaternions to computation with rotations. Technical report, Working Paper, 1979.
22. J. D. Schulman, K. Goldberg, and P. Abbeel. Grasping and fixturing as submodular coverage problems. In *International Symposium on Robotics Research*, pages 1–12, 2011.
23. R. Tedrake, I. R. Manchester, M. M. Tobenkin, and J. W. Roberts. LQR-Trees: Feedback motion planning via sums of squares verification. *International Journal of Robotics Research*, 29:1038–1052, July 2010.
24. G. Ziegler. *Lectures on polytopes*, volume 152. Springer Science & Business Media, 1995.