On Geometric Algorithms for Real-Time Grasping Force Optimization

Guanfeng Liu, Jijie Xu, and Zexiang Li, *Member, IEEE*

***Abstract*—Grasping force optimization with nonlinear friction constraints is a fundamental problem in dextrous manipulation with multifingered robotic hands. Over the last few years, by transforming the problem into convex optimization problems on Riemannian manifolds of symmetric and positive definite matrices, significant advances have been achieved in this area. Five promising algorithms: two gradient algorithms, two Newton algorithms, and one interior point algorithm have been proposed for real-time solutions of the problem. In this paper, we present in a unified geometric framework, the derivation of these five algorithms and the selection of step sizes for each algorithm. Using the geometric structure of the affine-scaling vector fields associated with the optimization problem, we prove that some of these algorithms have quadratic convergence properties, and their continuous versions are exponentially convergent. We evaluate the performance of these algorithms through simulation and experimental studies with the Hong Kong University of Science and Technology (HKUST) three-fingered hand. This study will facilitate selection and implementation of grasping force optimization algorithms for similar applications.**

***Index Terms*—Affine-scaling vector fields, convergence analysis, gradient algorithm, interior point algorithm, Newton algorithm, semidefinite programming, step size selection.**

I. Introduction

AUTOMATIC generation of grasping forces is a central problem in dextrous manipulation by multifingered robotic hands. The problem amounts to finding optimal finger forces by minimizing some suitably defined objective functions while respecting constraints associated with physics of contact imposed on the fingers. The difficulty of the problem stems from the fact that both the constraints and the objective functions are nonlinear, load wrench and grasp configurations often change with time and real-time solutions are required. Other applications of the problem include force distributions for walking robots [1], and coordinated control for multiple manipulating arms [2], [3].

Early research in this area includes mathematical models of contact and grasp [4]–[9], and formulation and solution of the grasping force optimization problem using either linear programming-based techniques [5], [1], [10] or nonlinear programming-based techniques [11]–[14]. In the former case, linearization of the friction cones are involved and the computed solutions are, thus, conservative. In the latter case, only offline solutions can be obtained with current hardware platforms and are, thus, impractical.

A major breakthrough in the study of grasping force optimization was made by Buss *et al.* [15]. Based on the important observation that the friction cone constraints are equivalent to positive definiteness of certain symmetric matrices, they transformed the problem into a convex optimization problem in some properly defined Riemannian manifolds with linear constraints, for which several gradient flow type algorithms were developed for real-time computation of optimal grasping forces [16], [17]. However, when these algorithms were applied to time-varying contacts and for systems with a modest number of fingers, the computation task could become excessive. The problem was significantly improved in Li and Qin [18] by splitting the computation into an online and an offline component and exploring block matrix inversion techniques with sparse matrices. For example, the computation time for a two-fingered manipulation with rolling contact was reduced from 3 to 0.08 s on a Motorola 68 040 processor [18]. Han *et al.* [19], [20] further realized that the friction cone constraints can be formulated as linear matrix inequalities (LMIs) and the grasping force optimization problem as a convex optimization problem involving LMIs, with the $\max - \det$ function as the objective function. The interior point algorithm [21], [22] is used to provide efficient solutions to the problem with either fixed or time-varying points of contact, and also for a modest number of fingers. Recently, Helmke *et al.* [23] refined the semidefinite representation of the friction cone constraints in which the structure constraints of [16], [17] are eliminated and the corresponding dimension of the optimization problem is significantly reduced. They proposed an estimation technique and a recursion method for selecting an appropriate step size in the gradient algorithms and proved their quadratic convergence properties.

The work of Han *et al.* [20], Buss *et al.* [15], and Helmke *et al.* [23] leads to five competing algorithms for the grasping force optimization problem. In order to facilitate selection, implementation and application of these algorithms in actual robotic systems, we aim to study in this paper several important aspects of these algorithms. First, after a review of these five algorithms in a common framework, we will address the issue of step size selection in each of the algorithms. Second, as all algorithms require an initial condition that satisfies the friction cone constraints and the force balance equation, we will develop a method for a complete solution of the initial point problem. This make it possible for automatic generation of grasping forces.

Manuscript received December 20, 2002; revised August 18, 2003. Manuscript received in final form February 11, 2004. Recommended by Associate Editor K. Schlacher. This work was supported in part by RGC Grants HKUST 6187/01E, HKUST 6221/99E, and CRC98/01.EG02, and in part by the National Science Foundation, P.R.C., under Grant 50029501.

The authors are with Department of Electrical and Electronic Engineering, Hong Kong University of Science and Technology, Kowloon, Hong Kong (e-mail: liugf@cs.rpi.edu; eexjj@ee.ust.hk; eezxli@ee.ust.hk).

Digital Object Identifier 10.1109/TCST.2004.833630

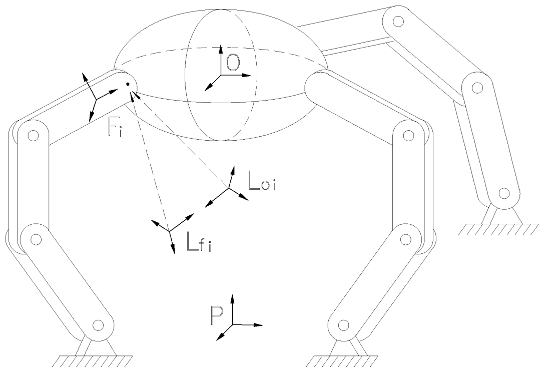


Fig. 1. k-fingered hand grasping an object.

Third, quadratic convergence properties for some of the algorithms will be established. Finally, simulation and experimental studies on the real-time performances and convergence rates of all five algorithms will be performed. The Hong Kong University of Science and Technology (HKUST) three-fingered hand, as shown in Fig. 2 will serve as the common platform for all experiments. The software codes implementing all five algorithms will be made available to the public and can be downloaded from www.ee.ust.hk/~ liugf/software.

The paper is organized as follows. In Section II, we formulate the grasping force optimization problem as a $\max - \det$ programming problem. In Section III, we review two Riemannian metrics on the underlying configuration space and their associated gradient vector fields. In Section IV, we formulate the five candidate algorithms for evaluation in a unified framework, and discuss selection of step sizes. In Section V, we present two methods for automatic generation of initial conditions for the grasping force optimization algorithms. In Section VI, we establish the quadratic convergence properties of some of the algorithms. In Section VII, we incorporate results from the grasping force optimization algorithms for eventual computation of fingertip motions in a multifingered hand manipulation system. In Section VIII, we give simulation and experimental results comparing the performance of the five algorithms. Finally, in Section IX, we conclude the paper with a brief discussion of future work.

II. Grasping Force Optimization as a Max-Det Problem

Consider the task of grasping or manipulating an object using a k-fingered hand, as shown in Fig. 1. Denote by $x = \left[x_1^T, \ldots, x_k^T\right]^T \in \mathbb{R}^{k \cdot l}$, $x_i = [x_{i,1}, \ldots, x_{i,l}]^T \in \mathbb{R}^l$ the finger forces of the hand and $G \in \mathbb{R}^{6 \times kl}$ the grasp map [24]. Static balance of all forces exerted on the object implies that

$$Gx = f_o, \tag{1}$$

where $f_o \in \mathbb{R}^6$ is an external wrench. The physics of contact imposes a nonlinear quadratic constraint on all finger forces. For a point contact with friction, we have $x_i \in \mathbb{R}^3$ and

$$x_{i,1}^2 + x_{i,2}^2 < \mu_i^2 x_{i,3}^2\ ,\ x_{i,3} > 0 \tag{2}$$

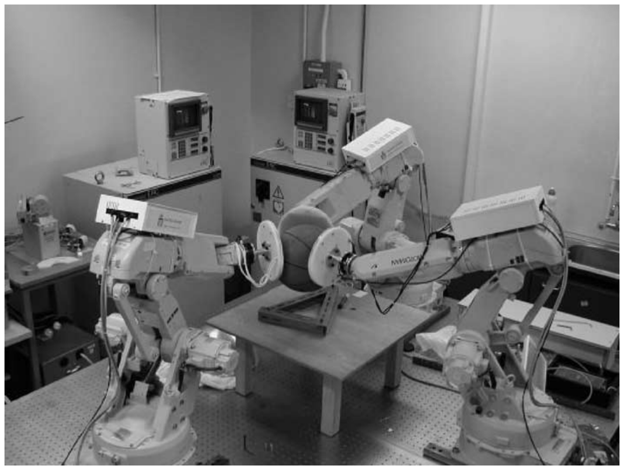


Fig. 2. HKUST 3-fingered hand manipulating a spherical object.

and for a soft finger contact, $x_i \in \mathbb{R}^4$, and

$$\frac{1}{\mu_i^2}(x_{i,1}^2 + x_{i,2}^2) + \frac{1}{\mu_{t,i}^2} x_{i,4}^2 < x_{i,3}^2\ ,\ x_{i,3} > 0. \tag{3}$$

Here, $x_{i,1}$ and $x_{i,2}$ are the tangential force components, $x_{i,3}$ the normal force, and $x_{i,4}$ the moment along the contact normal. μ_i and $\mu_{t,i}$ model the Coulomb friction coefficient and the torsional friction coefficient, respectively, see [15], [17], [20], and [25] for a detailed modeling of grasping statics and friction cone constraints.

Buss *et al.* [15] made an important observation that (2) is equivalent to positiveness of the following symmetric matrix:

$$Q_i = \begin{bmatrix} \mu_i x_{i,3} & 0 & x_{i,1} \\ 0 & \mu_i x_{i,3} & x_{i,2} \\ x_{i,1} & x_{i,2} & \mu_i x_{i,3} \end{bmatrix} > 0 \tag{4}$$

equation (3) to

$$Q_i = \begin{bmatrix} x_{i,3} & 0 & 0 & \frac{1}{\mu_i} x_{i,1} \\ 0 & x_{i,3} & 0 & \frac{1}{\mu_i} x_{i,2} \\ 0 & 0 & x_{i,3} & \frac{1}{\mu_{t,i}} x_{i,4} \\ \frac{1}{\mu_i} x_{i,1} & \frac{1}{\mu_i} x_{i,2} & \frac{1}{\mu_{t,i}} x_{i,4} & x_{i,3} \end{bmatrix} > 0 \tag{5}$$

and the totality of the hand constraints to

$$Q \in \mathbb{R}^{\bar{n} \times \bar{n}} = \operatorname{diag}(Q_1, \ldots, Q_k) > 0 \tag{6}$$

where $\bar{n} = 3k$ or $4k$. Helmke *et al.* [23] refined the constraint representation of (2) to the positive definiteness of the 2×2 matrix

$$P_i = \begin{bmatrix} \mu_i x_{i,3} + x_{i,1} & x_{i,2} \\ x_{i,2} & \mu_i x_{i,3} - x_{i,1} \end{bmatrix} > 0 \tag{7}$$

equation (3) to

$$P_i = \begin{bmatrix} x_{i,3} + \frac{1}{\mu_i} x_{i,1} & \frac{1}{\mu_i} x_{i,2} - j \frac{1}{\mu_{t,i}} x_{i,4} \\ \frac{1}{\mu_i} x_{i,2} + j \frac{1}{\mu_{t,i}} x_{i,4} & x_{i,3} - \frac{1}{\mu_i} x_{i,1} \end{bmatrix} > 0 \tag{8}$$

where $j = \sqrt{-1}$, and the totality of the hand constraints to

$$P \in \mathbb{R}^{n \times n} = \operatorname{diag}(P_1, \ldots, P_k) > 0 \tag{9}$$

where $n = 2k$. Note that the reduction on problem dimension due to this reduced representation of constraints in (7) and (8) is significant. For a two-fingered hand with soft finger contact, $n = 4$ versus $\bar{n} = 8$, and for a three-fingered hand with frictional point contact, $n = 6$ versus $\bar{n} = 9$. Furthermore, the structure constraints on the elements of (4) and (5), and then (6) have been eliminated. We will use representation (9) in the rest of the paper.

Another observation by Han *et al.* [20] is that (9) is of the form LMIs, studied extensively in [21]

$$P(x) = A_{P,0} + A_{P,1}x_1 + \cdots + A_{P,kl}x_{kl} > 0 \qquad (10)$$

with $A_{P,0} = 0$ and a reordering of indexes for the finger forces. The force balance (1) is translated into a set of linear constraints

$$\mathrm{Tr}(B_i P) = f_{oi}\,, \qquad i = 1, \ldots, 6 \qquad (11)$$

where $B_i = B_i^T$ are symmetric k-block diagonal matrices with dimension $n \times n$, f_{oi} the ith component of the external wrench $f_o \in \mathbb{R}^6$. We will assume that the B_i's, $i = 1, \ldots, 6$ are linearly independent, and using the scheme of [23] or the standard Gram–Schmidt process to orthonormalize the B_i's. This will greatly reduce the computational effort in some algorithms to be introduced later. Define the set of admissible finger forces by

$$\Omega_x = \{x \in \mathbb{R}^{kl} \mid P(x) > 0\,,\ \mathrm{Tr}(B_i P) = f_{oi}\,,\ i = 1, \ldots, 6\}.$$

Ω_x is a convex set as it is the intersection of a hyper-plane (convex) with a convex cone. The $\max - \det$ formulation of the grasping force optimization problem is stated as follows.

Problem 1: Max-Det Problem :

$$\begin{aligned} &\min \Phi(P) = \mathrm{Tr}(CP) + \mathrm{logdet} P^{-1} && (12)\\ \text{subject to}\quad &\mathrm{Tr}(B_i P) = f_{oi}\,,\ i = 1, \ldots, 6 && (13)\\ &P > 0 && (14)\end{aligned}$$

or in terms of x as

$$\begin{aligned} &\min \phi(x) = c^T x + \mathrm{logdet} P^{-1}(x) && (15)\\ \text{subject to}\quad &Gx = f_o && (16)\\ &P(x) = A_{P,0} + A_{P,1}x_1 + \cdots + A_{P,kl}x_{kl} > 0 && (17)\end{aligned}$$

where $C \in \mathbb{R}^{n\times n}$ is a constant weighting matrix with the same dimension as that of P and $c = [c_1, \ldots, c_{kl}]^T \in \mathbb{R}^{kl}$ with $c_i = \mathrm{Tr}(CA_{P,i})$. The first (or the linear) term of the objective function $\mathrm{Tr}(CP)$ is used to restrict the normal grasping force because that may destroy the object as well as the fingers. The second term, $\mathrm{logdet} P^{-1}$, tends to infinity as any contact force approaches the boundary of its friction cone and, thus, yields optimal grasp forces interior to their friction cones. Given the optimal solution P^* of either problems, the optimal finger force x^* can be derived accordingly.

III. Riemannian Metrics and Gradient Computation

Denote by S^n the set of $n \times n$ symmetric (or Hermitian) matrices and S^n_{++} the set of symmetric (or Hermitian) positive definite matrices. S^n_{++} is a Riemannian manifold of dimension $n(n+1)/2$ on which the cost function (12) is defined. Existing algorithms for Problem 1 are of the gradient type. To derive the gradient vectors of (12), we will need to specify the Riemannian metrics. Two such metrics are specified as follows. First, note that for all $p \in S^n_{++}$, the tangent space $T_p S^n_{++}$ to S^n_{++} at P is given by S^n, i.e., $T_p S^n_{++} = S^n$. The Euclidean metric on S^n_{++} is defined as

$$\langle \xi, \xi \rangle_P = \mathrm{Tr}(\xi \cdot \xi)\ , \forall \xi \in S^n. \qquad (18)$$

Similar to the hybrid velocity/force control literature [26]–[28], by differentiating the constraints (13), we obtain a decomposition of the total velocity space S^n as

$$S^n = T \oplus T^\perp \qquad (19)$$

where

$$\begin{aligned} T^\perp &= \mathrm{span}\{B_1, \ldots, B_6\}\\ \text{and } T &= \{\eta \in S^n \mid \mathrm{Tr}(B_i \eta) = 0\,,\ i = 1, \ldots, 6\}.\end{aligned}$$

Clearly, the subspace T of dimension $(n(n+1)/2) - 6$ represents the set of "allowable velocities" as in the hybrid control literature. To compute the directional derivative $D\Phi(P)$ of Φ at $P \in S^n_{++}$ in the direction $\xi \in S^n$, we let $Q \in S^n_{++}$ be such that $P = Q^2$. The curve $P(t) = Qe^{Q^{-1}\xi t Q^{-1}}Q$ satisfies $P(0) = P$ and $\dot{P}(0) = \xi$. Thus,

$$\begin{aligned} D\Phi(P)(\xi) &= \frac{d}{dt}\,|_{t=0}\ \Phi(P(t))\\ &= \mathrm{Tr}(C\xi) - \mathrm{Tr}(P^{-1}\xi). \end{aligned} \qquad (20)$$

The gradient vector $\nabla\Phi(P) \in S^n$ is defined as

$$\mathrm{Tr}(\nabla\Phi(P)\xi) = D\Phi(P)(\xi), \forall \xi \in S^n \qquad (21)$$

from which and (20), we obtain

$$\nabla\Phi(P) = C - P^{-1}. \qquad (22)$$

Denote by π the Euclidean projection of S^n onto T along $T^\perp$. We now project $\nabla\Phi(P) \in S^n$ to the constrained subspace T by writing

$$\nabla\Phi(P) = \nabla_T\Phi(P) + \sum_{i=1}^{6} \gamma_i B_i \qquad (23)$$

where $\nabla_T\Phi(P) = \pi(\nabla\Phi(P))$ is the projection of $\nabla\Phi(P)$ to T. Clearly, because of the orthonormality of the B_i's, we have $\gamma_i = \mathrm{Tr}(B_i(C - P^{-1}))$, and

$$\nabla_T\Phi(P) = C - P^{-1} - \sum_{i=1}^{6} \gamma_i B_i. \qquad (24)$$

To calculate the second Frechet derivative of Φ at $P \in S^n_{++}$, we use the curve $P(t) = Qe^{Q^{-1}\eta t Q^{-1}}Q$ that satisfies $P(0) = Q^2 = P$ and $\dot{P}(0) = \eta$. Then,

$$\begin{aligned} D^2\Phi(P)(\xi, \eta) &= \frac{d}{dt}\,|_{t=0}\ D\Phi(P(t))(\xi)\\ &= \mathrm{Tr}(P^{-1}\xi P^{-1}\eta)\ , \forall \xi, \eta \in S^n. \end{aligned} \qquad (25)$$

The cost function (12) is easily shown to be convex as

$$D^2\Phi(P)(\xi,\xi) = \mathrm{Tr}(P^{-1}\xi P^{-1}\xi) > 0\ ,\ \forall \xi \in S^n \neq 0. \quad (26)$$

This allows us to introduce another Riemannian metric on S^n

$$g(P;\xi,\eta) = \mathrm{Tr}(P^{-1}\xi P^{-1}\eta)\ ,\ \forall \xi,\eta \in S^n. \quad (27)$$

The gradient of Φ at P with respect to g is defined as [29]

$$g(P;\mathrm{grad}\Phi(P),\xi) = D\Phi(P)(\xi)\ ,\ \forall \xi \in S^n \quad (28)$$

from which we have

$$\mathrm{grad}\Phi(P) = P\nabla\Phi(P)P = P(C - P^{-1})P.$$

Similar to (23), we project $\mathrm{grad}\Phi(P)$ to T, yielding the constrained gradient

$$T \ni \mathrm{grad}_T\Phi(P) = P(C-P^{-1})P - \sum_{i=1}^{6}\beta_i PB_iP \quad (29)$$

where

$$\beta_i = \Gamma^{-1}\mathrm{Tr}(B_iP(C-P^{-1})P)\ ,\ \Gamma_{i,j} = \mathrm{Tr}(B_iPB_jP).$$

Note that the normal subspace of T under the new metric g translates into

$$T_g^{\perp} = \mathrm{span}\{PB_iP\ ,\ i = 1,\ldots,6\}. \quad (30)$$

It will be clear later that introducing the Riemannian metric g and its respective gradient will be very important when we establish the equivalence between the Newton algorithm and the constrained gradient algorithm.

IV. Algorithms for Grasping Force Optimization and Step Size Selections

The work of Buss *et al.* [15], Buss *et al.* [17], Han *et al.* [20], and Helmke *et al.* [23] lead to five algorithms competing for solutions of Problem 1. The main difference of these algorithms lies in the way that the friction cone constraints are represented and the step sizes are selected. Because of this, the computation efficiency of each algorithm is different. In this section, we formulate these algorithms in a unified framework and discuss selection of step sizes for each algorithm.

A. Dikin-Type Algorithm

Dikin-type algorithm was first applied by Faybusovich [30] to solve matrix linear programming problems. In the current setting, it can be conveniently summarized as [17], [23]

$$P_{k+1} = P_k - \alpha_k \frac{V(P_k)}{\beta} \quad (31)$$

where $V(P)$ denotes the gradient of $\Phi(P)$. Dikin-type algorithm first generates a maximal value $\alpha_{\max}$ such that

$$P_{k+1} > 0\ ,\ \forall \alpha_k \in [0,\alpha_{\max})$$

and then a line searching method is applied to search an $\alpha_k \in [0,\alpha_{\max})$ such that $\Phi(P_{k+1})$ is minimized. $\alpha_{\max}$ can be specified as

$$\max_{P_k - \alpha_k(V(P_k)/\beta) \geq 0} \alpha_k. \quad (32)$$

Lemma 1: If $P \in \mathbb{R}^{n\times n} > 0$, and $\mathrm{Tr}(P^{-1}XP^{-1}X) < 1$ is satisfied for the symmetric $X \in \mathbb{R}^{n\times n}$, then $P - X > 0$.

Proof: Note that

$$P - X = P^{1/2}(I - P^{-(1/2)}XP^{-(1/2)})P^{1/2}. \quad (33)$$

Since

$$\mathrm{Tr}(P^{-1}XP^{-1}X) = \mathrm{Tr}((P^{-(1/2)}XP^{-(1/2)})^2) < 1$$

we have that $\lambda_1^2 + \cdots + \lambda_n^2 < 1$, where λ_i is the ith eigenvalue of $P^{-(1/2)}XP^{-(1/2)}$. Hence, $I - P^{-(1/2)}XP^{-(1/2)} > 0$, and it follows that $P - X > 0$. □

Lemma 2: If $P > 0$, and $\sigma_{\max}(P^{-(1/2)}XP^{-(1/2)}) < 1$ is satisfied for the symmetric X, then $P - X > 0$. Here, $\sigma_{\max}(K)$ is the maximal singular value or the matrix two-norm of K.

Proof: Rewrite $P - X$ as (33). Note that

$$\sigma_{\max}(P^{-(1/2)}XP^{-(1/2)}) < 1$$

implies that $|\ \lambda_{\max}(P^{-(1/2)}XP^{-(1/2)})\ | < 1$, where $\lambda_{\max}(K)$ represents the maximal eigenvalue of K. Consequently, $I - P^{-(1/2)}XP^{-(1/2)} > 0$, or equivalently $P - X > 0$. □

Note that $\alpha_{\max}$ is closely related to β which can be specified in two ways. First, let

$$\begin{aligned}\beta &= \|V(P_k)\|_g = g(P_k;V(P_k),V(P_k))^{1/2}\\ &= \mathrm{Tr}(P_k^{-1}V(P_k)P_k^{-1}V(P_k))^{1/2}\end{aligned}$$

where the metric g is given in (27). Then $\alpha_{\max} = 1$ from $\|\alpha_k(V(P_k)/\beta)\|_g < 1$ and Lemma 1. Second, we introduce metric $\tilde{g}$ as

$$\tilde{g}(P;\xi,\xi) = \sigma_{\max}((P^{-(1/2)}\xi P^{-(1/2)})^2)$$

and let $\beta = \|V(P_k)\|_{\tilde{g}} = \tilde{g}(P_k;V(P_k),V(P_k))^{1/2}$, similarly $\alpha_{\max} = 1$ from $\|\alpha_k V(P_k)/\beta\|_{\tilde{g}} < 1$ and Lemma 2.

Algorithm 1: Dikin-Type Euclidean Gradient Algorithm: Based on the above discussion and adopting the Euclidean gradient (24), we have the following algorithm [30]:

$$P_{k+1} = P_k - \alpha_k^* \frac{\nabla_T\Phi(P_k)}{\|\nabla_T\Phi(P_k)\|_g} \quad (34)$$

where α_k^* is obtained through the line searching method

$$\alpha_k^* = \min_{\alpha_k \in [0,1)} \Phi(P_{k+1}).$$

Algorithm 2: Dikin-Type Riemannian Gradient Algorithm: Adopting the gradient (29) yields Dikin-type Riemannian gradient algorithm [17]

$$P_{k+1} = P_k - \alpha_k^* \frac{\mathrm{grad}_T(P_k)}{\|\mathrm{grad}_T(P_k)\|_g} \quad (35)$$

where a line searching method is applied to find the optimal $\alpha_k^* \in [0,1)$.

Remark 1: The norm $\|,\|_g$ used in (34) and (35) can be replaced by $\|,\|_{\tilde{g}}$.

B. Newton Algorithm

Another method to optimize $\Phi(P)$ is by using the familiar Newton algorithm [22]

$$P_{k+1} = P_k - \alpha_k D_T^2\Phi(P_k)^{-1}\nabla_T\Phi(P_k) \tag{36}$$

where $D_T^2\Phi$ is the restriction of $D^2\Phi$ to T such that $D_T^2\Phi(P_k)^{-1}\nabla_T\Phi(P_k) \in T$. Since $D_T^2\Phi(P_k)$ gives the Riemannian metric on T at P_k, as shown in (25) and (27), i.e.,

$$D_T^2(P_k)(\mathrm{grad}_T\Phi(P_k), \xi) = \mathrm{Tr}(\nabla_T\Phi(P_k)\cdot\xi)\ ,\ \forall\xi\in T$$

we have

$$D_T^2\Phi(P_k)^{-1}\nabla_T\Phi(P_k) = \mathrm{grad}_T\Phi(P_k). \tag{37}$$

Hence, the Newton algorithm (36) is essentially a Riemannian gradient algorithm. Let $\phi(\alpha_k) = \Phi(P_k - \alpha_k \mathrm{grad}_T\Phi(P_k))$. $\phi(\alpha_k)$ is convex as $\Phi(P)$ is convex, a standard result in convex analysis [31]. The maximum step size $\alpha_{\max}$ that ensures $P_{k+1} > 0$ is the smallest positive real root of $\det(P_k - \alpha_k \mathrm{grad}\Phi(P_k)) = 0$. A sufficient and necessary condition for the best step size α_k^* is

$$\phi'(\alpha)\mid_{\alpha_k^*} = 0. \tag{38}$$

Since $\lim_{\alpha\to\alpha_{\max}}\phi(\alpha)\to\infty$, we have $0\le\alpha_k^*\le\alpha_{\max}$. Based on the special property of (38), α_k^* can be estimated in two different ways which lead to two different algorithms, see [23] for the detail of derivation.

Algorithm 3: Newton Algorithm With the Estimated Step Size: One approximation of α_k^* is given by

$$\alpha_k^* = \frac{1+2\lambda(P_k)-\sqrt{1+4\lambda(P_k)}}{2\lambda(P_k)^2} \tag{39}$$

where

$$\lambda(P_k) = \sqrt{\mathrm{Tr}(\nabla_T\Phi(P_k)\mathrm{grad}_T\Phi(P_k))}.$$

Applying the standard Lyapunov-type argument yields the convergence of this algorithm.

Algorithm 4: The Newton Algorithm With the Recursive Step Size Estimation: α_k^* can also be estimated through iteration. We construct a monotone increasing sequence $\alpha_k(-1), \alpha_k(0), \ldots, \alpha_k(l), \ldots$ through the recursion $\alpha_k(l+1) = g(\alpha_k(l))$ such that

$$\begin{aligned}\alpha_k(-1) &= 0\\ \alpha_k(l) &< \alpha_k(l+1),\ l=-1\ldots\\ \phi'(\alpha_k(l)) &< 0,\ l=-1\ldots.\\ g(\alpha_k) &= \alpha_k, \qquad \text{if and only if } \alpha_k = \alpha_k^*.\end{aligned}$$

It is easy to show that such a sequence will converge to a fixed point of g, i.e., $\lim_{l\to\infty}\alpha_k(l)\to\alpha_k^*$. The iteration g can be constructed as follows:

$$\alpha_k(l+1) = \alpha_k(l) - s_k(l)\frac{\phi'(\alpha_k(l))}{\phi"(\alpha_k(l))}$$
$$s_k(l) = \frac{1+2\lambda_k(l)-\sqrt{1+4\lambda_k(l)}}{2\lambda_k(l)^2}, \qquad \lambda_k(l) = -\frac{\phi'(\alpha_k(l))}{\sqrt{\phi''(\alpha_k(l))}}.$$

where $\phi'(\alpha_k(l))$ and $\phi''(\alpha_k(l))$ are the first- and second-order derivatives of the cost function ϕ with respect to α_k

$$\begin{aligned}\phi'(\alpha_k) &= \mathrm{Tr}(C\Delta) - \mathrm{Tr}((P_k+\alpha_k\Delta)^{-1}\Delta)\\ \phi''(\alpha_k) &= \mathrm{Tr}(((P_k+\alpha_k\Delta)^{-1}\Delta)^2)\end{aligned}$$

where $\Delta = -\mathrm{grad}_T\Phi(P_k)$.

In fact, if $\alpha_{\max}$ is known (as given later), then α_k^* can be derived by using the line or bisection searching method.

C. Interior Point Algorithms for the Max-Det Problem Subject to LMIs

An alternative approach to the grasping force optimization problem, given by Han *et al.* [20], is to solve the $\max-\det$ problem of (15). The affine constraints $Gx = f_o$ can be eliminated by substituting $x = x_0 + Vy$ to (16)

$$\tilde{P}(y) = P(x_0+Vy) = \tilde{A}_{P,0} + \tilde{A}_{P,1}y_1 + \cdots + \tilde{A}_{P,kl-6}y_{kl-6} > 0$$

where $x_0 = (x_{0,1},\ldots,x_{0,kl})^T = G^+f_0 = G^T(GG^T)^{-1}f_0$ is a special solution, $V \in \mathbb{R}^{kl\times kl-6}$ a matrix whose columns forming a basis for the null space of G, $\tilde{A}_{P,0} = A_{P,0} + \sum_{i=1}^{kl}A_{P,i}x_{0,i}$, and $\tilde{A}_{P,j} = \sum_{i=1}^{kl}A_{P,i}V_{i,j}$, $1\le j\le kl-6$ with V_{ij} the (ij)th element of V. Under this transformation, (15) is translated to the standard $\max-\det$ problem subject to LMIs [21]:

$$\min_y \phi(y) = e^Ty + \mathrm{logdet}\tilde{P}(y)^{-1} \tag{40}$$
$$\tilde{P}(y) > 0 \tag{41}$$
$$F(y) = F_0 + F_1y_1 + \cdots, F_{kl-6}y_{kl-6} \ge 0 \tag{42}$$

where

$$e_j = \sum_{i=1}^{kl} c_iV_{i,j}\ , \qquad j=1,\ldots,kl-6.$$

Here, we have the freedom of arbitrarily selecting F_j's, $j=0,\ldots,kl-6$. The only requirement is that the matrices $\mathrm{diag}(\tilde{G}_j, F_j)$, $j=1,\ldots,kl-6$, be linearly independent. One particular choice is that $F_0 = 1$ and $F_i = 0, \forall i=1,\ldots,kl-6$, see [32] for details on other related parameters. We consider the central path of the $\max-\det$ problem (40)–(42)

$$\begin{aligned}y^*(t) &= \mathrm{argmin}_{\tilde{P}(y)>0, F(y)>0}\Psi(t,y)\\ \Psi(t,y) &= t(e^Ty + \mathrm{logdet}\tilde{P}^{-1}(y)) + \mathrm{logdet}F^{-1}(y).\end{aligned}$$

The central path $y^*(t)$ will converge to y^*, the optimal solution of (40), as t goes to ∞. We summarize the path-following interior point algorithm as follows, see [21] for a similar algorithm using the primal-dual properties of the problem.

Algorithm 5: Interior Point Algorithm: Input: given strictly feasible y, $t = t_0 \geq 1$, and the tolerance ϵ;

Output: the optimal solution y^* for the $\max - \det$ problem (40):

Step 1) compute $y^*(t)$ using the Newton algorithm;
Step 2) if $\|y^*(t) - y\| < \epsilon$, output $y^* = y^*(t)$;
Step 3) else $y = y^*(t)$ and increase t, go to Step 1.

Remark 2: Problem (40) also provides a way to generate $\alpha_{\max}$ that may be used in the Newton algorithm. This is done through

$$\begin{aligned}\min_{\alpha_k} \phi(\alpha_k) &= -\alpha_k \\ \tilde{P}(\alpha_k) &= 1 > 0 \\ F(\alpha_k) &= P_k - \alpha_k \mathrm{grad}_T \Phi(P_k) \geq 0.\end{aligned}$$

V. Initial Point Algorithms

The algorithms we discussed above still require an initial point x that satisfies $P(x) > 0$, or y that satisfies $\tilde{P}(y) > 0$. Automatic generation of a valid initial condition x or y is crucial for real-time generation solution of the grasping force optimization problem.

A. Han et al. Method

Han *et al.* [20] provided a partial solution to this problem by translating it into another $\max - \det$ problem subject to LMIs

$$\min_z \phi(z) = \tilde{e}^T z \tag{43}$$

$$\tilde{P} = I > 0 \tag{44}$$

$$F(z) := \tilde{A}_{P,0} + \sum_{i=1}^{kl-6} \tilde{A}_{P,j} z_j + I z_{kl-6+1} \geq 0 \tag{45}$$

where $z = [z_1, \ldots, z_{kl-6+1}]^T$ and $\tilde{e} = [0, \ldots, 0, 1]^T$. Note that $\tilde{A}_{P,kl-6+1} = I$ in (45) is a base matrix added artificially. Algorithm 5 can be applied to solve this problem with the following initial condition:

$$z = [0, \ldots, 0, t]^T \;,\; t > -\lambda_{\min}(\tilde{A}_{P,0})$$

where $\lambda_{\min}(\tilde{A}_{P,0})$ denotes the minimum eigenvalue of $\tilde{A}_{P,0}$. Once the optimal value of $\phi(z) = z_{kl-6+1} < 0$, the vector $z^* = [z_1, \ldots, z_{kl-6}]^T$ is then a valid initial point for (40), as seen from (45). In [25], we gave an example showing that this method can sometimes suffer from singularity problems and proposed the following gradient method when the system is singular.

B. Gradient Method

In (41), the matrix $\tilde{P}(y)$ is a linear combination of constant base matrices, a distinct property based on which we can derive the gradient flow of its minimal eigenvalue [25]. Let $\lambda_n(\tilde{P}(y))$ be the minimal eigenvalue of $\tilde{P}(y)$.

Theorem 1: Gradient Flow of $\lambda_n(\tilde{P}(y))$: The monotone increasing flow of $\lambda_n(\tilde{P}(y))$ is given by

$$\dot{y} = \nabla_y \lambda_n(\tilde{P}(y)) = \begin{bmatrix} \omega_y^T \tilde{A}_{P,1} \omega_y \\ \vdots \\ \omega_y^T \tilde{A}_{P,kl-6} \omega_y \end{bmatrix}$$

where ω_y is the unit eigenvector of $\tilde{P}(y)$ corresponding to the minimal eigenvalue.

Proof: See [25]. □

Algorithm 6: Gradient Algorithm for a Valid Initial Condition:

$$y(k+1) = y(k) + \alpha_k \nabla_y \lambda_n(\tilde{P}(y)).$$

The performance of this algorithm depends on the step size α_k.

VI. Convergence Analysis

Denote by

$$\Omega_P = \{P \mid P > 0 \;,\; \mathrm{Tr}(B_i P) = f_{oi} \;,\; i = 1, \ldots, 6\}$$

the domain of Problem 1 which is assumed to be bounded in all of the following discussions. Since both $\Phi(P)$ and Ω_P are convex, Problem 1 is termed a convex optimization problem, and possesses a unique optimal solution, denoted $P^* \in \Omega_P$ [31]. It is not difficult to prove that

Theorem 2: P^* is an optimal solution of Problem 1 if and only if

$$\nabla_T \Phi(P^*) = 0 \qquad \text{or} \qquad \mathrm{grad}_T \Phi(P^*) = 0.$$

Proof: Since $\mathrm{logdet} P^{-1} \to \infty$ when P goes to the boundary of Ω_P, the unique optimal solution $P^* \in \Omega_P$. Hence, at this point we have

$$\nabla \Phi(P^*) = C - {P^*}^{-1} \in T^{\perp} \tag{46}$$

or equivalently

$$\nabla_T \Phi(P^*) = 0.$$

The equivalence between $\nabla_T \Phi(P^*) = 0$ and $\mathrm{grad}_T \Phi(P^*) = 0$ can be established according to (29) and (30)

$$\begin{aligned}\nabla \Phi(P^*) \in T^{\perp} &\Leftrightarrow P^* \nabla \Phi(P^*) P^* \in T_g^{\perp} \\ &\Leftrightarrow \mathrm{grad}_T \Phi(P^*) = 0.\end{aligned}$$

□

Algorithm 2, 3, 4 can be conveniently summarized in a unified form

$$P_{k+1} = f(P_k) = P_k - \beta_k \mathrm{grad}_T \Phi(P_k) \;,\; \beta_k > 0 \tag{47}$$

which satisfies the following two remarkable common properties: (1) $\Phi(P_{k+1}) < \Phi(P_k)$ if $\mathrm{grad}_T \Phi(P_k) \neq 0$; (2) f has a unique fixed point P^* satisfying $\mathrm{grad}_T \Phi(P^*) = 0$. The only difference lies in the selection of the step size β_k. Note that if we replace $\mathrm{grad}_T \Phi(P_k)$ in (47) by $\nabla_T \Phi(P_k)$, we obtain Algorithm 1. These two important properties allow us to prove the following result.

Theorem 3: By Algorithm 1–4

$$\lim_{k\to\infty} P_k \to P^*.$$

Proof: Note that the sequence $\Phi(P_k), k = 1, \ldots$ is monotone decreasing and bounded from below. Therefore

$$\lim_{k\to\infty} \Phi(P_k) \to \Phi_0 \geq \Phi(P^*).$$

Let

$$S_0 = \{P \in \Omega_P \mid \Phi(P) = \Phi_0\}.$$

If $\Phi_0 > \Phi(P^*)$, S_0 can not be the invariant set under (47) as

$$\Phi(f(P)) < \Phi(P) \;\; \forall P \in S_0$$

according to the second property of (47). Therefore

$$\lim_{k\to\infty} \Phi(P_k) \to \Phi(P^*).$$

Second, we see that

$$\mathcal{T}_k = \{P \in \Omega_P \mid \Phi(P) \leq \Phi(P_k)\}$$

constitutes a set of closed neighborhoods of P^* which satisfy

$$\mathcal{T}_1 \supset \mathcal{T}_2 \supset \cdots \supset \mathcal{T}_i \supset \cdots.$$

There is only one point P^* contained in all these sets. A standard result from real analysis shows that

$$\lim_{k\to\infty} P_k \to P^*.$$

□

This will become more clear if we consider the continuous version of (47)

$$\dot{P} = -\beta \text{grad}_T \Phi(P) \tag{48}$$

with β being assumed to be constant, e.g., $\beta = 1$. Since (48) is a first-order O.D.E., we will obtain a unique integral curve $\gamma(t)$ given an initial point $\gamma(0) = P_0 \in \Omega_P$. Define

$$h(P) = \pi(P^{-1}),\; P \in \Omega_P.$$

h can be proved to be a diffeomorphism from Ω_P to T [33]. Moreover, $h(\gamma(t))$ satisfies the following first-order O.D.E.:

$$\begin{aligned} \dot{h} &= \beta\pi(P^{-1}\text{grad}_T\Phi(P)P^{-1}) = \beta(\pi(C) - h) \\ h(0) &= \pi(P_0^{-1}). \end{aligned}$$

Therefore, the integral curve $h(\gamma(t))$ is given by

$$h(\gamma(t)) = e^{-\beta t}(h(P_0) - \pi(C)) + \pi(C)$$

and

$$\gamma(t) = h^{-1}(e^{-\beta t}(h(P_0) - \pi(C)) + \pi(C)). \tag{49}$$

since $\pi(C) = \pi({P^*}^{-1}) = h(P^*)$ as seen from (46), we have

$$\lim_{t\to\infty} \gamma(t) = h^{-1}(\pi(C)) = P^*. \tag{50}$$

Previous discussion shows the convergence of Algorithm 1–4. The same result of Algorithm 5 can be found in [21]. However, this is not complete. Experimental results [15], [17], [20] have shown that some algorithms based on Problem 1 have better convergence rates than those nonlinear programming algorithms [11]. This motivates us to analyze the convergence rates of these five algorithms. There are two quantities by which we can measure how much P deviates from P^*. The first one is the distance between P and P^*

$$d_1(P_k, P^*) = (\text{Tr}(P_k - P^*)^2)^{1/2}.$$

S^n_{++} becomes a metric space by defining d_1. The other is the length of the gradient vector

$$d_2(P_k) = (\text{Tr}(P_k^{-1}\text{grad}_T\Phi(P_k)P_k^{-1}\text{grad}_T\Phi(P_k)))^{1/2}$$

as implied by Theorem 2. Helmke *et al.* [23] used d_1 to prove the quadratic convergence of Algorithm 3 and 4. However, their approach relies on a condition that the differential Df is Lipschitz continuous in a neighborhood of P^* that requires a quite complicated proof. Here, d_2, with its beautiful geometric structure, is utilized to prove the convergence of Algorithm 2–4. Let $W(C, B_1, \ldots, B_6)(P)$ be the affine scaling vector fields [33]

$$W(C, B_1, \ldots, B_6)(P) = \text{grad}_T\Phi(P)$$

because $\text{grad}_T\Phi(P)$, as given in (29), explicitly depends on C and $B_i, i = 1, \ldots, 6$. Consider the action of $GL(n)$ on S^n_{++}

$$GL(n) \times S^n_{++} \to S^n_{++} : (E, P) \to EPE^T.$$

This action is transitive and isometric on S^n_{++}.

Proposition 1: With the Riemannian metric defined as (27), the group $GL(n)$ acts on S^n_{++} by isometries.

Proof: Under the action of $GL(n)$, $P \in S^n_{++} \to EPE^T$, $\xi \in TS^n_{++} = S^n \to E\xi E^T$, and $\eta \in TS^n_{++} \to E\eta E^T$. From (27), we have

$$\begin{aligned} &g(EPE^T; E\xi E^T, E\eta E^T) \\ &\quad = \text{Tr}((EPE^T)^{-1}E\xi E^T(EPE^T)^{-1}E\eta E^T) \\ &\quad = \text{Tr}(P^{-1}\xi P^{-1}\eta) = g(P, \xi, \eta). \end{aligned}$$

□

The affine scaling vector field W processes another important property:

$$W(C, B_1, \ldots, B_6)(EPE^T) = EW(\tilde{C}, \tilde{B}_1, \ldots, \tilde{B}_6)(P)E^T \tag{51}$$

where $\tilde{C} = E^TCE$, and $\tilde{B}_i = E^TB_iE, i = 1, \ldots, 6$. This can be shown through replacing P in (29) by EPE^T and simplifying the left-hand side of (51).

Combining the above two results, we are able to calculate $d_2(P_k) = g(P_k; W(C, B_1, \ldots, B_6)(P_k), W(C, B_1, \ldots, B_6)(P_k))^{1/2}$.

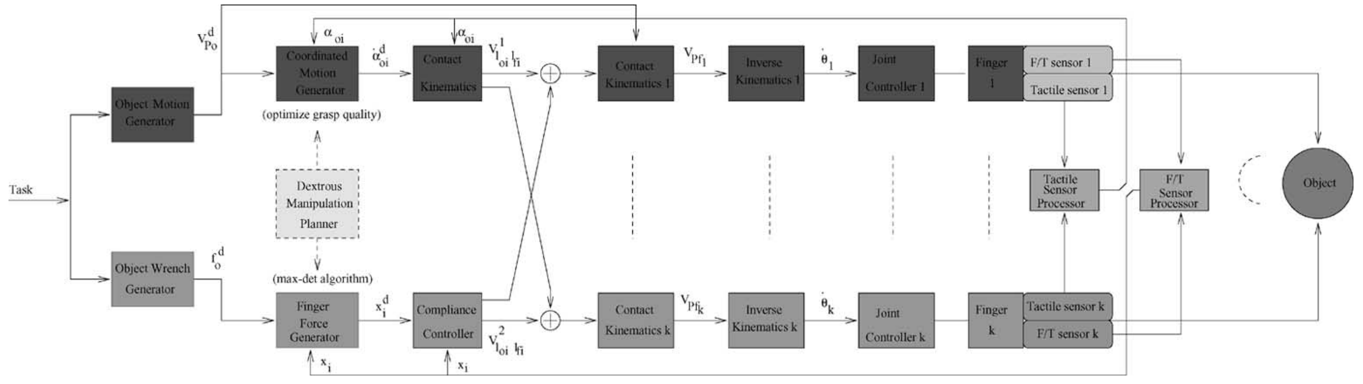


Fig. 3. Control diagram for a k-fingered hand manipulating an object.

Proposition 2:

$$d_2(P_k) = \min\left\{ \left\| \tilde{C} - I - \sum_{i=1}^{6} a_i \tilde{B}_i \right\|_F \mid a_i \in \mathbb{R} \right\}$$

where $\tilde{C} = P_k^{1/2} C P_k^{1/2}$ and $\tilde{B}_i = P_k^{1/2} B_i P_k^{1/2}$. $\|\cdot\|_F$ denotes the Frobenius norm.

Proof: Let $E = P_k^{1/2}$, by (51) we have

$$W(C, B_1, \ldots, B_6)(P_k) = P_k^{1/2} W(\tilde{C}, \tilde{B}_1, \ldots, \tilde{B}_6)(I) P_k^{1/2}.$$

Then, by Proposition 1, we have

$$\begin{aligned} &d_2(P_k) \\ &\quad = g(I, W(\tilde{C}, \tilde{B}_1, \ldots, \tilde{B}_6)(I), W(\tilde{C}, \tilde{B}_1, \ldots, \tilde{B}_6)(I))^{1/2}. \end{aligned}$$

Since

$$\begin{aligned} W(\tilde{C}, \tilde{B}_1, \ldots, \tilde{B}_6)(I) &= \tilde{C} - I - \sum_i \beta_i \tilde{B}_i \\ &= \pi(\tilde{C} - I) \end{aligned}$$

and the Riemannian metric at identity degenerates to the trivial metric, we have

$$d_2(P_k) = (\mathrm{Tr}(\pi(\tilde{C} - I)\pi(\tilde{C} - I)))^{1/2}$$

the results follows. □

With these preparations, we are able to prove the main result about the convergence of Algorithm 2–4.

Theorem 4: Under the recursion

$$P_{k+1} = P_k - \mathrm{grad}_T \Phi(P_k)$$

we have

$$d_2(P_{k+1}) \leq d_2^2(P_k).$$

Proof: See Appendix A. □

Remark 3: If $d_2(P_0) < 1$, we conclude that Algorithms 2, 3, and 4 have at least quadratic convergence rates since we have searched a better step size δ than just 1 in

$$P_{k+1} = P_k - \delta \mathrm{grad}_T \Phi(P_k).$$

Remark 4: As for the continuous version (48) is considered, $\gamma(t)$ converges exponentially to the optimal solution P^* as seen from (49) and (50).

VII. Hand Kinematics and Control

Based on the results of previous sections, we propose in this section a control algorithm for a multifingered robotic hand to manipulate an object from an initial configuration to a desired final configuration without dropping it. The control objectives to be achieved are desired motions of the object, optimal contact configurations and desired optimal grasping forces computed by the grasping force optimization algorithms. The control inputs to be specified are the velocities of the fingertips, which are in turn realized by some well-known joint-level control algorithms. See [18] and Fig. 3 for the control system architecture of the HKUST hand. In the following, we will derive the fingertip velocities from the velocities of the object, desired motion of the contact points, and the desired finger forces.

As shown in Fig. 1, we define the following coordinate frames: P is the palm frame fixed to the hand palm, and O the frame fixed to the center of the mass of the object. For each finger in the hand, attach a frame F_i to the fingertip, and local frames L_{oi} and L_{fi} to the object and the ith finger at the point of contact, respectively. Please refer to [24] and [8] for further notations and the various kinematic relations about the multifingered manipulation system.

Denote g_{ab} the forward kinematics map of frame b relative to frame a. The forward kinematics of the ith finger-object system is expressed as

$$g_{Po} = g_{Pf_i} g_{f_i l_{f_i}} g_{l_{f_i} l_{o_i}} g_{l_{o_i} o}, \qquad i = 1, \ldots, k \tag{52}$$

where $g_{f_i l_{f_i}}, g_{l_{o_i} o} \in \mathrm{SE}(3)$ are constant transformations. Differentiating (52) yields

$$Ad_{g_{l_{o_i} o}} V_{Po} = Ad_{g^{-1}_{f_i l_{o_i}}} V_{Pf_i} - Ad_{g_{l_{o_i} l_{f_i}}} V_{l_{o_i} l_{f_i}}, \qquad i = 1, \ldots, k$$

where $V_{l_{o_i} l_{f_i}} \in \mathbb{R}^6$ is the contact velocity of finger i with respect to the object. To simultaneously control finger forces and contact locations, we split the contact velocity $V_{l_{o_i} l_{f_i}}$ into two components $V^1_{l_{o_i} l_{f_i}}$ and $V^2_{l_{o_i} l_{f_i}}$ as follows. First, $V^1_{l_{o_i} l_{f_i}}$ is specified along the direction of contact coordinates variation through

Montana's kinematics of contact. For a frictionless point contact model, we have

$$V^1_{l_{o_i}l_{f_i}} = \begin{bmatrix} R_{\psi_i}M_{oi} & -M_{fi} & 0 \\ 0 & 0 & 0 \\ R_{\psi_i}R_cK_{oi}M_{oi} & -R_cK_{fi}M_{fi} & 0 \\ -T_{oi}M_{oi} & -T_{fi}M_{fi} & 1 \end{bmatrix} \dot{\eta}_i$$

for a point contact with friction model, we have

$$V^1_{l_{o_i}l_{f_i}} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ -R_c(K_{fi}+R_{\psi_i}K_{oi}R_{\psi_i})M_{fi} & 0 \\ -(T_{fi}R_{\psi_i}M_{oi}+T_{oi}M_{oi}) & 1 \end{bmatrix} \begin{bmatrix} \dot{\alpha}_{oi} \\ \dot{\psi}_i \end{bmatrix}$$

and for a soft finger contact model, we have

$$V^1_{l_{o_i}l_{f_i}} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ -R_c(K_{fi}+R_{\psi_i}K_{oi}R_{\psi_i})M_{fi} \\ 0 \end{bmatrix} \dot{\alpha}_{oi}.$$

Here, following the notation in [24], a contact configuration of finger i is described by $\eta_i = (\alpha^T_{oi}, \alpha^T_{fi}, \psi_i)^T$. $\alpha_{oi} = (u_{oi}, v_{oi})^T$ and $\alpha_{fi} = (u_{fi}, v_{fi})^T$ are the local coordinates of contact relative to the object and the finger, respectively. ψ_i denotes the contact angle. (M_{oi}, K_{oi}, T_{oi}) and (M_{fi}, K_{fi}, T_{fi}) are, respectively, the geometric parameters of the object and the finger at the ith point of contact, and R_{ψ_i} and R_c are given by

$$R_{\psi_i} = \begin{bmatrix} \cos\psi_i & -\sin\psi_i \\ -\sin\psi_i & -\cos\psi_i \end{bmatrix} \quad \text{and} \quad R_c = \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix}.$$

Note that the rate of change of the contact coordinates can be specified by minimizing some grasp quality functions as in [18], [34], and [35]. A general design methodology for grasp quality measures is given in [36]. The second component of $V^2_{l_{o_i}l_{f_i}}$ of the contact velocity is chosen to be perpendicular to $V^1_{l_{o_i}l_{f_i}}$ and is used to regulate finger force $x_i \in \mathbb{R}^l$ through a compliance control scheme $V^2_{l_{o_i}l_{f_i}} = C_ix_i$, where $C_i \in \mathbb{R}^{6\times l}$ is a compliance matrix.

Summarizing our discussion, we propose the following control algorithm for the desired finger velocity:

$$\begin{aligned} V^d_{Pf_i} =& Ad_{g_{f_io}}(\eta_i)V^d_{Po} \\ &+ Ad_{g_{f_il_{f_i}}}(\eta_i)(V^1_{l_{o_i}l_{f_i}}(\eta_i, \dot{\eta}^d_i) + C_i(x^d_i - x_i)) \end{aligned} \quad (53)$$

where $\dot{\eta}^d_i$ is the desired rate of change of the contact coordinates, as specified by minimizing some grasp quality functions, x^d_i is the desired finger force computed from the grasping force optimization algorithm. In real implementation, α_{fi} is directly measured through tactile sensors, and the remaining components of η_i are computed by inverting Montana's kinematic equations of contact. x_i is obtained through force/torque sensors integrated with the fingers (see Fig. 2). Fig. 3 shows the block-diagram of this controller.

TABLE I
COMPARISON OF REAL-TIME PERFORMANCE OF THE FIVE ALGORITHMS UNDER TWO DIFFERENT REPRESENTATIONS OF FRICTION CONE CONSTRAINTS

Optimization algorithms	Buss, Moore and Hashimoto (1996)	Helmke, Hueper and Moore (2002)
Algorithm 1	11s/1000 times	2s/1000 times
Algorithm 2	21s/1000 times	4s/1000 times
Algorithm 3	26s/1000 times	3s/1000 times
Algorithm 4	22s/1000 times	3s/1000 times
Algorithm 5	5s/1000 times	2s/1000 times

TABLE II
COMPARISON OF REAL-TIME PERFORMANCE OF THE FIVE ALGORITHMS FOR GRASPING FORCE OPTIMIZATION

Optimization algorithms	Used Tolerance	Sun Ultra60 and Unix	Motorola 68040 and VxWorks
Algorithm 1	$\mathrm{Tr}((P(k+1)-P(k))^2) < 1.0$	2s/1000 times	9s/100 times
Algorithm 2	$\mathrm{Tr}((P(k+1)-P(k))^2) < 1.0$	4s/1000 times	10s/100 times
Algorithm 3	$\mathrm{Tr}((P(k+1)-P(k))^2) < 1.0$	3s/1000 times	10s/100 times
Algorithm 4	$\mathrm{Tr}((P(k+1)-P(k))^2) < 1.0$	3s/1000 times	9s/100 times
Algorithm 5	$\mid \Phi(P(k+1)) - \Phi(P(k)) \mid < 1.0$ and $\mid \frac{\Phi(P(k+1))-\Phi(P(k))}{\phi(P(k))} \mid < 2.0$	2s/1000 times	9s/100 times

VIII. SIMULATION AND EXPERIMENTAL RESULTS: A COMPARATIVE STUDY

In this section, we perform simulation and experiments to evaluate the real-time performance of the proceeding algorithms for grasping force optimization.

A. Simulation Results

We first apply the five algorithms to a 3-fingered hand grasping a spherical object (see Fig. 2) and compare their computation time from a given initial force to the optimal grasping force under given tolerances. To show how the different representations of the friction cone constraints affect the real-time performance of the optimization algorithms, we use two different representations of the friction cone constraints, the Bush, Moore, and Hashimoto (BHM) representation [15] and the Helmke, Hueper, and Moore (HHM) representation [23]. The simulation results are shown in Table I. From Table I, we conclude that using the HHM representation, we can reduce the computation time in Algorithm 1–4 to almost one fifth that of the BHM representation, and in Algorithm 5 to about one half. This is because we only increase the dimension of the base matrices to which the interior point algorithm is not sensitive, but not the number of independent variables (finger forces). Moreover, we implement the five algorithms on two simulation systems with different setups to show how the external environment, such as tolerances, operation systems and processors, affect the real-time performance of the algorithms. The simulation results are shown in Table II. We conclude from this study that among the five algorithms, Dikin-type Euclidean gradient algorithm (Algorithm 1) and the interior point algorithm (Algorithm 5) have obvious advantage for real-time applications. It is possible for slow algorithms to have

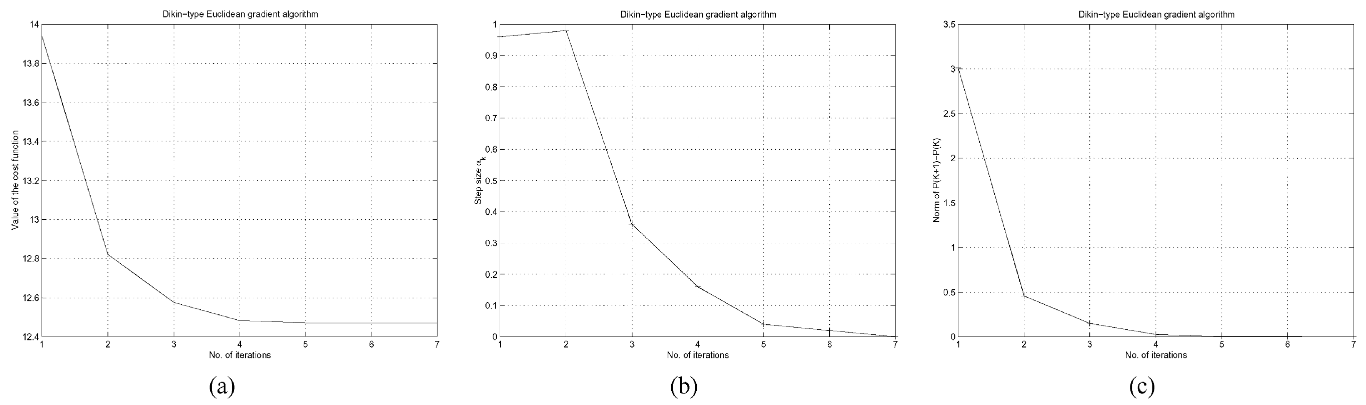


Fig. 4. Algorithm 1. (a) Trajectory of the cost function. (b) Trajectory of the best step size. (c) Trajectory of the error $\mathrm{Tr}((P(k+1) - P(k))^2)$.

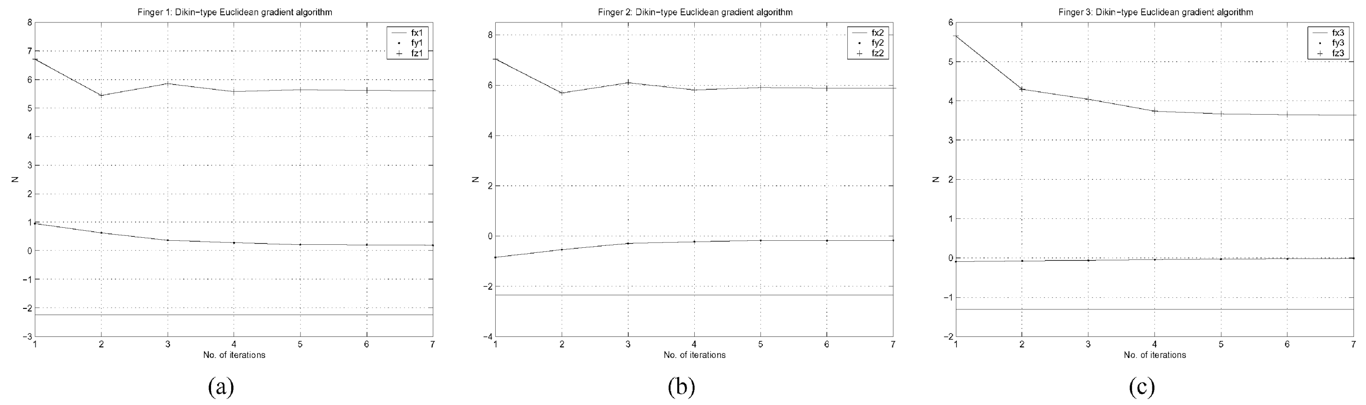


Fig. 5. Algorithm 1. (a) Trajectory of the forces of finger 1. (b) Trajectory of the forces of finger 2. (c) Trajectory of the forces of finger 3.

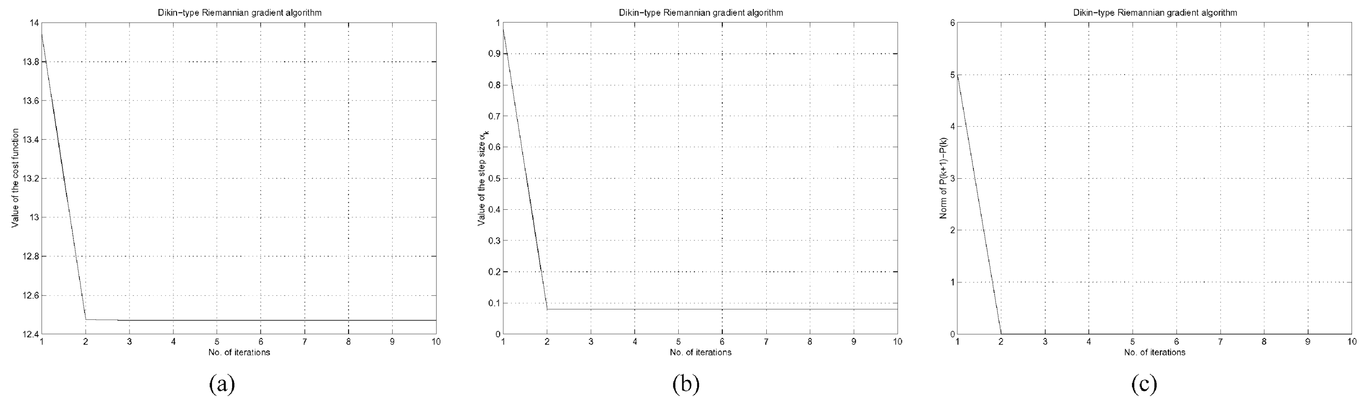
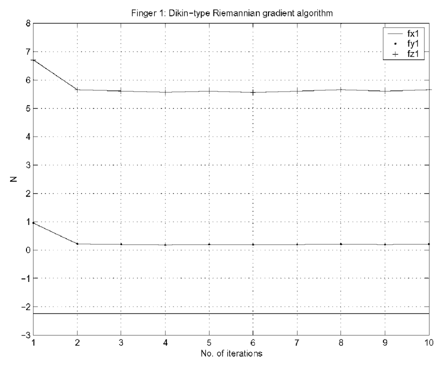


Fig. 6. Algorithm 2. (a) Trajectory of the cost function. (b) Trajectory of the best step size. (c) Trajectory of the error $\mathrm{Tr}((P(k+1) - P(k))^2)$.

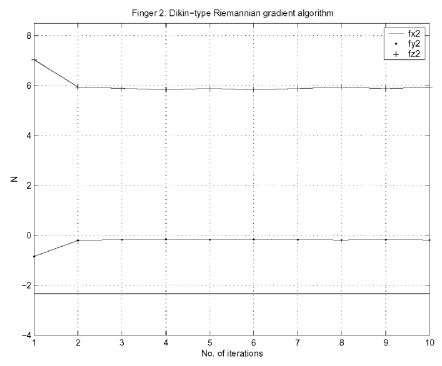
less iterations because the speed also relies on the computation time of each iteration.

It should be noted that the computation time in the last column of Table I was recorded in the experiments where the object was manipulated from an initial point to a final one. It also includes the part on kinematics computation and trajectory generation, which highly depends on the used kinematics and the planning algorithms.

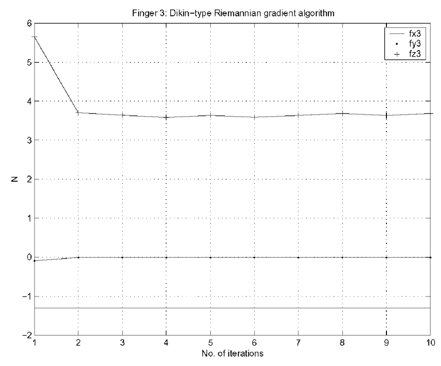
Second, we like to understand that under the same conditions (tolerances, external forces, contact coordinates, cost functions and initial finger forces), whether all these five algorithms will converge to the same optimal solutions. The simulation results obtained using the Dikin-type Euclidean gradient algorithm are shown in Figs. 4 and 5, those using the Dikin-type Riemannian gradient algorithm in Figs. 6 and 7, those using the Newton algorithm with the estimated step size in Figs. 8 and 9 , those using the Newton algorithm with the recursive step size estimation in Figs. 10 and 11, and those by using the interior point algorithm in Figs. 13 and 14. Comparing these figures, we conclude that using these five algorithms, (1) the five cost functions and the finger forces converge to the same optimal values ; (2) the number of iterations it takes each algorithm to achieve the optimal solutions, so-called the convergence rate of each algorithm, are different. As shown in Table III, Algorithms 2, 4 and, 5 have the best convergence rates. (3) the convergence rate is one but not the only factor, which affects the real-performance of the optimization algorithms. Comparing the results in Tables II and III, it is obvious that Algorithm 2 has be best convergence rate,



(a)

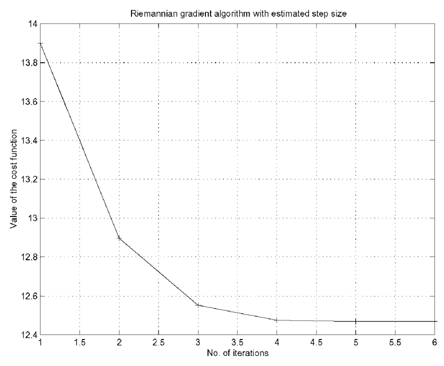


(b)

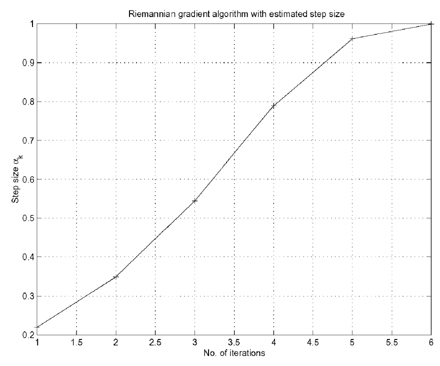


(c)

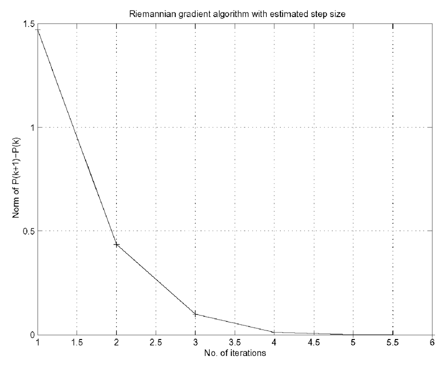
Fig. 7. Algorithm 2. (a) Trajectory of the forces of finger 1. (b) Trajectory of the forces of finger 2. (c) Trajectory of the forces of finger 3.



(a)

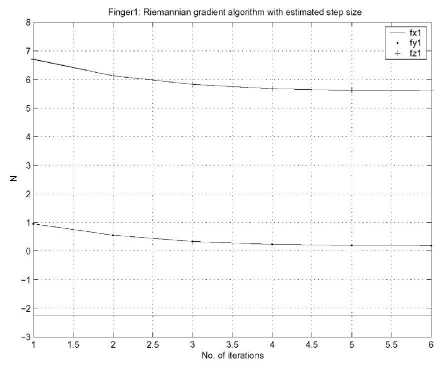


(b)

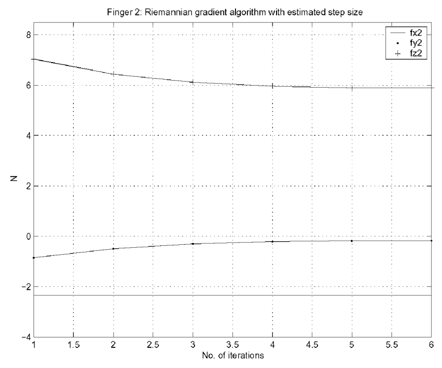


(c)

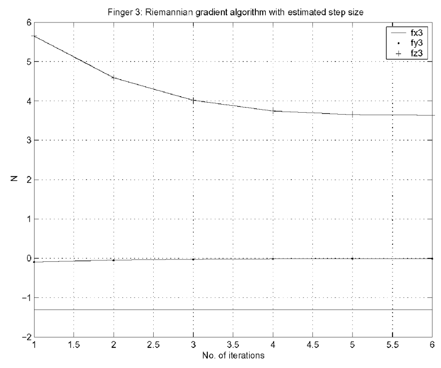
Fig. 8. Algorithm 3. (a) Trajectory of the cost function. (b) Trajectory of the best step size. (c) Trajectory of the error $\mathrm{Tr}((P(k+1) - P(k))^2)$.



(a)



(b)



(c)

Fig. 9. Algorithm 3. (a) Trajectory of the forces of finger 1. (b) Trajectory of the forces of finger 2. (c) Trajectory of the forces of finger 3.

but its real-time performance is not the best for both BHM and HHM representation. The optimal step sizes shown in Figs. 4(b) and 6(b) are obtained through line searching, those in Fig. 8(b) through the estimation (39), and those in Fig. 10(b) through the iteration given in Algorithm 4. The iteration of α_1 and λ_1 that lead to the best step size at the first step [about 0.43, as also shown in Fig. 10(b)] are given in Fig. 12. We can also conclude from these figures that the best step size will approach to 0 by the line searching method, and to 1 by either the estimation method or the recursive method, which coincides the theoretic analysis of these algorithms [15], [23]. The interior point algorithm [21] also utilizes the line searching method when each time calculating a point $y^*(t)$ in the central path using the Newton method.

The parameters used in the simulations are $f_o = (0, 0, -5.88N, 0, 0, 0)^T$ (or the gravity of the object is $600g$), $\alpha_{o1} = (0, 2\pi/3)^T$, $\alpha_{o2} = (0, 0)^T$, $\alpha_{o3} = (0, -2\pi/3)^T$, $C = I$ and $\mu_1 = \mu_2 = \mu_3 = 0.5$. The radius of the spherical object is $R = 110$ mm. $c = [0, 0, 1, 0, 0, 1, 0, 0, 1]^T$ in Algorithm 5 to ensure that $\phi(x) = \Phi(P)$.

B. Experimental Hardware and Software

All these five algorithms have been evaluated with the HKUST 3-fingered hand (see Fig. 2). Each finger of the HKUST hand consists of a Motorman K-3S robot, equipped with a force/torque sensor and a 16×16 tactile array fingertip. A VME-based multiprocessor control system with three 8-axis

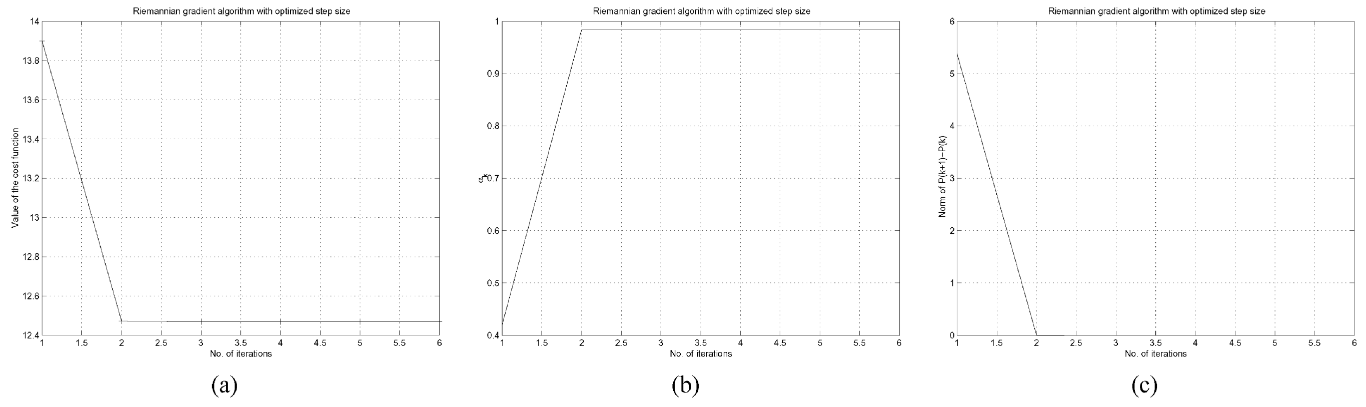


Fig. 10. Algorithm 4. (a) Trajectory of the cost function. (b) Trajectory of the best step size. (c) Trajectory of the error $\mathrm{Tr}((P(k+1)-P(k))^2)$.

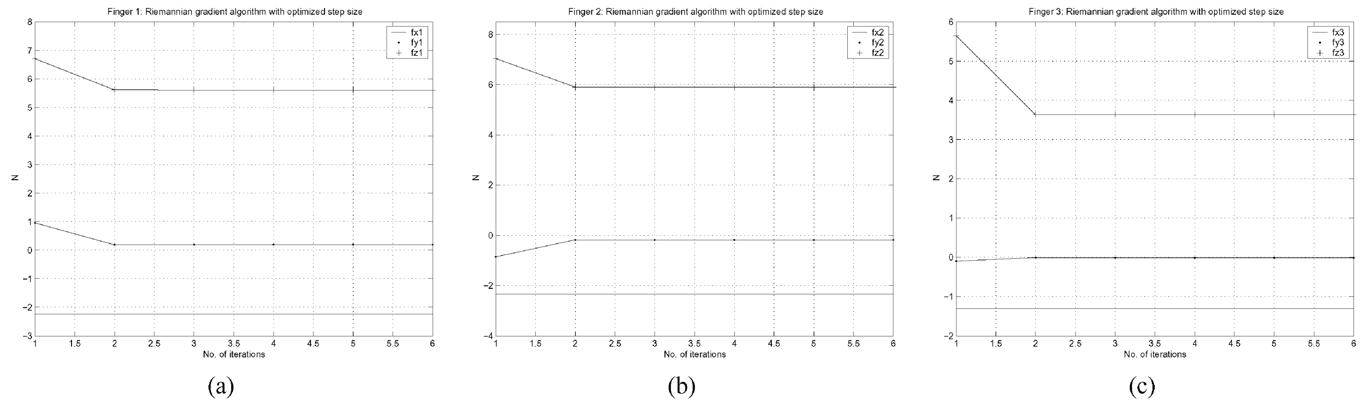


Fig. 11. Algorithm 4. (a) Trajectory of the forces of finger 1. (b) Trajectory of the forces of finger 2. (c) Trajectory of the forces of finger 3.

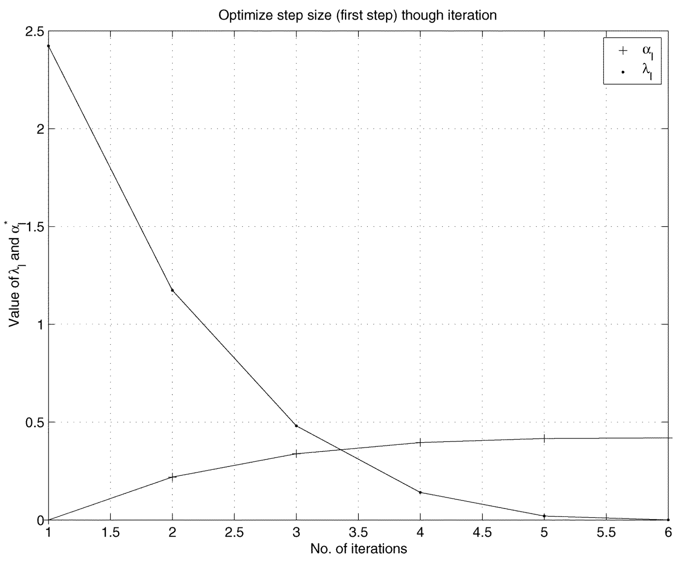


Fig. 12. Algorithm 4. Trajectory of α_1 and λ_1 in the first step.

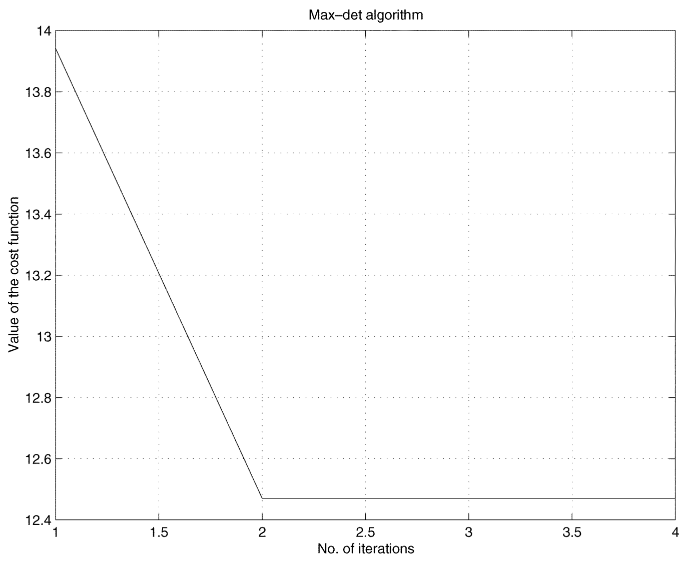
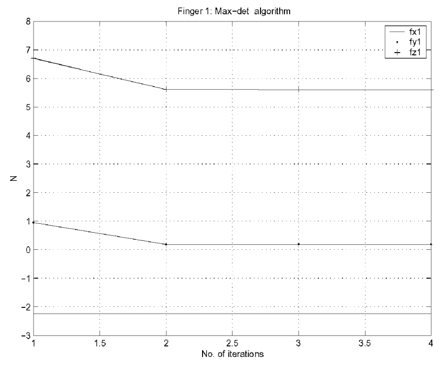


Fig. 13. Algorithm 5. Trajectory of the cost function.

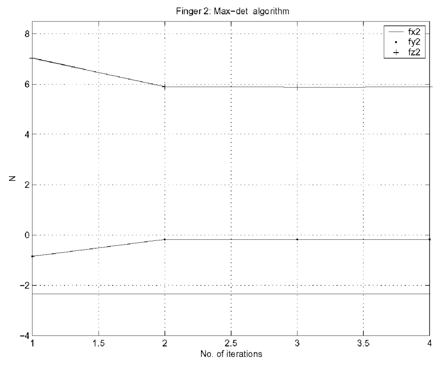
TABLE III
COMPARISON OF THE CONVERGENT RATE OF THE FIVE ALGORITHMS

Algorithm	1	2	3	4	5
No. of iterations	5	2	6	2	2

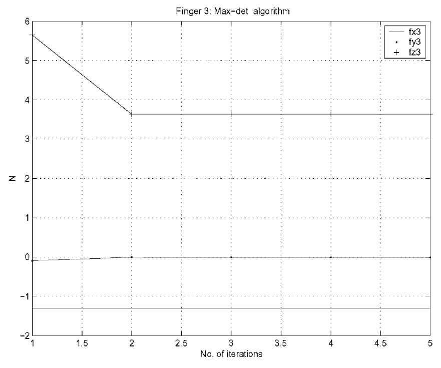
digital signal processor (DSP) motion control boards for joint-level control and two Motorola 68 040 processors for object-level motion and grasping force control is utilized, along with a VxWorks real-time operating system and a Sun workstation. Force/torque data is sampled at 1000 Hz (1 ms). The two CPUs work in parallel, with one running one of the five algorithms (according to the choice by the user) for grasping force generation and the other for planning and generation of contact coordinates and object motion. Synchronization of the tasks run in the two CPUs are realized through shared semaphores.



(a)

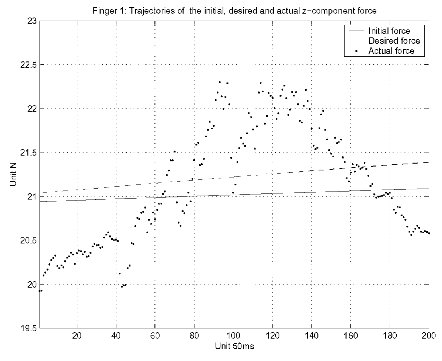


(b)

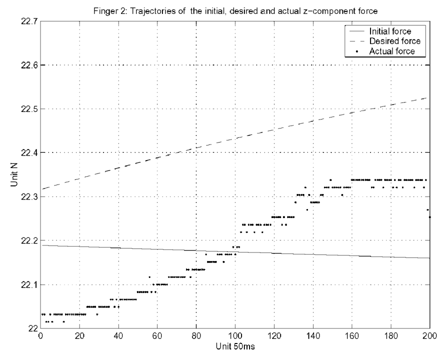


(c)

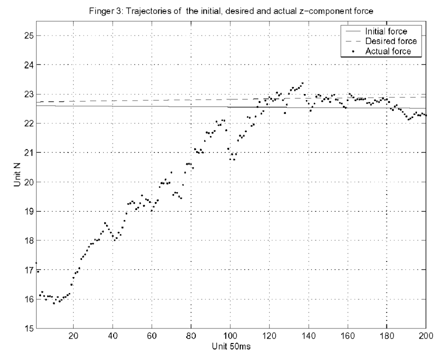
Fig. 14. Algorithm 5. (a) Trajectory of the forces of finger 1. (b) Trajectory of the forces of finger 2. (c) Trajectory of the forces of finger 3.



(a)



(b)



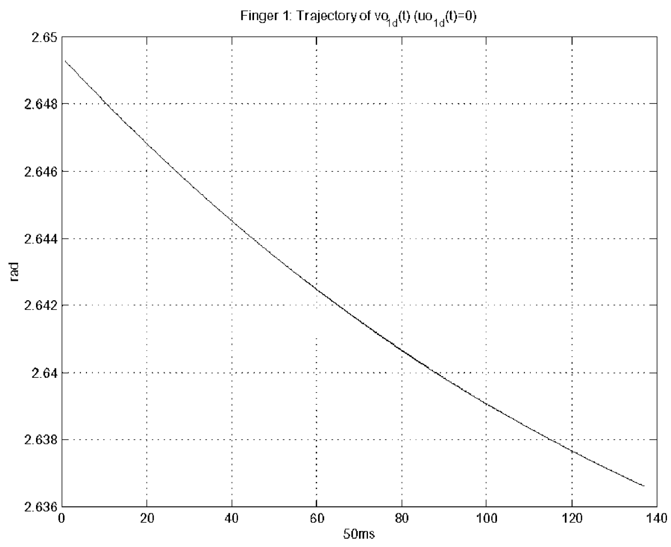
(c)

Fig. 15. (a) Finger 1: trajectory tracking of f_{z1}. (b) Finger 2: trajectory tracking of f_{z2}. (c) Finger 3: trajectory tracking of f_{z3}.

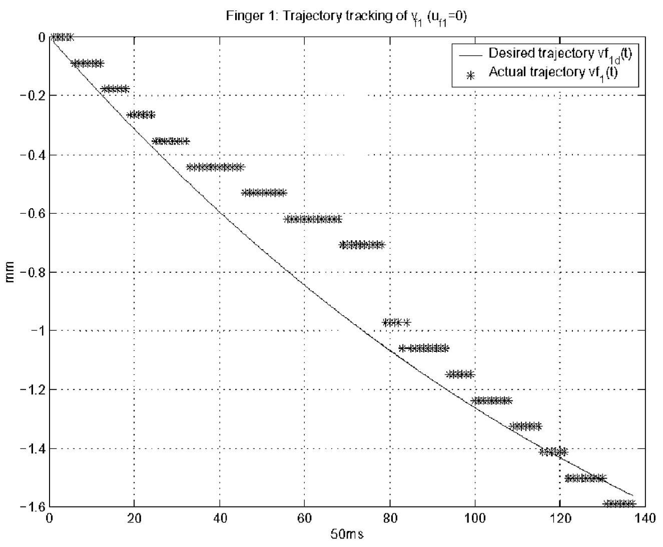
There are two software modules for grasping force optimization. First, the force-opt module, consisting of the five algorithms (Algorithms 1–5), is used to compute an optimal grasping force x^* at a fixed contact point. Second, the force-opt-call module, is used to calculate the grasp map G, a special solution x_0, and the null matrix V of G. In this module, we also compute a set of constant base matrices $\tilde{A}_{P,0}, \ldots, \tilde{A}_{P,kl-6}$ and use either the Han *et al.*'s method or the gradient method to compute a valid initial point (either $x(0)$ or $P(0)$). Then, we call the force-opt module for an optimal grasping force. We make the final executable file by linking the math library *clapack* written in **C** with the object files of the above two modules using *ld68 k*. We choose *ld68 k* to generate executable codes which are compatible with Motorola 68 040 processors.

C. Experimental Results

In the experiments, we verify the performance of the integrated algorithm (53) in manipulating an object along a desired trajectory under rolling contact. In the first experiment, the object is required to move 100 mm along the twist (0,0,1,0,0,0), i.e., manipulate the object along the z axis of the spatial frame, in 10 s. We test the force and position (of the object) tracking performance of the system by (53). Other parameters are $R = 122$ mm, $\alpha_{o1}(0) = (0, (5/6 + 1/100)\pi)^T$, $\alpha_{o2}(0) = (0, (-(1/2) - (1/40))\pi)^T$, $\alpha_{o3}(0) = (0, (1/6 + 1/40)\pi)^T$, and $C_i = 0.001\,[I_{3\times 3} \quad 0]^T$. Fig. 15 shows the z-component contact force response of the three contacts where the desired force trajectories are obtained by Algorithm 1 (or the other algorithms). It should be noted that the curves of the desired contact forces shown in these figures are given by optimal contact forces at the three contact curves. The initial contact forces of the three fingers, from either the Han *et al.*'s method or the gradient method, are also shown in these figures. Note that the *z–component* contact force is changed through the optimization algorithm since we only consider its contribution in the cost function by choosing C and c. The trajectory tracking results of the manipulated object are shown in Fig. 19(a) and (b). It is clearly shown from the experimental results that the tracking error of contact forces is less than 1.0 N, and the object displacement error is less than 1.0 mm. In the second experiment, the object undergoes the same motion but stops in 7 s. We test the performance of the system in tracking the desired trajectory of the grasp configuration (derives from the grasp quality function). Figs. 16(a), 17(a), and 18(a) show the desired curves of the local coordinates of the three contacts on the object, from which we can see that the three fingers are planned to locate at three symmetric points (regarded as the optimal grasp configuration) of the great circle $u_o = 0$. Figs. 16(b), 17(b), and 18(b) give the trajectory tracking results of v_{fi}, $i = 1, \ldots, 3$, respectively. The desired curves $v^d_{fi}(t)$ are obtained from $v^d_{oi}(t)$ by inverting Montana's contact kinematics equations [8], [24], while the actual trajectories $v_{fi}(t)$ are detected through tactile sensors. The contact angles ψ_{id}, $i = 1, \ldots, 3$, are planned to be constant in the experiments.

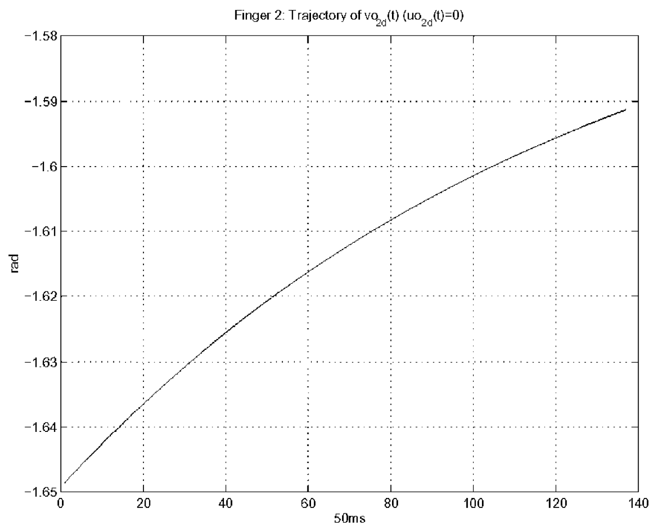


(a)

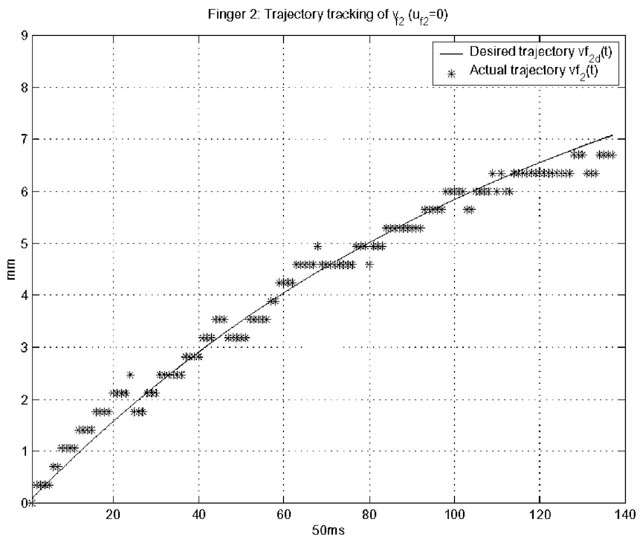


(b)

Fig. 16. (a) Finger 1: desired trajectory of v_{o1}. (b) Finger 1: trajectory tracking of v_{f1}.



(a)



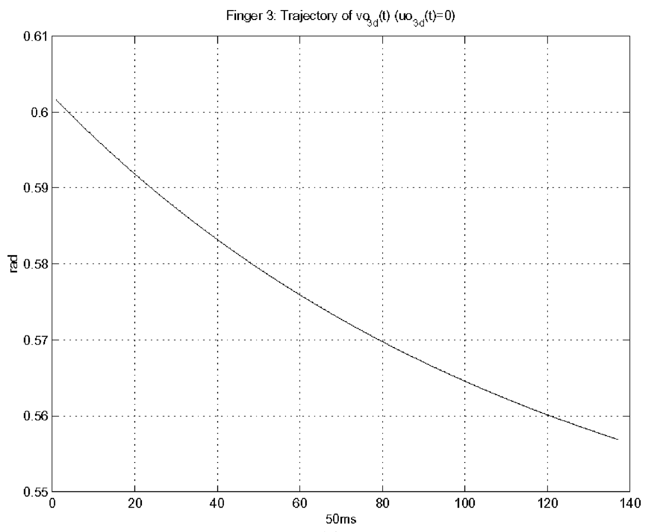
(b)

Fig. 17. (a) Finger 2: desired trajectory of v_{o2}. (b) Finger 2: trajectory tracking of v_{f2}.

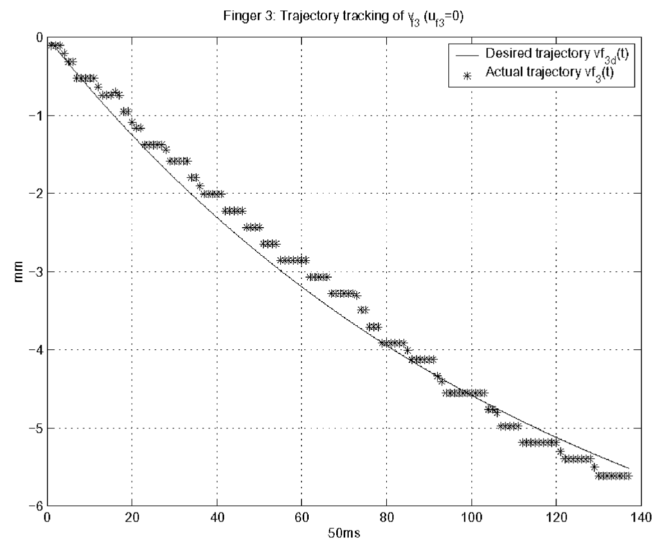
Remark 5: Currently, our algorithm can only be used for manipulation systems with fingers of 6 degree-of-freedoms. In future works, we wish to extend it to systems with fingers of less than 6 degree-of-freedoms by taking into account additional kinematic constraints.

IX. Conclusion

In this paper, starting from Buss *et al.*'s matrix inequalities, Helmke *et al.*'s reduced matrix inequalities, and Han *et al.*'s LMI of friction cone constraints, we formulated grasping force optimization as a convex optimization problem subject to either matrix inequalities or LMIs, and studied five algorithms proposed in these previous works. In particular, we made a detailed analysis on the selection of appropriate step sizes in each algorithm and their effects on convergence. By observing that all the five algorithms need a valid initial point to start the recursion, we proposed two methods for searching such an initial solution. We then proved the quadratic convergence of some of the algorithms by exploring the geometric structures of the affine scaling vector fields associated with them. We compared the real-time performance and the convergence rates of the five algorithms under different processors, operation systems, and representations of friction cone constraints through simulation. Finally, we verified the effectiveness of the manipulation controller through experiments.

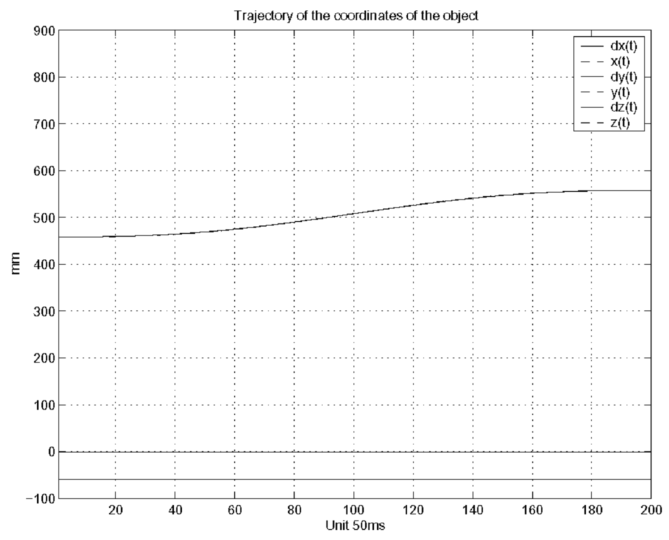


(a)

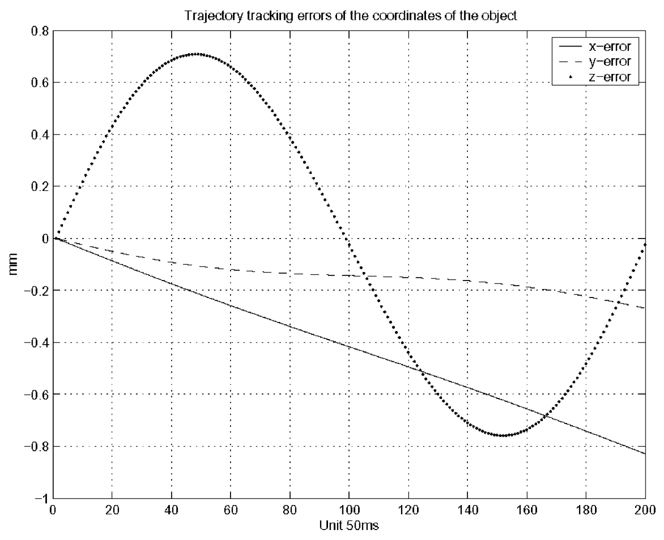


(b)

Fig. 18. (a) Finger 3: desired trajectory of v_{o3}. (b) Finger 3: trajectory tracking of v_{f3}.



(a)



(b)

Fig. 19. (a) Trajectory tracking of the object. (b) Trajectory tracking error of the object.

APPENDIX PROOF OF THEOREM 4

Proof

The iteration can also be written as

$$P_{k+1} = P_k - W(C, B_1, \dots, B_6)(P_k).$$

First, we consider the case $P_k = I$, then

$$W(C, B_1, \dots, B_6)(P_k) = C - I - \sum_{i=1}^{6} a_i B_i$$

for some $a_i \in \mathbb{R}$ and therefore

$$d_2(P_k) = \left\| C - I - \sum_{i=1}^{6} a_i B_i \right\|_F .$$

By Proposition 2, we have

$$d_2(P_{k+1}) = \min \left\{ \left\| \tilde{C} - I - \sum_{i=1}^{6} \nu_i \tilde{B}_i \right\|_F \mid \nu_i \in \mathbb{R} \right\}$$

which is less than $\|\tilde{C} - I - \sum_{i=1}^{6} a_i \tilde{B}_i\|_F$, where $\tilde{C} = P_{k+1}^{1/2} C P_{k+1}^{1/2}$ and $\tilde{B}_i = P_{k+1}^{1/2} B_i P_{k+1}^{1/2}$. Second, note that

$$P_{k+1} - I = -\left(C - I - \sum_{i=1}^{6} a_i B_i \right).$$

Both sides timing $P_{k+1}^{1/2}$ to the left and the right yields

$$\sum_{i=1}^{6} a_i \tilde{B}_i = P_{k+1}^2 - 2P_{k+1} + \tilde{C}.$$

Substituting it into $d_2(P_{k+1})$ yields

$$\begin{aligned} d_2(P_{k+1}) &\leq \left\| \hat{C} - I - \sum_{i=1}^{6} a_i \hat{B}_i \right\|_F \\ &= \left\| -(I - P_{k+1})^2 \right\|_F \leq \| P_{k+1} - I \|_F^2 \\ &= \| -W(C, B_1, \ldots, B_6)(I) \|_F^2 = d_2^2(P_k). \end{aligned}$$

If $P_k \neq I$, we have from (51) that

$$\begin{aligned} P_{k+1} &= P_k - W(C, B_1, \ldots, B_6)(P_k) \\ &= P_k^{1/2}(I - W(\bar{C}, \bar{B}_1, \ldots, \bar{B}_6)(I))P_k^{1/2} \end{aligned}$$

where $\bar{C} = P_k^{1/2} C P_k^{1/2}$ and $\bar{B}_i = P_k^{1/2} B_i P_k^{1/2}$. Let $\hat{P}_{k+1} = I - W(\bar{C}, \bar{B}_1, \ldots, \bar{B}_6)(I)$. Then, by Proposition 1 and (51), we have

$$\begin{aligned} &W(C, B_1, \ldots, B_6)(P_{k+1}) \\ &\qquad = P_k^{1/2} W(\bar{C}, \bar{B}_1, \ldots, \bar{B}_6)(\hat{P}_{k+1}) P_k^{1/2} \end{aligned}$$

and $d_2^2(P_{k+1})$ is given by

$$\begin{aligned} &g(\hat{P}_{k+1}; W(\bar{C}, \bar{B}_1, \ldots, \bar{B}_6)(\hat{P}_{K+1}) \\ &\qquad\qquad W(\bar{C}, \bar{B}_1, \ldots, \bar{B}_6)(\hat{P}_{K+1})) \end{aligned}$$

which is again $d_2^2(\hat{P}_{k+1})$. But now it is evaluated with respect to $\bar{C}$ and $\bar{B}_i$. Applying the results we have derived for $P_k = I$ yields

$$d_2^2(\hat{P}_{k+1}) \leq \| W(\bar{C}, \bar{B}_1, \ldots, \bar{B}_6)(I) \|_F^4. \tag{54}$$

By Proposition 1, the right-hand side of (54) is simply $g^2(P_k; P_k^{1/2} W(\bar{C}, \bar{B}_1, \ldots, \bar{B}_6)(I) P_k^{1/2}$, $P_k^{1/2} W(\bar{C}, \bar{B}_1, \ldots, \bar{B}_6)(I) \; P_k^{1/2})$. Finally, by (51), the right-hand side of (54) is given by

$$g^2(P_k; W(C, B_1, \ldots, B_6)(P_k), W(C, B_1, \ldots, B_6)(P_k))$$

which is exactly $d_2^4(P_k)$. Hence,

$$d_2^2(P_{k+1}) = d_2^2(\hat{P}_{k+1}) \leq d_2^4(P_k).$$

The result follows. □

REFERENCES

[1] F. T. Cheng and D. E. Orin, "Efficient algorithm for optimal force distribution–the compact-dual LP method," *IEEE Trans. Robot. Automat.*, vol. 6, pp. 178–187, Apr. 1990.

[2] E. Paljug, X. Yun, and V. Kummar, "Control of rolling contacts in multiarm manipulation," *IEEE Trans. Robot. Automat.*, vol. 10, pp. 441–452, Aug. 1994.

[3] N. Sarkar, X. P. Yun, and V. Kumar, "Dynamic control of 3-D rolling contacts in two-arm manipulation," *IEEE Trans. Robot. Automat.*, vol. 13, pp. 354–376, June 1997.

[4] J. K. Salisbury and J. Craig, "Articulated hands: Force control and kinematic issues," *Int. J. Robot. Res.*, vol. 1, no. 1, pp. 4–17, 1982.

[5] J. Kerr and B. Roth, "Analysis of multifingered hands," *Int. J. Robot. Res.*, vol. 4, no. 4, pp. 3–17, 1986.

[6] M. Cutkosky, *Robotic Grasping and Fine Manipulation*. Norwell, MA: Kluwer, 1985.

[7] C. Cai and B. Roth, "On the spatial motion of rigid bodies with point contact," in *Proc. IEEE Int. Conf. Robotics Automation*, 1987, pp. 686–695.

[8] D. Montana, "The kinematics of contact and grasp," *Int. J. Robot. Res.*, vol. 7, no. 3, 1988.

[9] ——, "The kinematics of multi-fingered manipulation," *IEEE Trans. Robot. Automat.*, vol. 11, pp. 491–503, 1995.

[10] J. Jameson and L. Leifer, "Automatic grasping: An optimization approach," *IEEE Trans. Syst., Man, Cybern.*, vol. 17, pp. 806–814, Sept. 1987.

[11] Y. Nakamura, K. Nagai, and T. Yoshikawa, "Dynamics and stability in coordination of multiple robotic mechanisms," *Int. J. Robot. Res.*, vol. 8, no. 2, pp. 44–61, 1989.

[12] P. R. Sinha and J. M. Abel, "A contact stress model for multifingered grasps of rough objects," *IEEE Trans. Robot. Automat.*, vol. 8, pp. 7–22, Feb. 1992.

[13] V. Kumar and K. Waldron, "Sub-optimal algorithms for force distribution in multifingered grippers," in *Proc. IEEE Int. Conf. Robotics Automation*, 1987, pp. 252–257.

[14] V. M. Kvrgic, "Computing of the sub-optimal grasping forces for manipulation of a rough object by multifingered robot hand," in *Proc. IEEE Int. Conf. Robotics Automation*, 1996, pp. 1801–1806.

[15] M. Buss, H. Hashimoto, and J. Moore, "Dextrous hand grasping force optimization," *IEEE Trans. Robot. Automat.*, vol. 12, pp. 406–418, June 1996.

[16] M. Buss, L. Faybusovich, and J. Moore, "Recursive algorithms for real-time grasping force optimization," in *Proce. IEEE Int. Conf. Robotics Automation*, 1997, pp. 682–687.

[17] ——, "Dikin-type algorithms for dextrous grasping force optimization," *Int. J. Robot. Res.*, vol. 17, no. 8, 1998.

[18] Z. X. Li, Z. Qin, S. Jiang, and L. Han, "Coordinated motion generation and real-time grasping force control for multifingered manipulation," in *IEEE Int. Conf. Robotics Automation*, 1998.

[19] L. Han, J. Trinkle, and Z. X. Li, "Grasp analysis as linear matrix inequalities," in *Proc. IEEE Int. Conf. Robotics Automation*, 1999.

[20] L. Han, J. C. Trinkle, and Z. X. Li, "Grasp analysis as linear matrix inequality problems," *IEEE Trans. Robot. Automat.*, vol. 16, pp. 663–674, Dec. 2000.

[21] L. Vandenberghe, S. Boyd, and S.-P. Wu, "Determinant maximization with linear matrix inequality constraints," *SIAM J. Matrix Anal. Appl.*, vol. 19, no. 2, pp. 499–533, 1998.

[22] Y. Nesterov and A. Nemirovsky, *Interior-Point Polynominal Methods in Convex Programming*. Philadelphia, PA: SIAM, 1994, vol. 13, Studies in App. Math..

[23] U. Helmke, K. Hueper, and J. B. Moore, "Quadratically convergent algorithms for optimal dextrous hand grasping," *IEEE Trans. Robot. Automat.*, vol. 18, pp. 138–146, Apr. 2002.

[24] R. Murray, Z. X. Li, and S. Sastry, *A Mathematical Introduction to Robotic Manipulation*. Boca Raton, FL: CRC, 1994.

[25] G. F. Liu and Z. X. Li, "Real-time grasping force optimization for multifingered manipulation: Theory and experiments," *IEEE/ASME Trans. Mechatron.*.

[26] N. H. McClamroch and D. W. Wang, "Feedback stabilization and tracking of constrained robots," *IEEE Trans. Automat. Contr.*, vol. 33, pp. 419–426, May 1988.

[27] J. M. Selig, "Curvature in force/position control," in *Proc. IEEE Int. Conf. Robotics Automation*, 1998, pp. 1761–1766.

[28] G. F. Liu and Z. X. Li, "A unified approach to modeling and control of constrained mechanical systems," *IEEE Trans. Robot. Automat.*, vol. 18, pp. 574–587, Aug. 2002.

[29] U. Helmke and J. Moore, *Optimization of Dynamical Systems*. New York: Springer-Verlag, 1993.

[30] L. Faybusovich, "Dikin's algorithm for matrix linear programming problems," in *System Modeling and Optimization, Lecture Notes in Control and Information Sciences*, J. Henry and J. P. Yvon, Eds., 1994, vol. 197, pp. 237–247.

[31] S. Boyd and L. Vandenberghe, *Convex Optimization*, 1st ed., 2001.

[32] S. P. Wu, L. Vandenberghe, and S. Boyd, "Maxdet software for determinant maximization problems. Tech. Rep.," EE. Dept., Inform. Syst. Lab., Stanford. Univ., 1996.

[33] L. Faybusovich, "On a matrix generalization of affine-scaling vector fields," *SIAM J. Matrix. Anal. Appl.*, vol. 16, no. 3, pp. 886–897, 1995.

[34] L. Han, Y. Guan, Z. X. Li, Q. Shi, and J. Trinkle, "Dextrous manipulation with rolling contacts," in *IEEE Int. Conf. Robotics Automation*, 1997, pp. 992–997.

[35] S. L. Jiang, K. K. Choi, and Z. X. Li, "Coordinated motion generation for multifingered manipulation using tactile feedback," in *Proc. IEEE Int. Conf. Robotics Automation*, 1999.

[36] G. F. Liu, J. J. Xu, and Z. X. Li, "Grasp planning as a max-volume problem, Tech. Rep.," Hong Kong University of Science and Technology , 2002.

Guanfeng Liu received the B.E. degree in electrical engineering from Zhejiang University, Hangzhou, China, in 1998, and the Ph.D. degree in electrical and electronic engineering from the Hong Kong University of Science and Technology, Hong Kong.

He is currently a postdoctor in the CS Department, Rensselaer Polytechnic Institute, Troy, NY. His research interests include multifingered manipulation, parallel manipulators, hybrid and nonlinear system theory.

Jijie Xu received the B.E. degree in control science and engineering from Harbin Institute of Technology, Heilongjiang, China, in 2001. He is currently working toward the Ph.D. degree in electrical and electronic engineering at the Hong Kong University of Science and Technology, Hong Kong.

His research interests include multifingered manipulation, grasping, and dextrous manipulation.

Zexiang Li (S'87–M'89) received the B.S. degree in electrical engineering and economics (with honor) from Carnegie Mellon University, Pittsburgh, PA, in 1983, the M.A. degree in mathematics, and the Ph.D. degree in electrical engineering and computer science, both from the University of California, Berkeley, in 1985 and 1989, respectively.

He is currently an Associate Professor at the EEE Department, Hong Kong University of Science and Technology, Hong Kong. His research interests include robotics, nonlinear system theory, and manufacturing.