

Recursive Algorithms for Real-Time Grasping Force Optimization

Martin Buss*

Leonid Faybusovich**

John B. Moore***

* Institute of Automatic Control Engineering, Technical University of Munich
D-80290 München, Germany, E-mail: M.Buss@ieee.org

** Department of Mathematics, University of Notre Dame, Notre Dame, IN 46556-5683, U.S.A.

*** Department of Systems Engineering, Research School of Information Sciences and Engineering
The Australian National University, Canberra ACT 0200, Australia

Abstract

In dextrous robotic hand grasping the external object force needs to be balanced with contact forces maintaining grasp stability. Redundancy in the grasp yields the optimization problem to minimize the grasp effort subject to nonlinear friction stability constraints. We reformulate the optimization problem as a semidefinite program with affine constraints.

In this paper, two versions of strictly convex cost functions including a barrier term, one of them self-concordant, are considered. For the general class of such cost functions Dikin-type algorithms are considered. It is shown that the proposed algorithms guarantee convergence to the unique solution of the underlying semidefinite program. Numerical examples demonstrate the simplicity of implementation and the good numerical properties of the approach.

1 Introduction

Dextrous robotic hand grasping is a broad area of research, see e.g. [1, 2] for some issues of interest. Amongst these issues is a stable manipulation phase, when the object is grasped stably by the robotic hand. The fingers apply contact forces such that the object is held at the desired position and external forces are compensated. An important task is to develop on-line algorithms for the calculation of optimal contact forces.

Early works to solve the grasping force optimization problem are based on a linear programming formulation with linearized friction constraints using a Simplex algorithm [3, 4]. Heuristic schemes and nonlinear programming methods are available [5-7].

Using linear programming the friction constraints are linearized resulting in conservative forces, sometimes larger than would be required. Further, linear programming approaches display undesirable discontinuities in grasping forces when a jump between local minima occurs [1]. Most linear and all nonlinear programming approaches presented so far are to be regarded as off-line with current computing resources.

The approach we take in this and a companion paper [8] solves the problem of grasping force optimization by applying certain interior-point methodology. It is shown that the nonlinear friction force limit constraints on grasping forces are equivalent to the positive definiteness of a matrix subject to linear constraints. Further, compensation of the external object force is also a linear constraint on this matrix. Consequently, the task of grasping force optimization can be formulated as a semidefinite program. This class of problems is discussed in greater detail in [9]. It is known that the specific algorithms we propose for grasping force optimization have real-time capability [10]. In this paper we consider Dikin-type recursive algorithms with improved convergence behavior.

The two forms of cost functions used as the minimization objective throughout the paper consist of two terms: the first term weights grasping cost in terms of force exertion, and the second term is a barrier function tending to infinity at the border of positive definiteness.

For the organization of the paper, Section 2 describes the problem of grasping force optimization to achieve both grasp stability by satisfaction of friction force limits as well as compensation of the external object force. Further, it is shown that grasping force optimization can be solved via semidefinite programming with particular cost functions, and the resulting gradient flows on positive definite matrices achieve the unique optimal solution. In Section 3 a standard recursive version for discrete iterative realization of the gradient flows is developed. An improved Dikin-type algorithm is presented and its convergence properties studied. Numerical examples show good performance

and rapid convergence of the algorithms.

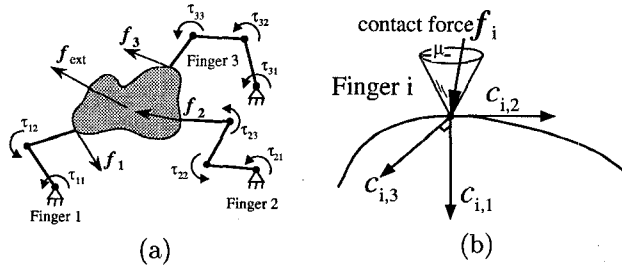


Figure 1: Object grasped by dextrous hand.

2 Grasping Force Optimization via Semidefinite Programming

2.1 Dextrous Grasping Force Optimization

The problem of dextrous grasping force optimization can be posed as a semidefinite program, see also [8] for details. Physical friction constraints on contact forces between hand and grasped object are found to be equivalent to the positive definiteness of a certain matrix P subject to linear constraints. Balancing the external force acting on the object is also viewed as a linear constraint on the grasping forces.

The goal of any dextrous grasping force optimization strategy is to minimize the grasping forces and at the same time to keep the balance of the external forces $f_{ext} \in \mathbb{R}^6$ and maintain friction force limits. The external force balance can be written as

$$-f_{ext} = Wc, \quad (1)$$

where c is a vector of M point contact forces (wrenches). The grip transformation matrix $W \in \mathbb{R}^{6 \times M}$ contains the M contact wrench directions in its columns and maps forces from contact frames to the coordinate frame attached to the object center of mass.

Let us review the particular matrix structure for a point contact model with friction according to Coulomb's law

$$\sqrt{c_{i,2}^2 + c_{i,3}^2} \leq \mu_i c_{i,1}, \quad c_{i,1} > 0, \quad (2)$$

where $c_{i,1}$ is the normal force component at the i th contact point, $c_{i,2}$, $c_{i,3}$ are the tangential components and μ_i denotes the Coulomb friction coefficient at contact point i (see [8] for a detailed discussion and other friction models).

Lemma 1 Consider the case of N fingers grasping an object with the Coulomb friction and positive contact force (towards the object) constraints (2) satisfied, assuming a point contact model. Then

these inequality constraints (2), for $i = 1, 2, \dots, N$ are equivalent to the positive definiteness of $P = \text{Blockdiag}(P_1, P_2, \dots, P_N)$, where $P_i = P_i'$ is linearly constrained to having identical diagonal elements and zeros for the (1, 2) and (2, 1) elements as follows:

$$P_i = \begin{bmatrix} \mu_i c_{i,1} & 0 & c_{i,2} \\ 0 & \mu_i c_{i,1} & c_{i,3} \\ c_{i,2} & c_{i,3} & \mu_i c_{i,1} \end{bmatrix}, \quad \text{for } i = 1, \dots, N. \quad (3)$$

Remark 1 The Coulomb constraints for each finger define the so-called second order cone, see [11]. One can proceed directly with these constraints without imbedding as above. Here our reason for imbedding is to motivate a general theory for Dikin type algorithms for problems involving semidefinite constraints.

The constraints that P_i has identical diagonal elements and zeros for the (1, 2) and (2, 1) elements may be summarized in the general affine constraint

$$A \text{vec}(P) = q, \quad (4)$$

where $A \in \mathbb{R}^{m \times l}$, $\text{rank} A = m < l$, $q \in \mathbb{R}^l$, $P \in \mathbb{R}^{n \times n}$, $P > 0$, $l = n^2$ and $\text{vec}(\cdot)$ denotes the vector operation. Further we need to balance the external object force f_{ext} as the linear constraint (1), which is also easily rewritten in the form of (4).

2.2 Semidefinite Programming

Let $\mathcal{P}(n)$ denote the set of positive definite symmetric $n \times n$ matrices $P = P^T > 0$ and consider a cost index of the form $\phi(P) : \mathcal{P}(n) \mapsto \mathbb{R}$, where ϕ is a strictly convex twice continuously differentiable function on $\mathcal{P}(n)$ with the property

$$\phi(P) \rightarrow +\infty, \quad P \rightarrow \partial\mathcal{P}(n), \quad (5)$$

where $\partial\mathcal{P}(n)$ is the boundary of $\mathcal{P}(n)$.

In particular, we will consider two types of cost functions, each consisting of two terms, one linearly weighting P (or the grasp cost) and the other being a barrier term with the property (5) on any sequence approaching the boundary from inside the manifold $\mathcal{P}(n)$. The first type of cost functions is identical to the one proposed and discussed in [8, 9].

$$\Phi : \mathcal{P}(n) \rightarrow \mathbb{R}, \quad \Phi(P) = \text{Tr}(W_p P + W_i P^{-1}), \quad (6)$$

with W_p, W_i symmetric positive definite matrices, and the second type includes a selfconcordant barrier term [11]

$$\Psi : \mathcal{P}(n) \rightarrow \mathbb{R}, \quad \Psi(P) = \text{Tr}(W_p P) - \log \det P. \quad (7)$$

Further, we endow \mathcal{P} with the Riemannian metric

$$g(P; \xi, \eta) = \text{Tr}(P^{-1}\xi P^{-1}\eta). \quad (8)$$

The explicit gradients of (6) and (7) with respect to the Riemannian metric (8) are:

$$\text{grad}\Phi(P) = PW_p P - W_i = P\nabla\Phi(P)P \quad (9)$$

$$\text{grad}\Psi(P) = PW_p P - P = P\nabla\Psi(P)P, \quad (10)$$

where $\nabla\Psi(P)$, $\nabla\Phi(P)$ are Euclidean gradients. In the absence of affine constraints, the gradient flows for Φ and Ψ are

$$\dot{P} = -\text{grad}\Phi(P); \quad \dot{P} = -\text{grad}\Psi(P),$$

respectively. Under (4), these flows projected into the affine space are

$$\text{vec}(\dot{P}) = -Q\text{vec}(\text{grad}\Phi(P)) \quad (11)$$

$$\text{vec}(\dot{P}) = -Q\text{vec}(\text{grad}\Psi(P)), \quad (12)$$

with Q the projection operator

$$Q = I - (P \otimes P)A^T[A(P \otimes P)A^T]^{-1}A \quad (13)$$

where \otimes denotes the Kronecker product. Clearly, $AQ = 0$ and $A\text{vec}(\dot{P}) = 0$, so that these flows, if initialized satisfying the constraint (4), will satisfy the constraint for all $t > 0$.

2.3 Gradient Flows on Positive Definite Matrices

Let us review without proof the results by Buss *et al.* of linearly constrained gradient flows to solve for the equilibrium of $\Phi(P)$ [8, 9] and $\Psi(P)$ [12].

Theorem 1 (*Linearly constrained gradient flow*)
The unique equilibrium of $\Phi(P)$ can be obtained by the following linearly constrained exponentially convergent gradient flow

$$\text{vec}(\dot{P}) = \bar{Q} \text{vec}(P^{-1}W_i P^{-1} - W_p), \quad (14)$$

where the projection operator is $\bar{Q} := (I - A^T(AA^T)^{-1}A)$ and \mathcal{P} is endowed with the standard Riemannian metric.

Theorem 2 (*Inverse barrier flow*) Endowing $\mathcal{P}(n)$ with the Riemannian metric (8), assuming the linear constraint (4) and that the initial condition $P(0) = P_0 \in \mathcal{P}(n)$ satisfies this constraint, yields a constrained Riccati gradient flow in terms of the projection operator Q of (13) as

$$\text{vec}(\dot{P}) = Q \text{vec}(W_i - PW_p P), \quad (15)$$

which exponentially converges to the unique equilibrium $P^* = \arg \min_P \Phi(P)$.

Theorem 3 (*Selfconcordant barrier flow*) Endowing $\mathcal{P}(n)$ with the Riemannian metric (8), the unique equilibrium $P^* = \arg \min_P \Psi(P)$ can be obtained by the exponentially convergent gradient flow

$$\text{vec}(\dot{P}) = Q \text{vec}(P - PW_p P), \quad (16)$$

where the initial condition $P(0) = P_0 \in \mathcal{P}(n)$ satisfies the linear constraints (4) and Q is the projection operator of (13).

2.4 Numerical Examples

As a grasp example let us consider a 4-fingered grasp (point contact with friction) of a rectangular object. Assume the external force f_{ext} of (1) be

$$f_{ext} = [1 \ 1 \ -1 \ 0 \ 0.5 \ 0.5]^T,$$

and W be

$$W^T = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & -a_1 \\ 1 & 0 & 0 & 0 & 0 & b \\ 0 & 0 & 1 & -b & a_1 & 0 \\ 0 & 1 & 0 & 0 & 0 & a_2 \\ 1 & 0 & 0 & 0 & 0 & b \\ 0 & 0 & 1 & -b & -a_2 & 0 \\ 0 & -1 & 0 & 0 & 0 & -a_3 \\ 1 & 0 & 0 & 0 & 0 & -b \\ 0 & 0 & 1 & b & -a_3 & 0 \\ 0 & -1 & 0 & 0 & 0 & a_4 \\ 1 & 0 & 0 & 0 & 0 & -b \\ 0 & 0 & 1 & b & a_4 & 0 \end{bmatrix},$$

where $a_1 = 0.1$, $a_2 = 0.15$, $a_3 = 0.05$, $a_4 = 0.065$, $b = 0.02$. A valid initial condition c_0 is

$$c_0 = \begin{bmatrix} 6.5780 & 0.8793 & 1.3718 & 5.4447 \\ 0.8793 & -1.8718 & 3.7020 & -0.3793 \\ -0.9960 & 7.3207 & -0.3793 & 0.4960 \end{bmatrix}^T,$$

The weighting matrices selection is $W_p = I$, $W_i = 0.512I$ and the friction coefficient is assumed to be $\mu = 0.4$ at all contact points. The particular value $W_i = 0.512I$ is chosen to place the equilibrium of both cost functions Φ , Ψ at roughly the same solution.

The three gradient flow versions (14), (15) and (16) are numerically solved using Runge-Kutta integration. Numerical results for these continuous flows are not shown in this paper, see e.g. [8]. Convergence of these gradient flows is exponential and the unique equilibrium of Φ is obtained as

$$c_\Phi^* = \begin{bmatrix} 3.8719 & 0.1222 & 1.2195 & 5.6597 \\ 0.4146 & -1.7195 & 4.5026 & 0.1677 \\ -1.3271 & 4.0291 & 0.2955 & 0.8271 \end{bmatrix}^T.$$

Likewise the minimum of Ψ is

$$c_{\Psi}^* = \begin{bmatrix} 4.0042 & 0.1595 & 1.2688 & 5.6803 \\ 0.3300 & -1.7688 & 4.4843 & 0.2200 \\ -1.2200 & 4.2002 & 0.2905 & 0.7200 \end{bmatrix}^T.$$

The convergence criterion and tolerance during numerical integration is the same for all cases and the solution is obtained in 46, 27 and 40 integration steps, respectively. In this particular example the choice of the Riemannian metric (8) yields better numerical properties. The computation time using MATLAB on a SUN-SPARCstation 20 is 9s, 38s, 60s, respectively.

3 Real-Time Application

Implementation of the proposed gradient flows causes significant complexity. At each iteration the actual projection \bar{Q} or Q has to be computed.

For the flows of Theorem 2 and 3 this involves an inverse calculation of a (rather sparse) $m \times m$ -matrix, where $m = 9N^2 - 3N + 6$ for a grasp situation with N fingers (e.g. $N = 3 \rightarrow m = 78$).

For the flow of Theorem 1 the projection can be computed off-line for grasping situations where W is constant implying A and \bar{Q} constant. Constant W means that the contact points of the fingers with the object are constant, i.e. the grasp situation is stable.

3.1 Standard Riemannian Metric Recursive Algorithms

Using the standard Riemannian metric the gradient flow of Theorem 1 is implemented as

$$\text{vec}(P_{k+1}) = \text{vec}(P_k) + \alpha_k \bar{Q} \text{vec}(P_k^{-1} W_i P_k^{-1} - W_p), \quad (17)$$

with a suitable step-size α_k and P_0 satisfying all constraints. In the experimental implementation an approximative line-search algorithm was used to find a suitable step-size α_k^* [13]. The computation time for the optimization algorithm during the experiments was 5ms on a MC 68040 (25Mhz).

3.2 Dikin-Type Recursive Algorithms

We encounter difficulties when selecting an appropriate step-size α_k in the recursive algorithm of the previous section. Here we discuss Dikin-type algorithms for solving semidefinite programs and in particular the grasping force optimization problem. The Dikin algorithm for semidefinite programming problems with linear cost functions is discussed in [14]. In this paper we briefly state the results without proof and show some numerical results of Dikin-type recursive algorithms.

Lemma 2 (Dikin ellipsoid) Suppose $P > 0$ is a positive definite matrix and that $g(P; X, X) = \text{Tr}(P^{-1} X P^{-1} X) < 1$ is satisfied for the symmetric matrix X . Then the following inequality holds

$$P + X > 0. \quad (18)$$

where $A > 0$ means that $\langle \xi, A\xi \rangle > 0$ for any $\xi \in \mathbb{R}^n$, $\xi \neq 0$.

Consider the following problem

$$\phi(P) \rightarrow \min, \quad (19)$$

subject to (4) and $P \in \mathcal{P}(n)$. Assume that $\phi(P)$ is a strictly convex twice continuously differentiable function on $\mathcal{P}(n)$ such that (5) holds. We will also assume that the constraint set is bounded, or otherwise that the cost function has compact sublevel sets. The cost function Φ of (6) has compact sublevel sets, see [9]; likewise, Ψ of (7). We will use the notation \mathcal{F} for the set defined by (4), $P \in \mathcal{P}(n)$. We assume that $\mathcal{F} \neq \emptyset$.

Suppose that $f : \mathcal{F} \rightarrow \mathcal{F}$ is a continuous map with the property

$$\phi(f(P)) < \phi(P) \quad \text{unless} \quad P = P^*, \quad (20)$$

where P^* is the unique solution to (19). Let $f^i(P_0)$ denote the i times application of f .

Theorem 4 For any map f satisfying (20) and any $P_0 \in \mathcal{F}$, the sequence $f^i(P_0)$ converges to the optimal solution P^* of the problem (19).

Example 1: Given $P_0 \in \mathcal{F}$, denote the set

$$D_\rho(P_0) = \left\{ \begin{array}{l} X + P_0 : g(P; X, X) \leq \alpha, X^T = X, \\ \text{Avec}(X) = 0 \end{array} \right\} \quad (21)$$

for some $0 < \rho \leq 1$. Let $f_\rho(P_0)$ be a unique solution of the problem:

$$\phi(P) \rightarrow \min, \quad P \in D_\rho(P_0). \quad (22)$$

Since by Lemma 2 $D_\rho(P_0) \subset \mathcal{F}$ and ϕ is strictly convex, we conclude that $f_\rho : \mathcal{F} \rightarrow \mathcal{F}$ is a continuous map. It is also obvious that (20) is satisfied. Hence, we formulate the following version of the Dikin algorithm for standard convex programming problems with linear constraints, see e.g. [15].

Theorem 5 For the map f_ρ , and any $P_0 \in \mathcal{F}$, the sequence $f_\rho^i(P_0) \rightarrow P^*$ when $i \rightarrow +\infty$.

Denote by $V(P)$ the gradient of the function ϕ relative to the metric (8) constrained by (4), see e.g. Theorems 2 and 3. Consider the map $f : \mathcal{F} \rightarrow \mathcal{F}$

$$f(P) = P - \alpha(P) \frac{V(P)}{\|V(P)\|_P}, \quad (23)$$

where $\|V(P)\|_P = \text{Tr}(P^{-1}V(P)P^{-1}V(P))$ and $\alpha(P)$ is a unique solution of

$$\phi\left(P - \alpha \frac{V(P)}{\|V(P)\|_P}\right) \rightarrow \min, \quad 0 \leq \alpha \leq 1. \quad (24)$$

It is easy to see that $\alpha(P) = 0$ if and only if $P = P^*$. Also, $\alpha(P)$ is a continuous function of P , as is $V(P)$ and f . Thus f satisfies (20) by virtue of Lemma 2. Hence,

Theorem 6 For the map f above, and any $P_0 \in \mathcal{F}$ the sequence $f^i(P_0) \rightarrow P^*$, $i \rightarrow +\infty$.

Remark 2 The explicit Dikin type recursions associated with the minimization of Φ and Ψ are summarized for convenience as

$$\text{vec}(P_{k+1}) = \text{vec}(P_k) - \alpha_k Q \frac{\text{vec}(P_k W_p P_k - W_i)}{\|P_k W_p P_k - W_i\|_{P_k}}, \quad (25)$$

$$\text{vec}(P_{k+1}) = \text{vec}(P_k) - \alpha_k Q \frac{\text{vec}(P_k W_p P_k - P_k)}{\|P_k W_p P_k - P_k\|_{P_k}}. \quad (26)$$

Here $0 \leq \alpha_k \leq 1$ is selected from a bounded line search to minimize $\Phi(P_{k+1})$ or $\Psi(P_{k+1})$. The line search is relatively inexpensive compared to the cost of calculating a new direction.

3.3 Numerical Examples

Figure 2 shows the numerical behavior of the standard recursive algorithm (17) with line search [8, 13]. Figures 3 and 4 show the result obtained by the Dikin-type algorithms (25) and (26). In the Figures (a) shows the normal contact wrenches $c_{i,1}$, (b) the cost index Φ , Ψ and (c) the step size α_k applied at each iteration, respectively.

The algorithm checks at each iteration k if a Dikin step of $\alpha_k = 1$ results in a decrease of the cost function Φ or Ψ , and if so performs an update with $\alpha_k = 1$. If no decrease in cost appears a line-search is initiated to find an α_k^* , $0 < \alpha_k < 1$ such that the decrease in cost is maximal. Convergence takes place in 3-7 steps.

4 Conclusions

We have reformulated the well known grasping force optimization problem as a semidefinite program. Exponentially convergent gradient flows or their recursive versions obtain the unique solution to the problem. Two recursive schemes were proposed, one based on a standard Euclidean gradient method and the other with Dikin-type recursions. Numerical examples have shown the simplicity and efficiency of our approach to grasping force optimization.

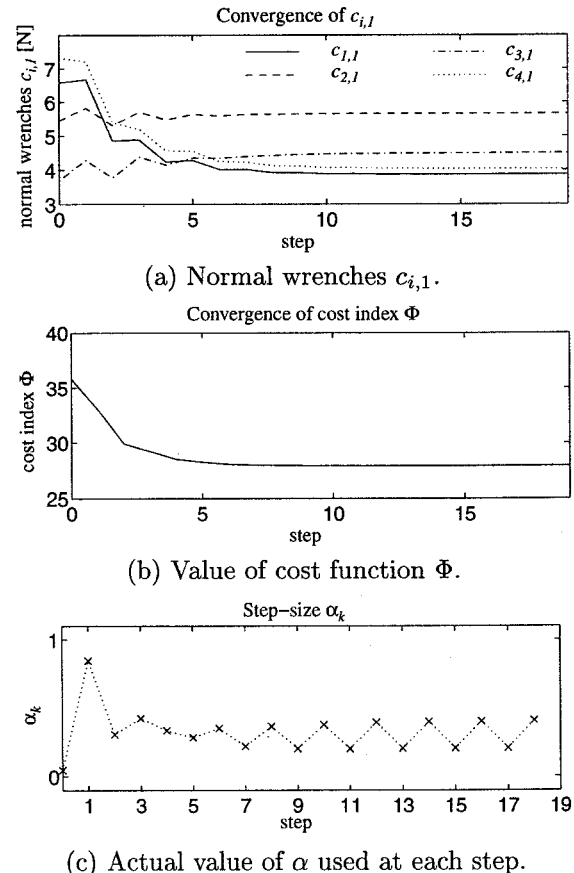
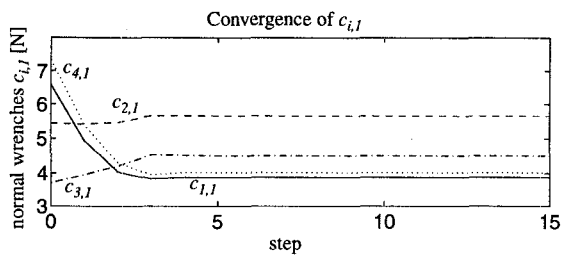


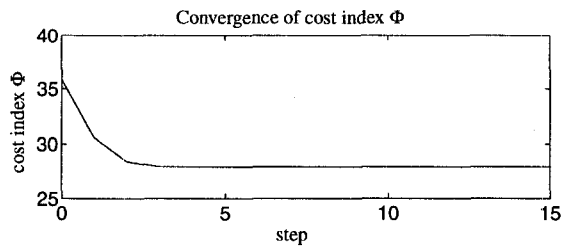
Figure 2: Standard recursive algorithm (17)

References

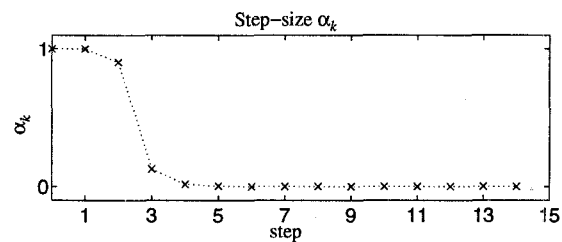
- [1] K. Shimoga, "Robot grasp synthesis algorithms: A survey," *International Journal of Robotics Research*, vol. 15, pp. 230–266, June 1996.
- [2] E. S. Venkaraman and T. Iberall, *Dextrous Robot Hands*. New York: Springer, 1990.
- [3] F.-T. Cheng and D. Orin, "Efficient algorithm for optimal force distribution—the compact-dual lp method," *IEEE Transactions on Robotics and Automation*, vol. 6, pp. 178–187, April 1990.
- [4] J. Kerr and B. Roth, "Analysis of multifingered hands," *International Journal of Robotics Research*, vol. 4, pp. 3–17, Winter 1986.
- [5] Y. Nakamura, K. Nagai, and T. Yoshikawa, "Dynamics and stability in coordination of multiple robotic mechanisms," *International Journal of Robotics Research*, vol. 8, pp. 44–61, April 1989.
- [6] P. Sinha and J. Abel, "A contact stress model for multifingered grasps of rough objects," *IEEE*



(a) Normal wrenches $c_{i,1}$.



(b) Value of cost function Φ .

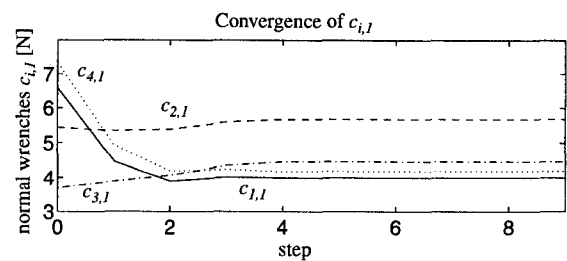


(c) Actual value of α used at each step.

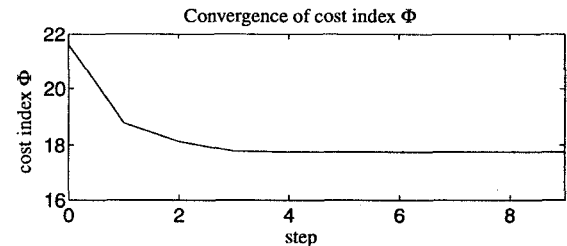
Figure 3: Dikin-type algorithm (25).

Transactions on Robotics and Automation, vol. 8, pp. 7–22, February 1992.

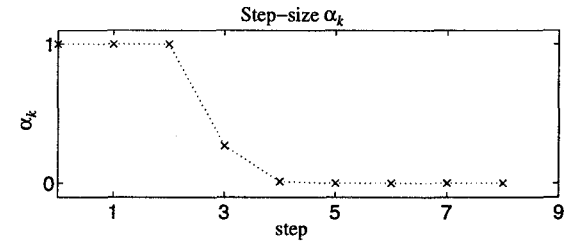
- [7] T. Yoshikawa and K. Nagai, “Evaluation and determination of grasping forces for multi-fingered hands,” in *Proceedings of the IEEE International Conference on Robotics and Automation*, (Philadelphia, Pennsylvania), pp. 245–251, 1988.
- [8] M. Buss, H. Hashimoto, and J. Moore, “Dextrous hand grasping force optimization,” *IEEE Transactions on Robotics and Automation*, vol. 12, no. 3, pp. 406–418, 1996.
- [9] U. Helmke and J. Moore, *Optimization and Dynamical Systems*. Springer, 1993.
- [10] M. Buss and K. Kleinmann, “Multi-fingered grasping experiments using real-time grasping force optimization,” in *Proceedings of the IEEE International Conference on Robotics and Automation*, (Minneapolis, Minnesota), pp. 1807–1812, 1996.



(a) Normal wrenches $c_{i,1}$.



(b) Value of cost function Ψ .



(c) Actual value of α used at each step.

Figure 4: Dikin-type algorithm (26).

- [11] Y. Nesterov and A. Nemirovsky, “Interior-point polynomial methods in convex programming,” *Studies in Applied Mathematics, SIAM*, vol. 13, 1994.
- [12] L. Faybusovich, “On a matrix generalization of affine-scaling vector fields,” *SIAM Journal of Matrix Analysis*, no. 3, 1995.
- [13] M. Buss and T. Schlegl, “Multi-fingered regrasping using on-line grasping force optimization,” in *Proceedings of the IEEE International Conference on Robotics and Automation*, (Albuquerque, New Mexico), 1997.
- [14] L. Faybusovich, “Dikin’s algorithm for matrix linear programming problems,” in *Lecture Notes in Control and Information Sciences 197: System modelling and Optimization* (J. Henry and J.-P. Yvon, eds.), pp. 237–247, Springer, 1994.
- [15] Y. Ye and E. Tse, “An extension of karmarkar’s projective algorithm for convex programming,” *Mathematical Programming*, vol. 44, pp. 157–179, 1989.