A Grasping Force Optimization Algorithm for Multi-arm Robots with Multi-fingered Hands

Vincenzo Lippiello, *Member, IEEE,* Bruno Siciliano, *Fellow, IEEE,* and Luigi Villani *Senior Member, IEEE*

***Abstract*—The computation of the grasping forces for a multi-arm robotic manipulation system (e.g. an anthropomorphic bimanual system) is considered in this paper. This problem is formulated as a convex optimization problem, considering also joint torque constraints. An algorithmic solution suitable for on-line implementation is presented, which allows a substantial reduction of the computational load by adopting a compact formulation and dynamically decreasing the number of active torque constrains. Moreover, for the case of a bimanual manipulation system, a sub-optimal single-hand optimization algorithm is proposed and compared with that providing the optimal solution. Finally, a new algorithm for a valid initial-point evaluation is proposed. The effectiveness of the described methods has been tested in a simulation case study where the grasping forces of a humanoid torso equipped with two five-finger robotic hands are modified on-line to handle a load with a time-varying mass.**

***Index Terms*—Grasping Force Optimization, Grasping, Manipulation, Multi-fingered Hands.**

I. Introduction

THE control of a multi-arm robotic manipulation system involves several aspects ranging from the synthesis of the optimal grasping contact points to load sharing and grasp control. With respect to this last issue, the evaluation of the grasping forces able to guarantee stability of the grasp and its feasibility, also in the presence of external disturbances, is a challenging task. The complexity of the problem relies on the need for on-line resolution of an optimization problem where both constraints and objective functions are non-linear, the number of variables and constraints is relatively large, and the grasp configuration and load wrench may change with time (e.g. when pouring water from a bottle into a glass as considered in the case study presented in this paper).

The force closure problem [1] and, more generally, the problem of computing the optimal grasp configurations, are not considered here, because it is assumed that the contact points, which guarantee the force closure property, are assigned by the grasp planner. On the other hand, the grasping force optimization (GFO) problem has not yet been intensively investigated for the case of bimanual human-like robotic systems, for which the computational complexity becomes a major issue.

The research leading to these results has been supported by the DEXMART and ARCAS Large-scale integrating projects, which has received funding from the European Community's Seventh Framework Programme (FP7/2007-2013) under grant agreement ICT-216239 and ICT-287617, respectively. The authors are solely responsible for its content. It does not represent the opinion of the European Community and the Community is not responsible for any use that might be made of the information contained therein.

The authors are with PRISMA Lab, Dipartimento di Informatica e Sistemistica, Università degli Studi di Napoli Federico II, via Claudio 21, 80125, Naples, Italy. Email: {lippiello, siciliano, lvillani}@unina.it.

The non-linearity of the contact friction models (point contact with friction or soft-finger contact) significantly complicates the solution of the GFO problem. For this reason, the analysis and synthesis of frictional force-closure grasps has been initially studied by linearizing the friction cone constraints and then applying linear programming techniques [2]–[4]. The corresponding problems, however, are ill-conditioned. More recently, a new formulation was presented in [5], which is fast when the grasp configuration does not change thanks to the off-line computation of the feasible region (i.e. a polytope depending on the grasp configuration). Non-linear programming techniques have been investigated in [6], but they are not suitable for real-time applications.

In [7] the friction cone constraints have been formulated in terms of linear matrix inequalities (LMIs), and the grasping optimization problem is addressed as a convex optimization problem involving LMIs with the max–det function as objective function. This problem can be efficiently solved with the interior point algorithm for a small number of fingers. Moreover, joint torque limits can be considered in the same framework as LMIs.

Starting from the observation that verifying the friction cone constraints is equivalent to testing the positive definiteness of certain symmetric matrices, in [8] the GFO problem has been formulated as a semi-definite programming problem, which is a convex optimization problem on a Riemannian manifold with linear constraints. Several gradient flow type algorithms have been proposed to provide solutions suitable for real-time applications [9], [10]; to reduce the computational complexity, the computation of the solution can be split into one on-line and one off-line phase and sparse matrix techniques can be adopted [11]. This technique has been employed and experientially tested with an impedance control approach addressing the regrasping problem for dextrous manipulation tasks [12].

A further improvement has been presented in [13], consisting in a new compact semidefinite representation of the friction cone constraints which allows a significant reduction of the dimension of the optimization problem. Moreover, an estimation technique and a recursion method for selecting the step size in the gradient algorithm are proposed, together with the proof of the quadratic convergence of the algorithm.

A fast interior point algorithm for solving the force optimization problem is presented in [14]. The complexity of the proposed formulation is linear in the number of contact points while it was quadratic or cubic in the previous approaches. However, the torque constraints are not explicitly taken into account. A computationally efficient approach to contact force feasibility and contact force distribution has been recently

presented in [15]. By adopting the GJK distance algorithm, the solution of the formulated linear system can be achieved without calculating the Minkowski sum, which usually is computationally expensive, resulting in a real-time feasible algorithm.

An interesting comparison of the methods developed in [7], [8], [13] is proposed in [16], [17], where a common framework has been developed to test the previous competing algorithms for grasping force optimization, and the issue of the step size selection for each method is addressed. Moreover, a new solution for the initial point problem, i.e. an initial solution that satisfies the friction cone constraints and the force equilibrium equation, has been provided.

A number of alternative approaches to GFO can be found in the literature. For example, the use of Lagrangian networks is investigated in [18], [19]. These neural networks are capable of taking into account the non-linearity of the friction constraints and of the joint torque limits, and asymptotically converge to a set of optimal grasping forces. In [20], [21] a method based on the minimization of a cost function is presented, which gives an analytical solution but does not ensure by itself the satisfaction of the friction constraints. An iterative correction algorithm allows modifying this function until the internal forces enter the friction cone, resulting in a fast sub-optimal solution suitable for real-time applications. The GFO problem in the case of whole hand grasp (also considering finger's inner links and the palm), is addressed in [22]. In this case, the problem is formulated as a convex optimization problem involving LMIs similarly to [7], but considering a decomposition of the contact force space into four orthogonal subspaces of active and passive forces. A technique suitable for the case of unknown external disturbances, investigated in [23], is based on tightening pre-strain forces rather than active, direction-related blocking forces. A recent approach, presented in [24], allows to reduce the computational complexity of the algorithms based on semi-definite programming by adopting a weighted barrier function formulation and the Newton method. A similar algorithm, originally proposed in [25], [26], is used in [27] to characterize the role of different postural synergies of the human hand in obtaining force-closure grasps. Finally, an n-dimensional ray-shooting algorithm has been presented in [28], which runs in 6D wrench space. This algorithm provides a fast method to determine the minimum grasping forces, suitable to perform GFO in real time.

The method proposed in [8] has the main disadvantage that it requires the on-line pseudo-inversion of a structurally constrained matrix whose dimension linearly increases with the number of fingers by a factor that depends on the contact type. By adopting the frictional cone constraint matrix representation proposed in [13], the dimension of the problem decreases considerably so that the solution can be computed in real time. However, if torque limits constraints are also considered, the complexity of the problem increases more than quadratically with the number of joints. In addition, at the beginning of each optimization cycle, it is required the evaluation of an initial point that satisfies the frictional cone constraints and the joint torque limits. The initial point can be computed with the methods proposed in [16], but at the expense of a significant computational effort.

The work presented here, based on the friction constraints formulation in [13] and on the solution of a convex optimization problem as in [10], includes and extends the results presented in our previous conference papers [29]–[31]. Joint torque constraints are taken into account in a very compact formulation (the proposed linear constraint matrix has a dimension that increases quadratically with the number of fingers, while the rate is cubic in [10]), with a minimum increase of computation complexity, compatible with real-time constraints. A new iterative formulation is proposed, which fully exploits the results of the previous optimization cycle, thus avoiding the evaluation of a new initial point at the beginning of each optimization cycle. The computation of the initial point, required only for the first step, can be made by using the new monotone increasing gradient flow algorithm presented here, which is fully compatible with the proposed framework. Finally, a sub-optimal single-hand optimization algorithm is proposed within the same framework to further simplify the problem. This algorithm is based on a new criterion for load sharing [32]–[34], which allows to improve the sub-optimal solution. The effectiveness of the proposed approach has been tested in a simulation scenario where a robotic torso equipped with two dextrous hands is used to empty a half-filled bottle.

II. Problem Formulation

Consider a bimanual robotic system equipped with two multi-fingered hands grasping an object with n contacts between the object and the fingertips, the links of the fingers and the palm. Denote the contact wrench of the grasp by $\boldsymbol{c} = \begin{bmatrix} \boldsymbol{c}_r^{\mathrm{T}} & \boldsymbol{c}_l^{\mathrm{T}} \end{bmatrix}^{\mathrm{T}} = \begin{bmatrix} \boldsymbol{c}_1^{\mathrm{T}} & \dots & \boldsymbol{c}_n^{\mathrm{T}} \end{bmatrix}^{\mathrm{T}} \in \mathbb{R}^{nm}$, where $\boldsymbol{c}_i \in \mathbb{R}^m$ is the wrench vector of the i-th contact with dimension m depending on the adopted contact model, and $\boldsymbol{c}_r$ and $\boldsymbol{c}_l$ are the corresponding wrench vectors of all the contact points of the right and left hand, respectively.

The grasping force optimization problem consists in finding the set of contact wrenches balancing the generalized external force $\boldsymbol{h}_e \in \mathbb{R}^6$ acting on the object (including object inertia and weight), which are feasible with respect to the kinematic structure of the hands and to the joint torque limits, and minimize the overall stress applied to the object, i.e, the internal forces. Moreover, to avoid the slippage of the fingers on the object surface, each contact wrench has to be confined within the friction cone.

The balance equation for the generalized forces applied to the object can be written in the form

$$\boldsymbol{h}_e = \boldsymbol{G}\boldsymbol{c}, \tag{1}$$

where $\boldsymbol{G} = \begin{bmatrix} \boldsymbol{G}_r & \boldsymbol{G}_l \end{bmatrix} \in \mathbb{R}^{6\times nm}$ is the grasp map composed of the grasp matrices of the right and left hand, which is full-rank for force-closure grasps [1]. It is assumed that the contact point configurations ensuring the force-closure constraints are assigned at each time by the planning system.

Although several contact models can be used, the two usually adopted models are the *point contact with friction* (PCWF) model and the *soft finger contact* (SFC) model.

In the PCWF case, the contact wrench has three degrees of freedom (DOFs) ($m = 3$): the normal component $c_{i,z}$ to the object surface and the two components $c_{i,x}$, $c_{i,y}$ on the tangent plane. The friction constraint is described as

$$\frac{1}{\mu_i^2}\left(c_{i,x}^2 + c_{i,y}^2\right) \leq c_{i,z}^2 \text{ and } c_{i,z} > 0, \tag{2}$$

where μ_i denotes the friction coefficient at the i-th contact point.

In the SFC case, the contact wrench has an additional DOF $c_{i,t}$ ($m = 4$), corresponding to the torsional component of the moment about the contact normal. In this case, the elliptic approximation of the friction constraint can be expressed as

$$\frac{1}{\mu_i}\left(c_{i,x}^2 + c_{i,y}^2\right) + \frac{1}{\mu_{t,i}} c_{i,t}^2 \leq c_{i,z}^2 \text{ and } c_{i,z} > 0, \tag{3}$$

where μ_i and $\mu_{i,t}$ denote the tangential and the torsion friction coefficients at the i-th contact point, respectively.

The balance equation for the torques applied to the fingers joints of the hands can be written in the form

$$\boldsymbol{J}^{\mathrm{T}}(\boldsymbol{q})\boldsymbol{c} + \boldsymbol{\tau}_e = \boldsymbol{\tau}, \tag{4}$$

where $\boldsymbol{J}(\boldsymbol{q}) = \left[\begin{array}{cc} \boldsymbol{J}_r^{\mathrm{T}} & \boldsymbol{J}_l^{\mathrm{T}} \end{array}\right]^{\mathrm{T}}$ is the $(nm \times N)$ hands Jacobian matrix, depending on the N-vector $\boldsymbol{q}$ of the joint variables — being N the total number of joints—, $\boldsymbol{\tau}_e$ is the external torque, including gravity, Coriolis, centripetal and inertia effects at the fingers joints, and $\boldsymbol{\tau}$ is the torque provided by the actuators.

To ensure that the joint actuators are able to provide the required torques, a joint torque constraint must also be considered

$$\boldsymbol{\tau}_L \leq \boldsymbol{\tau} \leq \boldsymbol{\tau}_H, \tag{5}$$

where $\boldsymbol{\tau}_L$ and $\boldsymbol{\tau}_H$ denote the lower and upper joint torque bound, respectively.

The simultaneous satisfaction of the force balance equation (1), with the friction constraints (2) and (3), and of the joint torque balance equation (4) with constraint (5), implies that the grasp is stable and feasible.

The GFO problem considered here consists in finding the optimal grasp wrench that minimizes the internal forces acting on the object, under the above constraints. The internal forces are contact wrenches that belong to the null space of the grasp matrix $\boldsymbol{G}$. These wrenches $\boldsymbol{c}_{int}$ do not contribute to the balance equation (1), being $\boldsymbol{G}\boldsymbol{c}_{int} = \boldsymbol{0}$, but are used to satisfy the friction cone constraints at the contact points.

III. Grasping Constraints

A. Inequality constraints

As shown in [13], the frictional inequalities (2) and (3) are equivalent to the positive definiteness of the block-diagonal matrix

$$\boldsymbol{F}(\boldsymbol{c}) = \mathrm{diag}\left(\boldsymbol{F}_1(\boldsymbol{c}_1), \ldots, \boldsymbol{F}_n(\boldsymbol{c}_n)\right), \tag{6}$$

where $\boldsymbol{F}_i(\boldsymbol{c}_i)$ is the symmetric (2×2) matrix

$$\boldsymbol{F}_i(\boldsymbol{c}_i) = \left[\begin{array}{cc} c_{i,z} + \dfrac{c_{i,x}}{\mu_i} & \dfrac{c_{i,y}}{\mu_i} \\ \dfrac{c_{i,y}}{\mu_i} & c_{i,z} - \dfrac{c_{i,x}}{\mu_i} \end{array}\right] \tag{7}$$

in the PCWF case, while it is the Hermitian (2×2) matrix

$$\boldsymbol{F}_i(\boldsymbol{c}_i) = \left[\begin{array}{cc} c_{i,z} + \dfrac{c_{i,x}}{\sqrt{\mu_i}} & \dfrac{c_{i,y}}{\sqrt{\mu_i}} - j\dfrac{c_{i,t}}{\sqrt{\mu_{i,t}}} \\ \dfrac{c_{i,y}}{\sqrt{\mu_i}} + j\dfrac{c_{i,t}}{\sqrt{\mu_{i,t}}} & c_{i,z} - \dfrac{c_{i,x}}{\sqrt{\mu_i}} \end{array}\right], \tag{8}$$

in the SFC case.

Similarly, the torque limit constraint (5), in view of the torque balance equation (4), is equivalent to the positive definiteness of the diagonal matrix

$$\boldsymbol{T}(\boldsymbol{c}, \boldsymbol{q}, \boldsymbol{\tau}_e) = \mathrm{diag}\left(\boldsymbol{\tau}_R\right), \tag{9}$$

where

$$\boldsymbol{\tau}_R = \left[\begin{array}{c} \boldsymbol{\tau}_{R,L} \\ \boldsymbol{\tau}_{R,H} \end{array}\right] = \left[\begin{array}{c} \boldsymbol{J}^{\mathrm{T}}(\boldsymbol{q})\boldsymbol{c} - \boldsymbol{\tau}_L + \boldsymbol{\tau}_e \\ -\boldsymbol{J}^{\mathrm{T}}(\boldsymbol{q})\boldsymbol{c} + \boldsymbol{\tau}_H - \boldsymbol{\tau}_e \end{array}\right] \tag{10}$$

are the residual joint actuator torques with respect to the upper ($\boldsymbol{\tau}_{R,H}$) and lower ($\boldsymbol{\tau}_{R,L}$) torque limits.

Hence, the simultaneous satisfaction of both frictional and joint torque constraints is equivalent to the positive definiteness of the linearly constrained block-diagonal matrix

$$\boldsymbol{P} = \mathrm{diag}\left(\boldsymbol{F}, \boldsymbol{T}\right). \tag{11}$$

Notice that the elements of the matrices $\boldsymbol{F}$ and $\boldsymbol{T}$ depend linearly on $\boldsymbol{c}$. Moreover, the force balance equation (1) and the torque balance equation (4) correspond to linear equality constraints imposed on matrix $\boldsymbol{P}$.

B. Linear constraints

Let be $\boldsymbol{c}_i(\boldsymbol{F}_i)$ the linear operator which extracts the contact wrench vector of the i-th contact point from the corresponding frictional constraint matrix, with $i = 1, \ldots, n$, i.e.,

$$\boldsymbol{c}_i(\boldsymbol{F}_i) = \frac{1}{2}\left[\begin{array}{c} \mu_i(f_{i,1,1} - f_{i,2,2}) \\ 2\mu_i f_{i,1,2} \\ (f_{i,1,1} + f_{i,2,2}) \end{array}\right] \tag{12}$$

in the PCWF case, with $\boldsymbol{F}_i = \{f_{i,j,k}\}$ and $j, k = 1, 2$, or

$$\boldsymbol{c}_i(\boldsymbol{F}_i) = \frac{1}{2}\left[\begin{array}{c} \sqrt{\mu_i}(f_{i,1,1} - f_{i,2,2}) \\ \sqrt{\mu_i}(f_{i,1,2} + f_{i,2,1}) \\ (f_{i,1,1} + f_{i,2,2}) \\ \dfrac{\sqrt{\mu_{i,t}}}{j}(f_{i,2,1} - f_{i,1,2}) \end{array}\right] \tag{13}$$

in the SFC case. By denoting with $\boldsymbol{c}(\boldsymbol{F}) = \left[\begin{array}{ccc} \boldsymbol{c}_1(\boldsymbol{F}_1)^{\mathrm{T}} & \cdots & \boldsymbol{c}_n(\boldsymbol{F}_n)^{\mathrm{T}} \end{array}\right]^{\mathrm{T}}$ the linear operator that extracts the contact wrench vector from the frictional constraint matrix, and with $\boldsymbol{\tau}_R(\boldsymbol{T}) = \mathrm{diag}(\boldsymbol{T})$ the linear operator that extracts the diagonal elements of the joint torque constraint matrix, the linear operator $\boldsymbol{\xi}(\boldsymbol{P}) = \left[\begin{array}{cc} \boldsymbol{c}(\boldsymbol{F})^{\mathrm{T}} & \boldsymbol{\tau}_R(\boldsymbol{T})^{\mathrm{T}} \end{array}\right]^{\mathrm{T}}$ can be defined. Therefore, the linear constraints on matrix $\boldsymbol{P}$ imposed by (1) and (4) can be represented in the following affine general form

$$\boldsymbol{A}\boldsymbol{\xi}(\boldsymbol{P}) = \boldsymbol{b} \tag{14}$$

with

$$\boldsymbol{A} = \begin{bmatrix} \boldsymbol{G} & \boldsymbol{O}_{6\times 2N} \\ & \boldsymbol{A}_\tau \end{bmatrix}, \quad \boldsymbol{b} = \begin{bmatrix} \boldsymbol{h}_e \\ \boldsymbol{\tau}_L - \boldsymbol{\tau}_e \\ \boldsymbol{\tau}_H - \boldsymbol{\tau}_e \end{bmatrix}, \tag{15}$$

where $\boldsymbol{A}_\tau$ is the $(2N \times nm + 2N)$ matrix defined as

$$\boldsymbol{A}_\tau = \begin{bmatrix} \boldsymbol{J}(\boldsymbol{q})^{\mathrm{T}} & -\boldsymbol{I}_N & \boldsymbol{O}_N \\ \boldsymbol{J}(\boldsymbol{q})^{\mathrm{T}} & \boldsymbol{O}_N & \boldsymbol{I}_N \end{bmatrix}, \tag{16}$$

being $\boldsymbol{O}_\times$ the null matrix and $\boldsymbol{I}_\times$ the identity matrix of the indicated dimensions. Notice that the dimension of the constraint matrix $\boldsymbol{A}$ increses quadratically with the number of employed fingers (assuming that all the fingers have the same number of actuated joints), while the number of elements of the contraint matrix used in [8] increases at cubic rate.

IV. Semidefinite Programming

A. Cost function

Let be $\mathcal{P}(r)$ the set of positive definite Hermitian matrices

$$\mathcal{P}(r) = \{\boldsymbol{P} \in \mathbb{C}^{r\times r} | \boldsymbol{P} = \boldsymbol{P}' > 0\}, \tag{17}$$

where $\boldsymbol{P}'$ denotes the transpose (Hermitian transpose) of $\boldsymbol{P}$ in the PCWF case (SFC case). The optimization procedure is based on the minimization of the cost function $\Phi(\boldsymbol{P})$: $\mathcal{P}(r) \to \mathbb{R}$, defined as

$$\Phi(\boldsymbol{P}) = \mathrm{tr}\left(\boldsymbol{W}_p\boldsymbol{P} + \boldsymbol{W}_b\boldsymbol{P}^{-1}\right), \tag{18}$$

where $\mathrm{tr}(\cdot)$ denotes the trace operator, and $\boldsymbol{W}_p$ and $\boldsymbol{W}_b$ are symmetric positive definite matrices. Function Φ is a strictly convex twice continuously differentiable function on $\mathcal{P}(r)$ and $\Phi(\boldsymbol{P}) \to +\infty$ for $\boldsymbol{P} \to \partial\mathcal{P}(r)$, being $\partial\mathcal{P}(r)$ the boundary of $\mathcal{P}(r)$.

The first addendum in (18) can be rewritten as

$$\mathrm{tr}(\boldsymbol{W}_p\boldsymbol{P}) = \mathrm{tr}(\boldsymbol{W}_{p,f}\boldsymbol{F}) + \mathrm{tr}(\boldsymbol{W}_{p,t}\boldsymbol{T}). \tag{19}$$

By setting $\boldsymbol{W}_{p,f} = w_{p,f}\boldsymbol{I}$, with $w_{p,f} > 0$, the quantity $\mathrm{tr}(\boldsymbol{W}_{p,f}\boldsymbol{F})$ will depend only on the normal forces $\boldsymbol{c}_{i,z}$ at each contact point, i.e. the pressure forces on the object, that will be minimized. The other components of the contact forces will be minimized as well, because they have to conform to the friction constraints. Different weights can be chosen for the fingers allowing higher contact forces for stronger fingers.

By noting that the sum of two joint torque constraints for the i-th joint is constant and equal to $\tau_{r,i} = \tau_{H,i} - \tau_{L,i}$, with the choice $\boldsymbol{W}_{p,t} = w_{p,t}\boldsymbol{I}$, with $w_{p,t} > 0$, the quantity $\mathrm{tr}(\boldsymbol{W}_{p,t}\boldsymbol{T})$ is constant and will not contribute to the variation of the cost function with the constraint (14). On the other hand, if different weights are assigned to the residual torques of joint i, then joint torques closer to the lower or to the higher limits can be preferred.

The second addendum $\boldsymbol{W}_b\boldsymbol{P}^{-1}$ represents a barrier function, which goes to infinity when $\boldsymbol{P}$ tends to a singularity, i.e. when friction or torque limits are approached. The barrier weight matrix is also chosen diagonal $\boldsymbol{W}_b = \mathrm{diag}(\boldsymbol{W}_{b,F}, \boldsymbol{W}_{b,T})$, with

$$\begin{aligned} \boldsymbol{W}_{b,F} &= w_{b,F}\mathrm{diag}\left(\mu_1, \ldots, \mu_n\right) \\ \boldsymbol{W}_{b,T} &= w_{b,T}\mathrm{diag}\left(\tau_{r,1}, \ldots, \tau_{r,N}, \tau_{r,1}, \ldots, \tau_{r,N}\right), \end{aligned} \tag{20}$$

being $w_{b,F} > 0$, $w_{b,T} > 0$.

Hence, the minimization of the cost function (18) with the linear constraint (14) corresponds to the minimization of the normal contact wrench components applied to the object while satisfying the friction and torque constraints.

B. Gradient flow method on positive definite matrices

The minimization problem can be solved using the linearly constrained gradient flow approach on the smooth manifold of positive definite matrices presented in [35]. It can be shown that $\Phi(\boldsymbol{P})$ presents a unique minimum $\boldsymbol{P}_\infty \in \mathcal{P}(r)$ given by

$$\boldsymbol{P}_\infty = \boldsymbol{W}_p^{-\frac{1}{2}}\left(\boldsymbol{W}_p^{\frac{1}{2}}\boldsymbol{W}_i\boldsymbol{W}_p^{\frac{1}{2}}\right)^{\frac{1}{2}}\boldsymbol{W}_p^{-\frac{1}{2}}, \tag{21}$$

that is the only critical point. The gradient flow $\dot{\boldsymbol{P}}(t) = -\nabla\Phi(\boldsymbol{P}(t))$ on $\mathcal{P}(r)$, defined as

$$\dot{\boldsymbol{P}}(t) = \boldsymbol{P}^{-1}\boldsymbol{W}_i\boldsymbol{P}^{-1} - \boldsymbol{W}_p, \tag{22}$$

ensures that, for every initial condition $\boldsymbol{P}_0 = \boldsymbol{P}(0) \in \mathcal{P}(r)$, $\boldsymbol{P}(t) \in \mathcal{P}(r)$ exists for all $t \geq 0$ and converge exponentially fast to $\boldsymbol{P}_\infty$ at $t \to \infty$.

The affine constraint (14) requires an orthogonal projection of the gradient onto the tangent space. In particular, in Appendix A it is shown that the minimum of $\Phi(\boldsymbol{P})$ can be reached through the linear constrained exponentially convergent gradient flow $\boldsymbol{\xi}(\dot{\boldsymbol{P}}) = -\mathrm{grad}(\Phi(\boldsymbol{P}))$ defined as

$$\boldsymbol{\xi}(\dot{\boldsymbol{P}}) = \boldsymbol{Q}\boldsymbol{\xi}(\boldsymbol{P}^{-1}\boldsymbol{W}_b\boldsymbol{P}^{-1} - \boldsymbol{W}_p), \tag{23}$$

where $\boldsymbol{Q} = (\boldsymbol{I} - \boldsymbol{A}^\ddagger\boldsymbol{A})$ is a linear projection operator onto the null space of $\boldsymbol{A}$, and $\boldsymbol{A}^\ddagger$ is a suitable weighted pseudo-inverse of $\boldsymbol{A}$. Consequently, $\boldsymbol{A}\boldsymbol{Q} = \boldsymbol{O}$ and $\boldsymbol{A}\boldsymbol{\xi}(\dot{\boldsymbol{P}}) = \boldsymbol{0}$; hence, if the initial point satisfies (14), $\boldsymbol{P}(t)$ will satisfy the constraint for all $t \geq 0$.

A discrete-time version of (23) based on the Euler numerical integration algorithm is

$$\boldsymbol{\xi}(\boldsymbol{P}_{k+1}) = \boldsymbol{\xi}(\boldsymbol{P}_k) + \alpha_k\boldsymbol{Q}_k\boldsymbol{\xi}(\boldsymbol{P}_k^{-1}\boldsymbol{W}_b\boldsymbol{P}_k^{-1} - \boldsymbol{W}_p), \tag{24}$$

where the step size α_k is chosen to ensure down-hill steps. Notice that the choice of α_k strongly affects the performance of the optimization algorithm. A wrong choice could determine a very slow convergence or the break of the barrier. Several strategies have been proposed for the self-tuning of α_k at each iteration (see [16] for details). The sensitivity to the step size choice can be reduced by adopting a Dikin-type recursive algorithm [10], [36], that leads to the discrete flow

$$\boldsymbol{\xi}(\boldsymbol{P}_{k+1}) = \boldsymbol{\xi}(\boldsymbol{P}_k) - \alpha_k\boldsymbol{Q}_k\frac{\boldsymbol{\xi}(\boldsymbol{P}_k^{-1}\boldsymbol{W}_b\boldsymbol{P}_k^{-1} - \boldsymbol{W}_p)}{\|\boldsymbol{P}_k^{-1}\boldsymbol{W}_b\boldsymbol{P}_k^{-1} - \boldsymbol{W}_p\|_{\boldsymbol{P}_k}}, \tag{25}$$

where $\|\boldsymbol{X}\|_{\boldsymbol{Y}} = \mathrm{tr}(\boldsymbol{Y}^{-1}\boldsymbol{X}\boldsymbol{Y}^{-1}\boldsymbol{X})$, and $0 \leq \alpha_k \leq 1$ can be evaluated with a bounded line search minimizing $\Phi(\boldsymbol{P}_{k+1})$.

V. Initial Point Evaluation

The previous optimization algorithm requires the computation of a valid initial solution —a positive definite matrix $\boldsymbol{P}_0$ satisfying (14)— at each step time. The presence of joint torque constraints could make the evaluation of this initial value not so trivial as for the unconstrained case. Namely, a

matrix $\boldsymbol{P}'_0$ which satisfies (14) can be easily found, but it not necessarily positive definite.

By observing that the positive definiteness of a given matrix is equivalent to the positive definiteness of its eigenvalues, i.e. of its minimum eigenvalue [17], a new algorithm based on a constrained monotone increasing flow of the minimum eigenvalue until it becomes positive is proposed here.

Let $\lambda_r(\boldsymbol{P}_0)$ be the minimum eigenvalue of $\boldsymbol{P}_0$. A constrained monotone increasing flow of $\lambda_r(\boldsymbol{P}_0)$ is given by

$$\boldsymbol{\xi}(\dot{\boldsymbol{P}}_0) = \boldsymbol{Q}\boldsymbol{\xi}(\boldsymbol{v}_r\boldsymbol{v}_r^{\mathrm{T}}), \tag{26}$$

where $\boldsymbol{v}_r$ is the eigenvector corresponding to the minimal eigenvalue λ_r (for the proof see Appendix B).

A discrete-time version of the previous flow (26) based on the Euler numerical integration algorithm is

$$\boldsymbol{\xi}(\boldsymbol{P}_{0,k+1}) = \boldsymbol{\xi}(\boldsymbol{P}_{0,k}) + \beta_k\boldsymbol{Q}_k\boldsymbol{\xi}(\boldsymbol{v}_{r,k}\boldsymbol{v}_{r,k}^{\mathrm{T}}), \tag{27}$$

where $\boldsymbol{v}_{r,k} = \boldsymbol{v}_r(\boldsymbol{P}_{0,k})$ and β_k is the integration step size. The algorithm starts from a solution $\boldsymbol{P}'_0$ of (14), that can be computed from $\boldsymbol{c}'_0 = \boldsymbol{G}_0^{\dagger}\boldsymbol{h}_{e,0}$; then, the projected gradient flow (27) is applied until a positive definite matrix $\boldsymbol{P}_0$ is found.

Notice that the evaluation of the gain factor β_k could impact on the performance of the algorithm. To this purpose, a linear search algorithm to find a value of β_k maximizing λ_r can be employed.

VI. Improvements For Real-Time Application

A. Affine constraint decomposition

The proposed algorithm requires the inversion of a $(6+2N)$ square matrix needed for the evaluation of $\boldsymbol{A}^{\ddagger}$ at each iteration, also when the grasping configuration is unchanged, i.e. when $\boldsymbol{G}$ is constant, due to the variation in (16). Starting from the discrete version of the gradient flow (24) for a given grasp configuration, it can be shown that:

$$\boldsymbol{c}_{k+1} = \boldsymbol{c}_k + \alpha_k\bar{\boldsymbol{Q}}_k\boldsymbol{\xi}(\boldsymbol{P}^{-1}(\boldsymbol{c}_k)\boldsymbol{W}_b\boldsymbol{P}^{-1}(\boldsymbol{c}_k) - \boldsymbol{W}_p), \tag{28}$$

where $\bar{\boldsymbol{Q}}_k = (\boldsymbol{I} - \boldsymbol{G}_k^{\ddagger}\boldsymbol{G}_k)[\ \boldsymbol{I}_{nm} \quad \boldsymbol{O}_{nm,2N}\ ](\boldsymbol{I} - \boldsymbol{A}_{\tau,k}^{\ddagger}\boldsymbol{A}_{\tau,k})$ is the result of the projection onto the null space of matrix $\boldsymbol{A}_\tau$ in (16), which guarantees the coherence of the elements of matrix $\boldsymbol{P}$, and of the subsequent projection onto the null space of the grasp matrix, ensuring that the force balance constraint (1) is fullfilled. Notice that the matrix $[\ \boldsymbol{I}_{nm} \quad \boldsymbol{O}_{nm,2N}\ ]$ selects only the first nm components of matrix $(\boldsymbol{I} - \boldsymbol{A}_\tau^{\ddagger}\boldsymbol{A}_\tau)$. The expressions of $\boldsymbol{G}_k^{\ddagger}$ and $\boldsymbol{A}_\tau^{\ddagger}$ are given in Appendix A.

The computational complexity for the evaluation of $\boldsymbol{Q}$ is $O((6+2N)^{2.376} + 2(6+2N)^2(nm+2N) + (6+2N)(nm+2N)^2)$, using Coppersmith-Winograd algorithm for matrix inversion, while for the evaluation of $\bar{\boldsymbol{Q}}$ it is $O(6^{2.376} + (2N)^{2.376} + nm(72+7nm) + 8N^2(nm+2N) + (1+2N)(nm+2N)^2)$, which is lower than the previous quantity already for a small value of the ratio $N/(nm)$. Moreover, if the grasp configuration remains unchanged, i.e. $\boldsymbol{G}$ does not change, the projector $(\boldsymbol{I} - \boldsymbol{G}^{\ddagger}\boldsymbol{G})$ can be evaluated off-line with a further computational complexity reduction.

Figure 1 shows the number of joint torque constraints N for which the adoption of the affine constraint decomposition (28) results in a reduction of computational complexity, versus the number of contact points n. The computational complexity was measured in terms of number of flops (floating point operations) using Matlab. A reduction is achieved for all the couples (n, N) that are outside the colored areas.

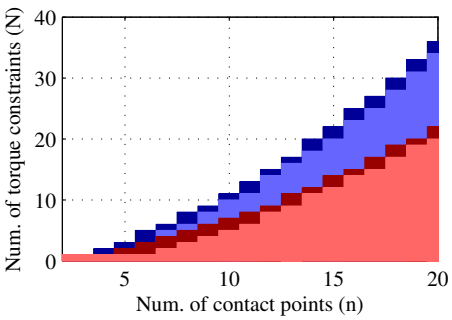

Fig. 1. Evaluation of the reduction of the computational cost with the affine constraint decomposition (28). A reduction is achieved for all the couples (n, N) that are outside the red (blue) area. The red area correspond to the PCWF case ($m = 3$) while the blue area correspond to the SFC case ($m = 4$). The darker (red and blue) areas correspond to the case of time varying grasp configuration.

Notice that the same decomposition can be adopted also for the gradient flows (25) and (27).

B. Dynamic joint-torque constraints selection

Under the reasonable assumption that the solutions of the optimization algorithm evaluated at successive sampling times are quite close, the number of active joint constraints can be reduced. For example, if for the current optimal solution the actuator of joint i provides a torque close to the upper bound $\tau_{H,i}$, then the constraint on the lower bound $\tau_{L,i}$ can be deactivated at the next sampling time, being the corresponding barrier term in the cost function negligible. More in general, for a given grasp configuration, if a measured contact force has small variations, it is realistic to assume that the corresponding joint torques will not change significantly at the next sampling time. Therefore, the number of joint torque constraints can be dynamically reduced, by using the values of the residual torques, computed at the previous sampling time, as the criterion for selecting the constraints to be activated. Only those constraints related to residuals that are higher than a given threshold (set as a fraction $\sigma_\tau \in [0, 1]$ of the corresponding torque limit) will be activated. Notice that the case $\sigma_\tau = 0$ corresponds to the selection of the closest torque limits (the upper or the lower one), and thus only half of the total number of torque constraints are activated. The higher is σ_τ, the lower is the number of active constraints. The case $\sigma_\tau = 1$ corresponds to the unconstrained case.

Chattering phenomena on constraints activation/disactivation can be avoided by adopting a double threshold with hysteresis.

C. Initial point iterative self-evaluation

The evaluation of the initial point must be performed at the initial time $t_j = 0$, but also at each sampling time $t_j > 0$, i.e., at the beginning of each optimization cycle. In fact, both

the hand configuration $\boldsymbol{q}$, the external wrench $\boldsymbol{h}_e$, and the grasp configuration $\boldsymbol{G}$ may change with time, and thus the optimal solution at time t_{j-1} could be not compatible with the constraints at time t_j. However, a significant reduction of computational time can be achieved by adopting, at time $t_j > 0$, the iterative algorithm proposed here.

Namely, let define the following "virtual" quantities, for $k \geq 0$:

$$\boldsymbol{c}'_k = (\boldsymbol{I} - \boldsymbol{G}^{\dagger}_{t_j}\boldsymbol{G}_{t_j})\boldsymbol{c}_k + \gamma_k \boldsymbol{G}^{\dagger}_{t_j}\boldsymbol{h}_{e,t_j} + \bar{\gamma}_k \boldsymbol{G}^{\dagger}_{t_{j-1}}\boldsymbol{h}_{e,t_{j-1}} \quad (29)$$

$$\boldsymbol{q}'_k = \gamma_k \boldsymbol{q}_{t_j} + \bar{\gamma}_k \boldsymbol{q}_{t_{j-1}} \quad (30)$$

$$\boldsymbol{\tau}'_k = \boldsymbol{J}^{\mathrm{T}}(\boldsymbol{q}'_k)\boldsymbol{c}'_k + \gamma_k \boldsymbol{\tau}_{e,t_j} + \bar{\gamma}_k \boldsymbol{\tau}_{e,t_{j-1}} \quad (31)$$

where $\gamma_k \in (0, 1]$, $\bar{\gamma}_k = 1 - \gamma_k$, and $\boldsymbol{G}^{\dagger}$ is a pseudo-inverse of $\boldsymbol{G}$. The initial condition is set as $\boldsymbol{c}_0 = \boldsymbol{c}(t_{j-1})$. The coefficient $\gamma_k \in (0, 1]$ is chosen at each iteration according to a monotone sequence, using a linear search algorithm, as the maximum value that does not produce invalid solutions, i.e.

$$\gamma_k = \max\{\gamma \in (\gamma_{k-1}, 1] \ \mid \ \boldsymbol{P}(\boldsymbol{c}'_k, \boldsymbol{\tau}'_k) > 0\}, \quad (32)$$

with $\gamma_{-1} = 0$. Hence, the following modified gradient flow is computed:

$$\boldsymbol{c}_{k+1} = \boldsymbol{c}'_k + \alpha_k \bar{\boldsymbol{Q}}'_k \boldsymbol{\xi}(\boldsymbol{P}'^{-1}_k \boldsymbol{W}_b \boldsymbol{P}'^{-1}_k - \boldsymbol{W}_p). \quad (33)$$

where $\bar{\boldsymbol{Q}}'_k = \bar{\boldsymbol{Q}}(\boldsymbol{q}'_k, \boldsymbol{\tau}'_k, \boldsymbol{G}_{t_j})$ and $\boldsymbol{P}'_k = \boldsymbol{P}(\boldsymbol{c}'_k, \boldsymbol{\tau}'_k)$.

The algorithm works as follows. If the optimal solution $\boldsymbol{c}(t_{j-1})$ is still compatible with the constraints at the current time t_j, then γ_0 is set to 1 at the first iteration, and thus $\boldsymbol{c}'_k = \boldsymbol{c}_k$, $\boldsymbol{q}'_k = \boldsymbol{q}_k$, $\boldsymbol{\tau}'_k = \boldsymbol{\tau}_k$, and the gradient flow (33) is the same as (28). On the other hand, if $\boldsymbol{c}(t_{j-1})$ is not compatible with the constraints, a value $0 < \gamma_0 < 1$ is likely to be found so that $\boldsymbol{P}(\boldsymbol{c}'_k, \boldsymbol{\tau}'_k) > 0$. The reason is that, when γ_0 is close to zero, the virtual contact force vector $\boldsymbol{c}'_k$ has to balance a "virtual" external wrench close to that of the previous sampling time $\boldsymbol{h}_{e,t_{j-1}}$; moreover, the quantities $\boldsymbol{q}'_k$ and $\boldsymbol{\tau}'_k$ are close to the joint positions and torques at the previous sampling time. The new solution computed with the modified gradient flow (33), thanks to the effect of the barrier function in the cost function, goes away from the boundary. This allows to increase γ_k at each step, i.e., to balance an external wrench closer to $\boldsymbol{h}_{e,t_j}$ as well as to consider joint positions and torques closer to $\boldsymbol{q}_{e,t_j}$ and $\boldsymbol{\tau}_{e,t_j}$ respectively, until $\gamma_k = 1$. Obviously, the optimization cycle can not end until $\gamma_k = 1$.

Notice that, if $\boldsymbol{P}(\boldsymbol{c}'_k) \leq 0$ also with $\gamma_0 = 0$, then a new initial solution has to be evaluated with the algorithm proposed in Section V.

D. Iterative single-hand grasping force optimization

To speed up the computation, a further simplification in the algorithm can be introduced by splitting the bimanual optimization problem into two simpler single-hand problems. In this case, the initial point iterative self-evaluation algorithm presented in Subsection VI-C can be employed to find the initial common solution. Then two independent optimization procedures can be started separately for each hand, and the corresponding solutions are composed only at the end to achieve a unique wrench vector solution. The price to pay with the simplified algorithm is that the solution is not optimal in a global sense and it is possible that the single-hand optimization problems have no solution.

This latter situation can be avoided in most cases by considering a suitable weighted pseudo-inverse of the grasp matrix in (29), with the goal of achieving a load sharing between the hands. In detail, at each sampling time, the minimum residual torque is evaluated for for the right ($\delta_{\tau,r}$) and left ($\delta_{\tau,l}$) hand. Then the weighting matrix:

$$\boldsymbol{W}_G = \mathrm{diag}\left(\frac{\delta_{\tau,r} + \delta_{\tau,l}}{\delta_{\tau,r}}\boldsymbol{I}_{n_r m}, \frac{\delta_{\tau,r} + \delta_{\tau,l}}{\delta_{\tau,l}}\boldsymbol{I}_{n_l m}\right), \quad (34)$$

is computed, where n_r and n_l are the number of contact points for the right and for the left hand, respectively. The above matrix is used for the evaluation of the weighted pseudo-inverse of the grasp matrix

$$\boldsymbol{G}^{\dagger} = \boldsymbol{W}_G^{-1}\boldsymbol{G}(\boldsymbol{G}\boldsymbol{W}_G^{-1}\boldsymbol{G}^{\mathrm{T}})^{-1}. \quad (35)$$

With this choice, the quadratic form $\boldsymbol{c}^{\mathrm{T}}\boldsymbol{W}_G\boldsymbol{c}$ is minimized, resulting in a redistribution of the load between the hands in reason of their capability to provide torques.

This approach, as demonstrated in the following case study, can produce a reduction of up to 50% of the computational time when a large number of joint torque constraints are active. Moreover this approach allows to compute in parallel the two GFO problems for the two hands, thus achieving a further 50% reduction of the computational time in case of multi-threading execution.

VII. Case Study

The proposed GFO algorithm has been tested in simulation using two DEXMART hands [37] (5 fingers with 20 joints, 15 are independent) mounted on an anthropomorphic torso (7 DOFs for each arm plus 3 DOFs for the torso), as shown in Fig. 2. The total number of actuated joints of the two hands is $N = 30$. It is assumed that the external wrench acting on the object is estimated by using the contact force measurements provided by force/tactile sensors mounted on the fingertips of the DEXMART hands [38]. The hands grasp a cylinder representing a bottle half filled with water in two different grasp cases: $n = 5 + 5$ (all fingers are employed) and $n = 3 + 3$ (only the thumb, the forefinger, and the little finger are employed for each hand). The case of contact points with friction is considered, i.e. $m = 3$, while the task consists in pouring water by reorienting the bottle. In details, the bottle is initially grasped with the main axis aligned to the vertical direction, then the task can be decomposed into three phases:

- a rotation of $135\,\mathrm{deg}$ about the horizontal axis through the geometric center of the cylinder is commanded;
- the hand is stopped while some water is poured from the bottle (the mass and inertia of the bottle change accordingly);
- the opposite rotation is commanded to bring the bottle back to the initial pose.

The joint trajectories of the robot have been computed using a classical closed loop inverse kinematics algorithm. The contact points on the bottle have been set fixed, so that the grasp

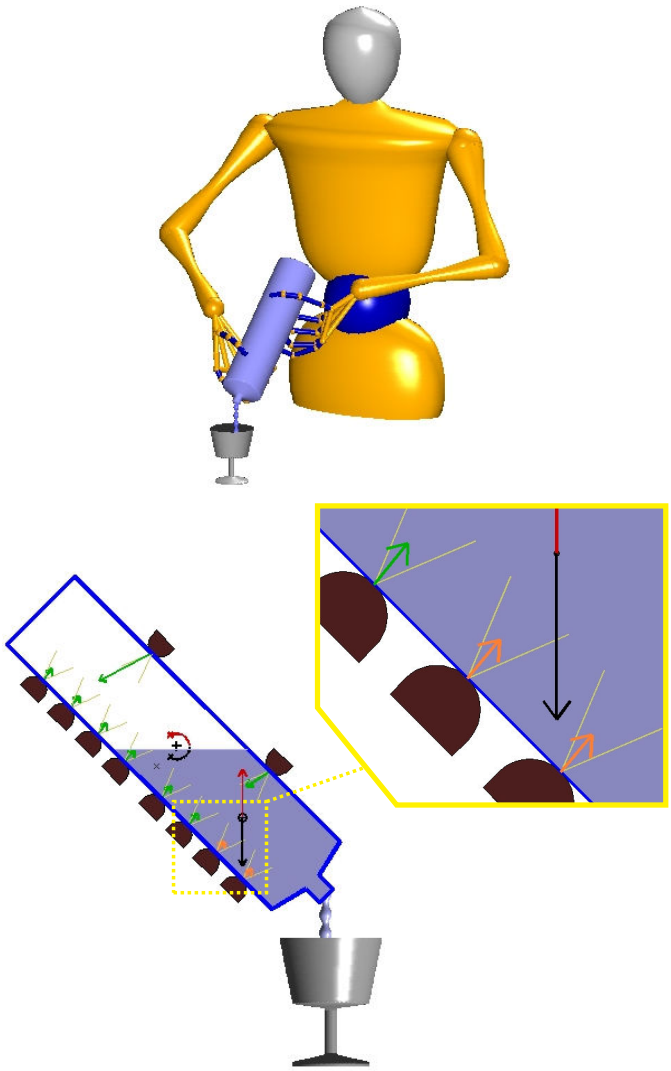

Fig. 2. Top: Anthropomorphic torso with two DEXMART hands grasping a bottle. Bottom Section of the grasped bottle with graphical representation of the gravity force and torque (black arrows), of the resultant force and torque applied by the fingers (red arrows), of the optimal contact forces (green arrows if not affected by joint torque constraints, orange arrow otherwise), and of the friction cones.

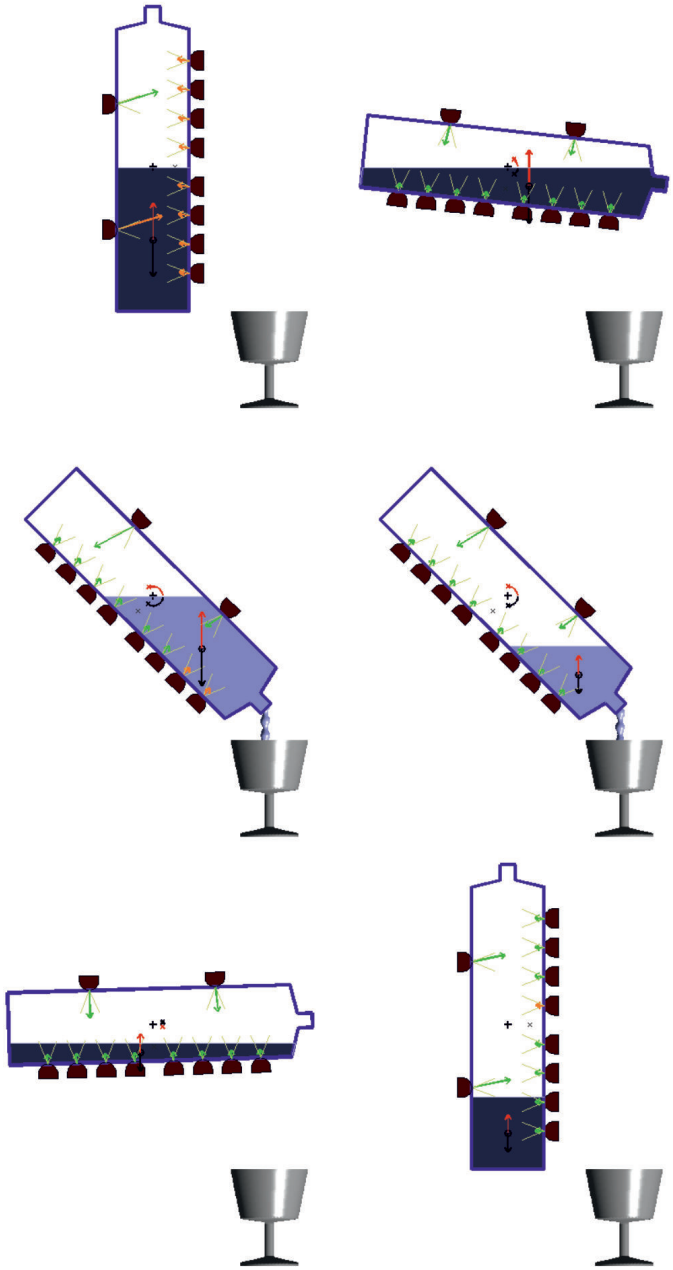

Fig. 3. Sequence of significant configurations of the bottle and of the forces during task execution with $n = 10$.

matrices are constant, while the Jacobian matrices of the two hands are variable.

A dynamic simulation has been performed using Matlab/Simulink, where the variation of the position of the center of mass of the water and that of its weight have been considered. Figure 2 shows on the bottom a section of the bottle half filled with water. The intensity of the gravity force is proportional to the length of the black vertical arrow applied to the instantaneous center of mass (of length proportional to the intensity of the force), while the intensity of the gravity torque with respect to the center of the bottle is proportional to the length of the black circular arrow. The red arrows represent the external force and torque balancing the gravity effects and resulting from the contact forces applied by the fingers. These latter are represented by green arrows if not affected by joint torque constraints, orange arrows otherwise. The friction cone limits in the contact points are colored in yellow. A sequence of significant configurations of the bottle during task execution with $n = 10$ is shown in Fig. 3.

A. *Joint torque limits constraints*

The action of the friction and of joint torque limits constraints is shown considering two different simulations: in the first one only the friction constraint is considered, without any constraint on the joint torque limits, while in the second one different torque limits are set for the fingers. In particular, the maximum torques of the thumb actuators are assumed higher than those of the actuators of the other fingers of the hand (± 0.5 vs. $\pm 0.075\,\mathrm{Nm}$ for the cases $n = 10$, ± 0.5 vs. $\pm 0.125\,\mathrm{Nm}$ for the cases $n = 6$).

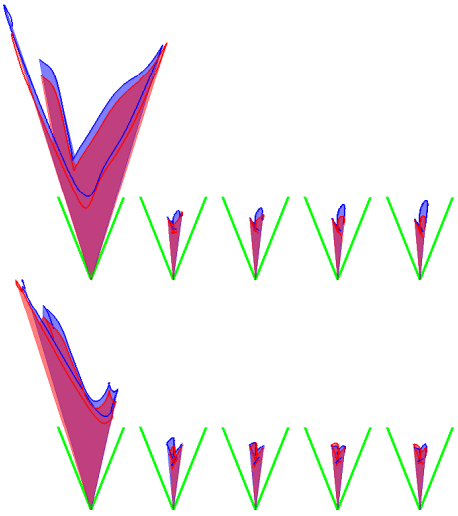

Fig. 4. Evolution of the contact forces at the fingers from the thumb (left) to the little finger (right), in the case $n = 10$. Top: right hand. Bottom: left hand. The blue (red) curves correspond to the forces without (with) torque constraints. The green lines represent the friction cones.

In Fig. 4, a synthetic graphical representation of the evolution of the contact forces on the plane containing all the contact points is provided, for the case $n = 10$. In detail, for a given contact point, the couple of green lines represents the friction cone limits, while the contact force at a given time is represented by a line segment that begins in the contact point. The other endpoint draws over time the red (blue) curve, in the case that joint constraints are (are not) considered. The red/blue shadowed areas are the cones filled by the contact forces during the execution of the task. As expected, the contact forces remain always inside the friction cones, in both simulation cases, according to the barrier function considered into the cost function (18).

In Fig. 5 a comparison of the norm of the contact wrenches and of the joint torques is shown, with and without joint torque constraints, in the case $n = 10$. The difference between the torques in the two cases is very small while the contact wrenches are smaller (in norm) in the presence of torque constraints, which impose a better distribution of the load between the fingers.

The time history of the minimum residual torque (for all the actuators) is shown in Fig. 6. The effect of the barrier function acting also on the torques allows full respect of the limits, without affecting significantly the contact wrenches as shown in Fig. 5.

B. Affine constraint decomposition

The effectiveness of the proposed affine constraint decomposition has been verified by computing the amount of flops which are required for the evaluation of the matrices $\boldsymbol{Q}$ in (28) and $\bar{\boldsymbol{Q}}$ in (23), respectively. In particular, for the considered case study with $n = 10$, when all the torque joint constraints are activated ($N = 30$), the evaluation of $\boldsymbol{Q}$ requires $3.54\,\text{Mflops}$ ($0.28\,\text{ms}$ on a Intel single-core at $2.8\,\text{GHz}$), while the evaluation of $\bar{\boldsymbol{Q}}$, for a time-varying grasp configuration, requires $3.02\,\text{Mflops}$ ($0.24\,\text{ms}$), with a reduction

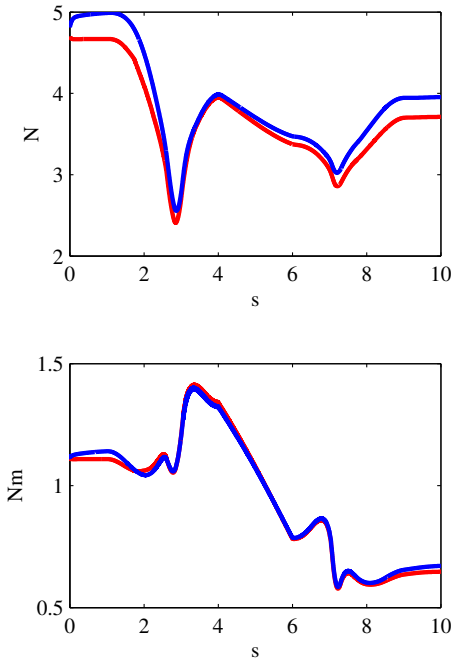

Fig. 5. Time history of the norm of the contact wrenches (top) and of the joint torques (bottom) in the case $n = 10$, with (red) and without (blue) joint torque constraints.

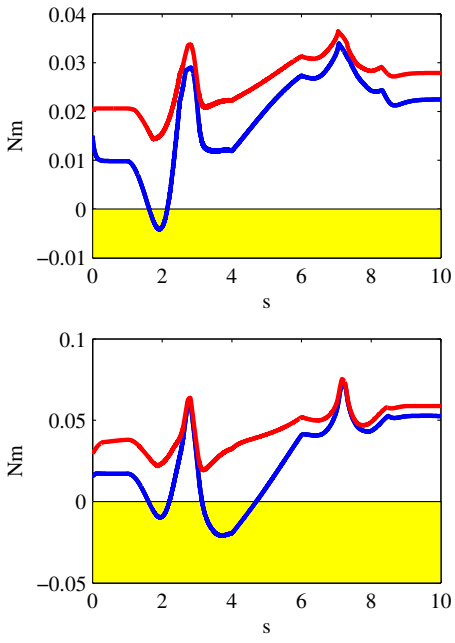

Fig. 6. Time history of the residual torques (*Top*: case $n = 10$, *Bottom*: case $n = 6$), with (red) and without (blue) joint constraints. Negative values (the yellow area) correspond to the violation of one or more joint torque limits.

of 14.8%. On the other hand, for the case of constant grasp configuration, the evaluation of $\bar{\boldsymbol{Q}}$ requires $3.01\,\text{Mflops}$, which corresponds to a reduction of 15.2%.

C. On-line joint torque constraints selection

The benefits resulting from the adoption of the on-line joint-torque constraints selection are shown in Fig. 7, where the time

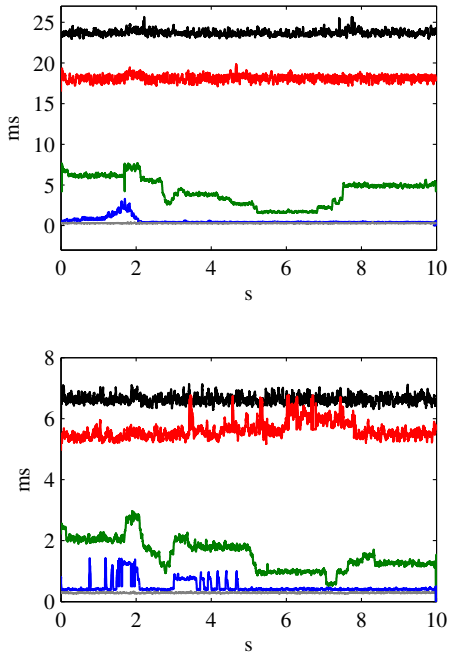

Fig. 7. Time history of the computational time effort for the cases with all constraints (black), $\sigma_\tau = 0$ (red), $\sigma_\tau = 0.5$ (green), $\sigma_\tau = 0.8$ (blue), and unconstrained (gray). Top: case $n = 10$. Bottom: case $n = 6$.

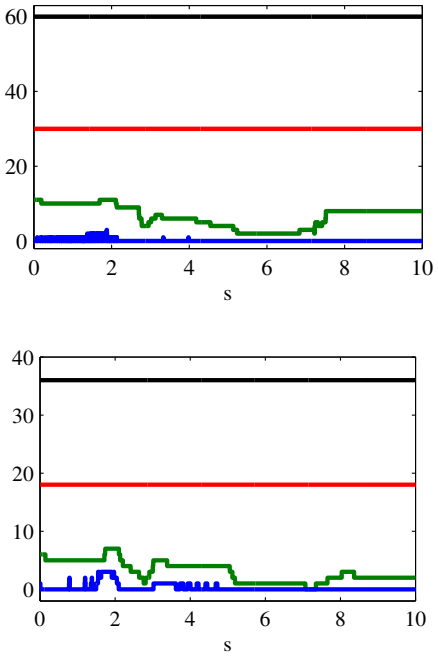

Fig. 8. Time history of the number of active joint torque constraints for the cases with: all constraints (black), $\sigma_\tau = 0$ (red), $\sigma_\tau = 0.5$ (green), $\sigma_\tau = 0.8$ (blue), and no constraints (gray). Top: case $n = 10$. Bottom: case $n = 6$.

history of the computational time effort is represented. All the presented simulations are performed on an Intel i7 dual-core at 2.8 Ghz. The corresponding number of active joint torque constraints is reported in Fig. 8.

In particular, four different cases are compared: all constraints (black lines), $\sigma_\tau = 0$ (red lines), $\sigma_\tau = 0.5$ (green lines), $\sigma_\tau = 0.8$ (blue lines), and unconstrained (gray lines), where σ_τ is the threshold for joint torque constraints activation. By considering the case $n = 10$, the achieved reduction of the average computational time varies between a minimum of about 24% for $\sigma_\tau = 0$ to a maximum of about 98% for $\sigma_\tau = 0.8$. The numerical values of the average computational times for different values of σ_τ are shown in Table I.

TABLE I
AVERAGE COMPUTATIONAL TIMES

Case	$n = 5 + 5$ [ms]	$n = 3 + 3$ [ms]
All constraints	23.7	6.64
$\sigma_\tau = 0$	18.1	5.64
$\sigma_\tau = 0.5$	4.21	1.53
$\sigma_\tau = 0.8$	0.56	0.48
No constraints	0.30	0.29

Notice that, already with $\sigma_\tau = 0.5$ the computational time is less than 5 ms for the case $n = 10$ and 2 ms for the case $n = 6$. With $\sigma_\tau = 0.8$ a computational time less than 1 ms is achieved for both $n = 6$ and $n = 10$ cases. Those values are fully compatible with force feedback control, which typically runs at $0.5 \div 1$ kHz. Higher values of σ_τ allow to reduce the computational time at the expense of a less balanced torque distribution among the joints. Namely, some torques can reach values that are closer to their limits, thus producing higher consumption of the corresponding actuators, gears and/or tendons.

D. Initial point iterative self-evaluation

Assuming that all the constraints are active, the average computational time in the case that the initial point self-evaluation algorithm is not employed, is 31 ms in the case $n = 10$ and 11.5 ms in the case $n = 6$. The corresponding average times achieved using the initial-point iterative self-evaluation algorithm are those reported in the first row of Table I. Therefore, the proposed algorithm allows a reduction of computational time of about 23% in the case $n = 10$ and about 42% in the case $n = 6$.

E. Iterative single-hand grasp force optimization

The adoption of the sub-optimal single-hand GFO algorithm can provide a significant reduction of the computational time up to 50% with respect to the optimal dual-hand algorithm, as shown in Fig. 9 for the case $n = 10$, assuming that all the joint torque constraints are active. The adoption of a multi-threading parallel computing on a dual-core CPU can further reduce the overall computation time by 70%.

However, the sub-optimal solution has reduced performance in terms both of the norm of contact wrenches and of the norm of the joint torques (see Fig. 10). Consequently, also the residual torques are significantly reduced, as shown in Fig. 11, although the imposed torque constraints are never violated. As shown in these figures, the adoption of the weighted pseudo-inverse of the grasp matrix in (35) can improve the achieved solution resulting in a well-shared load between the two hands. This behavior is mainly due to the reduction of the

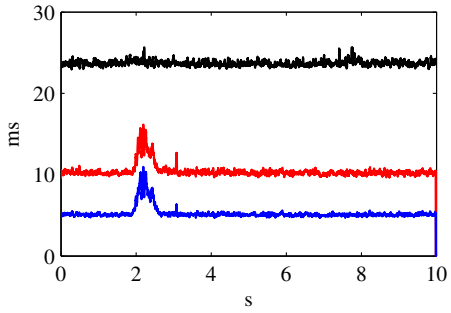

Fig. 9. Time history of the computational time effort for the cases of dual-hand (black) and single-hand GFO without (red) and with (blue) parallel computing, in the case $n = 10$ and considering all the constraints.

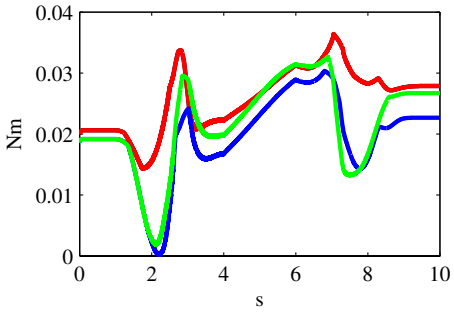

Fig. 11. Time history of the minimum residual torque ($n = 10$) for the cases of local single-hand optimization without (blue) and with (green) weighted pseudo-inverse of the grasp matrix, and global dual-hand (red).

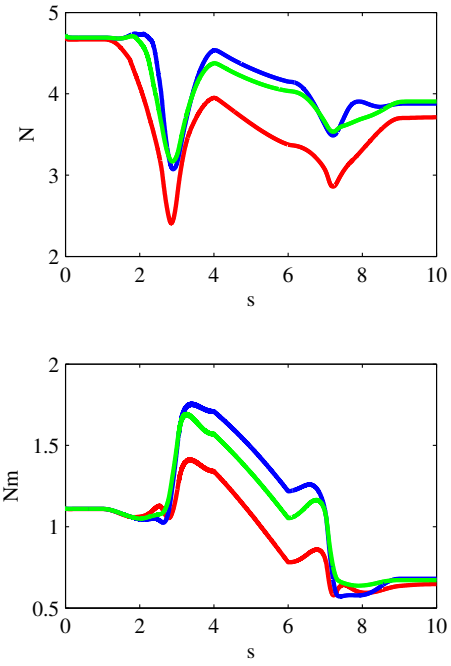

Fig. 10. Time history of the norm of the contact wrenches (top) and of the joint torques (bottom) with $n = 10$ for the cases of local single-hand optimization without (blue) and with (green) weighted pseudo-inverse of the grasp matrix, and global dual-hand (red).

DOFs available to the optimization algorithm by considering separately the two hands instead of both together.

On top of Fig. 12 the time history of the normalized load-sharing coefficients $\delta_{\tau,r}/(\delta_{\tau,r}+\delta_{\tau,l})$ (red) and $\delta_{\tau,l}/(\delta_{\tau,r}+\delta_{\tau,l})$ (blue) employed in (35) is shown in the case $n = 10$, while the time history of the norm of the load force and moments for the right and left hand in the cases of sub-optimal and optimal method are shown on the bottom-left and bottom-right of the figure. As expected, the whole balance of effort between the hands is degraded with respect to the optimal solution, despite the adoption of the online load sharing technique.

VIII. Conclusion

A new algorithm for on-line grasping force optimization for bimanual dextrous-hand robotic systems has been presented in this paper, considering also joint torque limits. The computational load of the algorithm has been reduced by adopting an iterative formulation based on a dynamic set of active constraints and by avoiding the evaluation of the initial point at the beginning of each iteration, as required by alternative approaches. Moreover, a sub-optimal single-hand optimization algorithm has been proposed and compared with respect to the dual-hand centralized algorithm. Finally, a new initial-point evaluation algorithm has been proposed. A simulation case study has been presented to show the feasibility and the effectiveness of the proposed technique.

Appendix A
Exponential convergence of the constrained gradient flow (23)

The existence of $\boldsymbol{P}_\infty$, its uniqueness, as well as the fact that every critical point of Φ is a global minimum have been addressed in [35], while the proof of convergence of (22) has been presented in [8].

The proof the exponential convergence of the linearly constrained gradient flow (23) can be made similarly to [8], [35], although the extension is not straightforward.

The first step is to prove that equation (23) defines a constrained gradient flow of Φ with respect to a suitable Riemannian metric on the smooth manifold

$$\mathcal{C} = \{\boldsymbol{P} \in \mathbb{C}^{r\times r} | \boldsymbol{P} = \boldsymbol{P}' > 0, \boldsymbol{A\xi}(\boldsymbol{P}) = \boldsymbol{b}\}$$

of the Hermitian positive definite matrices that satisfy the linear constraints. To this purpose, it is convenient to start from the unconstrained case, i.e. from the mapping $\Phi : \mathcal{P}(r) \to \mathbb{R}$ defined on the smooth manifold $\mathcal{P}(r)$ of the Hermitian positive definite matrices. Using standard differential geometry concepts, the derivative of $\Phi : \mathcal{P}(r) \to \mathbb{R}$ at $\boldsymbol{P}$ is defined as the linear map $D_\Phi|_{\boldsymbol{P}}(\boldsymbol{X}) : \mathrm{T}_{\boldsymbol{P}}(\mathcal{P}(r)) \to \mathbb{R}$

$$D_\Phi|_{\boldsymbol{P}}(\boldsymbol{X}) = \mathrm{tr}((\boldsymbol{W}_p - \boldsymbol{P}^{-1}\boldsymbol{W}_i\boldsymbol{P}^{-1})\boldsymbol{X}), \quad \boldsymbol{X} \in \mathrm{T}_{\boldsymbol{P}}(\mathcal{P}(r)), \tag{36}$$

where $\mathrm{T}_{\boldsymbol{P}}(\mathcal{P}(r))$ is the tangent space of manifold $\mathcal{P}(r)$ at $\boldsymbol{P}$. This definition coincides with the usual derivative of $\Phi(\boldsymbol{P})$, when expressed in local coordinates.

To define the gradient vector field $\nabla\Phi$ of the mapping $\Phi : \mathcal{P}(r) \to \mathbb{R}$, it is required to specify a Riemannian metric on $\mathcal{P}(r)$. The standard Riemannian metric is:

$$\langle\langle \boldsymbol{X}, \boldsymbol{Y} \rangle\rangle = \mathrm{tr}(\boldsymbol{X}'\boldsymbol{Y}) = \mathrm{vec}(\boldsymbol{X})'\mathrm{vec}(\boldsymbol{Y}) \quad \boldsymbol{X}, \boldsymbol{Y} \in \mathrm{T}_{\boldsymbol{P}}(\mathcal{P}(r)), \tag{37}$$

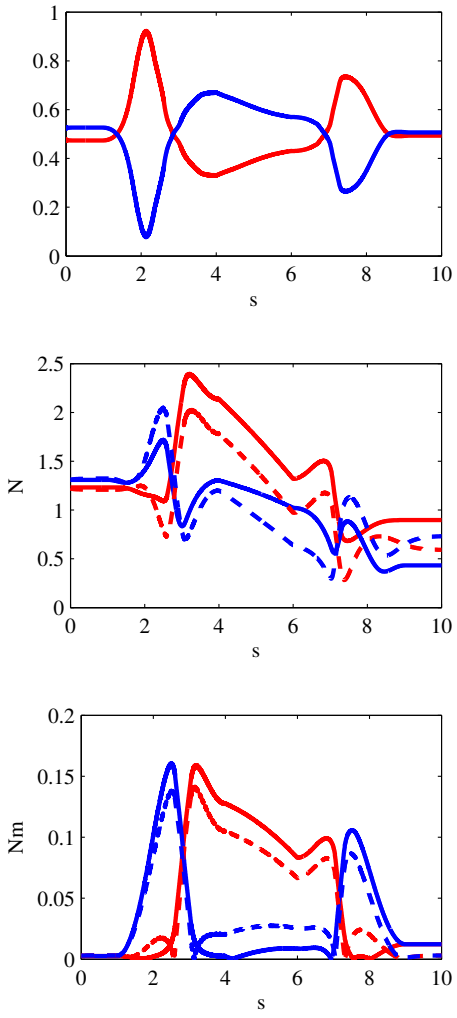

Fig. 12. *Top*: Time history of the normalized load-sharing coefficients $\delta_{\tau,r}/(\delta_{\tau,r}+\delta_{\tau,l})$ (red) and $\delta_{\tau,l}/(\delta_{\tau,r}+\delta_{\tau,l})$ (blue) in the case $n=10$. *Middle*: Time history of the norm of the load force for the right (red) and left (blue) hand in the cases of local (continuous lines) and global (dashed lines) method. Bottom: time histories of the norm of the corresponding load moments.

where the vec$(\cdot)$ operator stacks the columns of a matrix into a vector. The gradient vector field $\nabla\Phi(\boldsymbol{P})$ is uniquely characterized by the following two properties:

1) $\nabla\Phi(\boldsymbol{P}) \in \mathrm{T}_{\boldsymbol{P}}(\mathcal{P}(r))$ for all $\boldsymbol{P} \in \mathcal{P}(r)$
2) $D_\Phi|_{\boldsymbol{P}}(\boldsymbol{X}) = \langle\langle \nabla\Phi(\boldsymbol{P}), \boldsymbol{X}\rangle\rangle$ for all $\boldsymbol{X} \in \mathrm{T}_{\boldsymbol{P}}(\mathcal{P}(r))$.

Therefore, in view of the expression (36), by using the Riemannian metric (37), it is

$$\nabla\Phi(\boldsymbol{P}) = \boldsymbol{W}_p - \boldsymbol{P}^{-1}\boldsymbol{W}_i\boldsymbol{P}^{-1}.$$

This proves that Eq. (22) defines a gradient flow of Φ on $\mathcal{P}(r)$.

In the constrained case, if Eq. (23) defines a gradient flow, then the constrained gradient vector field grad$\Phi(\boldsymbol{P})$ of $\Phi : \mathcal{C} \to \mathbb{R}$ must satisfy the identity:

$$\boldsymbol{\xi}(\dot{\boldsymbol{P}}) = -\boldsymbol{\xi}(\mathrm{grad}\Phi(\boldsymbol{P})) = \boldsymbol{Q}\boldsymbol{\xi}(\boldsymbol{W}_p - \boldsymbol{P}^{-1}\boldsymbol{W}_i\boldsymbol{P}^{-1}). \quad (38)$$

On the other hand, grad$\Phi(\boldsymbol{P})$ is a gradient vector field if and only if, with a suitable Riemannian metric, the properties 1) and 2) are satisfied, i.e.:

1') grad$\Phi(\boldsymbol{P}) \in \mathrm{T}_{\boldsymbol{P}}(\mathcal{C})$ for all $\boldsymbol{P} \in \mathcal{C}$
2') $D_\Phi|_{\boldsymbol{P}}(\boldsymbol{X}) = \langle\langle \mathrm{grad}\Phi(\boldsymbol{P}), \boldsymbol{X}\rangle\rangle$ for all $\boldsymbol{X} \in \mathrm{T}_{\boldsymbol{P}}(\mathcal{C})$,

where $\mathrm{T}_{\boldsymbol{P}}(\mathcal{C})$ is the tangent space of $\mathcal{C}$ at $\boldsymbol{P}$, defined as:

$$\mathrm{T}_{\boldsymbol{P}}(\mathcal{C}) = \{\boldsymbol{X} \in \mathbb{C}^{r\times r} | \boldsymbol{A}\boldsymbol{\xi}(\boldsymbol{X}) = \boldsymbol{0}\}.$$

In order to prove that the above properties hold with grad$\Phi(\boldsymbol{P})$ in (38), it is useful to consider the expressions:

$$\boldsymbol{\xi}(\boldsymbol{P}) = \boldsymbol{H}\mathrm{vec}(\boldsymbol{P}) \quad (39)$$
$$\mathrm{vec}(\boldsymbol{P}) = \boldsymbol{K}\boldsymbol{\xi}(\boldsymbol{P}), \quad (40)$$

where $\boldsymbol{H}$ is a constant (sparse) $(nm+2N \times r^2)$ matrix, $\boldsymbol{K}$ is a constant (sparse) $(r^2 \times nm+2N)$ matrix, with $\boldsymbol{H}\boldsymbol{K} = \boldsymbol{I}_{nm+2N}$ and $\boldsymbol{K}\boldsymbol{H} = \boldsymbol{I}_{r^2}$, and the vec$(\cdot)$ operator stacks the columns of a matrix into a vector. Notice that, being $r = 2(n+N)$, $\boldsymbol{H}$ is a low rectangular matrix while $\boldsymbol{K}$ a high rectangular matrix.

Notice that, being $\boldsymbol{Q} = (\boldsymbol{I} - \boldsymbol{A}^\ddagger\boldsymbol{A})$, for all $\boldsymbol{X} \in \mathrm{T}_{\boldsymbol{P}}(\mathcal{C})$ it is $\boldsymbol{Q}\boldsymbol{\xi}(\boldsymbol{X}) = \boldsymbol{\xi}(\boldsymbol{X})$, or equivalently $\boldsymbol{K}\boldsymbol{Q}\boldsymbol{H}\mathbf{vec}(\boldsymbol{X}) = \mathbf{vec}(\boldsymbol{X})$. The weighted pseudo-inverse of the matrix $\boldsymbol{A}$ is chosen as

$$\boldsymbol{A}^\ddagger = \boldsymbol{M}\boldsymbol{A}^{\mathrm{T}}(\boldsymbol{A}\boldsymbol{M}\boldsymbol{A}^{\mathrm{T}})^{-1}, \quad (41)$$

where $\boldsymbol{M}$ is the following square diagonal $(nm+2N)$-matrix

$$\boldsymbol{M} = \boldsymbol{H}\boldsymbol{H}^{\mathrm{T}} = \begin{bmatrix} \boldsymbol{M}_G & \boldsymbol{O} \\ \boldsymbol{O} & \boldsymbol{I}_{2N} \end{bmatrix}, \quad (42)$$

being $\boldsymbol{M}_G$ a square diagonal (nm)-matrix. With this choice $\boldsymbol{K}\boldsymbol{Q}\boldsymbol{H}$ becomes Hermitian, i.e. $\boldsymbol{K}\boldsymbol{Q}\boldsymbol{H} = \boldsymbol{H}'\boldsymbol{Q}'\boldsymbol{K}'$.

Using the above definitions, it is straightforward to prove that property 1') is satisfied, because, in view of (38), it is

$$\boldsymbol{A}\boldsymbol{\xi}(\mathrm{grad}\Phi(\boldsymbol{P})) = \boldsymbol{0} \Rightarrow \mathrm{grad}\Phi(\boldsymbol{P}) \in \mathrm{T}_{\boldsymbol{P}}(\mathcal{C}) \quad \forall \boldsymbol{P} \in \mathcal{C}$$

Moreover, by using the Riemannian metric (37) on $\mathcal{C}$, the following equalities hold:

$$\begin{aligned}
&\langle\langle \mathrm{grad}\Phi(\boldsymbol{P}), \boldsymbol{X}\rangle\rangle = \mathrm{vec}(\boldsymbol{X})'\mathrm{vec}(\mathrm{grad}\Phi(\boldsymbol{P})) \\
&= \mathrm{vec}(\boldsymbol{X})'\boldsymbol{K}\boldsymbol{Q}\boldsymbol{\xi}(\boldsymbol{W}_p - \boldsymbol{P}^{-1}\boldsymbol{W}_i\boldsymbol{P}^{-1}) \\
&= \mathrm{vec}(\boldsymbol{X})'\boldsymbol{K}\boldsymbol{Q}\boldsymbol{H}\mathrm{vec}(\boldsymbol{W}_p - \boldsymbol{P}^{-1}\boldsymbol{W}_i\boldsymbol{P}^{-1}) \\
&= (\boldsymbol{K}\boldsymbol{Q}\boldsymbol{H}\mathrm{vec}(\boldsymbol{X}))'\mathrm{vec}(\boldsymbol{W}_p - \boldsymbol{P}^{-1}\boldsymbol{W}_i\boldsymbol{P}^{-1}) \\
&= \mathrm{vec}(\boldsymbol{W}_p - \boldsymbol{P}^{-1}\boldsymbol{W}_i\boldsymbol{P}^{-1})'\mathrm{vec}(\boldsymbol{X}) \\
&= D_\Phi|_{\boldsymbol{P}}(\boldsymbol{X}) \qquad \forall \boldsymbol{X} \in \mathrm{T}_{\boldsymbol{P}}(\mathcal{C})
\end{aligned}$$

showing that also property 2') is satisfied. Hence, Eq. (23) defines a constrained gradient flow on $\mathcal{C}$.

The next step consists in proving that the convergence rate of $\boldsymbol{P}$ with the constrained gradient flow (23) is the same as in the unconstrained case. To this purpose, notice that the eigenvalues of $\boldsymbol{Q}$ are either zero or one, being an orthogonal projector. Moreover, they are the solutions of the equation

$$\det(\lambda\boldsymbol{I} - \boldsymbol{Q}) = \det(\lambda^*\boldsymbol{I} - \boldsymbol{A}^\ddagger\boldsymbol{A}) = 0, \quad (43)$$

where $\lambda^* = 1 - \lambda$ and λ^* are the eigenvalues of $\boldsymbol{A}^\ddagger\boldsymbol{A}$. Since $\mathrm{rank}(\boldsymbol{A}^\ddagger\boldsymbol{A}) = 6 + 2N < nm + 2N$ already for $n > 2$, then $nm - 6$ eigenvalues are zero. Hence $nm - 6$ eigenvalues of $\boldsymbol{Q}$ are 1 and the remaining $6 + 2N$ are zero, i.e. $\lambda_{1,\ldots,nm-6} = 1$ and $\lambda_{nm-5,\ldots,nm+2N} = 0$. Hence, the singular value decomposition of $\boldsymbol{Q}$ is

$$\boldsymbol{Q} = \boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^{\mathrm{T}}, \quad (44)$$

where $\boldsymbol{\Sigma} = \text{diag}(1, \ldots, 1, 0, \ldots, 0)$, and $\boldsymbol{U}$ and $\boldsymbol{V}$ are unitary matrices. Multiplying both sides of (23) by $\boldsymbol{U}^{\mathrm{T}}$ yields:

$$\begin{aligned} &\boldsymbol{U}^{\mathrm{T}}\boldsymbol{\xi}(\dot{\boldsymbol{P}}) = \boldsymbol{\Sigma}\boldsymbol{V}^{\mathrm{T}}\boldsymbol{\xi}(\boldsymbol{W}_p - \boldsymbol{P}^{-1}\boldsymbol{W}_i\boldsymbol{P}^{-1}) \\ &= \begin{bmatrix} \boldsymbol{I}_{nm-6} & \boldsymbol{O} \\ \boldsymbol{O} & \boldsymbol{O} \end{bmatrix} \begin{bmatrix} \boldsymbol{V}_1 \\ \boldsymbol{V}_2 \end{bmatrix} \boldsymbol{\xi}(\boldsymbol{W}_p - \boldsymbol{P}^{-1}\boldsymbol{W}_i\boldsymbol{P}^{-1}), \end{aligned} \tag{45}$$

corresponding to a change of coordinates. The constrained elements of $\boldsymbol{\xi}(\boldsymbol{P})$ are those in the lower part of (45) and their derivatives are zero meaning that the $\boldsymbol{P}(t)$ remains on the constraint surface. Moreover, for the unconstrained elements, the same convergence rate of (22) is preserved.

The convergence of the gradient flow (28) can be proven in a similar way. In this case, the following weighted pseudo-inverse matrices must be chosen

$$\boldsymbol{G}^{\ddagger} = \boldsymbol{M}_G\boldsymbol{G}^{\mathrm{T}}(\boldsymbol{G}\boldsymbol{M}_G\boldsymbol{G}^{\mathrm{T}})^{-1} \tag{46}$$

$$\boldsymbol{A}_\tau^{\ddagger} = \boldsymbol{M}\boldsymbol{A}_\tau^{\mathrm{T}}(\boldsymbol{A}_\tau\boldsymbol{M}\boldsymbol{A}_\tau^{\mathrm{T}})^{-1}. \tag{47}$$

APPENDIX B
CONSTRAINED MONOTONE INCREASING FLOW (26)

Let $\boldsymbol{P}_0 = \{p_{i,j}\}$ denote an Hermitian matrix, $\lambda_r(\boldsymbol{P}_0)$ its minimum eigenvalue, and $\boldsymbol{v}_r = \{v_{r,i}\}$, with $i, j = 1, \ldots, r$ the corresponding eigenvector. The following identity holds

$$(\boldsymbol{P}_0 - \lambda_r\boldsymbol{I})\boldsymbol{v}_r = \boldsymbol{0}. \tag{48}$$

Differentiating (48) with respect to $p_{i,j}$ yields

$$\left(\frac{\partial \boldsymbol{P}_0}{\partial p_{i,j}} - \frac{\partial \lambda_r \boldsymbol{I}}{\partial p_{i,j}}\right)\boldsymbol{v}_r + (\boldsymbol{P}_0 - \lambda_r\boldsymbol{I})\frac{\partial \boldsymbol{v}_r}{\partial p_{i,j}} = \boldsymbol{0}. \tag{49}$$

Hence, by multiplying the above equation by $\boldsymbol{v}_r^{\mathrm{T}}$, we have

$$\boldsymbol{v}_r^{\mathrm{T}}\left(\frac{\partial \boldsymbol{P}_0}{\partial p_{i,j}} - \frac{\partial \lambda_r \boldsymbol{I}}{\partial p_{i,j}}\right)\boldsymbol{v}_r = 0, \tag{50}$$

that can be rewritten as

$$\frac{\partial \lambda_r}{\partial p_{i,j}} = \boldsymbol{v}_r^{\mathrm{T}}\frac{\partial \boldsymbol{P}_0}{\partial p_{i,j}}\boldsymbol{v}_r = v_{r,i}v_{r,j}, \tag{51}$$

being $\boldsymbol{v}_r^{\mathrm{T}}\boldsymbol{v}_r = 1$. Thus, the following identity holds

$$\frac{\partial \lambda_r}{\partial \boldsymbol{P}_0} = \boldsymbol{v}_r^{\mathrm{T}}\boldsymbol{v}_r. \tag{52}$$

On the other hand, the time derivative of λ_r can be expressed as

$$\dot{\lambda}_r = \text{tr}\left(\left(\frac{\partial \lambda_r}{\partial \boldsymbol{P}_0}\right)'\dot{\boldsymbol{P}}_0\right) = \text{vec}(\boldsymbol{v}_r\boldsymbol{v}_r^{\mathrm{T}})'\text{vec}(\dot{\boldsymbol{P}}_0). \tag{53}$$

Hence, the choice of the constrained gradient flow (26) yields

$$\begin{aligned} \dot{\lambda}_r &= \text{vec}(\boldsymbol{v}_r\boldsymbol{v}_r^{\mathrm{T}})'\boldsymbol{K}\boldsymbol{Q}\boldsymbol{\xi}(\boldsymbol{v}_r\boldsymbol{v}_r^{\mathrm{T}}) \\ &= \text{vec}(\boldsymbol{v}_r\boldsymbol{v}_r^{\mathrm{T}})'\boldsymbol{K}\boldsymbol{Q}\boldsymbol{H}\text{vec}(\boldsymbol{v}_r\boldsymbol{v}_r^{\mathrm{T}}) \geq 0, \end{aligned} \tag{54}$$

being $\boldsymbol{K}\boldsymbol{Q}\boldsymbol{H}$ Hermitian and positive-semidefinite.

REFERENCES

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

[2] J. Kerr and B. Roth, "Analisys of multifingered hands", *International Journal of Robotics Research*, vol. 4, pp. 3–17, 1986.

[3] F.T. Cheng and D.E. Orin, "Efficient algorithm for optimal force distribution–The compact-dual LP method", *IEEE Transactions on Robotics and Automation*, vol. 6, pp. 178–187, 1990.

[4] Y.H. Liu, "Qualitative test and force optimization of 3D frictional form-closure grasps using linear programming", *IEEE Transactions on Robotics and Automation*, vol. 15, pp. 163–173, 1999.

[5] Y. Zeng and W.-H. Qian, "A fast procedure for optimizing dynamic force distribution in multifingered grasping", *IEEE Transactions on Systems, Man, and Cybernetics, part B: Cybernetics*, vol. 36, no. 6, pp. 1417–1422, 2006.

[6] Y. Nakamura, k. Nagai and T. Yoshikawa, "Dynamics and stability in coordination of multiple robotic mechanisms", *International Journal of Robotics Research*, vol. 8, no. 2, pp. 45–61, 1989.

[7] L. Han, J. C. Trinkle and Z.X. Li, "Grasp analysis as linear matrix inequality problems", *IEEE Transactions on Robotics and Automation*, vol. 16, pp. 663–674, 2000.

[8] M. Buss, H. Hashimoto and J.B. Moore, "Dextrous hand grasping force optimization", *IEEE Transactions on Robotics and Automation*, vol. 12, pp. 406–418, 1996.

[9] M. Buss, L. Faybusovich and J.B. Moore, "Recursive algorithms for real-time grasping force optimization", *IEEE International Conference on Robotics and Automation*, Albuquerque, New Mexico, 1997.

[10] M. Buss, L. Faybusovich and J.B. Moore, "Dikin-type algorithms for dextrous grasping force optimization", *International Journal of Robotics Research*, vol. 17, pp. 831–839, 1998.

[11] Z.X. Li, Z. Quin, S. Jiang and L. Han, "Coordinated motion generation and real-time grasping force control for multifingered manipulation", *IEEE International Conference on Robotics and Automation*, Leuven, Belgium, 1998.

[12] T. Shlegl, M. Buss, T. Omata and G. Schmidt, "Fast dextrous regrasping with optimal contact force and contact sensor-based impedance control", *IEEE International Conference on Robotics and Automation*, Seoul, Korea, 2001.

[13] U. Helmke, K. Hper and J.B. Moore, "Quadratically convergent algorithms for optimal dexterous hand grasping", *IEEE Transactions on Robotics and Automation*, vol. 168, pp. 138–146, 2002.

[14] S.P. Boyd and B. Wegbreit, "Fast computation of optimal contact forces", *IEEE Transactions on Robotics*, vol. 23, pp. 1117–1132, 2007.

[15] Y. Zheng, C.M. Chew, and A.H. Adiwahono, "A GJK-based approach to contact force feasibility and distribution for multi-contact robots", *Robotics and Autonomous Systems*, vol. 59, pp. 194–207, 2011.

[16] G. Liu, J. Xu and Z. Li, "On geometric algorithms for real-time grasping force optimization", *IEEE Transactions on Control Systems Technology*, vol. 12, pp. 843–859, 2004.

[17] G. Liu and Z. Li, "Real-time grasping-force optimization for multi-fingered manipulation: Theory and experiments", *IEEE Transactions on Mechatronics*, vol. 9, pp. 65–77, 2004.

[18] W.S. Tang and J. Wang, "A Lagrangian network for multifingered hand grasping force optimization", *IEEE International Joint Conference on Neural Networks*, Honolulu, Hawaii, 2002.

[19] L.M. Fok and J. Wang, "Two recurrent neural networks for grasping force optimization of muulti-fingered robotic hands", *IEEE International Joint Conference on Neural Networks*, Onululu, Hawaii, 2002.

[20] C. Remond, V. Perdereau and M. Drouin, "A Multi-fingered hand control structure with on-line grasping force optimization", *IEEE/ASME International Conference on Advanced Intelligent Mechatronics*, Como, Italy, 2001.

[21] J.P. Saut, C. Remond, V. Perdereau and M. Drouin, "Online computation of grasping force in multi-fingered hands", *IEEE/RSJ International Conference on Intelligent Robots and Systems*, Edmonton, Alberta, Canada, 2005.

[22] J. Xu, Y. Lou and Z. Li, "Grasping force optimization for whole hand grasping", *IEEE/RSJ International Conference on Intelligent Robots and Systems*, Beijing, China, 2006.

[23] R. Michalec and A. Micaelli, "Optimal tightening for multi-fingered robust manipulation", *IEEE/RSJ International Conference on Intelligent Robots and Systems*, St. Louis, Missouri, 2009.

[24] H. Borgstrom, M.A. Batalin, G.S. Sukhatme and W.J. Kaiser, "Weighted barrier functions for computation of force distributions with friction cone constraints", *IEEE International Conference on Robotics and Automation*, Anchorage, Alaska, 2010.

[25] A. Bicchi, "On the closure properties of robotic grasping, *The International Journal of Robotics Research*, vol. 14, no. 4, pp. 319-334, 1995.
[26] A. Bicchi and D. Prattichizzo, "Analysis and optimization of tendinous actuation for biomorphically designed robotic systems, *Robotica*, vol. 18, pp. 23-31, 2000.
[27] M. Gabiccini, A. Bicchi, D. Prattichizzo, M. Malvezzi, "On the role of hand synergies in the optimal choice of grasping forces, *Autonomous Robots*, vol. 31, pp. 235–252, 2011.
[28] L.Y. Zheng, M.C. Lin and D. Manocha, "A fast n-dimensional ray-shooting algorithm for grasping force optimization", *IEEE International Conference on Robotics and Automation*, Anchorage, Alaska, 2010.
[29] V. Lippiello, B. Siciliano and L. Villani, "Online dextrous-hand grasping force optimization with dynamic torque constraints selection", *IEEE International Conference on Robotics and Automation*, Shanghai, China, 2011.
[30] V. Lippiello, B. Siciliano and L. Villani, "A grasping force optimization algorithm with dynamic torque constraints selection for multi-fingered robotic hands", *American Control Conference*, San Francisco, California, 2011.
[31] V. Lippiello, B. Siciliano and L. Villani, "A grasping force optimization algorithm for dexterous robotic hands", *IEEE International Conference on Robotics and Automation*, St. Paul, Minnesota, 2012.
[32] Y.F. Zheng and J.Y.S. Luh, "Optimal load distribution for two industrial robots handling a single object", *IEEE International Conference on Robotics and Automation*, Philadelphia, Pennsylvania, 1988.
[33] I.D. Walker, S.I. Marcus and R.A. Freeman, "Distribution of dynamic loads for multiple cooperating robot manipulators", *Journal of Robotic Systems*, vol. 9, pp. 35–47, 1989.
[34] M. Uchiyama, "A unified approach to load sharing, motion decomposing, and force sensing of dual arm robots", in *Robotics Research: The Fifth International Symposium* (H. Miura and S. Arimoto, eds.), pp. 225-232, MIT Press, Cambridge, MA, 1994.
[35] U. Helmke and J.B. Moore, *Optimization of Dynamic Systems*, Springer-Verlag, New York, 1993.
[36] L. Faybusovich, "Dikin's algorithm for matrix linear programming problems", in *System Modelling and Optimization* (J. Henry and J.-P. Yvon, eds.), pp. 237-247, Springer, New York, 1994.
[37] G. Palli, C. Melchiorri, G. Berselli and G. Vassura, "On the design of anthropomorphic dexterous robot hands: The UB hand evolution", *IEEE Robotics and Automation Magazine*, accepted, 2012.
[38] G. De Maria, C. Natale, and S. Pirozzi, "Force/tactile sensor for robotic applications", *Sensors and Actuators A: Physical*, pp. 60–72, 2012.

Vincenzo Lippiello was born in Naples, Italy, on June 19, 1975. He received the Laurea degree in electronic engineering and the Research Doctorate degree in information engineering from the University of Naples, Naples, Italy, in 2000 and 2004, respectively. He is an Assistant Professor of Automatic Control in the Department of Computer and Systems Engineering, University of Naples. His research interests include visual servoing of robot manipulators, hybrid visual/force control, adaptive control, grasping and manipulation. He has published more than 40 journal and conference papers and book chapter. He is chair of the IFAC Technical Committee on Robotics.

Bruno Siciliano (M'91–SM'94–F'00) was born in Naples, Italy, on October 27, 1959. He received the Laurea degree and the Research Doctorate degree in Electronic Engineering from the University of Naples in 1982 and 1987, respectively. He is Professor of Control and Robotics, and Director of the PRISMA Lab in the Department of Computer and Systems Engineering at University of Naples. He has co-authored 7 books, 70 journal papers, 180 conference papers and book chapters; his book Robotics: Modelling, Planning and Control is one of the most widely adopted textbooks world-wide. He has delivered 100 invited lectures and seminars at institutions worldwide.

He is a Fellow of IEEE, ASME and IFAC. He is Co-Editor of the Springer Tracts in Advanced Robotics series, and has served on the Editorial Boards of several journals as well as Chair or Co-Chair for numerous international conferences. He co-edited the Springer Handbook of Robotics, which received the PROSE Award for Excellence in Physical Sciences & Mathematics and was also the winner in the category Engineering & Technology.

He has been the coordinator of the large-scale integrating project DEXMART on dexterous and autonomous dual-arm/hand manipulation, funded by the European Commission in the 7th Framework Programme, and his group is currently involved in another five FP7 projects.

He has served the IEEE Robotics and Automation Society as Vice-President for Technical Activities and Vice-President for Publications, as a member of the AdCom, as a Distinguished Lecturer, and as the Society President.

Luigi Villani (S'94–M'97–SM'03) was born in Avellino, Italy, on December 5, 1966. He received the Laurea degree in electronic engineering and the Research Doctorate degree in electronic engineering and computer science from the University of Naples, Naples, Italy, in 1992 and 1996, respectively. He is an Associate Professor of Automatic Control in the Department of Computer and Systems Engineering, University of Naples. His research interests include force/motion control of manipulators, cooperative robot manipulation, lightweight flexible arms, adaptive and non-linear control of mechanical systems, visual servoing, fault diagnosis, and fault tolerance for dynamical systems. He has coauthored 5 books, 40 journal papers and 100 conference papers and book chapters.

Dr. Villani was Associate Editor of IEEE Transactions on Robotics from 2007 to 2011 and he is Associate Editor of IEEE Transactions on Control Systems Technology from 2005.