Exploiting Redundancy in Closed-Loop Inverse Kinematics for Dexterous Object Manipulation

Vincenzo Lippiello, Fabio Ruggiero, Luigi Villani

Abstract— **In this paper, a kinematic model for motion coordination of a redundant multi-fingered robotic hand is derived, which allows to compute the object pose from the joint variables of each finger as well as from a suitable set of contact variables. Then, a prioritized inverse kinematics scheme with redundancy resolution, both with inverse and transpose Jacobian matrix, is developed. This algorithm can be used for kinematic control as well as a local planning method for dexterous manipulation. A simulation case study is presented to demonstrate the effectiveness of the proposed approach.**

I. Introduction

Object manipulation with multi-fingered mechanical hands is a challenging task, especially in service robotics applications. In order to achieve the desired motion of the manipulated object, the fingers should operate in a coordinated fashion. In the absence of physical interaction between the fingers and the object, simple motion synchronization shall be ensured. On the other hand, the execution of object grasping or manipulation requires controlling also the interaction forces to ensure grasp stability [1], [2].

From a purely kinematics point of view, an object manipulation task can be assigned in terms of the motion of the fingertips and/or in terms of the desired motion of the manipulated object. The work of a planner (or a control module) is to map the desired task into the corresponding joint trajectories for the fingers, and always requires the solution of an inverse kinematics problem.

In this paper, a kinematic model for object manipulation using a multi-fingered robotic hand is derived, which allows the object pose to be computed from the joint variables of each finger (active joints), as well as from a set of contact variables, modeled as passive joints [3]. Suitable conditions are derived ensuring that a given motion can be imposed to the object using only the active joints. A closed-loop inverse kinematics algorithm (CLIK) has been rearranged in this contest to compute fingers and contact variables, given the desired object trajectories. Both the schemes based on the transpose and on the inverse of the Jacobian matrix can be adopted [4].

Notice that the manipulation system can be redundant also if the single fingers are not: this is due to the presence of the additional degrees of freedom (DOFs) provided by the contact variables. These redundant DOFs are suitably exploited to satisfy a certain number of secondary tasks, aimed at ensuring grasp stability and manipulation dexterity, besides the main task corresponding to the desired object motion. Conflict between different tasks is avoided by adopting a suitable task priority strategy [5]. The resulting prioritized CLIK can be used for kinematic control as well as a local planning method for object dexterous manipulation. To our knowledge, the focus of previous papers on kinematics of multi-fingered manipulation was on constrained kinematic control [3], [6], or manipulability analysis [7], without considering redundancy resolution. A case study, representing a simplified bimanual manipulation task is presented to show the effectiveness of the proposed approach.

II. Kinematic model

A. Kinematics of object and fingers

Let us consider a robotic hand composed by N rigid fingers, numbered from 1 to N, holding a rigid object, and let $\boldsymbol{q}_i$ denote the joint vector of finger i, with n_i components. To derive the kinematic mapping between the joint variables of the fingers and the pose (position and orientation) of the object, it is useful introducing an object frame Σ_o attached to the object, usually chosen with the origin in the object center of mass. The pose of Σ_o with respect to a base frame Σ_b fixed to the hand (also known as hand frame) can be represented by the (4×4) homogeneous transformation matrix

$$\boldsymbol{T}_o = \begin{bmatrix} \boldsymbol{R}_o & \boldsymbol{o}_o \\ \boldsymbol{0}^{\mathrm{T}} & 1 \end{bmatrix},$$

where $\boldsymbol{R}_o$ is the (3×3) rotation matrix, $\boldsymbol{o}_o$ is the (3×1) position vector of the origin of Σ_o with respect to the base frame, while $\boldsymbol{0}$ denotes the (3×1) null vector. The velocity of Σ_o with respect to the base frame can be represented by the (6×1) twist vector $\boldsymbol{v}_o^{\mathrm{T}} = \begin{bmatrix} \dot{\boldsymbol{o}}_o^{\mathrm{T}} & \boldsymbol{\omega}_o^{\mathrm{T}} \end{bmatrix}^{\mathrm{T}}$, where $\boldsymbol{\omega}_o$ is the angular velocity such that $\dot{\boldsymbol{R}}_o = \boldsymbol{S}(\boldsymbol{\omega}_o)\boldsymbol{R}_o$, with $\boldsymbol{S}(\cdot)$ the skew-symmetric operator representing the vector product.

Assuming that each finger has only one contact point with the object, it is useful introducing a frame Σ_{f_i} attached to the i-th fingertip ($i = 1 \ldots N$) and with the origin $\boldsymbol{o}_{f_i}$ at the point of contact. The pose of Σ_{f_i} with respect to the base frame can be computed on the basis of the finger kinematics as $\boldsymbol{T}_{f_i} = \boldsymbol{T}_{f_i}(\boldsymbol{q}_i)$, while the velocity can be expressed as

$$\boldsymbol{v}_{f_i} = \begin{bmatrix} \dot{\boldsymbol{o}}_{f_i} \\ \boldsymbol{\omega}_{f_i} \end{bmatrix} = \begin{bmatrix} \boldsymbol{J}_{P_i} \\ \boldsymbol{J}_{O_i} \end{bmatrix} \dot{\boldsymbol{q}}_i = \boldsymbol{J}_i(\boldsymbol{q}_i)\dot{\boldsymbol{q}}_i, \tag{1}$$

The research leading to these results has been supported by the DEXMART Large-scale integrating project, which has received funding from the European Community's Seventh Framework Programme (FP7/2007-2013) under grant agreement ICT-216239. 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 {vincenzo.lippiello, fabio.ruggiero, luigi.villani}@unina.it

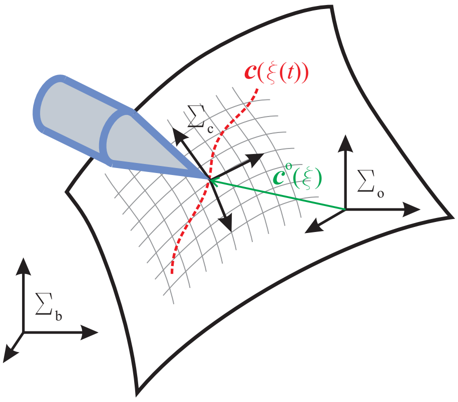

Fig. 1. Local parametrization of the object surface with respect to the object frame.

where $\boldsymbol{J}_i$ is the $(6 \times n_i)$ Jacobian of the finger, while $\boldsymbol{J}_{P_i}$ and $\boldsymbol{J}_{O_i}$ are $(3 \times n_i)$ matrices known as position Jacobian and orientation Jacobian respectively.

B. Contact kinematics

Grasping situations may involve moving rather than fixed contacts: often, both the object and the robotic fingers are smooth surfaces, and manipulation involves rolling and/or sliding of the fingertips on the object surface, depending on the contact type. If the fingers and object shapes are completely known, the contact kinematics can be described introducing contact coordinates defined on the basis of a suitable parametrization of the contact surfaces [8], [9].

To gain insight into the kinematics of contact, in this paper it is assumed that the fingertips are sharp so that the contact point $\boldsymbol{o}_{f_i}$ of each finger is fixed and coincides with (or can be approximated by) the fingertip position, while the object is a smooth surface (see, e.g., [10]). Let Σ_{c_i} be the contact frame attached to the object and with the origin at the contact point $\boldsymbol{o}_{c_i}$. Notice that, instantaneously, the object contact point $\boldsymbol{o}_{c_i}$ and the finger contact point $\boldsymbol{o}_{f_i}$ are coincident. One of the axes of Σ_{c_i}, e.g., the Z axis, is assumed to be normal to the tangent plane to the object surface at the contact point, and pointing outward the object surface.

Assume that, at least locally, the position of the contact point with respect to the object frame $\boldsymbol{o}^o_{o,c_i} = \boldsymbol{o}^o_{c_i} - \boldsymbol{o}^o_o$ can be parameterized in terms of a coordinate chart $\boldsymbol{c}^o_i : U_i \subset \mathbb{R}^2 \mapsto \mathbb{R}^3$ which maps a point $\boldsymbol{\xi}_i = [u_i \quad v_i]^{\mathrm{T}} \in U_i$ in the point $\boldsymbol{o}^o_{o,c_i}(\boldsymbol{\xi}_i)$ of the surface of the object.

To simplify notation, for the reminder of this subsection, index i will be dropped.

In the hypothesis that $\boldsymbol{c}^o$ is a diffeomorphism and that the coordinate chart is orthogonal and right-handed, contact frame Σ_c can be chosen as a Gauss frame [8], with the rotation matrix $\boldsymbol{R}^o_c$ computed as

$$\boldsymbol{R}^o_c(\boldsymbol{\xi}) = \begin{bmatrix} \dfrac{\boldsymbol{c}^o_u}{\|\boldsymbol{c}^o_u\|} & \dfrac{\boldsymbol{c}^o_v}{\|\boldsymbol{c}^o_v\|} & \dfrac{\boldsymbol{c}^o_u \times \boldsymbol{c}^o_v}{\|\boldsymbol{c}^o_u \times \boldsymbol{c}^o_v\|} \end{bmatrix},$$

where tangent vectors $\boldsymbol{c}^o_u = \partial \boldsymbol{c}^o / \partial u$ and $\boldsymbol{c}^o_v = \partial \boldsymbol{c}^o / \partial v$ are orthogonal.

Consider the contact kinematics from the object point of view. Let $\boldsymbol{c}^o(\boldsymbol{\xi}(t))$ denote a curve on the surface of the object, with $\boldsymbol{\xi}(t) \in U$ (see Fig. 1). The corresponding motion of the contact frame with respect to the base frame can be determined as a function of: object motion, geometric parameters of the object and "velocity" of the curve. Namely, computing the time derivative of equation $\boldsymbol{o}_c = \boldsymbol{o}_o + \boldsymbol{R}_o \boldsymbol{c}^o(\boldsymbol{\xi})$, which denotes the position of the object contact point in the base frame, yields

$$\dot{\boldsymbol{o}}_c = \dot{\boldsymbol{o}}_o + \boldsymbol{S}(\boldsymbol{c}^o(\boldsymbol{\xi}))\boldsymbol{\omega}_o + \boldsymbol{R}_o \frac{\partial \boldsymbol{c}^o}{\partial \boldsymbol{\xi}} \dot{\boldsymbol{\xi}}, \tag{2}$$

where the first two terms on the right-hand side specify the velocity contribution due to the object motion, while the last term represents the velocity of the motion on the object surface relative to the object frame. On the other hand, for the angular velocity, the following equality holds

$$\boldsymbol{\omega}_c = \boldsymbol{\omega}_o + \boldsymbol{R}_o \boldsymbol{\omega}^o_{o,c}, \tag{3}$$

being $\boldsymbol{\omega}^o_{o,c}$ the angular velocity of the motion of the contact frame relative to the object frame, which can be expressed in the form

$$\boldsymbol{\omega}^o_{o,c} = \boldsymbol{C}(\boldsymbol{\xi})\dot{\boldsymbol{\xi}}, \tag{4}$$

with $\boldsymbol{C}(\boldsymbol{\xi})$ a (3×2) matrix depending on geometric parameters of the surface [9]. Matrix $\boldsymbol{C}$ is not necessarily full rank; for example, in the case of planar surfaces, this matrix is null.

In view of (2), (3), (4), the velocity of the contact frame can be expressed as

$$\boldsymbol{v}_c = \begin{bmatrix} \dot{\boldsymbol{o}}_c \\ \boldsymbol{\omega}_c \end{bmatrix} = \boldsymbol{G}^{\mathrm{T}}(\boldsymbol{\xi})\boldsymbol{v}_o + \boldsymbol{J}_\xi(\boldsymbol{\xi})\dot{\boldsymbol{\xi}}, \tag{5}$$

where

$$\boldsymbol{G}(\boldsymbol{\xi}) = \begin{bmatrix} \boldsymbol{I} & \boldsymbol{0} \\ \boldsymbol{S}(\boldsymbol{c}(\boldsymbol{\xi})) & \boldsymbol{I} \end{bmatrix}, \qquad \boldsymbol{J}_\xi(\boldsymbol{\xi}) = \begin{bmatrix} \boldsymbol{R}_o \dfrac{\partial \boldsymbol{c}^o}{\partial \boldsymbol{\xi}} \\ \boldsymbol{R}_o \boldsymbol{C}(\boldsymbol{\xi}) \end{bmatrix}$$

are respectively (6×6) and (6×2) full rank matrices.

Consider now the contact kinematics from the finger point of view. The contact can be modeled as an unactuated 3-DOF ball and socket kinematic pair centered at the contact point, possibly moving on the surface if sliding is allowed. Therefore, the orientation of contact frame Σ_c with respect to finger frame Σ_f can be computed in terms of a suitable parametrization of the ball and socked joint, e.g., Euler angles, axis-angle or unit quaternion. For the purpose of this work, a vector $\boldsymbol{\theta} = [\theta_1 \quad \theta_2 \quad \theta_3]^{\mathrm{T}}$ of XYZ Euler angles is considered, thus $\boldsymbol{R}^f_c = \boldsymbol{R}^f_c(\boldsymbol{\theta})$. In detail, θ_1 and θ_2 parameterize the so-called "swing" motion aligning axis Z of a moving frame to axis Z of the contact frame, while θ_3 corresponds to the "twist" motion about axis Z of the contact frame. Singularities occurs for $\theta_2 = \pm\pi/2$, but they do not correspond to physical singularities of the kinematics pair. Therefore, the angular velocity of Σ_c relative to Σ_f can be expressed as

$$\boldsymbol{\omega}^f_{f,c} = \boldsymbol{T}(\boldsymbol{\theta})\dot{\boldsymbol{\theta}},$$

with $\boldsymbol{T}$ a suitable transformation matrix. In view of the decomposition $\boldsymbol{\omega}_c = \boldsymbol{\omega}_f + \boldsymbol{R}_f(\boldsymbol{q})\boldsymbol{\omega}_{f,c}^{f}$, and from (1), the angular velocity of Σ_c can be computed also as a function of finger and contact variables in the form

$$\boldsymbol{\omega}_c = \boldsymbol{J}_O(\boldsymbol{q})\dot{\boldsymbol{q}} + \boldsymbol{R}_f(\boldsymbol{q})\boldsymbol{T}(\boldsymbol{\theta})\dot{\boldsymbol{\theta}}. \tag{6}$$

Moreover, since the origins of Σ_c and Σ_f coincide, the following equality holds

$$\dot{\boldsymbol{o}}_c = \dot{\boldsymbol{o}}_f = \boldsymbol{J}_P(\boldsymbol{q})\dot{\boldsymbol{q}}. \tag{7}$$

Using (6) and (7), the velocity of the contact frame can be expressed as

$$\boldsymbol{v}_c = \boldsymbol{J}(\boldsymbol{q})\dot{\boldsymbol{q}} + \boldsymbol{J}_\theta(\boldsymbol{\theta},\boldsymbol{q})\dot{\boldsymbol{\theta}}, \tag{8}$$

where $\boldsymbol{J}$ is the finger Jacobian and

$$\boldsymbol{J}_\theta = \begin{bmatrix} \boldsymbol{0} \\ \boldsymbol{R}_f(\boldsymbol{q})\boldsymbol{T}(\boldsymbol{\theta}) \end{bmatrix}$$

is a (6×3) full rank matrix (far from representation singularities).

Hence, from (5) and (8), the contact kinematics of finger i has the form

$$\boldsymbol{J}_i(\boldsymbol{q}_i)\dot{\boldsymbol{q}}_i + \boldsymbol{J}_{\eta i}(\boldsymbol{\eta}_i,\boldsymbol{q}_i)\dot{\boldsymbol{\eta}}_i = \boldsymbol{G}_i^{\mathrm{T}}(\boldsymbol{\eta}_i)\boldsymbol{v}_o, \tag{9}$$

where $\boldsymbol{\eta}_i = \begin{bmatrix}\boldsymbol{\xi}_i^{\mathrm{T}} & \boldsymbol{\theta}_i^{\mathrm{T}}\end{bmatrix}^{\mathrm{T}}$ is the vector of contact variables and $\boldsymbol{J}_{\eta i} = \begin{bmatrix}-\boldsymbol{J}_{\xi i} & \boldsymbol{J}_{\theta i}\end{bmatrix}$ is a (6×5) full rank matrix. This equation can be interpreted as the differential kinematics equation of an "extended" finger corresponding to the kinematic chain including the finger joint variables (active joints) and the contact variables (passive joints), from the base frame to the contact frame [3].

It is worth pointing out that equation (9) involves all the 6 components of the velocity, differently from the grasping constraint equation usually considered (see, e.g., [9]), which contains only the components of the velocities that are transmitted by the contact. The reason is that the above formulation takes into account also the velocity components not transmitted by contact i, parameterized by the contact variables and lying in the range space of $\boldsymbol{J}_{\eta_i}$. As a consequence, $\boldsymbol{G}_i$ is always a full rank matrix.

Planar motions can be analyzed as a particular case of the general 6-DOF motion by rewriting equation (9) in terms of 3 components.

Depending on the type of contact considered, some of the parameters of $\boldsymbol{\xi}_i$ and $\boldsymbol{\theta}_i$ are constant. For the three models usually considered for grasp analysis [2], it is:

- *point contact without friction*: all the parameters may vary (i.e., the finger may slide on the object surface and the orientation of Σ_{f_i} with respect to Σ_{c_i} may vary);
- *hard finger with friction*: vector $\boldsymbol{\xi}_i$ is constant, while vector $\boldsymbol{\theta}_i$ may vary (i.e., the finger is not allowed to slide on the object surface, but the orientation of Σ_{f_i} with respect to Σ_{c_i} may vary);
- *soft finger with friction*: vector $\boldsymbol{\xi}_i$ is constant, as well as the last parameter of vector $\boldsymbol{\theta}_i$, corresponding to the rotation about the Z axis of Σ_{c_i} (i.e., the finger is not allowed to slide and to twist about the normal to the object surface).

Hence, assuming that the type of contact remains unchanged during task execution, the variable parameters at each contact point are grouped in a $(n_{c_i} \times 1)$ vector $\boldsymbol{\eta}_i$ of contact variables, with $n_{c_i} \leq 5$.

C. Kinematic classification of grasp

On the basis of (9), it is possible to make a kinematic classification of the grasp [2].

A grasp is *redundant* if the null space of the matrix

$$\tilde{\boldsymbol{J}}_i = \begin{bmatrix}\boldsymbol{J}_i & \boldsymbol{J}_{\eta_i}\end{bmatrix}$$

is non-null, for at least one finger i. In this case, the mapping between the joint variables of "extended" finger i and the object velocity is many to one: motions of active and passive joints of the extended finger are possible when the object is locked. Notice that a single finger could be redundant if the null space of $\boldsymbol{J}_i$ is non null, i.e., in the case of a kinematically redundant finger; in this case, motion of the active joints are possible when both the passive joints and the object are locked. On the other hand, for the type of contacts considered here (point contact), the null space of $\boldsymbol{J}_{\eta_i}$ is always null: this implies that motions of the passive joints are not possible when the active joints and the object are locked. In typical situations, the fingers of the robotic hand are not redundant, but the extended fingers may be redundant thanks to the presence of the additional DOFs provided by the passive joints.

A grasp is *indeterminate* if the intersection of the null spaces of $[-\boldsymbol{J}_{\eta_i} \quad \boldsymbol{G}_i^{\mathrm{T}}]$, for all $i = 1, \ldots, N$, is non-null. In this case, motions of the object and of the passive joints are possible when the active joints of all the fingers are locked. The kinematic indetermination derives from the fact that the motion of the object cannot be completely controlled by finger motions, but depends on the dynamics of the system [9]. An example of indeterminate grasp may be that of a box grasped by two hard-finger opposite contacts: in this case, the box may rotate about the axis connecting the two contact points while the fingers are locked.

It is worth pointing out that, also in the case of redundant and indeterminate grasps, for a given object pose and fingers configuration, the value of the contact variables is uniquely determined.

III. Inverse kinematics with redundancy resolution

In the case of kinematically determinate and, possibly, redundant grasp, a suitable inverse kinematics algorithm can be adopted to compute the fingers and contact variables corresponding to a desired object motion.

In detail, in view of (9), the CLIK algorithm with redundancy resolution, based on the preudo-inverse of the Jacobian matrix, is given by equation:

$$\begin{bmatrix}\dot{\boldsymbol{q}}_i \\ \dot{\boldsymbol{\eta}}_i\end{bmatrix} = \tilde{\boldsymbol{J}}_i^{\dagger}(\boldsymbol{q}_i,\boldsymbol{\eta}_i)\boldsymbol{G}_i^{\mathrm{T}}(\boldsymbol{\eta}_i)(\boldsymbol{v}_d + \boldsymbol{K}_i\boldsymbol{e}_{o_i}) + \boldsymbol{N}_{o_i}(\boldsymbol{q}_i,\boldsymbol{\eta}_i)\boldsymbol{\sigma}_i, \tag{10}$$

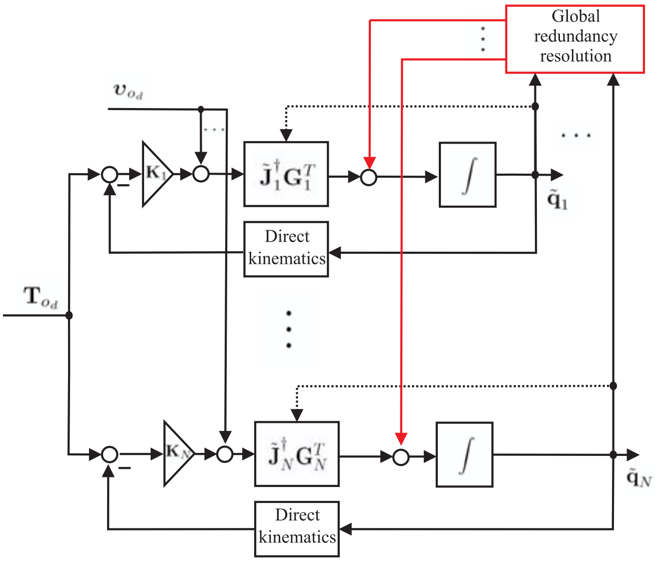

Fig. 2. Block scheme of the CLIK algorithm with redundancy resolution based on equation (10).

where $\tilde{\boldsymbol{J}}_i^{\dagger}$ is a right (weighted) pseudo-inverse of $\tilde{\boldsymbol{J}}_i$, $\boldsymbol{e}_{o_i}$ is a pose error between the desired and the current object pose, $\boldsymbol{K}_i$ is a (6×6) symmetric and positive definite matrix, $\boldsymbol{\sigma}_i$ is a suitable velocity vector corresponding to a secondary task, and

$$\boldsymbol{N}_{o_i} = \boldsymbol{I} - \tilde{\boldsymbol{J}}_i^{\dagger}\tilde{\boldsymbol{J}}_i \tag{11}$$

is a projector in the null space of the Jacobian matrix. The asymptotic stability of the equilibrium $\boldsymbol{e}_{o_i} = \boldsymbol{0}$ for system (10) can be easily proven.

The computation of the Jacobian pseudo-inverse can be avoided by adopting an alternative CLIK algorithm based on the transpose of the Jacobian matrix, given by equation:

$$\begin{bmatrix} \dot{\boldsymbol{q}}_i \\ \dot{\boldsymbol{\eta}}_i \end{bmatrix} = \tilde{\boldsymbol{J}}_i^{\mathrm{T}}(\boldsymbol{q}_i, \boldsymbol{\eta}_i)\boldsymbol{G}_i^{\mathrm{T}}(\boldsymbol{\eta}_i)\boldsymbol{K}_i\boldsymbol{e}_{o_i} + \boldsymbol{N}_{o_i}(\boldsymbol{q}_i, \boldsymbol{\eta}_i)\boldsymbol{\sigma}_i. \tag{12}$$

The asymptotic stability of the equilibrium $\boldsymbol{e}_{o_i} = \boldsymbol{0}$ for system (12) can be easily proven in the case $\boldsymbol{v}_d = \boldsymbol{0}$. Notice that, if $\boldsymbol{N}_{oi}$ is computed according to (11), the computation of the Jacobian pseudo-inverse is not avoided with algorithm (12).

In principle, N independent CLIK algorithms can be used, one for each finger, all with the same input, namely, the desired object pose $\boldsymbol{T}_d$ and velocity $\boldsymbol{v}_d$. However, some secondary tasks may involve all the variables of the system at the same time, e.g., those related to grasp quality.

Hence, the complete CLIK scheme with redundancy resolution includes N decentralized feedback loops, one for each finger, and a centralized feedforward action depending on the whole system configuration. This is shown in the block diagram of Fig. 2 for the scheme based on the Jacobian pseudo-inverse, where $\tilde{\boldsymbol{q}}_i = \begin{bmatrix} \boldsymbol{q}_i^{\mathrm{T}} & \boldsymbol{\eta}_i^{\mathrm{T}} \end{bmatrix}^{\mathrm{T}}$.

Since the system may be highly redundant, multiple tasks could be fulfilled, provided that they are suitably arranged in a priority order [5]. For example, assume that two secondary tasks, involving all the variables of the system, are assigned in the form: $\boldsymbol{\sigma}_a = \boldsymbol{f}_a(\boldsymbol{q}, \boldsymbol{\eta})$, $\boldsymbol{\sigma}_b = \boldsymbol{f}_b(\boldsymbol{q}, \boldsymbol{\eta})$, where $\boldsymbol{q} = [\boldsymbol{q}_1^{\mathrm{T}} \ \ \dots \ \ \boldsymbol{q}_N^{\mathrm{T}}]^{\mathrm{T}}$ and $\boldsymbol{\eta} = [\boldsymbol{\eta}_1^{\mathrm{T}} \ \ \dots \ \ \boldsymbol{\eta}_N^{\mathrm{T}}]^{\mathrm{T}}$ are the stacked vector of joint and contact variables. Adopting the *augmented projection method* [5], equation (10) must replaced by

$$\begin{bmatrix} \dot{\boldsymbol{q}}_i \\ \dot{\boldsymbol{\eta}}_i \end{bmatrix} = \tilde{\boldsymbol{J}}^{\dagger}\boldsymbol{G}_i^{\mathrm{T}}(\boldsymbol{v}_o + \boldsymbol{K}\boldsymbol{e}_{o_i}) + \boldsymbol{N}_{o_i}\boldsymbol{J}_a^{\dagger}\boldsymbol{K}_a\boldsymbol{e}_a + \boldsymbol{N}_{i_{ab}}\boldsymbol{J}_b^{\dagger}\boldsymbol{K}_b\boldsymbol{e}_b, \tag{13}$$

where $\boldsymbol{J}_a$ and $\boldsymbol{J}_b$ are the Jacobian matrices of the secondary tasks, $\boldsymbol{e}_a = \boldsymbol{\sigma}_{a_d} - \boldsymbol{f}_a(\boldsymbol{q})$ and $\boldsymbol{e}_b = \boldsymbol{\sigma}_{b_d} - \boldsymbol{f}_b(\boldsymbol{q})$, being $\boldsymbol{\sigma}_{a_d}$ and $\boldsymbol{\sigma}_{b_d}$ the desired values of the secondary tasks, $\boldsymbol{N}_a$ is the projector in the null space of $\boldsymbol{J}_a$, and $\boldsymbol{N}_{i_{ab}}$ is the projector in the null space of matrix

$$\boldsymbol{J}_{i_{ab}}(\boldsymbol{q}, \boldsymbol{\eta}) = \begin{bmatrix} \tilde{\boldsymbol{J}}^{\mathrm{T}}(\boldsymbol{q}_i, \boldsymbol{\eta}_i) & \boldsymbol{J}_a^{\mathrm{T}}(\boldsymbol{q}, \boldsymbol{\eta}) & \boldsymbol{J}_b^{\mathrm{T}}(\boldsymbol{q}, \boldsymbol{\eta}) \end{bmatrix}^{\mathrm{T}}.$$

A similar scheme can be adopted for the transpose-based CLIK algorithm in the case of multiple secondary tasks.

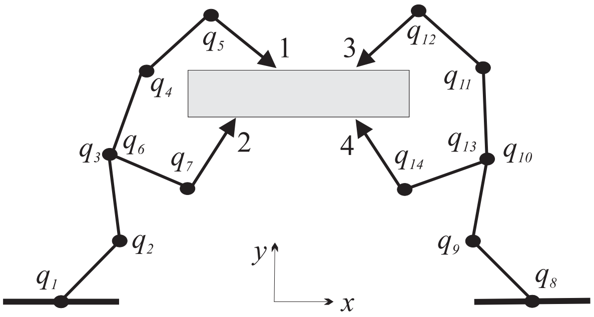

Fig. 3. Manipulation system.

IV. Case study

The CLIK scheme with Jacobian pseudo-inverse has been tested on a manupulation system, represented in Fig. 3, composed by two identical planar grippers, each with two branches and 7 DOFs, grasping a rectilinear bar. For simplicity, all the links have the same length $l = 1$ m. The idea is that of performing a simplified bimanual manipulation task.

It is assumed that, in the initial configuration (reproduced in Fig. 3), the system grasps the object with tips 3 and 4 aligned to y axis, and with tips 1 and 3, as well as tips 2 and 4, aligned to x axis. The distance between tips 1 and 3 is 0.7 m, while the distance between points 2 and 4 is 1.2 m.

The contact at point 1 and 3 is assumed of type "point contact without friction", while the contact at points 2 and 4 is of type "hard finger with friction", i.e., the object surface is smooth on the top side and rough on the bottom side. This imply that two contact variables θ_i and ξ_i are required to represent rotation and sliding of finger i $(i = 1, 2)$ on the object surface, while two contact variables θ_2 and θ_4 are to be introduced to represent the rotation of finger i $(i = 2, 4)$ with respect to the object surface. It is easy to verify that this grasp is force closure [9] and kinematically determinate [3].

The manipulation system has a total of 20 DOFs that are not all independent, for the presence of 9 closed-chain kinematic constraints; the resulting 11 DOFs can be exploited to satisfy a certain number of tasks.

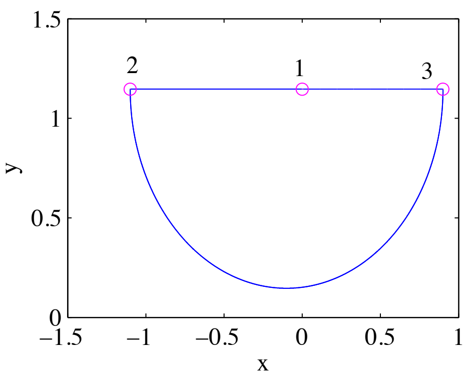

Fig. 4. Path imposed to object position.

The main task consists in a desired trajectory for the object position, and a desired constant orientation with the bar aligned to x axis. The position path, represented in Figure 4, can be decomposed in the sequence: line segment 1-2, arc segment 2-3, line segment 3-2, arc segment 2-3. The time law for each segment is a fifth-order polynomial with null first and second derivative at initial and final time, of a 10 s duration. A 1 s pause is present before the execution of each segment.

Three simple secondary tasks, with decreasing priorities, are considered, according to the augmented projection method. Namely:

a. *joint limits*: a constraint $q_3 \leq \pi/2$ is imposed to joint variable q_3;

b. *collision avoidance*: the distance between fingers 1 and 3, which can slide on the surface, must be greater than a threshold $d = 0.3$ m;

c. *grasp quality*: the contact points of fingers 1 and 3 are to be kept as close as possible to the middle point between the contact points of fingers 2 and 4.

Notice that these tasks do not saturate all the available DOFs of the system. Additional tasks could be imposed, but are not considered here for brevity.

The secondary tasks are not all active at the same time, but they start on the basis of a threshold mechanism. Namely, subtask **a** is active only when q_3 is in the neighbour of $\pi/2$, while subtask **b**. is active when the distance between tips 1 and 3 is lower than 0.4 m. Moreover, the gains of the subtasks are not constant; in detail, the gain of subtask **a** depends on the value of q_3, the gain of subtask **b** depends on the norm of the distance between fingertip 1 and 3, and the gain of subtask **c** depends on the distance of the fingertip 1 and 3 from the center of the tips 2 and 4. As an example, the gain of subtask **b**, as a function of the norm of the distance between fingertips 1 and 3, is reported in Fig. 5.

Two different simulations have been made, with and without the presence of secondary tasks. The gain matrix for the main task in (13) is chosen as $\boldsymbol{K} = 100\boldsymbol{I}$.

The results of Fig. 6 show the norm of the object pose error with and without secondary tasks. It can be observed that the error in the two cases is the same, because the secondary tasks are in the null space of the main task.

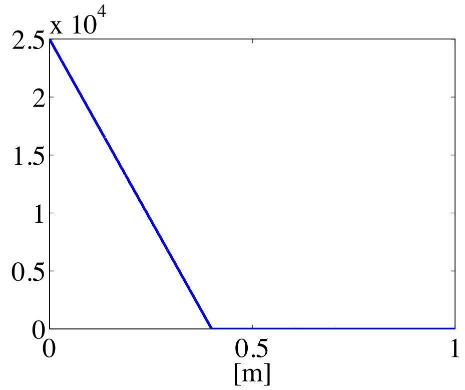

Fig. 5. Gain of subtask **b** as a function of the norm of the distance between fingertips 1 and 3.

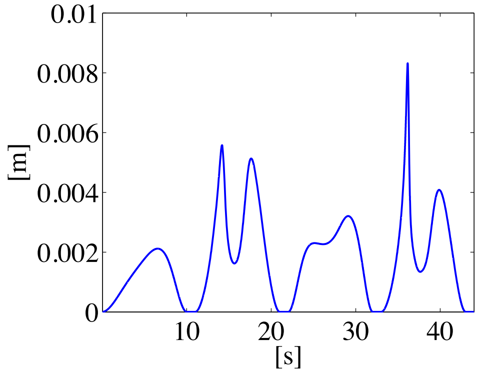

Fig. 6. Time history of the object pose error in terms of the average of the norm of the pose errors of the 4 parallel CLIK algortithms. Dashed line: without secondary tasks. Continuous line: with secondary tasks.

The time histories of the significant variables for the secondary tasks are reported in Figs. 7–9.

Fig. 7 shows the time history of the joint 3 variable, assuming that, at time $t = 0$, this variable is close to the upper joint limit ($\pi/2$). It can be verified that, without secondary tasks, the joint variable violates the limit, differently from the case when the joint limit constraint is imposed as secondary task.

Fig. 8 reports the time history of the distance between tips 1 and 3. It can be seen that, without secondary tasks, the distance between contact points 1 and 3 changes sign, meaning that fingers 1 and 3 overlaps. On the other hand, using redundancy, the distance between finger 1 and 3 remains always positive, as requested by the collision avoidance task between tips 1 and 3.

Fig. 9 reports the time history of the x position of all the contact points with respect to the center of the object. The constant lines are the position of fixed contact points 2 and 4. It can be seen that, without secondary tasks, the sliding contact points 1 and 3 have large displacements on the object surface. On the other hand, using redundancy, they remain close to each other as imposed by the secondary task **c**, without violating task **b**.

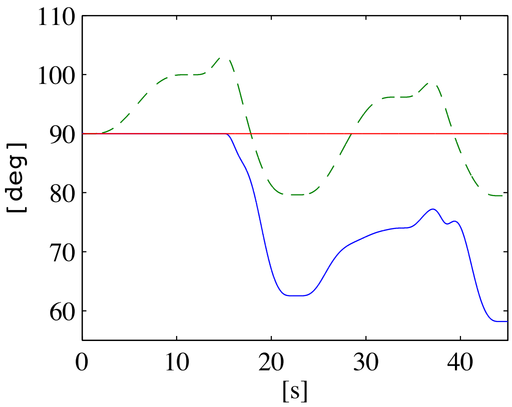

Fig. 7. Time history of joint 3 variable. Dashed line: without secondary tasks. Continuous line: with secondary tasks.

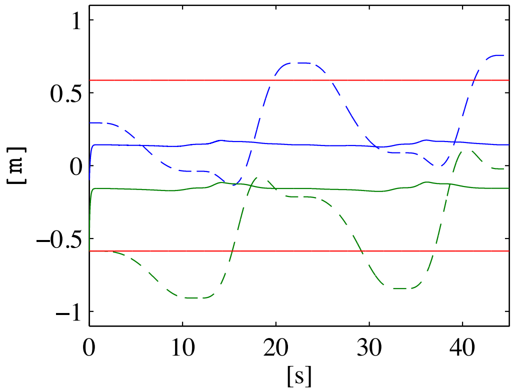

Fig. 9. Time history of the x position of the contact points with respect to the center of the object. Dashed line: without secondary tasks. Continuous line: with secondary tasks.

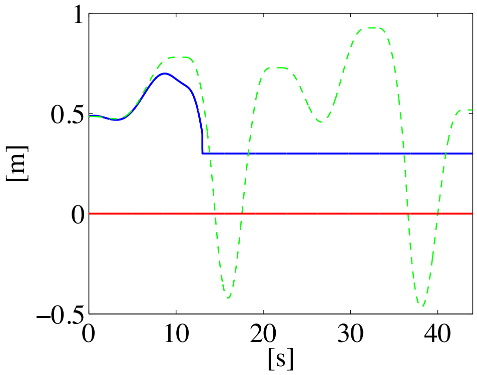

Fig. 8. Time history of the distance between tips 1 and 3. Dashed line: without secondary tasks. Continuous line: with secondary tasks.

V. Conclusion

In this paper, the problem of inverse kinematics for a multi-fingered hand was considered. In particular, a kinematic model has been introduced which allows the object pose to be computed from the joint variables of each finger as well as from a suitable set of contact variables. Then, a prioritized close loop inverse kinematics scheme with redundancy resolution has been proposed, both with transpose and inverse Jacobian matrix, for the case of kinematically determinate and possibly redundant grasp. This scheme allows computing joint and contact variables for a multi-fingered manipulation task assigned in terms of the sole object desired position and orientation, while preserving grasp stability and manipulation dexterity. Therefore, it can be used also as a local planning method for dexterous manipulation.

References

[1] A. M. Okamura, N. Smaby, M.R. Cutkosky, “An Overview of Dextrous Manipulation,” *Proc. IEEE International Conference on Robotics and Automation*, pp. 255–262, 2000.

[2] D. Prattichizzo, J.C. Trinkle, “Grasping” *Springer Handbook of Robotics*, B. Siciliano, O. Khatib (Eds.), Springer, Heidelberg, D, pp. 671–700, 2008.

[3] D. Montana, “The kinematics of multi-fingered manipulation”, *IEEE Transactions on Robotics and Automation*, vol. 11, pp. 491–503, 1995.

[4] B. Siciliano, L. Sciavicco, L.Villani, G. Oriolo, “Robotics. Modelling, Planning and Control.”, *Springer*, 2008.

[5] G. Antonelli, “Stability analysis for prioritized closed-loop inverse kinematic algorithms for redundant robotic systems”, *Proc. IEEE International Conference on Robotics and Automation*, pp. 1993-1998, 2008.

[6] L. Han, J. C. Trinkle, “The instantaneous kinematics of manipulation,” *Proc. IEEE International Conference on Robotics and Automation*, pp. 1944–1949, 1998.

[7] A. Bicchi, D. Prattichizzo, “Manipulability of cooperative robots with unactuated joints and closed-chain mechanisms”, *IEEE Transactions on Robotics and Automation*, vol. 16, pp. 336–345, 2000.

[8] D. Montana, “The Kinematics of Contact and Grasp,” *International Journal of Robotics Research*, vol. 7, no. 3, pp. 17–32, 1988.

[9] R.M. Murray, Z.X. Li, S.S. Sastry, *A mathematical introduction to robotic manipulation*, CRC press, Boca Raton, 1993.

[10] A.A. Cole, P. Hsu, S.S. Sastry, “Dynamic control of sliding by robot hands for regrasping”, *IEEE Transactions on Robotics and Automation*, vol. 8, pp. 42–52, 1992.