AN EXPERIMENTAL COMPARISON OF REDUNDANCY RESOLUTION SCHEMES

Alessandro Bettini **Alessandro De Luca** **Giuseppe Oriolo**

Dipartimento di Informatica e Sistemistica
Università degli Studi di Roma "La Sapienza"
Via Eudossiana 18, 00184 Roma, Italy
{bettini}@labrob1.ing.uniroma1.it
{deluca,oriolo}@labrob.ing.uniroma1.it

Abstract: We present an experimental comparison between two schemes for solving robot kinematic redundancy at the velocity level, based on the Projected Gradient and the Reduced Gradient methods. The schemes have been implemented on the 8R dof robot DEXTER, with the available joint range as a local optimization criterion. Performance is compared for two basic tasks: a self-motion reconfiguration and a point-to-point motion of the end-effector. *Copyright ©2000 IFAC*

Keywords: Redundant manipulators, kinematics, optimization, gradient methods, projected gradient, reduced gradient.

1. INTRODUCTION

Kinematic redundancy in robot manipulators allows to obtain more dextrous motion and improved interaction with the environment. The potential benefits of using redundancy are balanced by the difficulty in designing effective inverse kinematic schemes. In general, the redundant degrees of freedom are used to generate joint motions that reconfigure the structure according to secondary task specifications while executing the (primary) cartesian motion. Different techniques are available in the literature to specify a secondary task, like task space augmentation (Nakamura, 1991) or optimization of an objective function (Siciliano, 1990).

Most of the proposed techniques solve redundancy locally, with information limited to the current point of the task trajectory. The determination of an inverse kinematic scheme is usually approached at the velocity level, i.e., computing the nominal joint velocities that realize a given velocity task. Global methods have also been developed to achieve optimal behaviour along the whole task trajectory (Nakamura and Hanafusa, 1987), but they are impractical for real-time robot control applications.

In this paper, we review the use of the Projected Gradient (PG) and the Reduced Gradient (RG) schemes for redundancy resolution, introduced first in the robotic literature in (Liégeois, 1977) and in (De Luca and Oriolo, 1990*b*; De Luca and Oriolo, 1991), and apply them to the DEXTER robot, an 8-dof manipulator with all revolute joints. Experimental results are compared for a self-motion and a trajectory execution task, using the available joint range as the objective function.

2. LOCAL REDUNDANCY RESOLUTION

Consider a robot manipulator with n joints executing an m-dimensional task, being $n - m > 0$ the degree of redundancy. The direct kinematics is $p = f(q)$, with joint variables $q \in I\!R^n$ ($q = \theta$

for all revolute joints) and task variables $p \in \mathbb{R}^m$. The first-order differential kinematics is

$$\dot{p} = J(q)\dot{q}, \tag{1}$$

where the $m \times n$ matrix $J = \partial f / \partial q$ is the analytic Jacobian.

Given a desired task trajectory $p(t)$, an associated joint trajectory $q(t)$ is obtained by inverting eq. (1) at each time instant. Due to the arm redundancy, an infinite number of local joint solutions $\dot{q}$ exist, leaving room for optimization.

2.1 *Projected Gradient (PG) method*

This method is based on the use of homogeneous solutions to eq. (1), obtained by projecting a joint velocity vector in the null space of the Jacobian matrix. Homogeneous solutions generate internal motions that do not affect the motion in task space.

We have

$$\dot{q} = J^{\dagger}(q)\dot{p} + (I - J^{\dagger}(q)J(q))\,\eta, \tag{2}$$

where $J^{\dagger}$ is the unique pseudoinverse of J and $\eta \in \mathbb{R}^n$ is an arbitrary vector. The $n \times n$ matrix $(I - J^{\dagger}J)$ is the orthogonal projection operator into the null space of the Jacobian J and is of generic rank $n - m$. In the full row rank case, one has $J^{\dagger} = J^T(JJ^T)^{-1}$.

The standard PG method (Liégeois, 1977) is obtained by setting in eq. (2)

$$\eta = \alpha \nabla_q H(q), \tag{3}$$

where $H(q)$ is a joint-level performance criterion to be maximized and $\alpha > 0$ is a scalar step in the gradient direction. Note that, for a given q, the joint velocity solution (2) minimizes exactly the quadratic function

$$H'(\dot{q}) = \frac{1}{2}\dot{q}^T\dot{q} - \eta^T\dot{q},$$

subject to the linear constraint (1).

2.2 *Reduced Gradient (RG) method*

An alternative optimization method is based on the observation that the actual number of free variables in the problem is only $n - m$. Therefore, optimization of joint velocity can be performed in a reduced space.

Assume that the Jacobian is full row rank at q. We can always reorder and partition the joint variables into (q_a, q_b), with $q_a \in \mathbb{R}^m$ and $q_b \in \mathbb{R}^{n-m}$, so that in the Jacobian

$$J(q) = [\, J_a(q) \quad J_b(q) \,],$$

the $m \times m$ basis matrix J_a is nonsingular.

In (De Luca and Oriolo, 1991), it has been shown that all joint solutions can be generated as

$$\begin{bmatrix} \dot{q}_a \\ \dot{q}_b \end{bmatrix} = \begin{bmatrix} J_a^{-1}(q) \\ 0 \end{bmatrix} \dot{p} + \begin{bmatrix} J_R(q)J_R^T(q) & -J_R(q) \\ -J_R^T(q) & I \end{bmatrix} \eta, \tag{4}$$

where $J_R = J_a^{-1}J_b$ and the $(n - m) \times n$ matrix $[\, -J_R^T \quad I \,]$ is the reduction operator into the space of redundant joints q_b.

For optimizing an objective function $H(q)$, vector η can be defined as in eq. (3). As a result,

$$\dot{q}_b = \alpha\, [\, -J_R^T \quad I \,]\, \nabla_q H(q) \tag{5}$$

defines a motion in the direction of the reduced gradient of the objective function H.

2.3 *Objective function*

Examples of objective functions for optimal redundancy resolution include manipulability measures, distance from obstacles, and available joint range. The latter has been used in this paper to provide a benchmark for resolution methods. The objective function to be minimized is then

$$H(q) = \frac{1}{2n} \sum_{i=1}^{n} \left(\frac{q_i - q_{ic}}{q_{iM} - q_{im}} \right)^2, \tag{6}$$

where q_{iM}, q_{ic}, and q_{im} are respectively the upper bound, the central position and the lower bound of the available range for joint i ($i = 1, \ldots, n$). Thus, $\eta = -\alpha \nabla_q H(q)$ in the PG and RG methods.

2.4 *Comparison of PG and RG methods*

Both PG and RG methods solve redundancy locally and are thus suitable for real-time applications where execution time constraints are critical. In (De Luca and Oriolo, 1991), the following facts were shown theoretically and by simulations on simple manipulators.

- The RG and PG methods generate different motion directions in the configuration space, except when the degree of redundancy is $n - m = 1$ and a self-motion is performed (i.e., $\dot{p} \equiv 0$). In this particular case, the norm of the natural update (i.e., with $\alpha = 1$) of the RG method is always larger than with the PG method.

- Although the asymptotic rate of convergence of the two methods is similar, for most problems the RG method will converge faster.
- Once a feasible partition of the joint space is available, the number of computations at each step is smaller with the RG method. In fact, this method essentially requires the inversion of the $m \times m$ matrix J_a, while the PG method requires the pseudoinversion of the $m \times n$ Jacobian matrix J.

However, the RG method has some drawbacks.

- It requires to determine at each step a valid partition of the joint variables, such that J_a is nonsingular.
- A change of partition in the joint space corresponds to a velocity discontinuity.

The first problem can be tackled by several simple strategies. For example, one may evaluate the $\binom{n}{m}$ minors of the Jacobian at the initial configuration, order them by their minimum singular value (or determinant), and pick the best J_a to start with. By keeping trace of the first few minors in the list, one can replace the original joint partition with a better one only if needed. This strategy proved to be effective in our experiments.

A practical solution to the second problem is to smooth the velocity transition by a simple digital filter. Note also that this problem disappears if the RG method is defined at the acceleration level (De Luca and Oriolo, 1990*a*).

3. DEXTER ROBOT SYSTEM

The DEXTER robot, an 8R degrees of freedom manipulator developed by Scienzia Machinale, Pisa, is shown in Fig. 1. DC motors are located near the robot base and motion is transmitted to the remote links through pulleys and cables. Encoders mounted on the motor axes are used to measure joint positions.

The control architecture consists of a supervisor PC (Pentium-based), a motion control board and drivers. The PC executes C++ programs for cartesian motion planning and redundancy resolution, supplying velocity references at each sampling instant. The motion control board implements a PID control, receiving feedback from the robot (encoder values, end of joint range, brakes). The drivers realize a PWM on the input motor voltages.

3.1 *DEXTER Kinematics*

The direct kinematics is based on the frame assignment of Fig. 2 with the D-H parameters given in Tab. 1. This table reports also the available

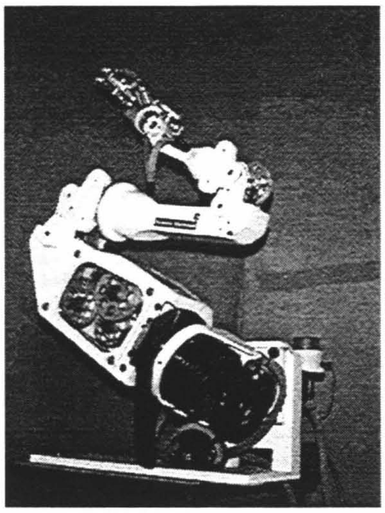
Fig. 1. The DEXTER robot

range for each joint. Note that joints are labeled from 0 to 7 and that there is a constant rotation matrix ${}^bR_0$ between the base frame and the 0-th robot frame.

The geometric 6×8 Jacobian $\hat{J}(q)$ of the robot, relating joint velocities to linear and angular velocities of the end-effector frame, is quite complex. However, the mid-frame Jacobian concept can be used for analysis and algorithmic implementation (Fijany and Bejczy, 1988). In particular, the DEXTER Jacobian takes on its simplest form when considering the origin of the fourth frame as the reference point attached to the end-effector, and expressing the linear and angular velocities in the fourth frame. The Jacobian ${}^4\hat{J}_4(q)$ is:

$$
{}^4\hat{J}_4 = \left[\begin{array}{c}
d_1 s_1 s_3 + d_3 s_3 c_2 s_1 - a_1 c_3 c_1 s_2 - d_1 c_3 c_1 c_2 - d_3 c_1 c_3 \\
-a_3 s_3 c_2 s_1 + a_3 c_3 c_1 + a_1 c_1 c_2 - d_1 c_1 s_2 \\
-d_3 c_3 c_2 s_1 - a_1 s_3 c_1 s_2 - d_1 s_3 c_1 c_2 - d_3 s_3 c_1 - d_1 s_1 c_3 + a_3 s_2 s_1 \\
-c_3 c_2 s_1 - s_3 c_1 \\
-s_2 s_1 \\
-s_3 c_2 s_1 + c_3 c_1
\end{array} \right.
$$

$$
\begin{array}{ccccc}
a_1 s_3 + d_3 s_3 s_2 & d_3 c_3 & 0 & 0 & 0 \\
-a_3 s_3 s_2 & -a_3 c_3 & 0 & 0 & 0 \\
-a_1 c_3 - d_3 c_3 s_2 - a_3 c_2 & d_3 s_3 & -a_3 & 0 & 0 \\
-c_3 s_2 & s_3 & 0 & 0 & -s_4 \\
c_2 & 0 & 1 & 0 & c_4 \\
-s_3 s_2 & -c_3 & 0 & 1 & 0
\end{array}
$$

$$
\left. \begin{array}{cc}
-a_5 s_4 - d_5 c_5 c_4 & -a_5 s_5 c_4 c_6 + d_5 s_5 s_6 c_4 \\
-d_5 c_5 s_4 + a_5 c_4 & d_5 s_5 s_6 s_4 - a_5 s_5 s_4 c_6 \\
d_5 s_5 & -a_5 c_6 c_5 + d_5 c_5 s_6 \\
-c_4 s_5 & -c_4 c_5 s_6 + s_4 c_6 \\
-s_4 s_5 & -s_4 c_5 s_6 - c_4 c_6 \\
-c_5 & s_5 s_6
\end{array} \right] ,
$$

where $s_i = \sin q_i$ and $c_i = \cos q_i$, for $i = 0, 1, \ldots, 7$.

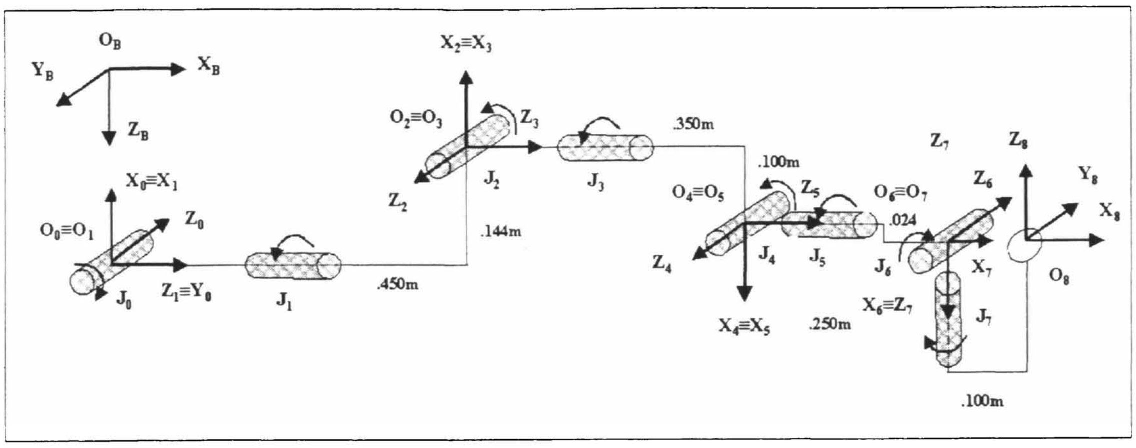

Fig. 2. Denavit-Hartenberg reference frames

i	a (mm)	d (mm)	α (rad)	range θ (deg)
0	0	0	$-\pi/2$	[-12.56, 179.89]
1	144	450	$-\pi/2$	[-83, 84]
2	0	0	$\pi/2$	[7, 173]
3	100	350	$\pi/2$	[65, 295]
4	0	0	$-\pi/2$	[-174, -3]
5	24	250	$-\pi/2$	[57, 265]
6	0	0	$-\pi/2$	[-129.99, -45]
7	100	0	π	[-55.05, 30]

Table 1. Denavit-Hartenberg parameters and joint range limits

4. IMPLEMENTATION

We describe some implementation details in the design of the overall kinematic robot controller.

4.1 *Task description*

A cartesian task of dimension $m = 6$ is described in the base frame in terms of end-effector position and orientation, using a minimal representation of the latter with roll-pitch-yaw angles (ϕ, θ, ψ)

$$^b p_e(t) = [x(t) \quad y(t) \quad z(t) \quad \phi(t) \quad \theta(t) \quad \psi(t)]^T .$$

In order to take advantage of the simple structure of the geometric Jacobian $^4\hat{J}_4$, we need to suitably transform the task description at the velocity level. First, we convert $^b\dot{p}_e$ into a linear and angular velocity vector $^b v_e \in I\!R^6$

$$^b v_e = \begin{bmatrix} I & 0 \\ 0 & T \end{bmatrix} {}^b\dot{p}_e = \begin{bmatrix} I & 0 \\ 0 & T \end{bmatrix} J(q)\dot{q},$$

where

$$T(\phi, \theta) = \begin{bmatrix} 0 & -\sin\phi & \cos\phi\cos\theta \\ 0 & \cos\phi & \sin\phi\cos\theta \\ 1 & 0 & -\sin\theta \end{bmatrix} .$$

Next, we change the reference point to the origin O_4 (see Fig. 2) and represent vectors in the fourth frame. As a result, given the end-effector task $^b\dot{p}_e$, the equivalent task velocity vector $^4 v_4$ is determined as

$$^4 v_4 = \begin{bmatrix} ^4R_b & ^4R_b\, ^bS_{4e}\, T \\ 0 & ^4R_b T \end{bmatrix} {}^b\dot{p}_e = {}^4\hat{J}_4(q)\dot{q}, \quad (7)$$

where ^4R_b is the rotation matrix from the fourth to the base frame and the skew-symmetric matrix

$$^bS_{4e} = \begin{bmatrix} 0 & ^b r_{4e,z} & -^b r_{4e,y} \\ -^b r_{4e,z} & 0 & ^b r_{4e,x} \\ ^b r_{4e,y} & -^b r_{4e,x} & 0 \end{bmatrix}$$

contains the components of the position vector $^b r_{4e}$ from O_4 to the end-effector position, expressed in the base frame. Via eq. (7), the PG and RG methods are defined using directly the mid-frame geometric Jacobian.

4.2 *Task error compensation*

Due to the tangent linearization intrinsic in velocity schemes, numerical errors accumulate introducing a drift in the execution of the cartesian task. To compensate for this, a task space position error control loop must be added. Given a desired task motion $^b p_{e,d}(t)$, the actual velocity reference is

$$^b\dot{p}_e(t) = {}^b\dot{p}_{e,d}(t) + K_p e(t),$$

where $e = {}^b p_{e,d} - f(q)$ and $K_p > 0$ is a gain matrix.

4.3 *Damping near singular configurations*

Both PG and RG methods share the same problems near singular configurations of the manipulator. To avoid a buildup of joint velocities, we have implemented a damped least-squares (DLS) method to realize a trade-off between task error and joint velocity (Chiaverini and Egeland, 1990).

In the PG method, a DLS routine is switched on when the smallest singular value σ_m of the Jacobian falls below a threshold $\varepsilon > 0$. Using the Singular Value Decomposition (Nakamura, 1991) of the Jacobian, $J = V\Sigma U^T$ with V and U orthonormal matrices of dimensions $m \times m$ and $n \times n$, the standard pseudoinverse $J^\dagger$ is

$$J^\dagger = U\Sigma^\dagger V^T = U \begin{bmatrix} \mathrm{diag}\{\sigma_1^\dagger, \dots, \sigma_m^\dagger\} \\ 0_{(n-m)\times m} \end{bmatrix} V^T,$$

where $\sigma_i^\dagger = 1/\sigma_i$ if $\sigma_i > 0$, and zero otherwise. The DLS scheme replaces the $\sigma_i^\dagger$ by

$$\sigma_i^{\mathrm{DLS}} = \frac{\sigma_i}{\sigma_i^2 + \lambda_i^2}, \quad i = 1, \dots, m.$$

In our implementation, the ith damping factor λ_i is a smooth function of the relative singular value:

$$\lambda_i^2(\sigma_i) = \begin{cases} 0, & \text{for } \varepsilon < \sigma_i \\ \lambda_{\max}^2 - \frac{\lambda_{\max}^2}{2\pi}\left(\frac{2\pi(\sigma_i - \rho\varepsilon)}{(1-\rho)\varepsilon} - \sin\frac{2\pi(\sigma_i - \rho\varepsilon)}{(1-\rho)\varepsilon}\right), & \\ & \text{for } \rho\varepsilon < \sigma_i \le \varepsilon \\ \lambda_{\max}^2, & \text{for } \sigma_i \le \rho\varepsilon \end{cases}$$

with the constants $0 \le \rho < 1$ and $\lambda_{\max} > 0$.

The same DLS scheme can be used also in the RG method for damping the inverse of the $m \times m$ square matrix J_a, whenever it is not possible to find a well-conditioned minor of J.

4.4 *Step choice in the gradient direction*

The choice of an optimal stepsize α in eqs. (3) or (5) may improve considerably the convergence rate to the locally optimal solution.

Given a joint configuration q_k at a sampling time t_k, both eqs. (2) and (4) result, after discretization, in updates of the form

$$q_{k+1} = a_k + \alpha\, b_k. \tag{8}$$

Using the objective function (6), the optimal step α_{opt} is determined by the unique solution to the unconstrained quadratic optimization problem

$$\min_{\alpha \in \mathbb{R}} \frac{1}{2n} \sum_{i=1}^{n} \left(\frac{a_{ki} + \alpha\, b_{ki} - q_{ic}}{q_{iM} - q_{im}}\right)^2,$$

where a_{ki} and b_{ki} are components of a_k and b_k.

However, α_{opt} may not be feasible in the presence of hard constraints on joint velocity and joint range:

$$q_m \le q \le q_M, \qquad \dot{q}_m \le \dot{q} \le \dot{q}_M.$$

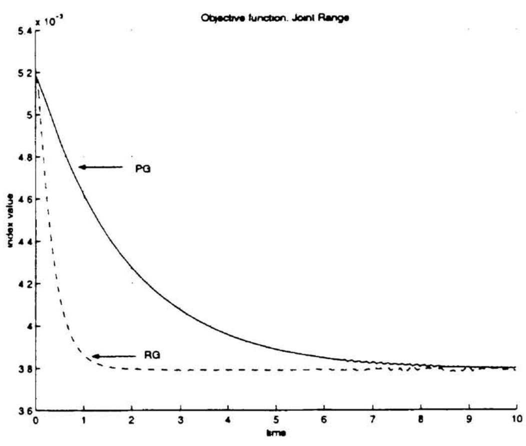

Fig. 3. Objective function in a self-motion task

Combining these, q_{k+1} given by eq. (8) should satisfy

$$\max\{q_m, q_k + \dot{q}_m T_c\} \le q_{k+1} \le \min\{q_M, q_k + \dot{q}_M T_c\}, \tag{9}$$

where T_c is the sampling time, while max and min are intended componentwise. A maximum feasible step $\alpha_{\max}$, satisfying inequalities (9), can be determined in closed form.

5. EXPERIMENTAL RESULTS

The comparison between PG and RG methods has been made on self-motions and point-to-point motion tasks. The initial configuration is

$$q_{\mathrm{in}} = [40 \;\; 0 \;\; 90 \;\; 180 \;\; -90 \;\; 180 \;\; -90 \;\; 0]^T \text{ (deg)},$$

and we have used $K_P = 5I$ and $T_c = 25$ ms. The DLS routine uses $\lambda_{\max} = 0.1$, $\varepsilon = 0.2$, and $\rho = 0.25$.

For the self-motion task, the end-effector is kept fixed at the cartesian pose $f(q_{\mathrm{in}})$, i.e.,

$$p_{\mathrm{in}} = [1020 \;\; 0 \;\; 180 \;\; 0 \;\; -40 \;\; 0]^T \text{ (mm,deg)},$$

and the robot reconfigures its internal structure so as to minimize the objective function (6).

The optimization performance is shown in Fig. 3, from which the faster convergence of the RG method is clear. No optimization is made on the gradient stepsize, using the same values $\alpha = 10$ for PG and RG. The J_a matrix in the RG method is obtained by deleting the second and seventh columns from ${}^4\hat{J}_4$. No change of basis is needed here.

For the point-to-point task, we have chosen a coordinated motion along a straight line in the task space between p_{in} and

$$p_{\mathrm{fin}} = [600 \;\; -300 \;\; 500 \;\; 10 \;\; -20 \;\; 40]^T \text{ (mm,deg)}.$$

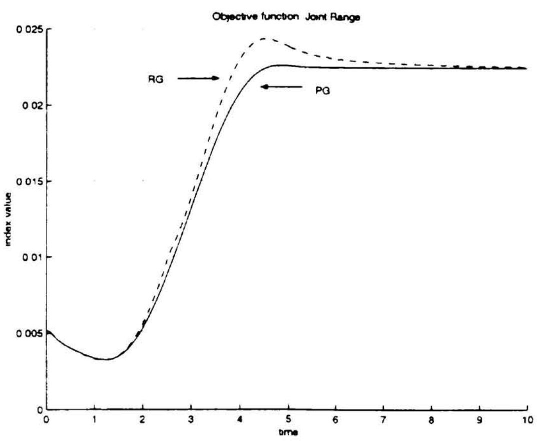

Fig. 4. Objective function in a point-to-point task

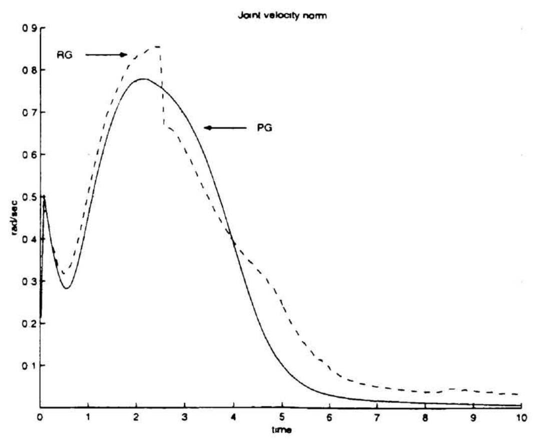

Fig. 5. Joint velocity norm in a point-to-point task

The desired trajectory

$$^{b}p_{e,d}(t) = p_{\text{in}} + s(t)\,(p_{\text{fin}} - p_{\text{in}})$$

is parametrized by a quintic polynomial with zero initial and final velocity and acceleration

$$s(t) = 6\left(\frac{t}{T}\right)^5 - 15\left(\frac{t}{T}\right)^4 + 10\left(\frac{t}{T}\right)^3, \quad t \in [0, T],$$

where $T = 5$ s is the motion time. The whole experiment lasts 10 seconds and the extra time is used to correct small residual cartesian errors, if present, and possibly continue the optimization of the objective function.

The optimization performance of the two methods is analogous (Fig. 4). From Fig. 5, one can see that the RG method switches basis around $t = 2.5$ s, where the fifth column of $^{4}\hat{J}_4$ is replaced by the seventh one. This transition is accomodated over three sampling intervals. The maximum task error is less than 0.5 cm on the linear components and less than 0.35 deg on the angular components.

6. CONCLUSIONS

We have shown the feasibility of an alternative approach to optimization-based redundancy resolution, the so-called Reduced Gradient method introduced in (De Luca and Oriolo, 1991). Real-time execution constraints are satisfied without problems, while the potential drawbacks (need for a search of a suitable minor of the Jacobian, discontinuous velocity when changing the base) are overcome by simple heuristics. Compared to the Projected Gradient method, we have found a faster convergence to the optimal objective value in a self-motion task and comparable performance for trajectory execution. The structure of the mid-frame robot Jacobian could be further exploited in order to efficiently predict the best basis for the Reduced Gradient method. We have also implemented this method at the acceleration level with satisfactory results.

REFERENCES

Chiaverini, S. and O. Egeland (1990). An efficient pseudo-inverse solution to the inverse kinematic problem for six-joint manipulators. *Modeling, Identification and Control*, **11**(4), 201–222.

De Luca, A. and G. Oriolo (1990*a*). Efficient dynamic resolution of robot redundancy. In: *Proc. 1990 American Control Conf.*. San Diego, CA. Pp. 221–227.

De Luca, A. and G. Oriolo (1990*b*). Kinematic resolution of redundancy via joint-space decomposition. In: *Proc. 8th CISM-IFToMM Symp. on Theory and Practice of Robots and Manipulators (Ro.Man.Sy. '90)*. Krakow, PL. Pp. 64–71.

De Luca, A. and G. Oriolo (1991). The reduced gradient method for solving redundancy in robot arms. *Robotersysteme*, **7**(2), 117–122.

Fijany, A. and A.K. Bejczy (1988). Efficient jacobian inversion for the control of simple robot manipulators. In: *Proc. 1988 IEEE Int. Conf. on Robotics and Automation*. Philadelphia, PA. Pp. 999–1007.

Liégeois, A. (1977). Automatic supervisory control of the configuration and behavior of multibody mechanisms. *IEEE Trans. on Systems, Man, and Cybernetics*, **7**, 868–871.

Nakamura, Y. (1991). *Advanced Robotics: Redundancy and Optimization*. Addison-Wesley.

Nakamura, Y. and H. Hanafusa (1987). Optimal redundancy control of robot manipulators. *Int. J. of Robotics Research*, **6**(1), 32–42.

Siciliano, B. (1990). Kinematic control of redundant robot manipulators: A tutorial. *J. of Intelligent and Robotic Systems*, **3**, 201–212.