ELSEVIER

Robotics and Autonomous Systems 41 (2002) 41–59

Robotics and Autonomous Systems

www.elsevier.com/locate/robot

Real-time control of redundant robotic manipulators for mobile obstacle avoidance

V. Perdereau[*], C. Passi, M. Drouin

Laboratoire des Instruments et Systèmes d'Ile-de-France, Université Pierre et Marie Curie, boite 164, 4 place Jussieu, 75252 Paris Cedex 05, France

Received 21 November 2001; received in revised version 25 March 2002
Communicated by F.C.A. Groen

Abstract

This paper offers an efficient solution for the real-time operation of a redundant robot moving in a variable environment. An approach for fixed and mobile obstacle avoidance is proposed. The basic idea is to solve the redundancy as an unconstrained optimization problem where the redundancy is integrated with the path tracking and the obstacle avoidance constraints into an augmented objective function. Here, an iterative solution of the Inverse Geometric Model (IGM) is used, which requires no matrix inversion and iterates directly on the joint position, being thus suitable for on-line application and also preserving repeatability. A method of formulating anti-collision constraints using a novel concept of pseudo-distance unifies in a simple and original way the modeling of various obstacles.
© 2002 Elsevier Science B.V. All rights reserved.

Keywords: Collision avoidance; Redundancy resolution; Inverse kinematic model; Manipulator control

1. Introduction

Robot manipulators are destined to perform efficiently very complex tasks in cluttered environments. In particular, they are required to move in the presence of fixed or even mobile obstacles, tracking a prescribed path without collision in no way. Hence, robot control aims to improve the arm dexterity and allow it to react fast enough to sudden and unpredictable changes in its workspace configuration.

Some methods for generating collision-free paths are adapted from mobile robots [12,22,26,32,33]. They generally consist of two steps: (i) description of the accessible space and (ii) search for a collision-free path, transforming the arm into an equivalent point-like mobile robot. Such an approach allows only off-line path planning. Moreover, robotic serial manipulators—in contrast to mobile robots—need to avoid collisions of both the end-effector and the links. For this reason, their accessible workspace is rather limited unless their number of joints increases. These methods are, however, not suitable for redundant robots since the arm transformation does not lead to a unique point.

In the so-called kinetic analysis method [7,16,17,23], the manipulator is reduced to its spinal cord made of its joints and links, requiring the use of differential geometry of curves. Since the arm skeleton must adhere to the path, manipulators of the hyper-redundant type having a large number of degrees of freedom are best suited for this approach. This is a global method—

[*] Corresponding author. Tel.: +33-1-44-27-62-11; fax: +33-1-44-27-62-14.
E-mail address: vperd@ccr.jussieu.fr (V. Perdereau).

which ends up at trajectory planning—dealing mainly with planar manipulators. Its applicability is therefore limited.

In other methods coming from the idea of causing a manipulator back movement whenever an obstacle is approached, each obstacle is surrounded by a security zone. Within it, the obstacle is perceived by the manipulator as generating a virtual force that increases with the proximity. In one approach [31], the repulsive force is generated by a fictitious spring–damper system via a force control loop external to the position loop. This method is suitable mainly for end-effector collision avoidance. In another approach introduced by Khatib [18], the virtual torques applied on the joints are produced by an artificial repulsive potential field [18–20], which makes possible for the entire arm to avoid even mobile obstacles. The repulsive forces are derived from a Cartesian energy function and then projected onto the joint space in order to obtain the obstacle avoidance torque.

The collision avoidance problem can also be formulated as a constrained optimization problem since the minimum distance between links and obstacle must become greater than a security margin. For that purpose, global and local optimization methods are available. Global methods minimize a cost function (actuator energy or travel time) over the entire path, subject to constraints like end-effector path tracking or obstacle avoidance. They give once and for all the successive configurations for the execution of the task. Hence, they cannot consider a variable environment; they are not suitable for on-line applications. With local methods, joint positions are successively obtained as the robot moves along its path, taking into account only local conditions. They are best suited for mobile obstacle avoidance.

In this last approach, the collision avoidance problem can be treated effectively by solving on-line the redundancy of the robot. Indeed, as invoked earlier, increasing their number of joints can improve the dexterity of robotic manipulators and extend their reachable workspace. A kinematic redundancy leads to an infinity of possible joint positions for the same pose of the end-effector. This leads naturally to the idea that such a robot can accomplish a primary task with his end-effector and simultaneously optimize a secondary criterion and/or satisfy certain constraints with the supplementary degrees of freedom.

The redundancy solution of serial manipulators is obtained by solving either their *Inverse Kinematic Model* (IKM) or their *Inverse Geometric Model* (IGM) depending on the kind of position control scheme implemented. The two basic position control schemes are joint space control and Cartesian space control [36]. The latter uses the IKM, which can be done in two ways: (i) by the generalized inverse Jacobian [5,9,21,24,39], whose computation is very time consuming and whose repeatability is not always satisfied, and (ii) by the reduced or extended Jacobian [4], where problems of inevitability and kinematic/algorithmic singularity may appear. Moreover, all inverse Jacobian approaches are based on Cartesian position control, while joint position control is the preferable method for the sake of stability [27] owing to the absence of transformations inside the position loop.

In joint space control, the IGM is used, the constraints are then integrated into the model by the augmented space approach [2,3,25]. This may induce, however, algorithmic singularities due to the conflict between path tracking (primary task) and obstacle avoidance (secondary task). A remedy consists of prioritizing the task [6], but this solution uses the Jacobian pseudo-inverse which does not ensure repeatability and is time consuming.

For all these reasons, we have already developed a rather general method [28] allowing to invert the geometric model of a robot in every arm configuration and also to account for various types of optimization criteria and/or constraints. This paper presents its adaptation to the collision avoidance problem. Now, the major difficulty is to express the anti-collision constraints. In the literature, they are usually formulated using Euclidean distance measures [13,15,35]. Collision avoidance constraints based on the minimum Euclidean distance requires the knowledge of the analytical expression of the distance, which is very difficult to obtain for complex-shaped objects. An alternative solution is the polyhedral approach [32,35], which deals with those difficult obstacles by surrounding them with complex polyhedrons. Not only this reduces the available workspace but also results in non-differentiable distance functions.

Finally, all existing minimum distance approaches, including other algorithmic methods, are time consuming, therefore not suitable for real-time implementation. Hence, a simpler formulation of anti-collision constraints is proposed in this paper. The obstacle is surrounded by a super-quadratic surface and the robot proximity is evaluated by a pseudo-distance which allows us to easily express the constraint. This anti-collision constraint, together with other task-or-robot related constraints are introduced in the extended objective function via penalty functions which is then minimized to yield the inverse solution of the IGM algorithm.

This paper is organized as follows. The iterative algorithm we have already proposed for solving the redundancy in the geometric model inversion is first recalled in Section 2. Then, Section 3 deals with the collision avoidance issue as a constrained optimization problem where in order to account for the anti-collision constraints, an extended objective function is formed using penalty functions. The new concept of pseudo-distance is introduced in Section 4 to express in a simple manner the anti-collision constraints. The latter are adapted to avoid collision with mobile obstacles in Section 5 and simulation results prove in Section 6 the efficiency of the proposed approach to control on-line the global arm configuration.

2. On-line iterative solution of the inverse geometric model

It has been long recognized that one way to improve versatility of robotic manipulators is to increase their number of joints leading to an infinity of possible joint positions for the same pose of the end-effector, that is, a kinematic redundancy, thereby offering the capability to reconfigure the arm without affecting the tool position. However, the main difficulty for their end-effector position control is to obtain the inverse model of the redundant serial manipulators either the IGM in joint space or the IKM in Cartesian space [36]. Implementing the controller in joint space is preferable since the absence of transformations in the position loop offers many advantages such as an easier controller design independent on the task configuration ensuring stability and a lower sampling period leading to better reference matching and disturbance rejection. In this control scheme (Fig. 1), the problem lies in determining the desired joint position vector q_d for a given Cartesian position x_d on the path, that is, to solve the IGM in selecting one particular arm configuration over an infinity.

In order to solve the kinematic redundancy, it is necessary to specify the evolution of the robot configuration while the end-effector is moving on a prescribed path, that is, to parameterize the self motion of the manipulator corresponding to the movements of the links that do not change the end-effector location. Hence, the robot can accomplish a specified basic task such as trajectory tracking and additional tasks simultaneously. The manipulator control can therefore be viewed as a constrained optimization problem where in order to ensure the end-effector position, one looks for the joint trajectory minimizing the objective function $\varphi(q)$ subject to the path tracking constraint:

$$x = f(q), \tag{1}$$

where f is the well-known geometric model of the robot mechanical structure.

The objective function $\varphi(q)$ is any general function representing a measure of some kinematic characteristic of the robot performance so that the redundant degrees of freedom are exploited to meet additional purposes: minimization of joint torques, maximization of manipulability or dexterity measures, singularity or mechanical limit avoidance, posture control, obstacle avoidance, etc. [8,21,30,38,40].

2.1. Extended task space formulation

A solution to this constrained optimization problem is found using the Lagrangian approach. Given a matrix N which spans the null space of the Jacobian

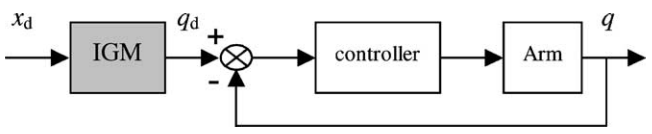


Fig. 1. Joint space position control.

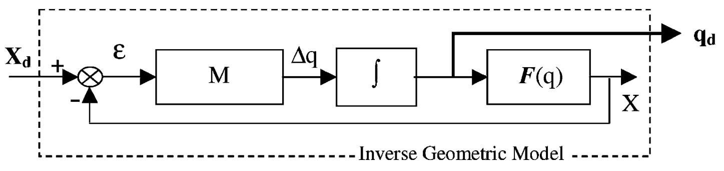


Fig. 2. IGM, the equivalent closed loop process.

matrix, the optimality condition becomes

$$N^{\mathrm{T}}\frac{\partial\varphi}{\partial q}=0. \tag{2}$$

Hence, Baillieul [1] introduces the projection of the objective function gradient onto the null space in an extended Cartesian space vector:

$$X=\begin{bmatrix} f(q) \\ N^{\mathrm{T}}\dfrac{\partial\varphi}{\partial q}\end{bmatrix}, \tag{3}$$

and the optimality of the solution is ensured by defining an extended desired Cartesian position:

$$X_{\mathrm{d}}=\begin{bmatrix} x_{\mathrm{d}} \\ 0\end{bmatrix}. \tag{4}$$

Consequently, an extended forward kinematic model relates the new task vector X to the joint angle vector q:

$$X=F(q)=\begin{bmatrix} f(q) \\ N^{\mathrm{T}}\dfrac{\partial\varphi}{\partial q}\end{bmatrix}, \tag{5}$$

and the corresponding extended Jacobian matrix $J_{\mathrm{e}}=\partial F(q)/\partial q$ is square.

Indeed, this kinematic representation is no longer redundant: the dimensions of the augmented task space and joint space are the same. However, the analytical expression of the inverse geometric model results from a very complicated even impossible inversion of a non-linear function of constraints that, moreover, may change widely and rapidly during the trajectory. It is therefore preferable to use an iterative method for that purpose.

2.2. Iterative solution

Contrary to global optimal schemes, local approaches do not consider the whole tool trajectory but incrementally specify joint displacements at a sequence of points $X_{\mathrm{d}}(k)$ along the path, where k is the time index. Hence, for a given Cartesian position, the proposed on-line inversion looks for a local joint solution q_{d} in an iterative manner:

$$q_{c+1}=q_c+\eta\Delta q_c, \tag{6}$$

where Δq_c is the correction term at iteration c and η is a reduction factor ($0<\eta<2$). The extended Cartesian error is defined by

$$\varepsilon_c=X_{\mathrm{d}}-F(q_c). \tag{7}$$

We consider that the solution is found:

$$q_{\mathrm{d}}(k)=q_c, \tag{8}$$

when this error becomes lower than a predefined limit:

$$|\varepsilon_c|<\varepsilon_{\mathrm{lim}}. \tag{9}$$

The challenging problem of this algorithm is therefore to minimize the extended Cartesian error ε_c in a finite number of iterations.

Several algorithms take the form of a closed loop inverse kinematic scheme where the inverse kinematic problem is reformulated in terms of the convergence for an equivalent feedback system [6] as represented in Fig. 2.

Each iteration c involves the computation of a matrix M_c, the updated joint position q_c, the extended geometric model $F(q)$ and the extended error ε_c. The various local solutions in the literature [10,11,14,34,37] differ only by the choice of the matrix M controlling the recurrent correction of q, requiring the computation of either the extended Jacobian transpose or the Jacobian pseudo-inverse, but all incur problems of matrix positive definiteness and time-consuming matrix inversion. To eliminate these shortcomings, we have proposed a novel correction matrix M [28].

2.3. The proposed algorithm

First, a positive definite Lyapunov function depending on the Cartesian position error is chosen:

$$V_c = \tfrac{1}{2}\varepsilon_c^{\mathrm{T}}\varepsilon_c, \tag{10}$$

which ensures stability and rapid convergence of the algorithm when its variation between two iterations is adjusted as

$$\Delta V_c = V_{c+1} - V_c = -\alpha V_c, \tag{11}$$

since it imposes an exponential decrease of the Lyapunov function V_c $(0 < \alpha \le 1)$. In the following, the parameter α is set to 1 in order to obtain the algorithm convergence in one iteration:

$$V_{c+1} = (1-\alpha)V_c = 0. \tag{12}$$

Then

$$\Delta V_c = \varepsilon_c^{\mathrm{T}}\Delta\varepsilon_c + \tfrac{1}{2}\Delta\varepsilon_c^{\mathrm{T}}\Delta\varepsilon_c = -\tfrac{1}{2}\varepsilon_c^{\mathrm{T}}\varepsilon_c, \tag{13}$$

which is equivalently written as

$$[\varepsilon_c + \Delta\varepsilon_c]^{\mathrm{T}}[\varepsilon_c + \Delta\varepsilon_c] = 0, \tag{14}$$

implying

$$\Delta\varepsilon_c = -\varepsilon_c. \tag{15}$$

Since

$$\Delta\varepsilon_c = \Delta(X_{\mathrm{d}} - X_c) = -\Delta X_c, \tag{16}$$

and

$$\Delta X_c = J_{\mathrm{e}}\Delta q_c, \tag{17}$$

we obtain the following iterative correction term:

$$\Delta q_c = J_{\mathrm{e}}^{-1}\varepsilon_c. \tag{18}$$

Here, the correction matrix is then simply the inverse extended Jacobian matrix:

$$M = J_{\mathrm{e}}^{-1}. \tag{19}$$

However, such a solution may lead to undesirable joint velocities whenever J_{e} is singular. This happens not only at joint configuration for which the Jacobian matrix $\partial f(q)/\partial q$ is rank deficient, that is, at any singular configuration (kinematic singularities) but also at some other configurations hardly predictable due to the additional task requirements (algorithmic singularities) [30]. We have therefore developed another version of this algorithm [29].

A first order approximation in (13) gives

$$\varepsilon_c^{\mathrm{T}}\Delta\varepsilon_c = -\tfrac{1}{2}\varepsilon_c^{\mathrm{T}}\varepsilon_c, \tag{20}$$

which leads immediately to

$$\varepsilon_c^{\mathrm{T}}J_{\mathrm{e}}\Delta q_c = \tfrac{1}{2}\varepsilon_c^{\mathrm{T}}\varepsilon_c. \tag{21}$$

Hence, the iterative correction term is now

$$\Delta q_c = \tfrac{1}{2}(\varepsilon_c^{\mathrm{T}}J_{\mathrm{e}})^{\dagger}\varepsilon_c^{\mathrm{T}}\varepsilon_c, \tag{22}$$

involving a pseudo-inversion of the vector $\varepsilon_c^{\mathrm{T}}J_{\mathrm{e}}$ so that the matrix M has the following simple expression:

$$M_c = \frac{(\varepsilon_c^{\mathrm{T}}J_{\mathrm{e}})^{\mathrm{T}}\varepsilon_c^{\mathrm{T}}}{2\left\|J_{\mathrm{e}}^{\mathrm{T}}\varepsilon_c\right\|^2}. \tag{23}$$

The solution requires a low number of iterations but in any cases the number of iterations is bounded. The maximum limit attained means that the algorithm is not able to converge. However, this situation never happens since the solution is to be found iteratively from one configuration to another close one, after a correct initialization.

Due to the first order approximation, the convergence of the algorithm is inherently slower compared to the solution derived in (18) this means that more iterations are required to find the solution but the computation of the matrix M is much simpler and involves only few additions and multiplications so that the global computing time between two points on the trajectory is just a little bit higher. This algorithm can still converge within a sampling period making possible its on-line implementation.

The chosen correction matrix implies, on the other hand, no inversion of the extended Jacobian matrix J_{e} so that neither kinematic nor algorithmic singularities are introduced. Finally, the on-line IGM solution using this version of M appears to be a very efficient solution due to its good stability, small response time and simplicity of control tuning.

3. Collision avoidance: an optimization problem

The primary task of a robotic arm is to move its end-effector on a desired trajectory but, thanks to a

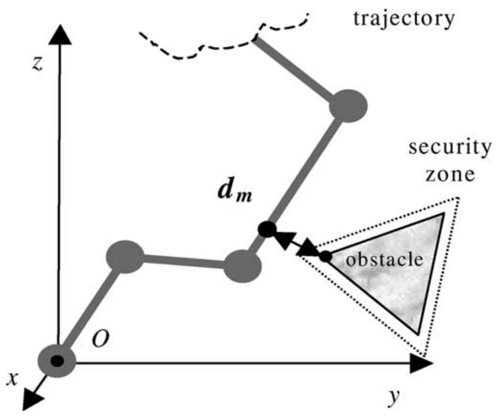


Fig. 3. The collision avoidance constraint.

redundant structure, the arm can simultaneously accomplish a secondary task. In the particular case of obstacle avoidance as secondary task, virtual torques produced by artificial potential fields can be applied on the joints [18–20]. The obstacle avoidance constraint, together with other task—or robot—related constraints, can also be accounted for in an objective function $\varphi(q)$ so that the collision avoidance problem becomes a constrained optimization.

The primary task which is path tracking is formulated as an equality constraint:

$$f(q) - x_{\mathrm{d}} = 0, \tag{24}$$

expressing zero Cartesian error. Hence, solving the redundancy is carried out by minimizing the objective function $\varphi(q)$ characterizing the additional task, subject to the previous equality constraint, provided that the additional constraints are included into the criterion.

Collision avoidance constraints are usually based on the minimum Euclidean distance d_{m} between a robot link and the obstacle, requiring this distance to be higher than a security margin d_{s} so that the robot arm never enters the security zone surrounding the obstacle (Fig. 3). The anti-collision constraint $h(q)$ is equivalently written in this case as

$$h(q) = -d_{\mathrm{m}}^2 + d_{\mathrm{s}}^2 \le 0. \tag{25}$$

Hence, these constraints are expressed as inequality constraints in the form

$$h(q) \le 0. \tag{26}$$

3.1. Penalty function approach

Since a manipulator is made of several links and there may be several obstacles in the environment, several constraints must be introduced into the objective function. For this purpose, one possible way, based on a penalty function approach, is to define an augmented objective function as

$$\varphi_{\mathrm{e}}(q) = \varphi(q) + \sum_{i=1}^{l} \alpha_i p(h_i(q)), \tag{27}$$

where l is the number of constraints and α_i the ith weighting coefficient.

Indeed, a penalty function $p(h(q))$ aims to describe how the task requirements are fulfilled, that is, this function tends to a null effect when the constraint $h(q)$ is satisfied. There are several formulations of the penalty functions applied to the constraints in order to include them in the IGM solution.

- *External penalty functions* account for the constraint only when this is violated:

$$\begin{array}{lll} p(h) > 0 & \text{if} & h(q) \ge 0, \\ p(h) = 0 & \text{if} & h(q) = 0. \end{array} \tag{28}$$

One possible external penalty function is

$$p(h(q)) = (h(q))^2 \Gamma(h), \tag{29}$$

as illustrated in Fig. 4, where $\Gamma(h)$ is the Heaviside function. In one hand, this function maximizes the free space, but on the other impacts may occur creating important torques and steady state errors. Hence, this is the worst suited penalty function for avoiding moving obstacles.

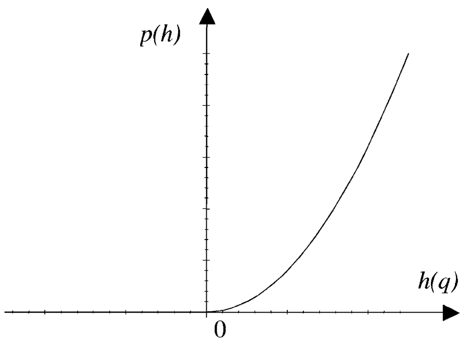


Fig. 4. The external penalty function.

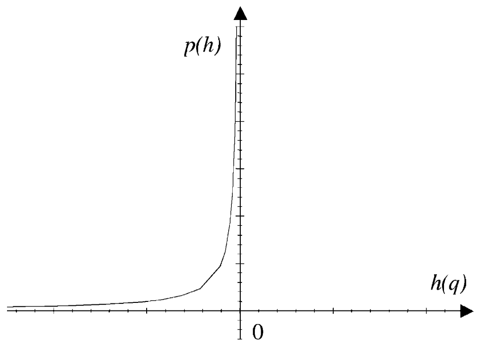


Fig. 5. The internal penalty function.

- *Internal penalty functions*, on the contrary, avoid impacts and acts at all times in a repulsive field manner:

$$\begin{aligned} p(h) &> 0 \quad \text{if } h(q) < 0, \\ p(h) &\to \infty \quad \text{if } h(q) \to 0. \end{aligned} \tag{30}$$

  In general, this function is expressed as

$$p(h) = -\frac{1}{h(q)}, \tag{31}$$

  which behaves as represented in Fig. 5. It smoothens the joint torques but reduces the workspace.
- *Limited internal penalty functions* are the best suited for obstacle avoidance. They limit the influence of the constraint to a range h_0 so as not to affect the robot when the obstacle is far away:

$$p(h) = \begin{cases} \dfrac{1}{2}\left(\dfrac{1}{h(q)} - \dfrac{1}{h_0}\right)^2 & \text{if } h_0 < h(q) < 0, \\ 0 & \text{if } h(q) \le h_0. \end{cases} \tag{32}$$

  This function is illustrated in Fig. 6.

3.2. A new extended direct geometric model

The appropriate joint configuration vector q_d avoiding collision is the inverse solution in the IGM algorithm described in the previous section, when minimizing now the augmented objective function $\varphi_e(q)$ instead of $\varphi(q)$. The augmented geometric model is then

$$X = F(q) = \begin{bmatrix} f(q) \\ N^T \dfrac{\partial \varphi_e}{\partial q} \end{bmatrix}, \tag{33}$$

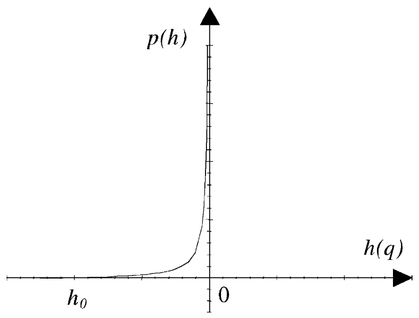


Fig. 6. The limited internal penalty function.

but nothing else is changed in the iterative method. This clearly shows the main advantage of a joint position control loop versus a Cartesian control loop: no changes are introduced in the controller due to the inclusion of the anti-collision constraints. Next section is now devoted to describe constraints so that they are differentiable and simple.

4. Anti-collision constraints: a pseudo-distance approach

In the extended kinematic model (33), the augmented objective function will be differentiated twice with respect to the joint variables. Therefore, the constraints must be twice differentiable scalar functions of the joint variables. Moreover, they must be simple in order to allow an on-line implementation of the algorithm.

It is intuitively clear that the minimum distance is the Euclidean distance between the closest points, one belonging to the arm and the other to the obstacle. However, it is very difficult to express it in an analytical form. The minimum distance has to be evaluated for each particular case since there is no general formulation. Moreover, the Euclidean distance becomes complicated for complex object shapes and is then often itself the result of an optimization method so that the solution is non-differentiable.

In fact, we do not need to compute the Euclidean distance between the obstacle and a given point. The aim is only to obtain an inequality constraint ensuring that the point is not within the obstacle or its envelop if this obstacle is complex. For all these reasons, a new constraint formulation is proposed in this paper where the obstacle shape is encompassed within an analytical surface function and the robot proximity evaluated by a pseudo-distance which allows to express the constraint.

4.1. Pseudo-distance with respect to a point

We introduce here the concept of pseudo-distance for evaluating the position of a point located on the robot with respect to an object. The idea is to define an analytical function describing the object surface (or, alternatively, to envelop it in an hyper-surface of known analytical expression) and then to check the point position with respect to this envelope.

For that purpose, we have chosen to use super-quadric surface functions. Indeed, the problem with surface function is that beside differentiability and simplicity, they must also describe closely the object volume otherwise a reduction of the workspace will result. In this sense, super-quadric surface functions allow to approximate in a simple manner a large number of obstacles (see Appendix A where few examples are shown). Using the ellipse or the ellipsoid as basic shapes, it is possible to approximate various shapes like rectangles, parallelepipeds, cylinders and others by simple adjustment of the powers in the function expressions. Even more complex forms like trapezoids, pyramids and cones can be approximated by replacing the semi-axes a, b, c of the ellipse/ellipsoid with scalar functions of x, y, z so that the analytical expression of any obstacle is

$$S(x,y,z)=\left(\frac{x}{f_1(x,y,z)}\right)^{2n}+\left(\frac{y}{f_2(x,y,z)}\right)^{2m}+\left(\frac{z}{f_3(x,y,z)}\right)^{2p}. \tag{34}$$

However, the closer the envelop is, the larger the workspace is but the use of complex hyper-surfaces obviously increases the computational load. These functions have already been introduced in collision avoidance methods [18,19] but were used in a radically different manner since they were contributing to the determination of artificial potential fields. We rather propose to use them for evaluating the proximity of an object to a robot arm.

To illustrate this idea, let us consider a solid whose volume is closely described by a surface equation:

$$S(X)=0. \tag{35}$$

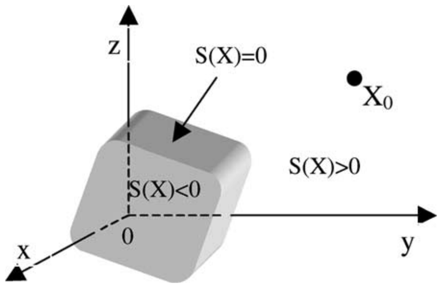


Fig. 7. The pseudo-distance.

At any point X_0 in the Cartesian space (Fig. 7), the surface function satisfies by definition the following properties:

$$\begin{aligned} &S(X_0)<0 \quad \text{if } X_0 \text{ is inside the solid},\\ &S(X_0)=0 \quad \text{if } X_0 \text{ is on its surface},\\ &S(X_0)>0 \quad \text{if } X_0 \text{ is anywhere else}. \end{aligned} \tag{36}$$

Hence, $S(X_0)$ is clearly related to the proximity of the point X_0 to the object. Proximity is then evaluated by applying the surface function at the desired point, with no need to compute the Euclidean distance and the result is a *pseudo-distance* on which the collision constraint will act, expressed as monotonous and increasing cost function of the distance.

Adapted to the collision avoidance problem, this concept is now used to express the constraint. The considered point X_0 is located on the robot link and is supposed not to enter the object volume augmented by a security margin d_s. This condition is equivalently written as

$$h(q)=-S(X_0)+d_s^2\le 0. \tag{37}$$

The remaining problem is to define the location of the point X_0 to be penalized on the robot link.

4.2. Pseudo-distance with respect to a link

A robot arm is made of several links. Reducing it to its skeleton, each robot link can be approximated by a straight line since the volume of the link is easily taken into account in increasing the security margin d_s. To formulate the anti-collision constraints $h(q)$ between this obstacle and one link of the robot arm, it is necessary to evaluate the point X_m on the link, closest to the obstacle. The search for the analytical expression of the closest point coordinates is the result of a one parameter optimization, which offers a unique solution since the surface function is convex.

Indeed, the coordinates (x, y, z) of any Cartesian point X belonging to the link $[M_1M_2]$ are given in the reference frame by the following expression:

$$\vec{OX} = \vec{OM_1} + \lambda \vec{M_1M_2}, \tag{38}$$

where λ is a constant coefficient $(0 \leq \lambda \leq 1)$. In order to simplify the following equations, distance $\overline{M_1M_2}$ will be noted U. The closest point of the robot arm X_m whose coordinates are expressed with the optimal parameter λ_m is obtained with the minimum pseudo-distance.

As an example, let us consider an obstacle whose shape can be modeled with an ellipsoid as represented in Fig. 8. This case is rather general since such a surface function can approximate a large number of objects when adjusting the semi-axes a, b, c of the ellipsoid. The surface is described in the base frame by the following equation:

$$X^\mathrm{T}QX + B^\mathrm{T}X + C = 0, \tag{39}$$

where matrix Q, vector B and scalar C are constant. As explained earlier, the pseudo-distance at any point X_0 in the Cartesian space is, in this case

$$S(X_0) = X_0^\mathrm{T}QX_0 + B^\mathrm{T}X_0 + C. \tag{40}$$

Given the expression of one point X on the robot link (38), the pseudo-distance at this point becomes

$$V(\lambda) = (\overline{OM_1} + \lambda U)^\mathrm{T} Q(\overline{OM_1} + \lambda U) + B^\mathrm{T}(\overline{OM_1} + \lambda U) + C, \tag{41}$$

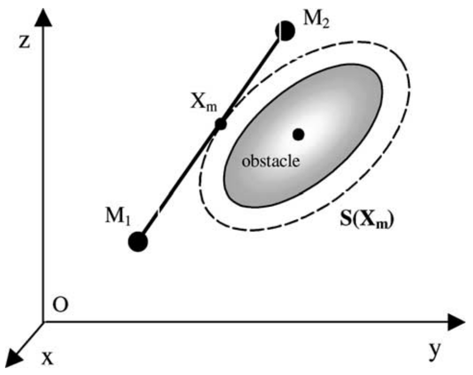


Fig. 8. Pseudo-distance between a link and an obstacle with ellipsoidal shape.

equivalently written in the form

$$V(\lambda) = \lambda^2 U^\mathrm{T}QU + \lambda U^\mathrm{T}\nabla_1 + V_1, \tag{42}$$

with

$$V_1 = \overline{OM_1}Q\overline{OM_1} + B\overline{OM_1} + C,$$
$$\nabla_1 = 2Q\overline{OM_1} + B.$$

The minimum pseudo-distance:

$$\frac{\mathrm{d}(V(\lambda))}{\mathrm{d}\lambda} = 0 = 2U^\mathrm{T}QU\lambda + U^\mathrm{T}\nabla_1, \tag{43}$$

leads to the optimal parameter:

$$\lambda_\mathrm{m} = -\frac{1}{2}\frac{U^\mathrm{T}\nabla_1}{U^\mathrm{T}QU}. \tag{44}$$

If $\lambda_\mathrm{m} < 0$ then λ_m is set to 0 so that X_m is located at M_1, whereas if $\lambda_\mathrm{m} > 1$ then λ_m is set to 1 so that X_m is located at M_2.

Now, replacing X_m in the pseudo-distance $S(X)$ gives a unique anti-collision constraint:

$$h(q) = -X_\mathrm{m}^\mathrm{T}QX_\mathrm{m} - B^\mathrm{T}X_\mathrm{m} - C + d_\mathrm{s}^2 \leq 0. \tag{45}$$

One should notice here that this optimization process is very simple since only one parameter concerning the robot link is unknown, the obstacle being completely determined by its surface equation while, on the contrary, the evaluation of the minimum Euclidean distance requires, in the case of a super-quadric surface function, the resolution of a fourth-order equation which is quite impossible to derive in an analytical form.

4.3. Set of chosen points

When the object surface equation does not lead to an easy solution, an alternative approach is the arbitrary choice of several points uniformly distributed on the arm link as shown in Fig. 9. Then, there is no need to find the minimum distance. The anti-collision constraint is rather expressed as a set of constraints depending on the various pseudo-distances defined at each chosen points:

$$h(q) = \begin{cases} -S(X_\mathrm{A}) + d_\mathrm{s}^2 \leq 0, \\ -S(X_\mathrm{B}) + d_\mathrm{s}^2 \leq 0, \\ -S(X_\mathrm{C}) + d_\mathrm{s}^2 \leq 0. \end{cases} \tag{46}$$

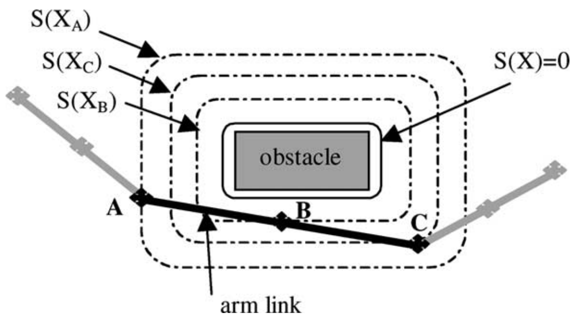


Fig. 9. Pseudo-distances at chosen points.

These constraints, as many as necessary, are all included in the augmented objective function via penalty functions. However, their number is a result of a compromise. If it is large, the computational load may be too heavy whereas conversely a collision might occur.

5. Mobile obstacle avoidance

When the robot arm is moving in a cluttered environment, its structure may unpredictably encounter an obstacle, itself either fixed or mobile. Hence, a vision system is often associated to an obstacle avoidance algorithm to detect the presence of the external body in the close area of the robot arm and to determinate its location and shape. Here, the vision system is not required to precisely supply every fine detail on the object surface but rather to give a rough description of the object shape in order to encompass it with a well-known super-quadric surface function $S(X)$. The vision system, however, helps to adjust the coefficients involved in $S(X)$ and to define the rotation and translation matrices necessary to describe the object in the reference frame since super-quadric functions are more naturally defined with respect to a frame linked to their principal axes. This first step is not part of the chosen collision avoidance algorithm but the performance of the latter is clearly linked to the ability of the vision system to operate on-line. When the object moves, the transformations involved must be updated each sampling time.

Next, anti-collision constraints are formulated using pseudo-distances to include them via penalty functions into an extended objective function as explained in the previous section and the collision avoidance is ensured by the appropriate choice of an arm configuration owing to the resolution of the redundancy in the inverse geometric model as illustrated in Fig. 10.

The pseudo-distance yields a very simple expression of the anti-collision constraint $h(q)$. Given the shape of the obstacle to be avoided, that is, given the super-quadric surface function surrounding it, it is very easy to compute in an analytical form the derivatives of $h(q)$ in order to determine the expression of the extended objective function $\varphi_e(q)$. This is a great advantage for the on-line implementation of the collision avoidance algorithm. Indeed, when the arm moves, only the new coordinates of the considered point X on the link are to be updated in the objective function. Moreover, when the obstacle is moving too, only few parameters are changed: the coefficients of the transform matrices to account in the super-quadric function for the new position and orientation of the object in space. Hence, such a formulation is a major contribution to the collision avoidance problem for both fixed and mobile obstacles.

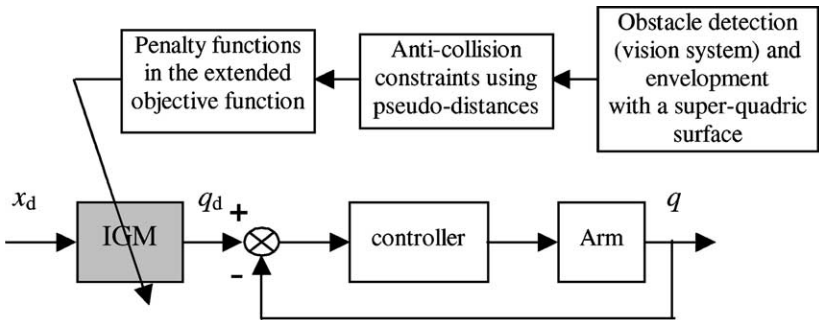

Penalty functions in the extended objective function
Anti-collision constraints using pseudo-distances
Obstacle detection (vision system) and envelopment with a super-quadric surface
x_d
IGM
q_d
+
-
controller
Arm
q

Fig. 10. The position control loop with obstacle avoidance.

Having completely described this new method for on-line solving the IGM of redundant robots in the presence of fixed or mobile obstacles, next section is devoted to test its performance during the execution of various tasks.

6. Simulation results

Our aim is not only to prove the fixed and mobile obstacle avoidance capability of the proposed method in path tracking but also to show its performance in terms of cycle time and cyclicity and evaluate the contribution of its backgrounds like penalty functions and pseudo-distance.

The simulations have been performed on a planar 4 degree of freedom robot with position control for path tracking. This robot is described in Appendix B, together with its parameters and direct models.

The position controller is implemented in joint space as justified earlier and represented in Fig. 10. The constraints are accounted for in the IGM by introducing them via penalty functions and weighting factors into the augmented cost function (25).

The proposed approach is a rather general method which can be applied to any objective function. Among the usual optimization criteria (minimizing energy, optimum manipulability, etc.) the minimum joint displacement between two configurations:

$$\varphi(q) = \frac{1}{2}\sum_{i=1}^{n}\mu_i(q_i(k) - q_i(k-1)), \tag{47}$$

where i designates one particular link of the n-links robot structure (here $n = 4$), μ_i a weighting coefficient and k is the current iteration, appears as the most adequate for on-line control in a variable environment, since it:

- can be adapted to any non-singular initial configuration,
- avoids sudden changes of arm configuration,
- allows to reduce joint torques by appropriate weighting of joints closer to the robot base.

During the estimation of the joint vector q for a specified Cartesian position, the algorithm stops when the Cartesian error $\varepsilon < \varepsilon_{\text{lim}}$ ($\varepsilon_{\text{lim}} = 10^{-3}, 10^{-6}$) or when the maximum number of iterations $N_{\max}$ is attained ($N_{\max} = 50$).

6.1. The penalty function

This section discusses the influence of various penalty functions on the collision avoidance. The task of the considered robot is to track a linear trajectory with a trapezoidal velocity profile in the presence of a circular obstacle (radius $r_0 = 0.4$ m), its center being located at $P_0 = (1, 1.4)^{\mathrm{T}}$. The object shape is chosen very simple so that it is easier to define the Euclidean distance between links and obstacle. This leads to only four constraints (one per link) and allows us to study the penalty functions independently on the approximation effect of the pseudo-distance. In order to avoid collision, the minimum distance d_{m} between one link and the object must remain less than a prescribed distance d_{p}:

$$d_{\text{m}} < d_{\text{p}}, \tag{48}$$

where $d_{\text{p}} = r_{\text{b}} + r_{\text{o}} + d_{\text{s}}$, r_{b} and r_{o} being the radius of the link and the object, respectively, and d_{s} the security margin. The minimum distance d_{m} is the gap between the closest points M_{m} and N_{m} on the link and the obstacle, respectively, as represented in Fig. 11:

$$d_{\text{m}} = \left\| \overrightarrow{M_{\text{m}}N_{\text{m}}} \right\|. \tag{49}$$

Hence, d_{m} is the solution of the norm minimization:

$$d^2 = (MN)^{\mathrm{T}}(MN), \tag{50}$$

where the location of the point M depends on an

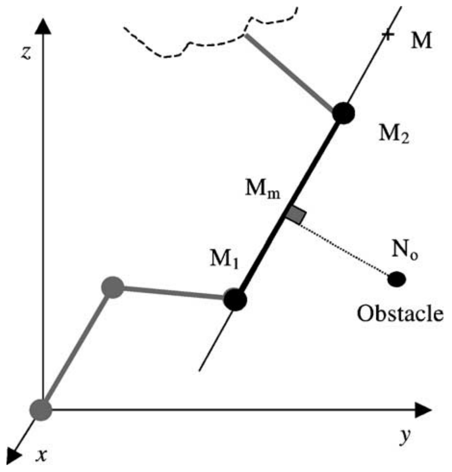


Fig. 11. The Euclidean distance with a circular obstacle.

unknown parameter as follows

$$\vec{OM} = \vec{OA} + \lambda \vec{AB}, \tag{51}$$

whereas the obstacle is fixed so that $N = N_0$. M_1 and M_2 are the extremities of the considered link. Eq. (48) is now equivalently written as

$$d^2 = (\overline{AN_0})^{\mathrm{T}} (\overline{AN_0}) - 2\lambda (\overline{AN_0})^{\mathrm{T}} \overline{AB} + \lambda^2 \overline{AB}^{\mathrm{T}} \overline{AB}, \tag{52}$$

and the optimal parameter is easily found as

$$\lambda_{\mathrm{m}} = \frac{(\overline{AN_0})^{\mathrm{T}} \overline{AB}}{\overline{AB}^{\mathrm{T}} \overline{AB}}. \tag{53}$$

If $\lambda_{\mathrm{m}} < 0$ or $\lambda_{\mathrm{m}} > 1$ then X_{m} is located at an link extremity, M_1 or M_2, respectively.

The minimum distance:

$$d_{\mathrm{m}} = d(\lambda_{\mathrm{m}}) \tag{54}$$

is introduced in the anti-collision constraint:

$$h(q) = -d_{\mathrm{m}}^2 + d_{\mathrm{p}}^2 < 0. \tag{55}$$

To adapt it to another link, one only needs to change the locations of the link extremities M_1 and M_2.

The weighting factors in the augmented cost function are adjusted off-line on a chosen set of trajectories. In fact, they must appropriately balance the effect of the anti-collision constraints and the criterion, avoiding that the formers become predominant since they describe a secondary task. These coefficients are chosen equal for all links so that they give them the same role ($\alpha = 100$ for the external penalty function and $\alpha = 0.006$ for the internal penalty function).

As expected, with *external penalization*, the workspace is maximized (Fig. 12) but the sudden obstacle avoidance when one link passes over the bound causes peak joint torques (Fig. 13) which may saturate the motors and perturb the control. The opposite occurs with *internal penalization* which reduces the workspace (Fig. 14) but results in lower torques far from the obstacle, increasing as the distance from the obstacle decreases (Fig. 15).

The best is to use the *limited internal penalization*: by bounding the obstacle zone of influence, the links located far from the obstacle moves freely, which increases the workspace (Fig. 16) and avoids the torque increase close to the obstacle (Fig. 17).

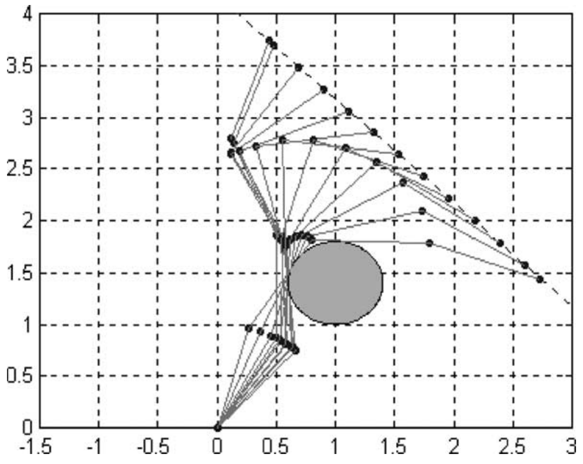


Fig. 12. Successive configurations obtained with external penalization.

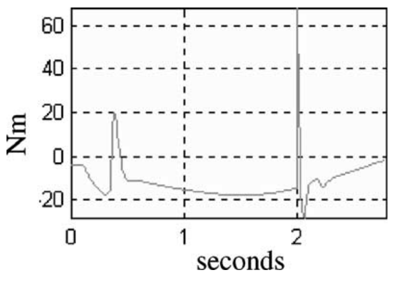


Fig. 13. Closest link joint torque with external penalization.

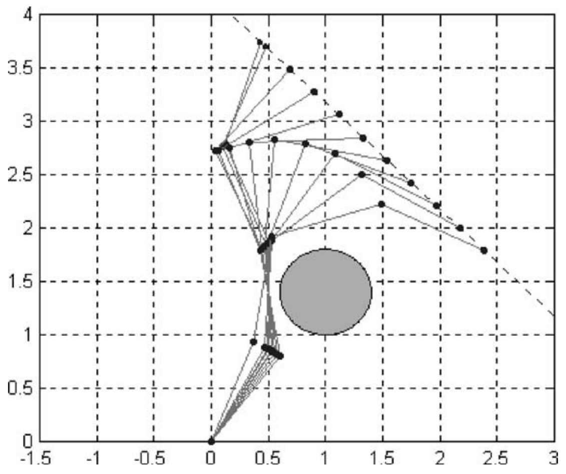


Fig. 14. Successive configurations obtained with internal penalization.

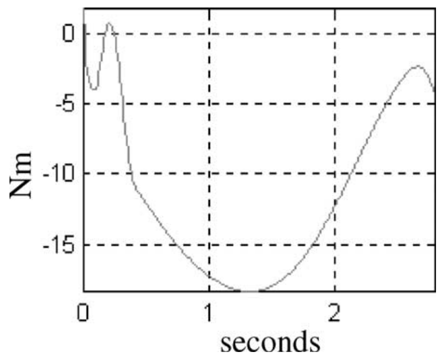


Fig. 15. Closest link joint torque with internal penalization.

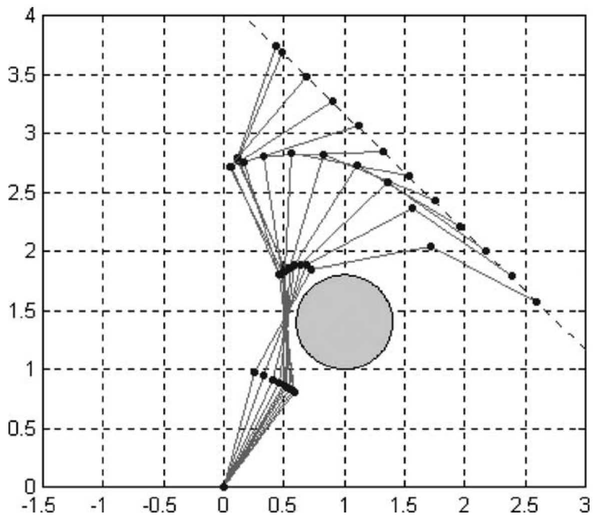


Fig. 16. Successive configurations obtained with limited internal penalization.

Irrespective of the penalization, the robot avoids smoothly the obstacle without much influence on the end-effector. This advantage is in fact implicit in the proposed method since it looks for the optimal configuration in the robot null-space containing joint motions that cause no end-effector motions. However, the Euclidean distance used in the simulations becomes complicated for complex object shapes. Two examples using the proposed pseudo-distance approach are now presented.

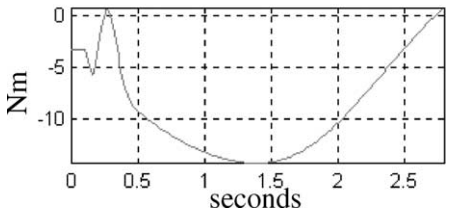


Fig. 17. Closest link joint torque with limited internal penalization.

6.2. *The pseudo-distance*

6.2.1. *Example 1*

The first example repeats the previous internal penalty example using the pseudo-distance. For this particular case of a circular obstacle, the closest point on the robot link is easily determined by introducing the optimal parameter (53) into (52). Only four constraints are then required to ensure the collision avoidance of the entire arm.

Each anti-collision constraint is written from the surface function:

$$S(X) = X^{\mathrm{T}}X - r_0^2, \tag{56}$$

as follows:

$$h(q) = -[\overline{OM_{\mathrm{m}}} - \overline{ON_0}]^{\mathrm{T}}[\overline{OM_{\mathrm{m}}} - \overline{ON_0}] + r_0^2 + d_{\mathrm{s}}^2 < 0, \tag{57}$$

where r_0 is the radius and N_0 the center of the obstacle.

With a weighting factor $\alpha = 2 \times 10^{-4}$, the pseudo-distance approach leads to similar results compared to the previous ones obtained with the Euclidean distance, which confirms the validity of the method (Fig. 18).

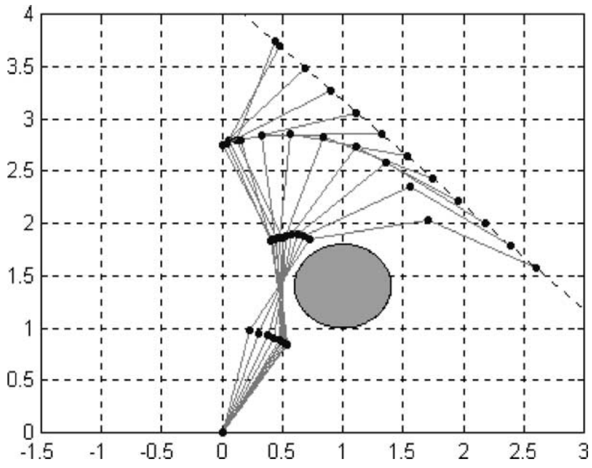


Fig. 18. Successive configurations obtained with internal penalization using pseudo-distance.

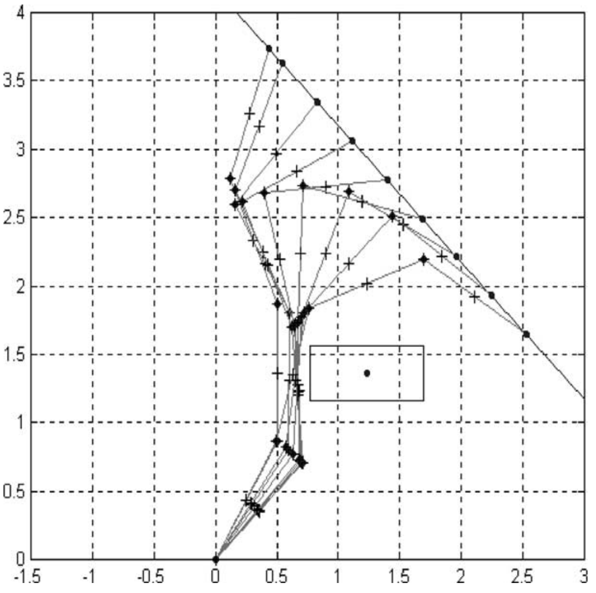


Fig. 19. Configurations for avoiding a rectangle obstacle.

6.2.2. *Example 2*

The second example deals with a rectangular obstacle so that the point X_m on each link cannot be easily determined as a result of an optimization problem but is arbitrarily chosen from three equidistant points. The number of constraints is now increased to seven

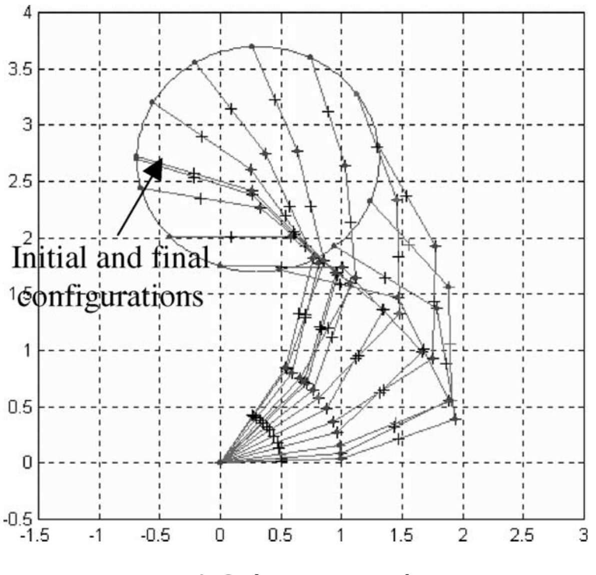


Fig. 20. Cyclic trajectory tracking.

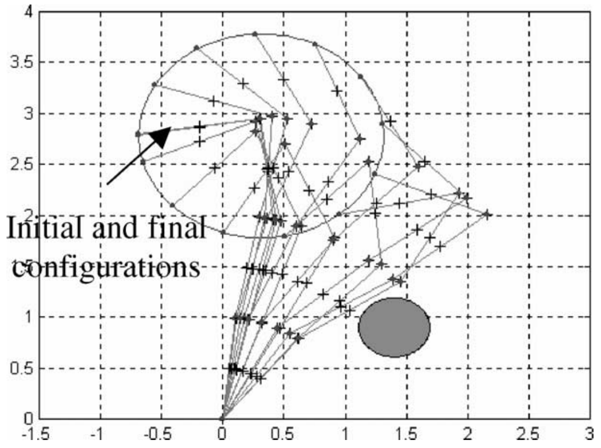


Fig. 21. Cyclic trajectory tracking together with obstacle avoidance.

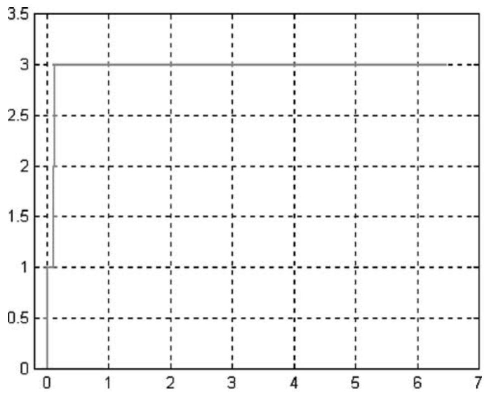


Fig. 22. Number of iterations at each point on the trajectory.

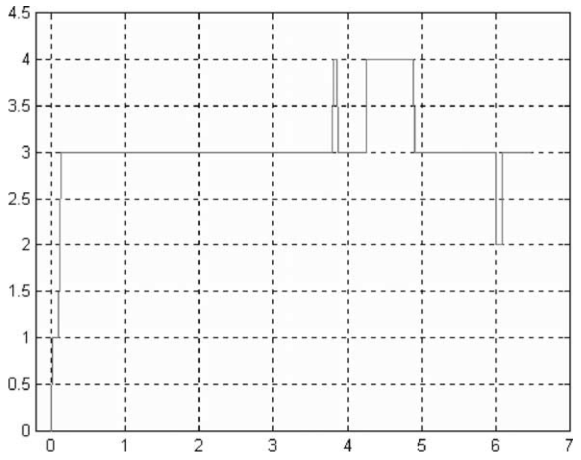


Fig. 23. Number of iterations at each point on the trajectory with obstacle avoidance.

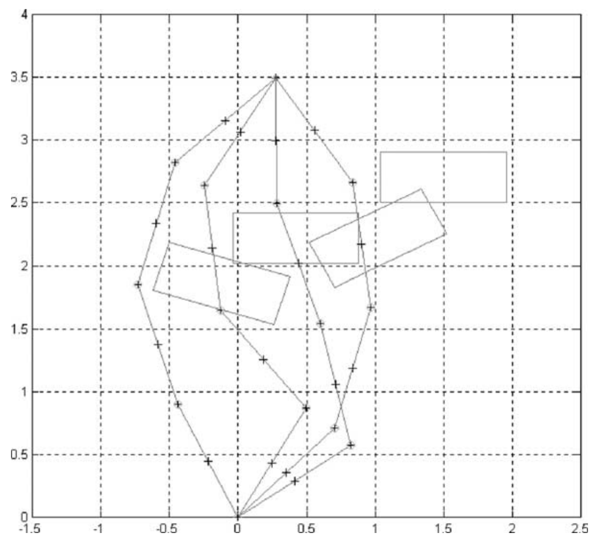


Fig. 24. Mobile obstacle avoidance.

as shown in Fig. 19: three are written at the joints "·", three at the middle of each link "+" and the last one at the end-effector. For a rectangular obstacle which length and width are $2a = 0.92$ m and $2b = 0.4$ m, respectively, each anti-collision constraint is given by

$$h_i(q) = -\left(\frac{x_i(q)}{a}\right)^8 - \left(\frac{y_i(q)}{b}\right)^8 + 1 < 0, \qquad (58)$$

where (x_i, y_i) are the coordinates of the considered point. The weighting coefficient for the limited internal penalization is adjusted at 4.8×10^{-4} and the security margin is equal to 0.2 m. The solution is easily reached (Fig. 19), showing the efficiency of the pseudo-distance approach based on obstacle modeling using surface function.

6.3. Algorithm properties

The following simulations aim to evaluate the number of iterations and the computing time required by the proposed method with and without obstacles. At the same time, the repeatability property is also verified, the end-effector tracking a circular cyclic trajectory at a 0.8 m/s velocity. The results prove that repeatability is preserved in both cases (Figs. 20 and 21, respectively) with a low number of iterations (Figs. 22 and 23) slightly higher in the presence of an obstacle. The computing time is about 0.15 ms between two points on the trajectory.

6.4. Mobile obstacle avoidance

The capability of the proposed approach to help the arm to avoid mobile obstacles is now demonstrated. First, the robot with its end-effector kept fixed reacts to a translating and rotating rectangle, showing that

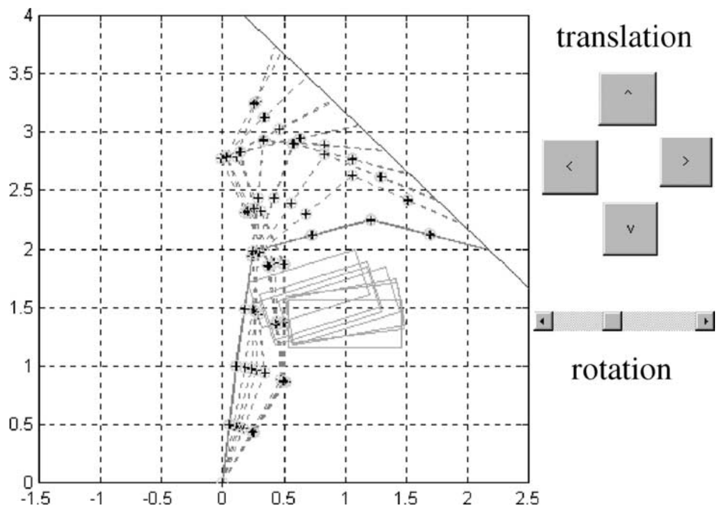


Fig. 25. Mobile obstacle avoidance together with trajectory tracking.

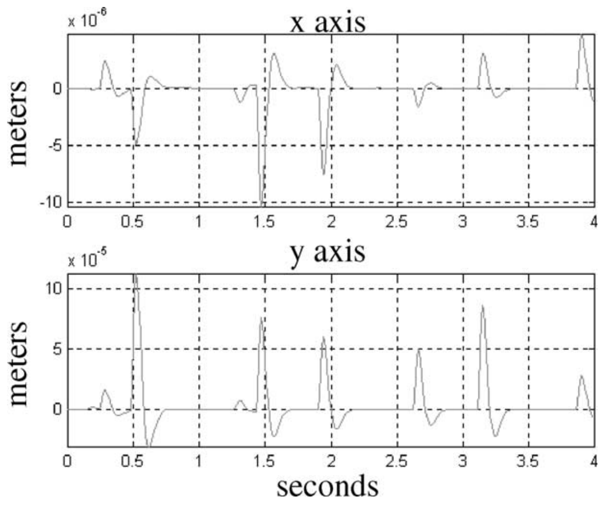


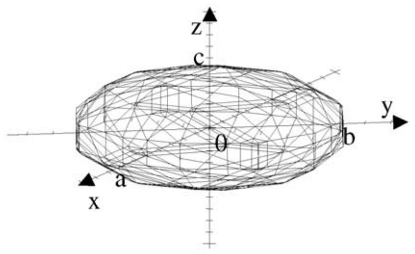
Fig. 26. End-effector position errors during the trajectory.

the algorithm succeeds to modify the robot configuration as the obstacle approaches (Fig. 24). The steady state error of the end-effector position remains lower than 10^{-4} m. Next, the robot is required to track a linear path in the presence of the same moving obstacle, which is also achieved with small errors (Figs. 25 and 26).

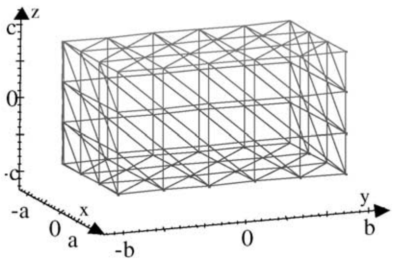
7. Conclusion

This paper has presented an efficient algorithm for the obstacle avoidance of a manipulator moving in variable environment. The collision avoidance problem is treated efficiently by solving on-line the kinematic redundancy of the arm owing to an iterative solution of the inverse geometric model. This algorithm makes possible for several constraints, that is for several obstacles, to be simultaneously taken into account via penalty functions. Each obstacle is encompassed with a super-quadric surface whose analytical expression is well-known so that, instead of determining a minimum Euclidean distance, an original concept of pseudo-distance is introduced leading to a formulation of the anti-collision constraints which is not only simple but also easily updated to account on-line for the arm and/or obstacles motion.

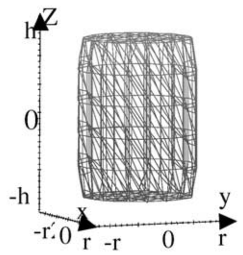
Appendix A. Super-quadric surface functions



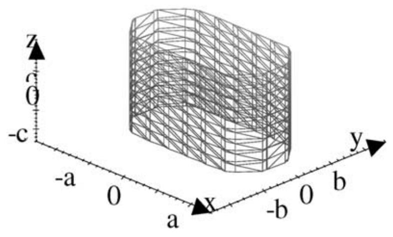
$$S(x,y,z)=\left(\frac{x}{a}\right)^2+\left(\frac{y}{b}\right)^2+\left(\frac{z}{c}\right)^2-1.$$



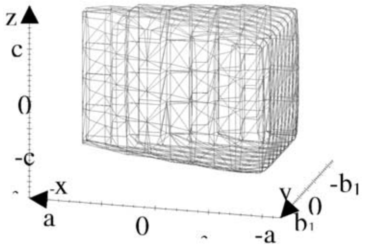
$$S(x,y,z)=\left(\frac{x}{a}\right)^{2n}+\left(\frac{y}{b}\right)^{2n}+\left(\frac{z}{c}\right)^{2n}-1.$$



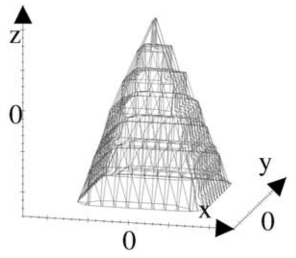
$$S(x,y,z)=\left(\frac{x}{r}\right)^2+\left(\frac{y}{r}\right)^2+\left(\frac{z}{h}\right)^{2n}-1.$$



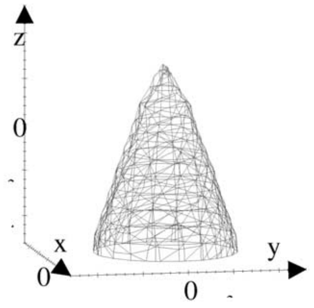
$$S(x,y,z)=\left(\frac{x}{a}\right)^{2l}+\left(\frac{y}{b}\right)^{2m}+\left(\frac{z}{c}\right)^{2n}-1.$$



$$S(x,y,z)=\left(\frac{x}{a}\right)^{2n}+\left(\frac{y}{mx+d}\right)^{2n}+\left(\frac{z}{c}\right)^{2n}-1.$$



$$S(x,y,z)=\left(\frac{x}{m_1z+d_1}\right)^{2n}+\left(\frac{y}{m_2z+d_2}\right)^{2n}+\left(\frac{z}{c}\right)^{2n}-1.$$



$$S(x,y,z)\mapsto \left(\frac{x}{mz+d}\right)^{2}+\left(\frac{y}{mz+d}\right)^{2}+\left(\frac{z}{c}\right)^{2n}-1=0.$$

Appendix B. Planar four degree of freedom robot

(1) Geometric model

$$\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} l_1\cos(q_1)+l_2\cos(q_1+q_2) \\ \quad +l_3\cos(q_1+q_2+q_3) \\ \quad +l_4\cos(q_1+q_2+q_3+q_4) \\ l_1\sin(q_1)+l_2\sin(q_1+q_2) \\ \quad +l_3\sin(q_1+q_2+q_3) \\ \quad +l_4\sin(q_1+q_2+q_3+q_4) \end{bmatrix}.$$

(2) Kinematic model

$$\begin{bmatrix} \dot{x} \\ \dot{y} \end{bmatrix} = \begin{bmatrix} J_{11} & J_{12} & J_{13} & J_{14} \\ J_{21} & J_{22} & J_{23} & J_{24} \end{bmatrix} \begin{bmatrix} \dot{q}_1 \\ \dot{q}_2 \\ \dot{q}_3 \\ \dot{q}_4 \end{bmatrix},$$

with

$$\begin{aligned}
J_{11} &= -l_1\sin(q_1)-l_2\sin(q_1+q_2) \\ &\quad -l_3\sin(q_1+q_2+q_3) \\ &\quad -l_4\sin(q_1+q_2+q_3+q_4), \\
J_{12} &= -l_2\sin(q_1+q_2)-l_3\sin(q_1+q_2+q_3) \\ &\quad -l_4\sin(q_1+q_2+q_3+q_4), \\
J_{13} &= -l_3\sin(q_1+q_2+q_3) \\ &\quad -l_4\sin(q_1+q_2+q_3+q_4), \\
J_{14} &= -l_4\sin(q_1+q_2+q_3+q_4), \\
J_{21} &= l_1\cos(q_1)+l_2\cos(q_1+q_2) \\ &\quad +l_3\cos(q_1+q_2+q_3) \\ &\quad +l_4\cos(q_1+q_2+q_3+q_4), \\
J_{22} &= l_2\cos(q_1+q_2)+l_3\cos(q_1+q_2+q_3) \\ &\quad +l_4\cos(q_1+q_2+q_3+q_4), \\
J_{23} &= l_3\cos(q_1+q_2+q_3) \\ &\quad +l_4\cos(q_1+q_2+q_3+q_4), \\
J_{24} &= l_4\cos(q_1+q_2+q_3+q_4).
\end{aligned}$$

(3) Null space matrix

$$N^{\mathrm{T}} = \begin{bmatrix} N_{11} & N_{12} & N_{13} & N_{14} \\ N_{21} & N_{22} & N_{23} & N_{24} \end{bmatrix},$$

with

$$\begin{aligned}
N_{11} &= -l_2 l_3 \sin(q_3) - l_2 l_4 \sin(q_3+q_4),\\
N_{12} &= l_1 l_3 \sin(q_2+q_3) + l_2 l_3 \sin(q_3)\\
&\quad + l_1 l_4 \sin(q_2+q_3+q_4)\\
&\quad + l_2 l_4 \sin(q_3+q_4),\\
N_{13} &= -l_1 l_2 \sin(q_2) - l_2 l_3 \sin(q_2+q_3)\\
&\quad - l_1 l_4 \sin(q_2+q_3+q_4),\\
N_{14} &= 0,\\
N_{21} &= -l_2 l_4 \sin(q_3+q_4) - l_3 l_4 \sin(q_4),\\
N_{22} &= l_1 l_4 \sin(q_2+q_3+q_4) + l_2 l_4 \sin(q_3+q_4)\\
&\quad + l_3 l_4 \sin(q_4),\\
N_{23} &= 0,\\
N_{24} &= -l_1 l_2 \sin(q_2) - l_1 l_3 \sin(q_2+q_3)\\
&\quad - l_1 l_4 \sin(q_2+q_3+q_4).
\end{aligned}$$

References

[1] J. Ballieul, Kinematic programming alternatives for redundant manipulators, in: Proceedings of the IEEE International Conference on Robotics and Automation, St Louis, MO, March 1985, pp. 722–728.

[2] J. Baillieul, A constraint oriented approach to inverse problems for kinematically redundant manipulators, in: Proceedings of the IEEE International Conference on Robotics and Automation, Raleigh, NC, 1987, pp. 1827–1833.

[3] P.H. Chang, A closed-form solution for the control of manipulators with kinematics redundancy, in: Proceedings of the IEEE International Conference on Robotics and Automation, San Francisco, CA, April 1986, pp. 722–728.

[4] F.-T. Cheng, T.-H. Chen, Y.-S. Wang, Y.-Y. Sun, Obstacle avoidance for redundant manipulators using the compact QP method, in: Proceedings of the IEEE International Conference on Robotics and Automation, Atlanta, GA, May 1993, pp. 262–269.

[5] C. Chevallereau, W. Khalil, A new method for the solution of the inverse kinematics of redundant robots, in: Proceedings of the IEEE International Conference on Robotics and Automation, Philadelphia, PA, April 1988, pp. 37–42.

[6] P. Chiacchio, S. Chiaverini, L. Sciavicco, B. Siciliano, Closed-loop inverse kinematics schemes for constrained redundant manipulators with task space augmentation and task priority strategy, International Journal of Robotics Research 10 (4) (1991) 410–426.

[7] G.S. Chirikjian, J.W. Burdick, An obstacle avoidance algorithm for hyper-redundant manipulators, in: Proceedings of the IEEE International Conference on Robotics and Automation, Cincinnati, OH, May 1990, pp. 625–631.

[8] S.L. Chiu, Control of redundant manipulators for task compatibility, in: Proceedings of the IEEE International Conference on Robotics and Automation, Raleigh, NC, 1987, pp. 1718–1724.

[9] A. De Luca, G. Oriolo, The reduced gradient method for solving redundancy in robot arms, Robotersysteme 7 (2) (1991) 112–117.

[10] J.E. Dennis, Numerical methods for unconstrained optimization and non-linear equations, Society for Industrial and Applied Mathematics, Philadelphia, PA, 1996.

[11] A.S. Deo, I.D. Walker, Adaptive non-linear least squares for inverse kinematics, in: Proceedings of the IEEE International Conference on Robotics and Automation, Atlanta, GA, May 1993, pp. 186–193.

[12] B. Faverjon, Obstacle avoidance using an octree in the configuration space of a manipulator, in: Proceedings of the IEEE International Conference on Robotics and Automation, Atlanta, GA, March 1984.

[13] B. Faverjon, P. Tournassoud, A local based approach for path planning of manipulators with a high number of degrees of freedom, in: Proceedings of the IEEE International Conference on Robotics and Automation, Raleigh, NC, 1987, pp. 1152–1159.

[14] R. Featherstone, Accurate trajectory transformations for redundant and non-redundant robots, in: Proceedings of the IEEE International Conference on Robotics and Automation, San Diego, CA, May 1994, pp. 8–13.

[15] E.G. Gilbert, D.W. Johnson, Distance functions and their application to robot path planning in the presence of obstacles, IEEE Journal of Robotics and Automation 1 (1) (1985) 21–30.

[16] A. Hemami, Studies on a light weight and flexible robot manipulator, Robotics 1 (1985) 27–36.

[17] S. Hirose, Y. Umetani, Kinematic control of active cord mechanism with tactile sensors, in: Proceedings of the Second International CISM-IFT Symposium on Theory and Practice of Robots and Manipulators, 1976, pp. 241–252.

[18] O. Khatib, Real-time obstacle avoidance for manipulators and mobile robots, International Journal of Robotics Research 5 (1) (1986) 90–99.

[19] P. Khosla, R. Volpe, Superquadric artificial potentials for obstacle avoidance and approach, in: Proceedings of the IEEE International Conference on Robotics and Automation, Philadelphia, PA, April 1988, pp. 1778–1784.

[20] J.O. Kim, P. Khosla, Real-time obstacle avoidance using harmonic potential functions, IEEE Transactions on Robotics and Automation 8 (3) (1992) 338–349.

[21] A. Liégeois, Automatic supervisory control of the configuration and behavior of multi-body mechanisms, IEEE Transactions on Systems Man and Cybernetics SMC-9 (1977) 245–250.

[22] T. Lozano-Pérez, Spatial planning: a configuration space approach, IEEE Transactions on Computers C-32 (2) (1983) 108–120.

[23] S. Ma, M. Konno, An obstacle avoidance scheme for hyper-redundant manipulators: global motion planning in posture space, in: Proceedings of the IEEE International Conference on Robotics and Automation, Albuquerque, NM, April 1997, pp. 161–166.

[24] A. Mohri, M. Yamamoto, G. Hirano, Cooperation path planning for two manipulators, in: Proceedings of the IEEE International Conference on Robotics and Automation, Minneapolis, MN, April 1996, pp. 2853–2858.

[25] D. Nenchev, Restricted Jacobian matrices of redundant manipulators in constrained motion task, International Journal on Robotics Research 11 (1992) 584–597.

[26] C. O'Dùnlaing, C.K. Yap, A retraction method for planning the motion of a disk, Journal of Algorithms 6 (1982) 104–111.

[27] V. Perdereau, M. Drouin, About kinematic instability and stabilization of hybrid control scheme, in: Proceedings of the International Symposium on Robotics and Manufacturing: Research, Education and Applications, Maui, HI, August 1994, pp. 525–530.

[28] A. Ramdane-Cherif, V. Perdereau, M. Drouin, Penalty approach for a constrained optimization to solve on-line the inverse kinematic problem of redundant manipulators, in: Proceedings of the IEEE International Conference on Robotics and Automation, Minneapolis, MN, April 1996, pp. 133–138.

[29] A. Ramdane-Cherif, D.Y. Meddah, V. Perdereau, M. Drouin, Inverse kinematic solution based on Lyapunov function for redundant and non-redundant robots, in: Proceedings of the IEEE International Symposium on Computational Intelligence in Robotics and Automation, Monterey, CA, July 1997, pp. 226–231.

[30] H. Seraji, R. Colbaugh, Improved configuration control for redundant robots, Journal of Robotic Systems 7 (6) (1990) 897–928.

[31] H. Seraji, B. Bon, Real-time collision avoidance for position-controlled manipulators, IEEE Transactions on Robotics and Automation 15 (4) (1999) 670–677.

[32] M. Sharir, A. Schorr, On shortest paths in polyhedral spaces, in: Proceedings of the 16th ACM Symposium on Theory of Computing, Washington, DC, 1984, pp. 144–153.

[33] Z. Shiller, Interactive time optimal robot motion planning and work-cell layout design, in: Proceedings of the IEEE International Conference on Robotics and Automation, Scottsdale, AZ, May 1989, pp. 964–969.

[34] J.J.E. Slotine, D.R. Yoerger, A rule-based inverse kinematics algorithm for redundant manipulators, International Journal of Robotics and Automation 2 (2) (1987) 86–89.

[35] D. Wang, Y. Hamam, Optimal trajectory planning of manipulators with collision detection and avoidance, International Journal of Robotics Research 11 (5) (1992) 460–468.

[36] D.E. Whitney, Resolved motion rate control of manipulators and human prostheses, IEEE Transactions on Man–Machine Systems MMS-10 (2) (1969) 47–53.

[37] W.A. Wolovich, H. Elliott, A computational technique for inverse kinematics, in: Proceedings of the Third IEEE Conference on Decision and Control, Las Vegas, NV, December 1984, pp. 1359–1363.

[38] T. Yoshikawa, Analysis and control of robot manipulators with redundancy, in: Proceedings of the First International Symposium of Robotics Research, Bretton Woods, NH, August 1983.

[39] T. Yoshikawa, Analysis and control of robot manipulators with redundancy, in: Proceedings of the First International Symposium, MIT Press, Cambridge, MA, 1984, pp. 735–748.

[40] T. Yoshikawa, Manipulability of robotic mechanisms, in: Proceedings of the Second International Symposium of Robotics Research, Kyoto, August 1984, pp. 91–98.

V. Perdereau received the Doctorate degree in Automatic Control from the University of Paris VI, France, in February 1991. She is currently an Associate Professor at the same university where she has been giving courses since December 1987 in automatic control, electronics, robot control, neural networks and fuzzy logic. Her research interests include hybrid position/force control of robot manipulators, multi-robot cooperation, multi-fingered hand control, inverse kinematics and redundant manipulator control.

C. Passi obtained the Master of Science degree from the University of Paris XII, France, in 1995 and received the Ph.D. degree in Robotics from the University of Paris VI, France, in July 2001.

M. Drouin is currently Full Professor at the University of Paris VI, France, in the Electronics and Automatic Control Department. From 1975 to 1983, he has been at the Laboratory of Signals and Systems where he worked on hierarchical and complex system control. From 1983 to 1993, he has been a member of the Laboratory of Robotics and since 1993, he is at the Laboratory of Instruments and Systems. His research interests are the hybrid position/force control of robotic arms, the redundant robot control including mobile obstacle avoidance and the cooperation of multi-robots. He is now working on the control of multi-fingered robotic hands for handling and fine manipulation.