Iterative strategies for obstacle avoidance of a redundant manipulator

CORNEL SECARĂ, LUIGI VLĂDĂREANU
Robotics-Mechatronics Group
Institute of Solid Mechanics of Romanian Academy
C-tin Mille 15, 010141 Bucharest
ROMANIA
corsecus@yahoo.com   http://www.imsar.ro

Abstract: - This paper presents four different iterative strategies for obstacle avoidance of a redundant manipulator. The end-effector task consists in generating the references along the contour of a curve. The proposed strategies are iterative in the sense that the joint configuration computed in the previous step represents the current point around which the methods provide the next joint configuration corresponding to the imposed end-effector posture. The objective of the strategies is to simultaneously minimize the end-effector location error and the manipulator total joint displacement while the collision with the obstacle is avoided. The strategies are implemented using the Matlab software and the comparative simulation results are obtained for a planar redundant manipulator with four degrees of freedom and with its end-effector following the contour of a circle, whose surface is considered to be restrictive for all elements of the manipulator.

Key-Words: - redundancy resolution, iterative strategy, obstacle avoidance, multiple criteria.

1 Introduction

A redundant manipulator has more joints than necessary for achieving the end-effector task. The human body, composed of several redundant parts like arms and legs, is a proper example of redundancy provided by the nature [1].

Kinematic redundancy has become increasingly popular in robotics field through the attempts to improve the overall performance of robots in a large variety of tasks. The extra *degrees of freedom* (DOF) offered by redundancy can be used to optimize additional performance criteria while solving the main end-effector task. These performance criteria can be defined in terms of the kinematic or dynamic parameters and can be related to the different aspects of performance. Numerous studies have revealed the importance of using performance criteria for performance enhancement [2-4]. Obstacle avoidance is one of the most important domains of redundant manipulators application because of the incapacity of non-redundant structures to avoid collisions with workspace obstacles [5-8,22].

Introduction of additional criteria changes the redundancy problem from a known inverse kinematics problem to a non-linear optimization problem with non-linear constraints. Solving the inverse kinematics problem of redundant robots is not common since the necessary mapping from the task coordinates to the joint coordinates is not unique, and yields an infinite number of solutions.

This paper presents four different iterative strategies for obstacle avoidance of a redundant manipulator. The first three strategies are based on linearized solutions and the fourth is based on a non-linear optimization technique using genetic algorithms. The first two methods are based on the *Gradient Projection Method* (GPM). The constraint functions, based on the *Maximum Distance Criterion* (MXDC) and the *Repulsive Potential Field* (RPF), respectively, are added in the null space of the Jacobian matrix and are used for local optimisation purposes. The third approach is based on the *Extended Jacobian Matrix* (EJM), which augments the manipulator forward kinematics with a set of kinematic functions in operational space or in joint space reflecting the desired additional task. The desired additional task is to minimize an objective function, which is, in our case, the sum of inverses of the distances between the obstacle and the *Configuration Control Points* (CCP) situated on the elements of the manipulator.

A genetic algorithm (GA) based strategy for redundancy resolution with two performance criteria accomplishment, while the end-effector achieves a number of prescribed configurations, was developed in a previous paper [9,22] of the authors. The additional constraints added to the main end-effector task were the same with those introduced in this paper. The disadvantage of the previous strategy was the reduced number of the end-effector imposed configurations (five, in the simulation results). Instead, the present fourth iterative strategy offers the possibility to accomplish an end-effector prescribed path following. The problem is formulated as a constrained optimization problem and solved using an iterative GA based strategy. The objective of this optimization problem is to simultaneously minimize the end-effector location error and the manipulator total joint displacement while the collision with the obstacle is avoided.

A short review of the redundancy resolution methods is presented in second section. The third section described the two methods with linearized solutions. First subsection deals with the Gradient Projection Method with two performance criteria: Maximum Distance Criteria and Repulsive Potential field, respectively. The Extended Jacobian Method is presented in the second subsection. The fourth section presents the genetic algorithms based strategy. First subsection describes a brief overview of the concept of GA. In second subsection the proposed GA iterative strategy is characterized through its variables, fitness function and non-linear constraint functions. The comparative simulation results are obtained for a planar redundant manipulator with four DOF and with its end-effector following the contour of a circle, whose surface is considered to be restrictive for all elements of the manipulator. These simulations are presented in fifth section and sixth section contains the conclusions of this paper.

2 Theoretical background

Redundant manipulators possess at least one DOF more than necessary for their imposed end-effector location. In order to place the end-effector of a redundant manipulator on a desired location (position and orientation), a proper configuration of the manipulator must be specified, i.e. the suitable values of the joint angles which place the end-effector to the given location must be computed. This is the well-known inverse kinematics problem.

Mapping from the world coordinates to the joint coordinates for a redundant robot is not unique, meaning that there are an infinite number of joint angles settings which results for a given end-effector location.

The direct geometric model gives the relation between the end-effector configuration vector **x** and the joint coordinates (angles vector **θ**, for rotation joint case):

$$\begin{aligned} &\mathbf{x} = f(\boldsymbol{\theta}); \\ &\mathbf{x} = [\, x_1 \; x_2 \; \ldots \; x_m \,]^{\mathrm{T}}; \\ &\boldsymbol{\theta} = [\, \theta_1 \; \theta_2 \; \ldots \; \theta_n \,]^{\mathrm{T}}, \end{aligned} \tag{1}$$

where n is the number of DOF and m is the workspace dimension.

In inverse kinematics, which is necessary for robot control, the desired posture of the end-effector is given by the user. Inverse kinematics resolution requires the computation of all joint angles in the chain that place the end-effector in the imposed configuration. In order to derive **θ** with a given **x**, the inverse of the equation (1) is required:

$$\boldsymbol{\theta} = f^{-1}(\mathbf{x}) \, . \tag{2}$$

Solving equation (2) is quite difficult since is non-linear, involving rotations at the joint transformations. The redundancy resolution approaches can be classified into two categories: methods with linearized solutions and non-linear optimization methods.

In the first approach, because the non-linear property of function makes the solution difficult, the problem can be made linear by localizing around the current operating position. As a first step, equation (1) is differentiated with respect to **θ,** obtaining the inverse differential model:

$$\delta\mathbf{x} = \mathbf{J}(\theta)\,\delta\boldsymbol{\theta} \, , \tag{3}$$

where $\mathbf{J}(\theta) = \dfrac{\delta f(\theta)}{\delta\theta}$ is the manipulator Jacobian matrix.

If we invert equation (3) and iterate towards a final goal configuration with incremental steps, the inverse kinematic problem can be linearly solved. For non-redundant manipulator structures, $n = m$, the inverse differential model is simple obtained using the inverse of the Jacobian matrix:

$$\delta\boldsymbol{\theta} = \mathbf{J}^{-1}\delta\mathbf{x} \, . \tag{4}$$

Unfortunately, in case of redundancy, the Jacobian is not a square matrix, i.e. $n > m$. In this situation, instead of $\mathbf{J}^{-1}$, a pseudoinverse of Jacobian is used being defined as:

$$\mathbf{J}^{+} = \mathbf{J}^{\mathrm{T}}(\mathbf{J}\mathbf{J}^{\mathrm{T}})^{-1} \, . \tag{5}$$

However, the simple use of this $n \times n$ square matrix $\mathbf{J}^{+}$ is not sufficient since this right-hand side term of the equation (4) represents just the least norm solution, which guarantees only the end-effector task accomplishment and a minimization of the sum of joint displacements, but not the additional obstacle avoidance constraint.

To deal with the obstacle avoidance additional constraint as well, two basic methods with linearized solutions can be distinguished in redundancy literature [5-8,19]: the *Gradient Projection Method* and *Extended Jacobian Method*.

GPM obtains the solution of the non-linear optimization problem with non-linear constraints, by adding to the least norm solution $\mathbf{J}^{+}\delta\mathbf{x}$ (which ensures that the end-effector follows the task path), the so-called null space solution, which takes into account additional constraints and consists of a self-motion of the links in the joint space only, without any effect on the end-effector task configuration vector.

With respect to the EJM, this second method defines $n-m$ additional constraints including the obstacle avoidance for the given task, thus the relationship between the joint space and the end-effector space becomes non-redundant, and the extended Jacobian is now a square $n\times n$ square matrix which can be easily inverted.

The secondary redundancy resolution approaches are non-linear optimization techniques and treats the problem as a minimization problem. Let $e(\theta)$ be the positional and orientation definition of end-effector depending of joint angles and g be the positional and orientation definition at the desired goal.

$$P(e(\theta)) = (g - e(\theta))^2, \tag{6}$$

where $P(e(\theta))$ is a potential function that gives the error between the end-effector and the goal. If the value of the potential function is zero, then the goal is reached. If the goal is not reachable because of the joint limits, the potential function value is tried to be minimized as much as possible. The optimization problem can be formulated as follows:

$$\begin{array}{ll} \text{Minimize} & P(e(\theta)) = (g - e(\theta))^2, \\ \text{subject to} & h(\theta) \le 0, \quad l_i \le \theta \le u_i, \ \text{for } i = 1 \div n \end{array} \tag{7}$$

where $h(\theta)$ includes the non-linear constraints, l_i and u_i are the lower and upper limits of the joint angles, respectively.

The most known non-linear optimization methods for redundancy resolution are based on genetic algorithm [9-11], direct search [12], neural networks [13], or fuzzy techniques [14].

Recent approaches have tried to solve the redundancy problem by combining a method based on inverse kinematics with one based on the direct geometric model. Such combined methods can avoid, for example, the joint angle drift problem caused by the disadvantage that the pseudoinverse control is not repeatable. Da Graça Marcos *et al.* [15] proposes a technique that combines the closed-loop pseudoinverse method with genetic algorithms. An open-loop genetic algorithm based on the direct geometric model is used to find the initial joint configuration, and also to compare the results provided by the new combined method.

3 Methods with linearized solutions

There are two basic methods with linearized solutions, used for redundancy resolution: *Gradient Projection Method* and *Jacobian Extended Method*.

3.1 Gradient Projection Method

The *Gradient Projection Method* was firstly introduced by Liegeois [16] with the purpose of using the redundancy to avoid mechanical joint limits. Extending the pseudoinverse solution, a general solution to the inverse kinematics problem can be expressed as:

$$\delta\boldsymbol{\theta} = \mathbf{J}^{+}\delta\mathbf{x} + (\mathbf{I}\text{-}\mathbf{J}^{+}\mathbf{J})\mathbf{z}\,, \tag{8}$$

where:

- $\delta\boldsymbol{\theta}$ is the joint angular differential variation, $\delta\boldsymbol{\theta} \in \Re^{n\times 1}$;
- $\mathbf{J}^{+} = \mathbf{J}^{\mathrm{T}}\left(\mathbf{J}\,\mathbf{J}^{\mathrm{T}}\right)^{-1}$ is the Moore-Penrose pseudoinverse of manipulator Jacobian matrix, $\mathbf{J}^{+} \in \Re^{n\times m}$, $\mathbf{J} \in \Re^{m\times n}$;
- $\delta\mathbf{x}$ is the differential variation of end-effector posture (computed in closed-loop).

$\delta\mathbf{x}(k) = \mathbf{x}(k) - \mathbf{x}(k-1)$, with:
$\mathbf{x}(k)$ - imposed actual posture;
$\mathbf{x}(k-1)$ - previous achieved configuration.
$\delta\mathbf{x} \in \Re^{m\times 1}$;

- $\mathbf{I} - \mathbf{J}^{+}\,\mathbf{J} = \mathbf{P}$ is the “projector” matrix, $\mathbf{P} \in \Re^{n\times n}$;
- $\mathbf{z}$ is an arbitrary vector, $\mathbf{z} \in \Re^{n\times 1}$.

The first term on the right of equation (8) is the least norm solution. The second term is the homogeneous (null-space) solution, which is orthogonal to the first term. The homogeneous solution is called the self-motion of the manipulator and produces no end-effector motion. For a desired end-effector trajectory, a homogeneous solution is selected such that the resulting robot configuration optimizes a performance measure.

To optimise a performance criterion $F(\theta)$, $\mathbf{z}$ is chosen to be:

$$\mathbf{z} = \pm\psi\nabla F(\theta)\,, \tag{9}$$

where ψ is a positive real number and $\nabla F(\theta)$ is the gradient of $F(\theta)$. A positive sign in equation (9) indicates that the criterion is to be maximized, while a negative sign indicates minimization.

3.1.1 Maximum Distance Criterion

For first GPM development, the *Maximum Distance Criterion* is used. U. Sezgin [5] introduces an objective function, which is based on the sum of the distances between key points fixed on the links of the both arms of a cooperative robot, for maximization of the sum of distances between both arms. These points are defined as *Configuration Control Points* and they sufficiently represent the robot configurations in Cartesian space. A proper selection of configuration control points set is of major importance.

Taking into consideration the MXDC, a new constraint function is build introducing the following modifications:

- the constraint function is defined for a single manipulator;
- the positive scaling factor ψ is variable to increase constraint function influence only where performance criterion is imposed.

In order to define the constraint vector **z**, one has to consider the Euclidian distances d_p $(p = 1 \div n_{\mathrm{CCP}})$ between the p-th CCP and the obstacle (n_{CCP} is the number of the CCP). Thus, the performance criterion F is defined as:

$$F = \sum_{p=1}^{n_{\mathrm{CCP}}} \frac{1}{d_p}, \tag{10}$$

where n_{CCP} is the number of CCP. The constraint function **z** is:

$$\mathbf{z} = \psi \begin{bmatrix} \frac{\partial F}{\partial \theta_1} & . \; . \; . & \frac{\partial F}{\partial \theta_n} \end{bmatrix}^{\mathrm{T}}. \tag{11}$$

The computation procedure of GPM with MXDC is composed of the following steps:

- select the CCP on the given manipulator structure;
- express, relative to the base frame, the coordinates of the CCP depending on the joint coordinates;
- write the relations that specify the distances between the selected CCP and the obstacle;
- compose the performance criterion F by summing up the inverse of the distances d_p;
- write the equations of the direct kinematic model for the given manipulator;
- determine the constraint function **z** by differentiating the function F with respect to the joint coordinates;
- obtain the Jacobian matrix **J** and determine the pseudoinverse matrix $\mathbf{J}^{+}$;
- solve equation (8) and determine the joint vector $\delta\boldsymbol{\theta}$;
- obtain the joint vector $\boldsymbol{\theta}$.

3.1.2 Repulsive Potential Field

A *Repulsive Potential Field* (RPF) is used for the performance function construction of the second GPM. Khatib [6] originally proposed the Potential Field Method for on-line collision avoidance of a robot with proximity sensors. This method treats the robot as being under the influence of a virtual potential field U. The potential function is defined as the sum between an attractive potential, which attracts the robot toward the goal configuration, and a repulsive potential, which takes off the robot of obstacles.

The use of potential fields is a strong and efficient tool to solve the collision avoidance problems in the workspace. Virtual potential fields generated by the restriction surfaces are introduced. The manipulator links are subjected to potential field forces arising from the obstacles, these forces inducing a virtual repulsion from the obstacle surface. The basic idea is the application of RPF in all CCP on the manipulator links.

A possible choice for the mathematical expression of the RPF is:

$$\mathrm{U} = \begin{cases} \frac{1}{2}\eta \left(\frac{1}{\rho} - \frac{1}{\rho_0} \right)^2 & \text{if } \rho \le \rho_0, \\ 0 & \text{if } \rho > \rho_0, \end{cases} \tag{12}$$

where:
- η is a positive scaling factor,
- ρ is the minimum distance between robot and obstacle,
- ρ_0 is a positive constant called *distance of influence*.

The constraint function **z** is defined as the sum of the potential repulsive forces that induce a virtual repulsion from the restriction surface:

$$\mathbf{z} = \sum_{p=1}^{n_{\mathrm{CCP}}} R_p, \tag{13}$$

where the potential repulsive forces R_p are given by:

$$R_p = \begin{bmatrix} -\frac{\partial \mathrm{U}_j}{\partial \theta_1} & . \; . \; . & -\frac{\partial \mathrm{U}_j}{\partial \theta_n} \end{bmatrix}^{\mathrm{T}}; \cdot \tag{14}$$

the steps of the computational procedure of GPM with RPF are:
- determine the CCP where the repulsive potential forces act;
- express, relative to the base frame, the coordinates of these CCP depending on the joint coordinates;
- compose the distances between the CCP and the restriction surface;
- write the equations of the direct kinematic model for the given manipulator;

- determine the constraint functions **z** by summing the potential repulsive forces;
- obtain the Jacobian matrix **J** and determine the pseudoinverse matrix $\mathbf{J}^+$;
- solve equation (8) and determine the joint angular differential variation $\delta\boldsymbol{\theta}$;
- obtain the joint vector $\boldsymbol{\theta}$.

3. 2 Extended Jacobian Method

The *Configuration Control* M*ethod* augments the manipulator forward kinematics with a set of kinematic functions in operational space or in the joint space that reflects the desired additional task [17]. Let be $\mathbf{x}_E = f_E(\boldsymbol{\theta})$ the forward kinematic model of the robot, mapping the $n\times 1$ joint displacement vector $\boldsymbol{\theta}$ to the $m\times 1$ end-effector coordinate vector $\mathbf{x}_E$. Let $\mathbf{x}_c = f_c(\boldsymbol{\theta})$ define a set of $r = n - m$ kinematic functions. The augmented kinematic model is given by:

$$\mathbf{x} = \begin{pmatrix} \mathbf{x}_E \\ \mathbf{x}_c \end{pmatrix} = \begin{pmatrix} f_E(\theta) \\ f_c(\theta) \end{pmatrix}, \tag{15}$$

where $\mathbf{x}$ is the $n\times 1$ configuration vector. The user can then set up the desired additional task by imposing the constraint $\mathbf{x}_c(t) = \mathbf{x}_{cd}(t)$, where $\mathbf{x}_{cd}(t)$ is the user specified desired time variation of $\mathbf{x}_c$. The configuration control problem must ensure that the configuration vector $\mathbf{x}$ tracks the desired trajectory $\mathbf{x}_d(t) = \begin{pmatrix} \mathbf{x}_{Ed}(t) \\ \mathbf{x}_{cd}(t) \end{pmatrix}$ using a kinematic or dynamic control law.

The direct differential model obtained from the direct geometric model presented in equation (15) is:

$$\delta\mathbf{x} = \begin{pmatrix} \dfrac{\partial f_E(\theta)}{\partial\theta} \\ \dfrac{\partial f_c(\theta)}{\partial\theta} \end{pmatrix} \delta\boldsymbol{\theta} = \begin{pmatrix} \mathbf{J}_E(\theta) \\ \mathbf{J}_c(\theta) \end{pmatrix} \delta\boldsymbol{\theta} = \mathbf{J}\,\delta\boldsymbol{\theta}, \tag{16}$$

where $\mathbf{J}_E$ is the end-effector Jacobian matrix, $\mathbf{J}_c$ is the additional constraints Jacobian matrix and $\mathbf{J}$ is the extended Jacobian matrix.

If the desired additional task is to optimise an objective function, then this method is called the *Extended Jacobian Method*, introduced by Baillieul [18]. One defines $f_c(\theta) = \mathbf{N}^{\mathrm{T}} \dfrac{\partial g}{\partial\theta}$, where $g(\theta)$ is the scalar kinematic objective function to be optimised and $\mathbf{N}$ is the $n\times r$ null space matrix of $\mathbf{J}$ that corresponds to the self-motion of the redundant manipulator:

$$\mathbf{N} = \det(\mathbf{J}_a) \begin{pmatrix} \mathbf{J}_a^{-1}\mathbf{J}_b \\ -\mathbf{I}_r \end{pmatrix}, \quad \mathbf{J} = (\mathbf{J}_a \mathbf{J}_b), \tag{17}$$

where $\mathbf{J}_a$ is an m-squared matrix of the first columns of $\mathbf{J}$ and $\mathbf{J}_b$ is an $m\times r$ matrix of the remaining columns.

The necessary optimality condition of $g(\theta)$ is $f_c = 0$. Thus, if the desired trajectory is defined as $f_{cd}(t) = 0$ and the configuration control is used to track $\mathbf{x}_d(t)$, then the kinematic optimization problem can be solved.

The *Transpose Jacobian Matrix* method provides a solution of the EJM [19]:

$$\delta\boldsymbol{\theta} = \mathbf{J}^{\mathrm{T}}(\theta)\mathbf{K}\varepsilon, \tag{18}$$

where $\mathbf{K}$ is a positive definite matrix used to vary the additional constraints effect on the constraints imposed to the end-effector and $\boldsymbol{\varepsilon}$ is the error, $\boldsymbol{\varepsilon} = \mathbf{x}_d - \mathbf{x}$.

Because an appropriate matrix $\mathbf{K}$ is difficult to be chosen in order to obtain $\boldsymbol{\varepsilon} \le \boldsymbol{\varepsilon}_d$, where $\boldsymbol{\varepsilon}_d$ is the desired error, another method for inverse kinematic problem resolution must be used. This method consists in matrix $\mathbf{K}$ elimination followed by the Transpose Jacobian application, applied iteratively until $\boldsymbol{\varepsilon} \le \boldsymbol{\varepsilon}_d$ [7]. For a sampling step of end-effector task, this iterative algorithm is:

```
while $\boldsymbol{\varepsilon} > \boldsymbol{\varepsilon}_d$
    $\mathbf{x}_E^{(i)} = f_E(\boldsymbol{\theta}^{(i)})$
    $f_c^{(i)}(\theta) = (\mathbf{N}^{\mathrm{T}})^{(i)} \left(\frac{\partial g}{\partial\theta}\right)^{(i)}$
    $\boldsymbol{\varepsilon} = \mathbf{x}_d - \mathbf{x}^{(i)} = \begin{pmatrix} \mathbf{x}_{Ed} \\ 0 \end{pmatrix} - \begin{pmatrix} f_E(\boldsymbol{\theta}^{(i)}) \\ (\mathbf{N}^{\mathrm{T}})^{(i)} \left(\frac{\partial g}{\partial\theta}\right)^{(i)} \end{pmatrix}$        (19)
    $\delta\boldsymbol{\theta} = \mathbf{J}^{\mathrm{T}}\boldsymbol{\varepsilon}$
    $\boldsymbol{\theta}^{(i+1)} = \boldsymbol{\theta}^{(i)} + \delta\boldsymbol{\theta}$
end
```

4 Genetic Algorithm based strategy

4.1 Concept of genetic algorithm

The *genetic algorithm* is an efficient global optimization algorithm that uses operators taken from natural selection and survival of the fittest, characteristic to biological structures [20]. Due to the fact that this method needs no previous experience on the problem, it is applied on various problems. This method is fundamentally iterative operating on a set of candidate solutions, which form a so called population. An initial randomized or imposed population that consists of a group of *chromosomes* and represents the problem variables produces new population through successive iterations, using various genetic operators. The elements of the chromosome are called *genes*.

There are many reasons that make GA suitable for use in redundancy resolution:

- GA finds global optimum in complex spaces;
- does not need the computation of Jacobian matrix;
- GA solutions need only the forward kinematic equations of the manipulator in inverse kinematics resolution;
- does not require any additional constraints on the joint angles;
- GA allows additional non-linear constraints to be specified.

The common genetic operators are: *selection, elitism, crossover* and *mutation*. A function called *fitness function* determines when a new chromosome will replace a previous one or not, according to its value. Through several repetitions the evolution of the individuals leads to the domination of stronger ones. The applied operators in each step are:

- *Selection*: The selection function chooses parents for the next generation based on their fitness value. The common selection functions are: *roulette* and *tournament*. The roulette function simulates a roulette wheel with the area of each segment proportional to its expectation. The algorithm then uses a random number to select one of the sections with a probability equal to its area. The tournament function selects each parent by choosing individuals at random and then choosing the best individual out of that set to be a parent. *Tournament size* specifies the number of individuals from which only one is chosen.

- *Elitism*: In order to preserve the optimum individual of each generation for the next generation, the elitism operator is activated. The result is to keep the optimum individual of all previous generations in the current population and avoid the possibility of losing good individuals. *Elite count* is a positive integer specifying how many individuals in the current generation are guaranteed to survive in the next generation.

- *Crossover*: This genetic operator combines two individuals, or parents, to form a new individual, or child, for the next generation. The most common method uses *a single point crossover* operator. This operator chooses a random integer number between 1 and the number of variables and selects the vector entries numbered, less or equal to that number chosen, from the first parent, select genes numbered greater than the number chosen from the second parent, and finally combines these entries to form the child. For example,

$p_1 =$ [a b c d e f g h];

$p_2 =$ [1 2 3 4 5 6 7 8];

crossover point (at random) = 3

child = [a b c 4 5 6 7 8]

- *Mutation*: Mutation function makes small random changes in the individuals in the population, which provide genetic diversity. It operates on each binary bit of each chromosome and reverses the value of 1 to 0 and conversely.

4.1 Proposed strategy

The manipulator is considered as an open chain with *n* revolute joints. The proposed strategy starts with an imposed joint configuration adequate to a certain end-effector posture. These *n* joint angles represent the point around which the GA will search and provide the joint configuration adequate to the following imposed end-effector location. The strategy is, thus, iterative (Fig. 1) and is stopped when the number of end-effector references generations, n_g, is accomplished.

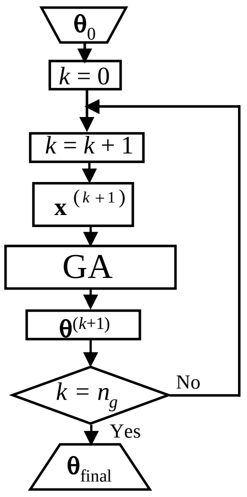

Fig. 1 Iterative schema of the strategy

The genetic algorithm variables are the joint angles vector corresponding to every end-effector imposed configuration. Thus, the number of genetic algorithm variables is equal with the number of DOF, *n*.

The variables vector has the following terms:

$$\mathbf{v}(i) = \theta_i;\quad \mathrm{i} = 1 \div n. \tag{20}$$

where θ_i is the *i*-th angle joint of the manipulator.

The lower and upper bounds for the vector variables at the *k*+1-th step of generation are obtained from the *k*-th joint angle values, subtracting and, respectively, adding a feasible difference $\Delta\theta_i$.

$$\theta_i^{(k)} - \Delta\theta_i \le \mathbf{v}(i)^{(k+1)} \le \theta_i^{(k)} + \Delta\theta_i; \quad i = 1 \div n,\ k = 1 \div n_g. \tag{21}$$

The fitness function of the genetic algorithm is the objective function to minimize. In our case, this function is the sum of joint displacements between two successive end-effector locations:

$$\sum_{i=1}^{n} \left\| \theta_i^{(k+1)}(i) - \theta_i^{(k)}(i) \right\| \quad i = 1 \div n,\ k = 1 \div n_g. \tag{22}$$

The end-effector task and the other additional constraint (obstacle avoidance) are expressed in one non-linear constraint function of the form $\mathbf{C} \le 0$, where **C** is a 2 dimensional vector containing the following non-linear expressions:

$$\begin{aligned} &\mathbf{C}(1) = \|\mathbf{x} - \mathbf{x}_d\| - \varepsilon_d; \\ &\mathbf{C}(2) = d_0 - \min(d_p); \\ &p = 1 \div n_{\mathrm{CCP}}, \end{aligned} \tag{23}$$

The first term of the vector verifies that the positioning and orientation error of the end-effector is smaller than a desired error, ε_d imposed by the user. $\mathbf{x}$ and $\mathbf{x}_d$ are the vectors of the real and desired end-effector configuration. The second term guarantees the obstacle avoidance because the minimum of the distances d_p is greater than a desired distance, d_0 imposed by the user. The d_p distances are calculated between the *p*-th *Configuration Control Point* and the obstacle. n_{CCP} is the number of the CCP. The CCP are imposed by the user on the manipulator structure.

The above described mathematical models for fitness and non-linear constraint functions are solved for every *k*-th sampling step of end-effector references generation using the genetic algorithm tool of MATLAB. Thus, the proposed strategy involves a number of n_g successive GA resolutions.
The input data for the GA at the k+1-th step of the proposed strategy for redundancy resolution are:
- links dimensions and previous joint configuration vector $\boldsymbol{\theta}^{(k)}$;
- $\Delta\theta_i$ which gives the lower and upper bounds of GA variables;
- imposed end-effector configuration for *k*+1-th sampling step;
- desired positioning and orientation error of the end-effector, ε_d;
- number n_{CCP} of CCP and its positions on manipulator structure;
- value of the desired distance d_0;
- GA parameters: population size, the selection function, the elite count, the crossover rate, the mutation function, the algorithm stopping criteria options, etc.

The output data is the joint configuration vector $\boldsymbol{\theta}^{(k+1)}$.

The starting population is randomly generated to set the variable values, which are used to calculate the fitness function value. GA uses selection, elitism, crossover and mutation procedures to create new generations. The new generations converges towards a minimum for the fitness value while the expressions of the non-linear constraint function are accomplished.

The use of the nonlinear constraint function in GA supposes a rapidly convergence to a minimum for the fitness value because of elitism operator, which chooses only the individuals that respect the non-linear inequalities.

The main advantage of the proposed strategy, in contrast with redundancy resolution methods with linearized solutions, consists in fact that it uses, in inverse kinematics resolution, only the direct kinematics equations. Also, it allows additional non-linear constraints to be specified. The strategy does not need the computation of the Jacobian matrix and its pseudoinverse so that any problem related to the inversion of this matrix (kinematic singularities) is overcome. The algorithmic singularities, artificially introduced by any additional constraint working in the null-space of the manipulator Jacobian, do not occur as well.

5 Simulation results

The illustrative simulations are obtained for a laboratory model of planar redundant manipulator, possessing four DOF (Fig. 2). The experimental model was realized at the Laboratory of Robotics-Mechatronics Group of Institute of Solid Mechanics of Romanian Academy [21].

The proposed goal is to generate the references (position and orientations) of the end-effector along the contour of a circle with radius *r*, whose surface is considered to be restrictive for all four elements of the manipulator structure. The operational space dimension is in this case *m* = 3 because the position and orientation of the end-effector (EEF) are both taken into consideration. Thus, the degree of redundancy is n-m = 1.

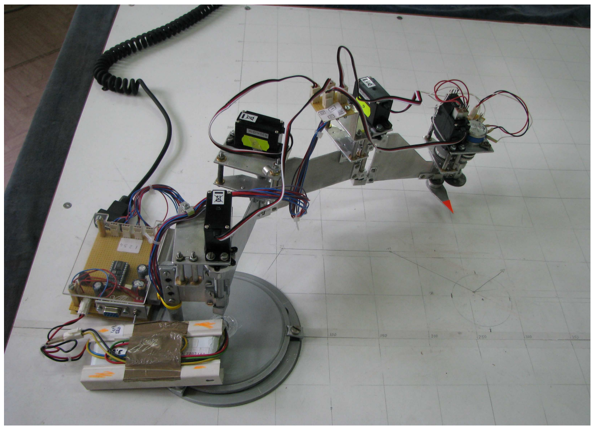

Fig. 2 Laboratory model [21]

The initial posture of the manipulator is illustrated in Fig. 3 and is given by the following measures:

$$\begin{aligned}&\boldsymbol{\theta}_0 = [0.72\ \ 5.49\ \ 5.55\ \ 3.93];\\&l_1 = 0.12;\, l_2 = 0.12;\, l_3 = 0.10;\, l_4 = 0.05;\, x_0 = 0;\, y_0 = 0;\\&r = 0.03;\, x_c = 0;\, y_c = 0.2.\end{aligned} \tag{24}$$

where $\boldsymbol{\theta}_0$ is the vector of initial joint coordinates expressed in radians, l_1, l_2, l_3 and l_4, are the lengths of the links expressed in meters, x_0 and y_0 are the manipulator base coordinates, r is the radius of the restriction circle and x_c and y_c are its centre coordinates.

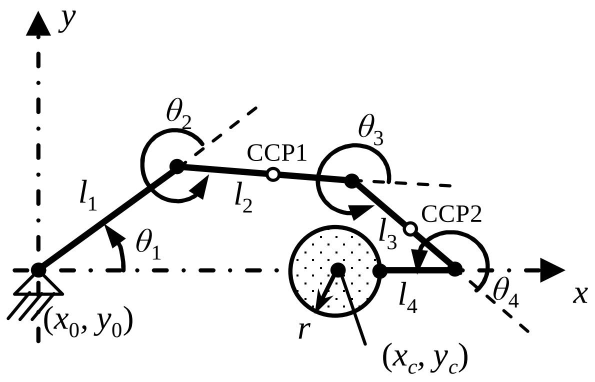

Fig. 3 Initial manipulator configuration

The end-effector references generation is a function of sampling step of generation, k:

$$\begin{aligned}&x_d^{(k)} = x_c + r\cdot\cos(k\cdot\Delta\alpha);\\&y_d^{(k)} = y_c + r\cdot\sin(k\cdot\Delta\alpha);\\&\Sigma\theta_d^{(k)} = 5\cdot pi + k\cdot\Delta\alpha.\end{aligned} \tag{25}$$

where $\Delta\alpha$ is the angular step of generation.

The end-effector coordinates, obtained using the direct geometric model, have the following expressions:

$$\begin{aligned}&x^{(k)} = \sum_{i=1}^{4} l_i\cdot\cos\left(\sum_{j=1}^{i}\theta_j^{(k)}\right);\\&y^{(k)} = \sum_{i=1}^{4} l_i\cdot\sin\left(\sum_{j=1}^{i}\theta_j^{(k)}\right);\\&\Sigma\theta^{(k)} = \sum_{i=1}^{4}\theta_i^{(k)}.\end{aligned} \tag{26}$$

The angular step of generation is $\Delta\alpha=3^0$ and, thus, the number of strategy steps is n_g=120. For all four methods, the CCP (n_{CCP} =2) are placed in the middle of second element and, respectively, in the middle of third element of the manipulator. For first method, ψ - the variable scaling factor gives the influence of MXCD. For this particular case, the chosen formula was $\psi = 0.4\cdot\frac{k}{n_g}$ and the sum of joint angles displacements for all $n_g = 120$ steps is 18.2 radians (Fig. 4).

For GPM with RPF, the influence of the repulsive forces is given by η - the positive scaling factor and ρ_0 - the distance of influence. For the simulation illustrated in Fig. 5, we have the following chosen values: $\eta = 0.3; \rho_0 = 0.02$. The sum of joint angles displacements is smaller than previous case, 15.06. The simulation results using EJM is illustrated in Fig.6. The sum of joint angles displacements is, in this case, 19.3.

For the last method based on GA we have the desired distance d_0 = 0.015 and the imposed positioning and orientation error of the end-effector is ε_d = (0.001 0.001 0.1°).

The vector that produces the lower and upper bounds of the GA variables is, constant for every k-th step, $\Delta\theta = [4^0\ \ 7^0\ \ 8^0\ \ 4^0]$.

The GA characteristics imposed in the Matlab GA tool are:

- initial population: randomly generated
- population size: 50
- selection function: tournament
- tournament size: 4
- elite count: 5
- crossover: single point

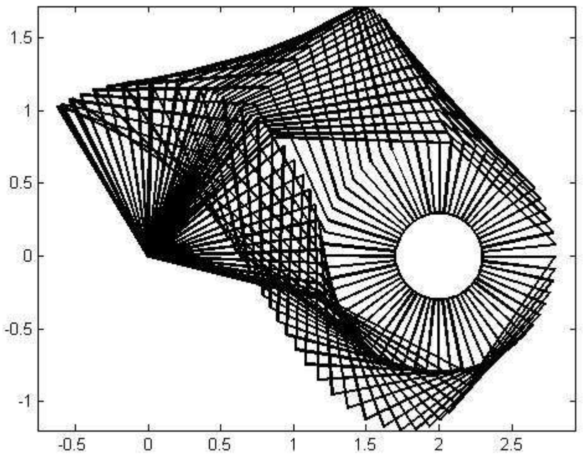
Fig. 4 Simulation results using GPM with MXDC

GA converges to a minimum for the value of the fitness function after about 20 generations, while the non-linear constraint function is fulfilled.

The simulation results obtained (Fig. 7) show a better performance of the GA based strategy compared with the redundancy resolution methods with linearized solutions.

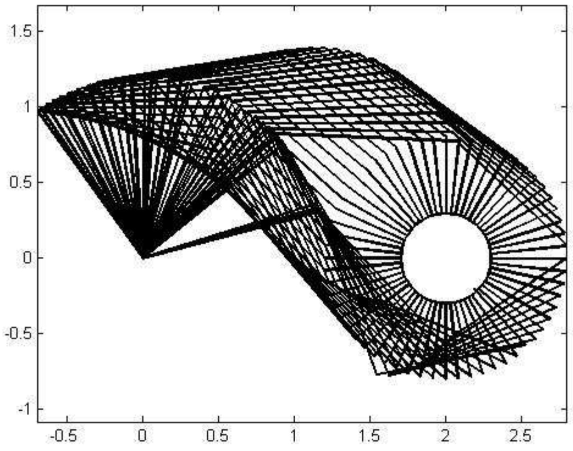
Fig. 5 Simulation results using GPM with RPF

For instance, the sum of joint angles displacements for all sampling steps is 13.49 radians, smaller than 15.06, obtained using the GPM with RPF working in the null-space of the Jacobian matrix.

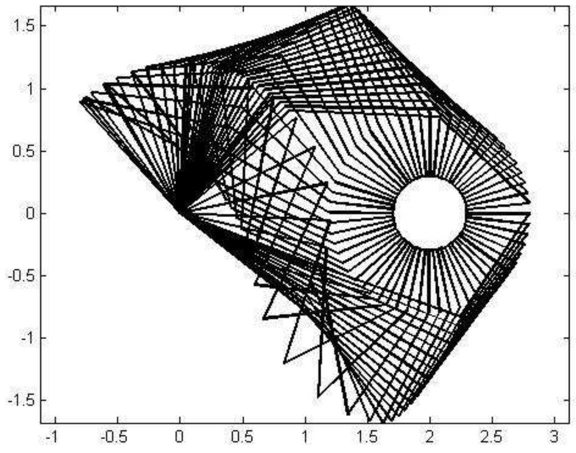
Fig. 6 Simulation results using EJM

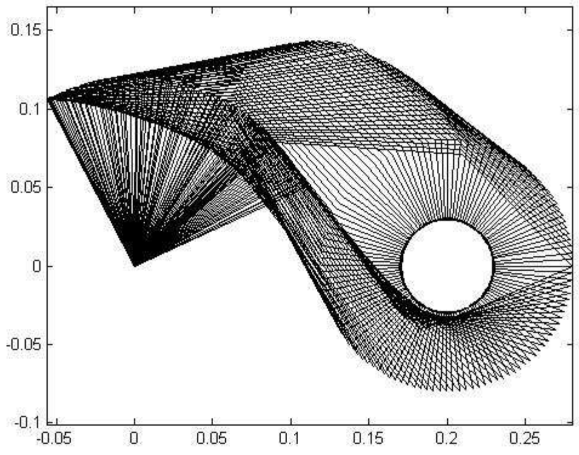
Fig. 7 Simulation results using GA

6 Conclusions

Four different iterative strategies for obstacle avoidance of a redundant manipulator were presented in this paper. The end-effector task consists in generating the references along the contour of a curve.

The objective to simultaneously minimize the end-effector location error and the manipulator total joint displacement while the collision with the obstacle is avoided was fulfilled by all four proposed strategies.

A major advantage of GPM is that works in real time. The computational time required is sensible smaller than using EJM, which requires expensive computational resources because of increasing Jacobian dimension and of a greater number of iterations necessary for optimal solution identification.

The most important advantages of the EJM are the fact of dealing with a square Jacobian matrix (thus, the use the pseudoinverse is eliminated) and the possibility of choosing a desired error value (which can guarantee improved end-effector task accuracy while obstacle avoidance is certainly accomplished).

The usual disadvantages of methods with linearized solutions consist in difficulties when choosing the constraint expressions (the used criteria, having complicated expressions in symbolic forms, must be differentiable) and in algorithmic singularities introduced by these additional constraints. These two disadvantages are eliminated by advanced redundancy resolution approaches, such as a GA based method, which offers, also, the possibility to add additional performance criteria through non-linear constraints.

The simulations results obtained for a redundant planar manipulator with four DOF indicate the superiority of the GA based strategy in what concerns the sum of joint angles displacements, but, in the same time, the most important disadvantage is that requires expensive computational resources and cannot deal with real-time applications.

Acknowledgements. The authors wish to express their gratitude to the Research and Educational Ministry (MEC) and to the Romanian Academy for their support of the program of work reported herein. The work took place as part of research project no. 263/2007-2010 in the framework of Grants of CNCSIS Program IDEAS, PN II (National Program for Scientific Research and Innovation Technologies).

References:

[1] F. Guenter, L. Roos, A. Guignard, A. G. Billard, *Design of a Biomimetic Upper Body for Humanoid Robot Robota*, In: Proceedings of the IEEE-RAS International Conference on Humanoid Robots, Tsukuba, Japan, December 5-7, 2005

[2] C. Kapoor, M. Cetin, D. Tesar, *Performance Based Redundancy Resolution with Multiple Criteria*, In: Proceedings of DETC98, ASME Design Engineering Technical Conference, September 13-16, 1998, Georgia, USA, 1998.

[3] H. Homaei, M. Keshmiri, *Redundancy Resolution with Minimum Vibration for Flexible Redundant Manipulators*, WSEAS Transactions on Systems, Issue **1**, Vol. **5**, pp. 299-304, ISSN 1109-2777, January 2006.

[4] M-C. Popescu, I. Borcosi, O. Olaru, N. Antonie, *Simulation of n-r robots*, WSEAS Transactions on Systems and Control, Vol. 3, Issue 3, pp. 149-158, ISSN 1919-8763, 2008.

[5] U. Sezgin, L.D. Seneviratne, S.W.E. Earles. Collision Avoidance in Multiple-Redundant Manipulators. The International Journal of Robotics Research, Vol. **16**, No. 5, pp. 714-724, 1997.

[6] O. Khatib. Real-Time Obstacle Avoidance for Manipulators and Mobile Robots. The International Journal of Robotics Research, Vol. **5**, No. 1, pp. 90-98, 1986.

[7] V. Perdereau, M. Drouin, C. Passi. Real-Time Collision Avoidance for Redundant Manipulators. Proceedings of the IASTED International Conference Robotics and Applications 2000, Honolulu, Hawaii, USA, August 14-16, 2000.

[8] C. Secară, *Control Strategies for obstacle avoidance by redundant manipulators*, Proceedings of the Romanian Academy, Series A: Mathematics, Physics, Technical Sciences, Information Science, vol. **9**, nr.1, 2008.

[9] C. Secară, *A genetic algorithm based strategy for redundancy resolution with multiple criteria*, Proceedings of the 9th WSEAS International Conference on International Conference on Automation and Information (ICAI'08), Bucharest, Romania, June 24-26, pp. 242-247, ISSN:1790-5117, 2008.

[10] S. Mitsi, K.-D. Bouzakis, D. Sagris, G. Mansour, *Determination of optimum robot base location considering discrete end-effector by means of hybrid genetic algorithm*, Robotics and Computer-Integrated Manufacturing, **24**, pp. 50-59, 2008.

[11] A. Nearchou, *Solving the inverse kinematics problem of redundant robots operating in complex environments via a modified genetic algorithm*, Mech. Mach Theory, Vol. **33**, No. 3, pp. 273-292, 1998.

[12] A. A. Ata, R.T. Myo, Optimal Point-to-Point trajectory tracking of Redundant Manipulators using Generalized Pattern Search, International Journal of Advanced Robotic Systems, Vol. **2**, No. 3, pp. 239-244, 2005.

[13] X. Du, H. Chen, W. Gu, *Neural network and genetic algorithm based global path planning in a static environment*, Journal of Zhejiang University Science, 6A(6), pp. 549-554, 2005.

[14] C. Ham, R. Johnson, *Robust fuzzy control for robot manipulators*, In: Proceedings of the IEEE Control Theory Applications, vol. 147, No. 2, pp. 212-216, March 2000.

[15] M. Da Graça Marcos, J.A. Tenreiro Machado, T.-P. Azevedo-Perdicoúlis, *Trajectory planning of redundant manipulators using genetic algorithms*, Communications in Nonlinear Science and Numerical Simulation, Vol. **14** (7), pp. 2858-2869, 2009.

[16] A. Liegeois. Automatic Supervisory Control of Configuration and Behavior of Multibody Mechanism. IEEE Trans. Sys. Man Cybernet, Vol. **7**, pp. 868-871, 1977.

[17] H. Seraji. Configuration Control of Redundant Manipulators: Theory and Implementation. IEEE Trans. On Robot and Automat., Vol. **5**, No. 40, pp. 427-490, 1989.

[18] J. Baillieul, J. Hoolerbach, R. Brockett. Programming and Control of Kinematically Redundant Manipulators. Proceedings of IEEE Conf. on Decision and Control, pp. 768-774, 1984.

[19] 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. The International Journal of Robotics Research, Vol. **10**, No. 4, pp. 410-425, 1991.

[20] D. Coley, *An Introduction to genetic algorithms for scientists and engineers,* World Scientific Press, 1999.

[21] I. Nițu, C. Secară, W. Racoviță, *A Scara type redundant robot programmed on a virtual model*, Bulletin of the Transilvania University of Brasov, Proceedings of the 4th International Conference Robotics '08, Braşov, Romania, November 13-14, 2008, Vol. 15(50)-series A, 2008.

[22] C. Secară, L. Vladareanu, Iterative genetic algorithm based strategy for obstacles avoidance of a redundant manipulator ISI Proceedings, Recent Advances in Applied Mathematics, Harvard University, Cambridge, USA, 2010, pg. 361-366, ISBN 978-960-474-150-2, ISSN 1790-2769