Iterative genetic algorithm based strategy for obstacles avoidance of a redundant manipulator

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

Abstract: - This paper presents a strategy for obstacles avoidance of a redundant manipulator based on an iterative genetic algorithm. The objective of the strategy is to simultaneously minimize the end-effector location error and the manipulator total joint displacement while the collision with the obstacles is avoided. The end-effector task consists in generating the references along the contour of a curve. The proposed strategy is iterative in the sense that the joint configuration computed in the previous step represents the current point around which the genetic algorithm finds the next joint configuration. The strategy is implemented using the genetic algorithm tool of Matlab software and the illustrative simulations 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, genetic algorithm, obstacle avoidance, multiple criteria.

1 Introduction
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 [1]. 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.

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 [2] 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 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 obstacles is avoided.

A short review of the redundancy resolution methods is presented in second section. The third section describes a brief overview of the concept of GA. In fourth section the proposed strategy for redundancy resolution is characterized through its variables, fitness function and non-linear constraint functions. The illustrative simulations 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 Redundancy resolution methods
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 \( \theta \), for rotation joint case):

\[
\begin{align*}
x &= f(\theta); \\
x &=[ x_1 \ x_2 \ldots \ x_m ]^T; \\
\theta &=[ \theta_1 \ \theta_2 \ldots \ \theta_n ]^T.
\end{align*}
\]

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 \( \theta \) with a given \( x \), the inverse of the equation (1) is required:

\[
\theta = f^{-1}(x).
\]

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 \( \theta \), obtaining the inverse differential model:

\[
\delta x = J(\theta)\delta \theta, \quad (3)
\]

where \( J(\theta) = \frac{\delta f(\theta)}{\delta \theta} \) 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 solved linearly. For non-redundant manipulator structures, \( n = m \), the inverse differential model is simple obtained using the inverse of the Jacobian matrix:

\[
\delta \theta = J^{-1}\delta x. \quad (4)
\]

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

\[
J^+ = J^T(JJ^T)^{-1}. \quad (5)
\]

Two basic methods with linearized solutions can be distinguished in the literature [3]. Gradient Projection Method - while the end-effector task is accomplished by means of the least norm solution, some performance criteria are carried out by means of the null space solution, which yields self-motion of the links in the joint space only, without any end-effector motion effect. Extended Jacobian Method defines additional constraints for the given task until the relationship between the joint space and the end-effector space becomes non-redundant. Thus, the Jacobian matrix become square and can be 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.

\[
P(e(\theta)) = (g - e(\theta))^2, \quad (6)
\]

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:

Minimize \( P(e(\theta)) = (g - e(\theta))^2 \),

subject to \( h(\theta) \leq 0, \quad l_i \leq \theta \leq u_i, \) for \( i = 1 \div n \) (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 [4,5,6,7], direct search [1,8], neural networks [7], or fuzzy techniques [9].

### 3 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 [10]. 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,

\[ p_1 = [a \ b \ c \ d \ e \ f \ g \ h] \]
\[ p_2 = [1 \ 2 \ 3 \ 4 \ 5 \ 6 \ 7 \ 8] \]

\[ \text{crossover point (at random) = 3} \]
\[ \text{child} = [a \ b \ c \ 4 \ 5 \ 6 \ 7 \ 8] \]

- 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 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](image-url)

The genetic algorithm variables are the joint angles vector corresponding to every end-effector imposed configuration. Thus, the number of genetic
The end-effector task and the other additional constraint (obstacle avoidance) are expressed in one non-linear expressions:

\[
\theta_i^{(k)} - \Delta \theta_i \leq v(i)^{(k+1)} \leq \theta_i^{(k)} + \Delta \theta_i;
\]

\[
i = 1 \div n, \quad k = 1 \div n_g.
\]

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:

\[
\sum_{i=1}^{n} \| \theta_i^{(k+1)}(i) - \theta_i^{(k)}(i) \|_2
\]

\[
i = 1 \div n, \quad k = 1 \div n_g.
\]

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

\[
C(1) = \| x - x_o \| - e_d;
\]

\[
C(2) = d_{po} - \min(d_{po});
\]

\[
p = 1 \div n_{CCP}, \quad o = 1 \div n_o.
\]

The first term of the vector verifies that the positioning and orientation error of the end-effector is smaller than a desired error, \( e_d \) imposed by the user. \( x \) and \( x_o \) are the vectors of the real and desired end-effector configuration. The second term guarantees the obstacle avoidance because the minimum of the distances \( d_{po} \) is greater than a desired distance, \( d_0 \) imposed by the user. The \( d_{po} \) distances are calculated between the \( p \)-th Configuration Control Point (CCP) and the \( o \)-th obstacle. \( n_{CCP} \) is the number of the CCP and \( n_o \) is the number of the obstacles. 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 \( \theta^{(k)} \);
- \( \Delta \theta \) 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, \( e_d \);
- number \( n_{CCP} \) of CCP and its positions on manipulator structure;
- number \( n_o \) of the obstacles and its locations in manipulator workspace;
- 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 \( \theta^{(k+1)} \).

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 [11].
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 \( n - m = 1 \).

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

\[
\theta_0 = [0.72 \ 5.49 \ 5.55 \ 3.93];
\]

\[
l_1 = 0.12; \quad l_2 = 0.12; \quad l_3 = 0.10; \quad l_4 = 0.05; \quad x_0 = 0; \quad y_0 = 0;
\]

\[
r = 0.03; \quad x_c = 0; \quad y_c = 0.2.
\]

where \( \theta_0 \) is the vector of initial joint coordinates expressed in radians, \( l_1, l_2, l_3, l_4 \) are the lengths of the links expressed in meters, \( x_0, y_0 \) are the manipulator base coordinates, \( r \) is the radius of the restriction circle and \( x_c, y_c \) are its centre coordinates.

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

\[
x_d^{(k)} = x_c + r \cdot \cos(k \cdot \Delta \alpha);
\]

\[
y_d^{(k)} = y_c + r \cdot \sin(k \cdot \Delta \alpha);
\]

\[
\Sigma \theta_d^{(k)} = 5 \cdot \pi + k \cdot \Delta \alpha.
\]

where \( \Delta \alpha \) is the angular step of generation.

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

\[
x^{(k)} = \sum_{i=1}^{4} l_i \cdot \cos \left( \sum_{j=1}^{i} \theta_j^{(k)} \right);
\]

\[
y^{(k)} = \sum_{i=1}^{4} l_i \cdot \sin \left( \sum_{j=1}^{i} \theta_j^{(k)} \right);
\]

\[
\Sigma \theta^{(k)} = \sum_{i=1}^{4} \theta_i^{(k)}.
\]

The CCP (\( n_{CCP} = 2 \)) are placed in the middle of second element and, respectively, in the middle of third element of the manipulator, and the desired distance, imposed by the user is \( d_0 = 0.015 \).

The imposed positioning and orientation error of the end-effector is \( \varepsilon_d = (0.001 \ 0.001 \ 0.1^\circ) \).

The angular step of generation is \( \Delta \alpha = 3^\circ \) and, thus, the number of strategy steps is \( n_s = 120 \). The vector that produces the lower and upper bounds of the GA variables is, constant for every \( k \)-th step, \( \Delta \theta = [4^\circ \ 7^\circ \ 8^\circ \ 4^\circ] \).

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

Average of the number of generations needed to converge

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 show a better performance of the proposed strategy compared with the redundancy resolution methods with linearized solutions. For instance, the sum of joint angles displacements for all sampling steps in our case is 13.49 radians, smaller than 15.06, obtained using the Gradient Projection method with an artificial repulsive field working in the null-space of the Jacobian matrix [3].

Fig. 2 Laboratory model [11]

Fig. 3 Initial manipulator configuration
6 Conclusions
An iterative genetic algorithm based strategy for obstacles avoiding of a redundant manipulator is developed in this paper. The objective of the strategy is to simultaneously minimize the end-effector location error and the manipulator total joint displacement while the collision with the obstacles is avoided. The end-effector task consists in generating the references along the contour of a curve.

The proposed technique is optimal in the sense that, it produces a quite small and acceptable positional and orientation error end-effector error with the minimum possible joint displacements, while generating a guaranteed collision-free motion of the robot.

The advantages of the proposed method in contrast with redundancy resolution methods with linearized solutions (Gradient Projection and Extended Jacobian) are singularities problems overcoming, the use in inverse kinematic resolution only of direct kinematics equations and possibility to add additional performance criteria through non-linear constraints.

References:


