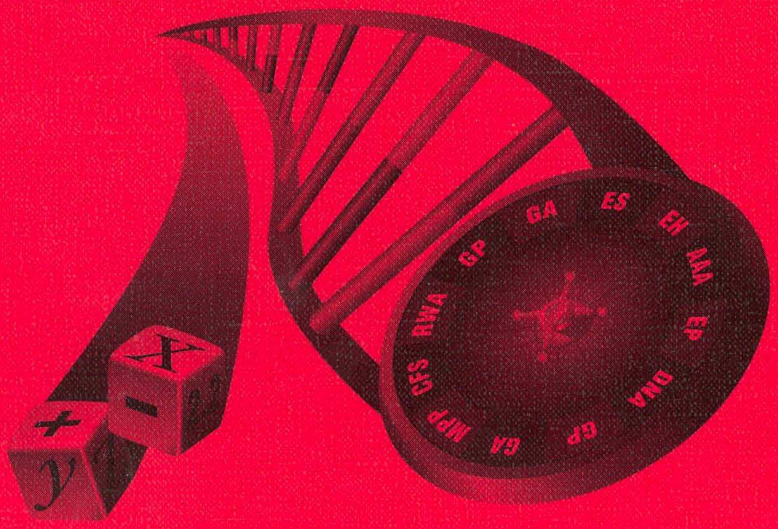

GECCO - 2000

LATE BREAKING PAPERS

AT THE

2000 GENETIC AND EVOLUTIONARY COMPUTATION CONFERENCE

Las Vegas, Nevada
July 8, 2000

Trajectory Optimization for Redundant Robots Using Genetic Algorithms with Heuristic Operators

E. J. Solteiro Pires

Universidade Trás-os-Montes e Alto Douro
Departamento de Engenharias
5001 Vila Real, Portugal
epires@utad.pt

J. A. Tenreiro Machado

Instituto Superior de Engenharia do Porto
Departamento de Engenharia Electrotécnica
R. Antonio Bernardino de Almeida
4200-072 Porto, Portugal
jtm@dee.isep.ipp.pt

Abstract

This paper proposes a genetic algorithm to generate trajectories for robotic manipulators. The objective is to minimize the ripple in the trajectory time evolution and to minimize the actuator energy requirements without colliding with any obstacles in the workspace. The article presents the results for several redundant and hyper-redundant manipulators.

Keywords: Genetic algorithms, Robotics, Trajectory planning, Optimization.

1 INTRODUCTION

In the last decade genetic algorithms (*GAs*) have been applied in a plethora of fields such as in control, parameter and system identification, robotics, planning and scheduling, image processing, pattern recognition, speech recognition. This paper addresses the area of robotics, namely the trajectory planning for mechanical manipulators. Planning a robot trajectory consists in finding a continuos motion that takes the arm from a given starting configuration, without collision with any obstacle, up to a desired end position in the workspace.

Various methods for trajectory planning and collision avoidance schemes based on *GAs* have been proposed. A possible approach consists in adopting the differential inverse kinematics, using the Jacobian matrix, for generating the manipulator trajectories (Chen and Zalzala, 1997; Davidator 1991). However, the algorithm must take into account the problem of kinematic singularities that may be hard to tackle. To avoid this problem, other algorithms for the trajectory generation are based on the direct kinematics (Doyle and Jones, 1996; Rana and Zalzala, 1996; Kubota *et al.* 1997; Wang and Zalzala 1996).

Chen and Zalzala (1997) propose a *GA* method to generate the position and the configuration of a mobile manipulator. The authors study the optimization of the least torque norm, the *manipulability*, the torque distribution and the obstacle avoidance, through the inverse kinematics scheme.

Davidor (1991) also applies *GAs* to the trajectory generation by searching the inverse kinematics solutions to pre-defined end-effector robot paths.

Kubota *et al.* (1997) study a hierarchical trajectory planning method for a redundant manipulator using a virus-evolutionary *GA*. This method runs, simultaneously, two processes. One process calculates some manipulator collision-free positions and the other generates a collision-free trajectory by combining these intermediate positions.

Rana and Zalzala (1996) developed a method to plan a near time-optimal, collision-free, motion in the case of multi-arm manipulators. The planning is carried out in the joint space and the path is represented as a *string* of via-points connected through cubic splines.

Doyle and Jones (1996) propose a path planning scheme that uses a *GA* to search the manipulator configuration space for the optimum path. The *GA* generates good path solutions but is not sufficiently robust.

In this line of thought, this paper proposes a path planning method based on a *GA* while adopting the direct kinematics and the inverse dynamics. The optimal trajectory is the one that minimizes both the path length, the ripple in the time evolution and the energy requirements, without any collision with the obstacles in the workspace.

Bearing these facts in mind, this paper is organized as follows. Section 2 introduces the problem and the *GA*-based method for its resolution. Sections 3, 4 and 5 describe the solution representation, the *GA* operators used in trajectory planning and the optimization criteria, respectively. Based on this formulation, section 6 presents the results for several simulations involving different robot structures and obstacles in the workspace. Finally, section 7 outlines the main conclusions.

2 PROBLEM FORMULATION

In the study we consider robotic manipulators that are required to move from a initial configuration up to a given final point. In the experiments we consider 2, 3 and 4 link planar manipulators with rotational joints. The arms have identical link lengths and weight, making a total reach distance of 2 *m* and a total gross weight of 2 kg, while the joints are free to rotate 360°. Therefore, the manipulator workspace is a circle with 2 *m* radius, that may present obstacles such as rectangles and circles. To test a possible collision between the manipulator and the obstacles, the arm structure is discretized is several points and then these points are checked in order to verify if they are inside any obstacle.

In what concerns the trajectory generator, it is adopted a *GA* scheme to search for a global optimal robot path. The manipulator trajectory consists in a set of strings that represent the joint positions between the initial and final robot configurations.

3 REPRESENTATION

The path is encoded, directly, as strings in the joint space to be used by the *GA* as:

$$[(q_{11},...,q_{k1}),..., (q_{1j},...,q_{kj}),..., (q_{1n},...,q_{kn})] \quad (1)$$

The *i*th joint variable for a robot intermediate *j*th position is q_{ij}, the chromosome is constituted by n genes (configurations) and each gene if formed by k values, where k is the number of robot links. The value of q_{ij} is represented as a floating-point number.

To increase the initials paths quality a heuristic initialization are used. So q_{ij} values are initialized according to the follows equation:

$$q_{ij} = q_{i,j-1} + k_r \; x \quad (2a)$$

$$x \sim U[-1; 1] \quad (2b)$$

where, $1 \le i \le k$, $1 \le j \le n$, q_{i0} is the initial configuration i, and k_r is a step parameter.

It should be noted that the initial configuration has not been encoded into the string because this configuration remains unchanged throughout the search.

For simplicity, the normalized time between two consecutive configurations is considered $\Delta t = 1$ sec, without losing generality, because it is always possible to perform a time re-scaling.

4 OPERATORS IN THE GENETIC ALGORITHM

An initial population of strings is constructed by generating random paths (see section 3) between the starting configuration and one final robot position. The search is then carried out among this population. The three different operators used in the genetic planning are reproduction, crossover and mutation, as described in the sequel.

In what concern the reproduction operator, the successive generations of new strings are reproduced on the basis of their fitness function. In this case, it is used a tournament selection (Goldberg 1989) to select the strings from the old population, up to the new population.

For the crossover operator, the strings in the new population are grouped together into pairs at random. Crossover is then performed among pairs. The crossover point is only allowed between genes (*i.e.* the crossover operator may not disrupt genes). The heuristic crossover operator adopted considers the initial strings (3) (where Q_i and P_i represents an entire configuration, $1 \le i \le n$) and the crossover point $j + 2$, $-1 \le j \le n-3$.

$$[Q_1,..., Q_j, Q_{j+1}, Q_{j+2}, Q_{j+3}, Q_{j+4}, Q_{j+5},..., Q_n)] \quad (3a)$$

$$[P_1,..., P_j, P_{j+1}, P_{j+2}, P_{j+3}, P_{j+4}, P_{j+5},..., P_n)] \quad (3b)$$

The resultant strings are:.

$$[P_1,..., P_j, 0.2\,Q_{j+1}+0.8\,P_{j+1}, 0.4\,Q_{j+2}+0.6\,P_{j+1}, 0.6\,Q_{j+3}+0.4\,P_{j+3}, 0.8\,Q_{j+4}+0.2\,P_{j+1}, Q_{j+5},..., Q_n)] \quad (4a)$$

$$[Q_1,..., Q_j, 0.2\,P_{j+1}+0.8\,Q_{j+1}, 0.4\,P_{j+2}+0.6\,Q_{j+1}, 0.6\,P_{j+3}+0.4\,Q_{j+3}, 0.8\,P_{j+4}+0.2\,Q_{j+1}, P_{j+5},..., P_n)] \quad (4b)$$

Finally, for the mutation operator one gene value is replaced with a given probability and follows the equation:

$$q_{ij}(t + 1) = q_{ij}(t) + k_m \, x \quad (5a)$$

$$x \sim U[-1; 1] \quad (5b)$$

where q_{ij} is the i value of j gene at generation t, x is a uniform random number between $[-1, 1]$ and k_m a parameter.

5 EVOLUTION CRITERIA

Six criteria have been selected to qualify the evolving manipulator paths. All constraints and criteria are translated into penalty functions to be minimized. Each criterion is computed individually and then, is used in the fitness function evaluation (Pires and Machado 1999).

The fitness function f, adopted to evaluate the candidate trajectories is defined as:

$$f = \begin{cases} -\alpha_1 \dot{q} - \alpha_2 \ddot{q} - \alpha_3 \dot{p} - \alpha_4 \ddot{p} - \alpha_5 \varepsilon - \alpha_6 P_a & nap = 0 \\ +\infty & nap \neq 0 \end{cases} \quad (6)$$

where q, q, p, p, ε and nap are the criteria defined in the sequel. The optimization goal consists in finding a set of design parameters that minimize f according to the priorities given by the values of α_i ($i = 1,\ldots,6$).

The joint velocities q are used to minimize the manipulator traveling distance. This criteria is defined as:

$$\dot{q} = \sum_{j=1}^{n} \sum_{i=1}^{k} \dot{q}_{ij}^{\,2} \quad (7)$$

where $\dot{q}_{ij}$ is the jth intermediate incremental position node of the ith joint and n is the number of the intermediate position in the simulation of the k- link manipulator. This equation is used to minimize the traveling distance because if the curve length is minimized, then the ripple in the space trajectory is indirectly reduced. For a function $g(x)$ the distance curve length is $\int[1 + (g')^2]\,dx$ and, consequently, to minimize the distance curve length it is adopted the expression $\int(g')^2\,dx$. The fitness function maintains the quadratic terms so that the robot configurations are uniformly distributed between the initial and final configurations.

The joint accelerations q are used to minimize the ripple in the time evolution of the robot trajectory. This criteria is calculated as:

$$\ddot{q} = \sum_{j=1}^{n} \sum_{i=1}^{k} \left|\ddot{q}_{ij}\right|^2 \quad (8)$$

where k, n, i and j are defined as previously.

The cartesian velocities $\dot{p}$ is introduced in the fitness function f to minimize the total trajectory length, from the initial point up to the final point. This criteria is defined as:

$$\dot{p} = \sum_{w=2}^{n} d(p_w, p_{w-1})^2 \quad (9)$$

where p_w is the robot w intermediate arm cartesian position and $d(\cdot,\cdot)$ is a function that gives the distance between the two arguments.

The cartesian acceleration p in the fitness functions is responsible for reducing the ripple in time evolution of the arm velocities. This criteria is formulated as:

$$\ddot{p} = \sum_{w=3}^{n} \left| d(p_w, p_{w-1}) - d(p_{w-1}, p_{w-2}) \right|^2 \quad (10)$$

where p_w and $d(\cdot,\cdot)$ are defined as previously.

The end point distance ε is a criterion that measures the distance between the desired end point and the end point reached in the simulation.

The power P is adopted in order to minimize the manipulator energy consumption. The key measure of this analysis is the average of the mechanical energy during the total trajectory time T (Silva and Machado 1999). It is computed assuming that power regeneration is not available by motors doing negative work, that is, by talking the absolute value of the power. Therefore, the power consumption is calculated by the formula:

$$P_a = \frac{1}{T} P = \frac{1}{T} \sum_{j=1}^{n} \sum_{i=1}^{k} \left| \tau_j \Delta\theta_{ji} \right| \quad (11)$$

The points that are not admissible give a conflict measure between the robot and the obstacles. In this perspective, the nap value is evaluated as follows: each manipulator link is divided in p equal parts ($p = (4, 3, 2)$ for the 2, 3 and 4 link manipulators). Therefore, the nap value is a criterion consisting on the sum of the manipulator points that are inside the obstacles.

6 SIMULATION RESULTS

This section presents the results of several simulations. The experiments consist on moving a robotic arm from the starting point $A \equiv (1, 1)$ up to the final point $B \equiv (-0.6697, 1.6168)$. The initial configuration of the $2R$ robot is $(q_1, q_2) = (0, 90°)$. The $3R$ and $4R$ robots adopt the configurations $(q_1, q_2, q_3) = (10.9°, 34.1°, 34.1°)$ and $(q_1, q_2, q_3, q_4) = (0°, 0°, 90°, 0°)$, respectively, because they have a 'geometric resemblance' that makes easier the result comparison. Moreover, the simulations are divided into two groups: workspace without obstacles and workspace with obstacles.

The algorithm adopts crossover and mutation probabilities of $p_c = 0.8$ and $p_m = 0.1$ respectively, $k_r = 1.8$, $k_m = 1.8$, $\alpha = (2/k, 1/k, , 2, 1, 50, 0.0001)$ and a 100-string population. For the experiment are used string lengths of $l = 16$ and the selection operator is based on tournament selection with elitism.

6.1 WORKSPACE WITHOUT OBSTACLES

The $2R$ and $3R$ robots (Figs 1-4 and Figs 5-8, respectively) are firstly tested in a workspace without obstacles and afterwards with one and two obstacles.

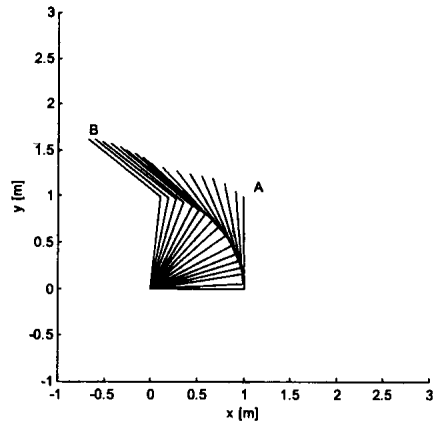

Figure 1: Successive configurations for a trajectory of the $2R$ robot and a workspace without obstacles.

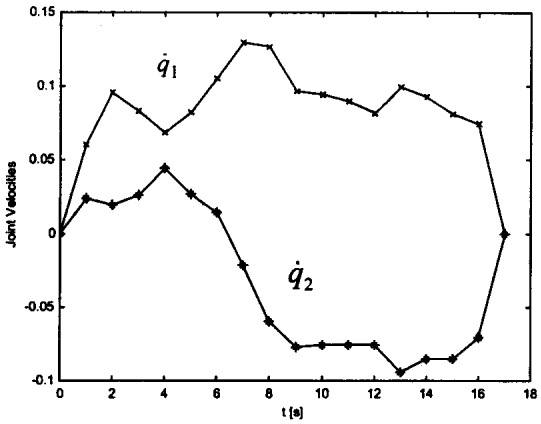

Figure 2: Joint velocities *versus* time for the $2R$ robot.

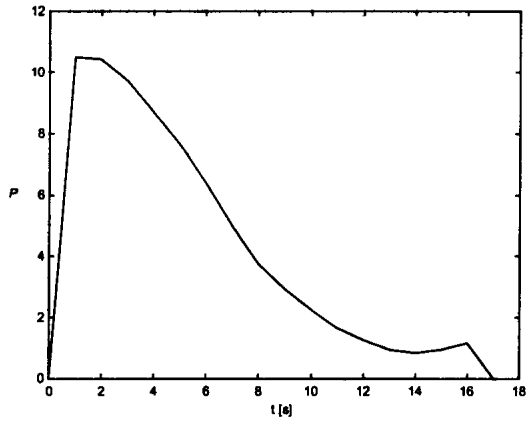

Figure 3: $P(t)$ for the $2R$ robot.

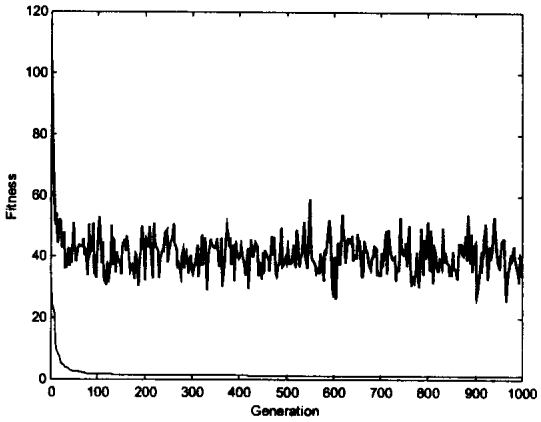

Figure 4: The best individual evolution and the fitness mean evolution versus generation for the $2R$ robot.

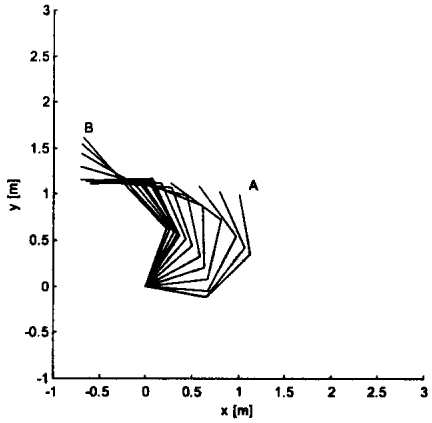

Figure 5: Successive configurations for a trajectory of the $3R$ robot and a workspace without obstacles.

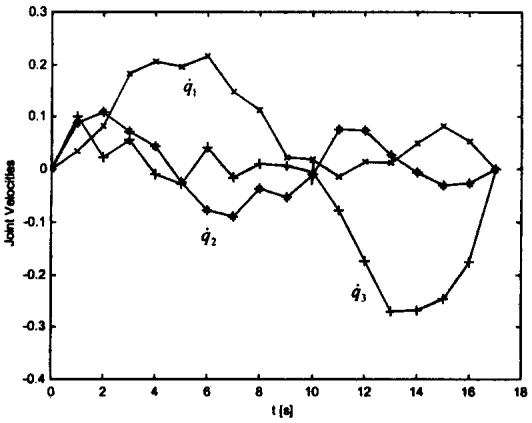

Figure 6: Joint velocities *versus* time for the $3R$ robot.

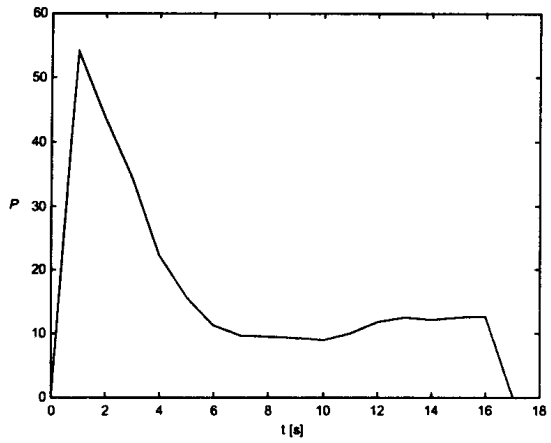

Figure 7: $P(t)$ for the $3R$ robot.

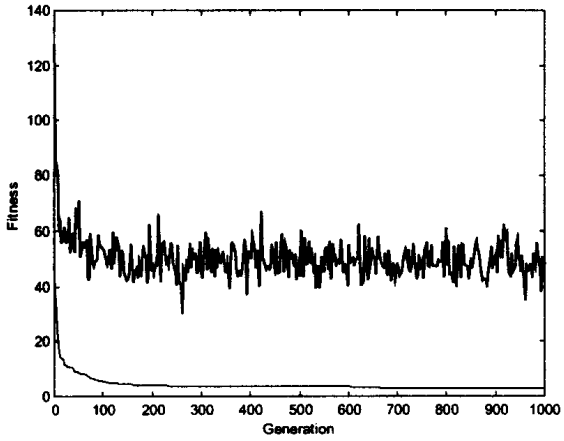

Figure 8: The best individual evolution and the fitness mean evolution versus generation for the $3R$ robot.

The total 'absolute' energies $E=PT$ required by the $2R$, $3R$ and $4R$ manipulators to reach the final point are 1259.5 J, 4936.5 J and 1986.6 J, respectively.

The results are satisfactory because the robot approaches the desired position without trajectory 'oscillations' and the time evolution of the variables presents a small ripple.

6.2 WORKSPACE WITH OBSTACLES

This section presents the manipulator trajectories for one and two obstacles in the workspace. The obstacles consist on one rectangle, with the upper left corner and the lower right corner with co-ordinates (0.3, 0.2) and (0.75, 0.8), respectively, and one circle, with center at (−0.7, −0.7) and radius 0.6.

The results for a $2R$ manipulator and one obstacle are shown in Figs 9 to 12. For a workspace with two obstacles the $2R$ robot can not reach the goal point because it is not physically possibly to pass between the obstacles.

For example, the results for a $4R$ robot and one obstacle workspace are shown in Figs 13 and 14.

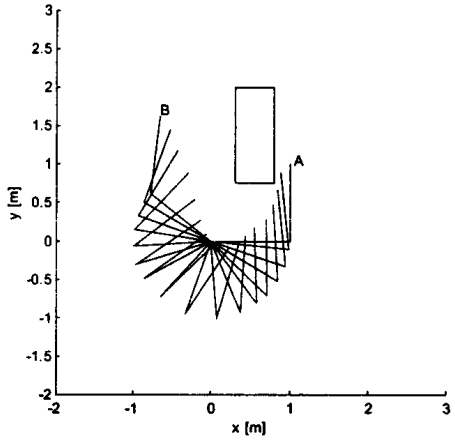

Figure 9: Successive configurations for a trajectory of the $2R$ robot and a workspace with a rectangular obstacle.

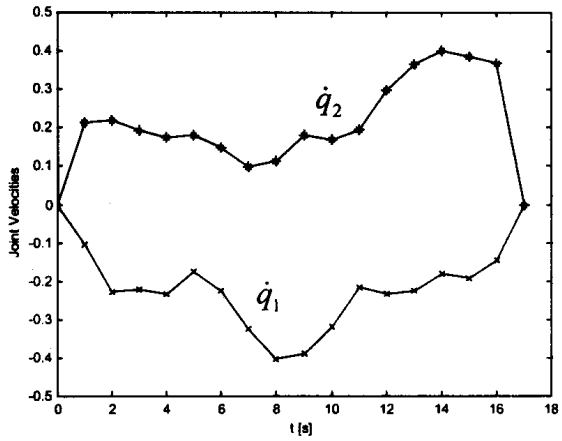

Figure 10: Joint velocities *versus* time for the $2R$ robot

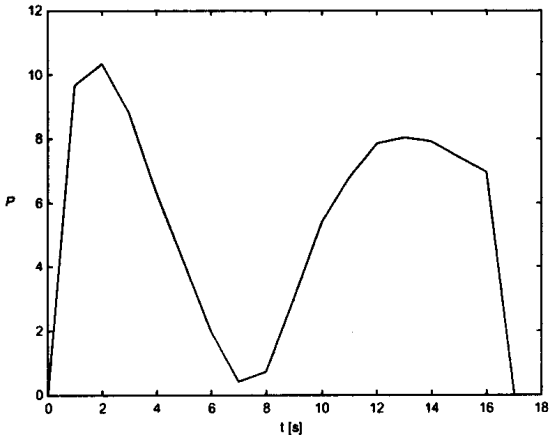

Figure 11: $P(t)$ for the $2R$ robot

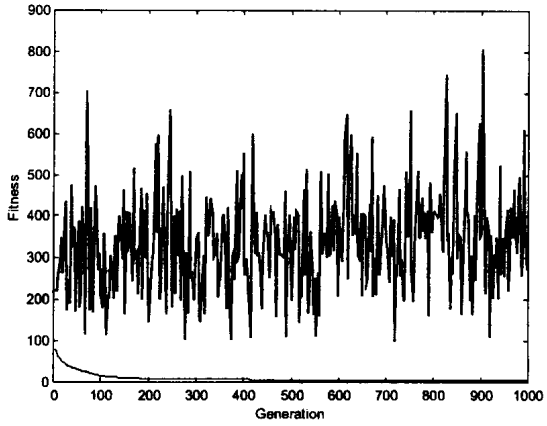

Figure 12: The best individual evolution and the fitness mean evolution versus generation for the $2R$ robot.

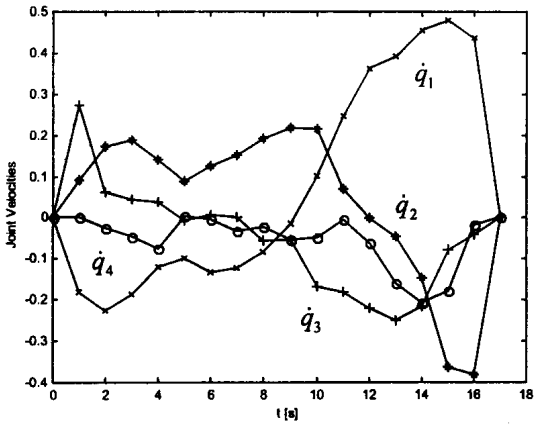

Figure 13: Joint velocities *versus* time for the $4R$ robot

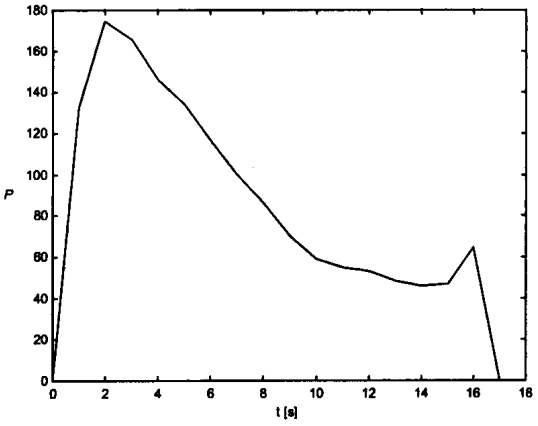

Figure 14: $P(t)$ for the $4R$ robot

The 'absolute' energies E required for the $2R$, $3R$ and $4R$ manipulators to reach the end points are 1.6 kJ, 1.3 kJ and 2.6 kJ, respectively, which reveals a high dependence with the number of robot dof.

For the $3R$ and $4R$ robots and a workspace with two obstacles, the results are shown in Figs. 15-19. The energies are 1 kJ and 3.1 kJ, respectively.

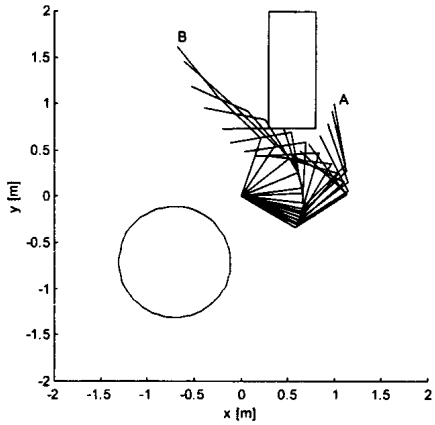

Figure 15: Successive configurations for a trajectory of the $3R$ robot and a workspace with two obstacles.

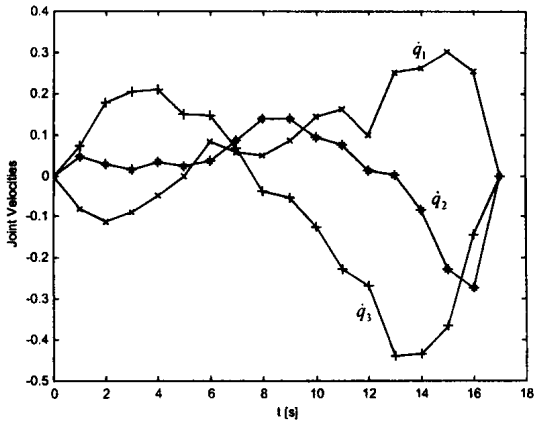

Figure 16: Joint velocities *versus* time for the $3R$ robot

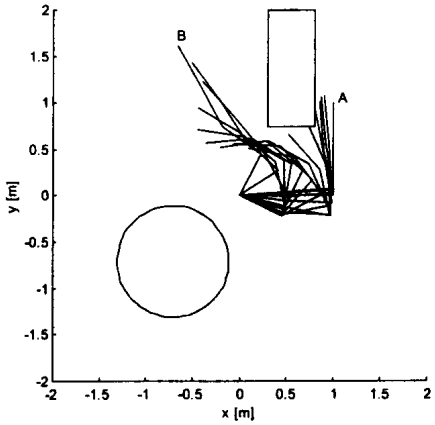

Figure 17: Successive configurations for a trajectory of the $4R$ robot and a workspace with two obstacles.

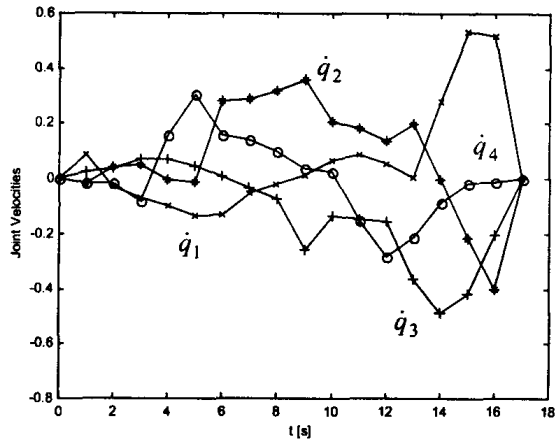

Figure 18: Joint velocities *versus* time for the 4*R* robot

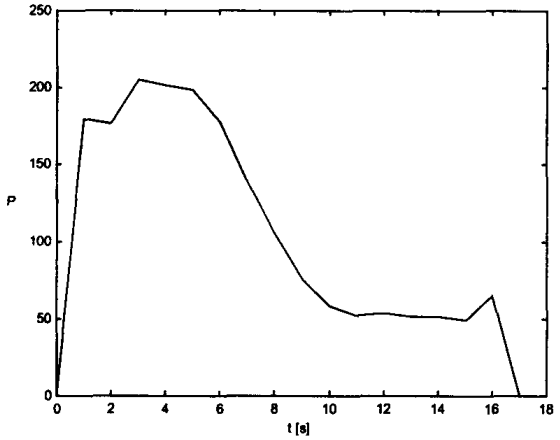

Figure 19: $P(t)$ for the 4*R* robot.

In a workspace with obstacles, the larger the number of manipulator links the better the ability to maneuver and reach the desired points. Furthermore, in order to obtain a small ripple in the velocities time evolution, a term related with the incremental accelerations must be also included in the fitness function. On the other hand, for actuator driving electronics with low efficiency, namely without the capability of regenerative braking, the larger the number of dof the higher the total robot energy consumption.

7 CONCLUSIONS

An off-line *GA* trajectory planner for robots, based on the kinematics and the dynamics was presented. The algorithm is able to reach a determined goal with a reduced ripple both in the space trajectory and the time evolution. Moreover, any obstacles, in the workspace do not represent a difficulty for the algorithm to reach the solution. Since the *GA* uses the direct kinematics the singularities do not constitute a problem. Furthermore, the algorithm is easily generalized for redundant robots.

In what concerns the dynamics, it was adopted an index based on the actuator power consumption for a non-regenerative braking. In this case the power consumption is highly dependent on the number of dof. Therefore, to take advantage of the robot redundancy, a more efficient driving electronics is required. In this perspective, other energy-inspired criteria are under investigation.

References

A.B. Doyle, D.I Jones (1996), *Robot Path Planning with Genetic Algorithms*, 2nd Portuguese Conf. on Automatic Control, pp. 312-218, Porto, Portugal.

A. Rana, A. Zalzala (1996), *An Evolutionary Planner for Near Time-Optimal Collision-Free Motion of Multi-Arm Robotic Manipulators*, Int. Conf. Control, vol 1, pp 29-35

David E. Goldberg (1989), *Genetic Algorithms in Search, Optimization, and Machine Learning*, Addison - Wesley.

Filipe Silva, J. Tenreiro Machado (1999), *Energy Analysis During Biped Walking*, Proc. IEEE Int. Conf. Robotics and Automation Detroit, Michigan, pp. 59-64.

Mingwu Chen, Ali M. S. Zalzala (1997), *A Genetic Approach to Motion Planning of Redundant Mobile Manipulator Systems Considering Safety and Configuration*, J. Robotic Syst. vol. 14, n. 7, pp. 529-544.

Naoyuki Kubota, Takemasa Arakawa, Toshio Fukuda (1997), *Trajectory Generation for Redundant Manipulator Using Virus Evolutionary Genetic Algorithm*, IEEE Int. Conf. on Robotics and Automation, pp. 205-210, Albuquerque, New Mexico.

Q. Wang, A. M. S. Zalzala (1996), *Genetic Control of Near Time-optimal Motion for an Industrial Robot Arm*, IEEE Int. Conf. on Robotics and Automation, pp. 2592-2597, Minneapolis, Minnesota.

Thomas Bäck, Ulrich Hammel, Hans-Paul Schwefel (1997), *Evolutionary Computation: Comments on the History and Current State*, IEEE Trans. on Evolutionary Computation, Vol. 1, n. 1, pp. 3-17, April.

Yaval Davidor (1991), *Genetic Algorithms and Robotics, a Heuristic Strategy for Optimization*, World Scientific.

Z. Michalewicz (1996), *Genetic Algorithms + Data Structures = Evolution Programs*, Springer-Verlag.

E. J. Solteiro Pires, J. Tenreiro Machado (1999), *A Trajectory Planner for Manipulators Using Genetic Algorithms*, IEEE Int. Symp. on Assembly and Task Planning, pg. 163-168, Porto, Portugal, July.