View metadata, citation and similar papers at core.ac.uk
brought to you by CORE
provided by Repositório Científico do Instituto Politécnico do Porto

eunite
2001

European Symposium on Intelligent Technologies, Hybrid Systems and their Implementation on Smart Adaptive Systems

13 -14 December 2001
Puerto de la Cruz, Tenerife, Spain

organized by the European Network of Excellence on Intelligent Technologies for Smart Adaptive Systems -
funded by the Information Society Technologies Programme (IST) within the European Union's Fifth RTD Framework Programme

Structure and Trajectory Optimization for Redundant Manipulators

E. J. Solteiro Pires, J. A. Tenreiro Machado*, P. B. de Moura Oliveira
Universidade de Trás-os-Montes e Alto Douro
Departamento de Engenharias, 5001-911 Vila Real, Portugal
email: {epires,oliveira}@utad.pt
* Instituto Superior de Engenharia do Porto
Dep. de Engenharia Electrotécnica, 4200-072 Porto, Portugal
email: jtm@dee.isep.ipp.pt

ABSTRACT: This paper proposes a genetic algorithm to generate a robot structure and the required manipulating trajectories. The objective is to minimize the space/time ripple in the trajectory without colliding with any obstacles in the workspace, while optimizing the mechanical structure.

KEYWORDS: Genetic Algorithms, Robotics, Optimization

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 and speech recognition. This paper addresses the generation of a robotic manipulator structure and the planning of trajectories, namely in finding a continuos motion that takes the hand from a given starting configuration, without collision with any obstacle, up to a desired end position in the workspace.

Various methods for trajectory planning, collision avoidance and manipulator structure definition have been proposed. A possible approach consists in adopting the differential inverse kinematics, using the Jacobian matrix, for generating the manipulator trajectories [1],[2]. 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 [3], [4], [5].

Davidor [2] also applies *GA*s to the trajectory generation by searching the inverse kinematics solutions to pre-defined end-effector robot paths.

Kubota *et al.* [3] 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 [4] 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.

Chocron and Bidaud [6] proposes an evolunionary algorithm to perform a task-based design of modular robotic systems. The system consists in a mobile base and an arm that may be built with serially assembled link and joint modules. The optimization design is evaluated with geometric and kinematic performance measures.

Kim and Khosha [7] presents the design of a manipulator that is best suited for a given task. The design consists of determining the trajectory and the length of a three dof manipulator.

Han *et al* [8] describe a design method of a modular manipulator. The method uses the kinematic equations to determine the robot configuration and, in a second phase, adopts a *GA* to find the optimal length.

In this line of thought, this paper proposes a method to obtain a robot arm and its path. This method is based on a *GA* adopting the direct kinematics. The optimal manipulator is the one that minimizes both the path trajectory length and the ripple in the time evolution, without any collision with the obstacles in the workspace.

Bearing these facts in mind, this paper is organized as follows. First is introduced the problem and the *GA*-based method for its resolution. Followed by, the descrition of the adopted algorithm, the solution representation, the *GA* operators used in the problem and the optimization criteria, respectively. Based on this formulation, are presented the results for several simulations involving different robot structures and trajectories in the workspace. Finally, are outlined the main conclusions.

PROBLEM AND ALGORITHM FORMULATION

In this study we consider *R* robotic manipulators that are required to move from an initial point up to a given final point. In the experiments we adopt 1 up to 4 *dof* planar manipulators with rotational joints. The arm link lengths are in the range [0, 4] *m*, and the robot rotational joints are free to rotate 360º. Therefore, the manipulator workspace is a circle with a 16 *m* maximum radius, that may have obstacles such as rectangles and circles. To test a possible collision between the manipulator and the obstacles, the arm structure is discretized into several points and then these points are checked in order to verify if they are inside any obstacle.

In what concern the structure generator, it is adopted a *GA* to search for a global optimal robot which presents the best performance. The mechanical structure consists of a set of strings that represent the link lengths. On the other hand, the trajectory generator uses a *GA* scheme to search for an optimal robot path. The trajectory consists in a set of strings that represent the joint positions between the initial and final robot configurations.

In conclusion, in this work are adopted a hierarchical *GA*. The main *GA* is used to calculate the robot's structure. For each arm two *GAs* are used to calculate the initial and final configurations of the trajectory. Finally, another *GA* determines the intermediate configurations between the two configurations calculated previously.

REPRESENTATION

The robotic structure is encoded as:

$$S = \{l_1, \ldots, l_i, \ldots, l_k\} \tag{1}$$

where l_i is the *i*th link length, in the range [0, 4] *m*. In order to limit the computational time the number of *dof* is limited to $k \leq 4$. All values used in this work are encoded through real values. The initial and the final configuration are encoded as:

$$[q_1, \ldots, q_k] \tag{2}$$

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

$$[(q_{11}, \ldots, q_{k1}), \ldots, (q_{1j}, \ldots, q_{kj}), \ldots, (q_{1n}, \ldots, q_{kn})] \tag{3}$$

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. The values of q_{ij} are initialized in the range]360º,+360º]. It should be noted that the initial and final configurations have not been encoded into the string because this configuration remains unchanged throughout the trajectory search. Without losing generality, for simplicity, it is adopted a normalized time of $\Delta t = 1$ sec between two consecutive configurations, because it is always possible to perform a time re-scaling.

OPERATORS IN THE GENETIC ALGORITHM

The initial populations of strings are generated at random. The search is then carried out among these populations. 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 [9] 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. Single crossover is then performed among pairs. The crossover point is only allowed between genes (*i.e.* the crossover operator may not disrupt genes). The mutation operator consists on several actions namely, modifying the link length and changing the joint variable. Therefore, the mutation operator replaces one gene value with a given probability that follows the equations:

$$q_{ij}(t+1) = q_{ij}(t) + \mathrm{k_m}\, \varphi_i \tag{4a}$$

$$l_i(t+1) = l_i(t) + \mathrm{k_m}\, \psi_i \tag{4b}$$

$$\{\varphi_i, \psi_i\} \sim \mathrm{U}[-1; 1] \tag{4c}$$

at generation *t*, while φ_i, ψ_i are uniform random numbers and $\mathrm{k_m}$ a parameter.

Finally, at the end of each GA structure iteration two operators take into action, randomly, over the l_i genes. One duplicates a given gene while the other removes another gene, with probabilities p_r and p_d, respectively.

EVOLUTION CRITERIA

Several criteria have been selected to qualify the evolving robotic manipulators. 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 [10].

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

$$f = \beta_1 f_T + \beta_2 f_I + \beta_3 f_F \tag{5}$$

$$f_T = \alpha_1 \dot{q} + \alpha_2 \ddot{q} + \alpha_3 \dot{p} + \alpha_4 \ddot{p} + \alpha_5 . nap \tag{6}$$

where β_i (i=1,2,3) are weighting factors. The f_I and f_F functions give a measurement of the distance between the initial or final desired point and the point actually reached by the robot configuration. The $\dot{q}$, $\ddot{q}$, $\dot{p}$, $\ddot{p}$ 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 β_i (i = 1,…,4).

The joint velocities $\dot{q}$ are used to minimize the manipulator traveling distance yielding the criteria:

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

This equation is used to optimize the traveling distance because if the curve length is minimized, then the ripple in the space trajectory is indirectly reduced. For a function $y = g(x)$ the distance curve length is $\int[1 + (dg/dt)^2]\, dx$ and, consequently, to minimize the distance curve length it is adopted the simplified expression $\int(dg/dt)^2\, dx$. The fitness function maintains the quadratic terms so that the robot configurations are uniformly distributed between the initial and final configurations.

The joint accelerations $\ddot{q}$ are used to minimize the ripple in the time evolution of the robot trajectory through the criteria:

$$\ddot{q} = \sum_{j=1}^{n} \sum_{i=1}^{k} \ddot{q}_{ij}^{\,2} \tag{8}$$

The Cartesian velocities $\dot{p}$ are 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:

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

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 $\ddot{p}$ in the fitness functions is responsible for reducing the ripple in time evolution of the arm velocities. This criteria is formulated as:

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

The points that are not admissible give a conflict measure. In this perspective, each manipulator link is discretized and the *nap* value is a criterion consisting on the sum of the manipulator points that are inside the obstacles.

SIMULATION RESULTS

In this section are presented the results of several simulations. The experiments consist on moving a robotic arm from the starting point A up to the final point B (Table I), for two types of situations:

- the algorithm optimizes the robot structure for a sequence of r trajectories (series optimization), tacking each trajectory at a time;
- the algorithm optimizes the robot structure for the r trajectories (parallel optimization), considering all trajectories

simultaneously.

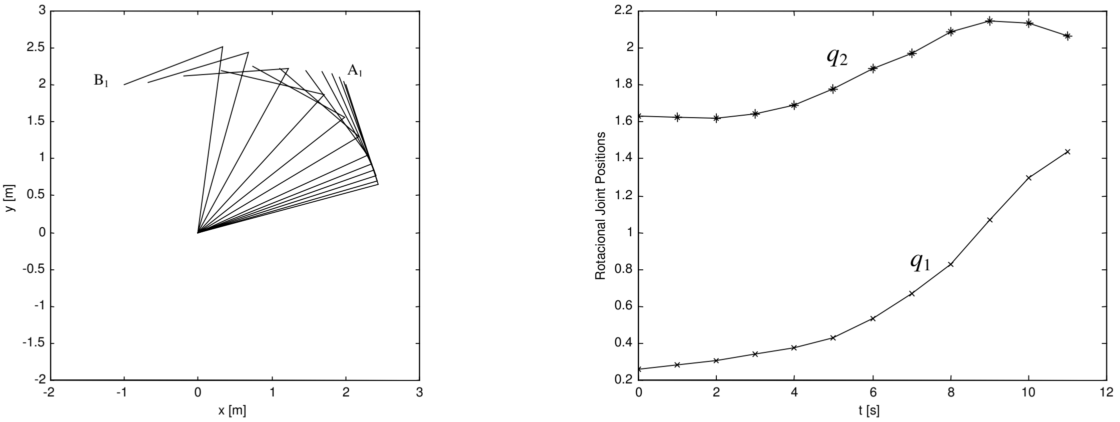
Table I: Trajectory simulations

Trajectory	Initial point A	Final point B
1	(2, 2)	(−1, 2)
2	(−1, 2)	(1, 1)
3	(−0.5, 0)	(−1, 0)
4	(3, −1)	(1.5, −1)

The algorithm adopts crossover and mutation probabilities of $p_c = 0.6$ and $p_m = 0.1$ respectively, $p_r = p_d = 0.1$, $k_m = 1.8$, a 50-string population for the robots, a 50-string population for the initial and final configurations and a 100-string population for the intermediate configurations. For the experiment are used strings length of $n = 10$ and the selection operator is based on tournament selection with elitism. The workspace may contains an obstacle, a circle with center at the point (0, 2) and radius 0.5.

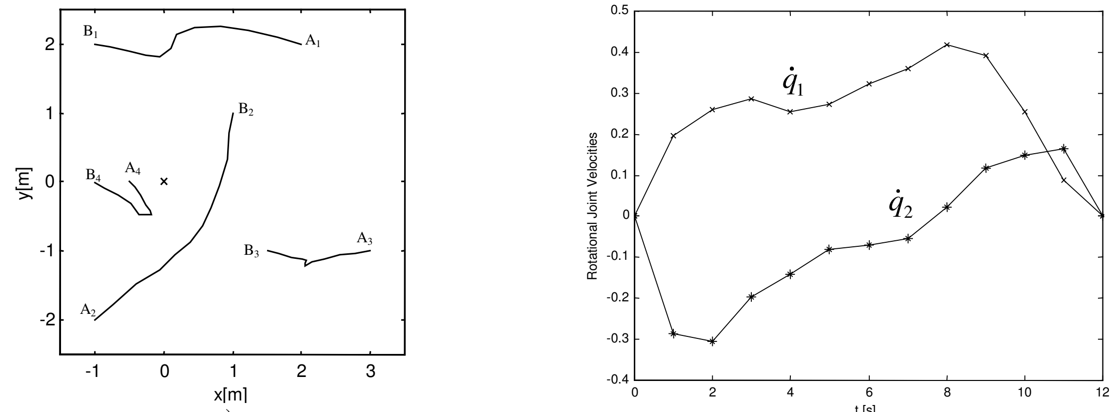
ONE TRAJECTORY OPTIMIZATION

For one trajectory only, there is no distinction between the series and parallel optimization methods. Therefore, this section presents the results of trajectory 1 optimization yielding a $2R$ manipulator with structure $S \equiv \{2.5324, 1.4220\}$. Figure 1 show some results of the robotic manipulator obtained.



a) b)

Figure 1: a) Successive robot configurations for trajectory 1, b) Joint positions *versus* time.



a)

b)

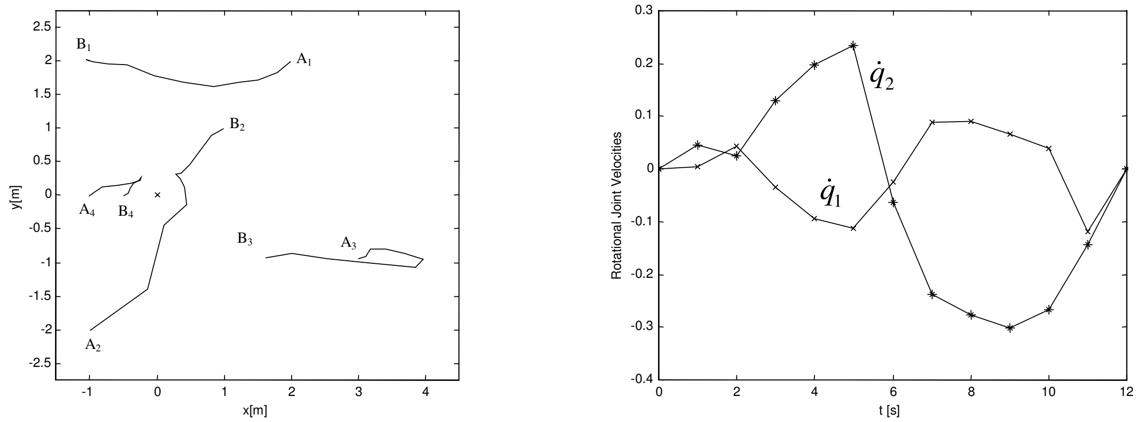
Figure 2: a) Robot hand trajectories, b) Joint velocities *versus* time of trajectory 2.

SEVERAL TRAJECTORIES SERIES OPTIMIZATION

This section presents the results when optimizing sequentially the group of trajectories, one trajectory at a time. The robot structures obtained for the five trajectories are: $S_1 \equiv \{2.7326, 1.4446\}$, $S_2 \equiv \{1.6166, 1.4446\}$, $S_3 \equiv \{1.6752, 1.4446\}$, $S_4 \equiv \{2.3650, 1.4446\}$. The results are shown in Figure 2.
The best individual function exhibits an abrupt transitions due to the change of the optimization trajectory. In same cases the step is negative because the length of the new trajectory is smaller.

SEVERAL TRAJECTORIES PARALLEL OPTIMIZATION

This section presents the resultant robot when optimizing the four trajectories simultaneously. The final 2 *R* robot mechanical structure is $S \equiv \{2.6810, 2.2563\}$. Figure 3 show the results for this case.



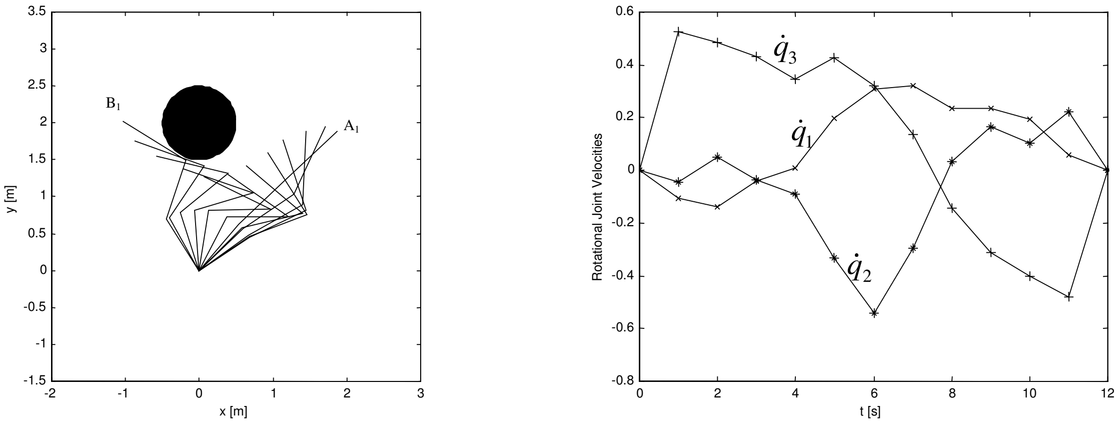
a)

b)

Figure 3: a) Terminal arm position for the trajectories, b) Joint position of trajectory 4.

ONE OBSTACLE TRAJECTORY OPTIMIZATION

This section shows the results when optimizing the trajectory 1 for a workspace with a circle obstacle. The resulting 3 *R* robot structures obtained is $S \equiv \{0.8239, 0.8369, 1.0000\}$. The time evolution of the joint velocities are shown in Figure 4.



a)

b)

Figure 4: a) Successive robot configurations, b) Joint velocities *versus* time.

RESULT ANALYSIS

The results are satisfactory because the solutions avoid the obstacles, and the time evolution of the variables presents a small ripple. Moreover, analyzing the final number of axis, in a workspace with obstacles, the larger the number of manipulator links the better the ability to maneuver and reach the desired points.
In the present form, the study does not consider energy requirement [10]. In this line of thought, future work will take into account the robot dynamics and we expect to have a clearest conclusions about the total number of *dof* and its links length. For more obstacles in the workspace the convergence seems more difficult and further experiments are still required.

SUMMARY AND CONCLUSIONS

A *GA* robot constructor and its trajectory planner, based on the kinematics, were 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 determines the robot structure more adaptable to a given number and type of tasks, maintaining good manipulating performances.

REFERENCES

[1] Mingwu Chen, Ali S. Zalzala, "A Genetic Approach to Motion Planning of Redundant Mobile Manipulator Systems Considering Safety and Configuration", J. Robotic System. vol. 14, n. 7, pp. 529-544, 1997.

[2] Yaval Davidor, Genetic Algorithms and Robotics, a Heuristic Strategy for Optimization, World Scientific, 1991.

[3] Naoyuki Kubota, Takemasa Arakawa, T. Fukuda, "Trajectory Generation for Redundant Manipulator Using Virus Evolutionary Genetic Algorithm", IEEE Int. Conf. on Robotics and Automation, pp. 205-210, Albuquerque, 1997.

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

[5] Q. Wang, A. M. S. Zalzala, "Genetic Control of Near Time-optimal Motion for an Industrial Robot Arm", IEEE Int. Conf. on Robotics and Automation, pp. 2592-2597, Minneapolis, USA, 1996.

[6] O. Chocron, P. Bidaud, "Evolutionary Algorithms in Kinematic Design of Robotic system", IEEE/RSJ Int. Conf. on Intelligent Robotics and Systems, pp. 279-286, Grenoble, France, 7-11 September 1997.

[7] Jin-Oh Kim, Pradeep K. Khosla, "A Multi-population Genetic algorithm and its Application to Design of Manipulators" IEEE Int. Conf. on Intelligent Robotics and Systems, pp. 279-286, Raleight, USA, 7-10 July 1992.

[8] J. Han, W. K. Chung, Y. Youm, S. H. Kim, "Task Based Design of Modular Robotic Manipulator using Efficient GA" IEEE Int. Conf. on Robotics and Automation, pp. 507-512, Albuquerque, USA, 20-25 April 1997.

[9] David E. Goldberg, Genetic Algorithms in Search, Optimization, and Machine Learning, Addison – Wesley, 1989.

[10]E. J. S. Pires, J. Tenreiro Machado, "A *GA* Perspective of the Energy Requirements for Manipulators Maneuvering in a Workspace with Obstacles", Congress on Evolutionary Computation, pg. 1110-1116, San Diego, USA, 2000.

[11]E. J. Solteiro Pires, J. A. Tenreiro Machado, "Trajectory Optimization for Redundant Robots Using Genetic Algorithms", Proc. of the Genetic and Evolutionary Computation Conference, pg. 967, July 2000, Las Vegas, USA.

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

[13]Z. Michalewicz, Genetic Algorithms + Data Structures=Evolution Programs, Springer-Verlag, 1996.