Inverse Kinematics of Redundant Robotic Manipulators Trajectories Using Genetic Algorithms

Munif Gebara Junior, Alfranci Freitas dos Santos, Heitor Silvério Lopes

Centro Federal de Educação Tecnológica do Paraná / CPGEI
Av. 7 de setembro, 3165
80230-901 Curitiba (PR) – Brazil
jackal@bsi.com.br, {freitas,hslopes}@cpgei.cefetpr.br

Abstract: The problem of inverse kinematics of manipulators is always present in robotics. The solution of this problem, particularly for redundant manipulators, is difficult because it includes nonlinear equations and an extremely large number of feasible solutions for most end-effector positions. This paper presents a Genetic Algorithm (GA) to solve this problem for trajectories. The results show that GA can be useful for real-world applications.

Keywords: genetic algorithms, robotic manipulator, inverse kinematics, PUMA

1. INTRODUCTION

The inverse kinematics is a difficult problem: given the desired position and orientation of the end-effector, compute the set of joint angles that will achieve this position and orientation. The solution of this problem is difficult because it includes nonlinear equations and transcedental functions. In some cases this problem has only iterative solutions. For inverse kinematics of trajectories, this problem becomes even more difficult because the inverse kinematics have to be repeated for every point of a trajectory. Hence, the use of alternative methods, like GA, for the resolution of this problem is justified [2] [4].

This work presents the solution of the inverse kinematics of trajectories using GA for two cases: a non-redundant and a redundant manipulator.

2. BACKGROUND

2.1 Genetic Algorithms

Genetic Algorithms - GAs are a search method that has been widely used in applications where the size of the search space is very large. In essence, GAs are "search algorithms based on the mechanics of natural selection and natural genetics" [3]. GAs are inspired on the principle of survival of the fittest, where the fittest individuals are selected to produce offspring for the next generation. In the context of search, individuals are candidate solutions to a given search problem. Hence, reproduction of the fittest individuals means reproduction of the best current candidate solutions. Genetic operators such as selection, crossover and mutation are applied to the fittest individuals to generate. One of the advantages of GAs over "traditional" search methods is that the former performs a kind of global search using a population of individuals, rather than performing a local, hill-climbing search. Global search methods are less likely to get trapped into local maxima, when compared to local search methods.

2.2 Forward kinematics, inverse kinematics and manipulator workspace

A robotic manipulator may be thought of a set of bodies chained by joints. Those bodies are called links. Each joint form a connection between a neighboring pair of links. The manipulator extremity is called end-effector.

A configuration of the manipulator is a set of values that represent the angles between pairs of consecutive links. The forward kinematics maps the joint angles of the manipulator into the Cartesian coordinates of position and orientation of the end-effector. Similarly, the inverse kinematics maps the Cartesian coordinate position and orientation of the end-effector into the joint angles of the manipulator. The set of all points reachable by the manipulator is called manipulator workspace.

The forward kinematics always has a unique solution. But the inverse kinematics can result in none, just one, several, or even infinite solutions.

A detailed discussion of the subjects of this subsection is found in [1].

3. METHODOLOGY

In this paper, only the inverse kinematics for the end-effector positioning is addressed. The inverse kinematics for orientation will be left for future work. In this work, two case studies are done using a well-known non-redundant manipulator, the PUMA 560.

In the first case study the objective is to experimentally adjust the GA for the problem. It is particularly concerned about the chromosome encoding, the fitness evaluation and the constraints.

The results of this study are compared with the results of traditional technique [1].

In the second case study, the basic methodology for applying GAs is used, but the manipulator is modified. The modification is the inclusion of a link in the extremity of the manipulator that transforms it in a redundant manipulator for positioning. The objective here is investigate the applicability of GA to this kind of problem.

For both cases, the inverse kinematics problem is repeated for all points of four different trajectories. It is supposed that those consecutive points are near. For each point (p_1, p_2, ..., p_n) the GA is run until a desired precision is reached. After the point p_k is reached, the GA uses the "knowledge" in the population to find the next point p_{k+1}. The first point of the trajectory is searched in whole manipulator workspace, thus it is expected to be rather difficult. After that, the search space is restricted to a region surrounding the last point found. The restriction of the search space is accomplished by means a modification in the joint limits, that will be discussed later. This makes the GA runs faster, and an experiment was done to prove this.

3.1 The genetic algorithm

3.1.1 Chromosome encoding and constraints

The variables of the problem are the joint angles, shown in figure 1.The chromosome encoding for the non-redundant manipulator is detailed in table 1 and for the redundant manipulator, in table 2. The precision required is 0.01 degree, an acceptable value for real problems.

Figure 1: Joint angles.

Gray code was used instead of natural binary code, in order to avoid disruptive mutations [5]. It was empirically observed that GAs using the Gray code converged to a accept solution faster than GAs that use the natural binary code.

Table 1: Variable coding for non-redundant manipulator.

Variable	Min	Max	Bits	Precision
Tetha1	-170^o	170^o	16	0.00519^o
Tetha2	-225^o	45^o	16	0.00412^o
Tetha3	-250^o	75^o	16	0.00496^o

Table 2: Variable coding for redundant manipulator.

Variable	Min	Max	Bits	Precision
Tetha1	-170^o	170^o	16	0.00519^o
Tetha2	-225^o	45^o	16	0.00412^o
Tetha3	-250^o	75^o	16	0.00496^o
Tetha4	-135^o	135^o	16	0.00411^o
Tetha5	-100^o	100^o	16	0.00305^o

The search space is about 10^{14} cases for the non-redundant manipulator and about 10^{24} for the redundant manipulator. Although these search spaces are not a big challenge for GAs, it is important to emphasize that the search is repeated for every point of a trajectory. The four trajectories tested in this work have 251 points each.
For the non-redundant manipulator, the joint angles are obtained by:

$$q_i = Center_i + \frac{Value_i \cdot Range_i}{2^{bits} - 1} - \frac{Range_i}{2} \qquad [1]$$

where: q_i is the angle of joint i, $Center_i$ is the average value of q_i, $Value_i$ is the i-th value in the chromosome, $Range_i$ is the range of q_i and *bits* is the number of bits used to encode each gene. For example, if q_i can assume values ranging from 20 to 100, *Center* is 60 and *Range* is 80.

The joint rotation limits are handled directly in the coding, so any solution determined by GA is physically feasible.

To increase the speed of GA, the search space is restricted to a region near to the last point found. This operation is done by reducing the value of *Range* and changing the value of *Center*. The joint limits are checked to avoid coding of non-feasible joint values.

3.1.2 Objective function

Under this aproach the inverse kinematics problem is a minimization problem, It is aimed to minimize the distance between a desired point and the point obtained with forward kinematics using the joint angles decoded from the chromosome. The distance is computed by:

$$Distance = \sqrt{(p_x - p_x^*)^2 + (p_y - p_y^*)^2 + (p_z - p_z^*)^2} \quad [2]$$

where p_x^*, p_y^* and p_z^* are the coordinates of the desired position of the end-effector, and p_x, p_y and p_z are the coordinates of the point obtained with forward kinematics. For the non-redundant manipulator the equations of forward kinematics are:

$$\begin{aligned} p_x &= \cos_1\left[a_2 \cos_2 + a_3 \cos_{23} - d_4 \operatorname{sen}_{23}\right] - d_3 \operatorname{sen}_1 \\ p_y &= \operatorname{sen}_1\left[a_2 \cos_2 + a_3 \cos_{23} - d_4 \operatorname{sen}_{23}\right] + d_3 \cos_1 \\ p_z &= -a_3 \operatorname{sen}_{23} - a_2 \operatorname{sen}_2 - d_4 \cos_{23} \end{aligned} \quad [3]$$

Where d_3, d_4, a_2 and a_3 are constants of the manipulator, cos_1, cos_2 , cos_{23}, sin_1 sin_2 and sin_{23} are respectively $cos(\theta_1)$, $cos(\theta_2)$, $cos(\theta_1 + \theta_2)$, $sin(\theta_1)$, $sin(\theta_2)$ and $sin(\theta_1 + \theta_2)$.

For the redundant manipulator the equations of forward kinematics are more complex:

$$\begin{aligned} p_x &= a_2c_1c_2 + a_3c_1c_2c_3 - d_3s_1 - d_4c_1c_3s_2 - d_7c_1c_3c_5s_2 \\ &\quad - d_4c_1c_2s_3 - d_7c_1c_2c_5s_3 - a_3c_1s_2s_3 \\ &\quad - d_7c_1c_2c_3c_4s_5 + d_7c_1c_4s_2s_3s_5 - d_7s_1s_4s_5 \\ p_y &= d_3c_1 + a_2c_2s_1 + a_3c_2c_3s_1 - d_4c_3s_1s_2 - d_7c_3c_5s_1s_2 \\ &\quad - d_4c_2s_1s_3 - d_7c_2c_5s_1s_3 - a_3s_1s_2s_3 \\ &\quad - d_7c_2c_3c_4s_1s_5 + d_7c_4s_1s_2s_3s_5 + d_7c_1s_4s_5 \\ p_z &= -(d_4c_2c_3) - d_7c_2c_3c_5 - a_2s_2 - a_3c_3s_2 \\ &\quad - a_3c_2s_3 + d_4s_2s_3 + d_7c_5s_2s_3 + d_7c_3c_4s_2s_5 \\ &\quad + d_7c_2c_4s_3s_5 \end{aligned} \quad [4]$$

Where d_3, d_4, a_2 and a_3 are constants of the manipulator, c_i and s_i are respectively $cos(\theta_i)$ and $sin(\theta)$

The objective function is clearly nonstationary: the algorithm is executed until the distance is smaller than a given tolerance (0.002 in = 0.00508 cm), then, the values of px^*, py^* and pz^* are updated for the next point of the trajectory. This changes the environment in which the GA has to optimize, thus enforcing it to adapt the search to the new objective.

3.1.3 Fitness function

GAs, by default tries to maximize a given function. In order to turn the objective function into a maximization problem, the fitness function used is:

$$Fitness = C - Distance\,, \quad [5]$$

where C is a constant greater than the maximum distance between any points within the manipulator's workspace. For this study the value used is 100.

3.1.4 Stop criteria

There are two stop criteria. The first is when the complete joint trajectory is successfully generated. The second one is when the GA reaches the upper limit for generations. This limit is set to 10,000 for the non-redundant and 20,000 for the redundant manipulator. This last stop criterion enables the GA to stop by timing out in case of non-convergence. Notice that if the point searched isn't in the workspace, the convergence will not happen.

3.1.5 Genetic operators and parameters

The population size is 100. The genetic operators are uniform crossover and mutation, which are applied with probabilities of 1.00 and 0.01 respectively. The selection method used is tournament selection, with tourney-size 15. This method keeps reasonable diversity levels throughout generations without excessive selective pressure. A high genetic diversity in the population is necessary since the GA does not stop when finding a point in the trajectory, but only when all points are found.

3.2 Test trajectories

The trajectories described in table 3 are used to evaluate the GAs for the redundant and the non-redundant manipulators. Figure 2 illustrates these trajectories.

Table 3: Test trajectories

	N# points	Timing	Equations
Traj. #1	251	0 to 2.5 s step = 0.01 s	Px=7+t Py=7-t/2 Pz=-17+2*t
Traj. #2	251	0 to 5 s step = 0.02 s	Px=7+cos(t*3)*t/5 Py=7+sin(t*3)*t/5 Pz=-17+2*t
Traj. #3	251	0 to 2.5 s step = 0.01 s	Px=7+sin(cos(t*2)) Py=7+t*cos(t)/2 Pz=-7+2*t*cos(t)*sin(t)
Traj. #4	251	0 to 2.5 s step= 0.01 s	Px=8+3*cos(t*4) Py=8 Pz=-13+t

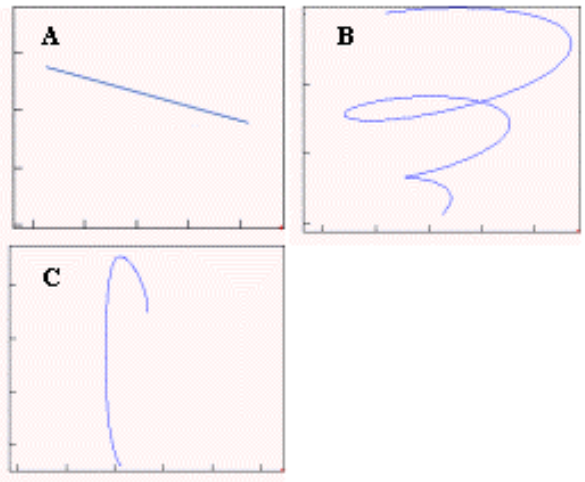


Figure 2: (A) Test trajectory #1, (B) test trajectory #2, (C) test trajectory #3.

4. RESULTS

Figure 3 shows the distance between each point of trajectory 1 and each point generated by the GA for the non-redundant manipulator, as generations goes by. The results are similar for the other trajectories and also for the redundant manipulator.

Table 4 presents the results of a run for the non-redundant manipulator case with and without range reduction. The number of generations necessary is indicated by *Gen*, the flag *Compl* indicates wheater the trajectory was completed or not, and *Gen/Pt* indicates the average number of generations to reach each point. *RR* means that GA was run with the range reduction technique. Table 5 presents the results for the redundant manipulator case.

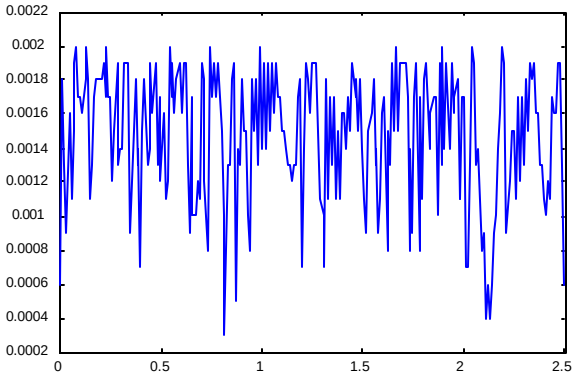


Figure 3: Distance (inches) versus time (s).

Table 4: Results of non-redundant manipulator.

		Gen	Compl	Gen/Pt
Traj. #1	**No RR**	9895*	18.7%	210
	RR	2000	100%	7
Traj. #2	**No RR**	9877*	16.3%	240
	RR	6678	100%	26
Traj. #3	**No RR**	9911*	19.9%	198
	RR	4245	100%	16
Traj. #4	**No RR**	9945	15.5%	255
	RR	8587	100%	34

*Did not reach the result after 10000 generations.

Table 5: Results of redundant manipulator.

	Gen.	Compl.	Gen/Pt
Traj. #1	7309	100%	29
Traj. #2	13137	100%	52
Traj. #3	10100	100%	40
Traj. #3	13817	100%	55

All runs reported here was done using a standard Pentium/300Mhz. For all trajectories of the non-redundant case, the running time was less than 1 minute. For the redundant manipulator a run took less than 3 minutes.

For the non-redundant manipulator, the first point of the test trajectories #1 and #2 was obtained in 359 generations, for test trajectory #3 it was necessary 335 generations and for test trajectory #4 it was necessary 356 generations. For the redundant manipulator, the first point of the test trajectories #1 and #2 was obtained in 689 generations, for test trajectory #3 it was necessary 902 generations and for test trajectory #4 it was necessary 402 generations.

The coincidence of the number of generations to obtain the first point of the trajectories 1 and 2 is due to the fact that the first points of these trajectories are the same (7, 7, -17).

5. DISCUSSION AND ANALYSIS

Figure 3 shows that the distance between a desired point and the point found by GA oscillated between the specified tolerance and very small values (of order 10^{-4}), thus giving an extremely satisfactory precision in the positioning.

The trajectories generated for the non-redundant robot were compared with the exact results obtained by means of the conventional method for inverse kinematics (closed form solution) and they were indistinguishable except by the tolerance value.

As expected, the number of generations to obtain the first point of a trajectory was much larger than the number of generations needed to reach the other points. An interesting result is that, in the non-redundant manipulator case, in approximately 25% of the points, the result was obtained with only one generation, what indicates that the point was already in the previous population.

As presented in table 4, the use of range reduction caused a sensible acceleration of the GA, finding points of the trajectory with approximately one tenth of the number of generations necessary without the use of this technique.

6. CONCLUSIONS AND FUTURE WORK

Using GAs, successful joint trajectories were generated for both redundant and non-redundant manipulators, in three different test trajectories.

The first case study, using a manipulator for which a closed form solution was known, was important in order to conveniently adjust the GA, and compare results. The working hypothesis that the same parameters of the GA working for the first case study would work as well for the second, was found to be valid. One could infer that this methodology of using GAs can also be valid for other similar problems. For manipulators (particularly redundant ones) for which the inverse kinematics is unknown or too complex to devise, the use of AGs in the way shown in this work is certainly a valid alternative.

It was shown that using the range reduction technique, the GA was dramatically accelerated.

Further work can address other speed improvements in the basic algorithm aiming to generate trajectories in real-time. Among these, specific operators as well as parallel processing can be cited. Future work will cover more complex problems of manipulators, including the generation of trajectories optimized according to some criterion such as minimization of joint movements or singular points avoidance.

7. REFERENCES

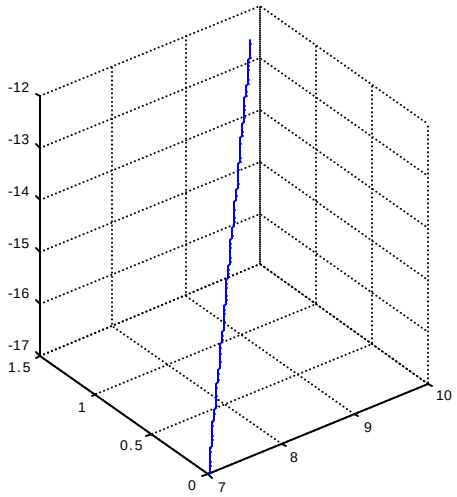
[1] Craig, J.J., *Introduction to Robotics Mechanics and Control*. . Reading, MA: Addison-Wesley, 1989.

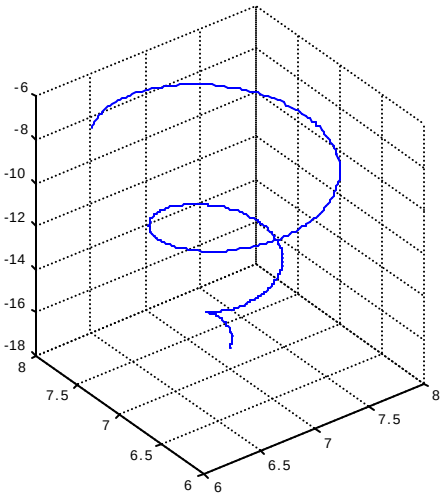
[2] Da Silveira, C.H., Coelho, L.S. & Campos, M.F.M., The use of genetic algorithms for the evaluation of inverse kinematics of manipulators, *Anais do 1° Congresso Brasileiro de Redes Neurais*, Itajubá, Brasil, 1994.

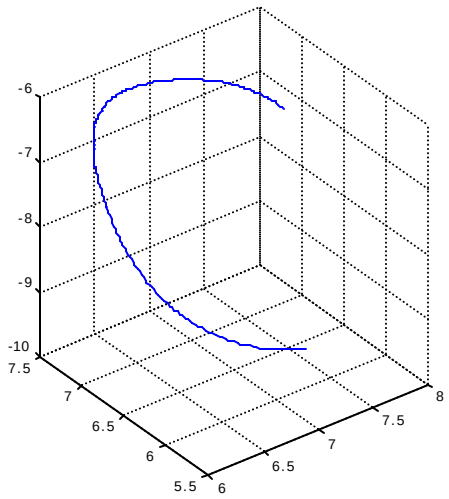
[3] Goldberg, D.E., *Genetic algorithms in search, optimization and machine learning*. Reading, MA: Addison Wesley, 1989.

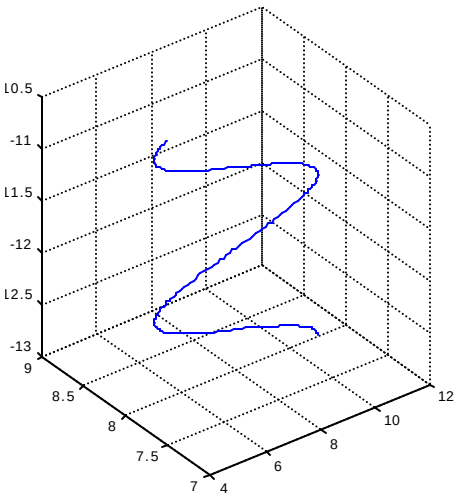
[4] Parker, J.K., Khoogar, A.R., Goldberg, D.E., Inverse Kinematics of Redundant Robots using Genetic Algorithms, *Proc. IEEE International Conference on Robotics and Automation*, pp. 271-276, 1989.

[5] Forrest, S., Genetic Algorithms: Principles of Natural Selection Applied to Computation, *Science*, vol. 261, pp. 872-878, 1993.


-12
-13
-14
-15
-16
-17
1.5
1
0.5
0
7
8
9
10


-6
-8
-10
-12
-14
-16
-18
8
7.5
7
6.5
6
6
6.5
7
7.5
8


-6
-7
-8
-9
-10
7.5
7
6.5
6
5.5
6
6.5
7
7.5
8


10.5
-11
11.5
-12
12.5
-13
9
8.5
8
7.5
7
4
6
8
10
12