INVERSE KINEMATICS OF A BINARY FLEXIBLE MANIPULATOR USING GENETIC ALGORITHMS

Felipe dos Santos Scofano
Department of Mechanical Engineering, Pontifical Catholic University of Rio de Janeiro
R. Marquês de São Vicente 225, Rio de Janeiro, RJ 22453-900, Brazil
e-mail: felipescofano@yahoo.com.br

Marco Antonio Meggiolaro
Department of Mechanical Engineering, Pontifical Catholic University of Rio de Janeiro
R. Marquês de São Vicente 225, Rio de Janeiro, RJ 22453-900, Brazil
e-mail: meggi@mec.puc-rio.br

Vivek Anand Sujan
Air Handling and Combustion Control Division, Cummins Engine Company
Columbus, IN 47201, USA
e-mail: vivek.a.sujan@cummins.com

Abstract. *This work presents an approach to calculate the inverse kinematics of a binary flexible manipulator. Binary manipulators are systems consisting of several actuators, each of them with only two potential states. By combining the binary state of each actuator, desired positions and trajectories can be obtained. A large number of actuators is desired to increase the number of end-effector positions and orientations attainable by the robot, approaching the performance of a continuous system. The main advantages of these redundant systems are ease of control and robustness to actuator failure. The studied manipulator structure is made of a polymer-reinforced elastomer, very flexible and lightweight, consisting of several independently pressurized chambers. Each chamber has only two states, pressurized or not, causing the manipulator structure to deform in a well-known direction. By leaving open or closed the valves of each chamber, the end-effector position and orientation can be controlled. However, to obtain the required inverse kinematics of a system consisting of m actuators, 2^m states would need to be evaluated through an exhaustive search. To avoid this computational burden, genetic algorithms are used in this work to obtain the required combination of valve states required to position the manipulator end-effector. Simulation results demonstrate the efficiency of the algorithm.*

Keywords*: inverse kinematics, pneumatic rubber hand, genetic algorithms*

1. Introduction

Recently, much interest has been devoted to the concept of continuous manipulators with flexible links (Robinson and Davies, 1999). Such systems consist of flexible links that can be continuously deformed in a controllable way, as opposed to traditional rigid link manipulators with rotary and prismatic joints. Flexible links allow an increase in performance when interacting with the environment, providing the required redundancy to avoid obstacles along the path between the manipulator base and end-effector. However, the hardware involved to control such multi degree-of-freedom (DOF) systems can be quite complex and expensive, due to the high number of required sensors and actuators.

Merging this concept with the field of binary robotics can result in a flexible system with many degrees of freedom and high mobility, without the need of a complex and costly hardware setup (Lichter et al., 2002). In a binary manipulator, each actuator can have only two states, either on or off, behaving as a discrete system. As the number of actuators is increased, so does the number of degrees of freedom and therefore its ability to behave as a conventional continuous manipulator. The control system required by such hiper-redundant binary manipulators is much simpler than conventional ones, since no feedback for continuous positioning is required. To change the binary manipulator end-effector, one only needs to switch the state of each actuator accordingly. As a result, the hardware weight and complexity can be significantly reduced.

The kinematics and control of conventional hiper-redundant manipulators has been extensively studied in the literature (Chirikjian and Burdick, 1995; Ebert-Uphoff and Chirikjian, 1996; Huang and Shou-Hung Ling, 1994). However, the kinematics and trajectory generation for binary manipulators is a more challenging problem if compared to conventional manipulators (Ebert-Uphoff and Chirikjian, 1996; Sujan et al., 2001; Sujan and Dubowsky, 2004). For instance, the inverse kinematic problem of a binary robot involves a huge search effort on the manipulator's discrete workspace, until finding the actuator configuration that best approaches the desired end-effector position.

The workspace of a binary manipulator is not a continuous region (Sen and Mruthyunjaya, 1994). Instead, the associated workspace is a finite set of points in space, see Fig. 1. For each point in the workspace, usually there is only one attainable end-effector orientation, represented by the arrows in Fig. 1(b). The density of such points increases with

the number of actuators, since each new actuator doubles the number of possible end-effector configurations. The higher the density, the closer the behavior to the one of a continuous system.

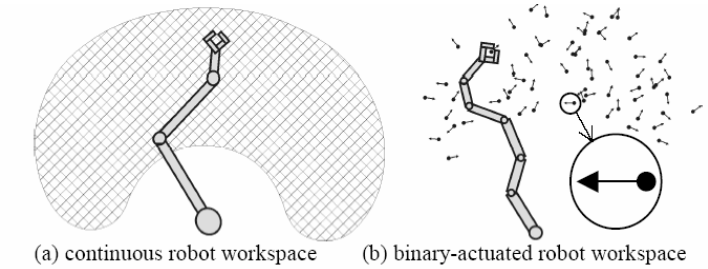

Figure 1. Distinction between continuous and binary manipulator workspaces (Lichter et al. 2002)

In this work, a genetic algorithm is used to obtain the inverse kinematics of a binary flexible manipulator. The manipulator concept involves a structure made of polymer-reinforced elastomeric hoses, which are deformed through pneumatic actuation. Such actuation principle has been described and implemented for a small number of degrees of freedom in (Hafez et al., 2003; Suzumori et al., 1991; Suzumori et al., 1992). The use of genetic algorithms to evaluate the inverse kinematics of manipulators is not new, however this work presents for the first time the application of such technique to a binary flexible robot.

The simulated binary system is a 48 DOF 3-D pneumatic flexible manipulator. Based on the equations of the manipulator direct kinematics, a genetic algorithm (GA) is used to find the best actuator configuration to achieve a desired end-effector position within the workspace. The algorithm performance is compared to the one obtained from random search techniques. This GA enables the calculation of the inverse kinematics and trajectory planning for such hiper-redundant binary manipulator. In the next section, the manipulator and its model are presented.

2. Manipulator model

The proposed pneumatic manipulator consists of n elastomeric links connected in series, actuated pneumatically. Each link has three independently pressurized chambers with pressures P_1, P_2 and P_3, see Fig. 2. The pressure difference among the chambers causes the link to bend in a known direction, otherwise the link remains straight if all pressures are equal (Suzumori et al., 1991; Suzumori et al., 1992).

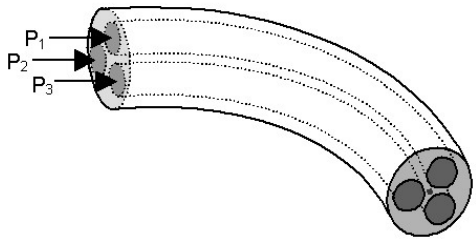

Figure 2. Schematics of one manipulator link with its three independently pressurized chambers

Each link has 3 DOFs, resulting in a $3n$ DOFs system. The differences $(P_1 - P_2)$ and $(P_1 - P_3)$ are enough to define the bending orientation and radius of each link, which result in most of the system mobility. In addition, the mean pressure $(P_1 + P_2 + P_3)/3$ is responsible for the stretching of the entire link in its axial direction. In most cases this stretching is relatively small if compared to the bending effect on the link. Even though this mean pressure effect must be accounted for in direct kinematics, it only provides a weak additional mobility to the end-effector. Therefore, only $2n$ DOFs provide most of the system mobility. The remaining n DOFs, which are associated with the mean pressure in each of the n links, are mostly responsible for changing the system compliance.

Assuming each link with an initial length L_0, the final length L can be calculated from the axial displacement ΔL caused by the mean pressure

$$L = L_0 + \Delta L = L_0 + \frac{L_0 A_p (P_1 + P_2 + P_3)}{3 A_0 E} \tag{1}$$

where A_p is the total cross section area of the 3 pressurized chambers, A_0 is the cross section area of the fiber-reinforced elastomeric structure (without including the chambers area), and E is the elastomer Young's modulus, see Fig. (3). The above equation assumes only axial deformation, without significant radial deformation due to the fiber reinforcement.

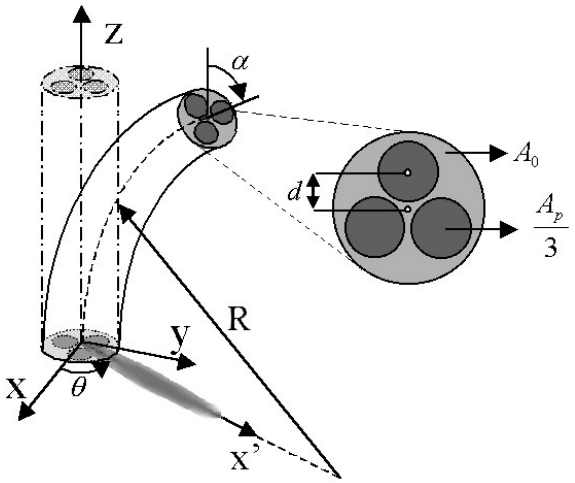

Figure 3. Analytical model of each link

The center of the chamber associated with pressure P_1 is defined at the y axis between the third and fourth quadrants of the xy plane in Fig. 3. The chambers with pressures P_2 and P_3 are defined on the second and first quadrants respectively. Assuming there is a pressure difference among the chambers, the link ideally deforms as a circular arc with angle α and radius R, calculated by

$$R = \frac{3EI}{A_p d\sqrt{P_1^2 + P_2^2 + P_3^2 - P_1P_2 - P_2P_3 - P_1P_3}} \tag{2}$$

$$\alpha = \frac{L}{R} \tag{3}$$

where d is the excentricity of each chamber (Fig. 3) and I is the bending moment of inertia of the link. Note that α is also the angle between the link ends.

The angle θ between the axes x and x', where x' is the direction of the projection of the link onto the xy plane in Fig. 3, can also be obtained from the applied pressures, resulting in:

$$\theta = \arctan\frac{2P_1 - P_2 - P_3}{(P_2 - P_3)\sqrt{3}} \tag{4}$$

2.1. Direct kinematics

From Equations (1-4) it is possible to obtain the values of L_i, R_i, α_i and θ_i of each link i of a manipulator. These values are used to obtain a homogeneous transformation matrix A_i^{i-1} between the coordinate frames of both link ends:

$$A_i^{i-1} = \begin{bmatrix} sin^2\theta_i + cos^2\theta_i \cdot cos\alpha_i & -sin\theta_i \cdot cos\theta_i(1-cos\alpha_i) & cos\theta_i \cdot sin\alpha_i & \frac{L_i}{\alpha_i}cos\theta_i(1-cos\alpha_i) \\ -sin\theta_i \cdot cos\theta_i(1-cos\alpha_i) & cos^2\theta_i + sin^2\theta_i \cdot cos\alpha_i & sin\theta_i \cdot sin\alpha_i & \frac{L_i}{\alpha_i}sin\alpha_i(1-cos\alpha_i) \\ -cos\theta_i \cdot sin\alpha_i & -sin\theta_i \cdot sin\alpha_i & cos\alpha_i & \frac{L_i}{\alpha_i}sin\alpha_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{5}$$

Note that the above matrix is a function of the pressures $P_{1,i}$, $P_{2,i}$ and $P_{3,i}$ applied at respectively at the chambers 1, 2 and 3 of link i. For a manipulator with n links, the matrix A_n^0 which correlates the base and end-effector coordinate frames is obtained by (Craig, 1989):

$$A_n^0 = A_1^0 \cdot A_2^1 \cdot \ldots \cdot A_n^{n-1} \tag{6}$$

When binary actuation is considered, then each chamber is either pressurized with a fixed pressure P or with no relative pressure at all. Each link has then only $2^3 = 8$ possible states, limiting the values of θ_i to the angles 30°, 90°, 150°, 210°, 270° or 330°, and the radius R as either infinity (for a straight link configuration) or to:

$$R = \frac{3EI}{A_p d \cdot P} \tag{7}$$

2.2 Inverse kinematics

Position and trajectory control of the manipulator require the calculation of its inverse kinematics. After defining a desired end-effector position and orientation of an n-link manipulator, the required $3n$ pressures $P_{1,i}$, $P_{2,i}$, $P_{3,i}$ (i = 1, 2, ..., n) must be obtained. But, as expected, there is no analytical solution for the inverse kinematics of the considered manipulator.

When binary actuation is considered, a manipulator with n links will only have a solution for its inverse kinematics for the 2^{3n} points that form its workspace. For any other desired end-effector position, a positioning error will always be present. It is desirable to choose which of the $3n$ chambers will be pressurized in order to minimize the error between the actual and desired end-effector positions. For a system with few links, this could be done using an exhaustive search among the 2^{3n} elements of the joint space.

However, as the number of links is increased, the number of elements of the joint space grows exponentially. Therefore, the exhaustive search approach is impractical for systems with several DOFs. Random search is not a good alternative, since a very large number of samples would be required to guarantee a small end-effector error. Two reasonable options are a combinatorial heuristic search algorithm and a genetic search algorithm (Sujan and Dubowsky, 2004). It has been found that while exhaustive search can be adequate for systems with up to 10 DOFs, combinatorial search is the most efficient method for systems between 10 and 40 DOFs approximately, and above 40 DOFs genetic algorithms are the best choice. In the next section, the implemented genetic algorithm is described.

3. Genetic algorithm

Genetic algorithms are a family of computational models inspired on evolution theory, used to find an optimum solution. These algorithms are basically function optimizers, even though the range of application of such methods is much wider. The GA begins with a set of random manipulator solution states, called a generation. The GA uses simple physically-based rules, or tests, to produce a fitness value for a given manipulator configuration (Sujan and Dubowsky, 2004). The fitness value is used to compare one manipulator state to another. To generate successive generations of solutions, the algorithm allows for reproduction, genetic crossover and genetic mutation. The probability of reproduction is proportional to the fitness value of a solution, thus solutions with better fitness values have a better chance of being reproduced in the next generation. The algorithm then combines some attributes from one solution with those of another, selected from the set of reproduced solutions, a process called crossover. The algorithm may also add new characteristics that were not present in the previous generation, a process called mutation. Using the techniques of crossover and mutation, a new generation of manipulator configurations is evolved. Appropriately designed genetic search algorithms will converge to a locally optimum solution. This algorithm design involves selection of a fitness function, crossover and mutation probabilities, and the population size.

Here a classical genetic algorithm approach for finding an optimum solution is used, where each cromossome consists of a $3n$-bit binary word describing the manipulator state. The first 3 bits represent the state of link 1, where each pressure $P_{1,1}$, $P_{2,1}$ and $P_{3,1}$ either receive a null value (bit 0) or a value P (bit 1). In general, bits at the $3i-2$, $3i-1$ and $3i$ position are associated with the state of pressures $P_{1,i}$, $P_{2,i}$ and $P_{3,i}$ of link i. These words are then used to find the transformation matrices of each link and ultimately the end-effector position (x, y, z):

$$\begin{pmatrix} x \\ y \\ z \end{pmatrix} = f(\underbrace{110}_{A_1^0}\underbrace{001}_{A_2^1}\underbrace{101}_{A_3^2}\underbrace{001}_{A_4^3}\ldots\underbrace{000}_{A_{n-1}^{n-2}}\underbrace{111}_{A_n^{n-1}}) \tag{8}$$

The fitness is simply defined as a function of the error between the current end-effector position (x_r, y_r, z_r) and the desired position (x_d, y_d, z_d), calculated using the Euclidean distance. In order to make the fitness increase as the positioning error decreases, the following fitness function is adopted:

$$fitness = \frac{1}{\sqrt{(x_d - x_r)^2 + (y_d - y_r)^2 + (z_d - z_r)^2} + 1} \tag{9}$$

The above expression results in fitness values between 0 and 1. The considered operators in the implemented genetic algorithm are single-point crossover and mutation. A standard genetic algorithm is adopted, with a genitor selection technique based on roulette, all implemented in Matlab using the genetic algorithm toolbox and the functions specific to binary numbers. The cromossome from the previous generation with best fitness value, represented as a binary variable in the implemented code, is always included in the next generation, warranting that the next generation will have a fitness value at least equal or better than the previous one. In the next section, the proposed GA is evaluated.

4. Simulation results

The proposed algorithm is applied to a 48-DOF system, consisting of a flexible pneumatic manipulator with 16 links. The considered manipulator parameters are shown in Table 1. These numbers are associated with the properties of a 3-DOF link prototype made of silicon rubber.

Table 1. Simulated manipulator parameters.

L_0 (mm)	d (mm)	E (MPa)	I (mm^4)	P (Mpa)	A_0 (mm^2)	A_p (mm^2)
40	10	0.5	1.1781x10^5	0.2	917.3	339.3

The eight possible configurations of each link are shown schematically in Figure 4, where e.g. the configuration associated with the 3-bit word 101 has chambers 1 and 3 with pressure P (marked in black) while chamber 2 is depressurized (marked in white).

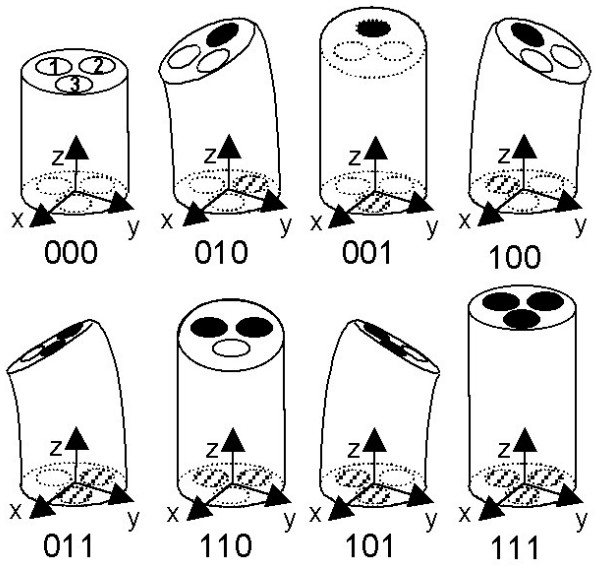

Figure 4. Eight possible configurations of a manipulator link, with the associated 3-bit words

Using the values defined in Table 1, the homogeneous transformation matrices associated with each 3-bit word can be computed:

$$A_{000} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 40 \\ 0 & 0 & 0 & 1 \end{bmatrix} \qquad A_{111} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 45.9178 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$A_{100}=\begin{bmatrix}1 & 0 & 0 & 0\\ 0 & 0.9870 & 0.1605 & 3.3751\\ 0 & -0.1605 & 0.9870 & 41.7911\\ 0 & 0 & 0 & 1\end{bmatrix} \qquad A_{110}=\begin{bmatrix}0.9893 & -0.0062 & 0.1454 & 3.2035\\ -0.0062 & 0.9964 & 0.0840 & 1.8495\\ -0.1454 & -0.0840 & 0.9858 & 43.7369\\ 0 & 0 & 0 & 1\end{bmatrix}$$

$$A_{010}=\begin{bmatrix}0.9903 & 0.0056 & 0.1390 & 2.9230\\ 0.0056 & 0.9968 & -0.0802 & -1.6876\\ -0.1390 & 0.0802 & 0.9870 & 41.7911\\ 0 & 0 & 0 & 1\end{bmatrix} \qquad A_{101}=\begin{bmatrix}0.9893 & 0.0062 & -0.1454 & -3.2035\\ 0.0062 & 0.9964 & 0.0840 & 1.8495\\ 0.1454 & -0.0840 & 0.9858 & 43.7369\\ 0 & 0 & 0 & 1\end{bmatrix} \tag{10}$$

$$A_{001}=\begin{bmatrix}0.9903 & -0.0056 & -0.1390 & -2.9230\\ -0.0056 & 0.9968 & -0.0802 & -1.6876\\ 0.1390 & 0.0802 & 0.9870 & 41.7911\\ 0 & 0 & 0 & 1\end{bmatrix} \qquad A_{011}=\begin{bmatrix}1 & 0 & 0 & 0\\ 0 & 0.9858 & -0.1679 & -3.6991\\ 0 & 0.1679 & 0.9858 & 43.7369\\ 0 & 0 & 0 & 1\end{bmatrix}$$

These matrices can then be used to compute the end-effector position given a 48-bit word in the joint space. Figure 5 shows the discrete workspace of the considered manipulator.

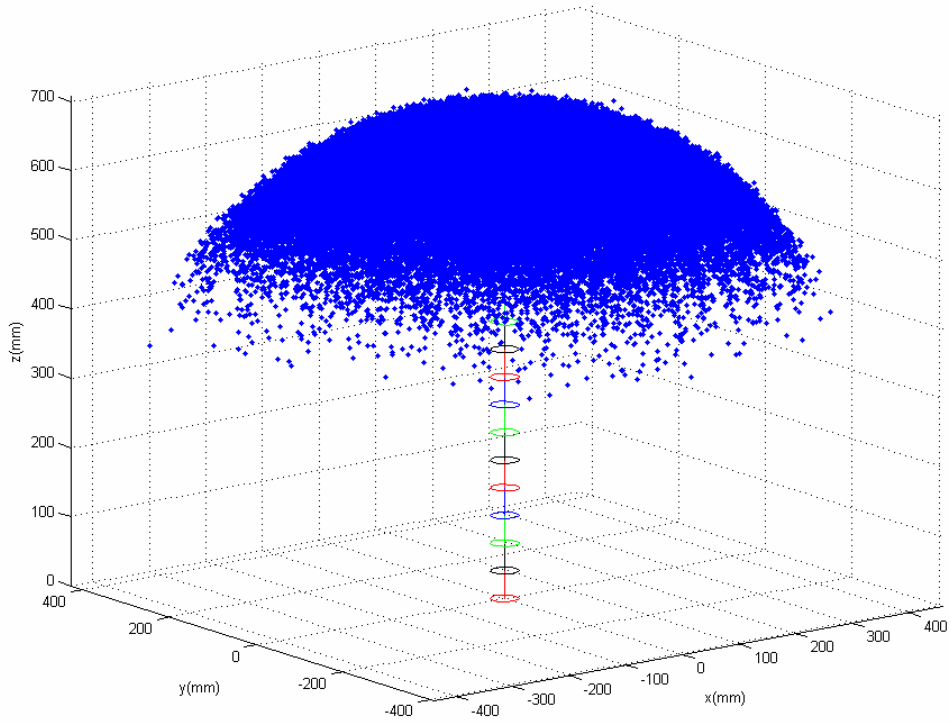

Figure 5. Discrete workspace of the considered 16-link manipulator

The genetic algorithm is implemented using the parameters shown in Table 2 in the software Matlab using the genetic algorithm toolbox. The desired end-effector positions are chosen within the region of the manipulator workspace shown in Fig. 5, but not necessarily at exactly one of the points in the discrete workspace. Therefore, the globally optimal solution in general will not be associated with a zero error.

Table 2. Genetic algorithm parameters

Crossover probability (%)	90
Mutation probability (%)	8
Number of generations	1000
Initial population size	30

As an example, the desired end-effector position (x_d, y_d, z_d) = (300, 300, 350)mm is chosen to demonstrate the behavior of the genetic algorithm. This desired position is within the region of the manipulator workspace, however

there is no inverse kinematic solution leading to this exact position. The first generation consists of 30 random cromossomes, where the one with best fitness is shown as the initial guess in Figure 6. The cromossomes with best fitness value for every 10 generations are also shown in the figure. For this example, the algorithm converged to a locally optimal solution (x_r, y_r, z_r) = (300.5, 300.3, 349.2)mm, resulting in a residual error of 1.02mm (about 0.16% of the manipulator characteristic length).

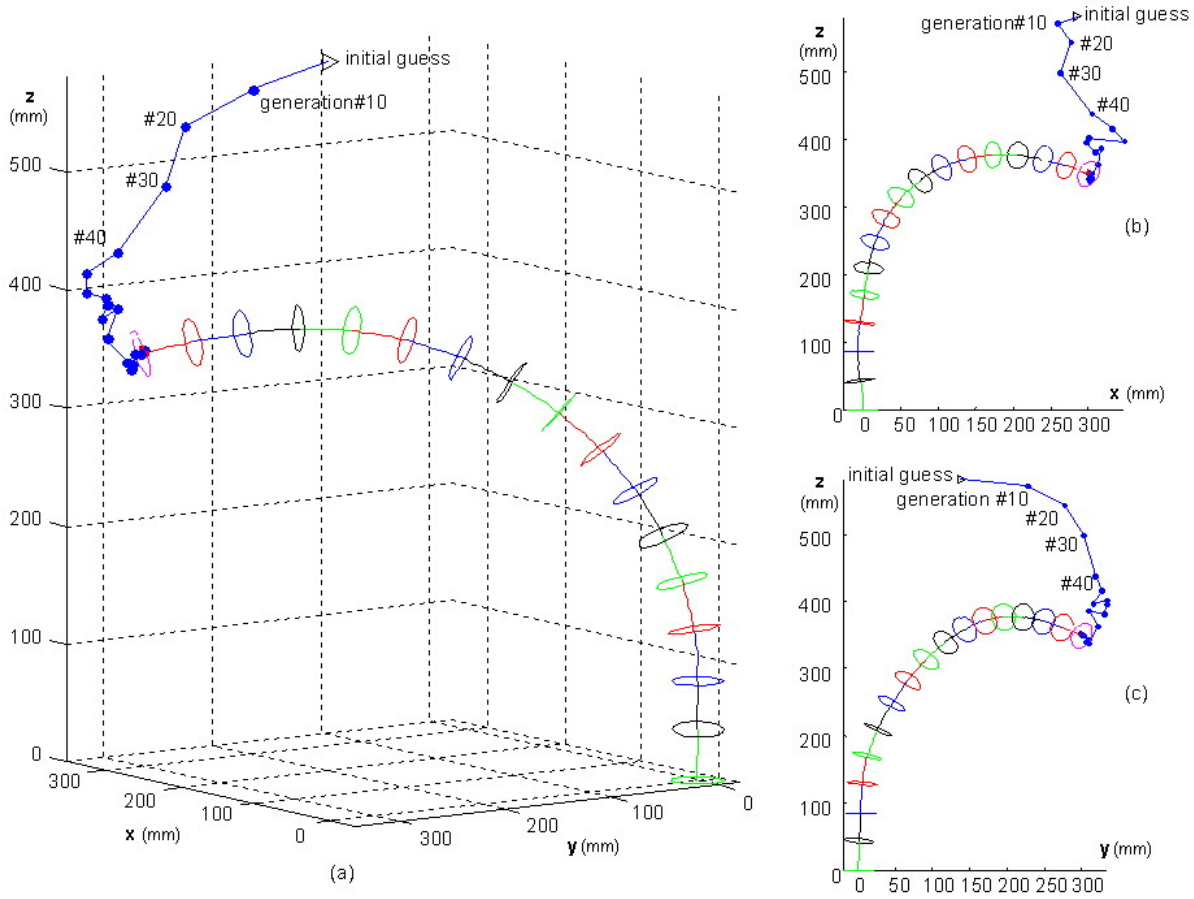

Figure 6. Evolution of the genetic algorithm to achieve a desired end-effector position

The above example only reflects the results for a specific desired position. To evaluate the overall performance of the algorithm, 500 different desired end-effector positions are considered, chosen randomically within the region of the manipulator workspace. The root mean square (RMS) error for the 500 desired positions is evaluated at each of the 1000 generations used in the genetic algorithm, as shown in Fig. 7. The results are compared with random search techniques, implemented by simply considering a genetic algorithm with a 100% mutation rate. In this way both approaches can be directly compared, see Fig. 7.

Note that the end-effector error rapidly decreases during the first few generations for both techniques. However, for random search it is found that a plateau associated with a 65mm RMS end-effector error is reached, which could not be overcome within the considered 1000 generations. On the other hand, the error of the genetic algorithm continued to decrease until reaching a 1,65mm RMS value for the 500 simulations considering 1000 generations. Further reduction is expected in average for a larger number of generations, however the computational cost would significantly increase. The average genetic algorithm calculation time for 1000 generations, using a Pentium V with 1.5GHz, is 32 seconds. Note however from Fig. 7 that if e.g. 4mm end-effector errors are tolerable in this example, then this calculation time can be lowered to 3.2 seconds as the number of generations is reduced to 100.

5. Conclusions

In this work, a pneumatic binary flexible manipulator was modeled. The manipulator consisted of *n* links with 3 chambers each that could be either pressurized or not. By combining the binary state of each chamber, desired end-effector positions and trajectories can be obtained. The inverse kinematics of this discrete system was evaluated using genetic algorithms. Simulation results demonstrated the efficiency of the algorithm when compared to random search methods, reducing the end-effector errors by a factor of 40 over 1000 generations.

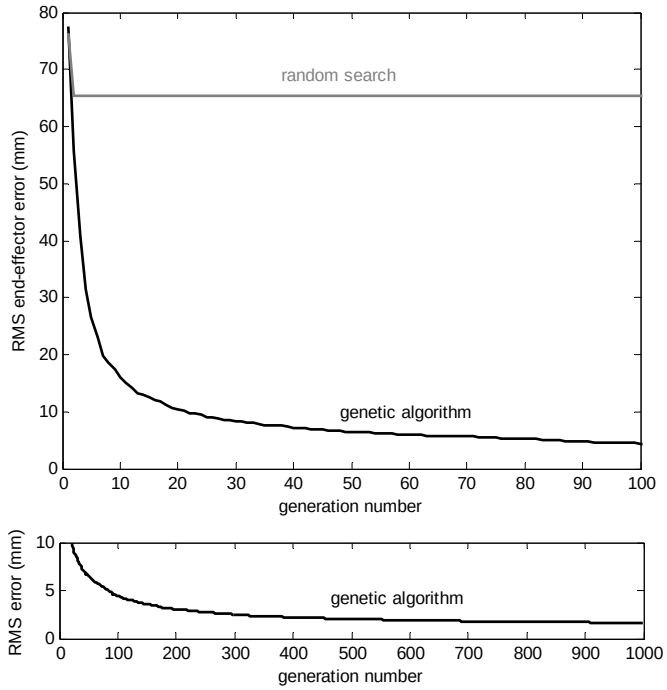

Figure 7. Evolution of the RMS end-effector error obtained at each of the 1000 generations

6. References

Chirikjian, G.S., Burdick, J.W, 1995, "*The kinematics of hyper-redundant robot locomotion*". IEEE Transactions on Robotics and Automation. Vol.11, N.6, pp.781-793.

Craig, J J., 1989, "*Introduction to Robotics: Mechanics and Control*". Second ed, Addison-Wesley, Reading, MA.

Ebert-Uphoff, I.; Chirikjian, G.S, 1996, "*Inverse kinematics of discretely actuated hyper-redundant manipulators using workspace densities*". Proceedings of the IEEE International Conference on Robotics and Automation, Vol.1, pp.139-145.

Hafez, M., Lichter, M.D., Dubowsky, S., 2003, "*Optimized Binary Modular Reconfigurable Robotic Devices*". Proceedings of the IEEE/ASME Transactions on Mechatronics, Vol. 8, No. 1.

Huang, M.Z., Shou-Hung Ling, 1994, "*Kinematics of a class of hybrid robotic mechanisms with parallel and series modules*". Proceedings of the IEEE International Conference on Robotics and Automation, Vol.3, pp.2180-2185.

Lichter, M.D., Sujan, V.A., Dubowsky, S., 2002, "Computational Issues in the Planning and Kinematics of Binary Robots." Proc. of the 2002 IEEE Int. Conference on Robotics and Automation (ICRA 02), Washington, DC.

Robinson, G., Davies, J.B.C., 1999, "*Continuum Robots – A State of the Art*". In: Proceedings 1999 IEEE International Conference on Robotics and Automation, pp. 2849-2854.

Sen, D., Mruthyunjaya, T.S., 1994, "*A Discrete State Perspective of Manipulator Workspaces*". Mech. Mach. Theory, Vol. 29, No.4, pp.591-605.

Sujan, V.A., Lichter, M.D., Dubowsky, S., 2001, "*Lightweight Hyper-redundant Binary Elements for Planetary Exploration Robots*". Proceedings of the IEEE/ASME International Conference on Advanced Intelligent Mechatronics (AIM '01), Como, Italy.

Sujan, V.A., Dubowsky, S., 2004, "Design of a Lightweight Hyper-Redundant Deployable Binary Manipulator". ASME Journal of Mechanical Design, Vol. 126, pp. 29-39.

Suzumori, K., Iikura, S., Tanaka, H., 1991, "*Flexible microactuator for miniature robots*," Proc. IEEE Micro Electro Mechanical Systems Conf. Nara, Japan, pp.204-209.

Suzumori, K., Iikura, S., Tanaka, H., 1992, "Applying *a flexible microactuator to robotic mechanisms*," IEEE Trans. Contr. Syst. Technol., Vol. 12, pp. 21-27.

7. Responsibility notice

The authors are the only responsible for the printed material included in this paper.