Proceedings of the 1996 IEEE
International Conference on Robotics and Automation
Minneapolis, Minnesota - April 1996

Inverse Kinematics of Discretely Actuated Hyper-Redundant Manipulators Using Workspace Densities

Imme Ebert-Uphoff * Gregory S. Chirikjian †

Department of Mechanical Engineering
Johns Hopkins University
Baltimore, MD 21218

Abstract

Hyper-redundant manipulators present an alternative to conventional 6 DOF manipulators for inspection, space, and medical applications. The additional degrees of freedom facilitate obstacle avoidance and allow tasks to be performed even if some of the actuators fail. In this paper we consider hyper-redundant manipulators that are actuated discretely, e.g. using two-state actuators or motors with finite resolution. The inverse kinematics problem for a discretely actuated manipulator is intrinsically different from the one for its continuously actuated counterpart. We present a framework for the discussion of the discretely actuated case and propose an algorithm for the inverse kinematics. The algorithm generates solutions in linear time with respect to the number of manipulator actuators, as opposed to the exponential time required by brute force search.

1 Introduction

Hyper-redundant architectures include cascades of modules such as Stewart Platforms, variable geometry trusses, or serial links. We call any of these types *macroscopically-serial* manipulators. One particular subclass of hyper-redundant manipulators are mechanisms consisting of serial links connected by active revolute joints. They have been studied extensively, e.g. in the early 70's a prototype called the Active Cord Mechanism was built in Japan by Shigeo Hirose, who based the design and motion planning procedures on studies on snakes [1]. More recent developments include a prototype of a Hyper-redundant Active Endoscope for use in remote and minimally invasive surgery [2], and a prototype with 12 degrees of freedom that is currently being tested at the Jet Propulsion Laboratory for inspection tasks on space stations [3]. A wide variety of algorithmical issues have been discussed for this kind of manipulator, see for example [4, 5] and references therein.

A second class of hyper-redundant manipulators uses variable geometry truss mechanisms (VGT's) stacked on top of one another. This design originated from space applications. Actuators were added to large static space structures so that they could be folded during transport and actively deployed in space. An advantage of this type of manipulator is that it offers excellent static stability. Different candidate topologies and their suitability for applications in space have been discussed extensively, e.g. [6, 7, 8].

One approach to the inverse kinematics of general macroscopically-serial manipulators is to prescribe their shape using *backbone curves* ([9, 10] and references therein).

In this paper we consider manipulators of general macroscopically-serial architecture that are actuated discretely. We consider discrete actuators because they offer high precision and reliability with low cost compared to actuators with continuous range-of-motion. For example, a continuous actuator can be turned into a very precise two-state actuator by replacing the position feedback by a mechanical constraint and overpowering it, to make sure that both states are reached under load. This is also an alternative for applications where position sensors and feedback are not avaiable or are too expensive.

The inverse kinematics problem for a discretely actuated manipulator is intrinsically different from the one for its continuously actuated counterpart. Assuming continuous motion capabilities to determine ideal joint values and rounding these to the closest discrete joint values available can result in large errors. The reason for this is that the discrete joint values closest to the ideal joint values do not necessarily lead to the discrete configuration closest to the ideal configuration (since the forward kinematics are in general non-linear). In addition small inaccuracies close to the base can result in large displacements at the tip.

We present a framework for the discussion that considers only the discrete configurations attainable by the manipulator and develop an algorithm that solves the inverse kinematics problem in linear time with respect to the number of manipulator modules. The algorithm is based on determining and fixing a configuration for each of the modules sequentially starting

*Graduate Student
†Assistant Professor, Presidential Faculty Fellow

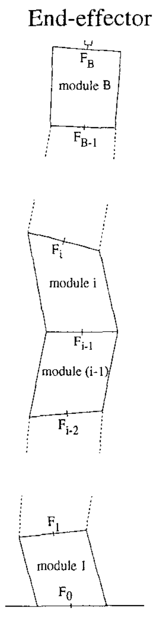

Figure 1: A macroscopically-serial manipulator

at the base while maximizing at every step the redundancy remaining to reach the goal. A measure for the remaining redundancy is derived from characteristics of the workspace of the more distal platforms. The approach is illustrated for binary manipulators (using two-state actuators), as an example of a hyper-redundant manipulator with extremely low resolution in each actuator.

2 Concepts for Discrete Manipulators

This section provides concepts and definitions for manipulators whose joints have only discrete states. The discussion is restricted to manipulators of the *macroscopically-serial* type, i.e. a serial collection of modules where each module is mounted on top of the previous one. Modules are numbered 1 to B, from the base to the end of the manipulator, as shown in Figure 1. Closed loops may exist in each module, but macroscopic loops are not permitted.

2.1 Point Density and Intermediate Workspaces

The workspace of a continuous range-of-motion manipulator is often described by its boundary and the ease with which points in the interior can be reached is described using dexterity measures. A discrete manipulator can only reach a finite number of points. Therefore not only the boundary of the workspace is important but also the distribution of the points inside this boundary.

For discrete manipulators one has to distinguish carefully between *redundant architecture* and *redundancy of motion*. In the context of discretely actuated manipulators redundancy of motion only has meaning when an error threshold is specified. That is, given an error radius r, a discrete manipulator is redundant for a specified position and orientation (frame), if there is more than one configuration within the distance r of the frame. The distance is measured according to a distance function that compares two frames with each other, e.g. such as in [11]. In the following we define the redundancy of motion based on the local density of reachable points/frames per unit workspace volume.

Depending on whether the manipulator under consideration is planar or spatial, and whether orientation is considered, the manipulator workspace W is a subset of $\mathbb{R}^2, \mathbb{R}^2 \times \mathbf{SO(2)}, \mathbb{R}^3$ or $\mathbb{R}^3 \times \mathbf{SO(3)}$. From now on we assume that the workspace W is divided into blocks (pixels) of equal size. The distribution of the points is described as follows: The *point density* ρ assigns each block of workspace W the number of points within the block that are reachable by the discrete manipulator, normalized by the volume of the block:

$$\rho(\text{block}) = \frac{\#\text{ reachable points in block}}{\text{unit volume/area}}$$

The point density is a probabilistic measure of the positional accuracy of the end-effector in a considered area of the workspace. The higher the density in the neighborhood of a point, the more accurately we expect to be able to reach the point.

The i^{th} *intermediate workspace*, W_i^I, of a macroscopically-serial manipulator composed of B modules is the workspace of the manipulator segment from module $i+1$ to the end-effector. End-effector position vectors in the i^{th} intermediate workspace are written in a coordinate frame attached to the top of module i. To visualize this, imagine that the manipulator arm is cut between module i and module $i+1$. The two resulting segments are considered as manipulators on their own: the lower segment (modules 1 to i), and the upper segment (modules $i+1$ to B) with its base in the separating plane. The workspace generated by the upper manipulator segment is the ith intermediate workspace of the whole structure.

Each module has a frame attached to its top. The frames are numbered such that frame F_i is on top of module i, and frame F_0 is the frame at the manipulator base. The number of discrete configurations of module i is denoted K_i. The K_i discrete configurations of module i, $(i = 1, \ldots, B)$ are represented by the *configuration set*:

$$C_i = \{(\mathbf{R}_1^{(i)}, \mathbf{b}_1^{(i)}), (\mathbf{R}_2^{(i)}, \mathbf{b}_2^{(i)}), \ldots, (\mathbf{R}_{K_i}^{(i)}, \mathbf{b}_{K_i}^{(i)})\},$$

where $\mathbf{R}_j^{(i)} \in \mathbf{SO(N)}$ are rotation matrices and $\mathbf{b}_j^{(i)} \in \mathbb{R}^N$ are translation vectors for $j = 1, ..., K_i$. These pairs describe all possible relative orientations and positions of frame i with respect to frame $i-1$.

2.2 The Configuration tree

In this subsection we present a computational structure for manipulators with discrete joint resolu-

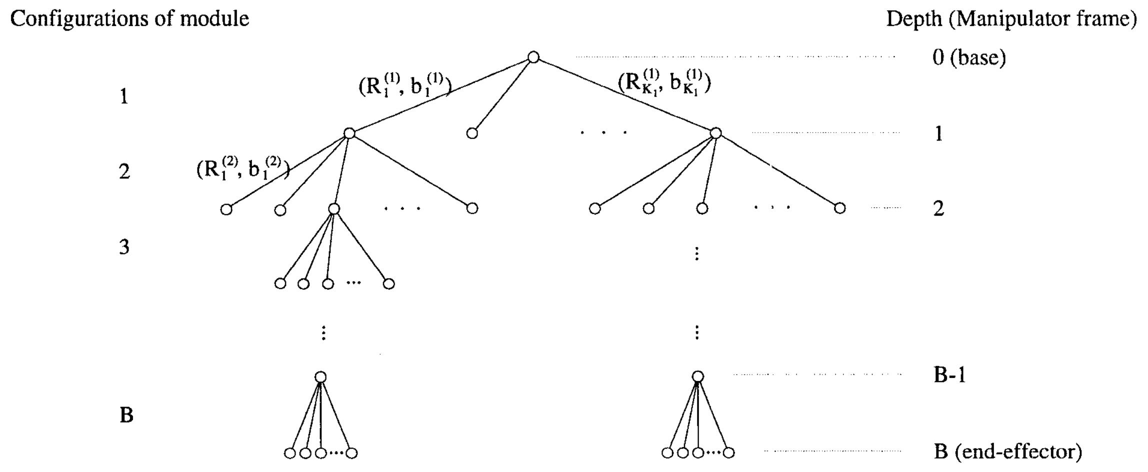

Figure 2: Configuration tree

tion: the *configuration tree*. In later sections we will explain how it can be used to solve the inverse kinematics for manipulators with low joint resolution, in particular for binary manipulators.

As described in the preceding subsection we consider macroscopically-serial manipulators. The modules are numbered, starting at the base with module 1, and proceeding up to the most distal module with the number B. The *configuration tree* (CT) of such a manipulator is a tree defined by the following conventions:

- Starting from the root of the tree with module 1, each level represents one consecutive module of the manipulator.
- At level i each node has exactly one child for each configuration attainable by module i. Each of these K_i branches has a geometric description of the module configuration assigned to it: $(\mathbf{R}_j^{(i)}, \mathbf{b}_j^{(i)}) \in C_i$.

As a result the nodes of the tree hold the following information:

- The nodes of level i enumerate all possible configurations of the partial manipulator given by module 1 to i.
- The leaves of the tree, i.e. the nodes of the last level, enumerate all configurations the whole manipulator can attain.
- For any node on level i the relative orientation and position of frame F_i with respect to the fixed base frame F_0 can be determined in the following way: we walk from the root of the tree to the desired node, consecutively applying all geometric relationships assigned to traversed links.

The inverse kinematics can now be implemented as a search in the tree. For the decision process the densities of intermediate workspaces are used. Note that the configuration tree is **never generated completely**, since the computational complexity is exponential in the number of modules ($\mathcal{O}(\prod_{i=1}^{B} K_i)$). Instead paths in the tree are only generated as necessary.

3 Inverse Kinematics

In this section we explain our approach for the inverse kinematics problem, avoiding the exponential complexity required by brute force computation. Given a target frame, we try to reach it with the end-effector of the manipulator.

The discussion consists of two parts. In the first part a measure is developed that determines how well the goal can be reached if modules close to the base are fixed in a particular configuration. The second part presents algorithms that use this measure to solve the inverse kinematics problem.

3.1 A measure for reachability

3.1.1 Generation of intermediate workspaces

As a prerequisite for the measure of reachability we need a description of intermediate workspaces that includes their internal point density.

Although every discrete manipulator has only a finite number of states, it is impossible to store (or calculate) the end-effector positions for all states of a discrete hyper-redundant manipulator due to exponential complexity ($\mathcal{O}(\prod_{i=1}^{B} K_i)$). On the other hand, if the workspace is discretized in a finite number of pixels of equal size, it is possible to quickly approximate and store the *number* of points contained in each of these pixels, which in turn can be converted to the point density.

The algorithm used in this paper to approximate the point densities of all intermediate workspaces is called the *Mapping Algorithm* and is described in [12] in much detail. Also included in that paper are numerical examples and an error bound on the approximation.[1] The following gives a brief overview of the algorithm.

The Mapping Algorithm determines intermediate workspaces starting at the end-effector and ending at the base. At each step we climb down one module, maintaining an approximation of the intermediate workspace corresponding to the segment of all modules above the current one. The point density of each intermediate workspace is described by an N-dimensional array of integers representing the number of points in each pixel of the discretized workspace, where N is the dimension of the workspace. Combined with administrative information, i.e. the pixel dimensions and the coordinates of a reference point of the workspace, this array represents the density distribution in the workspace.

Each iteration of the algorithm is based on the observation that workspace W_{i-1} (workspace of modules i to B) is connected to workspace W_i (workspace of modules $i+1$ to B) through the set C_i of all possible configurations of module i. To generate the point density of workspace W^I_{i-1} the algorithm interpretes the elements of C_{B-i} as homogeneous transformations and applies them to workspace W^I_i. Superposition of the results yields workspace W^I_{i-1}. Note, that the complexity of each homogeneous transformation depends only on the number of pixels chosen for the discretization of workspace W^I_i and can be implemented very efficiently as an identical operation on all indices of the array. The generation of all intermediate workspaces is performed offline and requires time linear in the number of modules of the manipulator ($O(B)$), linear memory, and quadratic but small error.

In more general terms this transformation can be reformulated as a convolution of functions on the Euclidean group, $SE(N)$. Two real-valued functions, $f_1(H)$ and $f_2(H)$, where $H \in SE(N)$, are used to describe the distribution of configurations C_{B-i} and the point density of intermediate workspace W^I_i, respectively. The convolution of the two functions [2] results in the density distribution $f_3(H)$ of the intermediate workspace W^I_i:

$$f_3(H) = (f_1 * f_2)(H) = \int_{SE(N)} f_1(\mathcal{H}) f_2(\mathcal{H}^{-1}H)\,d\mu(\mathcal{H}),$$

where $d\mu(\mathcal{H})$ is a volume element in $SE(N)$. An algorithm for workspace generation based on this equation as well as mathematical and physical interpretation of above equation are described in [13]. For identical modules the workspace generation is achieved in order $O(\log B)$ time.

[1] Although the algorithm is described in [12] only for the case of binary actuation, it can be generalized to other discrete actuation by a simple change, namely the numbering convention for the configurations of the modules has to be adjusted.

[2] f_1 and f_2 have to be nonnegative and bounded everywhere, and decrease rapidly enough to zero such that the convolution integral converges.

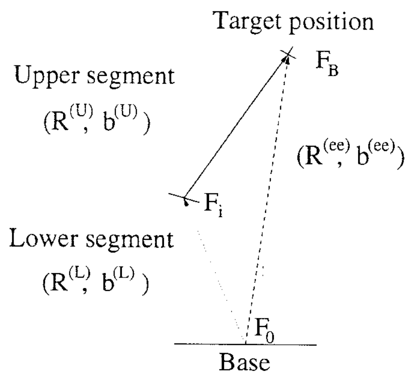

Figure 3: Manipulator divided into two segments

3.1.2 Derivation of Reachability Measure

The goal of this subsection is to develop a measure that predicts how well a goal specified by position vector $\mathbf{b}^{(ee)}$ and rotation matrix $\mathbf{R}^{(ee)}$ can be reached by the end-effector if modules close to the base are fixed in a particular configuration.

The manipulator is divided into a lower segment (modules 1 to i), and an upper segment (modules $i+1$ to B), as described in section 2.1 and shown in Figure 3. To describe their configurations we use the following notation:

- $(\mathbf{R}^{(L)}, \mathbf{b}^{(L)})$: orientation and position of frame F_i with respect to frame F_0 (lower segment)
- $(\mathbf{R}^{(U)}, \mathbf{b}^{(U)})$: orientation and position of frame F_B with respect to frame F_i (upper segment)
- $(\mathbf{R}^{(ee)}, \mathbf{b}^{(ee)})$: target orientation and position of frame F_B with respect to frame F_0 (whole manipulator)

Composing the frame transformations $(\mathbf{R}^{(L)}, \mathbf{b}^{(L)})$ and $(\mathbf{R}^{(U)}, \mathbf{b}^{(U)})$, one finds that

$$(\mathbf{R}^{(ee)}, \mathbf{b}^{(ee)}) = (\mathbf{R}^{(L)}\mathbf{R}^{(U)}, \quad \mathbf{b}^{(L)} + \mathbf{R}^{(L)}\mathbf{b}^{(U)}).$$

Consequently, the ideal configuration the upper module has to attain for any particular configuration of the lower segment is described by

$$\begin{aligned} \mathbf{R}^{(U)} &= \left(\mathbf{R}^{(L)}\right)^T \mathbf{R}^{(ee)}, \\ \mathbf{b}^{(U)} &= \left(\mathbf{R}^{(L)}\right)^T \left(\mathbf{b}^{(ee)} - \mathbf{b}^{(L)}\right). \end{aligned}$$

Recall, that the ith intermediate workspace of the manipulator is the workspace of the upper segment when considered as a manipulator on its own. The point density of this workspace is available through the methods mentioned in 3.1.1. We evaluate the point density at the pixel of the ideal orientation and position of the upper segment, $(\mathbf{R}^{(U)}, \mathbf{b}^{(U)})$. The higher the point density is around $(\mathbf{R}^{(U)}, \mathbf{b}^{(U)})$, the more accurately we expect to reach $(\mathbf{R}^{(ee)}, \mathbf{b}^{(ee)})$ with the end-effector.

Evaluation of reachability measure: In summary the measure on the configuration $(\mathbf{R}^{(L)}, \mathbf{b}^{(L)})$ of the lower module is implemented as:

1. Calculate the homogeneous transform $(\mathbf{R}^{(U)}, \mathbf{b}^{(U)})$ for the upper segment to reach the target.
2. Evaluate the density of the pixel of the ith intermediate workspace in which $(\mathbf{R}^{(U)}, \mathbf{b}^{(U)})$ lies to determine how well it can be reached.

Note, that the evaluation of the reachability measure in the examples for this paper is only based on the position $\mathbf{b}^{(U)}$ and neglects the information of the orientation $\mathbf{R}^{(U)}$. Consequently, the algorithms for the inverse kinematics that are derived for the general case (including orientation) in this section, are only simulated in positional coordinates.

3.2 Algorithms for Inverse Kinematics

Our discussion of the inverse kinematics problem consists of two parts. The first part shows how to resolve the geometric constraints arising from the task of reaching a target location and orientation as closely as possible. An algorithm that finds a solution to these constraints is presented in the first subsection. The second part of this subsection discusses strategies to resolve the remaining redundancy to avoid abrupt transitions between configurations.

3.2.1 Satisfying the geometric constraints

The criterion for reachability established in the previous subsection can be used in a simple algorithm that determines **a** solution for the given target position and orientation. The algorithm starts with the first module and considers the finite set of its possible configurations. For each configuration the reachability measure is evaluated as described in 3.1 and the algorithm picks the one with the maximal value. The module is fixed in this configuration and the algorithm continues with the next module. The following describes the algorithm in terms of the configuration tree:

Path of Probability Algorithm (POP)

1. Input: target location and orientation.
   Start with the root of the tree.
2. For all children of the current node: evaluate the reachability criterion. Choose the branch that maximizes the reachability criterion. Take the child as current node.
3. If the current node does not have any children:
   Output configuration. DONE.
   Otherwise: goto 2.

The reachability criterion can be evaluated in constant time. Considering the maximal number of configurations per module as a constant, the total order of this algorithm is of the number of modules, $\mathcal{O}(B)$, and is therefore also linear in the number of actuators.

3.2.2 Resolving the remaining redundancy

The algorithm of the previous section (POP) provides **a** solution close to the target position and orientation and hence solves the inverse kinematics problem. In practice, however, we are not only interested in finding the solution to one frame but to a set of frames or a trajectory, expecting some kind of smooth and repeatable motion. This is not guaranteed by the preceding algorithm. The problem is that since the redundancy in the interior of the workspace is not resolved, the algorithm arbitrarily picks one out of a number of possible solutions.

Our approach to resolve the redundancy is to maximize a secondary criterion in the algorithm. The secondary criterion prescribes characterisitics of the motion in some way while it resolves the remaining redundancy. Examples of secondary criteria are:

- The weighted number of modules that remain in the same state as in the previous configuration, with a higher weight close to the base.
- The weighted number of modules that are fully contracted (i.e. binary state 0), with a higher weight close to the base.

While the first criterion above compares two configurations with each other, (the previous one with the one under consideration), the second is independent of previous motion (history-independent). The second criterion above generates behavior similar to a long narrow spring: the manipulator expands only as far as necessary. In general history-independent criteria are preferred because they guarantee higher repeatability.

Improved Path of Probability (IPOP)

1. Input: target location and orientation.
   Start with the root of the tree.
2. For all children of the current node: evaluate the reachability criterion and one of the secondary criteria. Choose one branch based on a combination of the two criteria.
3. If the current node does not have any children:
   Output configuration. DONE.
   Otherwise: goto 2.

Our implementation of step 2 considers all configurations for which the primary criterion is above a certain threshhold (to bound the error). Out of these we choose the one that satisfies the secondary criterion the best.

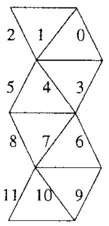

Figure 4: Numbering convention

Result: The Improved Path of Probability algorithm produces solutions that satisfy the geometric constraints imposed by the task of approaching a certain position and orientation. Close to the boundary of the workspace this already specifies the solution. In areas with higher point densities, and therefore higher redundancy, the solution satisfies an additional performance criterion which is chosen according to the application. The evaluation of any of these criteria requires only a constant amount of time. Hence the time requirement of the IPOP algorithm is also linear in the number of modules, $\mathcal{O}(B)$.

4 Simulations of Binary Manipulator Inverse Kinematics

The inverse kinematics algorithms are applied to a planar binary manipulator that consists of $B = 16$ variable geometry truss modules. Each module has three independent actuators which can only attain the lengths 0.2 and 0.25, while the base is fixed at width 0.2. This manipulator can reach $2^{3 \cdot 16} = 2^{48} \approx 2.8 \times 10^{14}$ points in the workspace. The state of the manipulator can be described as a bit sequence, using one bit for each actuator. State 0 indicates that the actuator is contracted, state 1 indicates that it is extended. The actuators are numbered from right to left, from the most distal module to the module at the base, as shown in Figure 4 for a manipulator consisting of only four modules.

The simulations are performed on a Pentium PC (60 MHz) using NextStep. The time required to calculate the inverse kinematics for a single position ranges from 50 to 120 msec, depending on the location in the workspace.

Figures 5(a-c) show results of the inverse kinematics using three different algorithms. The task is to reach three points with the end-effector, $(3.0, 0.7), (0.55, 2.9), (0.4, 2.7)$, in this order, starting from an upright contracted position (bit sequence 0). In the background of each figure the point density of the manipulator workspace is shown in a logarithmic gray scale. Point densities above a certain threshold are shown in identical, dark gray. The three target points are marked by black crosses. The bit sequences corresponding to all generated configurations are listed in Figure 6, together with resulting end-effector position, orientation and error. Each bit sequence is written in the octal system, i.e. each digit

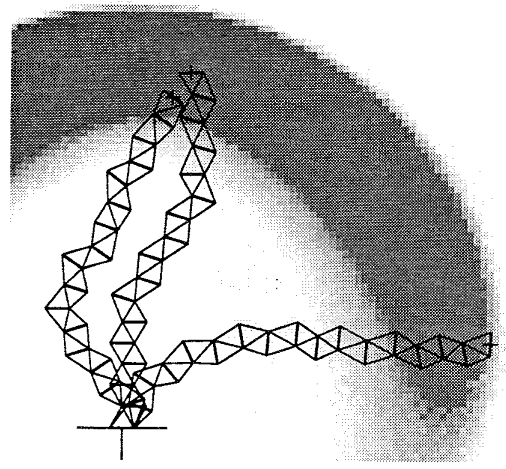
Figure 5(a): POP algorithm

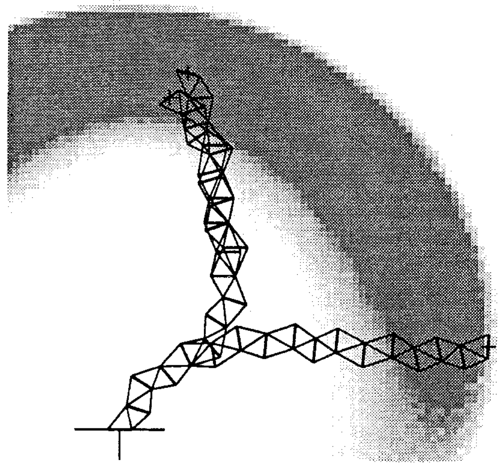
Figure 5(b): IPOP, configurations similar to previous

corresponds to the configuration of exactly one module.

In Figure 5(a) the POP algorithm is applied, Figure 5(bc) apply the IPOP algorithm. Figure 5(b) uses similarity to previous configurations as the secondary criterion while Figure 5(c) uses maximal contraction as the secondary criterion starting from the base.

Note that the configurations chosen to reach the point furthest to the right, $(3.0, 0.7)$, look very much alike in all figures. The reason is that the target point is close to the workspace boundary, where the density is low, thus restricting the shape of modules close to the base to be unique, as can be seen by comparing the most significant bits of the corresponding bit sequences in Figure 6. In contrast, the points $(0.55, 2.9)$ and $(0.4, 2.7)$ are in an area with very high point density and a significant redundancy is left after the geometric constraints are satisfied. The results from the three different figures for these two points give a good example of how the different algorithms resolve the redundancy. The POP algorithm resolves this redun-

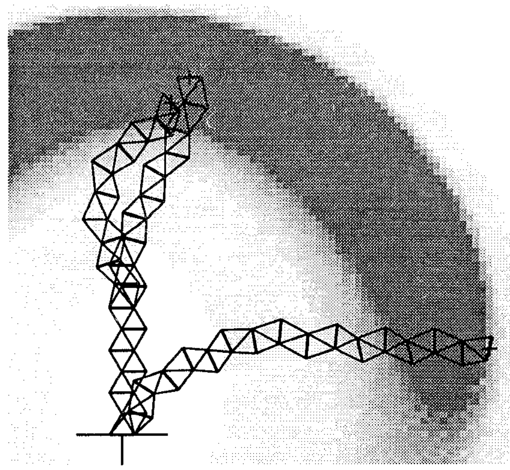
Figure 5(c): IPOP, favoring contracted modules

dancy randomly, i.e. a configuration is picked arbitrarily from those within an error threshhold. Hence the solutions in Figure 5(a) for the two points differ substantially. In Figure 5(b) modules are as similar as possible to the previous configuration with a higher weight given to modules close to the base. As a result the configurations for the upper two points are similar to each other and, as much as possible, to the configuration for the right point. Using maximal contraction as the secondary criterion, as shown in Figure 5(c), the solutions for the upper points are also similar to each other although the solution for each point is independent from the others.

5 Conclusion

This paper discusses an efficient way to solve the inverse kinematics of discretely actuated manipulators with many degrees of freedom.

A measure is developed which can test partial configurations for their suitability to reach a specific goal. Based on this measure an algorithm (POP) for the inverse kinematics is proposed that generates solutions satisfying the geometric constraints of the task. The remaining redundancy is resolved by additional constraints imposed on the algorithm. This improved form of the algorithm (IPOP) avoids abrupt transitions between configurations. Different constraints are discussed and demonstrated by simulations.

target (x,y)	(3.0,0.7)	(0.55,2.9)	(0.4,2.9)
high 24 bits	46 46 67 67	00 00 04 52	00 00 05 24
low 24 bits	00 77 12 06	52 51 65 21	02 22 22 12
	x = 3.000	x = 0.550	x = 0.397
POP	y = 0.700	y = 2.898	y = 2.900
	θ = -1.621	θ = 0.039	θ = 0.000
	e = 0.001	e = 0.002	e = 0.003
	46 46 00 66	46 46 31 31	46 46 31 31
	77 16 16 76	27 23 61 64	37 16 77 60
	x = 2.998	x = 0.550	x = 0.401
IPOP -	y = 0.699	y = 2.899	y = 2.899
previous	θ = -1.845	θ = -0.264	θ = 0.001
	e = 0.002	e = 0.001	e = 0.001
	46 46 00 66	00 00 00 00	00 00 00 00
	77 16 16 76	62 06 12 73	40 25 70 60
	x = 2.998	x = 0.550	x = 0.398
IPOP -	y = 0.699	y = 2.902	y = 2.901
contracted	θ = -1.845	θ = 0.039	θ = -0.567
	e = 0.002	e = 0.002	e = 0.002

Figure 6: Results for different end-effector coordinates from different algorithms

References

[1] S. Hirose, *Biologically Inspired Robots.* Oxford University Press, 1993.

[2] K. Ikuta, M. Nokata, and A. Satoshi, "Biomedical micro robots driven by miniature cybernetic actuator," in *Proceedings IEEE Micro Electro Mechanical Systems*, (Oiso, Japan), pp. 263–268, January 1994.

[3] E. Paljug, T. Ohm, and S. Hayati, "The JPL serpentine robot: a 12 dof system for inspection," in *Proceedings of the 1995 IEEE International Conference on Robotics and Automation*, vol. 3, (Nagoya, Japan), pp. 3143 – 3148, June 1995.

[4] S.-J. Kwon, Y. Youm, and W. K. Chung, "General algorithm for automatic generation of the workspace for *n*-link planar redundant manipulators," *ASME Transactions*, vol. 116, pp. 967 – 969, Sept 1994.

[5] H. Kobayashi and S. Ohtake, "Shape control of hyper redundant manipulator," in *Proceedings of the 1995 IEEE International Conference on Robotics and Automation*, vol. 3, (Nagoya, Japan), pp. 2803 – 2808, June 1995.

[6] K. Miura and H. Furuya, "Variable geometry truss and its application to deployable truss and space crane arm," *Acta Astronautica*, vol. 12, no. 7/8, pp. 599 – 607, 1985.

[7] P. Hughes, W. Sincarsin, and K. Carroll, "Trussarm-a variable-geometry-truss manipulator," *Journal of Intelligent Material Systems and Structures*, vol. 2, pp. 148–160, April 1991.

[8] H. Robertshaw and C. Reinholtz, "Variable geometry trusses," in *Smart Materials, Structures, and Mathematical Issues*, pp. 105 – 120, 1988.

[9] G. Chirikjian and J. Burdick, "Kinematically optimal hyper-redundant manipulator configurations," *IEEE Transactions on Robotics and Automation*, vol. 11, pp. 794–806, Dec 1995.

[10] K. R. Zanganeh and J. Angeles, "The inverse kinematics of hyper-redundant manipulators using splines," in *Proceedings of the 1995 IEEE International Conference on Robotics and Automation*, vol. 3, (Nagoya, Japan), pp. 2797 – 2802, June 1995.

[11] F. Park, "Distance metrics on the rigid-body motions with applications to mechanical design," *ASME Journal of Mechanical Design*, vol. 117, pp. 48–54, March 1995.

[12] I. Ebert-Uphoff and G. Chirikjian, "Efficient workspace generation for binary manipulators with many actuators," *Journal of Robotic Systems*, vol. 12, pp. 383–400, June 1995.

[13] G. Chirikjian and I. Ebert-Uphoff, "Numerical convolution on the euclidean group with applications to workspace generation," Tech. Rep. RMS 9-95-10, Dep. of Mechanical Engineering, Johns Hopkins University, 1995.