Journal of Intelligent and Robotic Systems **19:** 5–22, 1997.

Inverse Kinematics of Binary Manipulators Using a Continuum Model

GREGORY S. CHIRIKJIAN
Department of Mechanical Engineering, Johns Hopkins University, Baltimore, MD 21218; e-mail: gregc@jhu.edu

Abstract. Binary actuators have only two discrete states, both of which are stable without feedback. As a result, manipulators with binary actuators have a finite number of states. The major benefits of binary actuation are that extensive feedback control is not required, reliability and task repeatability are very high, and two-state actuators are generally very inexpensive, resulting in low cost robotic mechanisms. These manipulators have great potential for use in both the manufacturing and service sectors, where the cost of high performance robotic manipulators is often difficult to justify. The most difficult challenge with a binary manipulator is to achieve relatively continuous end-effector trajectories given the discrete nature of binary actuation. Since the number of configurations attainable by binary manipulators grows exponentially in the number of actuated degrees of freedom, calculation of inverse kinematics by direct enumeration of joint states and calculation of forward kinematics is not feasible in the highly actuated case. This paper presents an efficient method for performing binary manipulator inverse kinematics and trajectory planning based on having the binary manipulator shape adhere closely to a time-varying curve. In this way the configuration of the arm does not exhibit drastic changes as the end effector follows a discrete trajectory.

Key words: inverse kinematics, redundant manipulators

1. Introduction

The traditional assumption in robotics is that manipulators are actuated with continuous-range-of-motion actuators such as motors. However, there are many applications of robotic manipulators where a finite number of known locations need to be reached, and the robot's end-effector trajectory is not important as long as it is bounded. For these tasks, continuous range-of-motion machines are overkill.

A binary actuator is one type of discrete actuator which has only two stable states (denoted '0' and '1'). As a result, binary manipulators have a finite number of states. Major benefits of binary actuation are that extensive feedback control is not required, task repeatability can be very high, and two-state actuators are generally very reliable and inexpensive (e.g., pneumatic cylinders), thus resulting in reliable low cost robotic mechanisms.

A binary manipulator will never be able to reach an arbitrary specified position and orientation exactly (even with a large number of bits/actuators). This is not a problem that should preclude their use. The design of end-effector tools and fixtures with passive elements that allow for small errors should enable binary manipulators to perform pick-and-place tasks very reliably.

In principle, an analogy can be made between continuous vs. binary manipulators and analog vs. digital circuits. In the history of electronics and computing, digital devices replaced many of their analog counterparts because of higher reliability and lower cost – exactly the same reasons for developing a binary paradigm for robotics.

The methods developed in this paper represent a step in the formulation of a binary paradigm for robotic mechanisms. It is hoped that the development of a binary paradigm will lead to manipulators with lower cost, higher reliability and applicability, and more direct connectivity to computers.

While robotic hardware based on binary actuation is very inexpensive as compared to continuous range-of-motion robots (i.e., an order of magnitude cheaper), there are associated algorithmic and analytical issues in binary manipulator inverse kinematics, motion planning, and design which are currently much more difficult than the corresponding problems in the continuous range-of-motion case. The goal of this paper is to develop an efficient algorithm for binary manipulator inverse kinematics in order to make some of these algorithmic issues computationally tractable, and thus to make the cost benefits of binary manipulators attainable.

The number of configurations that a binary manipulator can attain is of the form 2^n, where n is the number of binary actuators. It is easy to see that for n large enough (e.g., $n \approx 40$) the explicit computation and storage of all workspace positions and/or orientations and corresponding sets of joint angles becomes impractical. To search for the ‘best’ set of bit values to reach a specific end-effector position and orientation would require 2^n computations of the forward kinematic equations. In this scenario, the best configuration is defined as the one which minimizes a measure of distance, or metric, between the desired and actual end-effector frame. Such a metric is presented in [9].

Given a set of m end-effector frames that are to be reached in a given order, then the set of best manipulator configurations to reach each frame individually is generally not optimal from the perspective of the whole path. That is, the m frames in the trajectory would have to be taken together as a unit to avoid the large changes in binary manipulator configuration which can correspond to close points in the workspace. Therefore, the best sequence of binary states to follow a trajectory would be attained using brute force search with $\mathrm{O}(2^n!/(2^n - m)!m!)$ calculations, which is the number of possible ways that m points can be chosen from 2^n. This number is huge even compared to 2^n.

Using a large number of binary actuators in a given manipulator is not unrealistic. These bits need not correspond to individual serial actuators. If for instance one revolute joint is resolved into eight bits, a five axis manipulator would have 40 bits/binary actuators. Therefore, efficient methods for computing binary

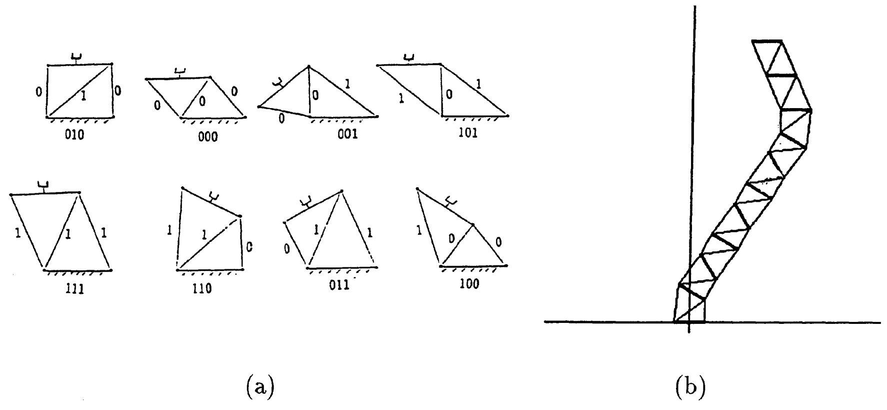


Figure 1. (a) Configurations of one module; (b) one configuration of a 30-bit binary manipulator.

manipulator inverse kinematics for large n are useful, and, in fact necessary, if such manipulators are to be widely used.

The manipulator architecture which is used in this paper to demonstrate the general problem of binary manipulator inverse kinematics is a variable geometry truss (VGT). Figure 1(a) illustrates all configurations of one VGT module, which consists of three prismatic actuators, each with two stable states (completely retracted, 0, or completely extended, 1). A schematic of a highly actuated structure composed of ten of these modules is shown in Figure 1(b) for one of its over one billion (2^{30}) configurations. This particular design is one of many possible binary manipulator morphologies. The concept of a VGT manipulator is not new (e.g., the concept is studied in [12]). In fact, the concept of a binary manipulator is not new either (e.g., see [6, 10]). However, the ability to efficiently plan smooth motions of a binary VGT manipulator is new.

Figure 2 illustrates the use of a binary VGT as an assistant to a person with disabilities. Other potential applications of binary manipulators include remote manipulation: (1) in outer space where bandwidth of communication is very low and time delays are very large, making it difficult to close the control loop; (2) in toxic/radioactive environments, where it can be safer and more cost effective to dispose of an inexpensive manipulator rather than cleaning and reusing it. Since binary actuators such as pneumatic cylinders tend to lock in one position if they fail, this also makes them failure tolerant, which is an important property of redundant systems (see, e.g., [7]).

While it might seem that 30 actuators would be expensive, binary actuators such as pneumatic cylinders with solenoid valves can be purchased at a retail cost of approximately \$50. Together with modest machining costs (since the actuators themselves are the major structural component), the author's students built a 15-bit binary manipulator for less than \$1000 in parts and materials. This is comparable to the cost of two or three high quality direct drive motors

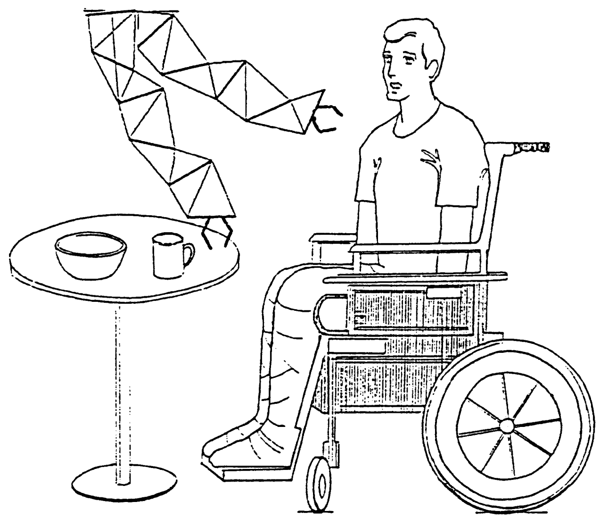


Figure 2. Illustration of binary manipulator application.

equipped with optical encoders. Furthermore, binary actuators are triggered with trivial interface circuitry from the parallel port of any personal computer, which circumvents the cost of expensive I/O boards.

The approach taken in this paper to solve the inverse kinematics problem and plan trajectories of binary manipulators is to use a ‘backbone curve’. The manipulator is made to act as though it is ‘glued’ to a continuous curve by selecting a binary state that causes the centerline of the manipulator to closely adhere to the curve. A new curve is defined for each target frame, and the dynamics of the manipulator carries it between states which approximate the specified curves. It is demonstrated that this approach leads to an evolution of manipulator states which vary smoothly, avoiding awkward and drastic changes in configuration, while at the same time reaching trajectory points/frames with reasonable accuracy.

The remainder of this paper is organized as follows: Section 2 reviews the literature. Section 3 reviews concepts from classical differential geometry needed to formalize our approach. This background deals primarily with the geometry of ‘backbone’ curves. Section 4 uses variational calculus to define optimally evolving sets of frames for a given curve. Section 5 introduces methods for making binary manipulators adhere as closely as possible to backbone curves. Section 6 examines the end-effector excursion from a desired discrete trajectory caused by open loop dynamics. Section 7 is the conclusion.

2. Literature Review

Due to the high cost/performance ratio of sophisticated robotic systems, a recent trend in ‘minimalist’ robotics has begun to gain momentum. For instance, there have been recent efforts to develop a reduced complexity paradigm in robotics (RISC) [1]. This trend runs counter to the idea that robots equipped with tactile feedback systems provide the solution to all industrial robotics problems. However, the minimalist school of thought has not previously included robotic manipulators with minimalist actuation which are completely devoid of joint level feedback (including position and velocity).

In the literature, sporadic efforts in binary actuation can be found, e.g., [10, 11, 6]. Despite the seemingly natural parallel between discretely actuated mechanical systems and the development of the computer, these efforts have not advanced much over the past quarter century due in large part to the lack of a framework in which to plan well-behaved motions of such manipulators.

Of course, a natural question that one might raise is how different binary manipulators are from current systems which use stepper motors, or pick-and-place machines used in circuit board fabrication, or even flexible automation systems in which technicians set joint stops. The answer is that just as in electronics, the benefits of binary robotic mechanisms are not only due to their discrete nature, but also due to the robustness of having only two states. Moreover, simple robots with only a few binary actuators cannot perform complicated tasks such as obstacle avoidance. Therefore, the true benefit of a binary paradigm for robotics can only be exploited if a relatively large number of actuated degrees of freedom are considered.

It is therefore important to determine how efficient the computational techniques used for binary manipulators will be as compared to standard continuous range of motion manipulators. That is, the low fabrication cost and high failure tolerance of binary manipulators alone are not sufficient to justify their use. The ease and speed with which binary manipulators can be used with inexpensive computers is also a big factor. For this reason, a brief (and by no means exhaustive) review of the redundancy resolution problem is now given in order to have a basis for comparison of algorithmic efficiency.

Recall that standard redundancy resolution schemes are based on variants of the expression

$$\dot{\mathbf{q}} = \mathbf{W}^{-1}\mathbf{J}^{\mathrm{T}}(\mathbf{J}\mathbf{W}^{-1}\mathbf{J}^{\mathrm{T}})^{-1}\dot{\boldsymbol{x}}_{\mathrm{ee}}, \tag{2.1}$$

where $\dot{\mathbf{q}}$ is the joint rate vector, $\dot{\boldsymbol{x}}_{\mathrm{ee}}$ is the desired end-effector velocity, $\mathbf{J}$ is the manipulator Jacobian, and $\mathbf{W}$ is a symmetric positive definite weighting matrix.

This is derived in a straight forward way. See [5] for issues pertaining to this pseudoinverse solution. Note that when $\mathbf{W} = \mathbf{1}$, the standard Moore–Penrose pseudoinverse results, which requires $\mathrm{O}(n)$ computations to compute for given $\mathbf{J}$ (where N is again considered a constant). This joint rate vector can then be integrated numerically or iterated to generate the appropriate joint angles to track a given trajectory. See [3] for a more detailed analysis of, and alternatives to, the Jacobian-based formulation of redundant manipulator inverse kinematics.

If a binary manipulator is to be competitive in terms of on-line performance, the computational complexity of the inverse kinematics/planning algorithm must be competitive with standard manipulators. The remainder of this paper presents an approach based on the geometry of continuous backbone curves.

3. Classical Geometry of Curves

It is assumed here that regardless of mechanical implementation, the important macroscopic features of a highly actuated manipulator (binary or otherwise) can be approximated by a *backbone curve* and associated set of reference frames which evolve along the curve. Using this model, inverse kinematics and trajectory planning tasks are reduced to the determination of the proper time varying behavior of the backbone curve. While similar approaches have been used in the past for hyper-redundant manipulators, e.g., [2, 3], the application of this general framework to manipulators with a finite number of states is new.

In this section classical techniques for describing the intrinsic geometry of curves are reviewed. This background will be essential for the inverse kinematics method presented in subsequent sections.

In the classical differential geometry of curves, curves are parametrized by arc-length, L. That is, a smooth curve $\mathbf{x}(L) = [x_1(L), x_2(L), x_3(L)]^T \in \mathbb{R}^3$ satisfies the condition:

$$\frac{\mathrm{d}\mathbf{x}}{\mathrm{d}L} \cdot \frac{\mathrm{d}\mathbf{x}}{\mathrm{d}L} = 1.$$

Without loss of generality, we will deal with unit length curve segments, and take $L = 0$ to be the base of the curve, and $L = 1$ to be the end. A coordinate frame is fixed at the base of the curve, which is also the base of the manipulator. All positions and orientations are measured relative to this fixed frame.

The *Frenet–Serret Apparatus* [8] is used to define how a curve bends and twists locally. The Frenet–Serret apparatus defines a unique frame for each point on the curve. For each L there is an associated frame consisting of the three orthonormal vectors: $\mathbf{u}(L)$, $\mathbf{n}(L)$, and $\mathbf{b}(L)$. These vectors are respectively termed the tangent, normal, and binormal vectors and are defined as:

$$\mathbf{u} = \frac{\mathrm{d}\mathbf{x}}{\mathrm{d}L}, \qquad \mathbf{n} = \frac{1}{\kappa}\frac{\mathrm{d}\mathbf{u}}{\mathrm{d}L}, \qquad \mathbf{b} = \mathbf{u} \times \mathbf{n}, \tag{3.2}$$

where

$$\kappa(L) = \left(\frac{\mathrm{d}\mathbf{u}}{\mathrm{d}L} \cdot \frac{\mathrm{d}\mathbf{u}}{\mathrm{d}L}\right)^{1/2}$$

is called the *curvature function*. Note that for nonzero curvature, $\mathbf{u}\cdot\mathbf{u} = 1$ implies $\frac{\mathrm{d}}{\mathrm{d}L}(\mathbf{u}\cdot\mathbf{u}) = 0$, and therefore $\mathbf{u}\cdot\mathbf{n} = 0$.

From the above definitions, the following relationships can be derived:

$$\frac{\mathrm{d}\mathbf{u}}{\mathrm{d}L} = \kappa\mathbf{n}; \qquad \frac{\mathrm{d}\mathbf{n}}{\mathrm{d}L} = \tau\mathbf{b} - \kappa\mathbf{u}; \qquad \frac{\mathrm{d}\mathbf{b}}{\mathrm{d}L} = -\tau\mathbf{n}, \tag{3.3}$$

where $\tau(L)$ is an additional intrinsic function of the curve, termed *torsion*:

$$\tau = \frac{1}{\kappa^2}\mathbf{u} \cdot \left(\frac{\mathrm{d}\mathbf{u}}{\mathrm{d}L} \times \frac{\mathrm{d}^2\mathbf{u}}{\mathrm{d}L^2} \right).$$

$\kappa(L)$ can be physically interpreted as the bending of the curve in the plane spanned by $\mathbf{u}(L)$ and $\mathbf{n}(L)$, while $\tau(L)$ measures the rate of change of orientation of that plane (whose normal is $\mathbf{b}(L)$). Let $\mathbf{Q}(L) = [\mathbf{u}(L)\ \mathbf{n}(L)\ \mathbf{b}(L)] \in \mathrm{SO}(3)$ be the rotation matrix representing the orientation of Frenet–Serret frame at $\mathbf{x}(L)$ for a given curve.[⋆] From Equation (3.3), it can be seen that the rate of change of $\mathbf{Q}$ with respect to L is governed by the equation:

$$\frac{\mathrm{d}\mathbf{Q}}{\mathrm{d}L} = \mathbf{Q}\boldsymbol{\Lambda}, \tag{3.4}$$

where

$$\boldsymbol{\Lambda} = \begin{bmatrix} 0 & -\kappa & 0 \\ \kappa & 0 & -\tau \\ 0 & \tau & 0 \end{bmatrix}.$$

In the planar case, where $\tau = 0$, a backbone curve and associated set of frames can be completely described in terms of curvature alone as follows:

$$\begin{aligned} x_1(L) &= \int_0^L \cos\theta(\sigma)\,\mathrm{d}\sigma, \qquad x_2(L) = \int_0^L \sin\theta(\sigma)\,\mathrm{d}\sigma, \\ \mathbf{Q}(L) &= \mathrm{ROT}[\mathbf{e}_3, \theta(L)], \end{aligned} \tag{3.5}$$

where $\theta(L) = \int_0^L \kappa(\sigma)\,\mathrm{d}\sigma$, and $\mathrm{ROT}[\boldsymbol{v}, \alpha]$ is a counterclockwise rotation about the unit vector $\boldsymbol{v}$ by angle α.

In our application, the curves will vary their shape with time, so $\mathbf{x} = \mathbf{x}(L, t)$, $\kappa = \kappa(L, t)$, and $\tau = \tau(L, t)$. Manipulators such as a VGT can stretch as well as bend. This is described using a *reparametrization* of the curve. A reparametrization of the form $L = L(s, t)$ is possible, where the function $L(\cdot, t)$ is strictly increasing in s for all possible t, $L(0, t) = 0$, $L(1, t) = 1$, and s is the new curve parameter. Time dependence will be expressed explicitly only when it is needed in this paper, otherwise all quantities are written as strict functions of L or s.

4. Choosing Optimal Frames by Variational Calculus

A better fit between a manipulator and backbone curve is generally achieved if the frames which evolve along the curve do not vary rapidly. This is because the fitting process, as will be described in Section 5, involves matching frames fixed on the curve at discrete points with frames fixed in the manipulator. If the

[⋆] SO(3) is the group of 3×3 orthogonal matrices with determinant $+1$.

curve frames vary rapidly, then the manipulator must twist and extend/contract more to keep up with the rapid changes, and the risk of hitting actuator limits increases.

Two problems with the use of the Frenet–Serret Apparatus are: (1) arc-length spacing along the curve is not necessarily optimal; (2) the assignment of Frenet frames to the curve is not necessarily the most smoothly evolving set of frames that can be chosen. The calculus of variations is used in this section to determine the best evolution of frames for our application. The problem addressed here, which is the modification of frames attached to a specified curve, should not be confused with the problem solved in [3], which generates smoothly varying curves using techniques from variational calculus.

Recall [4] that a smooth vector function $\boldsymbol{q}(s) \in \mathbb{R}^N$ will extremize the integral:

$$I = \int_{s_0}^{s_1} F(s, \boldsymbol{q}(s), \dot{\boldsymbol{q}}(s))\, \mathrm{d}s, \tag{4.6}$$

(where $\dot{} = \mathrm{d}/\mathrm{d}s$) only if it is a solution to the Euler–Lagrange equations:

$$\frac{\partial F}{\partial q_j} - \frac{\mathrm{d}}{\mathrm{d}s}\left(\frac{\partial F}{\partial \dot{q}_j}\right) = 0 \quad \text{for } j = 1, \ldots, N, \tag{4.7}$$

with appropriate boundary conditions $\boldsymbol{q}(s_0) = \boldsymbol{q}_0$ and $\boldsymbol{q}(s_1) = \boldsymbol{q}_1$.

This is now used to derive optimally evolving frames about a given curve shape because the Frenet frames are not necessarily the best frames to use for a specified backbone curve. An extension to the Frenet–Serret Apparatus which is relevant to hyper-redundant and binary manipulator kinematics is to include a *roll distribution*, $R(L)$, as seen in Figure 3. $R(L)$ is a measure of twist about the backbone curve tangent vector at each L measured in the Frenet Frames. While the macroscopic shape of highly actuated manipulators can be captured using a curve, the roll distribution specifies how a mechanism should twist about the curve relative to the Frenet frames. The orientation of the *modified* frame is given by

$$\mathbf{Q}_{\mathrm{M}} = \mathrm{ROT}[\mathbf{u}, R]\mathbf{Q} = \mathbf{Q}\mathrm{ROT}[\mathbf{e}_1, R],$$

where $\mathbf{e}_i$ represents the ith natural basis vector of $\mathbb{R}^3$ in the frame fixed at the base of the curve. The rate of change of the modified frame is then:

$$\frac{\mathrm{d}\mathbf{Q}_{\mathrm{M}}}{\mathrm{d}L} = \mathbf{Q}_{\mathrm{M}}\boldsymbol{\Lambda}_{\mathrm{M}}, \tag{4.8}$$

where

$$\boldsymbol{\Lambda}_{\mathrm{M}} = \begin{bmatrix} 0 & -\kappa\cos R & \kappa\sin R \\ \kappa\cos R & 0 & -\left(\tau + \frac{\mathrm{d}R}{\mathrm{d}L}\right) \\ -\kappa\sin R & \left(\tau + \frac{\mathrm{d}R}{\mathrm{d}L}\right) & 0 \end{bmatrix} = -\boldsymbol{\Lambda}_{\mathrm{M}}^{\mathrm{T}}. \tag{4.9}$$

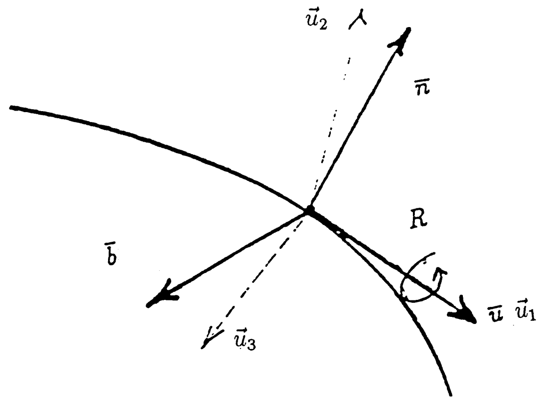


Figure 3. Definition of modified frames.

A set of frames which smoothly evolve along the curve can be defined by squaring the norm of $\mathrm{d}\mathbf{Q}_\mathrm{M}/\mathrm{d}L$, integrating over the length of the curve, and minimizing this quantity with respect to $R(L)$ for given $\kappa(L)$ and $\tau(L)$, and boundary conditions such as $\mathbf{Q}_\mathrm{M}(0) = \mathbf{1} = [\mathbf{e}_1, \mathbf{e}_2, \mathbf{e}_3]$ and $R(1) = R_\mathrm{ee}$. That is, we use the cost function

$$\begin{aligned} F\left(R, \frac{\mathrm{d}R}{\mathrm{d}L}\right) &= \mathrm{tr}\left(\frac{\mathrm{d}\mathbf{Q}_\mathrm{M}}{\mathrm{d}L}\frac{\mathrm{d}\mathbf{Q}_\mathrm{M}^\mathrm{T}}{\mathrm{d}L}\right) \\ &= \mathrm{tr}(\mathbf{Q}_\mathrm{M}\mathbf{\Lambda}_\mathrm{M}(\mathbf{Q}_\mathrm{M}\mathbf{\Lambda}_\mathrm{M})^\mathrm{T}) \\ &= -\mathrm{tr}(\mathbf{Q}_\mathrm{M}\mathbf{\Lambda}_\mathrm{M}\mathbf{\Lambda}_\mathrm{M}\mathbf{Q}_\mathrm{M}^\mathrm{T}) \\ &= -\mathrm{tr}({\mathbf{\Lambda}_\mathrm{M}}^2) = 2\left(\tau + \frac{\mathrm{d}R}{\mathrm{d}L}\right)^2 + 2\kappa^2. \end{aligned}$$

Solving the Euler–Lagrange equation with $R(L)$ as the variable, the optimal roll distribution is

$$R(L) = R(0) + \frac{\mathrm{d}R}{\mathrm{d}L}(0)L - \int_0^L [\tau(\sigma) - \tau(0)]\,\mathrm{d}\sigma.$$

This defines frames assigned to spatial curves in such a way that they vary as little as possible while still having the curve tangent as one of the axes of the curve frame. $R(0)$ specifies the roll at the fixed base (and so $R(0) = 0$), and $\frac{\mathrm{d}R}{\mathrm{d}L}(0)$ is a free parameter which is easily adjusted to provide the desired roll at the end effector.

One interpretation of the above equations is that for planar curves and helicies (where $\tau = \text{const}$), the Frenet frames are the optimal set if $R(0) = R(1) = 0$. The more the torsion varies with curve parameter, the more adjustment is needed relative to the Frenet frames for the modified frames to vary as little as possible along the curve while observing boundary conditions. Similar results are obtained when one simultaneously attempts to redistribute and rotate modified frames along the curve.

5. Making a Binary Manipulator Adhere to a Backbone Curve

Consider the general problem: Given a binary manipulator and a backbone curve specifying the shape which we want the manipulator to attain, how do we fit the manipulator to the curve knowing that there will generally be no exact fit?

In the case of a manipulator with continuous range-of-motion actuators this is easily solved by ‘constructing’ the configuration around the curve, and then extracting joint displacements needed to make the manipulator attain this configuration. The curve is divided up into equal increments of s, and the manipulator is divided into corresponding modules. The inverse kinematics of each manipulator module is used to ensure that its ends are fixed to the curve frames at the ends of the corresponding curve segment. This approach is described in detail in [2]. For the planar VGT structure this means that the center of each transverse VGT link is fixed to a point on the curve, and the orientation of these elements is fixed by aligning them with the curve normal. Doing this completely determines actuator displacements and causes the curve to be the centerline of the manipulator.

In the case of binary manipulators, the problem is a bit more difficult. Because there is generally no exact fit, the next best thing is to minimize a measure of distance between the manipulator and the curve, or more precisely between the manipulator in its allowable states and itself if it could be fit exactly to the curve. We therefore need a measure of distance between manipulator configurations.

The approach taken in this paper to measure the distance between two manipulator configurations is to sum distances between characteristic points on a manipulator while it is in one configuration, and the same points on the manipulator in the other configuration. The concept of distance between manipulator configurations used here is illustrated with a planar variable geometry truss in Figure 4. Here the characteristic points are the vertices on the left and right sides of the truss. The ideal configuration is the one which is fit exactly to the curve, and one binary configuration is shown. The error measure is simply

$$E = \sum_{i=1}^{N} E_i = \sum_{i=1}^{N} \left(\|\Delta\mathbf{x}_i^R\|^2 + \|\Delta\mathbf{x}_i^L\|^2 \right). \tag{5.10}$$

$\Delta\mathbf{x}_i^R$ is the difference of the position of the ith vertex on the right side of the truss in the two configurations, and $\Delta\mathbf{x}_i^L$ is the difference of the position of the ith vertex on the left side of the truss in the two configurations. This measure accounts for both rotational and translational differences between the ideal and actual manipulator configurations. In the case of cascaded spatial platform manipulators this is easily generalized to the sum of the magnitudes of the difference of all six vertices at each level of a platform. For other measures of distance between rigid bodies which could be applied to this problem see [9].

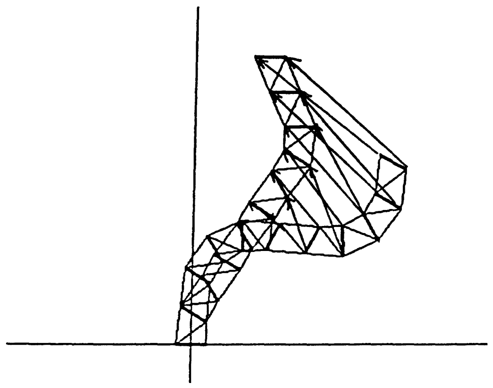


Figure 4. Measuring 'distance' between two configurations.

For fitting a binary manipulator to a curve, the distance between each possible configuration of the manipulator and the configuration that would lead the manipulator to exactly adhere to the curve could be measured. Because a binary manipulator will have 2^n states, the direct enumeration of all possible configurations, computation of E, and selection of the configuration for which E is minimal is impractical for $n \gg 25$. However, a qualitatively good fit of a binary manipulator to a given curve can be attained by sequentially fitting modules from the base of the manipulator to the curve. In this way, only $\mathrm{O}(n)$ computations are required.

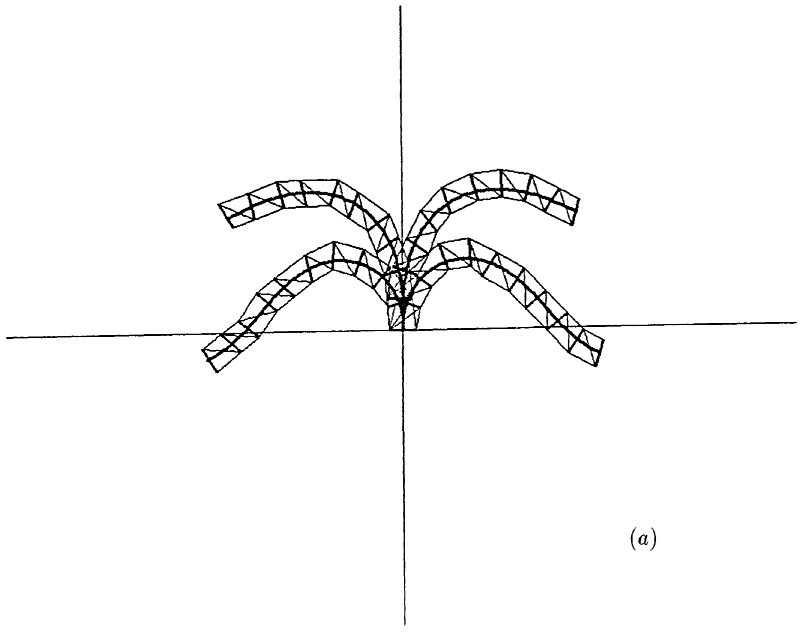
For example, with the planar binary variable geometry truss manipulator discussed in the introduction, the eight (2^3) configurations in the first module are tested, and the configuration for which E_1 is smallest is kept. Then, with the bits (actuator lengths) in the first module fixed, the configuration of the second module for which E_2 is minimized is determined, and it's configuration is fixed, etc. If the value of E_i is the same for two different binary configurations, then secondary criteria such as chosing the configuration which is least different from the current configuration is one possible way to choose. If both conditions are the same for two different configurations, one of the configurations can be chosen randomly.

This procedure is shown in Figure 5 for two binary manipulators which must adhere to arclength parametrized curves which have curvature functions of the form:

$$\kappa(L) = \pi(a_1 \cos \pi L + a_2 \sin \pi L)$$

for $(a_1, a_2) \in \{(-1.00, -1.00), (0.00, 1.00), (1.00, 1.00), (0.00, -1.00)\}$, corresponding to the end effector positions $(x_1(1), x_2(1)) \in \{(-0.73, -0.10), (0.65, 0.42), (0.73, -0.10), (-0.65, 0.42)\}$.

A natural question that arises is how to choose curvature functions to restrict the allowable range of backbone curve shapes. This issue is addressed in [2, 3]


(a)

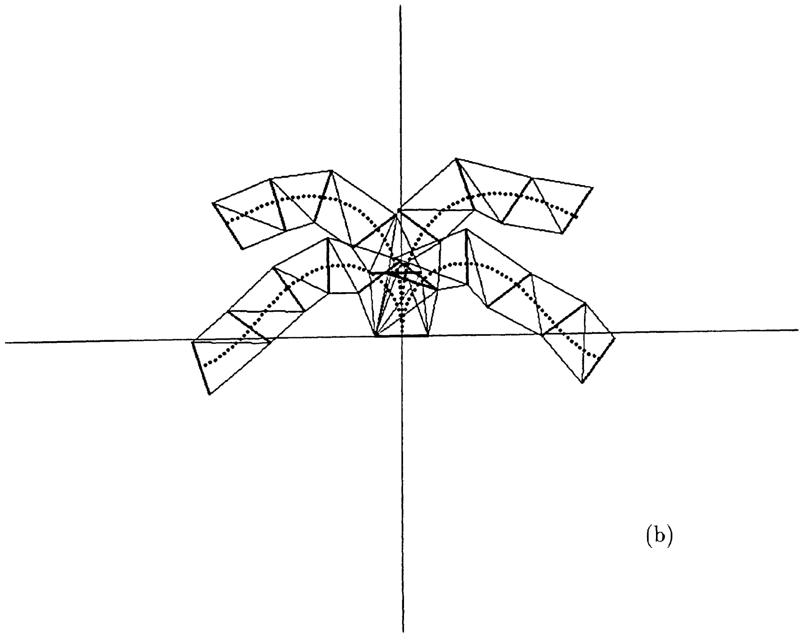

(b)

Figure 5. Manipulators with stops $(0.055, 0.145)$ fit to curves using the sequential approach; (a) 60 bits; (b) 30 bits.

for the case of continuous actuation. In the context of the current examples it suffices to say that if a larger workspace is desired then curvature functions with lower frequency should be chosen. Higher frequency curvature functions can be of benefit in highly constrained environments where the manipulator needs to occupy a small volume. In practice, a binary manipulator should be designed/synthesized with a particular application (and specific backbone curve properties) in mind so that the fit between the manipulator and curve can be made very good.

In Figure 5(a), the fit is so good because there are two bits (four states) per leg, so that each leg can attain the lengths $(0.055, 0.085, 0.115, 0.145)$. This allows the manipulator to bend and track curves with high curvature. In fact it works quite well for such a wide range of curvatures that if this manipulator were cut in half, it could still track a scaled version of the same sets of curves in Figure 5(a). Figure 5(b) shows this 30 bit manipulator (also 6 bits per truss module) adhering to these curves.

Many variants on this theme are also possible. For instance, because the fit of the end-effector frame to a given target frame is generally more important than the internal fit of the manipulator to frames along the curve, one can treat the most distal set of modules as a unit. This allows greater freedom and increases accuracy of the end-effector position and orientation at the cost of computational time. Another approach is to do a ‘running’ minimization. That is, instead of minimizing $E_1, E_2, \ldots, E_n$ and fixing states sequentially, one minimizes E_1+E_2 fixing only the bits in module 1, then minimizes $E_2 + E_3$ fixing the bits in module 2, etc. This has the effect of averaging out small deviations in curve shape that could otherwise cause the manipulator configuration to separate from the backbone curve. Many other variants also follow naturally.

6. The Trajectory Tracking Problem

In the previous section we saw that if a binary manipulator is designed so as to have a wide enough range of motion, it is possible for the configuration to closely adhere to a specified curve. In doing so, the end-effector frame coincides closely with the most distal modified frame of the backbone curve. The computational cost of this approach is $O(n)$, which makes it competitive with continuously actuated redundant manipulators using the unity weighted ($\mathbf{W} = \mathbf{1}$) pseudoinverse in Equation (2.1).

However, this is only part of the problem. When using a binary manipulator to do pick-and-place, it is desirable for the manipulator not to radically change configuration for small desired changes of the end effector. It is also desirable for the end effector to not take large excursions while doing pick-and-place between close points. This section shows that the continuum (backbone curve) approach satisfies these requirements.

In Subsection 6.1, the assumption of near constant speed of open loop pneumatic binary actuators is justified by appropriate choice of mechanical constants in the manipulator design. In Subsection 6.2 the deviation which the end effector takes while following a discretized trajectory is illustrated with two examples under the assumption that the speed of the actuator is constant over its stroke.

6.1. ANALYSIS OF BINARY ACTUATOR DYNAMIC BEHAVIOR

One criticism of the use of binary actuation comes from the perception that two state actuation is necessarily jerky. In this subsection it is shown that by appropriate mechanical design, this need not be the case.

Consider one of the simplest and most inexpensive binary actuators: a pneumatic cylinder. If the mass of the plunger is m, then in the absence of gravity the equation of motion for the cylinder is simply:

$$m\frac{\mathrm{d}^2x}{\mathrm{d}t^2} = PA - c\frac{\mathrm{d}x}{\mathrm{d}t}, \tag{6.11}$$

where P is the pressure in the tube, A is the area of the plunger head, c is a viscous friction coefficient, and x is the amount the plunger is displaced from a datum. For convenience we will use the variable $F = PA$, which is the force the actuator is able to provide.

Much can be learned from this simple model. For instance, if the datum is chosen such that $x(0) = 0$, and the viscosity and pressure are very high relative to the mass, then $m/c \to 0$ and we get

$$x(t) \approx \frac{F}{c}t. \tag{6.12}$$

Since pneumatic cylinders have a very high force to weight ratio, and air inlet/outlet can be made very restrictive, the assumption that m/c is small is not unreasonable. While energy dissipated due to damping might be considered wasteful, it is not necessarily more wasteful than energy dissipated due to resistive heating in motor coils and the power electronics used for control of continuous motion actuators.

Now then, we see that if the damping is high enough, a single binary pneumatic actuator can be made to behave quite nicely – the speed is approximately constant, and can be adjusted to the user's needs. In order to avoid wear and tear and prevent high frequency transients, one can insert short springs (or rubber bumpers) in the cylinder at both ends. This helps to ramp the speed up to steady state when the actuator is starting from rest, and allows for gradual deceleration at the other end. These effects operate over a small portion of the total stroke, and are therefore not significant for motion planning purposes compared to the behavior of the actuators over the remaining portion of the stroke.

An elementary analysis of the more complicated case of a whole manipulator composed of highly damped pneumatic actuators yields essentially the same result. That is, the manipulator dynamical equations are:

$$\mathbf{M}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{C}(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{q}} + \mathbf{g}(\mathbf{q}) = \mathbf{f} - \boldsymbol{\Delta}\dot{\mathbf{q}},$$

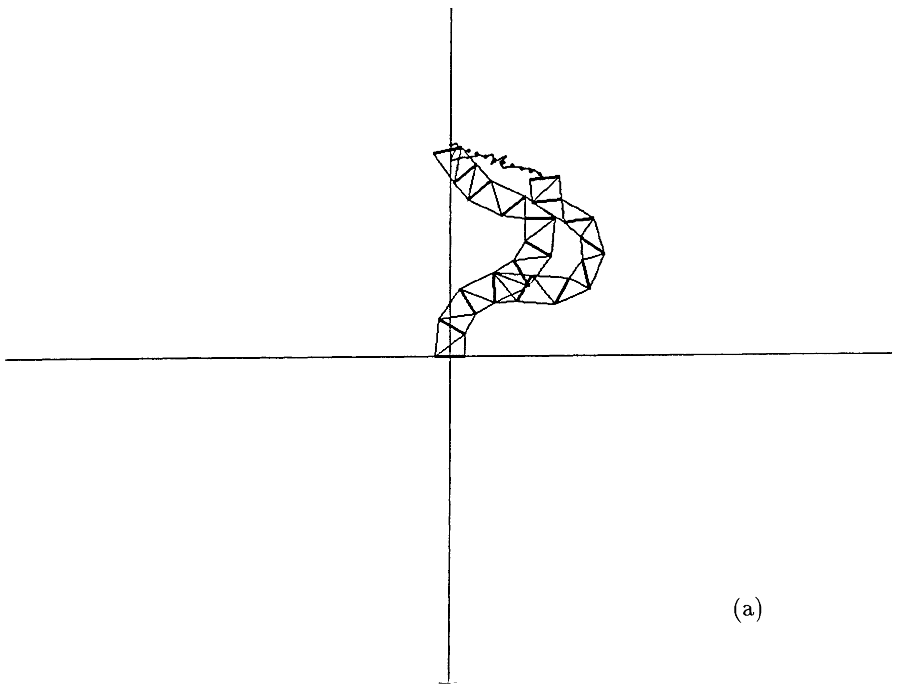
where $\mathbf{f}$ is the vector due to pressure, and $\boldsymbol{\Delta}$ is the diagonal viscosity matrix. Under the assumption of slow (quasistatic) motion, and very high pressure and viscosity, we can essentially neglect the left hand side of the manipulator dynamics. In this case, we simply get the linear behavior:

$$\mathbf{q}(t) = \mathbf{q}_0 + t\boldsymbol{\Delta}^{-1}\mathbf{f}.$$

6.2. APPROXIMATE TRAJECTORY TRACKING WITH BINARY MANIPULATORS

In Figure 6(a) we see a 30 bit manipulator (one bit per leg) with stops (0.075, 0.125) following a discrete trajectory. The desired trajectory consists of 11 parallel frames with origins equally spaced along a line. Not shown are the nine intermediate binary configurations the manipulator attains in order for the end effector to reach the desired points as closely as possible using the sequential fitting procedure discussed earlier. However, what is shown is the continuous path produced by the end effector as the manipulator changes states. It is assumed that all the actuators move with the same constant speed from one state to another. This assumption was justified in the previous subsection as an approximation to the fact that each actuator is heavily damped. This simple model is used only to illustrate that the deviation of the end-effector position during transition between binary states is small when using the approach presented in this paper, and not to characterize the end-effector motion in detail. In Figure 6(b) the same trajectory is tracked using the 60 bit manipulator with the wider stops used in Figure 5. Note that the deviation of the continuous trajectory is on the same order as in the 30-bit case.

There are several lessons to be learned from these numerical experiments: (1) binary manipulators constructed with fewer modules (each with more binary states) can be treated using the same approach as more ‘snakelike’ binary manipulators; (2) the performance of 60-bit binary manipulators doing trajectory tracking is not necessarily better than for 30-bit manipulators. Both of these have implications to the design of binary manipulators. For instance, since manufacturing errors accumulate more rapidly when the structure consists of more serial units, the benefits gained by structures with more of a parallel nature need to be weighed against the loss of workspace volume. Furthermore, the cost of additional binary actuators needs to be weighed against the diminishing returns they provide when using the serial fitting procedure, and the explosive computation time that results when using an exhaustive search.



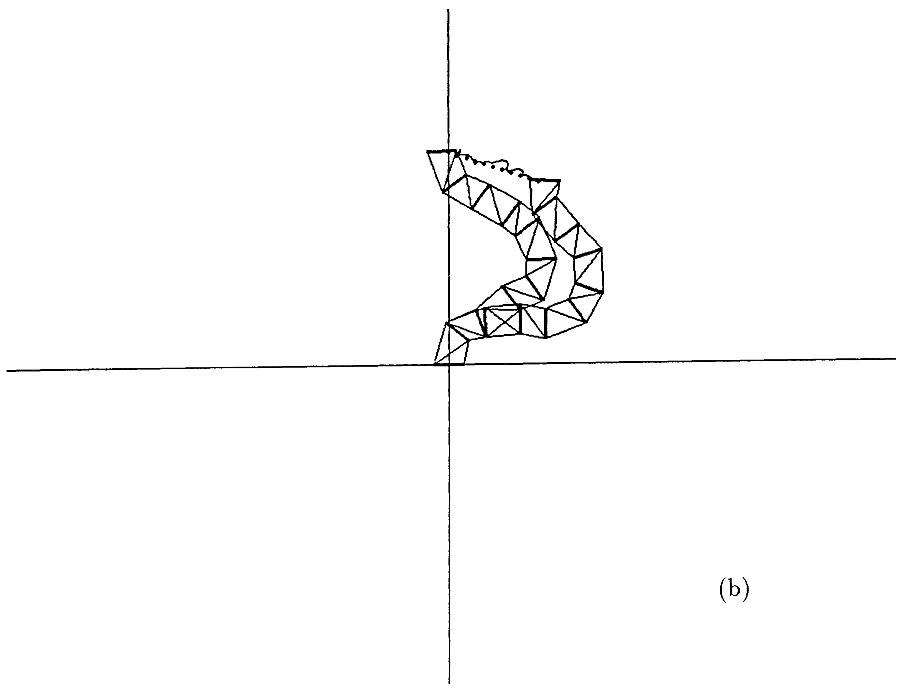


Figure 6. Trajectory tracking using a: (a) 30-bit manipulator with stops (0.075, 0.125); (b) a 60-bit manipulator with stops (0.055, 0.145).

It is also worth noting that the deviation from straight line paths in Figure 6 is due to the nonlinearity of the forward kinematic equations. For small motions between similar configurations, as would be the case if binary states are very fine, the linearized approximation of the forward kinematic equations becomes good. This means that linear (constant speed) joint motion leads to almost linear interpolation between adjacent points on the trajectory. In Figures 6(a) and (b), the deviation from the straight line is on the same order because the spacings between actuator states are on the same order. Therefore, the nonlinear effects of the forward kinematics are comparable.

7. Conclusions

This paper presented an efficient method for the inverse kinematics of robotic manipulators with binary (two-state) actuators. Methods of classical differential geometry were used to define a backbone curve and associated set of frames. These frames were used as a reference to which a binary manipulator was made to adhere. The algorithm produces binary manipulator shapes and end-effector frames which are close to those specified. An analysis of actuator dynamics showed that with high enough damping and pressure for a given manipulator mass, the speed of each actuator can be approximated as a constant. This assumption was used to show that end effector deviations from a given trajectory can be made small even though no feedback control is used.

A major benefit of this inverse kinematics approach is that the 2^n configurations attainable by a binary manipulator with n actuators do not have to be explicitly searched for each point on an end-effector trajectory, and the presented method only requires $\mathrm{O}(n)$ computations. Another benefit of this approach is that wide variations in configuration are avoided since the whole arm shape is specified. This method is particularly well suited for manipulators with hybrid structure such as a cascade of platform manipulators.

Acknowledgements

This work was made possible by NSF National Young Investigator Award IRI-9357738 from the Robotics and Machine Intelligence Program, and Presidential Faculty Fellow Award IRI-9453373. Thanks go to Dr. David Lees and Ms. Imme Ebert-Uphoff for their constructive comments during the writing of this paper.

References

1. Canny, J. and Goldberg, K.: A RISC paradigm for industrial robotics, *Technical Report ESRC 93-4/RAMP 93-2*, Engineering Systems Research Center, University of California at Berkeley, 1993.
2. Chirikjian, G. S. and Burdick, J. W.: A modal approach to hyper-redundant manipulator kinematics, in: *IEEE Trans. Robotics Automat.*, 1994, pp. 343–354.

3. Chirikjian, G. S. and Burdick, J. W.: Kinematically optimal hyper-redundant manipulator configurations, in: *IEEE Trans. Robotics Automat.*, 1995, pp. 794–806.
4. Ewing, G. M.: *Calculus of Variations With Applications*, W. W. Norton and Co. Inc., New York, 1969.
5. Klein, C. A. and Huang, C. H.: Review of the pseudoinverse for control of kinematically redundant manipulators, *IEEE Trans. Systems Man Cybernet.* **SMC-13**(2) (1983), 245–250.
6. Koliskor, A. Sh.: The l-coordinate approach to the industrial robot design, in: *5th IFAC Symposium*, Suzdal, USSR, April 22–25, 1986, pp. 108–115.
7. Maciejewski, A. A.: Fault tolerant properties of kinematically redundant manipulators, in: *IEEE Conf. Robotics Automat.*, 1990, pp. 638–642.
8. Millman, R. S. and Parker, G. D.: *Elements of Differential Geometry*, Prentice-Hall, Englewood Cliffs, NJ, 1977.
9. Park, F. C.: Distance metrics on the rigid-body motions with applications to mechanism design, *ASME J. Mech. Design* **117** (1995), 48–54.
10. Pieper, D. L.: The kinematics of manipulators under computer control, PhD Dissertation, Stanford University, 1968.
11. Roth, B., Rastegar, J., and Scheinman, V.: On the design of computer controlled manipulators, in: *First CISM-IFTMM Symposium on Theory and Practice of Robots and Manipulators*, 1973, pp. 93–113.
12. Salerno, R. J., Reinholtz, C. F., and Robertshaw, H. H.: Shape control of high degree-of-freedom variable geometry trusses, in: *Proceedings of the Workshop on Computational Aspects in the Control of Flexible Systems, Part 2*, Williamsburg, VA, July 12–14, 1988.