Shape Control of High Degree-of-Freedom Variable Geometry Truss Manipulators

by

Robert James Salerno

Thesis submitted to the Faculty of the
Virginia Polytechnic Institute and State University
in partial fulfillment of the requirements for the degree of

Master of Science

in

Mechanical Engineering

APPROVED:

C. F. Reinholtz, Chairman

S. G. Dhande

H. H. Robertshaw

June, 1989
Blacksburg, Virginia

Shape Control of High Degree-of-Freedom Variable Geometry Truss Manipulators

by

Robert James Salerno

Committee Chairman: Charles F. Reinholtz
Mechanical Engineering

Abstract

Variable Geometry Trusses (VGT's) can be used as the fundamental building blocks in constructing long-chain, high degree-of-freedom manipulators. This thesis focuses on the kinematics of two such manipulators. It also illustrates how the concept of shape control can be applied to simplify the computational aspects of controlling these devices. To serve as examples, algorithms are developed for the control of both a thirty degree-of-freedom planar manipulator and a sixty degree-of-freedom spatial manipulator. Based on a review of the literature, this work appears to be the first attempt to develop real-time, position control strategies for such highly-dexterous manipulators.

Acknowledgements

I would like to express my sincere appreciation to the chairman of my advisory committee, Prof. Charles F. Reinholtz. I am grateful not only for his many helpful suggestions and insights, but also for his positive outlook, quick wit and valued friendship. Studying under Prof. Reinholtz has emphasized what a pleasure it is to work with individuals who are enthusiastic about, and enjoy, their profession.

I must also extend my thanks to Prof. Sanjay G. Dhande, whose strong geometric modeling background contributed greatly to this work. I am pleased to have had him as a member of my advisory committee.

I wish to thank the other members of my advisory committee, Prof. Harry H. Robertshaw and Prof. Robert H. Fries, and my friends Paul Tidwell and Babu Padmanabhan for their many helpful comments

I am deeply grateful to my parents, . Their support and encouragement throughout my life, have enabled me to accomplish what I have. I am especially grateful to my father, whose tremendous enthusiasm for science and engineering has profoundly influenced my views

on education and the engineering profession.

Lastly, I must express my most heartfelt gratitude to my wife, . Her encouragement and understanding, never waivered despite many late nights and missed suppers.

Contents

List of Figures

Chapter 1

Introduction

Robotic manipulators in use today are generally serially connected devices which possess six or fewer degrees-of-freedom. As the name implies, serial manipulators are composed of a series of links connected to form an open chain. Each link of the manipulator is, in effect, a cantilever beam carrying the full load of all the links further out in the chain. Consequently, serial manipulators are inherently compliant and have relatively poor load carrying capacities.

In contrast, parallel manipulators are composed of links connected to form one or several closed loops. This thesis is concerned with a special subset of parallel manipulators referred to as Variable Geometry Truss Manipulators. Simply stated, a Variable Geometry Truss (VGT) is a statically determinant truss which contains some number of variable length members. These extensible members allow the truss to change shape in a precise controllable manner. Like a common static truss, if properly de-

signed, a VGT will contain only two-force (pure tension/compression) members. This structural characteristic results in an extremely high stiffness-to-weight ratio, which enables many VGT manipulators to carry heavy payloads. Some VGT geometries are capable of of supporting many other VGT's extended in a "snake-like" manner. This is, in essence, a very dexterous long-chain manipulator. One convenient measure of dexterity is the number of degrees-of-freedom a manipulator possesses in excess of the minimum number required [1]. This is sometimes referred to as the number of degrees-of-redundancy of the manipulator. For example, a four degree-of-freedom planar manipulator, performing a task which requires only three degrees-of-freedom, is said to have a dexterity rating of one, or a single degree-of-redundancy. In general, such a manipulator would be capable of achieving one infinity of different solutions for a specified end position and orientation.

The preceding discussion assumed that high dexterity is desirable for some tasks. Is this a valid assumption? To answer this question it is necessary to observe the environment for which the manipulators were designed.

Serial manipulators have proven to be effective for a wide variety of production operations where the task of the manipulator is preplanned and the workspace is relatively free of obstacles. Many of the tasks performed by industrial manipulators require at most six degrees of freedom. For these applications, a highly dexterous manipulator is unnecessary, and only serves

to increase the mechanical and computational complexity of the system.

The success associated with introducing robotics in large scale manufacturing soon created an interest in applying the technology in other hazardous environments. The proposed space station, fighting fires, and nuclear reactor maintenance are just a few examples. In these somewhat unstructured environments filled with obstacles, dexterity becomes very important. This has sparked interest in designing and controlling serial manipulators that have seven or eight degrees of freedom [2]. The approach typically taken is to find a solution which optimizes some structural objective function. Since the devices investigated were formed by serially connected links, there was no need to address manipulators of significantly higher degrees-of-freedom. Long chain serial devices simply cannot support their own weight or have such low payload capacities that they are not practical. VGT manipulators offer the first real hope of constructing highly dexterous long chain manipulators.

Solutions based solely on optimization methods work quite well for devices with seven or eight degrees of freedom, but they become cumbersome when applied to a sixty degree-of-freedom system. It simply takes too much time to conduct an exhaustive optimal search with this large number of free parameters. Because of the lack of adequate controller algorithms for very high degree-of-freedom devices, the field has remained virtually uninvestigated both from structural and computational aspects. This the-

sis will focus on the computational aspects of controlling long-chain VGT manipulators.

Many geometric configurations, both planar and spatial, are possible candidates for VGT manipulators. Two such geometries will be presented in great detail in subsequent chapters. First, the basic units, or cells, of these two geometries will be analyzed. Computational methods will then be developed which will utilize these fundamental elements in a repeating chain to construct very high degree-of-freedom "snake-like" manipulators.

Algorithms will then be developed for the control of these devices based on an assumed general shape characteristic. It will be shown that this greatly reduces the complexity of finding an acceptable solution.

Chapter 2

Review of Literature

The material presented within this thesis relies heavily on the fundamentals of kinematics and geometric modeling. This literature review will not contain references to texts which contain these fundamental concepts. Instead, when appropriate, these texts will be referenced within the the body of the thesis. This literature review will present publications which represent the past and current research on parallel manipulators. To facilitate the organization of this material, the literature review will be divided into two sections; kinematics of parallel manipulators and variable geometry truss research.

2.1 Kinematics of Parallel Manipulators

Much of the early research conducted in robotics centered around the use of serially-connected manipulators. No significant attention was given parallel manipulators until 1965, when Stewart [3] proposed the idea of a platform

type manipulator. This initial proposal showed that very rigid six degree-of-freedom manipulators could be developed, but unfortunately, they had very restricted workspaces. Other than their practical implementation as flight simulators, these devices remained somewhat of a novelty for many years. Perhaps one reason for this was the lack of adequate computing facilities to quickly solve the iterative kinematic constraint equations. The next decade witnessed tremendous advances in the field of robotics, but practically all of the research was concerned solely with serial manipulators.

Further advances in parallel manipulators were put forth by Tesar and Cox [4] in 1981, Hunt [5] in 1983, Yang and Lee [6] in 1984, and Fichter [7] in 1985. Tesar and Cox performed the kinematic and dynamic analysis of a three degree-of-freedom parallel manipulator. Hunt was concerned with the general analysis techniques for approaching the kinematics of parallel devices. Yang and Lee conducted a feasibility study on several geometrical variations of Stewart platforms. Fichter concentrated on the analysis and design of a Stewart platform based manipulator.

Research concerned with the dynamic analysis of parallel manipulators has been conducted by Sklar and Tesar [8] and Lee and Chao [9]. The work by Sklar and Tesar was concerned with the dynamic analysis techniques for hybrid serial/parallel manipulators. This included several useful geometries for industrial manipulators. Lee and Chao focused on the kinematic and dynamic analysis of one particular geometry of spatial three degree-

of-freedom manipulator. Sugimoto [10,11] has developed computational methods for the dynamic analysis of many types of parallel manipulators.

The papers cited above are all concerned with the analysis of parallel manipulators. Although, none of these papers address the concept of a truss manipulator, many of the analysis techniques presented could be applied to Variable Geometry Trusses.

2.2 Variable Geometry Truss Research

There are many applications in space which require very large, stiff structures. A natural choice for such structures is a static truss. However, because of transport problems, these large structures must either be assembled in space or be transported in a compact form for later automatic deployment. Typically these deployable structures become static once locked into their extended position. The National Aeronautics and Space Administration (NASA) has been actively developing different deployable structure geometries, including one investigated by Dorsey [12], which is very similar to the planar VGT described later in this thesis.

While investigating other possible geometries for these deployable structures, Rhodes and Mikulas [13] discovered that one certain geometry of deployable truss (an octahedral/octahedral) had properties that made it a suitable three degree-of-freedom spatial manipulator.

Rhodes and Mikulas in conjunction with Sincarsin [14] developed a

working proof-of-concept model. This model not only solved many of the complex joint geometry problems, but also successfully demonstrated the potential usefulness of these structures. This insight made possible the work of Miura, Furuya and Suzuki [15,16], who analyzed the kinematics of a one cell octahedral/octahedral VGT. Simultaneous with the development of the octahedral/octahedral truss model, Sincarsin and Hughes [17] also explored the characteristics of four other candidate geometries. Their evaluations concluded that the octahedral/octahedral was the most favorable geometry. Of primary interest in this study was the issue of collapsibility. Therefore, it should not be assumed that this is the best geometry for all applications. Jain and Kramer [18] also investigated another possible geometry and completed the design of a tetrahedral/tetrahedral VGT.

Other research concerning the use of a VGT cell as a replacement for more conventional devices has been conducted by Nayfeh [19], Padmanabhan [20] and Clark [21]. Nayfeh investigated the kinematics of a foldable space crane composed of many VGT cells. The analysis undertaken was limited to only one of the proposed cells. Padmanabhan studied the kinematics of a VGT jointed planar four degree-of-freedom manipulator. Here, the VGT joints were substituted for simple revolute joints. Clark investigated the use of these VGT modules for actively damping vibration in large truss structures. This study dramatically illustrates the superiority of VGT actuators over conventional proof-mass type actuators for vibra-

tion control. Natori, Iwasaki and Kuwao [22] have also investigated the vibration characteristics of long-chain planar trusses.

Although Miura and Furuya and others have analyzed the kinematics for one celled manipulators, there has been virtually no research on the position control of high degree-of-freedom, long-chain VGT trusses as presented in this thesis. The concept of using curves or surfaces to determine the shape of a structure was briefly addressed by Natori, Iwasaki and Kuwao. In their research, shape control concepts were utilized to yield an "adaptive planar truss structure" capable of forming a variety of parabolic shapes for large space antennae.

Chapter 3

Kinematic Analysis of the Basic Units

To obtain a complete understanding of the complex geometries associated with the long-chain VGT's, it is first necessary to gain a familiarity with the fundamental units used to construct such structures. The science of kinematics provides a methodical way of analyzing the position, velocity, acceleration, and all higher-order derivatives of motion of these complex truss configurations. One definition of kinematics, which is appropriate to the study of VGT's, is as follows; *kinematics is the study of constrained motion of interconnected rigid links.* In this definition the term *"rigid"* is used to describe non-elastic behavior of the links and does not preclude the use of intentionally extensible links.

From a robotics viewpoint there are two distinct subcategories of kinematics, namely, forward kinematic analysis and inverse kinematic analysis. Forward kinematic analysis, sometimes referred to as direct kinematic

analysis, is concerned with finding the position and orientation of any or all members of a device given only the geometric constraints of individual links, the order in which these links are assembled, and the set of control variables (either joint angles or extensible link lengths). The forward kinematic analysis could also be conducted to find the linear and angular velocities or accelerations of any member.

The inverse kinematic analysis is concerned with finding the set of control variables which yields a desired position and orientation of a set of members for a given device. In most useful devices it is not possible to control the position and orientation of all members simultaneously. Just as with the forward kinematic analysis, the inverse case could be solved for any higher order derivative of motion.

In the following sections these concepts will be further developed for both a three degree-of-freedom planar VGT and a six degree-of-freedom spatial VGT.

3.1 Three Degree-of-Freedom Planar VGT

The three degree-of-freedom planar VGT to be discussed is illustrated in Fig. 3.1. It consists of four members and a ground link, all of which are

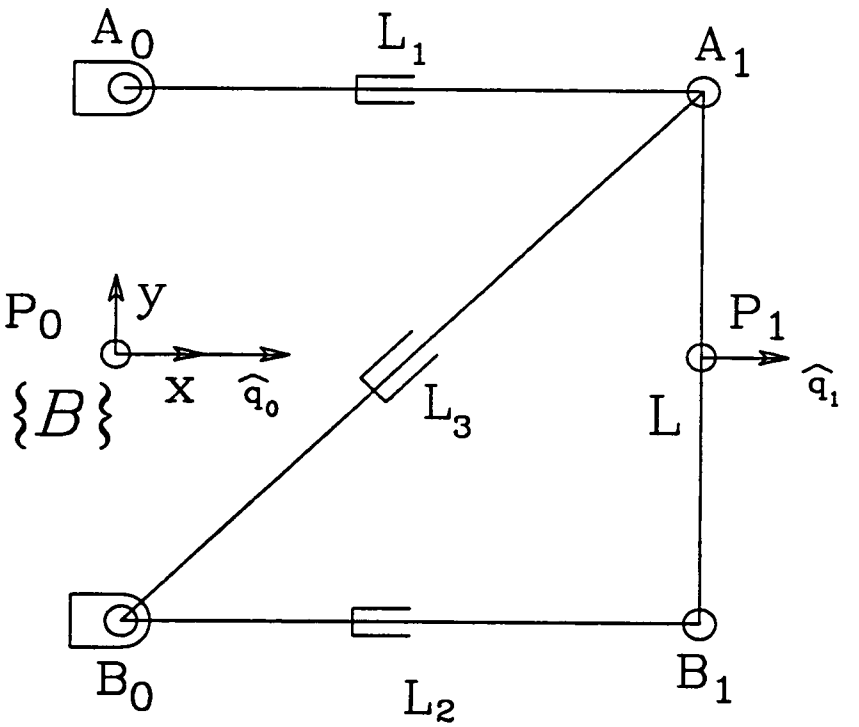

Figure 3.1: Three Degree-of-Freedom Planar VGT

joined by revolute joints. Three of the truss members, A_0A_1, B_0B_1, and B_0A_1, are variable length links and thus can be used to precisely control the position and orientation of the static length member, A_1B_1, relative to the static length ground member, A_0B_0. The lengths of the three extensible members, A_0A_1, B_0B_1, and B_0A_1, will be denoted as L_1, L_2, and L_3 respectively. In conventional kinematic notation, these extensible links would be viewed as a pair of links connected by prismatic joints. Either viewpoint is correct, but the concept of extensible links is more convenient in the present context. The length of the two fixed length members will be denoted by the constant L. This VGT is referred to as a three degree-of-freedom device because it possess a set of three independent control variables, L_1, L_2, L_3, which can be used to control any three independent parameters of the truss. For this analysis the three independent parameters chosen are the x and y displacements of point P_1 relative to P_0 and the angle of the unit vector $\hat{q}_1$ referenced to the vector $\hat{q}_0$. In general, the point P_i is defined as the midpoint of A_iB_i and the unit vector $\hat{q}_i$ is perpendicular to A_iB_i. Since it is desirable to find P_1 and $\hat{q}_1$ relative to P_0 and $\hat{q}_0$, it is convenient to assign a base coordinate frame , $\mathcal{B}$, at P_0 whose X axis is coincident with $\hat{q}_0$. All vectors in the following discussion will be defined with respect to this coordinate frame. Thus, the point P_1 is now located by the vector $\vec{P}_1$, and the unit vector $\hat{q}_1$ is completely described by the angle θ which lies between $\hat{q}_0$ and $\hat{q}_1$ measured in a right hand sense about the z axis.

With the notation defined above, the forward and inverse kinematics can proceed with minimum confusion.

3.1.1 Position, Forward Kinematic Analysis

Restated simply, the forward kinematic problem for position analysis is:

Given a set of control variables, $\{L_1, L_2, L_3\}$*, find the position,* P_1*, and orientation,* θ*, of the member* A_1B_1 *relative to coordinate frame* $\mathcal{B}$*.*

Figure 3.2 depicts the planar truss in a typical extended position with all of the variables needed for the following discussion illustrated. By examining the triangle $\Delta A_0A_1B_0$ and utilizing the law of cosines it is possible to solve for the angle ψ_1 as follows:

$$L_3^2 = L_1^2 + L^2 - 2L_1L(\cos\psi_1);$$

therefore,

$$\psi_1 = \arccos\left(\frac{L^2 + L_1^2 - L_3^2}{2LL_1}\right).$$

Operating again on $\Delta A_0A_1B_0$ with the law of cosines, the angle ψ_3 can

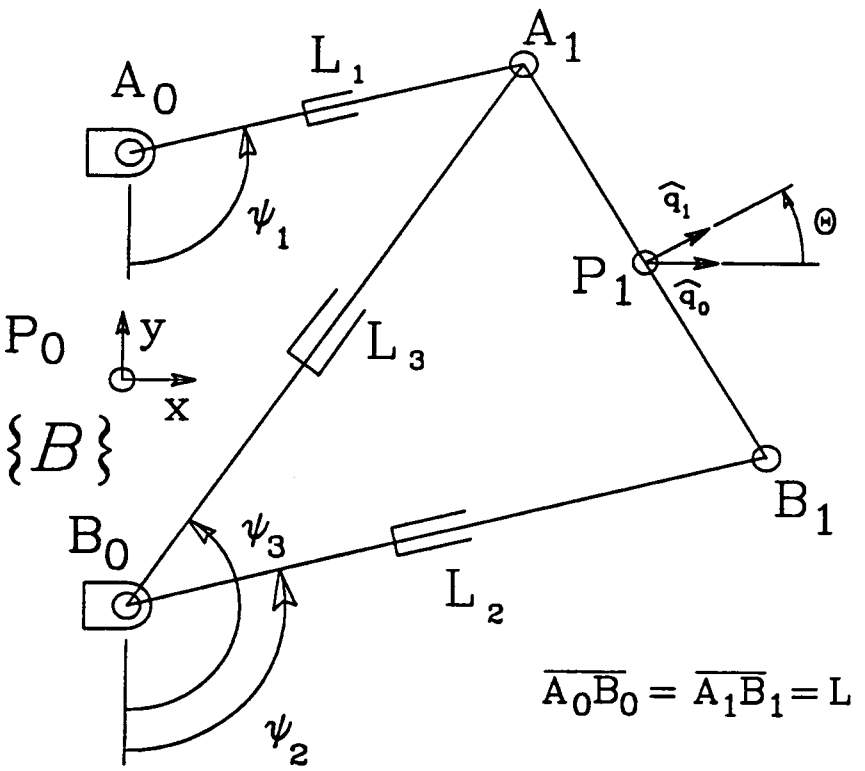

Figure 3.2: Displaced Three Degree-of-Freedom Planar VGT

also be found.

$$L_1^2 = L^2 + L_3^2 + 2LL_3(\cos\psi_3);$$

therefore,

$$\psi_3 = \arccos\left(\frac{-L^2 + L_1^2 - L_3^2}{2LL_3}\right).$$

A similar approach on $\triangle B_0A_1B_1$ yields the following:

$$L^2 = L_2^2 + L_3^2 - 2L_2L_3(\cos(\psi_3 - \psi_2));$$

therefore,

$$\psi_3 - \psi_2 = \arccos\left(\frac{-L^2 + L_2^2 + L_3^2}{2L_2L_3}\right).$$

Substituting the expression for ψ_3 into the above equation yields

$$\psi_2 = \arccos\left(\frac{-L^2 + L_1^2 - L_3^2}{2LL_3}\right) - \arccos\left(\frac{-L^2 + L_2^2 + L_3^2}{2L_2L_3}\right).$$

Since points A_0 and B_0 are immobile with respect to the coordinate frame $\mathcal{B}$, their position vectors $\vec{A}_0$ and $\vec{B}_0$ can be defined for all time as:

$$\vec{A}_0 = \begin{bmatrix} 0 \\ L/2 \end{bmatrix}, \qquad \vec{B}_0 = \begin{bmatrix} 0 \\ -L/2 \end{bmatrix}.$$

By simple vector addition $\vec{A}_1$ and $\vec{B}_1$ can now be solved for directly.

$$\vec{A}_1 = \vec{A}_0 + L_1 \begin{bmatrix} \sin\psi_1 \\ -\cos\psi_1 \end{bmatrix},$$

$$\vec{B}_1 = \vec{B}_0 + L_2 \begin{bmatrix} \sin\psi_2 \\ -\cos\psi_2 \end{bmatrix}.$$

Two of the three independent parameters are now specified by,

$$\vec{P}_1 = \frac{(\vec{A}_1 + \vec{B}_1)}{2}.$$

The final parameter to be specified is the angle θ. Referring to Fig. 3.3, the quantities s and t can be defined as,

$$\begin{bmatrix} -s \\ t \end{bmatrix} = \vec{A}_1 - \vec{B}_1.$$

Now θ can be defined as,

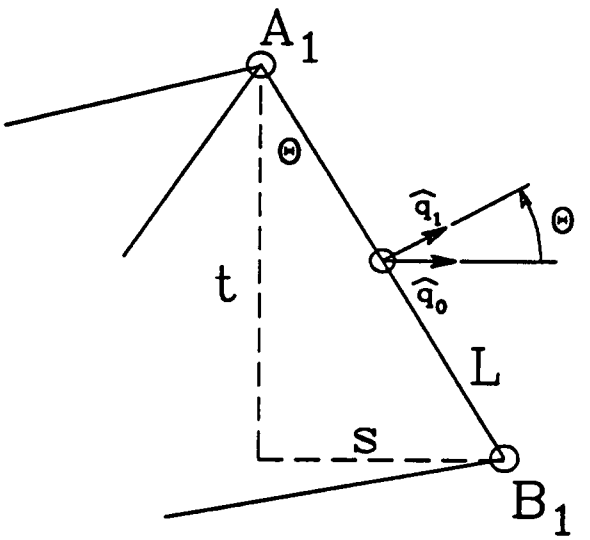

Figure 3.3: Parameters Needed to Define θ

$$\theta = \text{ATAN2}(s,t) \qquad t \neq 0;$$

or,

$$\theta = \arcsin(s/L) \qquad t = 0;$$

where ATAN2 is the four quadrant arc tangent function which uniquely defines θ from 0 to 360 degrees.

The results of the forward kinematic solution just presented are non-linear and hence no one unique solution exists. In fact, for this particular truss, four distinct assemblies, or branches, exist. All four of these solutions for a given set of $\{L, L_1, L_2, L_3\}$ are presented in Fig. 3.4. These solutions represent mathematically acceptable solutions which the truss could physically obtain, but only after unpinning the joints and reassembling the truss within this new branch. Obviously, from an operational standpoint, all of these solutions are not practical. To result in a practical solution, this mathematical solution technique must somehow be tailored to omit all but the desired solution. This is done simply by limiting the range of ψ_1 to 0–180 degrees, and imposing the constraint, $\psi_3 > \psi_2$. Note that this results in branch 1 of Fig. 3.4. By modifying the constraints, the solution could

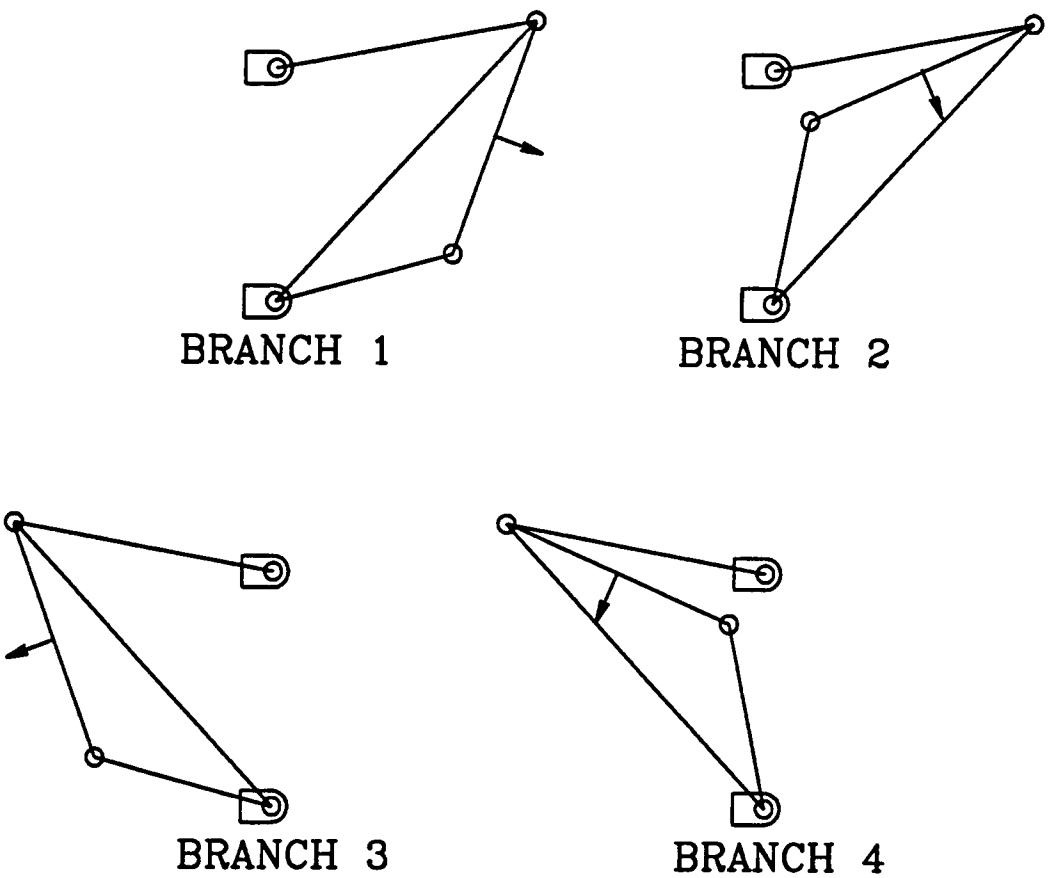

Figure 3.4: Four Different Assemblies of Planar VGT

just as easily lead to a result in any other branch.

Both of the trusses discussed in this thesis were designed such that it is impossible, during operation, to pass from one branch into another. This is not an absolute design requirement; but, as will be shown later, it is certainly advantageous.

Although the forward kinematic solution does not appear to be useful for robotic applications, it is in fact crucial to many calculations. For example, it is the ideal tool for conducting research on manipulator workspaces. By merely incrementing all of the variable length members through their entire range, a map of the reachable workspace can be constructed. Other applications of the forward kinematic solution include structural analyses, and sensitivity analyses - where for example the effect on θ caused by a variation in L_1 could be found. This is essentially the partial derivative of θ with respect to $L_1, (\partial\theta/\partial L_1)$. This information could in turn be used in an approximate velocity analysis of the device about a fixed point. This concept will be developed in greater depth in section 3.1.3.

3.1.2 Position, Inverse Kinematic Analysis

The inverse kinematic problem for position analysis can simply be stated as;

Given $\vec{P}_1$ and θ, solve for the set of control variables, $\{L_1, L_2, L_3\}$.

The inverse kinematic analysis of this truss is relatively simple. If the vector $\vec{P}_1$ and angle θ are known, vectors $\vec{A}_1$ and $\vec{B}_1$ can be solved as follows:

$$\vec{A}_1 = \vec{P} + \frac{L}{2}\begin{bmatrix} -\sin\theta \\ \cos\theta \end{bmatrix},$$

$$\vec{B}_1 = \vec{P} - \frac{L}{2}\begin{bmatrix} -\sin\theta \\ \cos\theta \end{bmatrix}.$$

Since these descriptions are referenced to coordinate frame $\mathcal{B}$, the definitions of $\vec{A}_0$ and $\vec{B}_0$ previously set forth are still valid. With this, a solution for L_1, L_2, L_3 can immediately be found as follows:

$$\begin{aligned} L_1 &= \|\vec{A}_1 - \vec{A}_0\| = \sqrt{(\vec{A}_1 - \vec{A}_0)\cdot(\vec{A}_1 - \vec{A}_0)}, \\ L_2 &= \|\vec{B}_1 - \vec{B}_0\| = \sqrt{(\vec{B}_1 - \vec{B}_0)\cdot(\vec{B}_1 - \vec{B}_0)}, \\ L_3 &= \|\vec{A}_1 - \vec{B}_0\| = \sqrt{(\vec{A}_1 - \vec{B}_0)\cdot(\vec{A}_1 - \vec{B}_0)}. \end{aligned}$$

Note that this inverse kinematic solution is not only closed form, but

also linear, and thus only one solution exists. This means that with the absence of any other constraints, a unique solution for any given $\vec{P}_1$ and θ always exists. In reality, of course, other constraints do exist. One of the primary constraints for this device is the range of its variable length members. For all of the truss structures presented the ratio of the maximum possible link extension to the minimum possible link extension was chosen to be 1.85. This allows for the extensible links to be self contained and collapse within themselves. Because of this limitation on range, not all inverse kinematic solutions are acceptable. A solution which violates any of the constraints simply means that the specified $\vec{P}_1$ and θ are outside of the manipulator's workspace.

Clearly, for most tasks requiring a robotic manipulator the inverse kinematic solution is the approach to be utilized. Here a desired position and orientation is specified for an end-effector, presumably mounted at P_1. The inverse kinematic algorithm then calculates the necessary control variables $\{L_1, L_2, L_3\}$ and outputs these values to a controller, which in turn servos the individual variable length members.

As with the forward kinematics analysis, the inverse analysis could also be utilized in a sensitivity analysis where, for example, the effect on L_1 caused by a variation in θ could be found.

3.1.3 Other Considerations

Alternative Specification of Goals

In the preceding analysis the three independent parameters needed to specify the truss were chosen as $\vec{P}_1$ and θ. In fact, any three independent parameters could have been chosen. For example, the three angles ψ_1, ψ_2, and ψ_3, or their compliments, are equally valid choices for specifying the truss position. This specification is, however, harder to visualize and is not useful in solving robotic tasks.

Higher-Order Analysis

A forward and inverse kinematic analysis for position has now been accomplished. As mentioned previously, a kinematic analysis could have been conducted for any higher order derivative of motion. This section will briefly present a velocity kinematic analysis of the described planar truss based on a formulation presented in [24, pages 144–149].

With the information that resulted from the forward kinematic solution, it is apparent that

$$\begin{aligned} P_x &= f_1(L_1, L_2, L_3), \\ P_y &= f_2(L_1, L_2, L_3), \\ \theta &= f_3(L_1, L_2, L_3). \end{aligned}$$

This can be more compactly expressed in matrix form with G representing the goal position, $[P_x, P_y, \theta]^T$.

$$G = F(L) .$$

Now a Jacobian can be formulated as follows:

$$J(L) = \begin{bmatrix} \frac{\partial f_1}{\partial L_1} & \frac{\partial f_1}{\partial L_2} & \frac{\partial f_1}{\partial L_3} \\ \frac{\partial f_2}{\partial L_1} & \frac{\partial f_2}{\partial L_2} & \frac{\partial f_2}{\partial L_3} \\ \frac{\partial f_3}{\partial L_1} & \frac{\partial f_3}{\partial L_2} & \frac{\partial f_3}{\partial L_3} \end{bmatrix} .$$

A forward velocity analysis can now be conducted about the goal position G as follows:

$$\dot{G} = J(L) \ \dot{L} .$$

The inverse velocity analysis can be computed in a similar manner as,

$$\dot{L} = J^{-1}(L) \ \dot{G} ;$$

where $|J(L)| \neq 0$, in which case $J(L)$ is singular and thus non-invertible. With this information available it is possible to calculate the required joint velocities to produce a given end-effector velocity.

3.2 Six Degree-of-Freedom Spatial VGT

The six degree-of-freedom VGT to be discussed is illustrated in Fig. 3.5. This truss consists of six static length members and six variable length members which are joined together at six nodes $(n_1, n_2, \ldots, n_6)$ with spheric joints. The fixed length members are joined together in sets of three to form two triangles. One of these triangles is used to define the base plane and has coordinate frame $\mathcal{B}$ at its centroid. The other triangle will form the top plane with frame $\mathcal{T}$ at its centroid. The length of the extensible members will be denoted by the control variables $L_1, L_2, \ldots, L_6$. In this analysis, all fixed length members are assumed to be of length L. This assumption is a property of the selected geometry; it is not essential that all fixed length members have the same length to obtain a solution. By assigning different values to the six control variables, the position and orientation of frame $\mathcal{T}$ relative to frame $\mathcal{B}$ can be altered. This relative position and orientation is described with six parameters. Three will be reserved for the x, y, and z position of frame $\mathcal{T}$ in frame $\mathcal{B}$ as described by the vector ${}^{\mathcal{B}}\vec{P}_1$. The three remaining parameters will describe the orientation of $\mathcal{T}$ relative to $\mathcal{B}$. This is done by utilizing the standard roll-pitch-yaw notation; where roll is a rotation about the X-axis of frame $\mathcal{B}$ by γ, pitch is a rotation about the Y-axis of frame $\mathcal{B}$ by β, and yaw is a rotation about the Z-axis of frame $\mathcal{B}$ by α. This rotation of frame $\mathcal{T}$ relative to $\mathcal{B}$ can be compactly expressed as ${}^{\mathcal{B}}_{\mathcal{T}}R_{\gamma\beta\alpha}$, a 3×3 matrix possessing three independent variables.

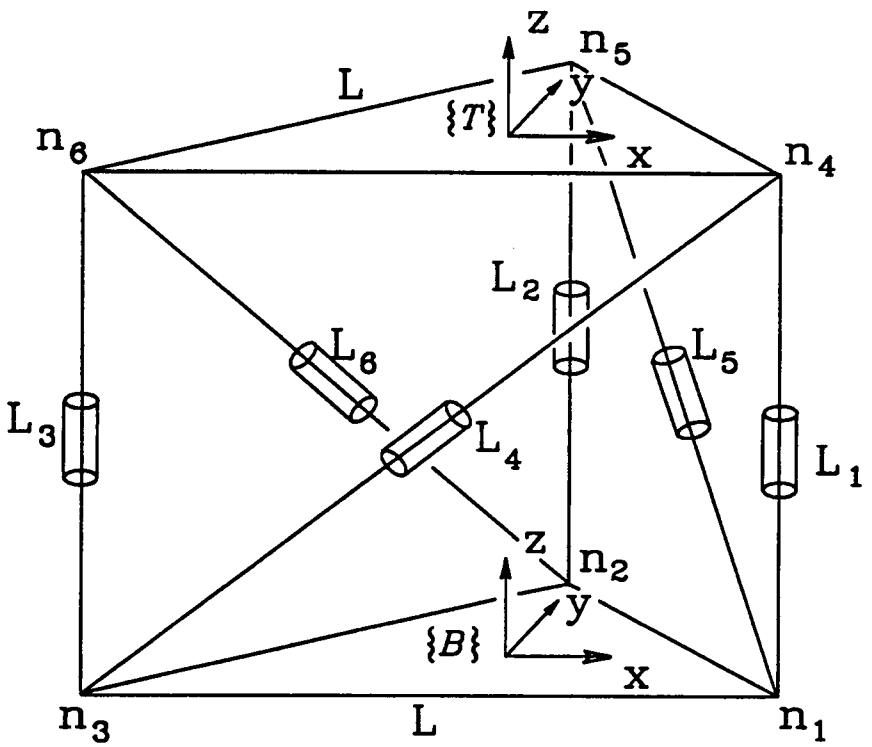

Figure 3.5: Six Degree-of-Freedom Spatial VGT

To simplify notation, vectors expressed in a frame $\mathcal{F}$ will be identified by a pre-superscript $\mathcal{F}$; for example, ${}^{\mathcal{F}}\vec{V}$. Also, all position vectors will be 4×1 matrices with the last entry being unity.

The following sections will present solutions to the forward and inverse kinematics of this manipulator.

3.2.1 Position Forward Kinematic Analysis

The forward kinematic analysis problem for the six degree-of-freedom cell can be posed in short as follows:

Given the lengths of the extensible members, $L_1, L_2, \ldots, L_6$, solve for the position and orientation of frame $\mathcal{T}$ relative to frame $\mathcal{B}$.

In general, for this configuration of manipulator, this problem can not be solved in closed form [23]. Instead an iterative procedure has been constructed which converges in a small number of iterations to an approximate solution. This solution is however, more accurate than can be physically measured or controlled, and hence is completely acceptable.

To assist in the forward kinematic analysis it is helpful to employ the concept of kinematically equivalent devices. Two devices are said to be kinematically equivalent with respect to a point if both devices exhibit

exactly the same motion characteristics for the point of interest. Note that the two devices need not be structurally equivalent. This concept will greatly simplify the mathematics involved and will aid tremendously in the visualization of the forward kinematic analysis.

Figure 3.6 depicts the kinematically equivalent device which will be substituted for the side triangle $\Delta n_3 n_1 n_4$. In this case, two spheric joints and two prismatic joints (Fig. 3.6-a) have been replaced by one cylindric joint and one prismatic joint (Fig. 3.6-b). The cylindric joint is positioned along its axis, $\hat{u}_1$, at the point b_1. An explicit expression for $\vec{b}_1$ can be found as follows:

$$\psi_1 = \arccos\left(\frac{L^2 + L_4^2 - L_1^2}{2LL_4}\right),$$

therefore,

$$\vec{b}_1 = \vec{n}_3 + L_4 \cos\psi_1 \hat{u}_1.$$

Now the distance h_1 can be defined. h_1 is simply the altitude of triangle $\Delta n_3 n_1 n_4$ as shown in Fig. 3.6-a. This is equivalent to the extension of the prismatic member which is shown in Fig. 3.6-b. h_1 is given by,

$$h_1 = L_4 \sin\psi_1.$$

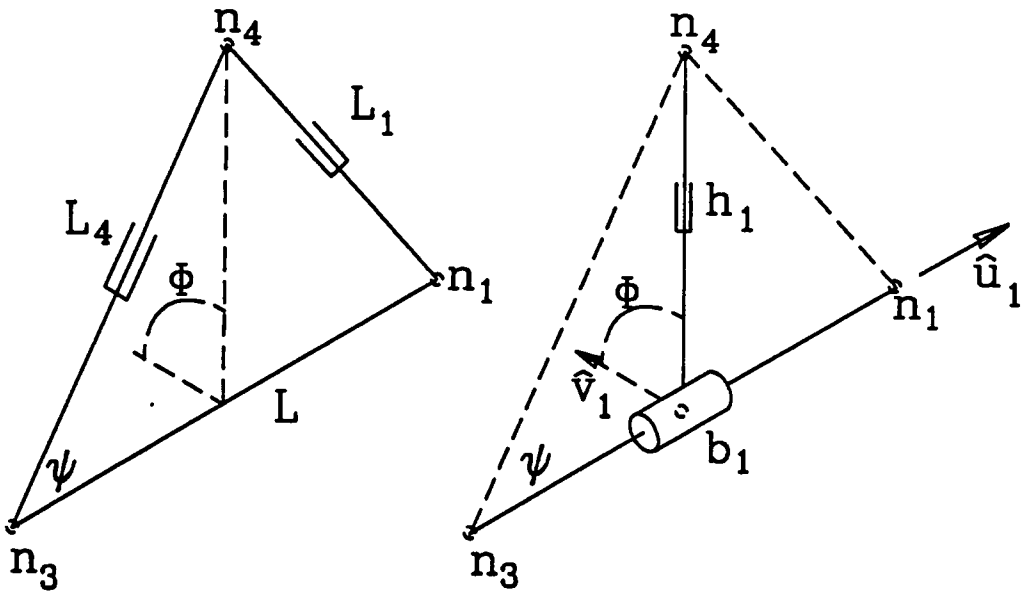

Figure 3.6: Kinematically Equivalent Devices Used for Analysis

The unit vector $\hat{v}_1$ is defined to be normal to $\hat{u}_1$ and to lie in the plane of $\Delta n_1 n_2 n_3$. Now the point n_4 can be located as follows:

$$\vec{n}_4 = \vec{b}_1 + h_1 R_{[\hat{u}_1, \phi_1]} \hat{v}_1;$$

where $R_{[\hat{u}_1, \phi_1]} \hat{v}_1$ represents a rotation of $\hat{v}_1$ about $\hat{u}_1$ an amount ϕ_1. Examination of the four previous equations reveals that $\vec{n}_4$ is a function dependent only on L, L_1, L_4, and ϕ_1. However, for the forward kinematics problem $L, L_1, L_2, \ldots, L_6$ are known. Therefore, the the vector $\vec{n}_4$ is only dependent on the unknown ϕ_1. Similar results can be obtained for the other two side triangles. These results are now summarized below:

$$\begin{aligned}
\vec{n}_4 &= \vec{n}_3 + L_4 \cos\psi_1 \hat{u}_1 + L_4 \sin\psi_1 R_{[\hat{u}_1, \phi_1]} \hat{v}_1, \\
\vec{n}_5 &= \vec{n}_1 + L_5 \cos\psi_2 \hat{u}_2 + L_5 \sin\psi_2 R_{[\hat{u}_2, \phi_2]} \hat{v}_2, \\
\vec{n}_6 &= \vec{n}_2 + L_6 \cos\psi_3 \hat{u}_3 + L_6 \sin\psi_3 R_{[\hat{u}_3, \phi_3]} \hat{v}_3;
\end{aligned}$$

where,

$$\psi_2 = \arccos\left(\frac{L^2 + L_5^2 - L_2^2}{2LL_5}\right),$$

$$\psi_3 = \arccos\left(\frac{L^2 + L_6^2 - L_3^2}{2LL_6}\right).$$

The distance between nodes of the top triangle can now be found as:

$$D_1 = \|\vec{n}_4 - \vec{n}_5\| = \sqrt{(\vec{n}_4 - \vec{n}_5) \cdot (\vec{n}_4 - \vec{n}_5)},$$

$$D_2 = \|\vec{n}_5 - \vec{n}_6\| = \sqrt{(\vec{n}_5 - \vec{n}_6) \cdot (\vec{n}_5 - \vec{n}_6)},$$

$$D_3 = \|\vec{n}_6 - \vec{n}_4\| = \sqrt{(\vec{n}_6 - \vec{n}_4) \cdot (\vec{n}_6 - \vec{n}_4)}.$$

It is now necessary to check for closure. In order to have a valid assembly, the following three equations must be satisfied:

$$f_1(\phi_1, \phi_2) = D_1 - L = 0,$$

$$f_2(\phi_2, \phi_3) = D_2 - L = 0,$$

$$f_3(\phi_3, \phi_1) = D_3 - L = 0.$$

This can be done simply by applying a Newton-Raphson root-finding algorithm which centers around the following linearization:

$$\begin{bmatrix} \frac{\partial f_1}{\partial \phi_1} & \frac{\partial f_1}{\partial \phi_2} & \frac{\partial f_1}{\partial \phi_3} \\ \frac{\partial f_2}{\partial \phi_1} & \frac{\partial f_2}{\partial \phi_2} & \frac{\partial f_2}{\partial \phi_3} \\ \frac{\partial f_3}{\partial \phi_1} & \frac{\partial f_3}{\partial \phi_2} & \frac{\partial f_3}{\partial \phi_3} \end{bmatrix} \begin{bmatrix} \Delta\phi_1 \\ \Delta\phi_2 \\ \Delta\phi_3 \end{bmatrix} = \begin{bmatrix} -f_1 \\ -f_2 \\ -f_3 \end{bmatrix}.$$

For any given iteration, $\Delta\phi_1, \Delta\phi_2$, and $\Delta\phi_3$ can be found, and the current " guessed " position can be modified by:

$$\begin{aligned}
\phi_{1_{new}} &= \phi_{1_{old}} + \Delta\phi_1, \\
\phi_{2_{new}} &= \phi_{2_{old}} + \Delta\phi_2, \\
\phi_{3_{new}} &= \phi_{3_{old}} + \Delta\phi_3.
\end{aligned}$$

In typical operation this procedure converges within an acceptable range after only four iterations. Other more efficient non-linear equation solving routines could also be implemented. Upon convergence of the routine outlined above, the position of nodes n_4, n_5, and n_6 are known. The origin of frame $\mathcal{T}$ can now be described in frame $\mathcal{B}$ as the centroid of these three nodes.

$$^{\mathcal{B}}\vec{P}_1 = \frac{[\vec{n}_4 + \vec{n}_5 + \vec{n}_6]}{3}.$$

The three unit vectors which form the coordinate axes of frame $\mathcal{T}$ can be described in frame $\mathcal{B}$ as follows:

$$^{\mathcal{B}}\hat{X}_{\mathcal{T}} = \frac{[\vec{n}_4 - \vec{n}_6]}{||\vec{n}_4 - \vec{n}_6||},$$

$$^{\mathcal{B}}\hat{Z}_{\mathcal{T}} = \frac{(\vec{n}_4 - \vec{n}_6) \times (\vec{n}_5 - \vec{n}_6)}{\| (\vec{n}_4 - \vec{n}_6) \times (\vec{n}_5 - \vec{n}_6) \|},$$

$$^{\mathcal{B}}\hat{Y}_{\mathcal{T}} = ^{\mathcal{B}}\hat{Z}_{\mathcal{T}} \times ^{\mathcal{B}}\hat{X}_{\mathcal{T}}.$$

Note that each of the above expressions are vector quantities and as such each has three components. Now the rotation matrix which describes the orientation of frame $\mathcal{T}$ with respect to frame $\mathcal{B}$ can be described as follows:

$$^{\mathcal{B}}_{\mathcal{T}}R_{\gamma,\beta,\alpha} = \begin{bmatrix} ^{\mathcal{B}}\hat{X}_{\mathcal{T}x} & ^{\mathcal{B}}\hat{Y}_{\mathcal{T}x} & ^{\mathcal{B}}\hat{Z}_{\mathcal{T}x} \\ ^{\mathcal{B}}\hat{X}_{\mathcal{T}y} & ^{\mathcal{B}}\hat{Y}_{\mathcal{T}y} & ^{\mathcal{B}}\hat{Z}_{\mathcal{T}y} \\ ^{\mathcal{B}}\hat{X}_{\mathcal{T}z} & ^{\mathcal{B}}\hat{Y}_{\mathcal{T}z} & ^{\mathcal{B}}\hat{Z}_{\mathcal{T}z} \end{bmatrix};$$

where each column of the matrix contains the x, y, and z components of each coordinate axis that composes frame $\mathcal{T}$. This expression can now be equated to the standard roll-pitch-yaw representation of the rotation matrix as presented below:

$$^{\mathcal{B}}_{\mathcal{T}}R_{\gamma,\beta,\alpha} = \begin{bmatrix} c\alpha c\beta & c\alpha s\beta s\gamma - s\alpha c\gamma & c\alpha s\beta c\gamma + s\alpha s\gamma \\ s\alpha c\beta & s\alpha s\beta s\gamma + c\alpha c\gamma & s\alpha s\beta c\gamma - c\alpha s\gamma \\ -s\beta & c\beta s\gamma & c\beta c\gamma \end{bmatrix};$$

where $c\beta = \cos\beta, s\beta = \sin\beta$, etc.

The roll-pitch-yaw description can now be obtained uniquely from one of the three following sets of relations [24, pages 41–42].

•Case 1. if $\left\| {}^{\mathcal{B}}\hat{X}_{\mathcal{T}z} \right\| \neq 1$,

$$\begin{aligned} \gamma &= \text{ATAN2}\left({}^{\mathcal{B}}\hat{Y}_{\mathcal{T}z}, {}^{\mathcal{B}}\hat{Z}_{\mathcal{T}z}\right) \\ \beta &= \text{ATAN2}\left(-{}^{\mathcal{B}}\hat{X}_{\mathcal{T}z}, \sqrt{{}^{\mathcal{B}}\hat{X}_{\mathcal{T}x}^2 + {}^{\mathcal{B}}\hat{X}_{\mathcal{T}y}^2}\right) \\ \alpha &= \text{ATAN2}\left({}^{\mathcal{B}}\hat{X}_{\mathcal{T}y}, {}^{\mathcal{B}}\hat{X}_{\mathcal{T}x}\right) \end{aligned}$$

•Case 2. if ${}^{\mathcal{B}}\hat{X}_{\mathcal{T}z} = +1$,

$$\begin{aligned} \gamma &= \text{ATAN2}\left({}^{\mathcal{B}}\hat{Y}_{\mathcal{T}x}, {}^{\mathcal{B}}\hat{Y}_{\mathcal{T}y}\right) \\ \beta &= 90^o \\ \alpha &= 0 \end{aligned}$$

•Case 3. if ${}^{\mathcal{B}}\hat{X}_{\mathcal{T}z} = -1$,

$$\begin{aligned} \gamma &= -\text{ATAN2}\left({}^{\mathcal{B}}\hat{Y}_{\mathcal{T}x}, {}^{\mathcal{B}}\hat{Y}_{\mathcal{T}y}\right) \\ \beta &= -90^o \\ \alpha &= 0 \end{aligned}$$

Just as with the planar forward kinematics example, the solution to the spatial forward kinematics is non-linear. This indicates that more than one possible closure exists. For this device it is believed that sixteen different branches exist. To ensure that the desired branch is selected, more con-

straints must be imposed. This is done by limiting the range of ϕ_1, ϕ_2, and ϕ_3 in the iterative procedure.

Historically, manipulators in which the kinematics must be solved iteratively have been avoided. This can primarily be attributed to convergence problems, and, to a lesser degree, the speed at which these algorithms can be executed. It is true that for large moves convergence of the proposed algorithms might present a problem. However, at the speed at which modern controllers can update a position, there is no need to move the system a great distance in one step. For small moves the values of ϕ_1, ϕ_2, and ϕ_3 are known relatively accurately to start with. Thus, convergence is not a problem unless singularites are present. Singularity points can be identified by forming a jacobian matrix as described in section 3.1.3 and setting its determinate equal to zero. For this particular device, the singularity points occur outside of the physical workspace, which is limited by the range of $L_1, L_2, \ldots, L_6$.

3.2.2 Position Inverse Kinematic Analysis

The inverse kinematic analysis of the six degree-of-freedom spatial cell is both closed-form and linear. The inverse kinematic analysis problem can

be stated as follows:

Given the position vector, $\vec{P}_1$, *and* α, β, γ, *solve for the set of control variables* $\{L_1, L_2, \ldots, L_6\}$.

Since α, β, and γ are known, the rotation from $\mathcal{B}$ to $\mathcal{T}$ can be described by the rotation matrix

$$ {}^{\mathcal{B}}_{\mathcal{T}}R_{\gamma,\beta,\alpha} = \begin{bmatrix} c\alpha c\beta & c\alpha s\beta s\gamma - s\alpha c\gamma & c\alpha s\beta c\gamma + s\alpha s\gamma \\ s\alpha c\beta & s\alpha s\beta s\gamma + c\alpha c\gamma & s\alpha s\beta c\gamma - c\alpha s\gamma \\ -s\beta & c\beta s\gamma & c\beta c\gamma \end{bmatrix}. $$

A transform can now be defined which encompasses both the rotation and translation of $\mathcal{T}$ relative to $\mathcal{B}$. This transform is defined as:

$$ {}^{\mathcal{B}}_{\mathcal{T}}T = \begin{bmatrix} & [{}^{\mathcal{B}}_{\mathcal{T}}R] & & [{}^{\mathcal{B}}\vec{P}_1] \\ 0 & 0 & 0 & 1 \end{bmatrix}. $$

Nodes n_4, n_5, and n_6, which are known in $\mathcal{T}$, can now be described in $\mathcal{B}$.

$$ \begin{aligned} {}^{\mathcal{B}}\vec{n}_4 &= {}^{\mathcal{B}}_{\mathcal{T}}T^{\mathcal{T}}\vec{n}_4, \\ {}^{\mathcal{B}}\vec{n}_5 &= {}^{\mathcal{B}}_{\mathcal{T}}T^{\mathcal{T}}\vec{n}_5, \\ {}^{\mathcal{B}}\vec{n}_6 &= {}^{\mathcal{B}}_{\mathcal{T}}T^{\mathcal{T}}\vec{n}_6. \end{aligned} $$

Finally, the length of the extensible members can be solved for directly

as the euclidean distance between appropriate nodes.

$$
\begin{aligned}
L_1 &= \left\| {}^{\mathcal{B}}\vec{n}_4 - {}^{\mathcal{B}}\vec{n}_1 \right\| = \sqrt{({}^{\mathcal{B}}\vec{n}_4 - {}^{\mathcal{B}}\vec{n}_1)\cdot({}^{\mathcal{B}}\vec{n}_4 - {}^{\mathcal{B}}\vec{n}_1)}, \\
L_2 &= \left\| {}^{\mathcal{B}}\vec{n}_5 - {}^{\mathcal{B}}\vec{n}_2 \right\| = \sqrt{({}^{\mathcal{B}}\vec{n}_5 - {}^{\mathcal{B}}\vec{n}_2)\cdot({}^{\mathcal{B}}\vec{n}_5 - {}^{\mathcal{B}}\vec{n}_2)}, \\
L_3 &= \left\| {}^{\mathcal{B}}\vec{n}_6 - {}^{\mathcal{B}}\vec{n}_3 \right\| = \sqrt{({}^{\mathcal{B}}\vec{n}_6 - {}^{\mathcal{B}}\vec{n}_3)\cdot({}^{\mathcal{B}}\vec{n}_6 - {}^{\mathcal{B}}\vec{n}_3)}, \\
L_4 &= \left\| {}^{\mathcal{B}}\vec{n}_4 - {}^{\mathcal{B}}\vec{n}_3 \right\| = \sqrt{({}^{\mathcal{B}}\vec{n}_4 - {}^{\mathcal{B}}\vec{n}_3)\cdot({}^{\mathcal{B}}\vec{n}_4 - {}^{\mathcal{B}}\vec{n}_3)}, \\
L_5 &= \left\| {}^{\mathcal{B}}\vec{n}_5 - {}^{\mathcal{B}}\vec{n}_1 \right\| = \sqrt{({}^{\mathcal{B}}\vec{n}_5 - {}^{\mathcal{B}}\vec{n}_1)\cdot({}^{\mathcal{B}}\vec{n}_5 - {}^{\mathcal{B}}\vec{n}_1)}, \\
L_6 &= \left\| {}^{\mathcal{B}}\vec{n}_6 - {}^{\mathcal{B}}\vec{n}_2 \right\| = \sqrt{({}^{\mathcal{B}}\vec{n}_6 - {}^{\mathcal{B}}\vec{n}_2)\cdot({}^{\mathcal{B}}\vec{n}_6 - {}^{\mathcal{B}}\vec{n}_2)}.
\end{aligned}
$$

These values must be checked to ensure that they are within the physical range of the extensible links. If not, the given goal position is outside of the manipulators workspace and cannot be reached.

Just as with the planar truss, there is also the possibility of conducting velocity, acceleration, and higher order analyses of this device. This is accomplished by simply differentiating the displacement constraint equations with respect to time. In all cases this will result in a system of simultaneous linear equations.

Chapter 4

Forward Kinematic Analysis of Long-Chain VGT's

The previous chapter illustrated how to accomplish the forward kinematic analysis of a one cell VGT. This same analysis is possible for any number of cells joined together in a predetermined fashion. The forward kinematic analyses of the basic units provide computational methods for solving for the position and orientation of any cell member with respect to the base frame of that cell. Thus, given a set of control variables for a ten cell truss, it is already possible to construct ten one-cell trusses with all members being referenced to ten independent base frames. This, however, gives no information about the relationship between the individual base frames, which is a function of how the cells are joined together. More importantly, it is not yet possible to describe the position and orientation of every truss member in one global coordinate frame. This single global coordinate system description is imperative if any graphical output is desired or if a finite

element or solids model is to be constructed. The following sections will illustrate the procedures necessary to analyze the forward kinematics of long-chain VGT's. As an example, the forward kinematic analysis of both a ten cell planar truss and a ten cell spatial truss will be presented.

4.1 Ten Cell Planar VGT

By replicating the structure shown in Fig. 3.1 and joining the cells end-to-end, the thirty degree-of-freedom planar truss illustrated in Fig. 4.1 can be constructed. Before undertaking the forward analysis of this device, it is first necessary to modify the notation for the single planar cell to accommodate multiple cells. Figure 4.2 shows the i^{th} cell of a multi-cell truss. The notation is basically the same as outlined previously with the exception of the one additional subscript i used to denote membership in the i^{th} cell. Any point of interest on the truss can be described in one of several coordinate frames. For example, the point A_i is located in coordinate frame $\mathcal{B}_i$ by the vector ${}^{\mathcal{B}_i}\vec{A}_i$, and it is located in the global coordinate frame, $\mathcal{G}$, by vector ${}^{\mathcal{G}}\vec{A}_i$. Similarly the direction of $\vec{q}_i$ can be represented in the local frame by ${}^{\mathcal{B}_i}\theta_i$ or in the global frame by ${}^{\mathcal{G}}\theta_i$.

Note that the base coordinate frame for the $(i+1)^{th}$ cell, $\mathcal{B}_{i+1}$, is located at point P_i with its x-axis coincident with $\vec{q}_i$. This establishes the relationship between $\mathcal{B}_i$ and $\mathcal{B}_{i+1}$, and thus provides enough information to transform coordinate descriptions from $\mathcal{B}_{i+1}$ to $\mathcal{B}_i$.

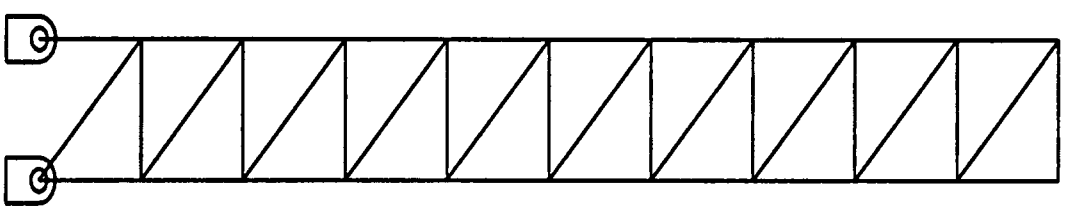

Figure 4.1: Ten Cell, Thirty Degree-of-Freedom Planar Truss

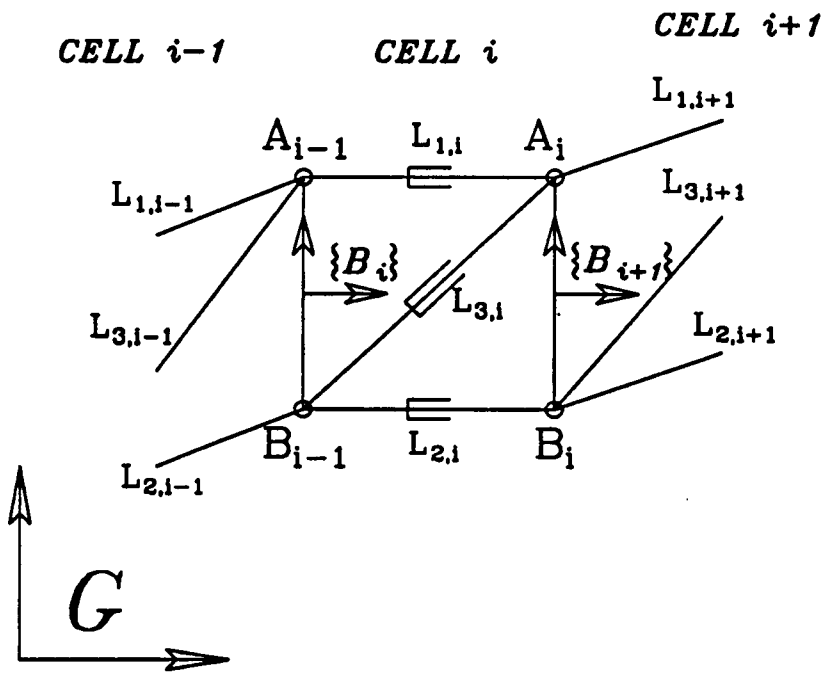

Figure 4.2: i^{th} Cell of a Planar Truss

$$
{}^{\mathcal{B}_i}_{\mathcal{B}_{i+1}}T = \begin{bmatrix} \cos\left({}^{\mathcal{B}_i}\theta_i\right) & -\sin\left({}^{\mathcal{B}_i}\theta_i\right) & {}^{\mathcal{B}_i}P_{xi} \\ \sin\left({}^{\mathcal{B}_i}\theta_i\right) & \cos\left({}^{\mathcal{B}_i}\theta_i\right) & {}^{\mathcal{B}_i}P_{yi} \\ 0 & 0 & 1 \end{bmatrix} ;
$$

where P_{xi} and P_{yi} are the X and Y components of $\vec{P}_i$. The above transform is only a function of ${}^{\mathcal{B}_i}P_{xi}$, ${}^{\mathcal{B}_i}P_{yi}$, and ${}^{\mathcal{B}_i}\theta_i$ which are the results of the forward kinematics for the i^{th} cell. This is intuitively reassuring because the $\mathcal{B}_{i+1}$ frame is rigidly attached to the i^{th} cell.

It shall be assumed that the transformation from the base coordinate frame of the first cell to the global coordinate frame is known. Typically this relationship is established either by assignment or by physically measuring the position and orientation of frame $\mathcal{B}_1$ in frame $\mathcal{G}$. In either case, the frame $\mathcal{B}_1$ is rigidly fixed in frame $\mathcal{G}$, which makes ${}^{\mathcal{G}}_{\mathcal{B}_1}T$ a matrix of constants. Now, any point described in one of the base frames can also be described in the global coordinate frame. For example a point R, which is known with respect to the j^{th} base frame, can be located in the global coordinate frame as follows:

$$
{}^{\mathcal{G}}\vec{R} = \left[{}^{\mathcal{G}}_{\mathcal{B}_1}T\right]\left[{}^{\mathcal{B}_1}_{\mathcal{B}_2}T\right]\left[{}^{\mathcal{B}_2}_{\mathcal{B}_3}T\right]\ldots\ldots\left[{}^{\mathcal{B}_{j-2}}_{\mathcal{B}_{j-1}}T\right]\left[{}^{\mathcal{B}_{j-1}}_{\mathcal{B}_j}T\right]\ {}^{\mathcal{B}_j}\vec{R};
$$

where each transform in the above expression is a function of the forward kinematics of only one cell. Notice that the transformation matrices are 3×3 linear operators (often called homogeneous transforms) that account for both rotation and translation. As a result, the vector $\vec{R}$ must be modified by adding a 1 as the last entry in the column. This makes the dimension of the vector (3×1).

In applying the above algorithm to solve the forward kinematics of a long-chain truss, an explicit assumption was made concerning the forward kinematic solution for each cell. If the solution to the forward kinematics of each cell is limited to one branch, as discussed in Chapter 3, then one unique set of P_x, P_y and θ is obtained. This results in a unique solution for each of the coordinate system transformations, and thus produces only one possible solution for the entire structure. If, however, all four branches of the kinematic solution of the basic cell are deemed acceptable, then there exists 4^{10} or $1,048,576$ mathematically correct assemblies of the ten cell truss. This further emphasizes the need to design the basic cell such that only one physical assembly is possible during operation.

4.2 Ten Cell Spatial VGT

The forward kinematic solution of a long-chain spatial VGT is best viewed as a natural extension of the long-chain planar VGT. Whereas the planar case had only three variable parameters per cell, the spatial truss requires

six parameters per cell. Just as with the planar long chain solution the spatial long-chain forward solution will also rely heavily on the forward kinematic analysis for the single basic cell. The sixty degree-of-freedom spatial truss is generated by replicating the the basic cell shown in Fig. 3.6 and joining the top plane of one cell to the base plane of the next. This produces the truss shown in Fig. 4.3. Once again it will be necessary to modify the single-cell notation to accommodate the existance multiple cells. Figure 4.4 shows the i^{th} cell of a multi-cell spatial truss. For clarity the variable length members which run diagonally across the cell faces have been omitted. Note that the individual cells have been joined such that the $\mathcal{T}_i$ coordinate frame is coincident with the $\mathcal{B}_{i+1}$ frame. In the single-cell forward kinematic analysis it was presented that the relative rotation of frame $\mathcal{T}_i$ described in $\mathcal{B}_i$ could be represented as,

$$ {}^{\mathcal{B}_i}_{\mathcal{T}_i}R = \left[\begin{array}{ccc} {}^{\mathcal{B}_i}\hat{X}_{\mathcal{T}_i} & {}^{\mathcal{B}_i}\hat{Y}_{\mathcal{T}_i} & {}^{\mathcal{B}_i}\hat{Z}_{\mathcal{T}_i} \end{array} \right]; $$

where $\hat{X}_{\mathcal{T}_i}, \hat{Y}_{\mathcal{T}_i}$ and $\hat{Z}_{\mathcal{T}_i}$ are the unit vectors which form the coordinate axes of frame $\mathcal{T}_i$. It was also found in Chapter 3 that these vectors are functions of the six variable link lengths of cell i. A homogeneous transform which completely describes the position and orientation of frame $\mathcal{B}_{i+1}$ ($\mathcal{B}_{i+1} = \mathcal{T}_i$), in frame $\mathcal{B}_i$ can now be formed as follows:

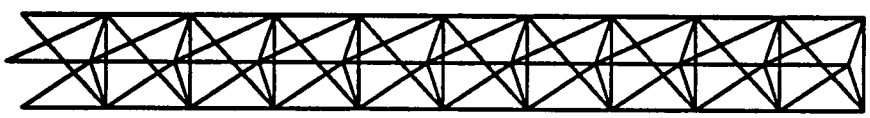

Figure 4.3: Ten Cell, Sixty Degree-of-Freedom Spatial Truss

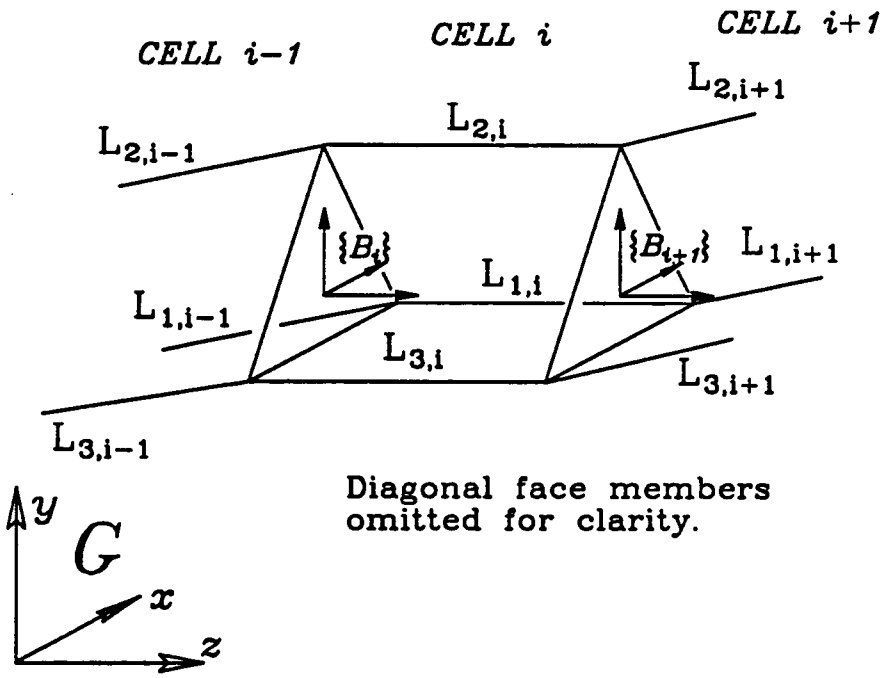

Figure 4.4: i^{th} Cell of a Spatial Truss

$$ {}^{\mathcal{B}_i}_{\mathcal{B}_{i+1}}T = \begin{bmatrix} & \left[{}^{\mathcal{B}_i}_{\mathcal{T}_i}R\right] & & {}^{\mathcal{B}_i}\vec{P}_i \\ 0 & 0 & 0 & 1 \end{bmatrix}. $$

Now it is possible to transform a description from any base frame to the global reference frame using the same procedure as outlined for the planar long chain device. That is to say, any point, R, known with respect to the base frame of the j^{th} cell can also be described in the global coordinate frame by;

$$ {}^{\mathcal{G}}\vec{R} = \left[{}^{\mathcal{G}}_{\mathcal{B}_1}T\right]\left[{}^{\mathcal{B}_1}_{\mathcal{B}_2}T\right]\left[{}^{\mathcal{B}_2}_{\mathcal{B}_3}T\right]\ldots\ldots\left[{}^{\mathcal{B}_{j-2}}_{\mathcal{B}_{j-1}}T\right]\left[{}^{\mathcal{B}_{j-1}}_{\mathcal{B}_j}T\right]\ {}^{\mathcal{B}_j}\vec{R}. $$

Once again the vector $\vec{R}$ must be modified by adding a 1 as the last element which makes $\vec{R}$ dimensionally (4×1).

It is interesting to note the number of possible assemblies for this sixty degree-of-freedom device. As before, if only one branch of the single-cell analysis is allowed, one unique solution for the entire structure results. If however all sixteen branches of the basic cell are deemed acceptable, then there exists 16^{10} or $1,099,511,627,776$ mathematically correct assemblies of the ten cell spatial truss; an overwhelming number of solutions for any modern computer.

4.3 Computational Considerations

It is apparent that the forward kinematic solution for one cell will be used repeatedly in forming the coordinate transformations described above. Because of this, it is very reasonable to use the one cell forward kinematic analysis as a subroutine in the forward kinematic solution of a multi-cell truss. Since each transformation is dependent only on the extensible link lengths of one cell, a more efficient solution could be accomplished by having several dedicated sub-processors which calculate the individual coordinate transformations in parallel. This information would then be passed back to the host processor for use in transforming local position descriptions to the global reference frame. This is a trivial advantage in the case of the planar truss which has closed form expressions for the forward kinematics. The time saved by parallel processing is almost entirely wasted on the time it takes to pass all thirty parameters back and forth between processors. The spatial truss however, presents an entirely different situation. Here, no closed form expressions exist for the forward kinematics. Because of the time required to accomplish even a relatively fast iterative evaluation, the parallel processing solution approaches a ten-fold increase in overall solution speed.

Chapter 5

Inverse Kinematic Solution for Long-Chain Planar Trusses Using Shape Control

This chapter will outline several possible algorithms for solving the inverse kinematics of long-chain high degree-of-freedom VGT manipulators. As an example, the discussion will focus on the thirty degree-of-freedom manipulator shown in Fig. 4.1. To exactly specify the position and orientation of this manipulator by conventional methods requires the specification of thirty lengths for the extensible members. In most cases this would require that an operator specify thirty independent parameters, for example the x, y, and θ of each cell. In practice, however, for many applications the only specific goal to be achieved is the end position of the final cell, X, Y, and Θ. [1] If only the three end parameters are specified, many possible

[1] For this and subsequent chapters, all coordinates which are referenced to a local base frame will be given lowercase characters, and all descriptions referenced to the global coordinate frame will be given uppercase characters.

solutions exist. Indeed, since each cell, for the device described, has an influence on the end position of the truss, there are twenty-seven orders of infinity of solutions. One approach to solve such a system is to optimize on one or several index parameters, such as truss curvature, stiffness, or strength. Unfortunately, for complicated devices such as the described manipulator, this proves to be an overwhelming computational burden for even pseudo-real time control. The goal of this section is to derive an algorithm which will assume a general shape for the manipulator and thus reduce the number of parameters required to attain a given end position and orientation.

5.1 The Shape Control Concept

The underlying philosophy of the shape control approach is to find some minimum order curve which satisfies the desired position requirements of the truss. Once a suitable curve is found, the truss cells are essentially "fitted" to the curve. Thus, the truss will maintain the shape of the specified curve. For example, if the desired end position of the truss was at some position, X, Y, Θ, then it might be possible to fit a line, a quadratic curve, a cubic curve or any higher order curve through the specified point and orientation. In general, a cubic curve will be the minimum order curve necessary to realize an arbitrary goal position . A linear function has no independent control of slope and a quadratic curve cannot have parallel ini-

tial and terminal slopes. Using curves of order higher than cubics will mean that more parameters are needed to define the curve, which consequentially increases the burden on the operator to specify these parameters.

5.2 Parametric Description of Planar Cubic Curves

The following discussion is based on the use of parametric cubic curves. These curves provide several advantages over non-parametric plane curve descriptions. One obvious advantage is that the parametric description allows for curve slopes to pass through vertical, $\frac{dY}{dX} = \infty$, without numerical difficulties. Another, more subtle, advantage is that the planar parametric description requires eight parameters to uniquely determine the curve as opposed to only four for the non-parametric plane curve. This simply allows for more flexibility in controlling the curve shape.

5.2.1 Cubic Specification by End Condition Constraints

The general form of a planar parametric cubic curve is as follows:

$$\begin{aligned} X(u) &= a_{0x} + a_{1x}u + a_{2x}u^2 + a_{3x}u^3, \\ Y(u) &= a_{0y} + a_{1y}u + a_{2y}u^2 + a_{3y}u^3. \end{aligned}$$

Or, utilizing vector notation:

$$\vec{C}(u) = \vec{A}_0 + \vec{A}_1 u + \vec{A}_2 u^2 + \vec{A}_3 u^3;$$

where u is the parametrization variable which progresses from 0 to 1; 0 corresponding to the initial condition or base of the truss, and 1 corresponding to the final condition or the end of the truss. Note that eight coefficients are needed to define the curve. These can be related to eight independent parameters of the curve. Six of these parameters can be utilized for describing the curve end conditions, which are the initial position and orientation (X_0, Y_0, Θ_0), and the final position and orientation (X_1, Y_1, Θ_1). The two remaining parameters, the initial and final tangent vector magnitudes, warrant a special discussion which will be presented in section 5.2.2. For the present discussion the tangent magnitudes will be fixed at unity. By knowing the curve end conditions it is possible to solve for any point along the curve as:

$$X(u) = \begin{bmatrix} 1 & u & u^2 & u^3 \end{bmatrix} [\mathbf{F}] \begin{bmatrix} X_0 \\ X_1 \\ \left.\frac{dX}{du}\right|_0 \\ \left.\frac{dX}{du}\right|_1 \end{bmatrix},$$

$$Y(u) = \begin{bmatrix} 1 & u & u^2 & u^3 \end{bmatrix} [\mathbf{F}] \begin{bmatrix} Y_0 \\ Y_1 \\ \left.\frac{dY}{du}\right|_0 \\ \left.\frac{dY}{du}\right|_1 \end{bmatrix};$$

where $\mathbf{F}$ is a 4×4 matrix which is defined as follows:

$$\mathbf{F} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ -3 & 3 & -2 & -1 \\ 2 & -2 & 1 & 1 \end{bmatrix}.$$

A complete derivation of this matrix is presented in Appendix A. Also note that for the planar curve described

$$\left.\frac{dX}{du}\right|_0 = \cos\Theta_0,$$

$$\left.\frac{dX}{du}\right|_1 = \cos\Theta_1.$$

Similarly,

$$\left.\frac{dY}{du}\right|_0 = \sin\Theta_0,$$

$$\left.\frac{dY}{du}\right|_1 = \sin\Theta_1.$$

To simplify notation the previous matrix equations will be combined into one compact expression as follows:

$$\vec{C}(u) = \begin{bmatrix} 1 & u & u^2 & u^3 \end{bmatrix} [\mathbf{F}] \begin{bmatrix} \vec{C}_0 \\ \vec{C}_1 \\ \vec{C}'_0 \\ \vec{C}'_1 \end{bmatrix};$$

where $\vec{C}(u)$ is the position vector of any point along the curve,

$$\vec{C}(u) = \begin{bmatrix} X(u) \\ Y(u) \end{bmatrix},$$

$\vec{C}_0$ and $\vec{C}_1$ are position vectors which define the initial and terminal end of the curve,

$$\vec{C}_0 = \begin{bmatrix} X_0 \\ Y_0 \end{bmatrix}, \qquad \vec{C}_1 = \begin{bmatrix} X_1 \\ Y_1 \end{bmatrix};$$

and $\vec{C}'_0$ and $\vec{C}'_1$ are vectors which point in the direction of the curve tangent at $u = 0$ and $u = 1$ respectively,

$$\vec{C}'_0 = \begin{bmatrix} \cos\Theta_0 \\ \sin\Theta_0 \end{bmatrix}, \qquad \vec{C}'_1 = \begin{bmatrix} \cos\Theta_1 \\ \sin\Theta_1 \end{bmatrix}.$$

It is now possible to assess the X and Y position of the curve for any value of u on the interval [0,1].

5.2.2 Tangent Vector Effect on Curve Shape

As mentioned previously, a parametric cubic curve needs eight independent parameters specified to uniquely define a curve. The preceding section actively utilized only six of these parameters. Now the effect of the initial and final tangent vectors, t_0, and t_1, can be observed. These can be thought of as the magnitudes of $\vec{C}_0'$ and $\vec{C}_1'$ which are the initial and final curve tangents. In the preceding example t_0 and t_1 were assumed to be unity. The definitions of $\vec{C}_0'$ and $\vec{C}_1'$ can now be modified to reflect the tangent vector magnitude parameters as follows:

$$\vec{C}_0' = \begin{bmatrix} t_0 \cos\Theta_0 \\ t_0 \sin\Theta_0 \end{bmatrix},$$

$$\vec{C}_1' = \begin{bmatrix} t_1 \cos\Theta_1 \\ t_1 \sin\Theta_1 \end{bmatrix}.$$

Figure 5.1 shows a plot of five cubic curves which have the same position and orientation specifications at the ends, however, each curve has a different set of tangent magnitudes. As illustrated in Fig. 5.1, increasing the tangent vector magnitude essentially makes the curve maintain the approximate slope for that end of the curve longer. This also implies that there exists two infinities of solutions for a cubic curve which is specified by only the position and orientation of its ends. This ability of the parametric cubic curve to produce multiple solutions is very attractive for the shape

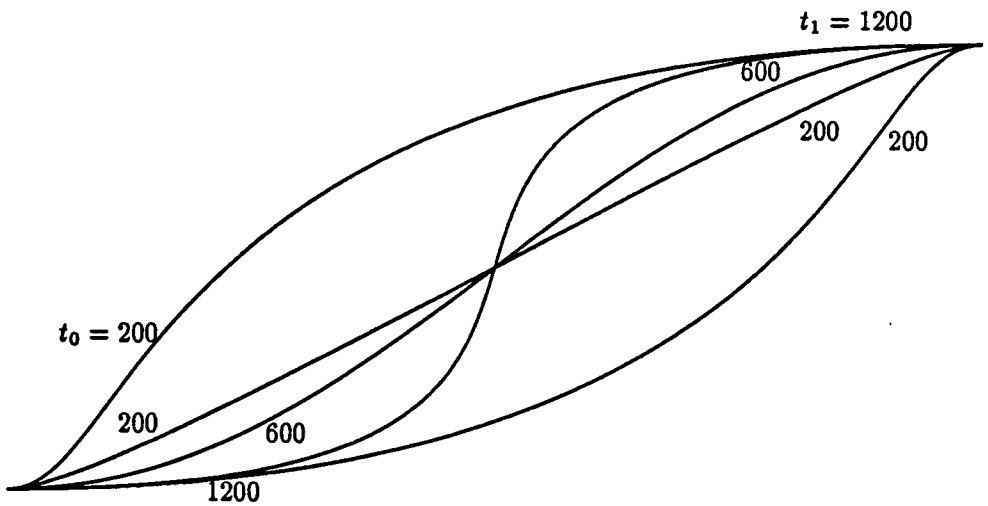

Figure 5.1: Effect of Tangent Vector Magnitude on Curve Shape

control of long devices. If the additional flexibility is not required, the two tangent magnitudes can be set to an arbitrary non-zero value. This property also provides alternative solutions for avoiding workspace obstacles or for optimizing some structural index, such as minimizing the maximum truss curvature or optimizing for strength or rigidity.

5.3 Basic Curve Partitioning and Completion of Truss Solution

Figure 5.2-a illustrates a cubic curve generated using the above procedure. Now a truss must be constructed around this curve. This section will describe a simple method for constructing such a truss.

The inverse kinematics solution for one bay, as presented in Chapter 3, not only needs information about the x and y position of the end link, it also requires information about the orientation of that member. There are a variety of ways to specify this orientation but the most logical is to constrain the end link to lie perpendicular to the curve tangent. Therefore, there curve tangent needs to be defined. This is obtained by differentiating $\vec{C}(u)$ with respect to u. Thus the tangent, $\vec{T}(u)$, can be found for any u on the interval [0,1] as

$$\vec{T}(u) = \frac{d}{du}\vec{C}(u) = \begin{bmatrix} 0 & 1 & 2u & 3u^2 \end{bmatrix} [\mathbf{F}] \begin{bmatrix} \vec{C}_0 \\ \vec{C}_1 \\ \vec{C}'_0 \\ \vec{C}'_1 \end{bmatrix};$$

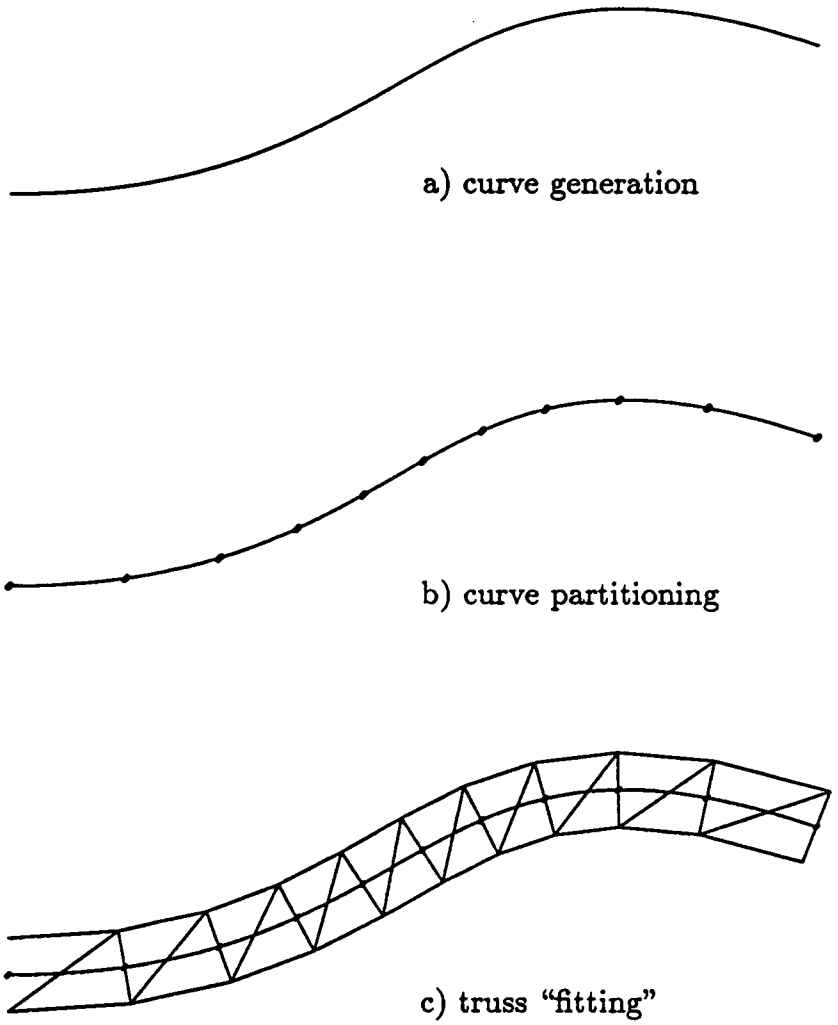

Figure 5.2: Stages for Constructing a Truss Shape Solution

where $\vec{T}(u)$ is a vector tangent to the curve at u. The angle the tangent makes with the X axis is

$$\Theta(u) = \text{ATAN2}(t_Y, t_X);$$

where t_Y and t_X are the X and Y components of the tangent vector.

Now for any value of u[0,1] values can be found for the X, Y, and Θ of the curve. Note that these values of X, Y, and Θ are all referenced to the global coordinate frame.

To solve for the variable length links of the individual cells, it is first necessary to partition the curve into n segments. Where n represents the total number of truss cells to be used. For the ten cell planar truss, eleven nodes along the length of the curve are needed. For demonstration purposes let these nodes be placed at $u = 0.0, 0.1, \ldots, 0.9, 1.0$. For a given cubic curve, this partitioning might result in a curve such as the one illustrated in Fig. 5.2-b. Notice that the partition points are identified as $n_0, n_1, \ldots, n_{10}$; where the subscript denotes which cell terminates at the given partition point. For example, the fifth cell terminates at point n_5, which is located, in this instance, by $\vec{C}(0.5)$. Since the position, (X_i, Y_i), and orientation, Θ_i, are known at any partition point, n_i, it is possible to calculate the pivot locations of any one cell as follows:

$$\vec{A}_i = \begin{bmatrix} X_i - L/2 \sin \Theta_i \\ Y_i + L/2 \cos \Theta_i \end{bmatrix},$$

$$\vec{B}_i = \begin{bmatrix} X_i + L/2 \sin \Theta_i \\ Y_i - L/2 \cos \Theta_i \end{bmatrix}.$$

The three variable link lengths for any one cell can now be found as the distance between appropriate pivots.

$$L_{1i} = ||\vec{A}_i - \vec{A}_{i-1}||,$$

$$L_{2i} = ||\vec{B}_i - \vec{B}_{i-1}||,$$

$$L_{3i} = ||\vec{A}_i - \vec{B}_{i-1}||.$$

These values for the link lengths must be checked to see if they are within the limits of the physical link range. Figure 5.2-c shows a solution generated with the preceding algorithm.

It is interesting to note that the last stages of this algorithm are identical to the inverse solution for one cell, except for a notational subscript change. Thus, the one cell inverse kinematic solution could be called as a subroutine from a shape control algorithm. Or, to attain even higher solution speeds, the shape control algorithm could pass the individual cell

parameters to multiple processors and the solution for all cells could be computed simultaneously in parallel.

5.4 More Advanced Curve Partitioning Techniques

It is evident from observation of Fig. 5.2-c that a regular u spacing (ie. $u = 0.0, 0.1, \ldots, 0.9, 1.0$), does not result in an evenly partitioned curve. This could produce unacceptable solutions in many cases. The goal of this section is to explore several different algorithms for redistributing the partition points to yield a better solution. The premise on which the first three algorithms are based is that the best solution is one which has each cell extended the same length. Utilizing the single cell notation of Chapter 3, the length of the i^{th} cell will be defined as a scalar quantity R_i.

$$R_i = ||\vec{p}_i|| = \sqrt{x_i^2 + y_i^2};$$

where once again the lowercase coordinates are referenced to the base frame of the i^{th} cell. Figure 5.3 shows a section of a parametric cubic curve with the first six partition points illustrated. The scalar variables $n_0, n_1, \ldots, n_5$, will be used to define the partition points. These scalar values represent the value of u which corresponds to each partition point. For example, if $n_3 = 0.3$, then the point n_3 is positioned at $\vec{C}(n_3) = \vec{C}(0.3)$.

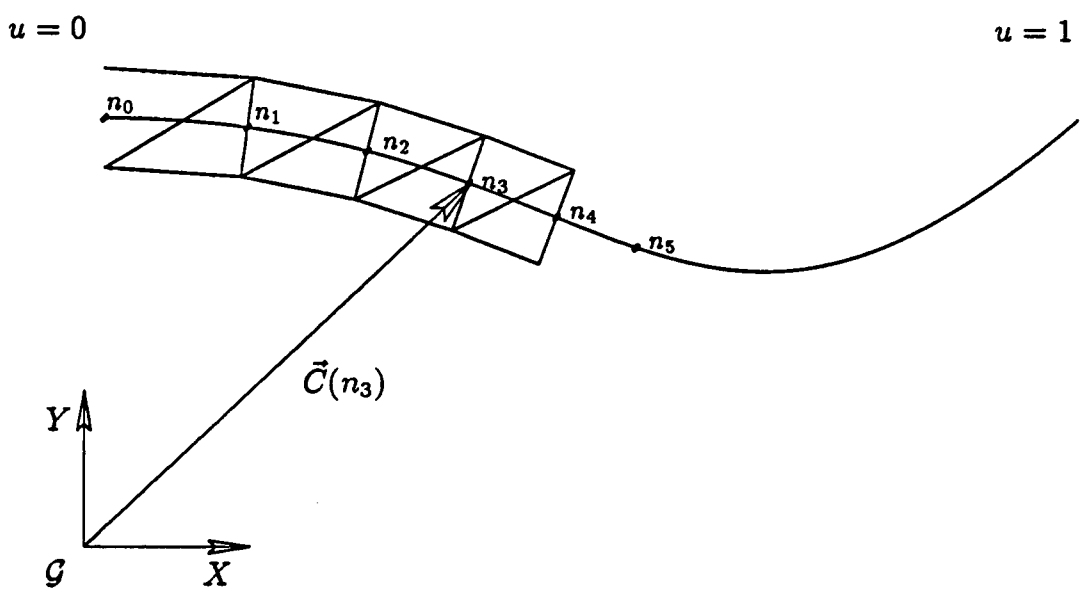

Figure 5.3: Description of Partition Point Locations

Thus the definition of R_i can be modified to reflect a description in global coordinates as follows:

$$R_i = ||\vec{C}(n_i) - \vec{C}(n_{i-1})||.$$

With this notation defined, more advanced curve partitioning techniques can be undertaken.

5.4.1 Nodal Respacing by Tangent Vector Optimization

Figure 5.4-a shows a truss fitted to a parametric cubic curve with the tangent vector magnitudes for both ends equal ($t_0 = t_1 = t_a$) and partitioned based on a regular spacing of u. It can be further observed that the same truss can be modified to have all the cells be of approximately the same length merely by adjusting the magnitude of the tangents. This case is shown in Fig. 5.4-b, where t_a has been adjusted to a new value, t_b ($t_b < t_a$). For most positions within the workspace, a satisfactory solution can be found by making the tangent vector magnitudes equal and adjusting them in unison ($t_0 = t_1 = t$). The appropriate value of t can be found by observing the effect t has on a performance index I.

$$I(t) = \frac{R_{max}}{R_{min}};$$

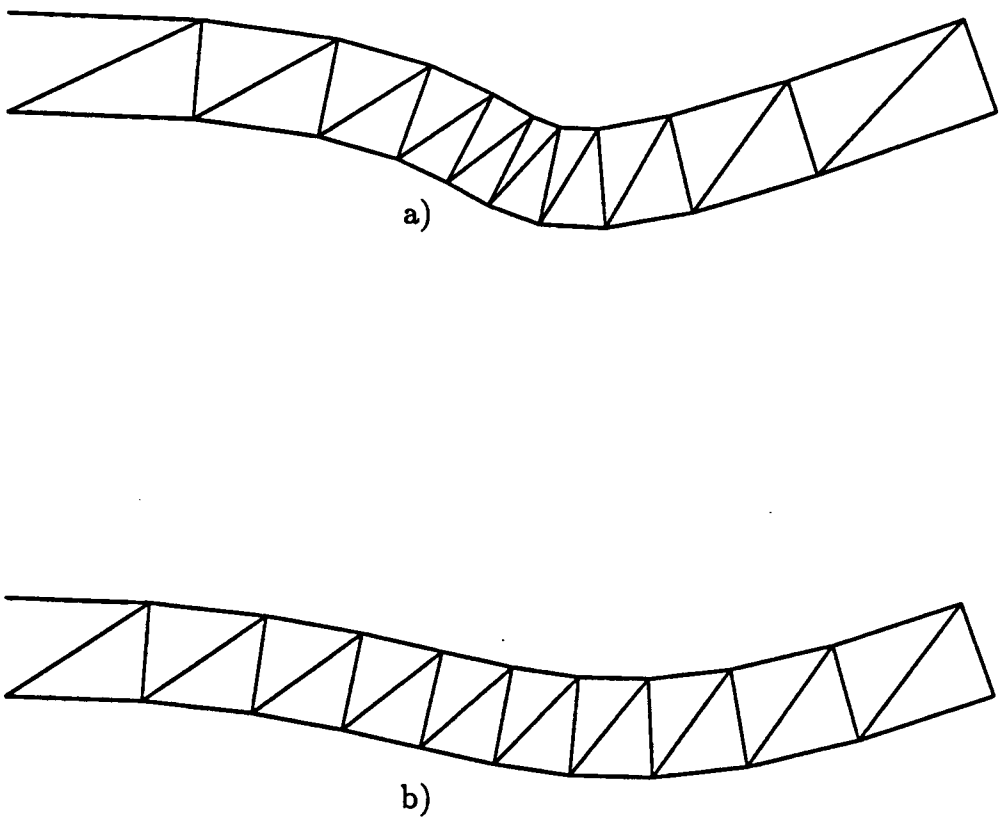

Figure 5.4: Two Planar Truss Solutions Generated During Tangent Vector Optimization

where R_{max} is the largest cell length for all ten cells, and R_{min} is the shortest cell length for all ten cells. Thus, for an ideal solution, $I = 1$. In most configurations this result is impossible, but it is possible to find a value of t which minimizes $I(t)$. This can be done by finding a value for which $\frac{d}{dt}I(t) = 0$. This expression is not a time derivative but a derivative with respect to the tangent vector magnitude. Since no explicit expression is available for $I(t)$, the derivative and the function itself must be numerically evaluated at a few select points of interest. Figure 5.5 shows plots of both $I(t)$ and $\frac{d}{dt}I(t)$ for the curve shape illustrated in Figs. 5.4-a and 5.4-b. Note that the two tangent magnitudes, t_a and t_b, are indicated on this plot.

A simple optimization routine which minimizes $I(t)$ proceeds as follows. First, evaluate $I(t)$ for an arbitrary non-zero value of t. This requires defining and partitioning the curve as outlined previously. $I(t)$ is then evaluated at two closely spaced points; $t = t \pm \epsilon$. These three points provide enough information to evaluate both $\frac{d}{dt}I(t)$ and $\frac{d^2}{dt^2}I(t)$. Expressions for $\frac{d}{dt}I(t)$ and $\frac{d^2}{dt^2}I(t)$ can be formulated from finite difference equations as follows:

$$\frac{d}{dt}I(t) = \frac{I(t+\epsilon) - I(t-\epsilon)}{2\epsilon},$$

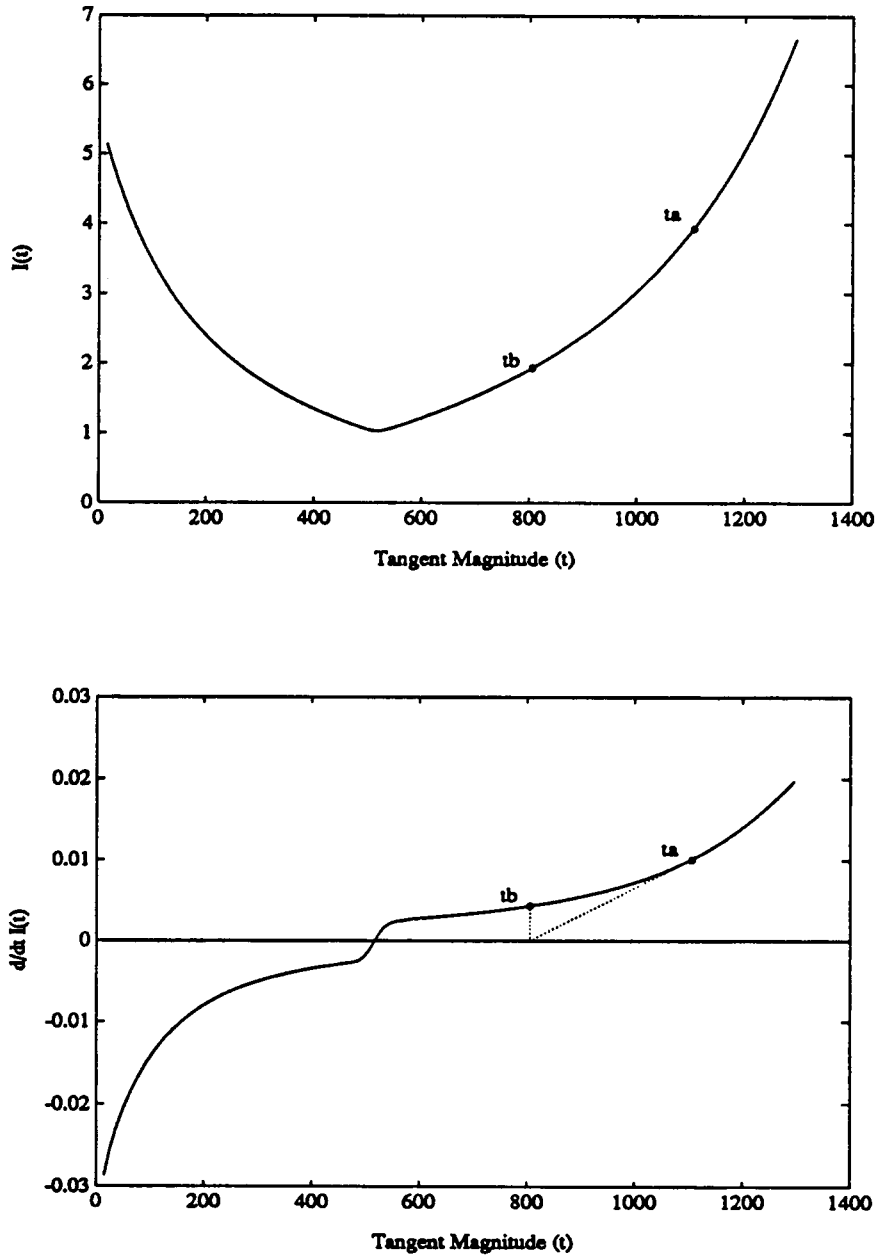

Figure 5.5: $I(t)$ and $\frac{d}{dt}I(t)$ Curves for Shape Shown in Fig. 5.4

$$\frac{d^2}{dt^2}I(t) = \frac{\frac{I(t+\epsilon)-I(t)}{\epsilon} - \frac{I(t)-I(t-\epsilon)}{\epsilon}}{\epsilon}$$

$$= \frac{I(t+\epsilon) - 2I(t) + I(t-\epsilon)}{\epsilon^2}.$$

Once again, these evaluations require forming and partitioning the curve for three different values of t. With the first and second derivative of $I(t)$ evaluated at some arbitrary value of $t = t_a$, it is possible to find a new value, t_b, using a simple linear approximation;

$$t_b = t_a - \frac{\frac{d}{dt}I(t_a)}{\frac{d^2}{dt^2}I(t_a)}.$$

The graphical representation of this first iteration is illustrated in Fig. 5.5. In general, this solution technique will converge to an acceptable solution in just a few iterations; typically less than seven from an arbitrary starting point, and less than four for a small displacement from a previously found optimal solution. However, cases such as the one illustrated in Fig. 5.5 present special convergence problems. It is therefore recommended that a more robust root-finding algorithm be utilized.

One advantage of this routine is that it does prove to be quite fast for small displacements. Usually during the position control of a robotic

device, there is no need to ever calculate a solution which is a large distance from the current location. The logic behind this reasoning is that the controller could just as easily compute intermediate points while the manipulator is moving to the last specified point rather than sitting idle while waiting for the manipulator to complete one large move. One other slight advantage of this algorithm is that it relieves the operator from specifying the tangent magnitude parameters. However, it must also be realized that this convenience simultaneously removes some of the flexibility of the cubic curve.

There are two serious drawbacks to implementing this method of partitioning. First, since the two tangent vector magnitudes are assigned automatically, only one unique solution results for a given set of $X_0, Y_0, \Theta_0, X_1, Y_1$, and Θ_1. If this one solution is unacceptable because of extensible link ranges or obstacles in the workspace, then the truss simply cannot reach the desired goal with this algorithm. Another major disadvantage is that the algorithm only minimizes $I(t)$; it does not necessarily force $I(t)$ to be unity. The next algorithms to be discussed will succeed in making $I(t) \approx 1$.

5.4.2 Nodal Respacing by Approximate Equal Arc Length

The previous method of nodal respacing was computationally fast, but it did not guarantee that $I(t)$ would be acceptably close to unity. The following method will assure that $I(t) \approx 1$. This is done simply by calculating

the length of the curve from $u = 0$ to $u = 1$ and then finding the correct intermediate values of u such that the arc length between any two adjacent nodes is one tenth of the total arc length. The total arc length cannot, in general, be found in closed form and must be calculated numerically as follows:

$$S_{total} = \int_0^1 \left(d\vec{C} \cdot d\vec{C}\right)^{1/2} du \approx \sum_{j=1}^{n} \|\vec{C}(j/n) - \vec{C}((j-1)/n)\|$$

where n is the total number of steps used to estimate the arc length. A larger number of steps will yield higher accuracy, but will also increase computation time. Selecting a value of n which is too large will introduce quantization errors into the result. A value of $n = 200$ will typically produce excellent results and still maintain a modest computation time. The remaining task is to determine at what values of j does the summation above equal $0.1S_{total}, 0.2S_{total}, \ldots, 0.8S_{total}$, and $0.9S_{total}$. The summation will have to be computed a second time to gain this information. Once this has been completed, the resulting values of j can be related to u as follows:

$$u = (j/n).$$

Now the solution can proceed as outlined in Section 5.3 with the above modification applied to the u spacing.

5.4.3 Nodal Respacing by Approximate Equal Chord Length

Whereas the approximate equal arc length algorithm concentrated on finding a solution in which $I \approx 1$, the following approach centers on satisfying the relation $I = 1$. This can be accomplished by comparing the length squared of each cell to that of its adjacent cells. This results in nine simultaneous non-linear equations which can be expressed as functions of the partition points as follows:

$$\begin{array}{rcccl}
f_1 & = & (R_2)^2 - (R_1)^2 & = & ||\vec{C}(n_2) - \vec{C}(n_1)||^2 - ||\vec{C}(n_1) - \vec{C}(n_0)||^2, \\
f_2 & = & (R_3)^2 - (R_2)^2 & = & ||\vec{C}(n_3) - \vec{C}(n_2)||^2 - ||\vec{C}(n_2) - \vec{C}(n_1)||^2, \\
\vdots & & \vdots & & \vdots \\
f_i & = & (R_{i+1})^2 - (R_i)^2 & = & ||\vec{C}(n_{i+1}) - \vec{C}(n_i)||^2 - ||\vec{C}(n_i) - \vec{C}(n_{i-1})||^2, \\
\vdots & & \vdots & & \vdots \\
f_9 & = & (R_{10})^2 - (R_9)^2 & = & ||\vec{C}(n_{10}) - \vec{C}(n_9)||^2 - ||\vec{C}(n_9) - \vec{C}(n_8)||^2;
\end{array}$$

where $n_0 \ldots n_{10}$ are the values of u which correspond to the partition points of the curve. If n_0 and n_{10} are considered fixed at $u = 0$ and $u = 1$ respectively, then the remaining task is to find values for $n_1, n_2, \ldots, n_9$ which simultaneously drive all nine equations to zero. This can be accomplished using a Newton-Raphson linear approximation as follows:

$$\begin{bmatrix} \frac{\partial f_1}{\partial n_1} & \frac{\partial f_1}{\partial n_2} & & & & & & & \\ \frac{\partial f_2}{\partial n_1} & \frac{\partial f_2}{\partial n_2} & \frac{\partial f_2}{\partial n_3} & & & & & & \\ & \frac{\partial f_3}{\partial n_2} & \frac{\partial f_3}{\partial n_3} & \frac{\partial f_3}{\partial n_4} & & & & & \\ & & \ddots & \ddots & \ddots & & & & \\ & & & \frac{\partial f_i}{\partial n_{i-1}} & \frac{\partial f_i}{\partial n_i} & \frac{\partial f_i}{\partial n_{i+1}} & & & \\ & & & & \ddots & \ddots & \ddots & & \\ & & & & & \frac{\partial f_7}{\partial n_6} & \frac{\partial f_7}{\partial n_7} & \frac{\partial f_7}{\partial n_8} & \\ & \mathcal{O} & & & & & \frac{\partial f_8}{\partial n_7} & \frac{\partial f_8}{\partial n_8} & \frac{\partial f_8}{\partial n_9} \\ & & & & & & & \frac{\partial f_9}{\partial n_8} & \frac{\partial f_9}{\partial n_9} \end{bmatrix} \begin{bmatrix} \Delta n_1 \\ \Delta n_2 \\ \Delta n_3 \\ \vdots \\ \Delta n_i \\ \vdots \\ \Delta n_7 \\ \Delta n_8 \\ \Delta n_9 \end{bmatrix} = \begin{bmatrix} -f_1 \\ -f_2 \\ -f_3 \\ \vdots \\ -f_i \\ \vdots \\ -f_7 \\ -f_8 \\ -f_9 \end{bmatrix}.$$

This linear system of equations can be efficiently solved using a specialized tridiagonal Gaussian elimination algorithm such as the one outlined by [25, pages 231–235].

Before applying this algorithm, values must be obtained for each of the elements within the matrix of partial derivatives. Examination of this matrix reveals that there are only three general forms of partial derivatives which need to be evaluated; $\frac{\partial f_i}{\partial ni-1}$, $\frac{\partial f_i}{\partial ni}$, and $\frac{\partial f_i}{\partial ni+1}$. Since VGT structures are themselves repetitive in nature, it is not surprising to find that all of the governing kinematic equations typically have the same form. This unique characteristic often leads to results which can be simplified computationally

by employing only a change of subscript to represent all of the equations. This property makes the use of closed form partial derivatives very attractive. Once the partials are derived for one equation, the results can be generalized for all equations of similar form.

The nine simultaneous equations can be represented, using the vector dot product, as,

$$\begin{aligned} f_i &= R_{i+1}^2 - R_i^2 \\ &= \left(\vec{C}(n_{i+1}) - \vec{C}(n_i)\right) \cdot \left(\vec{C}(n_{i+1}) - \vec{C}(n_i)\right) \\ &\qquad - \left(\vec{C}(n_i) - \vec{C}(n_{i-1})\right) \cdot \left(\vec{C}(n_i) - \vec{C}(n_{i-1})\right) \\ &= \vec{C}(n_{i+1}) \cdot \vec{C}(n_{i+1}) - 2\left(\vec{C}(n_{i+1}) \cdot \vec{C}(n_i)\right) \\ &\qquad + 2\left(\vec{C}(n_{i-1}) \cdot \vec{C}(n_i)\right) - \vec{C}(n_{i-1}) \cdot \vec{C}(n_{i-1}) \end{aligned}$$

Realizing that $\frac{\partial \vec{C}(u)}{\partial u} = \vec{T}(u)$, all of the partials derivative required to populate the tridiagonal matrix can be exactly expressed as functions of the curve partition points and the curve tangent at the partition points as follows:

$$\begin{aligned} \frac{\partial f_i}{\partial n_{i-1}} &= 2\left\{\vec{T}(n_{i-1}) \cdot \left[\vec{C}(n_i) - \vec{C}(n_{i-1})\right]\right\}, \\ \frac{\partial f_i}{\partial n_i} &= 2\left\{\vec{T}(n_i) \cdot \left[\vec{C}(n_{i-1}) - \vec{C}(n_{i+1})\right]\right\}, \\ \frac{\partial f_i}{\partial n_{i+1}} &= 2\left\{\vec{T}(n_{i+1}) \cdot \left[\vec{C}(n_{i+1}) - \vec{C}(n_i)\right]\right\}. \end{aligned}$$

There is now enough information to implement a tridiagonal solver and obtain values for $\Delta n_1, \ldots, \Delta n_9$. These values can then be used to modify the distribution of partition points as follows:

$$n_i = n_i + \Delta n_i, \qquad \text{for } i = 1, 2, \ldots 9.$$

This procedure then continues until all f_i 's are simultaneously driven to zero.

5.4.4 Nodal Respacing by One-Dimensional Optimal Distribution

This approach will not be discussed in great detail because of the many possibilities that exist for optimization criteria. The three preceding algorithms were based on the assumption that the best possible solution was one in which all cells were equally extended. Indeed for many applications this seems an appropriate assumption. However, there are limitations to such an approach. For example, in some orientations the equal chord length algorithm might yield a solution in which one or more extensible links of the truss are extended beyond their limit. One algorithm which could possibly avoid such a situation could be implemented as follows. To each extensible link assign a penalty. If a link is at its mid position, then little or no penalty would be assessed. If the link in question was at its limit of travel, then a prohibitively large penalty should be assessed. One possibility is to have

the penalty be an exponential function of the link's distance from its center position. Now a distribution of the partition points can be sought which minimizes the total penalties for the structure. Note that this algorithm only changes the location of the partition points, all eight parameters of the shape control are still at the disposal of the operator. This solution would maintain the cubic shape of the curve while enabling the manipulator to reach further into its workspace than is possible with the other algorithms. Another advantage of this approach is that it is in no way limited to purely kinematic optimization criteria. This approach could be utilized in finding a minimum energy move between two specified goal positions, or to produce a minimum stress solution for a particular goal position. This solution technique should not be confused with a fully optimal approach which must account for many more variables.

5.5 Control of an Intermediate Position and Orientation

It has been shown that a planar parametric cubic curve is capable of producing multiple solutions to a long-chain truss positioning problem if the only point of interest is the final link. In cases where the position and orientation of other links are also critical, a higher degree-of-freedom is required. This increase in the number of degrees of freedom can be obtained by using higher order curves or by using multiple cubic curves joined together to form

a cubic spline. The only restriction imposed on these two curves is that they be compatible at the junction, meaning they must have the same position and tangent direction at the junction. They need not have matching tangent magnitudes to satisfy this compatibility requirement. One advantage of using this cubic spline approach is that other than the compatibility requirement, the two curves are entirely independent. Thus it is possible to form a cubic spline in which one section of the curve is a true cubic and the other section is linear (a degenerate cubic curve). Figure 5.6 shows a truss which extends straight for most of its length and then curves abruptly upward for the final three cells. Clearly this assembly would not be possible with a single higher order curve.

Expressions can now be developed for the planar cubic spline described above. First it shall be assumed that the resulting curve will still be generated by incrementing a parametrization variable, u, from 0 to 1. With this in mind, it must now be decided at what value of u will the transition, or knot, between curves occur. For any of the curve partitioning techniques which redistribute the partition points, the u value of the transition is inconsequential as far as the resulting curve shape. It can however affect the accuracy of the total arc length calculation. This can be avoided by assigning the knot a u value which is roughly equal to the ratio of the first curve length to the total spline length. For example if it is estimated that the first and second cubic are of approximately the same length, then the

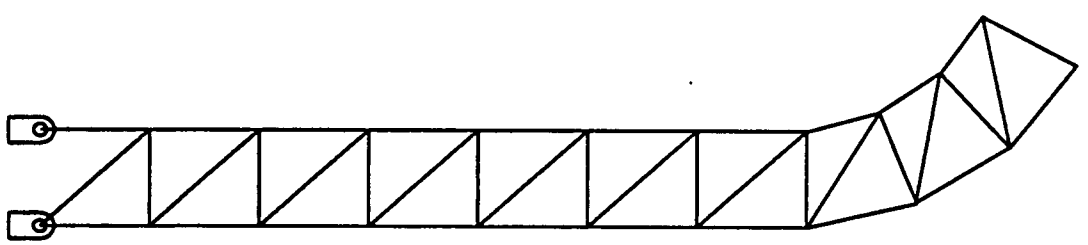

Figure 5.6: Planar Truss Formed with Cubic Spline

transition would occur at $u = 0.5$. Expressions for the two curves would then be obtained as follows:

$$\vec{C}(u) = \begin{bmatrix} 1 & u & u^2 & u^3 \end{bmatrix} [\mathbf{F}] \begin{bmatrix} \vec{C}_0 \\ \vec{C}_{0.5} \\ \vec{C}'_0 \\ \vec{C}'_{0.5} \end{bmatrix} \qquad 0 \leq u \leq 0.5,$$

$$\vec{C}(u) = \begin{bmatrix} 1 & u & u^2 & u^3 \end{bmatrix} [\mathbf{F}] \begin{bmatrix} \vec{C}_{0.5} \\ \vec{C}_1 \\ \vec{C}'_{0.5} \\ \vec{C}'_1 \end{bmatrix} \qquad 0.5 \leq u \leq 1.0.$$

In some cases it may be desirable to assign one more physical constraint to the partitioning of the curve. In the discussion above the only criteria was that the curve pass through the given intermediate position and slope. It was not required that this intermediate position correspond to a partition point. It is possible to force one of the partition points to occur at the knot. This is advantageous if a piece of equipment, such as a camera, needs to be accurately positioned. It should also be noted that this method could be easily extended to more than two cubic curves.

Chapter 6

Inverse Kinematic Solution for Long-Chain Spatial Trusses Using Shape Control

This chapter will extend many of the concepts developed for shape control of a long-chain planar truss to the control of a long-chain spatial truss. As an example, the discussion will focus on the sixty degree-of-freedom manipulator shown in Fig. 4.3. Conventional positioning methods would require the specification of sixty independent parameters. The shape control approach reduces this number to fourteen parameters for general arbitrary end conditions, or eight parameters for applications where one end of the truss is fixed.

Similar to the planar truss algorithms, the spatial shape control approach will seek a solution which smoothly progresses from one given end condition to the other. It shall be shown that a simple spatial cubic curve does not possess a sufficient number of degrees-of-freedom to satisfacto-

rily position a spatial element for use in constructing a continuous chain. Because of this, a bi-parametric, helically swept surface will be utilized.

6.1 Parametric Description of Swept Surface

The general cartesian form of a spatial cubic curve can be represented as follows:

$$X(u) = a_{0x} + a_{1x}u + a_{2x}u^2 + a_{3x}u^3,$$
$$Y(u) = a_{0y} + a_{1y}u + a_{2y}u^2 + a_{3y}u^3,$$
$$Z(u) = a_{0z} + a_{1z}u + a_{2z}u^2 + a_{3z}u^3.$$

Or, in vector form,

$$\vec{C}(u) = \vec{A}_0 + \vec{A}_1 u + \vec{A}_2 u^2 + \vec{A}_3 u^3.$$

Utilizing a notation similar to that of the previous chapter, the position of any point along the curve can be expressed as a function of the curve end conditions. Thus,

$$\vec{C}(u) = \begin{bmatrix} 1 & u & u^2 & u^3 \end{bmatrix} [\mathbf{F}] \begin{bmatrix} \vec{C}(0) \\ \vec{C}(1) \\ \vec{T}(0) \\ \vec{T}(1) \end{bmatrix};$$

where,

$$\vec{C}(u) = \begin{bmatrix} X(u) \\ Y(u) \\ Z(u) \end{bmatrix},$$

$$\vec{T}(0) = t_0 \left.\frac{d\vec{C}(u)}{du}\right|_0, \qquad \text{and} \qquad \vec{T}(1) = t_1 \left.\frac{d\vec{C}(u)}{du}\right|_1.$$

Specification of this cubic curve requires twelve independent parameters, three for each position vector and three for each tangent vector. These correspond to the twelve parameters afforded by the general parametric description.

Now for any value of u, both a spatial position vector and a spatial tangent vector can be found. The position vector will be utilized, as it was for the planar case, to specify where one of the local base coordinate frames will be located. See Fig. 6.1. The tangent vector will be used to orient one axis of the local base coordinate frame; for this particular application, the z axis. Note that there are still an infinite number of orientations which can satisfy these constraints. They are generated by rotating the local base frame about its z axis. Thus, there is still a need to describe the desired rotation about the local z axis. It is possible to extract the curve normal and bi-normal vectors and use these to assign the local base frame. These two

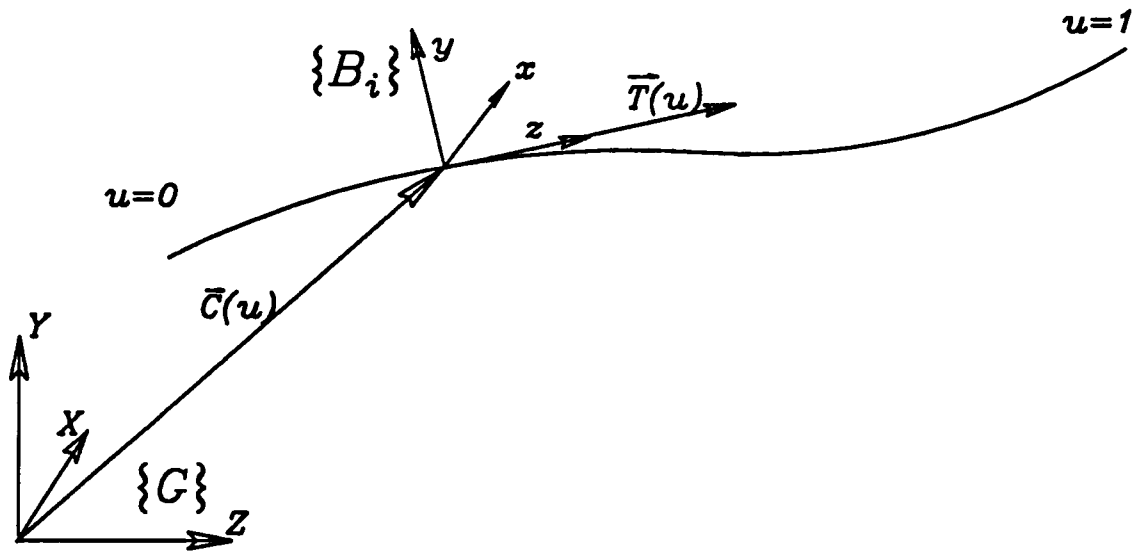

Figure 6.1: Description of i^{th} Base Frame in Global Coordinates

curve characteristics, however, are discontinuous or in some cases undefined along the length of the curve. Because of this shortcoming, a cubic curve alone will not provide enough information to properly orient the local base frames. In short, more than twelve parameters are needed to define the desired shape.

The additional degrees of freedom required can be obtained by employing a surface rather than a curve. In this case a swept surface is used. This surface is formed simply by moving a given shape, termed the generatrix, along a guiding curve, the directrix, such that the plane of the generatrix is always perpendicular to the directrix [26]. Such a motion results in the surface illustrated in Fig. 6.2-a; where the directrix function is the prescribed cubic curve and the generatrix function is an equilateral triangle with sides of length L. This method of generation produces a surface which smoothly progresses to the desired end position and tangent direction. It does not however, have any capacity to result in an arbitrary end rotation about the curve tangent. If however, simultaneous with the motion along the curve, the generatrix is also rotated about the curve, the result is a helically swept surface as shown in Fig. 6.2-b. By specifying the proper rate of twist, the rotation of the each individual base frame can be assigned such that the surface smoothly twists to the desired end orientation. Since it is convenient to describe the surface in terms of the specified end conditions, two more parameters are added to describe this helical twist. These are

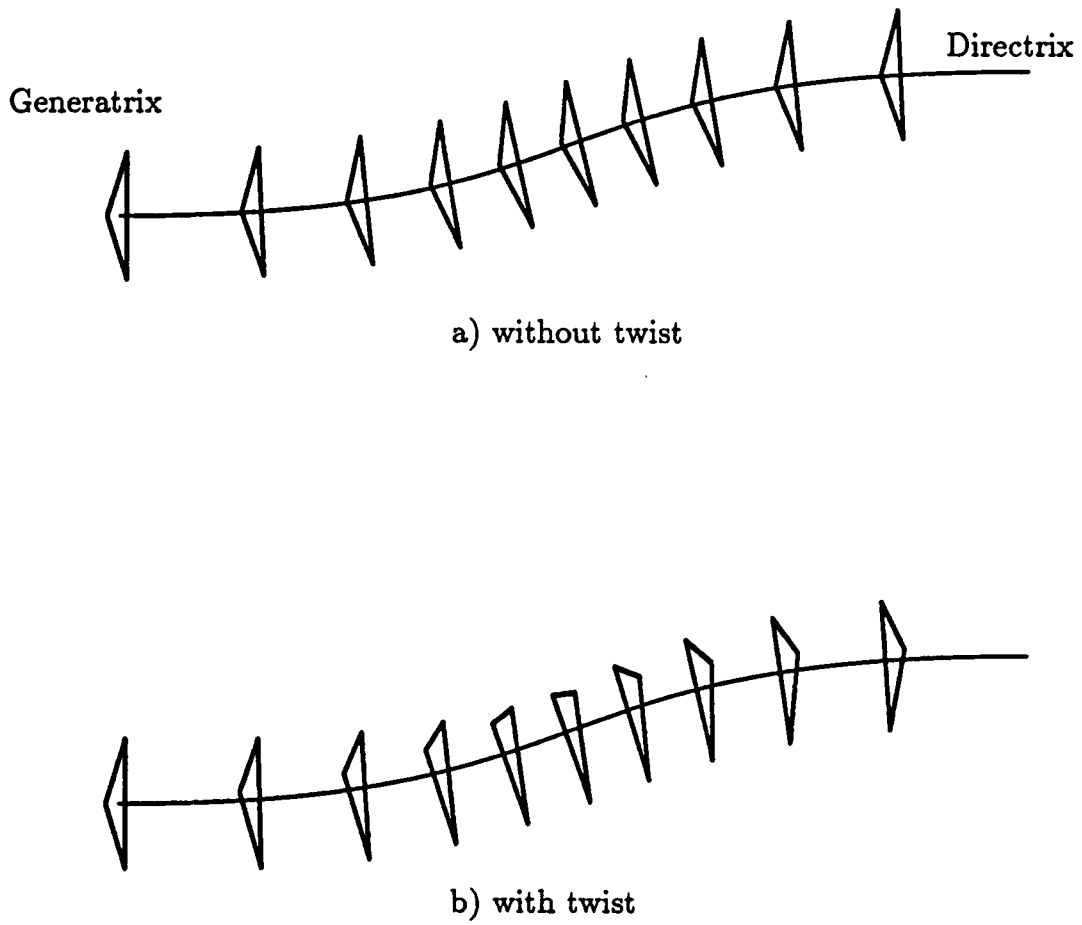

Figure 6.2: Generation of Swept Surface

the initial twist about the curve tangent, λ_i, and the final twist about the curve tangent, λ_f. Additional information about general swept surfaces is contained in [27, pages 455–460].

The defining equations for the directrix function have already been discussed. It is now necessary to briefly discuss how to conveniently form the generatrix function. This is most easily accomplished by forming a circle as shown in Fig. 6.3. The circle illustrated lies in the $x - y$ plane and has a radius of $r = \frac{2}{3}(L \sin 60°)$. Thus, any point on the circle can be defined as a function of ν as follows:

$$\vec{w}(\nu) = \begin{bmatrix} r\cos\nu \\ r\sin\nu \\ 0 \end{bmatrix}.$$

If this expression is evaluated at $\nu = 0, \frac{2\pi}{3}$, and $\frac{4\pi}{3}$, the three resulting points can be used to define the triangular generatrix. As this generatrix is swept along the cubic curve, a bi-parametric surface is generated which is a function of u and ν.

6.2 Basic Curve Partitioning and Completion of Truss Solution

This section will concentrate on the actual methods used to carry out the concepts discussed above. The surface shape has already been defined by

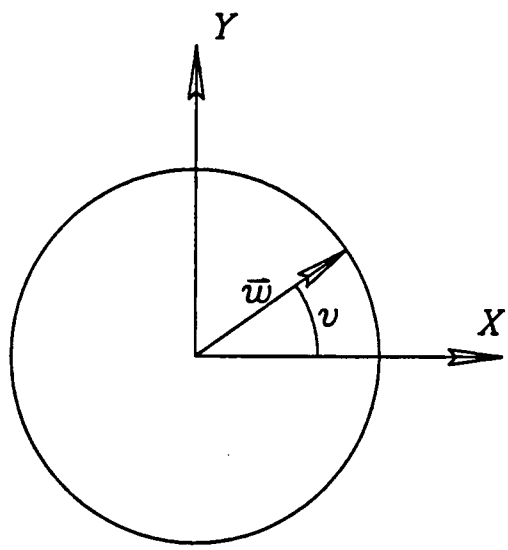

Figure 6.3: Formation of Generatrix Function

the directrix, generatrix, and twist angles. Now the task at hand is to use this information to somehow "fit" a truss structure inside the surface. For the present discussion it will be assumed that a regular u partitioning is sufficient to produce an acceptable solution. More refined partitioning techniques will be addressed in subsequent sections.

For any value of u the spatial cubic curve description provides information about the curve position and tangent vectors. If the rate of twist of the surface is assumed to be constant with respect to u, then the twist for any value of u is,

$$\lambda(u) = (1-u)\lambda_i + u\lambda_f$$

Now the rotation of any base coordinate frame, $\mathcal{B}_i$, which occurs at a partition point, n_i, can be found using the following procedure.

Begin with frame $\mathcal{B}_i$ coincident with the global reference frame, $\mathcal{G}$. It was mentioned previously that the z axis of frame $\mathcal{B}_i$ should be parallel to the curve tangent. This requires, in general, two rotations, shown in Fig. 6.4. Utilizing X-Y-Z Euler angle descriptions (sometimes referred to as Bryant angle descriptions) [24, pages 42–45][28, pages 347–352], the desired rotation about the x axis of frame $\mathcal{B}_i$ is,

$$\gamma(u) = -\text{ATAN2}({}^{\mathcal{G}}t_y, {}^{\mathcal{G}}t_z);$$

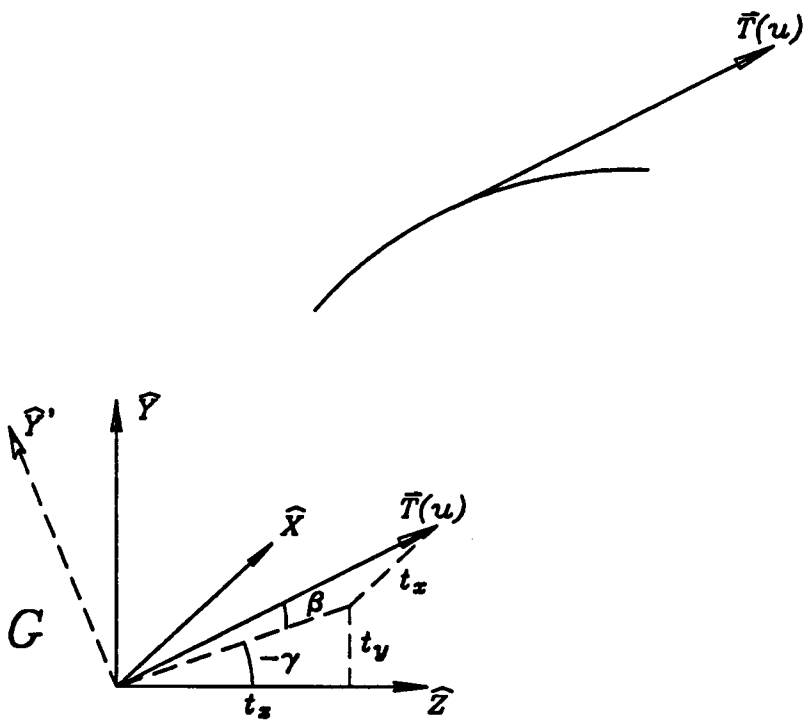

Figure 6.4: X-Y-Z Euler Angle Description of Tangent Vector

where t_x, t_y, and t_z are the X, Y, and Z components of the tangent vector $\vec{T}(u)$. The second rotation, which occurs about the now rotated y' axis of frame $\mathcal{B}_i$, is found as,

$$\beta(u) = \text{ATAN2}({}^{\mathcal{G}}t_x, \sqrt{{}^{\mathcal{G}}t_y{}^2 + {}^{\mathcal{G}}t_z{}^2}),$$

if t_z is positive. Or, if t_z is negative,

$$\beta(u) = \text{ATAN2}({}^{\mathcal{G}}t_x, \sqrt{{}^{\mathcal{G}}t_y{}^2 + {}^{\mathcal{G}}t_z{}^2}) + \pi.$$

Note that the z axis of $\mathcal{B}_i$ is now coincident with the curve tangent. Finally, the rotation of frame $\mathcal{B}_i$ about its own z'' axis ($z'' = \frac{\vec{T}(u)}{||\vec{T}(u)||}$) is given from the previous definition of the surface twist. That is,

$$\alpha(u) = \lambda(u).$$

These three rotations can now be combined into one rotation matrix as follows:

$$\begin{aligned} R_{X-Y-Z}(u) &= ROT(\hat{X}_{\mathcal{B}_i}, \gamma) ROT(\hat{Y}_{\mathcal{B}_i}, \beta) ROT(\hat{Z}_{\mathcal{B}_i}, \alpha) \\ &= \begin{bmatrix} c\alpha c\beta & -s\alpha c\beta & s\beta \\ c\alpha s\beta s\gamma + s\alpha c\gamma & -s\alpha s\beta s\gamma + c\alpha c\gamma & -c\beta s\gamma \\ -c\alpha s\beta c\gamma + s\alpha s\gamma & s\alpha s\beta c\gamma + c\alpha s\gamma & c\beta c\gamma \end{bmatrix} \end{aligned}$$

This rotation matrix can be used in conjunction with the translation information to yield a homogeneous transformation which maps vectors from the $\mathcal{B}_i$ coordinate frame to the global reference frame $\mathcal{G}$.

$$ {}^{\mathcal{G}}_{\mathcal{B}_i}T = \begin{bmatrix} \begin{bmatrix} & R_{X-Y-Z}(u) & \end{bmatrix} & \begin{bmatrix} \vec{C}(u) \end{bmatrix} \\ \begin{bmatrix} 0 & 0 & 0 \end{bmatrix} & 1 \end{bmatrix}. $$

Since the three corner points of the static triangles are fixed with respect to the local base frames, the preceding transformation can also be used to translate and rotate the generatrix to its new position and orientation. This results in a bi-parametric description of the corner points as follows:

$$ \vec{w} = \left[{}^{\mathcal{G}}_{\mathcal{B}_i}T(u) \right] \begin{bmatrix} r\cos\nu \\ r\sin\nu \\ 0 \\ 1 \end{bmatrix} $$

This expression can now be evaluated at any value of u for values of $\nu = 0, 2\pi/3$, and $4\pi/3$, to yield three vectors which define the three corner points. The lengths of the extensible members are then found as the distance between appropriate corners of adjacent triangles. Figure 6.5 shows a truss constructed using this algorithm with a regular u partition.

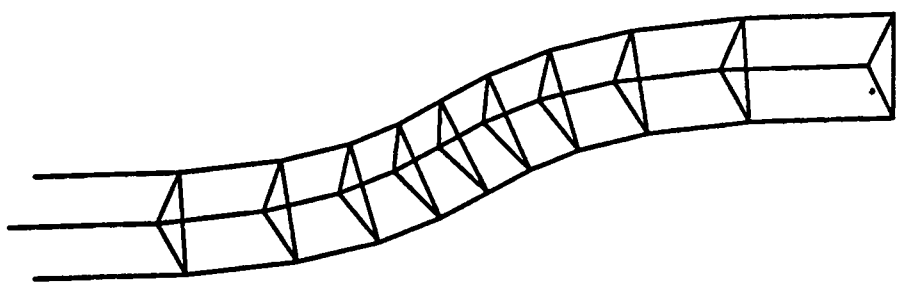

Figure 6.5: Spatial Truss Partitioned with Regular u spacing

6.3 More Advanced Partitioning Techniques

Because of the great similarity between the planar and spatial partitioning routines, this discussion will be somewhat abbreviated. Only necessary modifications to the planar routines will be presented here.

All of the partitioning methods discussed in Chapter 5 are easily extended to the spatial case. The only modification which must be made is to make the cell length a value which incorporates all three coordinate directions. Thus, for any cell i,

$$R_i = ||\vec{p}_i|| = \sqrt{x_i^2 + y_i^2 + z_i^2}.$$

Or, in terms of the defined curve in global coordinates,

$$R_i = ||\vec{C}(n_i) - \vec{C}(n_{i-1})||$$

The nodal respacing routines can now function exactly as they did for the planar truss. However, one parameter, which must be modified, is not addressed by these routines. This is the rate of twist of the helix. Earlier it was assumed that the rate of twist was constant with respect to u. This resulted in the twist angle being expressed solely as a function of u as follows:

$$\lambda(u) = (1-u)\lambda_i + u\lambda_f.$$

It is easy to see that such a twist law might result in a surface which has a variable rate of twist with respect to its arc length. A more satisfactory solution can be obtained by assuming the rate of twist is constant with respect to the arc length, s. This results in the twist law,

$$\lambda(s) = \left(1 - \frac{s}{S_{total}}\right)\lambda_i + \frac{s}{S_{total}}\lambda_f;$$

where s and S_{total} are calculated in section 5.4.2. This will result in the desired twist profile, but it is only convenient to apply this to the approximate equal arc length algorithm.

Since all of the respacing routines center on producing equal cell extensions, a more feasible solution is obtained by assuming the rate of twist is constant with respect to the cell number i. Thus an easy approximation for all of the respacing techniques is to have,

$$\lambda(i) = \left(1 - \frac{i}{n}\right)\lambda_i + \frac{i}{n}\lambda_f;$$

where i is the cell number and n is the total number of cells in the chain.

This produces practically identical results to the arc length method, but imposes no additional calculations. Figure 6.6 shows a truss partitioned with the equal arc length algorithm and twisted as a function of the cell number.

6.4 Control of an Intermediate Position and Orientation

The methods for joining two spatial surfaces are almost identical to those used to join two planar cubic curves. Only two additional compatibility requirements must be imposed. First, the generatrix must be the same for both curves. Also, the final twist angle of the first curve must equal the initial twist angle of the second curve. The rates of twist for each surface should be considered independent. Obviously, this method could easily be extended to joining more than two surfaces.

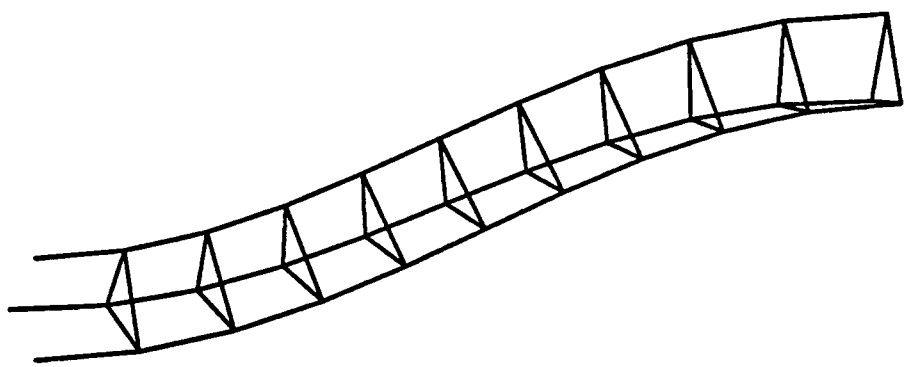

Figure 6.6: Spatial Truss Partitioned with Equal Arc Length Algorithm

Chapter 7

Conclusions and Recommendations for Further Research

The shape control concept has proven to be a viable method for controlling the position and orientation of long-chain VGT manipulators. For this thesis, the assumed shape was that of a parametric cubic curve. Other curves are also possible and the merits of these should be investigated. Two such possibilities are B-splines and Bezier curves. The methods for generating a swept surface with these curves as the directrix function would be virtually the same as outlined in Chapter 6.

Once the truss shape was specified, efficient methods were needed to partition the curve or surface and "fit" the truss to the desired shape. The algorithms outlined were programmed in C on an IBM compatible personal computer. To verify that the results obtained were correct, the computed assemblies were displayed on a graphics monitor. The figures

of long-chain trusses contained within this thesis were generated using the described algorithms.

All of the routines performed satisfactorily and accomplished the stated goals. The performance of each algorithm will now be discussed. The tangent vector optimization routine dramatically illustrates the effect the tangent vector magnitudes have on the overall truss shape. This method is interesting in that it provides insight into the behavior of parametric cubic curves, however it is not very useful for controlling the shape of a truss. This method simply takes too much control away from the operator and does not guarantee convergence to solution. The two remaining routines preserve all of the flexibility provided by the parametric cubic curves. To ensure that an acceptable redistribution of partition points occurs, the approximate equal arc length algorithm is the favored approach. This is the only method which does not rely on the convergence of a function. Thus, this method always results in a solution. It is, essentially, a multi-step closed form calculation. Depending on the number of cells in the truss, the approximate equal chord length algorithm might prove faster and will probably never have convergence problems. For the situations tested, both of these methods produced valid results without convergence problems.

One other point which must be emphasized is that these methods are in no way limited to the two geometries discussed. As an example of another possible geometry, Fig. 7.1 schematically illustrates a seven cell octahe-

dral/octahedral VGT which is being constructed at Virginia Polytechnic Institute and State University.

There remain many potentially fruitful areas of research concerning these VGT manipulators. One such area, which directly relates to this thesis, is the continued development of improved computational methods for the control of these devices. Perhaps the type of one dimensional optimal distribution of partition points alluded to in Chapter 5 will produce better results. Or, perhaps an entirely new way of viewing the shape curves would be helpful. The use of intrinsic curve properties, such as arc length, curvature, and torsion, might provide this alternative viewpoint. Other topics of interest include the development of real-time solids modelling for automated obstacle avoidance and real-time force analysis. Both of these tasks would require developing very fast, customized software which only needs to consider straight, two-force members. Finally, much attention must be given to the physical design of these manipulators.

In short, this thesis has illustrated that it is possible to control long-chain, very high degree-of-freedom manipulators with shape control algorithms.

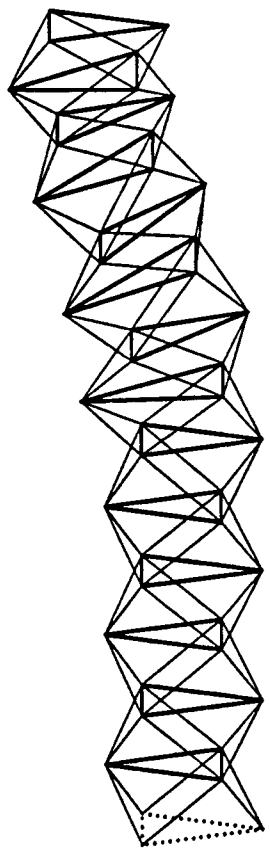

Figure 7.1: Seven cell octahedral/octahedral VGT manipulator

Bibliography

[1] Mabie, H., Reinholtz, C.F., **Mechanisms and Dynamics of Machinery,** J. Wiley and Sons, N.Y.,N.Y.,1987.

[2] Hollerbach, "Optimum Kinematic Design for a Seven Degree of Freedom Manipulator," Robotics Research - The Second International Symposium, MIT press, Cambridge, Mass., pp. 216-222.

[3] Stewart, D., "A Platform with Six Degrees of Freedom," *Proceedings of the Institute of Mechanical Engineers,* Vol. 180, Part 1, No. 15, 1965-1966, pp. 371-386.

[4] Tesar, D., Cox, D.J., "The Dynamic Modeling and Command Signal Formulation for Parallel Multi-Parameter Robotic Devices," publication of the Center for Intelligent Machines and Robotics, College of Engineering, University of Florida, Gainsville, Florida, 1981.

[5] Hunt, K.H., "Structural Kinematics of In-Parallel-Actuated Robot Arms," *Transactions of the ASME Journal of Mechanisms, Transmissions, and Automation in Design,* Vol. 105, 1983, pp. 705-712.

[6] Yang, D.C.H., Lee, T.W., "Feasibility Study of a Platform Type of Robotic Manipulator from a Kinematic Viewpoint," *Transactions of the ASME Journal of Mechanisms, Transmissions, and Automation in Design,* Vol. 106, June 1984, pp. 191-198.

[7] Fichter, E.F., " A Stewart Platform-Based Manipulator: General Theory and Practical Construction," *The International Journal of Robotic Research,* Vol. 5, No. 2, Summer 1985.

[8] Sklar, M., Tesar, D., "Dynamic Analysis of Hybrid Serial Manipulator Systems Containing Parallel Modules," *Transactions of the ASME Journal of Mechanisms, Transmissions, and Automation in Design,* Vol. 105, No. 2, June 1988, pp. 109-115.

[9] Lee, K.M., Chao, A., "On the Analysis of a Three-Degrees-of-Freedom Manipulator," *The International Journal of Robotics and Automation,* Vol. 3, No. 2, 1988, pp. 90-96.

[10] Sugimoto, K., "Kinematic and Dynamic Analysis of Parallel Manipulators by Means of Motor Algebra," *Transactions of the ASME Journal of Mechanisms, Transmissions, and Automation in Design,* Vol. 109, No. 1, March 1987, pp. 3-7.

[11] Sugimoto, K., "Computational Scheme for Dynamic Analysis of Parallel Manipulators," *Transactions of the ASME Journal of Mechanisms,*

Transmissions, and Automation in Design, Vol. 111, No. 1, March 1989, pp. 29-33.

[12] Dorsey, J.T., "Vibration Characteristics of a Deployable Controllable-Geometry Truss Boom," NASA Technical Paper 2160, June 1983.

[13] Rhodes, M.D., Mikulas Jr., M.M., "Deployable Controllable Truss Beam," NASA Technical Memorandum 86366, June 1985.

[14] Sincarsin, W.G., "Trussarm Conceptual Design," Dynacon Report 28-611/0402, April 1987.

[15] Miura, K., Furuya, H., Suzuki, K., "Variable Geometry Truss and its Application to Deployable Truss and Space Crane Arm," 35^{th} Congress of the International Astronautical Federation, Lausanne, Switzerland, Oct. 1984.

[16] Miura, K., Furuya, H., "An Adaptive Structure Concept for Future Space Applications," 36^{th} Congress of the International Astronautical Federation, Stockholm, Sweden, Oct. 1985.

[17] Sincarsin, W.G., Hughes, P.C., "Trussarm Candidate Geometries," Dynacon Report 28-611/0401, April 1987.

[18] Jain, S., Kramer, S., "Design of a Variable Geometry Robot Based on an n-Celled Tetrahedron-Tetrahedron Truss," 1988 ASME Design

Technology Conference Proceedings, Kissimmee, FL, DE Vol. 15-3, pp. 119-123.

[19] Nayfeh, A. H., "Kinematics of Foldable Discrete Space Cranes," publication of the Department of Aerospace Engineering and Engineering Mechanics, University of Cincinnati, Cincinnati, OH, 1985.

[20] Padmanabhan, B., "Design of a Robotic Manipulator using Variable Geometry Trusses as Joints," Masters Thesis, Virginia Polytechnic Institute and State University, Blacksburg, VA, 1989.

[21] Clark, W.W., "A Planar Comparison of Actuators for Vibration Control of Flexible Structures," 1989 Structures, Structural Dynamics and Materials Conference Proceedings, Mobile, AL, April 1989, pp. 1495-1503.

[22] Natori, M., Iwasaki, K., Kuwao, F., "Adaptive Planar Truss Structures and Their Vibration Characteristics," 1987 Structures, Structural Dynamics and Materials Conference Proceedings, Monterey, CA, April 1987, pp. 143-151.

[23] Griffis, M., Duffy, J., "A Forward Displacement Analysis od a Class of Stewart Platforms," publication of the Center for Intelligent Machines and Robotics, University of Florida, Gainesville, FL, Jan. 1989.

[24] Craig, J.J., **Introduction to Robotics, Mechanics and Control**, Addison-Wesley, Reading Mass., 1986.

[25] Cheney, W., Kincaid, D., **Numerical Mathematics and Computing**, Brooks/Cole, Monterey, CA, 1985.

[26] Lossing,D.L., Eshleman, A.L., "Planning a Common Database for Engineering and Manufacturing," SHARE XLIII Conference, Chicago, Ill., Aug. 1974.

[27] Mortenson, M.E., **Geometric Modeling**, J. Wiley and Sons, N.Y., N.Y., 1985.

[28] Nikravesh, P.E., **Computer Aided Analysis of Mechanical Systems**, Prentice Hall, Englewood Cliffs, N.J., 1988.

Appendix A

Derivation of Cubic Coefficient Matrix

As indicated in the main body of the text, a planar or spatial cubic curve can be represented by the following equation:

$$\vec{C}(u) = \vec{A}_0 + \vec{A}_1 u + \vec{A}_2 u^2 + \vec{A}_3 u^3.$$

Differentiating this with respect to u yields,

$$\vec{C}'(u) = \vec{A}_1 + 2\vec{A}_2 u + 3\vec{A}_3 u^2.$$

If these two equations are evaluated for the beginning and end of the curve, at $u = 0$ and $u = 1$, the four following simultaneous equations result.

$$\begin{aligned}
\vec{C}(0) &= \vec{A}_0 \\
\vec{C}(1) &= \vec{A}_0 + \vec{A}_1 + \vec{A}_2 + \vec{A}_3 \\
\vec{C}'(0) &= \vec{A}_1 \\
\vec{C}'(1) &= \vec{A}_1 + 2\vec{A}_2 + 3\vec{A}_3
\end{aligned}$$

These equations can be represented in matrix form as;

$$\begin{bmatrix} \vec{C}(0) \\ \vec{C}(1) \\ \vec{C}'(0) \\ \vec{C}'(1) \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 \\ 0 & 1 & 0 & 0 \\ 0 & 1 & 2 & 3 \end{bmatrix} \begin{bmatrix} \vec{A}_0 \\ \vec{A}_1 \\ \vec{A}_2 \\ \vec{A}_3 \end{bmatrix}$$

Now the cubic coeffients $\vec{A}_0, \vec{A}_1, \vec{A}_2$, and $\vec{A}_3$ can be evaluated in terms of the known parameters $\vec{C}(0), \vec{C}(1), \vec{C}'(0)$, and $\vec{C}'(1)$.

$$\begin{bmatrix} \vec{A}_0 \\ \vec{A}_1 \\ \vec{A}_2 \\ \vec{A}_3 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 \\ 0 & 1 & 0 & 0 \\ 0 & 1 & 2 & 3 \end{bmatrix}^{-1} \begin{bmatrix} \vec{C}(0) \\ \vec{C}(1) \\ \vec{C}'(0) \\ \vec{C}'(1) \end{bmatrix};$$

which reduces to,

$$\begin{bmatrix} \vec{A}_0 \\ \vec{A}_1 \\ \vec{A}_2 \\ \vec{A}_3 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ -3 & 3 & -2 & -1 \\ 2 & -2 & 1 & 1 \end{bmatrix} \begin{bmatrix} \vec{C}(0) \\ \vec{C}(1) \\ \vec{C}'(0) \\ \vec{C}'(1) \end{bmatrix}.$$

Thus, any point on the curve, within the interval $u[0,1]$ can be represented as follows:

$$\vec{C}(u) = \begin{bmatrix} 1 & u & u^2 & u^3 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ -3 & 3 & -2 & -1 \\ 2 & -2 & 1 & 1 \end{bmatrix} \begin{bmatrix} \vec{C}(0) \\ \vec{C}(1) \\ \vec{C}'(0) \\ \vec{C}'(1) \end{bmatrix}.$$

The vita has been removed from the scanned document