An Automated Methodology for Dynamic Force Analysis of Adaptive Spatial Trusses

by

David Terrell Lacy

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:

H. H. Robertshaw, Chairman

H. H. Cudney

R. L. West

May, 1991
Blacksburg, Virginia

An Automated Methodology for Dynamic Force Analysis of Adaptive Spatial Trusses

by

David T. Lacy

Committee Chairman: Harry H. Robertshaw

Mechanical Engineering

Abstract

The purpose of this thesis is to develop a formal methodology for determining the loads occurring in the members of an adaptive truss due to both gravity and acceleration. This force analysis can be used as the basis for a design code which will provide truss member dimensions and actuator characteristics. Three different truss structures are considered. The first is a planar, triangular truss consisting of one actuated member and two fixed length members. The second structure is a spatial, double-octahedral truss with three active members and eighteen fixed length members. The third structure is a truss consisting of several double-octahedral bays connected together as a chain. For each structure, the active link motion is first simulated and the position, velocity, and acceleration history of each of the member connecting points is calculated. The dynamic equations of motion for each member are developed and combined to form a system of equations describing the motion of the entire truss. These equations are then solved to find the forces occurring at each node. Once the forces are determined, the internal forces in each member can be found, and the resulting stresses are calculated. The members are also checked for buckling using Euler buckling theory. The stress calculations are checked against experimental values and show good agreement for both static only and static and dynamic loading.

Acknowledgements

I would like to thank Dr. Harry Robertshaw for his support and encouragement during my time at Virginia Tech. When times were tough, he always knew what to say to keep me going. Personally and professionally, thank you for helping me through everything. Thanks also to my other committee members, Dr. Robert West and Dr. Harley Cudney, for their help and support.

I also want to thank everyone in the office for their help; Will, Buddy, Robert, Babu, and Paul. Whenever I had any questions or needed to bounce ideas off of somebody, they were happy to help – I really appreciate it. A special note of thanks goes out to Tom for getting the kinematics done and for helping me to get my ideas straight, but mostly for being a good friend.

Most of all, I'd like to thank my wife, Joyce, for putting up with all of the long nights and hard work that have gone into this thesis.

Contents

List of Figures

List of Tables

Nomenclature

ϕ	triangular truss follower angle
θ	triangular truss active member angle
L_1	length of triangular truss follower
L_2	length of triangular truss leadscrew tube
L_{3min}	minimum length of triangular truss active base
L_{3max}	maximum length of triangular truss active base
L_4	length of triangular truss ground link
x	displacement of active member
T	total simulation time
ω_n	active member displacement model natural frequency
ζ	active member displacement model damping ratio
ts	2 percent settling time
m_i	mass of the i^{th} truss member
g	acceleration of gravity
F_{x_i}	i^{th} force in the Newtonian $\hat{n}_1$ direction
F_{y_i}	i^{th} force in the Newtonian $\hat{n}_2$ direction
F_{z_i}	i^{th} force in the Newtonian $\hat{n}_3$ direction
T_{int}	internal tensile force
V_{int}	internal shear force
M_{int}	internal bending moment
σ_T	internal tensile stress
σ_M	internal bending moment stress
τ	internal shear stress
I	moment of inertia
Q	first moment of area
E	modulus of elasticity
A	cross-sectional area
G_{ls}	leadscrew gain
G_{gh}	gearhead gain
θ_M	motor shaft angular displacement

T_p	motor torque to move load
T_a	motor torque to overcome armature inertia
T_m	total motor torque
J_m	motor armature inertia
d_m	leadscrew mean diameter
F	linear force applied by the leadscrew
μ	coefficient of friction
V_m	motor voltage
L_a	motor inductance
R_a	motor resistance
I_a	motor current
K_b	motor back emf coefficient
K_t	motor torque constant
Mi	i^{th} member of the spatial truss
Ni	i^{th} node of the spatial truss
N_i	x, y, z coordinates of the i^{th} node of the spatial truss
V_i	x, y, z velocities of the i^{th} node of the spatial truss
A_i	x, y, z accelerations of the i^{th} node of the spatial truss
$\hat{b}$	body-fixed coordinate system
$\hat{n}$	Newtonian reference coordinate system
$\mathbf{C^i}$	cosine transformation matrix for ith member
ω	angular velocity
α	angular acceleration
a_i	linear acceleration of the i^{th} point
SM	combined truss system matrix
$\mu\epsilon$	microstrain
bn	coordinate system attached to n^{th} bay
P_{base_n}	location of the origin of bn coordinate system
V_{base_n}	velocity of the origin of bn coordinate system
A_{base_n}	acceleration of the origin of bn coordinate system
$P_{i/base}$	location of the i^{th} node in the bn coordinate system
$V_{i/base}$	velocity of the i^{th} node in the bn coordinate system
$A_{i/base}$	acceleration of the i^{th} node in the bn coordinate system

Chapter 1

Introduction

1.1 Introduction

In recent years, much attention has been given to developing high-strength, low-weight supports and manipulators for use in current and future space applications. One device that appears to have much promise in these areas is the Adaptive Truss, or Variable-Geometry Truss (VGT). VGTs are trusses that contain some members that are capable of changing their length. The change in length of the active members results in a change in the geometry of the truss. The change in geometry causes motion of the top plane or of a manipulator attached to the top plane. The characteristics of this motion depend on the type of truss and the location of the active members, with each active member resulting in one degree of freedom for the truss.

VGTs are comprised of basic building blocks, called unit cells, and may be either planar (2-D) or spatial (3-D). The basic planar unit cell is the triangle. Reinholtz (15) defines several spatial unit cells, including the tetrahedron, the octahedron, and the decahedron. A bay is the simplest repeating structure in a truss, and may be made up of a single unit cell or a combination of unit cells. This thesis will examine two types of VGTs. The first is a planar truss comprised of one triangular

unit cell with one active link. The triangular truss analysis is performed to develop the force analysis method using a simple structure. The second truss is a spatial truss comprised of two octahedron cells (called a double-octahedral) with the three members of the midplane activated. A prototype double-octahedral truss available at VPI & SU is used to verify the accuracy of the model. We will also extend the analysis of the single bay, double-octahedral truss to a chain of double-octahedral bays.

The varying geometry capability of VGTs allows them to be used for several different types of tasks. Four general tasks have been defined for spatial VGTs. These are, in order of complexity:

1. Gimbal (Pointing) - Rotation of the top plane about both the x and y axes (2 DOF)

2. Positioning - Placing the top plane at some x, y, and z location in space (3 DOF)

3. Reflecting - Placing the top plane at a point in space and aiming it at another point (4 DOF)

4. Docking - Orienting the top plane in both x, y, and z space and rotating it about the three axes (6 DOF)

Within these general tasks are many different applications. The VGT can be used as a high-strength joint on a space crane, as an active vibration isolator, or as a multi-DOF walking device. In this study, only the gimbal task is analyzed. However, the analysis detailed in this thesis can be used for any of the above tasks as

long as the kinematics of the motion are known.

For space applications, the high cost of getting materials into orbit makes it imperative that the VGT be as light as possible. Therefore, we need to design the members only to withstand the expected forces occurring during its performance. We simply cannot use large factors of safety. The large range of motion that the VGT can undergo gives rise to large nonlinearities in the kinematics of the structure. These nonlinearities make a detailed dynamic force analysis of the structure difficult. The static case can be solved using finite element models, however, there are no readily available large angle dynamic analysis packages. The result is that a dynamic force analysis of these trusses has not been done in the past. The design of the prototype units that are being used for ground based research is done using rough approximations and "educated guesses" by the engineers.

The original intent of this work was to model the truss members as flexible beams. A developmental program called LATDYN (Large Angle Transient DYNamics) was to be used to create a finite-element based model with flexible members. However, the program did not work up to expectations, and LATDYN development was cancelled by NASA. A flexible body dynamic analysis was then attempted, but was not successful, due in part to the lack of flexible kinematic solutions. The scope of the work was then modified to include only rigid body analysis of the truss members. The kinematic analysis assumes that the members do not deform during the motion. Since the members are assumed to be rigid, no stiffness effects are included in the analysis. The inertial loading due to the mass of the members is included in the analysis. The mass is assumed to be evenly distributed throughout each member.

The objective of this thesis is to present a technique for performing a formal, detailed force analysis for all of the members of a VGT while undergoing a predetermined motion. Once the loading history of each member is known, the member dimensions can be specified so that it does not fail. As stated above, this analysis is very nonlinear and cannot be done in closed form. Therefore, an approximate technique has been developed to solve the problem.

The approximate solution is generated by discretizing the motion of the truss. At each timestep, the instantaneous position, velocity, and acceleration of the node points, as well as the angular velocities and accelerations of the members, is calculated. Using these approximate values, we can write the equations of motion (EOM) for each member. The member EOMs are combined to form a system of equations describing the entire truss. Known constraints are applied to the structure, and then the joint forces on each member are calculated. Gravitational forces are also applied to the model, so the static forces are included in the calculated joint forces. Once the joint forces are found, the loading on each member can be determined. This process is repeated at every timestep within the motion. The forces applied by the actuators onto the truss are calculated during this analysis.

1.2 Literature Review

Much has been written on the applicability of adaptive trusses for a wide variety of tasks. Rhodes (17) proposed using adaptive trusses for several tasks related to space applications. These include a serpentine robot arm that can place an end effector in

hard-to-reach places, as high-strength joints in a space crane, as vibration absorbers to isolate devices (such as antennae) with high position accuracy requirements from disturbances on the supporting structure, and as berthing devices which can dock with approaching space craft. Lovejoy (9) used a planar truss to control the vibrations in an attached flexible beam that was perpendicular to the gravitational field. Clark (5) uses a floating, free-free planar truss to suppress the vibrations in two flexible beams connected at each end of the truss. This work was extended by Robertshaw (18), Kung (7), and Wynn (26) to the three dimensional case where the flexible beam is hanging downward in the gravity field. Warrington (23) controls the vibrations of an attached beam that is positioned upward in the gravity field. He also controls beam vibration in a beam that is positioned upward and at an angle to the gravity field (24). This induces nonlinear gravitational effects in the motion of the flexible beam. Reinholtz (16) presents spatial VGTs as high dexterity robots. Stulce (20) examines the use of spatial VGTs as actuators in a walking device. In all of these studies, the VGT is treated as an external actuator and the forces within it are not studied. This thesis seeks to calculate the magnitude of the internal forces generated within the VGT while undergoing a specific task.

Recent developments in the field of spatial truss kinematics were necessary before the force analysis could be performed. Padmanabhan (13) developed a closed form solution for finding the active link lengths for a given orientation of both a double-octahedral and a quadruple-octahedral adaptive truss under a large angle gimbal rotation. The work was extended by Warrington (24,25) to find the orientation of the double-octahedral truss given a set of active link lengths. This method has no closed-form solution, but must be performed iteratively. In reality, when a VGT

performs a large-angle motion, the displacements of the active links are the controlled variables; the motion of the top plane is caused by the link displacements. Therefore, the kinematics relating the position, velocity, and acceleration of the active links to those of the top plane nodes must be found in order to develop a control algorithm for the truss motion. This kinematic relationship is also needed so that the accelerations of the truss members can be found and used for the dynamic force analysis.

Utku (21) develops a discretized method of analyzing the control forces required to perform a slewing maneuver of a planar VGT used as a crane. The crane has ten total members, two of which are variable length. The crane undergoes large angle motion and a method of developing the nodal forces based on the relative orientation of the members is found. The slewing maneuver is performed at a slow speed, so dynamic loads are not generated within the members. He develops the kinematics of the motion assuming that all of the members are rigid. He uses the stiffness matrix of the structure to find the deformation of the members at each time step (due to axial loading only) and then modifies his control forces based on the deformation. In contrast to the work in this thesis, Utku's analysis does not include the forces due to the inertial loading or mass of the members, only those from the static loads at each point in time.

Orin (12) performs a dynamic force analysis on a multi-legged walking device. He uses Newton's laws of motion to develop the equations of motion of each member. By writing 3-D coordinate transformation matrices for each member, he combines the equations of motion for each member into a total system of equations. The

position, velocity, and acceleration histories of each member are then calculated and the system is solved to find the unknown dynamic forces. There are several major differences between Orin's work and that undertaken in this thesis. Orin's device is not a truss structure, but an open chain linkage of members. Therefore, there are only two members connected at each joint. In addition, the joints can transmit torques to the members. However, the general method of analysis is similar to that used in this work.

Gupta (6) presented a method for formulating and solving the Newton-Euler equations of motion for a closed-loop series of rigid bodies. The kinematics and dynamic equations are solved used numerical integration techniques. The equations of motion of the members are developed using Newton's laws of motion. While the mechanisms he studied did not involve changing length members, the method of analysis is very similar to that presented in this paper.

Chalhoub (4) presents an analysis of a leadscrew-driven flexible robot arm. He includes the leadscrew dynamics in the development of the equations of motion of the robot arm. Friction forces within the leadscrew are included in the relationship between the motor torque and the linear applied force. The friction analysis is based on Shigley (19) and is the same as that presented in this thesis. One particular piece of information that would have been useful, the coefficient of friction that Chalhoub used for his analysis, was not provided in the paper. Brennan (2) has also developed a force model of a triangular truss based on a power-screw type lead screw. The force model is used to perform a "contention control" study of two planar, triangular trusses. The followers of each truss are connected together by a thin link.

The literature search did not provide any examples of analysis of the internal dynamic forces generated in VGTs during motion. All development of the equations of motion for the VGTs are done using LaGrange equations or Kanes equations, both of which operate by eliminating the interconnecting forces between members. Therefore, this analysis should provide some unique information on the interaction between the motion of the truss and the internal forces generated within its members.

1.3 Outline of Thesis

This thesis is presented in three major sections. In the first section, a simple, triangular truss, with one active member, is analyzed. This analysis was done to verify that the solution technique discussed above will work. The relationship between the active link length and the location of the nodes, while nonlinear, is relatively simple compared to the three dimensional case. The triangular truss model is developed in Chapter 2. Once the model is developed, a simulation program is used to implement the model. The simulation program and results are presented in Chapter 3.

In Chapter 4, the model for the spatial adaptive truss is developed. Here, a single double-octahedral bay is discussed. The kinematics of the double-octahedral bay are explained and the force analysis is developed. The simulation routine for the spatial case is presented in Chapter 5. Verification of the analytical results is presented for both the static and dynamic cases.

In Chapter 6, a discussion of extending the analysis to include multiple double-

octahedral bays is presented. A means of obtaining the velocity and acceleration information for each bay, based on a chain of coordinate transformations back to the fixed reference frame, is developed. Finally, in Chapter 7, the conclusions of the work and recommendations for future research are discussed.

Chapter 2

Development of Triangular Truss Model

2.1 Description of System

The planar, triangular truss is the simplest VGT configuration. The configuration used for this study is shown in Fig. 2.1. The truss consists of a fixed-length follower, a fixed ground link, one active member, and a lumped mass attached to the node. The active member is modelled as two links; the base link contains the motor, gearhead, and leadscrew and is referred to as the active base. The second link of the active member is called the leadscrew tube. It is a hollow tube which meets the active base at the leadscrew. The leadscrew tube moves along the longitudinal axis of the active member to create the motion of the truss. The follower and the leadscrew tube are attached through a revolute joint at the node, labeled point B. The lumped mass is also attached to the truss through a revolute joint at point B.

The task assigned to this truss is to move the follower through some arbitrary angle, from ϕ_0 to ϕ_1. Knowing the starting and ending angles, the starting and ending length of the active member can be determined using the law of cosines, as:

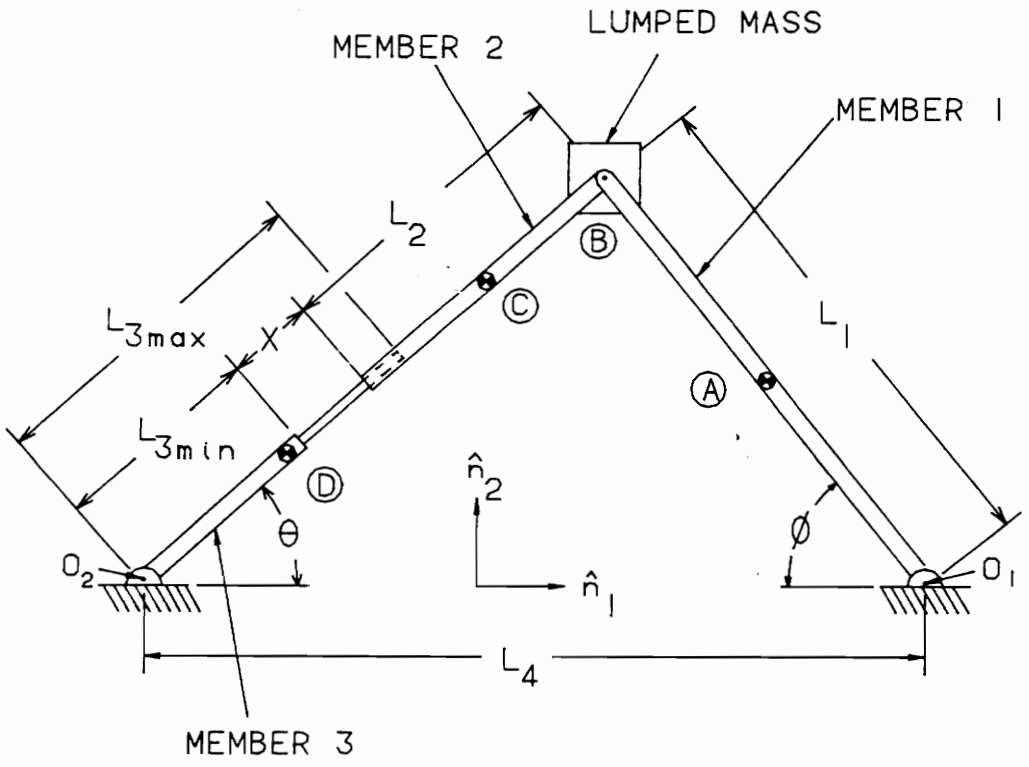

Figure 2.1: Triangular Truss Model

$$\phi_0 = \arccos\left(\frac{L_1^2 + L_4^2 - (L_{3min} + x_0 + L_3)^2}{2L_4L_1}\right) \tag{2.1}$$

$$\phi_{end} = \arccos\left(\frac{L_1^2 + L_4^2 - (L_{3min} + x_{end} + L_3)^2}{2L_4L_1}\right) \tag{2.2}$$

As shown in Fig. 2.1, the active member has a minimum length of L_{3min}, and a maximum extended length of L_{3max}. The length of the active member at any point during the motion is $(L_{3min} + x)$, where x is length that the active member has extended.

In order to find the EOMs for the members of the truss, we must know the accelerations of all of the centers of gravity as well as the angular velocities and accelerations, $\dot{\phi}$, $\dot{\theta}$, $\ddot{\phi}$, and $\ddot{\theta}$. However, all of these velocities and accelerations depend on the linear velocity and acceleration of the active member. Since the relationship between these values is nonlinear, we use a simulation where we discretize the motion of the active link and then solve for the position, velocity, and acceleration of the members at each time step. The first step in this analysis is to determine the position, velocity, and acceleration history of the active member.

2.2 Active Member Simulation

The active member is assumed to apply a pure displacement input to the truss. This means that the position, velocity, and acceleration of the active member does not depend of the force exerted on it by the truss. The active link motion can be modelled as a either a second or third-order system. There are several advantages

to using a higher order model, including lower accelerations at the beginning of the motion and the fact that the velocity and acceleration are states of the model. We first develop a model of the displacement using feedback control theory for a third-order system.

A state feedback model of the active member motion was developed, where the displacement, velocity, and acceleration of the member were used as feedback signals. The feedback gains were calculated to place the poles of the system to meet the specifications described below. We have to specify several values to use this model, including the natural frequency (ω_n), the damping ratio (ζ), and the total time of the motion (T).For a third-order model, we need to place the three poles of the characteristic equation. We will specify a fast real root, and a slower, complex conjugate pair of roots. Since this analysis is done only for a simulation, we are not constrained by physical parameters in our choice of these parameters. The rationale for the choice of these values is as follows.

We want to have a highly damped system to avoid excessive overshoot, therefore, we chose a damping ratio of 0.8. In addition, we wanted the response to a step input to reach a steady state value within 90 percent of the total simulation time ($ts = 0.9T$). The lowest resonant frequency was calculated so that the response would be within 2 percent of the final value after the settling time ($ts = 0.9T = \frac{4}{\zeta\omega_n}$). The real root was calculated to be twice the natural frequency. The total time of the simulation can be selected when the simulation is run. The input to the system, U, is calculated as the difference between the starting active member length and the final length.

Once these values are determined, a MATLAB simulation is performed to find the response of the active length to the input. This simulation develops the state matrices for the model, converts them to a discrete form (based on the number of time steps and total time of the simulation), and steps through the time span. The output from the simulation is the displacement, velocity, and acceleration of the active member.

2.3 Truss Acceleration Analysis

From the above simulation, we know the length, velocity, and acceleration of the active member at each time step. These values are necessary to calculate each member's center-of-gravity acceleration and the angular acceleration. The acceleration of the center-of-gravity of a member can be found by describing a vector from a fixed reference point to the center of gravity and differentiating twice with respect to time. The vector must be described in fixed reference frame coordinates. For the follower, an acceptable fixed reference point is point O_1. The center-of-gravity of the follower is labeled point A. The acceleration of point A is given as

$$\vec{O_1A} = -L_1 cos\phi\ \hat{n}_1 + L_1 sin\phi\ \hat{n}_2 \tag{2.3}$$

$$\dot{\vec{O_1A}} = L_1\dot{\phi} sin\phi\ \hat{n}_1 + L_1\dot{\phi} cos\phi\ \hat{n}_2 \tag{2.4}$$

$$\begin{aligned}\ddot{\vec{O_1A}} &= [L_1\ddot{\phi} sin\phi + L_1\dot{\phi}^2 cos\phi]\ \hat{n}_1 \\ &\quad + [L_1\ddot{\phi} cos\phi - L_1\dot{\phi}^2 sin\phi]\ \hat{n}_2\end{aligned} \tag{2.5}$$

For the leadscrew tube, the analysis becomes more complex, since the linear accel-

eration of the actuator enters into the equation. A suitable reference point for the vector to the leadscrew tube center-of-gravity (point C) is point O_2. The acceleration of point C is calculated as

$$\vec{O_2C} = [L_{3min} + x + \frac{L_2}{2}]cos\theta\ \hat{n}_1 + [L_{3min} + x + \frac{L_2}{2}]sin\theta\ \hat{n}_2 \tag{2.6}$$

$$\dot{\vec{O_2C}} = [\dot{x}cos\theta\ - (L_{3min} + x + \frac{L_2}{2})\dot{\theta}sin\theta]\ \hat{n}_1 + [\dot{x}sin\theta\ + (L_{3min} + x + \frac{L_2}{2})\dot{\theta}cos\theta]\ \hat{n}_2 \tag{2.7}$$

$$\begin{aligned}\ddot{\vec{O_2C}} = {} & [\ddot{x}cos\theta - 2\dot{x}\dot{\theta}sin\theta - (L_{3min} + x + \frac{L_2}{2})\ddot{\theta}sin\theta - (L_{3min} + x + \frac{L_2}{2})\dot{\theta}^2cos\theta]\hat{n}_1 \\ & + [\ddot{x}sin\theta\ + 2\dot{x}\dot{\theta}cos\theta\ + (L_{3min} + x + \frac{L_2}{2})\ddot{\theta}cos\theta \\ & - (L_{3min} + x + \frac{L_2}{2})\dot{\theta}^2sin\theta]\ \hat{n}_2\end{aligned} \tag{2.8}$$

The fixed length members of the truss are all assumed to be constant density and of constant cross-section, therefore, the center-of-gravity of each is half of the length. The accelerations for the centers of gravity of the active base and for the lumped mass are found using the same analysis.

Next, we have to calculate the angular velocities and angular accelerations. Since we know the length of the active member at each time step, we can find the value of the angles, ϕ and θ, by applying the law of cosines at each step. We then calculate the angular velocity as the difference in two consecutive angles divided by the time step. To find $\dot{\phi}$ at the n^{th} timestep, we use

$$\dot{\phi}_n = \frac{\phi_n\ -\ \phi_{n-1}}{\Delta t}$$

We use the same approach to calculate the angular acceleration where

$$\ddot{\phi}_n = \frac{\dot{\phi}_n - \dot{\phi}_{n-1}}{\Delta t}$$

The values for x, $\dot{x}$, and $\ddot{x}$ are the linear position, velocity, and acceleration of the active member, and are calculated during the simulation described in Section 2.2. Once we know the linear and angular accelerations of all of the members and nodes in the truss, we can develop the equations of motion.

2.4 Derivation of the Truss Equations of Motion

We develop the equations of motion for each member of the truss separately and then combine them to form a system of equations describing the motion of the entire truss. Since the active member is modelled as two links, we will have two sets of EOMs, one for the active base and one for the leadscrew tube. The first step in developing the EOMs for a member is to draw the free body diagram (FBD). The FBDs for each element in the truss are shown in Fig. 2.2. There are four FBDs, one each for the follower, the active base, the lead screw tube, and the lumped mass. The forces applied to each element are shown in the FBDs.

To apply Newton's Laws of Motion, we sum the forces in the $\hat{n}_1$ and $\hat{n}_2$ directions and sum the torques about a convenient point. We will carry out the analysis in detail for the active base; the analysis for the other members is similar.

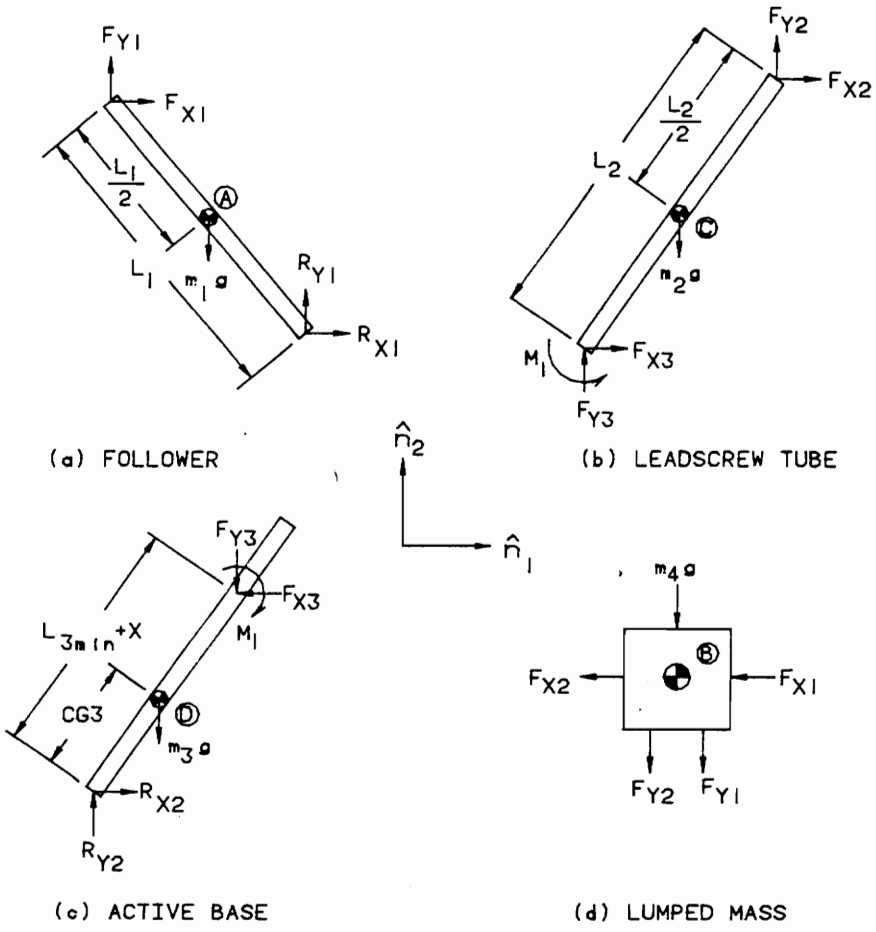

Figure 2.2: Free Body Diagrams of Triangular Truss Members

2.4.1 Active Base Equations of Motion

The active base FBD is shown in Fig. 2.2(c). There are two reaction forces from the ground applied at point O_2, two forces and a moment from the lead screw applied at the point of contact, $(L_{3min} + x)$, and the weight of the member acting at the center-of-gravity. The EOMS are found to be

$$\sum F_{\hat{n}_1} \Rightarrow \quad m_3 a_{D\hat{n}_1} = -F_{x_3} + R_{x_2} \tag{2.9}$$

$$\sum F_{\hat{n}_2} \Rightarrow \quad m_3 a_{D\hat{n}_2} = -F_{y_3} + R_{y_2} - m_3 g \tag{2.10}$$

$$\sum T_{O_2} \Rightarrow \quad I_{O_2}\ddot{\theta} = F_{x_3}(L_{3min} + x)sin\theta - F_{y_3}(L_{3min} + x)cos\theta - m_3 g(CG3)cos\theta - M_1 \tag{2.11}$$

Note that the moment arm for the forces applied on the active base by the lead screw tube changes as the member extends. This is the only member for which the moment arm changes. It is convenient to arrange the EOMs in matrix form. The matrix representation for the active base is:

$$\begin{bmatrix} m_3 a_{D_{\hat{n}_1}} \\ m_3 a_{D_{\hat{n}_2}} + m_3 g \\ I_{O_1}\ddot{\theta} + m_3 g(CG3)cos\theta \end{bmatrix} =$$

$$\begin{bmatrix} -1 & 0 & -1 & 0 & 0 \\ 0 & -1 & 0 & 1 & 0 \\ (L_{3min} + x)sin\theta & -(L_{3min} + x)cos\theta & 0 & 0 & -1 \end{bmatrix} \begin{bmatrix} F_{x_3} \\ F_{y_3} \\ R_{x_2} \\ R_{y_2} \\ M_1 \end{bmatrix} \tag{2.12}$$

There are three EOMs and five unknown forces, therefore, we get a 3x5 matrix. These equations cannot be solved by themselves to find the forces; it is necessary to combine this matrix with those for each of the other members. Then we will have a full rank matrix of 11 equations and 11 unknowns which can be solved.

2.4.2 Follower, Lead Screw Tube, and Lump Mass EOMs

The above analysis is completed for each member of the truss. Newton's laws of motion result in three equations for each member, except for the lumped mass which, having no inertia, will have no torque equation. We assume unique forces at each pinned connection and a set of forces at the cantilevered connection between the active base and the leadscrew tube. The FBDs of each of these members are shown in Fig. 2.2(a,b,d).

2.4.3 Combined System Matrix

Once each member's individual system matrix is developed, we form the combined system matrix for the entire truss. There are a total of 11 equations and 11 unknown forces. The combined system equations are given as

$$\{M\ddot{X}\} = [SM]\ \{F\} \tag{2.13}$$

where:

$$\left\{M\ddot{X}\right\} = \left\{\begin{array}{c} m_1 a_{A\hat{n}_1} \\ m_1 a_{A\hat{n}_2} + m_1 g \\ I_{O2}\ddot{\phi} + m_1 g \frac{L_1}{2} cos\phi \\ m_2 a_{C\hat{n}_1} \\ m_2 a_{C\hat{n}_2} + m_2 g \\ I_c \ddot{\theta} \\ m_3 a_{D\hat{n}_1} \\ m_3 a_{D\hat{n}_2} + m_3 g \\ I_{O1}\ddot{\theta} + m_3 g CG3 cos\theta \\ m_4 a_{B\hat{n}_1} \\ m_4 a_{B\hat{n}_2} + m_4 g \end{array}\right\}$$

$$\{F\} = \left\{\begin{array}{c} F_{x_1} \\ F_{x_2} \\ F_{x_3} \\ F_{y_1} \\ F_{y_2} \\ F_{y_3} \\ R_{x_1} \\ R_{x_2} \\ R_{y_1} \\ R_{y_2} \\ M_1 \end{array}\right\}$$

$$[SM] = \left[\begin{array}{cccccccccccc} 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ L_1 s\phi & 0 & 0 & L_1 c\phi & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & -\frac{L_2}{2} s\theta & \frac{L_2}{2} s\theta & 0 & \frac{L_2}{2} c\theta & -\frac{L_2}{2} c\theta & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & -1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & (L_{3min} + x)s\theta & 0 & 0 & -(L_{3min} + x)c\theta & 0 & 0 & 0 & 0 & -1 \\ -1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \end{array}\right]$$

We should note here the effect of the lumped mass. If there were no lumped mass,

we would have only 9 equations. We would then have to find two more equations to allow us to solve for the forces. The equations that we would need are the constraint equations at point B, the node connecting the leadscrew tube and the follower. The node, by itself, is infinitesimally small and massless. Therefore, the sum of the forces applied at the node must be zero. This allows us to write the additional constraint equations for the node, i.e., $\sum Forces_{\hat{n}_1} = 0$ and $\sum Forces_{\hat{n}_2} = 0$. Solving these equations results in the fact that, for no lumped mass, $F_{x_1} = -F_{x_2}$ and $F_{y_1} = -F_{y_2}$.

2.5 Solution of the Equations of Motion

Once we have the system equations of motion, we can solve for the unknown forces. The truss is statically determinate, that is, there are only as many constraints as are required to fully analyze the structure. Therefore, we can find the forces by multiplying the inverse of the system matrix by the inertial force vector.

$$\{F\} = [SM]^{-1}\{M\ddot{X}\} \tag{2.14}$$

This calculation is performed at every time step during the simulation. The history of the nodal forces through the entire motion is then known.

2.6 Internal Forces

Once the nodal forces are known, we need to find the internal forces generated in the members. Since all of the members are assumed to be rigid bodies during the dynamic analysis, the internal forces must be in equilibrium. By calculating the values of the internal forces, we can determine the stresses within the individual members.

The stress information is then used to determine whether or not the member will fail.

To perform the internal force analysis, we take a section of a member and draw a FBD for it. The internal FBD for the follower is shown in Fig. 2.3. The forces located at the node (point B) are known from the overall truss analysis. The center-of-gravity of the member section is half of the length of the section. The mass of the section is calculated as

$$m_{sect} = \frac{L_{sect}}{L_{total}} m_1 \tag{2.15}$$

The angular acceleration of the section is the same as for the total member. The acceleration of the section center-of-gravity is found by defining a vector from a fixed reference point to the section center-of-gravity and differentiating it twice with respect to time. All of these values can be calculated from the simulation of the overall truss.

There are three internal forces acting to keep the member rigid. These are a tensile force (T_{int}) along the longitudinal axis of the member, a shear force (V_{int}) which is perpendicular to the member, and a bending moment (M_{int}) that keeps the member from bending. These forces are shown in the FBD.

To find the internal forces, we apply Newton's laws of motion to the section and develop the three equations of motion. The process is shown below for the follower sectioned at the midpoint. The internal equations of motion become

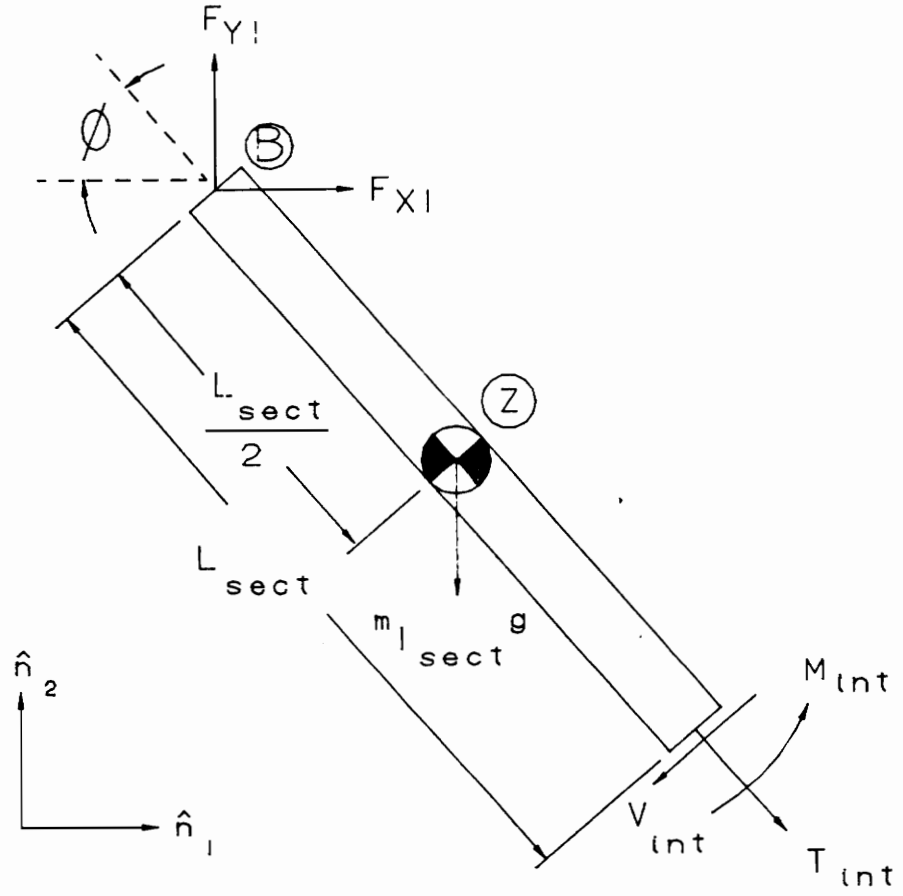

Figure 2.3: Free Body Diagram for Internal Forces of Triangular Truss Follower

$$\sum F_{\hat{n}_1} \Rightarrow \quad m_{1_{sect}} a_{z\hat{n}_1} = F_{x_1} + T_{int} cos\phi - V_{int} sin\phi \tag{2.16}$$

$$\sum F_{\hat{n}_2} \Rightarrow \quad m_{1_{sect}} a_{z\hat{n}_2} = F_{y_1} - T_{int} sin\phi - V_{int} cos\phi - m_{1_{sect}} g \tag{2.17}$$

$$\sum T_{z_{sect}} \Rightarrow \quad I_{z_{sect}} \ddot{\phi} = F_{x_1} \frac{L_{sect}}{2} sin\phi + F_{y_1} \frac{L_{sect}}{2} cos\phi + V_{int} \frac{L_{sect}}{2} - M_{int} \tag{2.18}$$

The only unknowns are the three internal forces. Thus, the equations can be solved to find the internal forces at a section for each point in time. The internal forces are calculated at seven equidistant points along the length of the member to find the largest forces. The largest forces are then used to determine the maximum stresses in the member.

2.7 Stress Analysis

The total stress at any point in the member is a combination of the stresses caused by the tension, shear, and bending moment. The stress created by the tension will be constant at all points on a given cross-section. The contributions from the shear force and the bending moment will depend on the point in the cross-section at which the stress is calculated.

The tensile stress is proportional to the tension in the member, and is calculated as

$$\sigma_T = \frac{T_{int}}{A} \tag{2.19}$$

where:

T_{int} = the tensile force in the member
A = the cross-sectional area of the member

The stress due to the bending moment depends on the distance from the neutral axis and the area moment of inertia of the cross-section. The neutral axis is the axis at the center of the cross-section where the bending moment is zero. Using these values, the bending stress is given by

$$\sigma_M = \frac{M_{int} y_1}{I} \tag{2.20}$$

where:

M_{int} = the internal bending moment
y_1 = the distance from the neutral axis
I = the area moment of inertia opposing the bending moment

The shear stress is dependent on the first moment of the cross-sectional area about the neutral axis, the width of the cross-section, and the area moment of inertia. The first moment of the area, denoted by Q, is defined as

$$Q = \int_{y_1}^{c} y dA \tag{2.21}$$

where y_1 is the point on the cross-section where the shear stress is calculated, c is the upper bound of the area across which the shear force is acting, y is the distance from the neutral axis to a point in the area, and dA is the differential area at the distance y and is equal to the width (b) at y multiplied by the differential thickness, dy. The equation describing the shear stress is

$$\tau = \frac{V_{int}Q}{Ib} \tag{2.22}$$

We can simplify this equation by substituting in the formulas for Q and I for the specific member with which we are dealing (in this case a hollow tube), resulting in

$$\tau = \frac{2V_{int}}{A}\left(1 - \frac{y_1^2}{c^2}\right) \tag{2.23}$$

Consider the cross-section of the follower shown in Fig. 2.4. There are two points at which we will calculate the stresses. At point S_1, which is a point on the neutral axis, the bending stress is a minimum and the shear stress is a maximum. At this point, y_1 is 0, therefore $\sigma_M = 0$. Also, for the shear stress, $\tau = \frac{2V}{A}$. The tensile stress is $\sigma_T = \frac{T}{A}$. We now combine these stresses using a Mohr's Circle analysis to find the maximum stress in the member. At S_1, the maximum stress is found from the equation

$$\sigma_{max} = \sqrt{\sigma_T^2 + 3\tau^2} \tag{2.24}$$

At point S_2, which is the farthest away from the neutral axis, the shear stress is zero and the bending stress is a maximum. At this point, the value of y_1 is equal to c, therefore $\tau = 0$. The bending stress is given as $\sigma_M = \frac{Mc}{I}$. As before, the tensile stress is $\sigma_T = \frac{T}{A}$. The maximum stress at point S_2 is given as

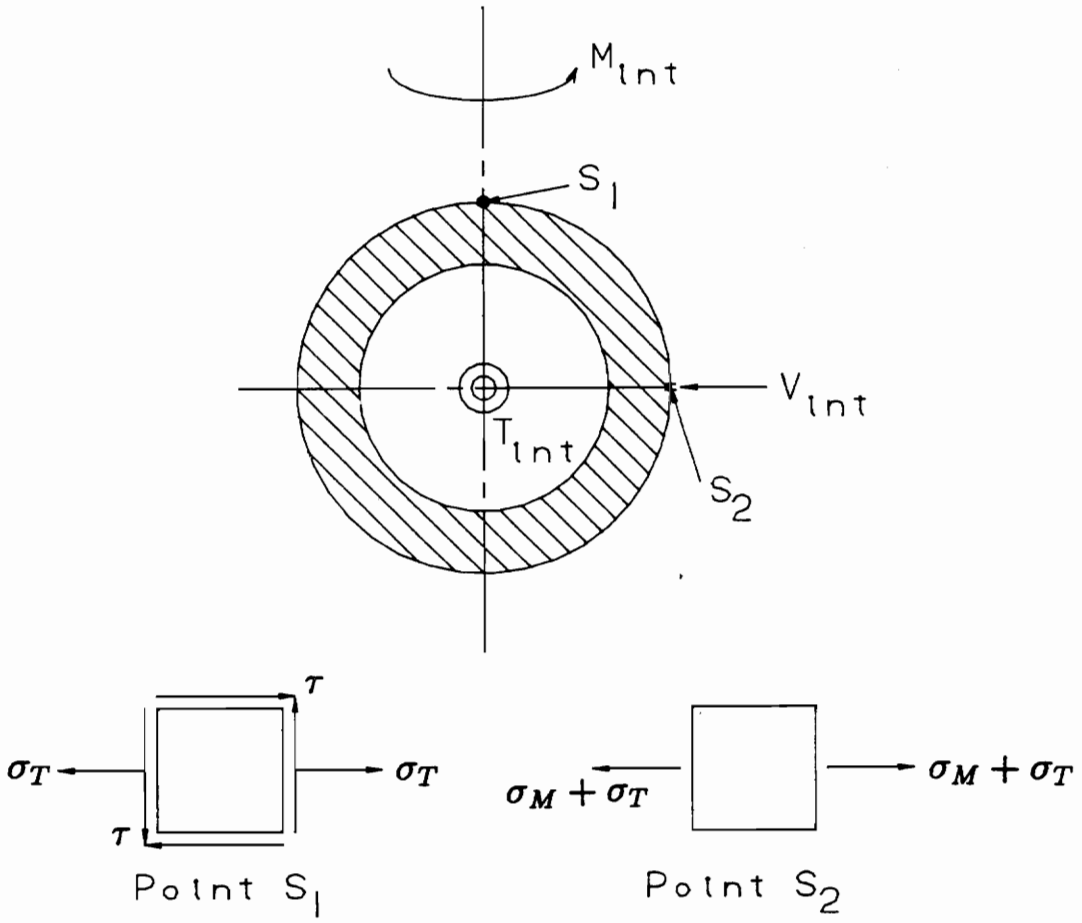

Figure 2.4: Cross Section of Triangular Truss Follower

$$\sigma_{max} = \sqrt{(abs(\sigma_T) + abs(\sigma_M))^2} \tag{2.25}$$

The absolute value of σ_T and σ_M are used because the maximum stress occurs where the tensile stress and bending stress are additive. The tensile stress is constant across the cross-section, however, the bending stress varies linearly across the cross-section with a value of 0 at the neutral axis, causing a compressive stress on one side and equal tensile stress on the other side. Therefore, the maximum stress occurs either at point S_1 or at the point directly across the tube.

To ensure that the member doesn't fail due to the internal stresses, the maximum stress must be less than the yield value for the material. However, the member may fail from buckling, so buckling failure must also be checked.

2.8 Buckling Analysis

For long, slender members that are loaded in compression, such as we have in the truss, failure from buckling must also be checked. Buckling failure is not related to the stresses in the member, but is a function only of the material properties (modulus of elasticity), the dimensions of the member (cross-sectional area, slenderness ratio), and the boundary conditions. The slenderness ratio is defined as $\frac{length}{radius\ of\ gyration}$ and depends only on the geometry of the member. The radius of gyration is defined as $\sqrt{\frac{A}{I}}$.

In this analysis, we use the Euler column theory, which defines the critical compressive load that will cause buckling. The critical load is given as

$$P_{cr} = \frac{C\pi^2 EA}{(\frac{l}{k})^2} \tag{2.26}$$

where:

C is a constant based on the end conditions of the member
E is the modulus of elasticity of the material
A is the cross-sectional area of the member
l is the length of the member
k is the radius of gyration

Shigley (19) provides values for the end condition constant. For the follower, the ends are pinned and the value of C is 1.0. For the leadscrew tube, there is a cantilever connection at one end and a pinned connection at the other end. A conservative value of C for a cantilever-pinned column is 1.0.

2.9 Development of Active Member Force Model

The active member (consisting of the active base and the leadscrew tube) provides the displacement input that drives the truss motion. The active base contains the motor, gearhead, leadscrew, and supports. The leadscrew tube is a structural member that has a nut that mates with the leadscrew and a pinned connection at point B. The connection between the leadscrew and the leadscrew tube is assumed to be a cantilever.

The development of the model assumes that the actuator provides a kinematic input to the truss. The motor provides a rotational input, denoted as θ_m, to the motor armature. The motor armature is connected to a planetary gearhead which reduces the rotational motion by a factor of G_{gh}, the gearhead gain. The output shaft of

the gearhead is connected to the leadscrew. The leadscrew converts the rotational motion into the linear motion of the leadscrew tube. The leadscrew factor, G_{ls}, is based on the pitch of the leadscrew thread and is the linear displacement per revolution of the shaft. The transformation between the motor armature rotation and the leadscrew tube linear motion is defined as

$$X = \frac{G_{ls}}{G_{gh}}\theta_m \tag{2.27}$$

where:

X = the linear motion of the leadscrew tube

Since this is a linear relationship, and both G_{gh} and G_{ls} are constant, the first and second derivatives of the linear motion can be found as

$$\dot{X} = \frac{G_{ls}}{G_{gh}}\dot{\theta}_m \tag{2.28}$$

$$\ddot{X} = \frac{G_{ls}}{G_{gh}}\ddot{\theta}_m \tag{2.29}$$

The force exerted on the leadscrew tube by the leadscrew depends on the linear acceleration of the active member and the masses of the truss members. We assume that the motor can provide enough torque, through the gearhead and the leadscrew, to provide the specified displacement, velocity, and acceleration. After the force analysis is done at each time step, the resulting linear force on the leadscrew tube is found.

To develop the relationship between the force exerted by the leadscrew and the

motor torque, it is necessary to perform a force analysis on the leadscrew thread. To do this, we "unroll" one turn of the thread and analyze the forces acting on it. There are two cases that we will look at: in the first, the screw is acting against the load; in the second case, the screw is acting with the load. To determine whether the torque is against or with the load, we look at the difference between the static load on the leadscrew tube and the actual applied load. If the applied load is in the same direction as the static load, the torque is acting with the load. If the directions of the static and applied loads are opposite, then the torque is acting against the load. Fig. 2.5 shows the force diagrams for both cases. We will go through the development for the first case only; the development for the second case is similar.

Summing the forces shown in Fig. 2.5(a) in the x and y directions yields the two force equations

$$\sum F_x = P - N sin\lambda - \mu N cos\lambda \tag{2.30}$$

$$\sum F_y = F - N cos\lambda + \mu N sin\lambda \tag{2.31}$$

where:

P= force required to move the load
F= force exerted on the leadscrew tube
N= normal force acting the thread
μ= the dynamic coefficient of friction
λ= the lead angle of the thread

Eliminating the normal force, N, from these equations and solving for the required force, P, on the thread yields

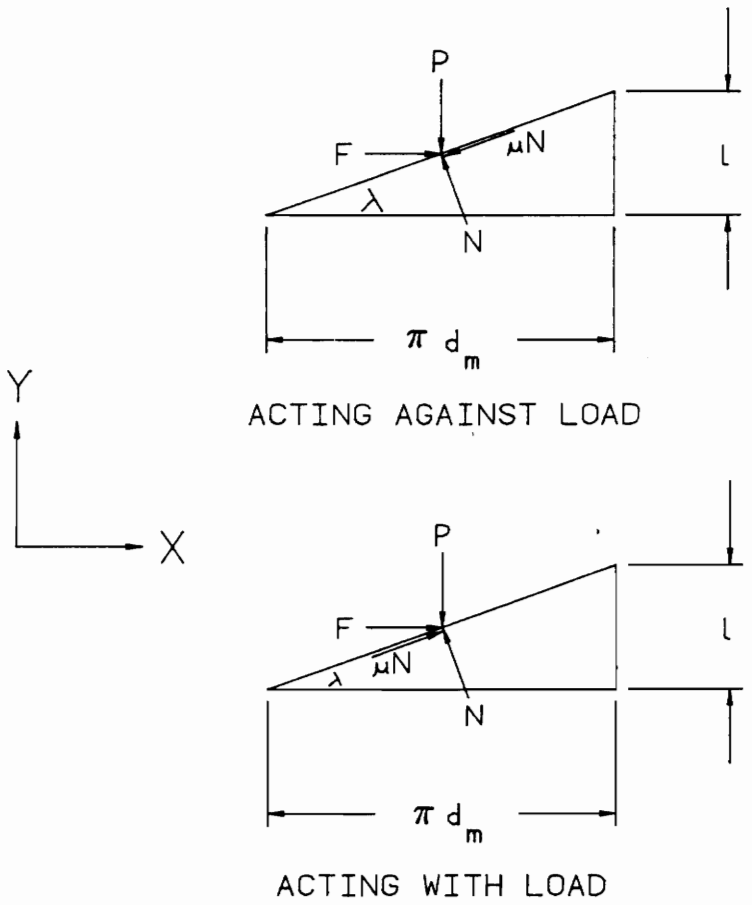

Figure 2.5: Leadscrew Thread Force Diagram

$$P = \frac{F(sin\lambda + \mu cos\lambda)}{cos\lambda - \mu sin\lambda} \tag{2.32}$$

The torque developed by the leadscrew is given as

$$T_P = \frac{d_m}{2} P \tag{2.33}$$

where:

d_m= the mean diameter of the leadscrew

Using the geometry of the thread, we can find the relation between the lead angle and the lead as

$$\lambda = tan^{-1} \frac{l}{\pi d_m} \tag{2.34}$$

where:

l=the lead of the leadscrew thread ($2\pi G_{ls}$)
d_m=the mean diameter of the leadscrew

Substituting these equations into equation 2.30 above and dividing by $cos\lambda$ provides the final relationship between the linear force exerted on the leadscrew tube and the torque provided by the leadscrew

$$T_p = \frac{F d_m}{2} \left(\frac{l + \pi \mu d_m}{\pi d_m - \mu l} \right) \tag{2.35}$$

The torque, T_p, is the torque required both to act against the load and to overcome the thread friction. We use the same procedure as above to find the torque required

to act with the load as

$$T_p = \frac{F d_m}{2}\left(\frac{\pi \mu d_m - l}{\pi d_m + \mu l}\right) \tag{2.36}$$

The above torque relationship is based on an a square thread screw. If an Acme or Unified thread screw is used, the effect of the thread angle, α_T, also comes into play. The thread angle is the angle that the thread makes with the longitudinal axis of the leadscrew. The effect of the thread angle is that the required force, F, acts at an angle, α_T from the vertical. The torque/ force relationship, including the effects of the thread angle, is given as

$$T_p = \frac{F d_m}{2}\left(\frac{l + \pi \mu d_m sec\alpha_T}{\pi d_m - \mu l sec\alpha_T}\right) \tag{2.37}$$

and

$$T_p = \frac{F d_m}{2}\left(\frac{\pi \mu d_m sec\alpha_T - l}{\pi d_m + \mu l sec\alpha_T}\right) \tag{2.38}$$

In order to use the above equations, it is necessary to know the coefficient of friction of the leadscrew. This is difficult to obtain with any certainty. For the triangular truss, the coefficient of friction is assumed to be 0.3.

The motor must provide the calculated torque and also overcome the effects of the armature inertia. The armature inertia torque is defined as

$$T_a = J_m \ddot{\theta}_m \tag{2.39}$$

where:

T_a= the torque required to overcome the armature inertia
J_m= the armature inertia
$\ddot{\theta}_m$= the rotational acceleration of the motor armature

The total torque that the motor must provide (for the torque acting against the load) is

$$T_m = T_a + T_p = J_m\ddot{\theta}_m + \frac{Fd_m}{2}\left(\frac{l + \pi\mu d_m sec\alpha}{\pi d_m - \mu l sec\alpha}\right) \tag{2.40}$$

The torque provided by the motor depends on the voltage and the current in the motor. The basic electrical equation describing the motor is

$$V_m = L_a\dot{I}_a + R_a I_a + K_b\dot{\theta}_m \tag{2.41}$$

where:

V_m= the motor input voltage
L_a= the motor inductance
I_a= the motor current
R_a= the motor armature resistance
K_b= the motor back emf coefficient
$\dot{\theta}_m$= the motor shaft speed

In this analysis, the electrical time constant caused by the inductance is assumed to be negligible compared to the mechanical time constant (26). The motor torque, in terms of motor constants, is defined as

$$T_m = I_a K_t \tag{2.42}$$

where:

T_m= the motor torque
K_t= the motor torque constant
I_a= the motor current

Knowing the required torque history and the required motor speed history (from the linear velocity), we can back out the required motor parameters needed to provide the specified displacement, velocity, and acceleration profile. For a given motor, the values of K_t, K_b, R_a and J_m will be specified. These values are used in the equations to find the voltage and current histories required by the motors to meet the specified motion. We find the necessary current at each time step by the relation

$$I_a = \frac{T_m}{K_t} \tag{2.43}$$

We can substitute the current relationship into the motor electrical equation to find the required voltage at each time step as

$$V_m = \frac{T_m R_a}{K_t} + K_b \dot{\theta}_m \tag{2.44}$$

We now have found the required voltage and current that must be input into the motor at each time step in order to achieve the specified linear displacement, velocity, and acceleration of the leadscrew tube.

The leadscrew gain is chosen so that the leadscrew is self-locking. That is, when the motor shaft is not rotating, the static axial load applied by the truss onto the leadscrew/gearhead will not backdrive the motor. The self-locking condition depends on the leadscrew gain and the diameter of the leadscrew (19) and is achieved when

$$G_{ls} < \frac{\mu d_m}{2} \tag{2.45}$$

Before linear motion can start, the motor must provide enough torque to overcome the static load and the friction in the leadscrew. Once the motion has begun, the motor torque must overcome both the static load caused by the weight of the truss and the dynamic load due to the motion.

2.10 Summary

In this chapter, we have developed a static and dynamic force model of the planar, triangular truss. The procedure involves solving the equations of motion of each member of the truss in parallel to find the interconnecting forces. These forces are then used to find the maximum stresses in the truss members throughout the motion. The motor parameters are also determined from the acceleration history and the loading.

Once the triangular truss model was developed, a simulation program was written in MATLAB command language to perform analysis. This simulation program is presented in Chapter 3.

Chapter 3

Triangular Truss Simulation

3.1 Development of Simulation Program

The model developed in Chapter 2 was used to perform a simulation of the triangular truss. This simulation was programmed using MATLAB command language. The simulation program is presented in Appendix A. Since we did not have a physical truss to model, the values for the member lengths, masses, areas, moments of inertia, actuator displacement model, and simulation time were chosen to correspond to a feasible design. All of the values were input when the simulation was performed. A set of base values were chosen and used as a reference for the rest of the simulations. First, a static simulation was performed. The results of the static simulation were then compared to a finite element model for the same truss configuration. Second, an initial dynamic simulation was performed. This simulation, using the base values, is known throughout this chapter as the Base Simulation. The simulation was then run with different values for the lumped mass and simulation time, and the results compared to the Base Simulation to demonstrate the changes in the forces and stresses.

3.1.1 Base Configuration

The values used for the Base Simulation are summarized in Table 3.1. These values correspond to the variables shown in the triangular truss model presented in Fig. 2.1.

Table 3.1: Base Simulation Dimensions and Constants

L_1=2.0m	x_{start}=0m	Material: Aluminum
L_2=2.0m	x_{end}=1.2m	Density=2711.5 $\frac{kg}{m^3}$
L_{3min}=0.5m	ID=0.02222m	Simulation Time=3s
L_4=2.0m	OD=0.0254m	No. of Steps= 50
Lumped Mass=10kg	$K_t = 0.0398\frac{N-m}{A}$	$K_b = 0.0398\frac{V}{rad-s}$
$J_m = 6.76 \times 10^{-6} Kg - m^2$	$R_a = 2$	$G_{gh} = 0.9618$
$G_{ls} = 0.003\frac{m}{rad}$		

3.2 Static Analysis

The first simulation performed was a static analysis in which the length of the active member did not change. The dimensions used for the static simulation are those shown in Table 3.1, except that, since there is no motion of the active link, x_{end} is also 0. The resulting reaction forces and internal member forces were then compared to the results from a finite element analysis. The commercial finite element package MSC/PAL was used to analyze the static case. The MSC/PAL model file and loading file are included in Appendix B. Due to way the MSC/PAL code treats gravity, the weight of the members was not included in the static analysis. MSC/PAL uses a lumped mass approach to modeling the members, thus grouping the weight of the members at the two nodes. The MATLAB simulation assumes that the weight

acts at the center of gravity of the member. This difference between the two means of applying weight forces will cause discrepancies between the two analyses. This difference was overcome by not applying gravitational forces to either model. The loads on the truss were generated by applying an external force to the node connecting the follower and the leadscrew tube. The external force of 98.1 N was calculated to be the gravitational force of the lumped mass and was applied to both models.

The results of the MATLAB simulation and the MSC/PAL analysis are presented in Table 3.2. The two analyses result in identical results for the static case.

Table 3.2: Comparison of Static MATLAB Simulation and MSC/PAL FEA Simulation

VALUES	MATLAB	MSC/PAL
Force Applied to Point B (Y-Dir)	-98.1 N	-98.1 N
Base Reaction at Point O_2 (X-Dir)	17.1813 N	17.18 N
Base Reaction at Point O_2 (Y-Dir)	21.4594 N	21.46 N
Base Reaction at Point O_1 (X-Dir)	-17.1813 N	-17.18 N
Base Reaction at Point O_1 (Y-Dir)	76.6406 N	76.64 N
Tension in Member 1	-78.5429 N	-78.54 N
Tension in Member 2	-27.4900 N	-27.49 N
von Mises Stress in Member 1	$6.6136 \times 10^5 \frac{N}{m^2}$	$6.614 \times 10^5 \frac{N}{m^2}$
von Mises Stress in Member 2	$2.3148 \times 10^5 \frac{N}{m^2}$	$2.315 \times 10^5 \frac{N}{m^2}$

Since the MATLAB and the MSC/PAL results are the same, we feel confident that the MATLAB model is correct, at least for the static case. When the effects of the weights of the members are taken into account in the MATLAB model, the reaction forces increase. For example, the base reactions at point O_1, with gravity applied, are -20.1303 N in the X-direction and 92.9545 N in the Y-direction. Note that for these calculations, no external force is applied. However, the weight of the 10 kg

lumped mass is applied and is equivalent to the previously applied external force.

Once we were satisfied that the static model was correct, the next step was to run the dynamic simulation.

3.3 Base Simulation

The task for the base simulation was to extend the active link from the starting length of 0.5m ($L_{3min} + x_{start} = 0.5$) to the end length of 1.7m ($L_{3min} + x_{end} = 1.7$) in 3 seconds. The lumped mass connected to point B was 10 kg. Gravitational forces were included for all of the members in the truss. The first step in the simulation is to determine the position, velocity, and acceleration history of the active member. The third order model for the active member described in Section 2.2 was used. The active member profiles for the Base Simulation are presented in Fig. 3.1.

The Base Simulation was run using 50 time steps. Doubling the number of time steps did not produce significant changes in any of the calculated values, accelerations or forces. The nodal forces, reaction forces, and internal stresses were calculated at each time step. Figure 3.2 presents the maximum von Mises stress generated in the follower during this motion compared to the static von Mises stress at each position. The maximum dynamic von Mises stress was found to be at point S_2 at the midpoint of the follower (see Fig. 2.4). At this point, the bending moment and the tension add together to provide a maximum von Mises stress of $4.3393 \times 10^6 \frac{N}{m^2}$ at 1.8 seconds. The static von Mises stress for this point is $3.9695 \times 10^6 \frac{N}{m^2}$. The maximum stress caused by the motion is $4.7361 \times 10^5 \frac{N}{m^2}$ and occurs at 1.5 sec.

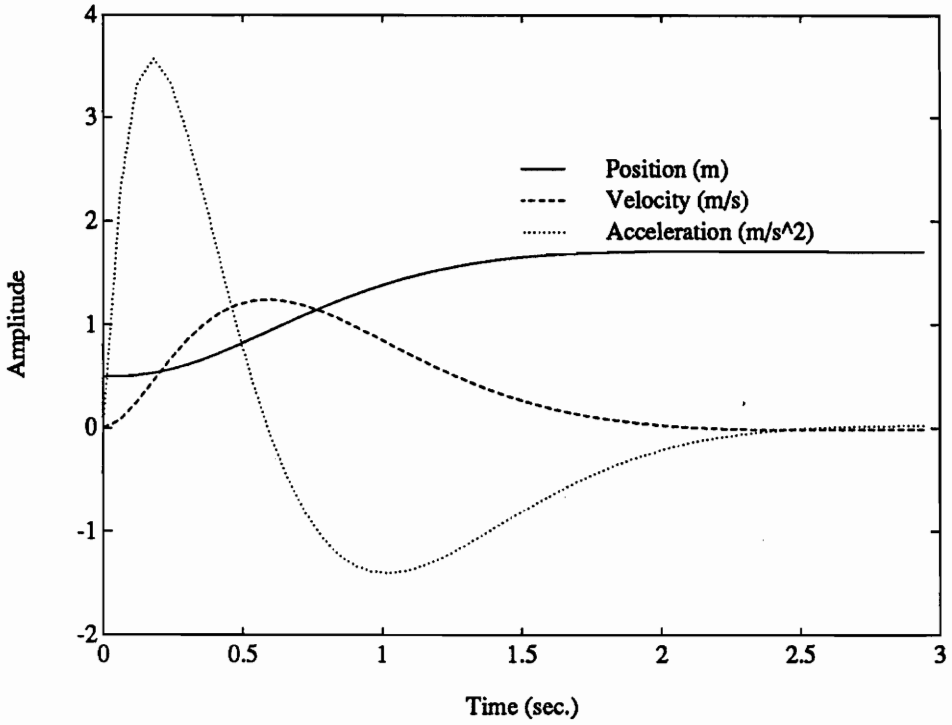

Figure 3.1: Position, Velocity, and Acceleration of the Active Member during the Base Simulation

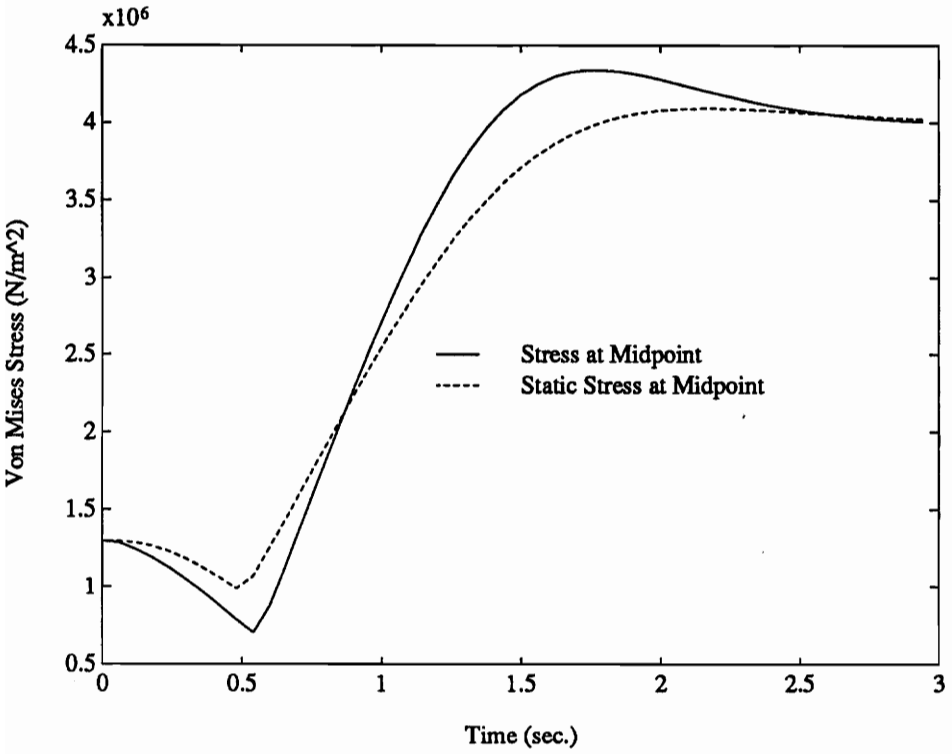

Figure 3.2: Maximum von Mises Stresses in the Follower

The maximum von Mises stress for the leadscrew is presented in Fig. 3.3 for both the dynamic and static cases. The maximum stresses occur at the cantilever connection with the active base and are caused by the combination of tensile and bending stresses. The maximum dynamic stress is $2.6256 \times 10^7 \frac{N}{m^2}$ and the maximum static stress is $2.4501 \times 10^7 \frac{N}{m^2}$. The maximum stress due to the dynamic motion of the leadscrew tube is $2.3014 \times 10^6 \frac{N}{m^2}$ and occurs at 1.44 seconds.

After the Base Simulation was performed, the values of the attached lumped mass and the acceleration profile were changed. The simulation was then rerun and the results compared to those of the Base Simulation.

3.4 Effects of Changing Acceleration Profile

To study the effects of changing the acceleration profile of the active member, two different simulations were performed. In each, the change in position of the active member was the same (i.e., to move from 0.5m to 1.7m). The difference was in the amount of time allowed to perform the motion. The simulation time was first increased from 3 sec. to 5 sec. Next, the simulation time was increased to 10 sec. For each simulation, the maximum von Mises stresses for both the follower and the lead screw tube were calculated and compared to those found during the Base Simulation. The results are shown below.

In Fig. 3.4, the total von Mises stresses generated in the follower during the 3 maneuvers are presented. Figure 3.5 presents the total von Mises stresses in the leadscrew tube for the 3 cases of differing acceleration. In each case, we see that the maximum stress increased when the acceleration was increased and decreased when the

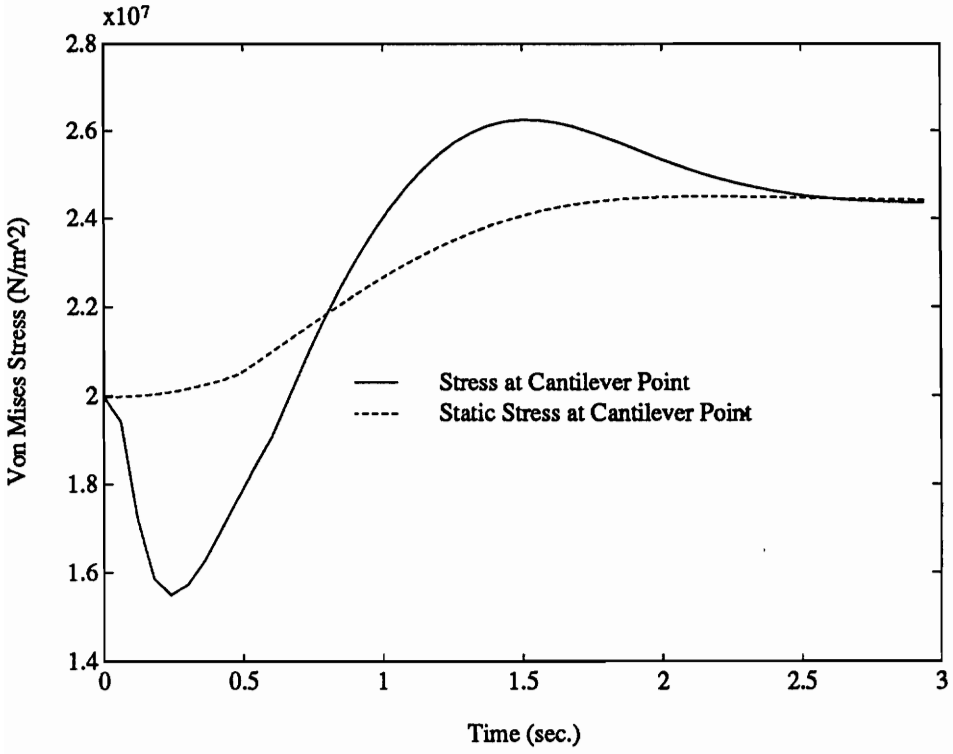

Figure 3.3: Maximum von Mises Stresses in the Leadscrew Tube

acceleration was decreased. We would expect that the faster motion would cause up higher stresses in the members. The static stress is the same for each acceleration case since it is dependent on position only. Note that as the acceleration decreases, the dynamic von Mises stresses generated during the motion approach the static values. Thus, for the 10 second simulation, the main contribution to the stress in the members is the static weight, not the motion.

3.5 Effect of Changing Lumped Mass

The next simulations performed looked at the effect of changing the value of the lumped mass attached to point B. It was expected that the static and total stresses would increase with an increase in the lumped mass and this is what was found. The lumped mass value was increased from the 10 kg value in the Base Simulation to 40 kg. The simulation was then run using a 3 sec. acceleration profile. The static and total von Mises stresses for the follower for both cases are presented in Fig. 3.6. The maximum static stress increases from $4.0917 \times 10^6 \frac{N}{m^2}$ for the 10 kg case to $1.0296 \times 10^7 \frac{N}{m^2}$ for the 40 kg case. The max stress caused by the motion increases from $4.7361 \times 10^5 \frac{N}{m^2}$ to $1.4617 \times 10^6 \frac{N}{m^2}$.

3.6 Actuator Forces

The active base provides a force input (called the actuator force) to the leadscrew tube that causes the motion of the truss. This force depends on both the loading on the truss and the acceleration profile. Figure 3.7 shows the linear force applied to the leadscrew tube during the 3, 5, and 10 second simulations with a 10 kg lumped mass. The static force for each position in the motion is also shown. During the

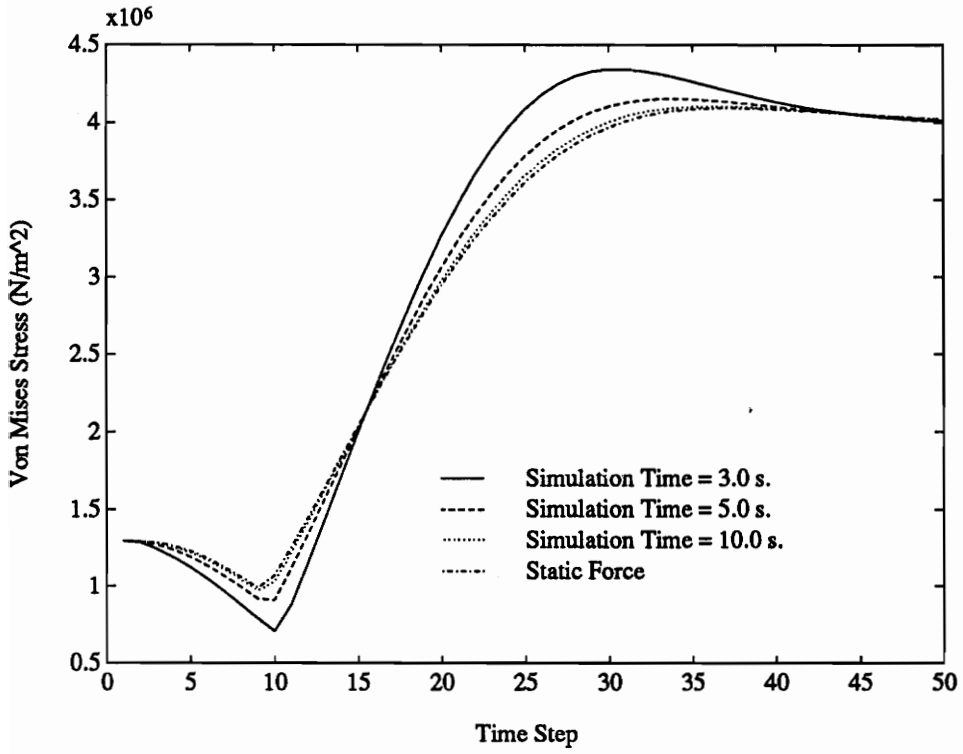

Figure 3.4: Comparison of Maximum von Mises Stresses in the Follower

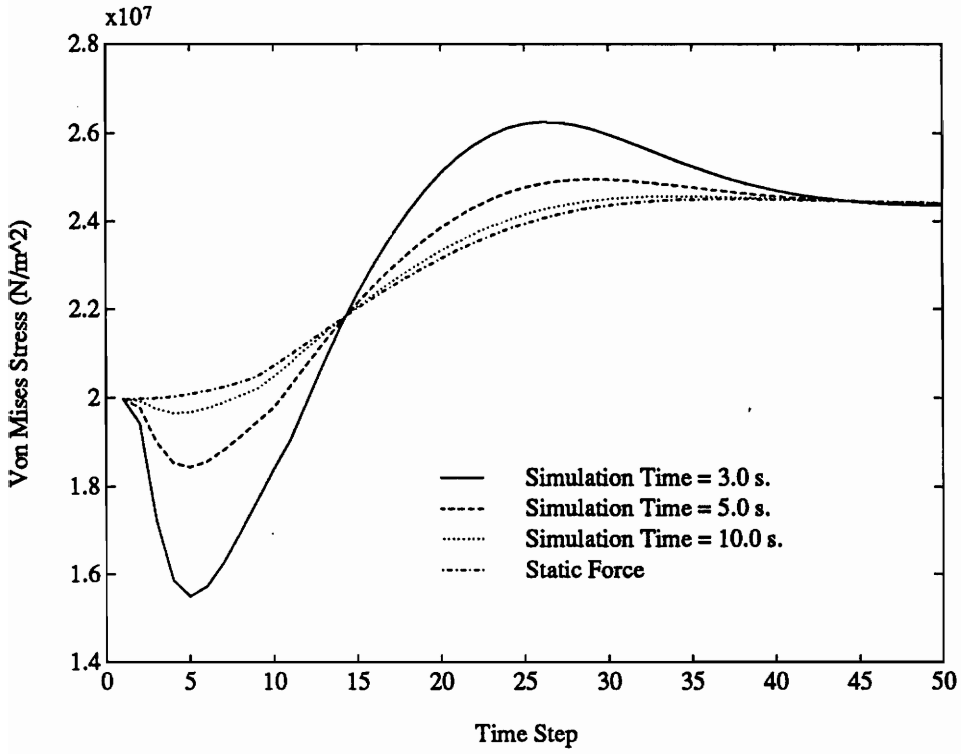

Figure 3.5: Comparison of Maximum von Mises Stresses in the Leadscrew Tube

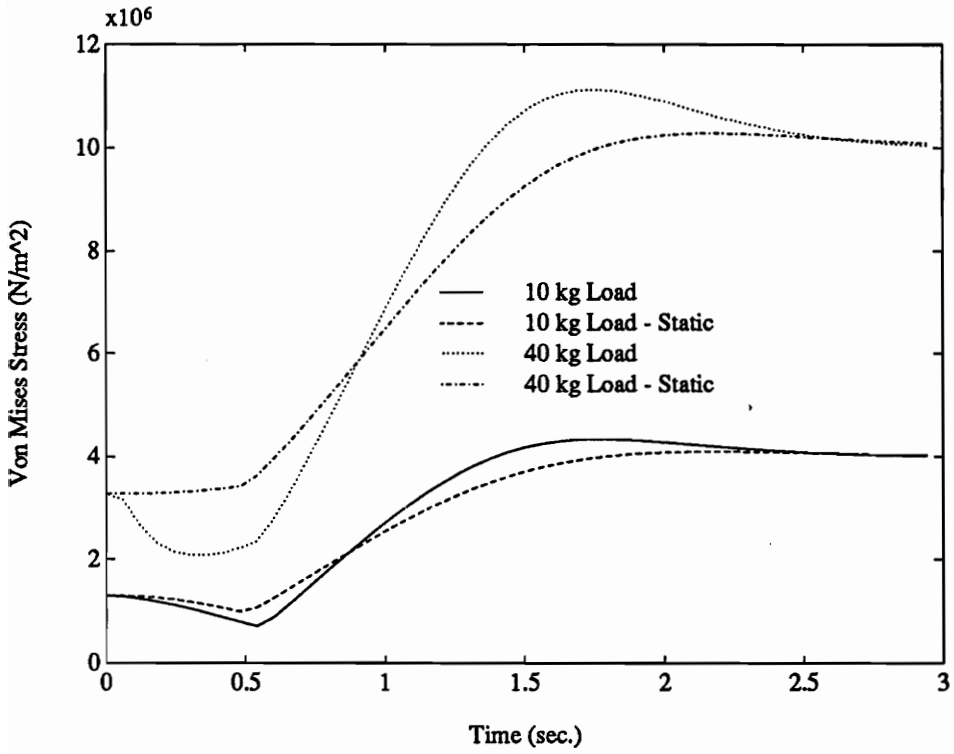

Figure 3.6: Comparison of von Mises Stresses in the Follower

positive acceleration of the active member, the actuator force causes compression on the leadscrew tube. When the active member is applying deceleration, the actuator force pulls back on the leadscrew tube and creates an additional tensile force in the leadscrew tube. As the acceleration goes to 0, the actuator force approaches the static force. The shorter the simulation time (and therefore, the higher the acceleration of the active member), the larger the loads that are applied to the leadscrew tube. As the simulation time approaches 10 seconds, the actuator loads applied to the leadscrew tube approach the static loads.

3.7 Motor Simulation

The motor simulation was performed to calculate the motor torque, voltage, and current required at each timestep. The motor was assumed to have a maximum speed of 3600 rpm (approximately 400 rad/s). The maximum velocity of the leadscrew tube was then used in conjunction with the maximum allowable motor speed to choose a gearhead gain (G_{gh}) and a leadscrew gain (G_{ls}). These gains are based on the acceleration requirements of the simulation, therefore, a different set of values must be used for each simulation. The results of the motor calculations are presented for the base simulation (T=3.0 s) only.

For the base simulation, the maximum linear velocity of the actuator is $1.2477\frac{m}{s}$. The mechanical gain was then chosen according to the relation

$$G_{mech} = \frac{\dot{X}_{max}}{\dot{\theta}_{m_{max}}} \tag{3.1}$$

where:

$G_{mech}=\frac{G_{ls}}{G_{gh}}$ - the mechanical gain of the motor/leadscrew

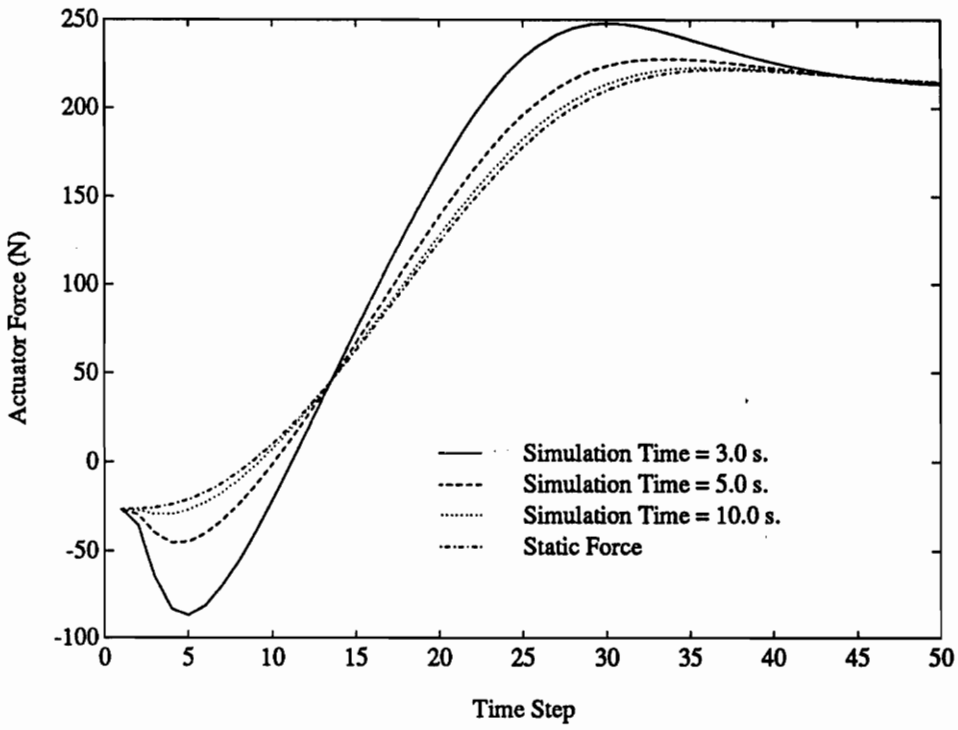

Figure 3.7: Comparison of Actuator Forces Applied to the Leadscrew Tube for 10kg Load

The leadscrew gain was selected so that the screw thread would meet the self locking condition, based on a coefficient of friction of 0.30 and mean diameter of the leadscrew of 0.02 m. These values result in a maximum value for the leadscrew gain of .003 m/rad. We then calculate the gearhead gain as $\frac{G_{ls}}{G_{mech}}$. For the base simulation, the gearhead gain was .9618.

We also assumed that the leadscrew used a square thread. The motor parameters were then calculated at each timestep in the motion. The motor torque history is presented in Fig. 3.8. The motor voltage and current histories are presented in Fig. 3.9 and Fig. 3.10. The motor torque approaches 0 $N-m$ at approximately 0.5 seconds. At this point, the follower is vertical and supports the entire weight of the lumped mass. Also, the actuator force applied to the leadscrew tube approaches 0 at this time. Therefore, the torque loading on the motor at this point is due to the angular acceleration of the motor armature. Since the torque is nearly 0, the current will approach 0 also. The voltage at this point is due to the back emf (related to motor armature speed). At the end of the motion, the motor torque is approaching a steady state value. This torque is the value required to support the static load at the final position. In actuality, as the motion of the truss ends, the self locking characteristic of the leadscrew will support the load and the motor will not be required to provide a torque.

3.8 Buckling Results

The maximum buckling load for the members are based on the geometry of the members. This maximum allowable load is then compared to the maximum com-

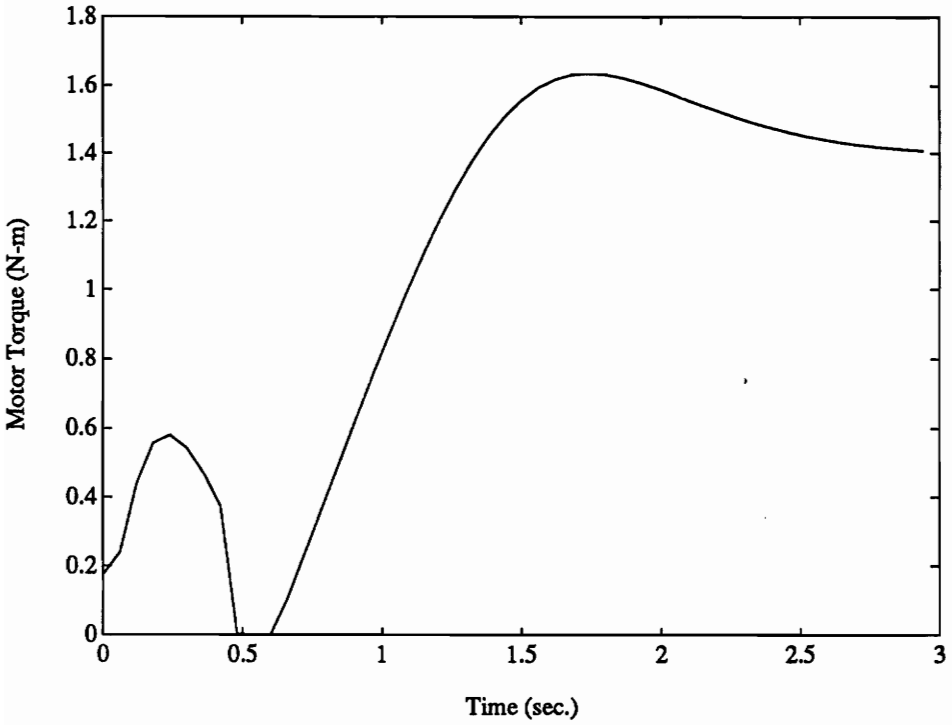

Figure 3.8: Motor Torque History for the Base Simulation

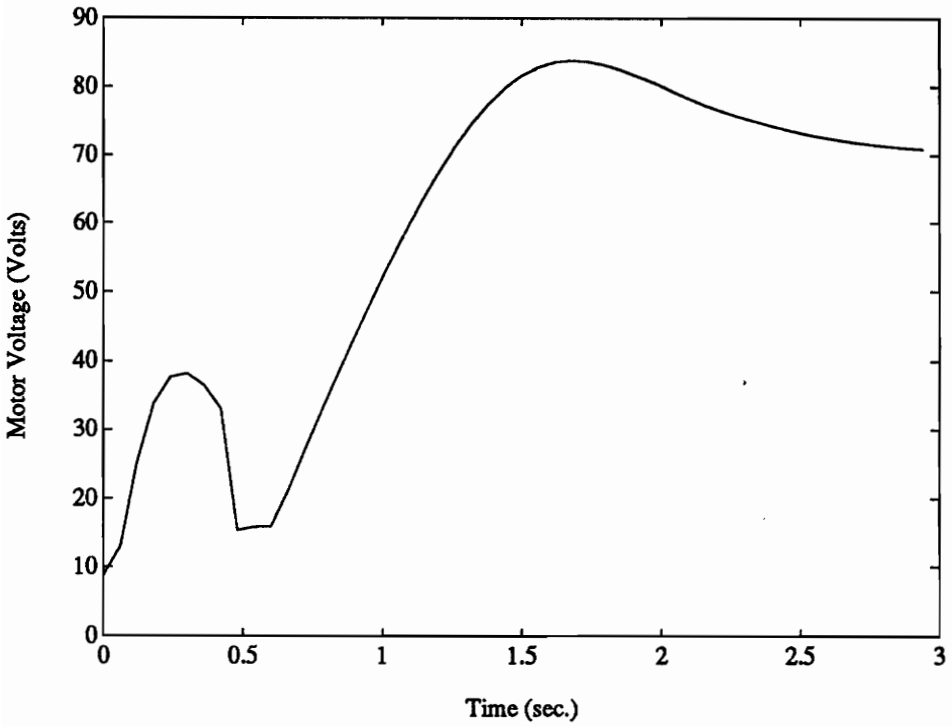

Figure 3.9: Motor Voltage History for the Base Simulation

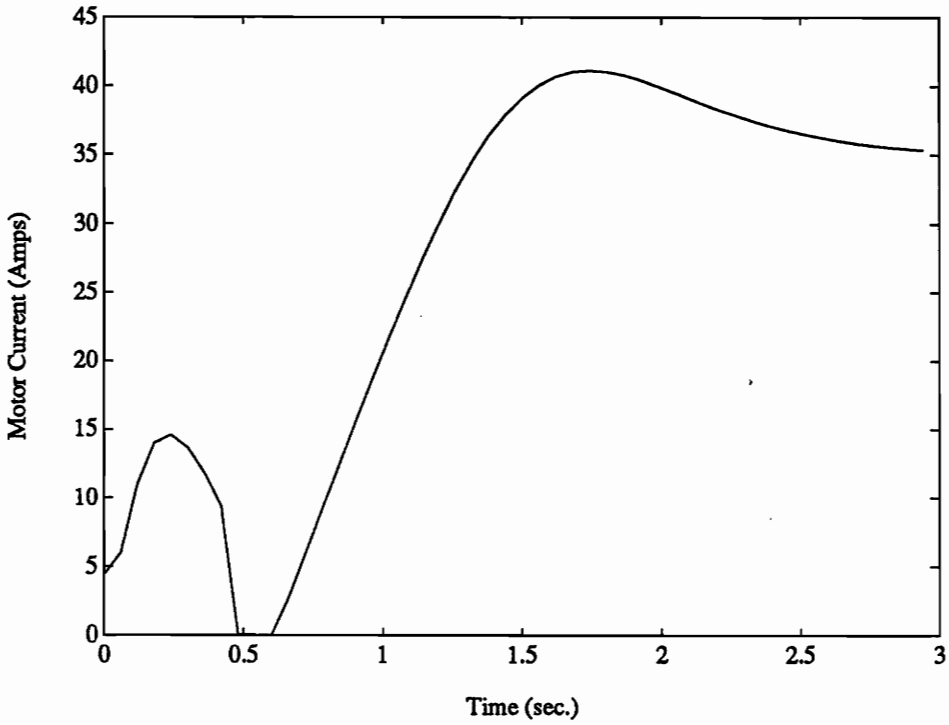

Figure 3.10: Motor Current History for the Base Simulation

pressive load applied to each member. The results of the buckling analysis for the triangular truss are summarized in Table 3.3.

Table 3.3: Buckling Analysis Results

MEMBER	ALLOWABLE LOAD	3 sec. - 10 kg Simulation	3 sec. - 40 kg Simulation
Follower	1481.2 N	308.6 N	1115.1 N
Leadscrew Tube	1481.2 N	86.9 N	336.2 N

Neither the follower nor the leadscrew tube will fail due to buckling during these maneuvers. However, we have not applied a safety factor to the maximum load. The maximum compressive force on the follower is getting close to the maximum allowable load; if we used a safety factor of 2, it would exceed the allowable load. In addition, the buckling analysis is based on a static loading with no bending moments on the member. Therefore, if we were designing the truss to perform the motion with the 40 kg load, we should change the follower dimensions to get a greater resistance to buckling.

3.9 Summary

The MATLAB simulation provided excellent results for the static analysis of the triangular truss when compared with the finite element analysis. The results achieved during the dynamic analyses were not compared to experimental results, but do agree with our expectations. The purpose of performing the triangular truss simulations was as much to work through the method of analysis as to get results. The results presented in this chapter are intended to inform the readers of the type of information we will be getting and to prepare them for the results of the spatial analysis.

Chapter 4

Development of Spatial Truss Model

The analysis is now extended to a three dimensional, spatial truss. We will first look at a single, double-octahedral bay. A schematic diagram of the truss is presented in Fig. 4.1. There are 21 members in the truss, connected together at 9 node points. The node points and members are identified in the figure; nodes are denoted Ni and members are denoted by Mi. We will refer to these identifiers throughout the model development. The reference coordinate axes are fixed at the base of the truss and are identified as $\hat{n}_1, \hat{n}_2$, and $\hat{n}_3$. The x, y, z coordinates of the nodes are identified in vector form where N(i) is $\{Ni(x), Ni(y), Ni(z)\}$. The midplane of the truss (members M7/8 through M11/12) is composed of active members. As in the model of the planar truss, the active members are modelled as two links: an active base (members M7, M9, and M11), and a leadscrew tube (members M8, M10, and M12). All other members are of fixed length.

Two steps are required to completely analyze the forces in the truss. In the first step, the unknown forces acting at the nodes are found. The model for this step is referred to as the overall truss model. Once the nodal forces are found, the second step is to

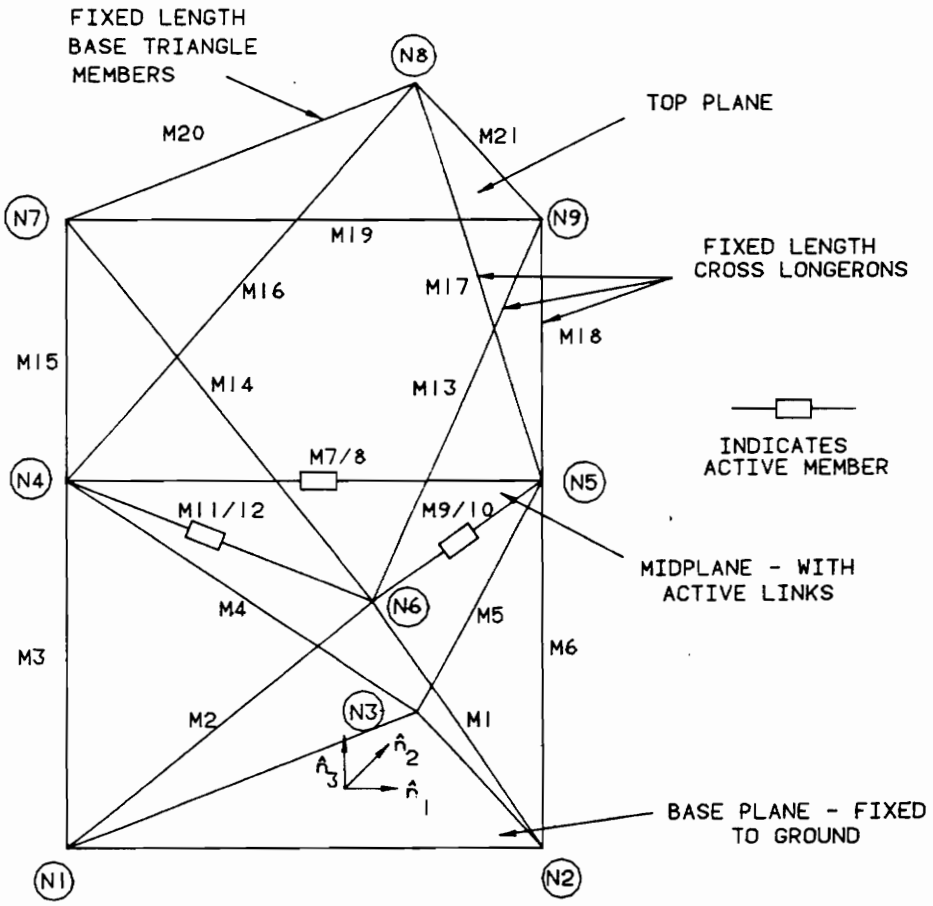

Figure 4.1: Double-Octahedral Truss Model

calculate the forces occurring in the active members. In developing the overall truss model, the active members are considered to be one link. This is done because, for an ideal truss, the members are pinned at the nodes and have a free degree of freedom of rotation about the longitudinal axis. However, the active members have a cantilever connection between the active base and the leadscrew tube. The cantilever connection introduces six forces into the model (three linear forces and three moments), but due to the free DOF, we only have five constraint equations. Therefore, including the free DOF in the model introduces an extra unknown but no corresponding constraint equation. Thus, the model becomes unsolvable since there are more unknowns than equations. To overcome this, the active members are modelled as a single, equivalent link during the analysis in which the nodal forces are calculated. The nodal forces are then used in a detailed analysis of the active member in which the forces acting at the cantilever connection are calculated.

Each member of the truss model has six nodal forces and gravity acting on it. The forces acting on each member are assumed to be independent. When the active members are modelled as a single link, the overall model is reduced to eighteen members. This results in 108 member nodal forces in the model. In addition, there are three forces required to support the truss at each of the base nodes, adding nine forces to the model. Therefore, there are a total of 117 unknown nodal forces in the model. For each member, applying Newton's second law gives five equations of motion, resulting in 90 EOMs for the model. The remaining equations needed to solve the model are provided by the fact that the sum of the forces at each node (in all three directions) must be zero. This adds 27 constraint equations, resulting in a total of 117 equations for the truss. Since we have an equal number of unknowns

and equations, the truss is statically determinate and we can solve for all of the unknown nodal forces.

In order to perform the force analysis, it is necessary to know the accelerations of the centers of gravity, and the angular accelerations, of each member. For the planar truss, this was a relatively simple matter of stepping through the motion of the single active member and finding the resulting orientations of the members. In the spatial case, however, this kinematic analysis becomes much more complex.

4.1 Spatial Truss Kinematic Analysis

The spatial truss geometry depends on the lengths of the three active members located in the midplane. There are basically two different types of kinematic analysis. The forward position analysis requires finding the location of the nodes based on the known lengths of the active members. The inverse analysis is to find the lengths of the active members given a known orientation of the truss. Padmanabhan (13) has developed a closed-form solution for the inverse analysis of the gimbal problem. That is, given the angular orientation of the top plane, the position of all nodes of the truss can be located in the fixed, Newtonian reference frame. The forward position analysis, however, has no closed-form solution and must be solved iteratively. In developing the motion of the spatial truss, the active link lengths are specified and we must solve the forward analysis to find the center of gravity and angular accelerations of the members. This requires solving the kinematics iteratively.

Warrington (24) developed a method for solving the forward analysis for a single,

double-octahedral spatial truss bay. For small motions about an operating point, the nonlinearities in the forward analysis can be linearized to give a good approximation for the location, velocity, and acceleration of the nodal points based on the small changes in the active member lengths. The position Jacobian is developed numerically by giving the active members small motions and calculating the resulting change in the nodal position. The velocity and acceleration Jacobians are then found by numerically calculating the partial derivatives of the velocities (or accelerations) with respect to the active members velocities (or accelerations). This analysis is done at a given operating point and is only valid for small motions about that point.

Warrington (25) also developed a method of finding the position, velocity, and acceleration of the nodes during a large angle slewing maneuver by iterating on the link lengths to find the angular orientation for those lengths. The method is as follows. The motion of the three active links is first discretized. Then at each time step, the inverse problem is solved iteratively to find the geometric orientation that corresponds to the given link lengths. Thus, the positions of the nodes are known at the time step. The link lengths are then incrementally changed during the next time step, and the next orientation is found. Using difference techniques, the velocity and acceleration of each of the nodes can be found. After the iterative solution is complete, we have the position, velocity, and acceleration history of each of the nodes throughout the entire maneuver. The position, velocity, and acceleration of the nodes are given in a coordinate frame that is fixed at the base of the truss. These node histories are used as inputs into the force analysis detailed in this chapter. The large-angle slewing kinematics is used in this thesis.

4.2 Newtonian and Body-Fixed Coordinate Systems

To perform a dynamic force analysis, it is necessary that the position, velocity, and acceleration of all of the truss nodes be known relative to a fixed, Newtonian reference system. The information provided by the kinematics analysis is given in terms of a reference system that is fixed at the base plane of the truss. For a single-bay truss, this is the Newtonian system. However, it may be more convenient to work in other coordinate systems when evaluating the equations of motion. In particular, a coordinate system that is fixed to the member allows us to use the principle axes of the member. Principle axes are defined as those in which the products of inertia go to zero. For a beam ot truss member, the principle axes consist of the longitudinal axis of the member (along the length) and two axes that are perpendicular to the longitudinal axis and to each other. In Fig. 4.2, both the Newtonian axes $(\hat{n}_1, \hat{n}_2, \hat{n}_3)$ and the body-fixed axes $(\hat{b}_1, \hat{b}_2, \hat{b}_3)$ are shown. Each member of the truss will have a unique body-fixed coordinate system. It is convenient to evaluate the torque (or rotational) equations of motion using body fixed coordinates where we only have to deal with moments of inertia. Newtonian coordinates are better for evaluating the force equations of motion since the accelerations of the nodes and the nodal forces are provided in Newtonian coordinates.

The Newtonian coordinate system and the body-fixed coordinate system for body i are related to each other through the cosine matrix as

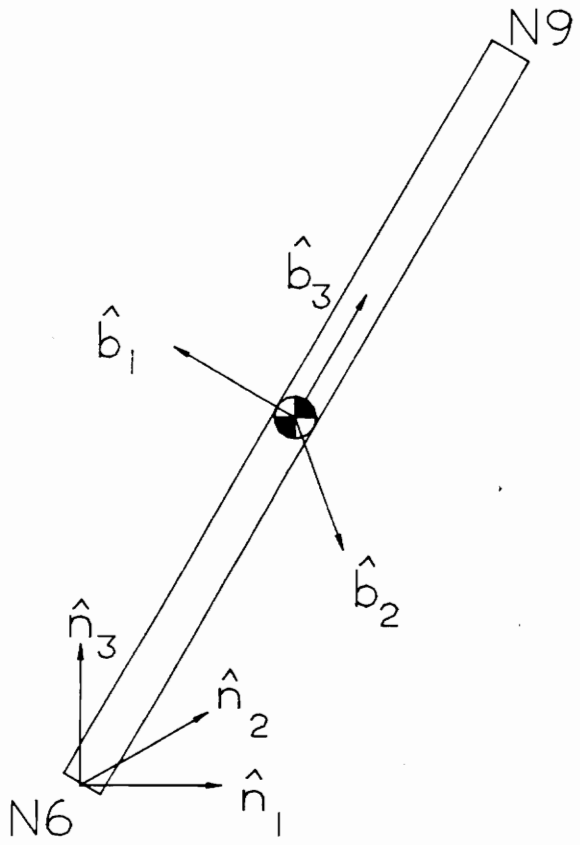

Figure 4.2: Member M13 Showing Global and Body-Fixed Axes

$$\begin{bmatrix} \hat{b}_1 \\ \hat{b}_2 \\ \hat{b}_3 \end{bmatrix} = \begin{bmatrix} C^i_{11} & C^i_{12} & C^i_{13} \\ C^i_{21} & C^i_{22} & C^i_{23} \\ C^i_{31} & C^i_{32} & C^i_{33} \end{bmatrix} \begin{bmatrix} \hat{n}_1 \\ \hat{n}_2 \\ \hat{n}_3 \end{bmatrix} \tag{4.1}$$

Using this transformation, we can change between the Newtonian and body-fixed coordinate systems at will. Each row of the cosine matrix is a unit vector describing the direction of the vector in Newtonian coordinates. The cosine matrix adheres to the right hand rule and has a determinate of +1. The method used for calculating the body-fixed coordinate systems will be described below.

4.2.1 Calculating Body-Fixed Coordinate Systems

For each member of the truss, we will have a unique body-fixed coordinate system. Consider member M13 shown in Fig. 4.2. Member 13 connects nodes N6 and N9. The $\hat{b}_3$ axis is always assumed to be taken along the longitudinal axis of the member, i.e., the axis connecting the two nodes. The third row of the cosine matrix is the transformation between the $\hat{n}$ coordinate system and the $\hat{b}_3$ axis and is calculated as

$$C^{13}_{3,:} = \frac{N_9 - N_6}{\| N_9 - N_6 \|} \tag{4.2}$$

where:

$C^{13}_{3,:}$-	third row of the cosine matrix for member 13
$N_9 - N_6$ -	vector difference between the nodes in the $\hat{n}_1, \hat{n}_2, \hat{n}_3$ coordinate system
$\| N_9 - N_6 \|$ -	absolute scalar distance between the nodes

The physical motion of the truss member is considered when developing the other

two axes of the body-fixed coordinate system. Due to the physical constraints of the truss, member 13 can only rotate about the axis connecting nodes N9 and N7. This axis is identified as $\acute{x}$ is this analysis. The two vectors, $C^{13}_{3,:}$ and $\acute{x}$ define a plane containing the member. An axis perpendicular to this plane will also be perpendicular to the member. This axis will be the $\hat{b}_1$ axis. Therefore, we define the $C^{13}_{1,:}$ transformation to be the cross product of these two vectors, i.e.,

$$C^{13}_{1,:} = \frac{\acute{x} \times C^{13}_{3,:}}{\| \acute{x} \times C^{13}_{3,:} \|} \tag{4.3}$$

Knowing two of the three unit vectors, we can find the third, $C^{13}_{2,:}$, using the cross product, as

$$C^{13}_{2,:} = C^{13}_{3,:} \times C^{13}_{1,:} \tag{4.4}$$

The cosine matrix relating the $\hat{b}$ and $\hat{n}$ coordinate systems for member 13 is given as

$$\left[\mathbf{C^{13}} \right] = \begin{bmatrix} C^{13}_{1,:} \\ C^{13}_{2,:} \\ C^{13}_{3,:} \end{bmatrix} \tag{4.5}$$

The cosine matrix for each of the truss members is developed using the analysis described above. Note that since these cosine matrices depend on the orientation of the members, they will be changing as the truss undergoes motion. Once we have the body-fixed coordinate system established, we can then develop the equations of motion of the members.

4.3 Overall Truss Equations of Motion

The equations of motion for the overall truss consist of the equations of motion for each member and the nodal constraint equations. The EOM analysis for a single member will be described below. Then the nodal constraint equations will be developed. Finally, these will be combined to give the system of equations describing the entire truss.

4.3.1 Member Equations of Motion

The equations of motion are developed for each member of the truss and then combined to form a description of the entire truss motion. To find the equations of motion for a member, we first draw an FBD and label the forces applied to it and the coordinate system. Consider the FBDs for member 13 that are presented in Fig. 4.3. There are two FBDs for the member, one using the Newtonian coordinate system (Fig. 4.3(a)) and one using the body-fixed coordinate system (Fig. 4.3(b)). At each node, there are three forces applied to the member. All forces applied to the members are described in the Newtonian coordinate system. This allows us to write the forces at all of the nodes in the same coordinate system and facilitates creating the combined system description. For member 13, we have $F_{x_{19}}, F_{y_{19}}, F_{z_{19}}$ applied to node N6 and $F_{x_{20}}, F_{y_{20}}, F_{z_{20}}$ applied to node N9. These global forces can be converted into body fixed forces by using the cosine matrix

$$\begin{bmatrix} F_{11\hat{b}_1} \\ F_{21\hat{b}_2} \\ F_{31\hat{b}_3} \end{bmatrix} = \begin{bmatrix} \mathbf{C^{13}} \end{bmatrix} \begin{bmatrix} F_{x_{19}\hat{n}_1} \\ F_{y_{19}\hat{n}_2} \\ F_{z_{19}\hat{n}_3} \end{bmatrix} \tag{4.6}$$

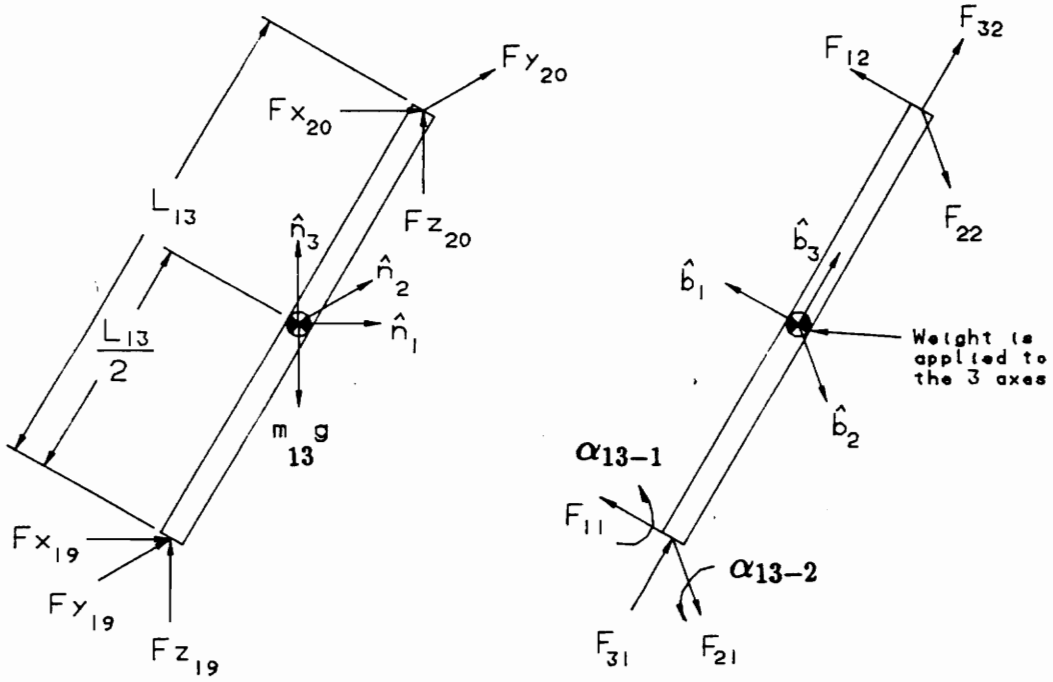

Figure 4.3: Free Body Diagrams for Member M13

$$\begin{bmatrix} F_{12\hat{b}_1} \\ F_{22\hat{b}_2} \\ F_{32\hat{b}_3} \end{bmatrix} = \begin{bmatrix} \mathbf{C^{13}} \end{bmatrix} \begin{bmatrix} F_{x_{20}\hat{n}_1} \\ F_{y_{20}\hat{n}_2} \\ F_{z_{20}\hat{n}_3} \end{bmatrix} \tag{4.7}$$

We use both the global forces and the body-fixed forces in developing the equations of motion. The global forces are used to develop the linear force EOMs (i.e., $\sum F_{\hat{n}_1}, \sum F_{\hat{n}_2}, \sum F_{\hat{n}_3}$) while the body fixed forces are used to evaluate the the moment equations ($\sum T_{\hat{b}_1}, \sum T_{\hat{b}_2}$). There are two angular motions that we must consider, α_{13-1} about the $\hat{b}_1$ axis and α_{13-2} about the $\hat{b}_2$ axis. As we mentioned above, we evaluate the angular accelerations in the body-fixed coordinate system so that we deal only with principle moments of inertia. Also, since the truss members are pinned at the nodes, there are no applied torques about the $\hat{b}_3$ axis.

Applying Newton's laws of motion to member 13 results in 5 EOMs

$$\sum F_{\hat{n}_1} \Rightarrow m_{13}a_{13\hat{n}_1} = F_{x_{19}} + F_{x_{20}} \tag{4.8}$$

$$\sum F_{\hat{n}_2} \Rightarrow m_{13}a_{13\hat{n}_2} = F_{y_{19}} + F_{y_{20}} \tag{4.9}$$

$$\sum F_{\hat{n}_3} \Rightarrow m_{13}a_{13\hat{n}_3} = F_{z_{19}} + F_{z_{20}} - m_{13}g \tag{4.10}$$

$$\sum T_{\hat{b}_1} \Rightarrow I_{cg_{13}}\alpha_{13-1} = \frac{L_{13}}{2}(F_{21} - F_{22}) \tag{4.11}$$

$$\sum T_{\hat{b}_2} \Rightarrow I_{cg_{13}}\alpha_{13-2} = \frac{L_{13}}{2}(F_{12} - F_{11}) \tag{4.12}$$

To facilitate combining the member EOMs into the system matrix, we arrange the EOMs in a 5 by 6 matrix as

$$\begin{bmatrix} m_{13}a_{13_{\hat{n}_1}} \\ m_{13}a_{13_{\hat{n}_2}} \\ m_{13}a_{13_{\hat{n}_3}} + m_{13}g \\ \frac{2}{L_{13}}(I_{cg_{13}}\alpha_{13-1}) \\ \frac{2}{L_{13}}(I_{cg_{13}}\alpha_{13-2}) \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 \\ C^{13}_{2,1} & C^{13}_{2,2} & C^{13}_{2,3} & -C^{13}_{2,1} & -C^{13}_{2,2} & -C^{13}_{2,3} \\ -C^{13}_{1,1} & -C^{13}_{1,2} & -C^{13}_{1,3} & C^{13}_{1,1} & C^{13}_{1,2} & C^{13}_{1,3} \end{bmatrix} \begin{bmatrix} F_{x_{19}} \\ F_{y_{19}} \\ F_{z_{19}} \\ F_{x_{20}} \\ F_{y_{20}} \\ F_{z_{20}} \end{bmatrix} \tag{4.13}$$

The EOM development described above is applicable to every member of the truss, both active and fixed-length. Only the identification of the forces and the values of the cosine matrices will change for the specific member. There are a total of 18 members in the overall truss model. The 18 members yield 90 equations (18x5) and 108 unknown nodal forces (18x6). Since there are more unknown forces than equations, we must find additional equations to solve for the forces. These additional equations are the nodal constraint equations.

4.3.2 Nodal Constraint Equations

There is no relative linear motion between the members connected at each node and the node point itself. Therefore the forces acting at the node must be in equilibrium. There are two types of equilibrium at the nodes, depending on whether or not a lump mass is attached to the node. If there is no attached mass, then the sum of the forces at the node must be equal to zero. If there is an attached mass, then the sum of the member forces must be equal to the lumped mass multiplied by the acceleration. In addition, external forces can be applied to the truss by applying forces at the nodes. At each node there can be forces and accelerations in each of the three Newtonian axes ($\hat{n}_1, \hat{n}_2$, and $\hat{n}_3$). Therefore, we need three equilibrium (or constraint) equations for each node. For example, at node N9, there are forces applied by member M13 ($F_{x_{20}}, F_{y_{20}}, F_{z_{20}}$), by member M18 ($F_{x_{30}}, F_{y_{30}}, F_{z_{30}}$), by member M19 ($F_{x_{32}}, F_{y_{32}}, F_{z_{32}}$),

and by member M21 ($F_{x_{36}}, F_{y_{36}}, F_{z_{36}}$). Assuming there are no lumped masses or external forces applied at the node, we can write the constraint equations as

$$\sum F_{\hat{n}_1} \Rightarrow \quad 0 = F_{x_{20}} + F_{x_{30}} + F_{x_{32}} + F_{x_{36}} \tag{4.14}$$

$$\sum F_{\hat{n}_2} \Rightarrow \quad 0 = F_{y_{20}} + F_{y_{30}} + F_{y_{32}} + F_{y_{36}} \tag{4.15}$$

$$\sum F_{\hat{n}_3} \Rightarrow \quad 0 = F_{z_{20}} + F_{z_{30}} + F_{z_{32}} + F_{z_{36}} \tag{4.16}$$

There will be three constraint equations at each of the nine nodes. This will add 27 equations to the truss model, resulting in a total of 117 equations. However, we only have 108 unknown forces from the member EOMs. The additional 9 forces come from the reaction forces from the ground applied at nodes N1, N2, and N3. At each of the ground nodes, there will be three forces applied to the truss (R_{x_i}, R_{y_i}, and R_{z_i}). For node 1, for example, the constraint equations become

$$\sum F_{\hat{n}_1} \Rightarrow \quad R_{x_1} = F_{x_4} + F_{x_6} \tag{4.17}$$

$$\sum F_{\hat{n}_2} \Rightarrow \quad R_{y_1} = F_{y_4} + F_{y_6} \tag{4.18}$$

$$\sum F_{\hat{n}_3} \Rightarrow \quad R_{z_1} = F_{z_4} + F_{z_6} \tag{4.19}$$

There are now 117 equations describing the motion of the truss and 117 unknown forces. These equations and forces are next combined to give the overall truss system matrices.

4.3.3 Combined System of Equations

Now that we have an equal number of forces and equations, we can set up the overall truss system of equations. Since the forces applied at each node are independent, each of the member matrices are independent. To combine these into a system matrix, we pad each matrix with columns of zeros. Therefore, for each member, we will have a 5x117 matrix. There are 27 constraint equations which will consist of ones and zeros and will give a 27x117 matrix. These matrices are then combined resulting in the overall system matrices. The description of the overall truss is given as

$$\left\{ m\ddot{x} \right\} = \left[\mathbf{SM} \right] \left\{ F \right\} \tag{4.20}$$

where:

$\{m\ddot{x}\}$	-	117x1 vector of inertial forces and constraint values
[SM]	-	117x117 matrix of force and constraint coefficients
$\{F\}$	-	117x1 vector of nodal forces and base reaction forces

In order to solve for the forces, we must now calculate the acceleration of the center of gravity and the angular acceleration of each member.

4.4 Acceleration Analysis

In Section 4.1, we discussed the development of the kinematic relationship between the motion of the active members and the motion of the nodes. From that analysis, the position, velocity, and acceleration of the nodes is provided as input to the force analysis. It is necessary to use absolute velocities and accelerations that are measured relative to a fixed, Newtonian reference frame to evaluate the EOMs. The

nodal information provided by the kinematic analysis assumes that the base plane (defined by nodes N1, N2, and N3) is fixed in space. For the force analysis of the singe double-octahedral bay, this assumption is true. When the force analysis is extended to multiple bays in Chapter 6, the base plane will not always be fixed and we will have to make some corrections to the kinematic nodal acceleration.

The velocity of the i^{th} node is identified as a vector quantity, V(i), given as

$$V(i) = [V_i(\hat{n}_1),\ V_i(\hat{n}_2),\ V_i(\hat{n}_3)]^T \tag{4.21}$$

where

$V_i(\hat{n}_1)$ - component of velocity of node Ni in the $\hat{n}_1$ direction
$V_i(\hat{n}_2)$ - component of velocity of node Ni in the $\hat{n}_2$ direction
$V_i(\hat{n}_3)$ - component of velocity of node Ni in the $\hat{n}_3$ direction

In a like manner, the acceleration of the i^{th} node is a vector quantity given by

$$A(i) = [A_i(\hat{n}_1),\ A_i(\hat{n}_2),\ A_i(\hat{n}_3)]^T \tag{4.22}$$

where

$A_i(\hat{n}_1)$ - component of acceleration of node Ni in the $\hat{n}_1$ direction
$A_i(\hat{n}_2)$ - component of acceleration of node Ni in the $\hat{n}_2$ direction
$A_i(\hat{n}_3)$ - component of acceleration of node Ni in the $\hat{n}_3$ direction

The position, velocity, and acceleration of each of the nodes will be provided at each time step as input to the force analysis. Using this information, the angular accelerations and accelerations of the centers of gravity can be calculated.

4.4.1 Angular Accelerations

Angular accelerations for each of the members is required in order to evaluate the torque components of the EOMs. We have two distinct angular acceleration problems for the spatial truss. The fixed members are components of fixed planes for which there is no relative angular motion between the members. In the case of the active members, however, there is relative angular motion in the midplane. These two cases will require different formulations for angular accelerations.

For the case of the fixed members, we calculate the angular accelerations based on the Newtonian velocities and accelerations of the nodes of the plane. Consider the top plane containing the fixed-length members M19, M20, and M21 and nodes N7, N8, and N9. The plane is shown in Fig. 4.1. In general, there can be three rotations (and corresponding angular velocities and accelerations) of the plane; one about each of the three Newtonian coordinate axes. The angular velocities are identified as ω_1, ω_2, and ω_3, and the angular accelerations are identified α_1, α_2, and α_3. We first have to solve for the angular velocities. Since there is no relative velocity between the members, we calculate the velocity of two nodes as

$$V_9 = V_7 + V_{9/7}$$

$$V_9 = V_7 + (\omega_1 \hat{n}_1 + \omega_2 \hat{n}_2 + \omega_3 \hat{n}_3) \times (N_9 - N_7) \tag{4.23}$$

where:

V_9 -	Vector containing the $\hat{n}_1, \hat{n}_2, \hat{n}_3$ components of the velocity of node $N9$
V_7 -	Vector containing the $\hat{n}_1, \hat{n}_2, \hat{n}_3$ components of the velocity of node $N7$
$(N_9 - N_7)$-	Vector containing the $\hat{n}_1, \hat{n}_2, \hat{n}_3$ distance between the two nodes

For convenience, we will write $(N_9 - N_7)$ as $(\Delta x_1\ \Delta y_1\ \Delta z_1)$. The velocity equations for node $N9$ with respect to node $N7$ (in each of the coordinate directions) are then given as

$$V_{9\hat{n}_1} = V_{7\hat{n}_1} + (\omega_2 \Delta z_1 - \omega_3 \Delta y_1) \tag{4.24}$$

$$V_{9\hat{n}_2} = V_{7\hat{n}_2} + (\omega_3 \Delta x_1 - \omega_1 \Delta z_1) \tag{4.25}$$

$$V_{9\hat{n}_3} = V_{7\hat{n}_3} + (\omega_1 \Delta y_1 - \omega_2 \Delta x_1) \tag{4.26}$$

or, in matrix form,

$$\begin{bmatrix} V_{9\hat{n}_1} - V_{7\hat{n}_1} \\ V_{9\hat{n}_2} - V_{7\hat{n}_2} \\ V_{9\hat{n}_3} - V_{7\hat{n}_3} \end{bmatrix} = \begin{bmatrix} 0 & \Delta z_1 & -\Delta y_1 \\ -\Delta z_1 & 0 & \Delta x_1 \\ \Delta y_1 & -\Delta x_1 & 0 \end{bmatrix} \begin{bmatrix} \omega_1 \\ \omega_2 \\ \omega_3 \end{bmatrix} \tag{4.27}$$

This matrix is indeterminate (determinate = 0), therefore, we need to combine it with information found from the relative velocity of node $N8$ with respect to (wrt) node $N7$. Using the same approach above, we find the velocity equations for $N8$ wrt $N7$ to be

$$V_{8\hat{n}_1} = V_{7\hat{n}_1} + (\omega_2 \Delta z_2 - \omega_3 \Delta y_2) \tag{4.28}$$

$$V_{8\hat{n}_2} = V_{7\hat{n}_2} + (\omega_3 \Delta x_2 - \omega_1 \Delta z_2) \tag{4.29}$$

$$V_{8\hat{n}_3} = V_{7\hat{n}_3} + (\omega_1 \Delta y_2 - \omega_2 \Delta x_2) \tag{4.30}$$

where:

Δx_2- $\hat{n}_1$ difference between nodes $N8$ and $N7$
Δy_2- $\hat{n}_2$ difference between nodes $N8$ and $N7$
Δz_2- $\hat{n}_3$ difference between nodes $N8$ and $N7$

To get a determinate system of equations to find the ω's, we need to combine one of the equations for V_8 *wrt* V_7 with two of the equations for V_9 *wrt* V_7. We arbitrarily choose to use the $\hat{n}_1$ and $\hat{n}_2$ equations of V_9 *wrt* V_7 and the $\hat{n}_3$ equation of V_8 *wrt* V_7. The combined system, in matrix form, is then

$$\begin{bmatrix} V_{9\hat{n}_1} - V_{7\hat{n}_1} \\ V_{9\hat{n}_2} - V_{7\hat{n}_2} \\ V_{8\hat{n}_3} - V_{7\hat{n}_3} \end{bmatrix} = \begin{bmatrix} 0 & \Delta z_1 & -\Delta y_1 \\ -\Delta z_1 & 0 & \Delta x_1 \\ \Delta y_2 & -\Delta x_2 & 0 \end{bmatrix} \begin{bmatrix} \omega_1 \\ \omega_2 \\ \omega_3 \end{bmatrix} \tag{4.31}$$

The matrix is now determinate and we can find the angular velocities of the top plane as

$$\begin{bmatrix} \omega_1 \\ \omega_2 \\ \omega_3 \end{bmatrix} = \begin{bmatrix} 0 & \Delta z_1 & -\Delta y_1 \\ -\Delta z_1 & 0 & \Delta x_1 \\ \Delta y_2 & -\Delta x_2 & 0 \end{bmatrix}^{-1} \begin{bmatrix} V_{9\hat{n}_1} - V_{7\hat{n}_1} \\ V_{9\hat{n}_2} - V_{7\hat{n}_2} \\ V_{8\hat{n}_3} - V_{7\hat{n}_3} \end{bmatrix} \tag{4.32}$$

Once we have the angular velocities, we can then calculate the angular accelerations of the plane. We use the same method of writing the relative accelerations of the nodes of the plane, where

$$A_9 = A_7 + A_{9/7}$$

$$A_9 = A_7 + \{\alpha\} \times \{N_9 - N_7\} + (\{\omega\} \times (\{\omega\} \times \{N_9 - N_7\})) \tag{4.33}$$

where:

A_9 -	$\hat{n}_1, \hat{n}_2, \hat{n}_3$ components of the acceleration of node $N9$
A_7 -	$\hat{n}_1, \hat{n}_2, \hat{n}_3$ components of the acceleration of node $N7$
$\{\alpha\}$-	Vector of Newtonian angular accelerations ($\{\alpha_1\hat{n}_1\ \alpha_2\hat{n}_2\ \alpha_3\hat{n}_3\}$)
$\{\omega\}$-	Vector of Newtonian angular velocities as calculated above
$\{N_9 - N_7\}$-	Vector containing the $\hat{n}_1, \hat{n}_2, \hat{n}_3$ distance between the two nodes ($\{\Delta x_1 \Delta y_1 \Delta z_1\}$)

Evaluating the cross products results in three equations for the relative accelerations of the nodes, $N9$ and $N7$.

$$\begin{aligned}
A_{9\hat{n}_1} &= A_{7\hat{n}_1} + (\alpha_2\Delta z_1 - \alpha_3\Delta y_1) + [\omega_2(\omega_1\Delta y_1 - \omega_2\Delta x_1) - \\
&\quad \omega_3(\omega_3\Delta y_1 - \omega_1\Delta z_1)] \\
A_{9\hat{n}_2} &= A_{7\hat{n}_2} + (\alpha_3\Delta x_1 - \alpha_1\Delta z_1) + [\omega_3(\omega_2\Delta z_1 - \omega_3\Delta y_1) - \\
&\quad \omega_1(\omega_1\Delta y_1 - \omega_2\Delta x_1)] \\
A_{9\hat{n}_3} &= A_{7\hat{n}_3} + (\alpha_1\Delta y_1 - \alpha_2\Delta x_1) + [\omega_1(\omega_3\Delta x_1 - \omega_1\Delta z_1) - \\
&\quad \omega_2(\omega_2\Delta z_1 - \omega_3\Delta y_1)]
\end{aligned} \tag{4.34}$$

Once again, the system of equations is indeterminate, therefore, we need to add an independent equation from the relative acceleration of node $N8$ *wrt* $N7$. The equation will be in the same form as above, except that the relative displacements between $N8$ and $N7$ will be identified as $\{\Delta x_2\ \Delta y_2\ \Delta z_2\}$. The $\hat{n}_3$ equation will be combined with the $\hat{n}_1$ and $\hat{n}_2$ equations above. This will yield a matrix representation of the system given as

$$\begin{bmatrix} A_{9\hat{n}_1} - A_{7\hat{n}_1} - \beta_{11} \\ A_{9\hat{n}_2} - A_{7\hat{n}_2} - \beta_{21} \\ A_{8\hat{n}_3} - A_{7\hat{n}_3} - \beta_{32} \end{bmatrix} = \begin{bmatrix} 0 & \Delta z_1 & -\Delta y_1 \\ -\Delta z_1 & 0 & \Delta x_1 \\ \Delta y_2 & -\Delta x_2 & 0 \end{bmatrix} \begin{bmatrix} \alpha_1 \\ \alpha_2 \\ \alpha_3 \end{bmatrix} \tag{4.35}$$

where:

$$\begin{aligned}
\beta_{11} &= \omega_2(\omega_1\Delta y_1 - \omega_2\Delta x_1) - \omega_3(\omega_3\Delta x_1 - \omega_1\Delta z_1) \\
\beta_{21} &= \omega_3(\omega_2\Delta z_1 - \omega_3\Delta y_1) - \omega_1(\omega_1\Delta y_1 - \omega_2\Delta x_1) \\
\beta_{32} &= \omega_1(\omega_3\Delta x_2 - \omega_1\Delta z_2) - \omega_2(\omega_2\Delta z_2 - \omega_3\Delta y_2)
\end{aligned}$$

The matrix is now determinate and we can solve for the angular accelerations as

$$\begin{bmatrix} \alpha_1 \\ \alpha_2 \\ \alpha_3 \end{bmatrix} = \begin{bmatrix} 0 & \Delta z_1 & -\Delta y_1 \\ -\Delta z_1 & 0 & \Delta x_1 \\ \Delta y_2 & -\Delta x_2 & 0 \end{bmatrix}^{-1} \begin{bmatrix} A_{9\hat{n}_1} - A_{7\hat{n}_1} - \beta_{11} \\ A_{9\hat{n}_2} - A_{7\hat{n}_2} - \beta_{21} \\ A_{8\hat{n}_3} - A_{7\hat{n}_3} - \beta_{32} \end{bmatrix} \tag{4.36}$$

We find the angular accelerations in body fixed coordinates by performing a coordinate transformation using the cosine matrix. There are three members (M19, M20, and M21) in the top plane. The body fixed angular accelerations for member M19 are given by

$$\begin{bmatrix} \alpha_{19-1}\hat{b}_1 \\ \alpha_{19-2}\hat{b}_2 \\ \alpha_{19-3}\hat{b}_3 \end{bmatrix} = \begin{bmatrix} \mathbf{C^{19}} \end{bmatrix} \begin{bmatrix} \alpha_1\hat{n}_1 \\ \alpha_2\hat{n}_2 \\ \alpha_3\hat{n}_3 \end{bmatrix} \tag{4.37}$$

α_{19-1} and α_{19-2} are used to evaluate the EOMs for member M19. α_{19-3} is not used since the members are pinned at the ends and are free to rotate about the $\hat{b}_3$ axis. The body-fixed angular accelerations for members M20 and M21 are found using the cosine matrix for each member.

The above angular acceleration calculations are performed for each fixed-length member of the truss. The same analysis cannot be used to determine the angular accelerations of the active members since there is relative angular motion between the links of the midplane. For these members, the angular accelerations are calculated in body-fixed coordinates to begin with. The relative velocity and acceleration equations of the midplane members are more complicated than for the fixed-length

members since there is linear motion within the links. Consider member M7/8 which connects nodes N4 and N5. The relative velocity is given by

$$V_{5\hat{b}} = V_{4\hat{b}} + V_{5/4\hat{b}} \tag{4.38}$$

The relative velocity term, $V_{5/4\hat{b}}$, is not only a matter of angular velocity but also includes the linear velocity applied by the active member. Assuming that the angular velocity about the $\hat{b}_3$ axis is negligible (since it is a free degree of freedom), the velocity equation becomes

$$V_{5\hat{b}} = V_{4\hat{b}} + \dot{x}_7\hat{b}_3 + (\{\omega_{7-1}\hat{b}_1 + \omega_{7-2}\hat{b}_2\} \times \{\| N_5 - N_4 \| \hat{b}_3\}) \tag{4.39}$$

where:

$\dot{x}_7$= linear velocity applied by active member, M7/8

Evaluating the cross products and separating into like components yields

$$\begin{aligned} V_{5\hat{b}_1} &= V_{4\hat{b}_1} + \omega_{7-2} \| N_5 - N_4 \| \\ V_{5\hat{b}_2} &= V_{4\hat{b}_2} - \omega_{7-1} \| N_5 - N_4 \| \\ V_{5\hat{b}_3} &= V_{4\hat{b}_3} + \dot{x}_7 \end{aligned} \tag{4.40}$$

From these equations, we can immediately find ω_{7-1} and ω_{7-2} as

$$\omega_{7-1} = \frac{V_{4\hat{b}_2} - V_{5\hat{b}_2}}{\|N_5 - N_4\|} \tag{4.41}$$

$$\omega_{7-2} = \frac{V_{5\hat{b}_1} - V_{4\hat{b}_1}}{\|N_5 - N_4\|} \tag{4.42}$$

To calculate the relative acceleration, we assume that the angular acceleration about

the $\hat{b}_3$ axis is also zero. The acceleration equation is then

$$A_{5\hat{b}} = A_{4\hat{b}} + A_{5/4\hat{b}} \tag{4.43}$$

The acceleration of N5 *wrt* N4 is found by differentiating the relative velocity, $V_{5/4\hat{b}}$, given as

$$V_{5/4\hat{b}} = \dot{x}_7\hat{b}_3 + (\{\omega_b\} \times \{\| N_5 - N_4 \| \hat{b}_3\}) \tag{4.44}$$

The differential acceleration is then

$$A_{5/4\hat{b}} = \ddot{x}_7\hat{b}_3 + \{\omega_b \times \dot{x}_7\} + \{\alpha_b \times \| N_5 - N_4 \|\} + \{\omega_b \times (\dot{x}_7 + \omega_b \times \| N_5 - N_4 \|)\} \tag{4.45}$$

Substituting into Equation 4.43, combining like coordinates and evaluating the cross products results in the three acceleration equations

$$A_{5\hat{b}_1} = A_{4\hat{b}_1} + \alpha_{7-2} \| N_5 - N_4 \| + 2\omega_{7-2}\dot{x}_7 \tag{4.46}$$

$$A_{5\hat{b}_2} = A_{4\hat{b}_2} - \alpha_{7-1} \| N_5 - N_4 \| - 2\omega_{7-1}\dot{x}_7 \tag{4.47}$$

$$A_{5\hat{b}_3} = A_{4\hat{b}_3} + \ddot{x}_7 - \omega_{7-1}^2 \| N_5 - N_4 \| - \omega_{7-2}^2 \| N_5 - N_4 \| \tag{4.48}$$

From the $\hat{b}_1$ and $\hat{b}_2$ equations, we can solve for α_{7-1} and α_{7-2} where

$$\alpha_{7-1}\hat{b}_1 = \frac{A_{4b_2} - A_{5b_2} - 2\omega_{7-1}\dot{x}_7}{\|N_5 - N_4\|} \tag{4.49}$$

$$\alpha_{7-2}\hat{b}_2 = \frac{A_{5b_1} - A_{4b_1} - 2\omega_{7-2}\dot{x}_7}{\|N_5 - N_4\|} \tag{4.50}$$

(4.51)

We can find the angular accelerations in body-fixed coordinates for each of the active members. Using the above analysis, the angular accelerations for each of the truss members is known. In order to evaluate the EOMs, we next need to calculate the acceleration of the centers of gravity of each of the members.

4.4.2 Center of Gravity Accelerations

The accelerations of the centers of gravity of the members depend on the nodal accelerations supplied by the kinematics analysis. As with the angular accelerations, the center of gravity acceleration calculation depends on the type of member, fixed-length or active. First consider the center of gravity acceleration of a fixed-length member.

Consider fixed-length member M17 (connecting nodes N7 and N9). The acceleration equation for the two nodes (in global coordinates) is given in Equation 4.33. Since the member is fixed-length and constant density, the center of gravity is the midpoint of the member. We denote the center of gravity as CG13. The distance between CG13 and node N7 one half the distance between nodes N7 and N9. Therefore, using the notation that $N_9 - N_7 = \{\Delta x_1 \; \Delta y_1 \; \Delta z_1\}$ we can immediately write that the distance between CG13 and N7 is

$$\{CG13 - N_7\} = \left\{ \frac{\Delta x_1}{2} \; \frac{\Delta y_1}{2} \; \frac{\Delta z_1}{2} \right\} \tag{4.52}$$

We can now find the acceleration of the center of gravity as

$$
\begin{aligned}
A_{CG13\hat{n}_1} = \; & A_{7\hat{n}_1} + (\alpha_2\frac{\Delta z_1}{2} - \alpha_3\frac{\Delta y_1}{2}) + [\omega_2(\omega_1\frac{\Delta y_1}{2} - \omega_2\frac{\Delta x_1}{2}) - \\
& \omega_3(\omega_3\frac{\Delta y_1}{2} - \omega_1\frac{\Delta z_1}{2})] \\
A_{CG13\hat{n}_2} = \; & A_{7\hat{n}_2} + (\alpha_3\frac{\Delta x_1}{2} - \alpha_1\frac{\Delta z_1}{2}) + [\omega_3(\omega_2\frac{\Delta z_1}{2} - \omega_3\frac{\Delta y_1}{2}) - \\
& \omega_1(\omega_1\frac{\Delta y_1}{2} - \omega_2\frac{\Delta x_1}{2})] \\
A_{CG13\hat{n}_3} = \; & A_{7\hat{n}_3} + (\alpha_1\frac{\Delta y_1}{2} - \alpha_2\frac{\Delta x_1}{2}) + [\omega_1(\omega_3\frac{\Delta x_1}{2} - \omega_1\frac{\Delta z_1}{2}) - \\
& \omega_2(\omega_2\frac{\Delta z_1}{2} - \omega_3\frac{\Delta y_1}{2})]
\end{aligned}
\tag{4.53}
$$

which is equivalent to

$$
A_{CG13} = A_7 + \frac{A_{9/7}}{2} \tag{4.54}
$$

where:

$A_{9/7}$= $A_9 - A_7$

Therefore, we find the equation of the center of gravity for member M13 to be

$$
A_{CG13\hat{n}_1} = A_{7\hat{n}_1} + \frac{A_{9\hat{n}_1} - A_{7\hat{n}_1}}{2} = \frac{A_{7\hat{n}_1} + A_{9\hat{n}_1}}{2} \tag{4.55}
$$

$$
A_{CG13\hat{n}_2} = A_{7\hat{n}_2} + \frac{A_{9\hat{n}_2} - A_{7\hat{n}_2}}{2} = \frac{A_{7\hat{n}_2} + A_{9\hat{n}_2}}{2} \tag{4.56}
$$

$$
A_{CG13\hat{n}_3} = A_{7\hat{n}_3} + \frac{A_{9\hat{n}_3} - A_{7\hat{n}_3}}{2} = \frac{A_{7\hat{n}_3} + A_{9\hat{n}_3}}{2} \tag{4.57}
$$

This analysis is valid for each of the fixed-length members, including those connected to the fixed base. In that case the base node acceleration will be 0.

The calculation for the center of gravity acceleration of the active members is more complex. This is because that, due to the leadscrew tube translating over the active base, the center of gravity is changing at each time step. Consider the schematic of the active base/leadscrew tube shown in Fig. 4.4. The total length of the active member at a given timestep is L_i. Point A is the connection of the active base with a node, point B is the connection of the leadscrew tube and a node. The length of the leadscrew tube is a constant value of L_{ls} and its center of gravity is located at a distance $\frac{L_{ls}}{2}$ from point B. The center of gravity of the active base is located a distance L_{CGbase} from point A. The combined center of gravity of the active base/leadscrew tube is located at the distance CG_{tot} from point A. The leadscrew tube has a mass of m_{ls} and the active base mass is m_{ab}. The distance CG_{tot} is found as

$$CG_{tot} = \frac{m_{ls}(L_i - \frac{L_{ls}}{2}) + m_{ab}L_{CGbase}}{m_{ls} + m_{ab}} \tag{4.58}$$

As opposed to the fixed-length members where the center of gravity is $\frac{1}{2}$ the total length, the active member center of gravity is a fraction $\frac{CG_{tot}}{L_i}$ of the total length. The acceleration of the center of gravity of the active member is then given as

$$A_{CGact} = A_A + \frac{CG_{tot}(A_B - A_A)}{L_i} \tag{4.59}$$

For member M7/8, points A and B correspond to nodes N4 and N5 respectively, and the acceleration becomes

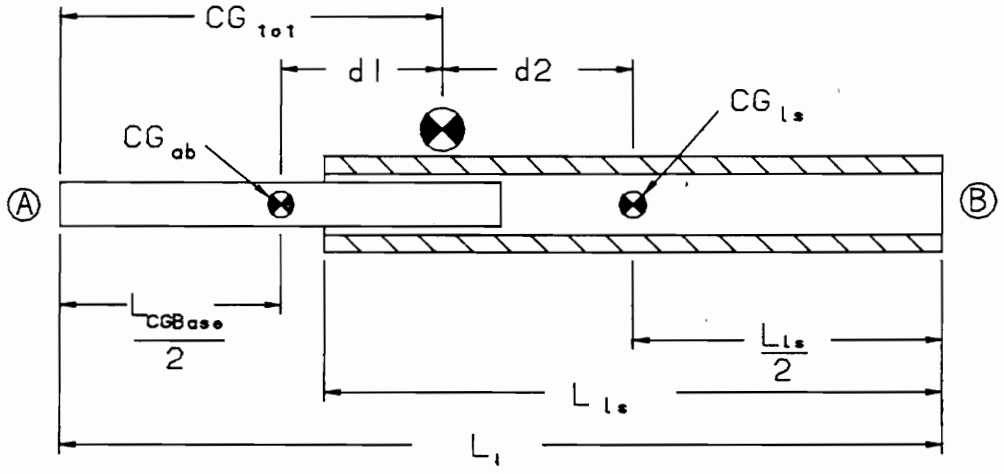

Figure 4.4: Schematic of Active Member

$$A_{CG7/8} = A_4 + \frac{CG_{7/8}(A_5 - A_4)}{L_{7/8}} \tag{4.60}$$

We have now calculated all of the center of gravity and angular accelerations for the members of the truss. The only values left to calculate before we can solve for the unknown nodal forces are the moments of inertia and the masses.

4.5 Calculation of Member Constants

The fixed-length members of the truss are all assumed to be hollow tubes. The tube dimensions are input when the simulation is run, and the values for mass and moment of inertia are calculated at that time. The only complicated calculation is that for the moment of inertia of the active members. Since the length and center of gravity of the active member is changing, the moment of inertia will change at each time step. The distance from the total center of gravity (CG_{tot}) to the center of gravity of the active base (CG_{ab}), shown in Fig. 4.4, is given as $d1$. The distance from the total center of gravity to the center of gravity of the leadscrew tube (CG_{ls}) is $d2$. The moments of inertia of the leadscrew tube (I_{ls}) and active base (I_{ab}) about their respective centers of gravity are constant. We then use the parallel axis theorem to find the moment of inertia of the active member about CG_{tot} as

$$I_{CG_{tot}} = I_{ls} + I_{ab} + m_{ls}d1^2 + m_{ab}d2^2 \tag{4.61}$$

Since the value of $d1$ and $d2$ will change at each time step, the moment of inertia of the active member will also change. Knowing all of the member properties, center

of gravity accelerations, and angular accelerations, we are now ready to solve the system of equations for the unknown nodal forces.

4.6 Solution of the Equations of Motion

At this point, we have the system of equations for the overall truss model, and we have calculated all of the center of gravity accelerations, member angular accelerations, masses and moments of inertia. The nodal forces are found by multiplying the inverse of the system matrix by the known inertial forces (acceleration forces) as

$$\{F\} = [SM]^{-1}\{M\ddot{X}\} \tag{4.62}$$

This results in a nodal force vector containing 117 forces (6 nodal forces for each of the 18 members, and 3 reaction forces for each of the 3 base nodes). The identification of the forces and their locations within the nodal force vector is given in Table 4.1.

Once we have the nodal forces from the overall truss model, we need to look closer at the active member simulation.

4.7 Active Member Analysis

As stated above, during the overall truss simulation, the active members are each modelled as a single member with varying length, center of gravity, and moment of inertia. In actuality, each active member consists of two links, an active base and a leadscrew tube, with a cantilever connection between them. The location of the

Table 4.1: Identification of Forces in Nodal Force Vector

MEMBER	NODE	DIRECTION	ROW OF FORCE VECTOR	FORCE IDENTIFICATION
1	6	$\hat{n}_1$	1	F_{x_1}
	6	$\hat{n}_2$	2	F_{y_1}
	6	$\hat{n}_3$	3	F_{z_1}
	2	$\hat{n}_1$	4	F_{x_2}
	2	$\hat{n}_2$	5	F_{y_2}
	2	$\hat{n}_3$	6	F_{z_2}
2	6	$\hat{n}_1$	7	F_{x_3}
	6	$\hat{n}_2$	8	F_{y_3}
	6	$\hat{n}_3$	9	F_{z_3}
	1	$\hat{n}_1$	10	F_{x_4}
	1	$\hat{n}_2$	11	F_{y_4}
	1	$\hat{n}_3$	12	F_{z_4}
3	4	$\hat{n}_1$	13	F_{x_5}
	4	$\hat{n}_2$	14	F_{y_5}
	4	$\hat{n}_3$	15	F_{z_5}
	1	$\hat{n}_1$	16	F_{x_6}
	1	$\hat{n}_2$	17	F_{y_6}
	1	$\hat{n}_3$	18	F_{z_6}
4	4	$\hat{n}_1$	19	F_{x_7}
	4	$\hat{n}_2$	20	F_{y_7}
	4	$\hat{n}_3$	21	F_{z_7}
	3	$\hat{n}_1$	22	F_{x_8}
	3	$\hat{n}_2$	23	F_{y_8}
	3	$\hat{n}_3$	24	F_{z_8}
5	5	$\hat{n}_1$	25	F_{x_9}
	5	$\hat{n}_2$	26	F_{y_9}
	5	$\hat{n}_3$	27	F_{z_9}
	3	$\hat{n}_1$	28	$F_{x_{10}}$
	3	$\hat{n}_2$	29	$F_{y_{10}}$
	3	$\hat{n}_3$	30	$F_{z_{10}}$
6	5	$\hat{n}_1$	31	$F_{x_{11}}$
	5	$\hat{n}_2$	32	$F_{y_{11}}$
	5	$\hat{n}_3$	33	$F_{z_{11}}$
	2	$\hat{n}_1$	34	$F_{x_{12}}$
	2	$\hat{n}_2$	35	$F_{y_{12}}$
	2	$\hat{n}_3$	36	$F_{z_{12}}$

Table 4.1 (cont): Identification of Forces in Nodal Force Vector

MEMBER	NODE	DIRECTION	ROW OF FORCE VECTOR	FORCE IDENTIFICATION
7/8	4	$\hat{n}_1$	37	$F_{x_{13}}$
	4	$\hat{n}_2$	38	$F_{y_{13}}$
	4	$\hat{n}_3$	39	$F_{z_{13}}$
	5	$\hat{n}_1$	40	$F_{x_{14}}$
	5	$\hat{n}_2$	41	$F_{y_{14}}$
	5	$\hat{n}_3$	42	$F_{z_{14}}$
9/10	5	$\hat{n}_1$	43	$F_{x_{15}}$
	5	$\hat{n}_2$	44	$F_{y_{15}}$
	5	$\hat{n}_3$	45	$F_{z_{15}}$
	6	$\hat{n}_1$	46	$F_{x_{16}}$
	6	$\hat{n}_2$	47	$F_{y_{16}}$
	6	$\hat{n}_3$	48	$F_{z_{16}}$
11/12	4	$\hat{n}_1$	49	$F_{x_{17}}$
	4	$\hat{n}_2$	50	$F_{y_{17}}$
	4	$\hat{n}_3$	51	$F_{z_{17}}$
	6	$\hat{n}_1$	52	$F_{x_{18}}$
	6	$\hat{n}_2$	53	$F_{y_{18}}$
	6	$\hat{n}_3$	54	$F_{z_{18}}$
13	6	$\hat{n}_1$	55	$F_{x_{19}}$
	6	$\hat{n}_2$	56	$F_{y_{19}}$
	6	$\hat{n}_3$	57	$F_{z_{19}}$
	9	$\hat{n}_1$	58	$F_{x_{20}}$
	9	$\hat{n}_2$	59	$F_{y_{20}}$
	9	$\hat{n}_3$	60	$F_{z_{20}}$
14	6	$\hat{n}_1$	61	$F_{x_{21}}$
	6	$\hat{n}_2$	62	$F_{y_{21}}$
	6	$\hat{n}_3$	63	$F_{z_{21}}$
	7	$\hat{n}_1$	64	$F_{x_{22}}$
	7	$\hat{n}_2$	65	$F_{y_{22}}$
	7	$\hat{n}_3$	66	$F_{z_{22}}$
15	4	$\hat{n}_1$	67	$F_{x_{23}}$
	4	$\hat{n}_2$	68	$F_{y_{23}}$
	4	$\hat{n}_3$	69	$F_{z_{23}}$
	7	$\hat{n}_1$	70	$F_{x_{24}}$
	7	$\hat{n}_2$	71	$F_{y_{24}}$
	7	$\hat{n}_3$	72	$F_{z_{24}}$

Table 4.1 (cont): Identification of Forces in Nodal Force Vector

MEMBER	NODE	DIRECTION	ROW OF FORCE VECTOR	FORCE IDENTIFICATION
16	4	$\hat{n}_1$	73	$F_{x_{25}}$
	4	$\hat{n}_2$	74	$F_{y_{25}}$
	4	$\hat{n}_3$	75	$F_{z_{25}}$
	8	$\hat{n}_1$	76	$F_{x_{26}}$
	8	$\hat{n}_2$	77	$F_{y_{26}}$
	8	$\hat{n}_3$	78	$F_{z_{26}}$
17	5	$\hat{n}_1$	79	$F_{x_{27}}$
	5	$\hat{n}_2$	80	$F_{y_{27}}$
	5	$\hat{n}_3$	81	$F_{z_{27}}$
	8	$\hat{n}_1$	82	$F_{x_{28}}$
	8	$\hat{n}_2$	83	$F_{y_{28}}$
	8	$\hat{n}_3$	84	$F_{z_{28}}$
18	5	$\hat{n}_1$	85	$F_{x_{29}}$
	5	$\hat{n}_2$	86	$F_{y_{29}}$
	5	$\hat{n}_3$	87	$F_{z_{29}}$
	9	$\hat{n}_1$	88	$F_{x_{30}}$
	9	$\hat{n}_2$	89	$F_{y_{30}}$
	9	$\hat{n}_3$	90	$F_{z_{30}}$
19	7	$\hat{n}_1$	91	$F_{x_{31}}$
	7	$\hat{n}_2$	92	$F_{y_{31}}$
	7	$\hat{n}_3$	93	$F_{z_{31}}$
	9	$\hat{n}_1$	94	$F_{x_{32}}$
	9	$\hat{n}_2$	95	$F_{y_{32}}$
	9	$\hat{n}_3$	96	$F_{z_{32}}$
20	8	$\hat{n}_1$	97	$F_{x_{33}}$
	8	$\hat{n}_2$	98	$F_{y_{33}}$
	8	$\hat{n}_3$	99	$F_{z_{33}}$
	7	$\hat{n}_1$	100	$F_{x_{34}}$
	7	$\hat{n}_2$	101	$F_{y_{34}}$
	7	$\hat{n}_3$	102	$F_{z_{34}}$
21	8	$\hat{n}_1$	103	$F_{x_{35}}$
	8	$\hat{n}_2$	104	$F_{y_{35}}$
	8	$\hat{n}_3$	105	$F_{z_{35}}$
	9	$\hat{n}_1$	106	$F_{x_{36}}$
	9	$\hat{n}_2$	107	$F_{y_{36}}$
	9	$\hat{n}_3$	108	$F_{z_{36}}$

Table 4.1 (cont): Identification of Forces in Nodal Force Vector

MEMBER	NODE	DIRECTION	ROW OF FORCE VECTOR	FORCE IDENTIFICATION
Reaction Forces	1	$\hat{n}_1$	109	R_{x_1}
	1	$\hat{n}_2$	110	R_{y_1}
	1	$\hat{n}_3$	111	R_{z_1}
	2	$\hat{n}_1$	112	R_{x_2}
	2	$\hat{n}_2$	113	R_{y_2}
	2	$\hat{n}_3$	114	R_{z_2}
	3	$\hat{n}_1$	115	R_{x_3}
	3	$\hat{n}_2$	116	R_{y_3}
	3	$\hat{n}_3$	117	R_{z_3}

cantilever connection moves along the length of the active base during the motion. It was shown above that modeling the active members as two links in the overall truss model adds three extra unknown forces but no extra constraint equations. Therefore, the overall truss model is solved modeling the active members as one link. The overall truss model will provide the nodal forces acting at each end of an active member. Knowing the location of the cantilever connection and that the leadscrew tube is constant length, we use the nodal forces to find the forces acting at the cantilever connection.

There are 3 forces and 2 moments acting at the cantilever connection. The 3 forces can be assumed to be acting either in the $\hat{n}$ coordinate system or in the $\hat{b}$ coordinate system. Since the principle moments of inertia are in the $\hat{b}$ coordinate system, it is less complex to use that system to evaluate the moment equations at the cantilever connection. The $\hat{b}$ forces are found by using the cosine matrix for the member. The $\hat{n}$ forces are used to evaluate the force equations. The FBD for the leadscrew tube of member M7/8 is given in Fig. 4.5. Summing the forces in the $\hat{n}$ axes and the torques in the $\hat{b}$ axes results in the 5 EOMs

$$\sum F_{\hat{n}_1} \Rightarrow \quad m_{ls} a_{CG_{ls}} \hat{n}_1 = F_{x_{14}} + F_{x_{ls}} \tag{4.63}$$

$$\sum F_{\hat{n}_2} \Rightarrow \quad m_{ls} a_{CG_{ls}} \hat{n}_2 = F_{y_{14}} + F_{y_{ls}} \tag{4.64}$$

$$\sum F_{\hat{n}_3} \Rightarrow \quad m_{ls} a_{CG_{ls}} \hat{n}_3 = F_{z_{14}} + F_{z_{ls}} - m_{ls} g \tag{4.65}$$

$$\sum T_{\hat{b}_1} \Rightarrow \quad I_{ls} \alpha_{7-1} = \frac{L_{ls}}{2}(F_{21} - F_{22}) + M_{7-1} \tag{4.66}$$

$$\sum T_{\hat{b}_2} \Rightarrow \quad I_{ls} \alpha_{7-2} = \frac{L_{ls}}{2}(F_{12} - F_{11}) - M_{7-2} \tag{4.67}$$

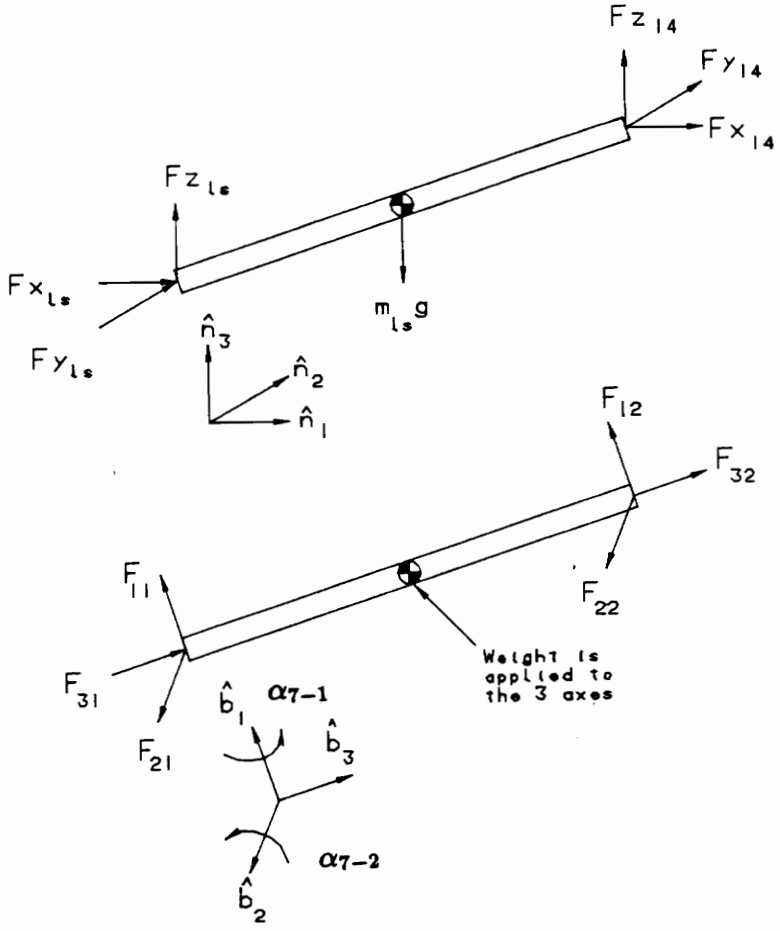

Figure 4.5: Free Body Diagram for Leadscrew Tube

where:

$F_{x_{l_s}}$= Force at the cantilever connection in the $\hat{n}_1$ direction
$F_{y_{l_s}}$= Force at the cantilever connection in the $\hat{n}_2$ direction
$F_{z_{l_s}}$= Force at the cantilever connection in the $\hat{n}_3$ direction
M_{7-1}= Moment at the cantilever connection about the $\hat{b}_1$ axis
M_{7-2}= Moment at the cantilever connection about the $\hat{b}_2$ axis

and:

$$\begin{bmatrix} F_{11\hat{b}_1} \\ F_{21\hat{b}_2} \\ F_{31\hat{b}_3} \end{bmatrix} = \begin{bmatrix} \mathbf{C^{7/8}} \end{bmatrix} \begin{bmatrix} F_{x_{l_s}\hat{n}_1} \\ F_{y_{l_s}\hat{n}_2} \\ F_{z_{l_s}\hat{n}_3} \end{bmatrix}$$

$$\begin{bmatrix} F_{12\hat{b}_1} \\ F_{22\hat{b}_2} \\ F_{32\hat{b}_3} \end{bmatrix} = \begin{bmatrix} \mathbf{C^{7/8}} \end{bmatrix} \begin{bmatrix} F_{x_{14}\hat{n}_1} \\ F_{y_{14}\hat{n}_2} \\ F_{z_{14}\hat{n}_3} \end{bmatrix}$$

This provides a system of 5 equations with 5 unknowns, therefore, we can solve for the forces and moments at the cantilever connection. We note that there is no torque about the $\hat{b}_3$ axis. This is because the member is pinned at the node and the entire active member is free to rotate about the axis. In a physical truss, there is a torque about the $\hat{b}_3$ axis that is caused, on one end, by the leadscrew force, and on the other end, by a constraint in the offset joint. These torques are not included in this model because we do not have the kinematics for the offset joints and because the moment of inertia of member about the $\hat{b}_3$ axis is very small. The cantilever connection forces will be used in a later section on the development of the actuator model.

4.8 Internal Forces

As in the case of the triangular truss, we are interested in finding the internal forces generated by both the static and dynamic loading on the truss members. The truss members are assumed to be rigid, therefore, the internal forces must withstand the tendency of the member to deform under static and dynamic loads. The evaluation of the internal forces is quite similar to that of the cantilever forces in the active members. We first "cut" the member at some point along its length, draw the FBD for the member section, apply Newton's laws of motion, and develop the equations of motion for the section. There are 3 internal forces and 2 internal moments that we are interested in finding. The forces consist of a tensile force in the $\hat{b}_3$ direction and two shear forces in the $\hat{b}_1$ and $\hat{b}_2$ directions. The 2 moments are those about the $\hat{b}_1$ and $\hat{b}_2$ axes.

As an example, consider the member M19 which connects nodes N7 and N9. To find the internal forces at the midpoint of the member, we draw a FBD as in Fig. 4.6. We are interested in the forces in the $\hat{b}$ axes since we will ultimately use the internal forces to find stresses and strains and to check buckling loads. Summing the forces in the $\hat{b}$ coordinates yields

$$\sum F_{\hat{b}_1} \Rightarrow \frac{m_{19}}{2} a_{CGsect} \hat{b}_1 = F_1 - V_{1-19} - C^{19}_{1,3} \frac{m_{19}}{2} g \tag{4.68}$$

$$\sum F_{\hat{b}_2} \Rightarrow \frac{m_{19}}{2} a_{CGsect} \hat{b}_2 = F_2 - V_{2-19} - C^{19}_{2,3} \frac{m_{19}}{2} g \tag{4.69}$$

$$\sum F_{\hat{b}_3} \Rightarrow \frac{m_{19}}{2} a_{CGsect} \hat{b}_3 = F_3 - T_{19} - C^{19}_{3,3} \frac{m_{19}}{2} g \tag{4.70}$$

$$\sum T_{\hat{b}_1} \Rightarrow I_{sect} \alpha_{19-1} = \frac{L_{19}}{4}(-F_2 - V_{2-19}) + M_{19-1} \tag{4.71}$$

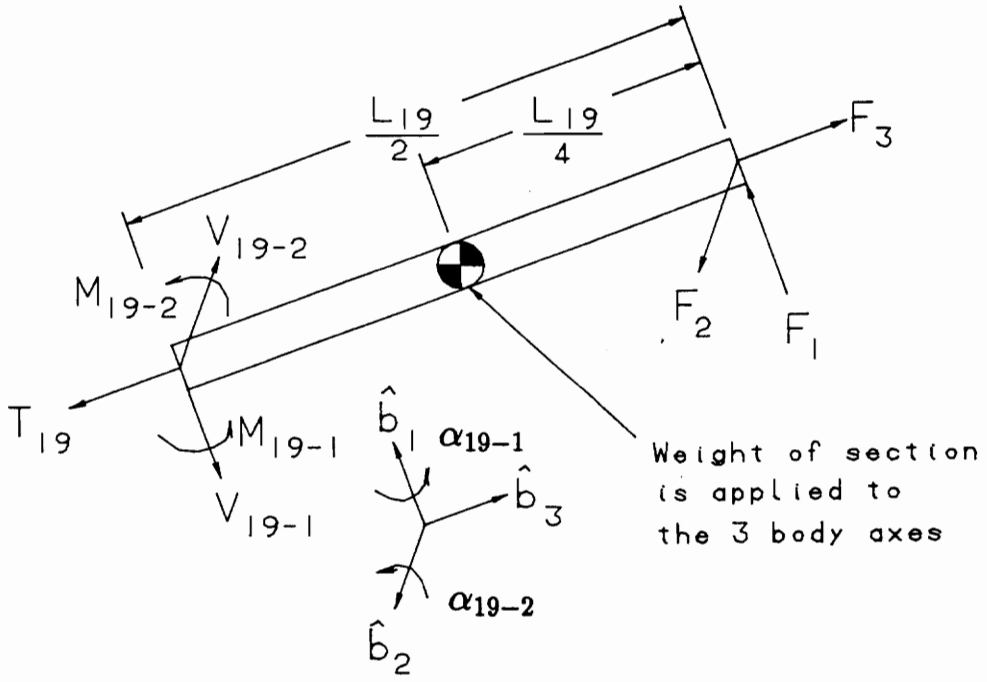

Figure 4.6: Free Body Diagram for Internal Force Evaluation

$$\sum T_{\hat{b}_2} \Rightarrow I_{sect}\alpha_{19-2} = \frac{L_{19}}{4}(F_1 + V_{1-19}) - M_{19-2} \tag{4.72}$$

where:

$$\begin{bmatrix} F_{1\hat{b}_1} \\ F_{2\hat{b}_2} \\ F_{3\hat{b}_3} \end{bmatrix} = \begin{bmatrix} \mathbf{C}^{7/8} \end{bmatrix} \begin{bmatrix} F_{x_{30}\hat{n}_1} \\ F_{y_{30}\hat{n}_2} \\ F_{z_{30}\hat{n}_3} \end{bmatrix} \tag{4.73}$$

Using the acceleration analysis described in Chapter 4.4.2 above, the acceleration of the center of gravity of the section is found to be

$$\begin{bmatrix} A_{CGsect}\hat{b}_1 \\ F_{CGsect}\hat{b}_2 \\ F_{CGsect}\hat{b}_3 \end{bmatrix} = \begin{bmatrix} \mathbf{C}^{19} \end{bmatrix} \begin{bmatrix} A_7\hat{n}_1 + \frac{3(A_9\hat{n}_1 - A_7\hat{n}_1)}{4} \\ A_7\hat{n}_2 + \frac{3(A_9\hat{n}_2 - A_7\hat{n}_2)}{4} \\ A_7\hat{n}_3 + \frac{3(A_9\hat{n}_3 - A_7\hat{n}_3)}{4} \end{bmatrix} \tag{4.74}$$

Once again, since we have a set of 5 equations and 5 unknowns, we can solve for the internal forces of the section. Using these internal forces, we can then evaluate the stresses generated in the members.

4.9 Stress Analysis

The total stress at any point in the truss members is caused by a combination of the forces acting at that point. The forces that cause the stress are the internal forces of axial load, shear, and bending moments. The development of the individual stresses caused by each type of force was presented in Chapter 2.7 and will not be repeated here. The only difference is that we now have two shear forces and two bending moments acting in orthogonal axes. These additional forces will add terms to the Von Mises stresses calculated at the two stress points. We will develop the Von

Mises stresses for one member; the stresses in the other members are calculated in the same member.

Consider the general cross section of a hollow tube member given in Fig. 4.7. The axial load in the member is coming out of the paper, the two shear forces and the two bending moments are acting as indicated. The are two points where the Von Mises stress is evaluated, S_1 and S_2. (Note: S_1 and S_2 can be on either side of the member - on one side the bending moment will be the opposite sign of the axial load while on the other side they will be of the same sign. We use the absolute values of the axial and bending moment stresses to get the maximum values when developing the von Mises stress.) At point S_1, the bending stress caused by moment M1 is a maximum and the bending stress caused by M2 is 0. Also, the shear stress caused by V1 is a maximum and the shear stress caused by V2 is 0. The tensile stress is constant over the cross section. The point S_1 is chosen so that the tensile stress and the bending stress is additive (i.e., both cause tension). The Von Mises stress at this point is then given by

$$\sigma_{S_1 max} = \sqrt{(abs(\sigma_{M1}) + abs(\sigma_T))^2 + 3\tau_{V1}^2} \tag{4.75}$$

Likewise, at point S_2, the bending stress caused by M1 and the shear stress caused by V1 are both 0 while the bending stress caused by M2 and the shear stress caused by V2 are both maximums. The Von Mises stress at this point is given as

$$\sigma_{S_2 max} = \sqrt{(abs(\sigma_{M2}) + abs(\sigma_T))^2 + 3\tau_{V2}^2} \tag{4.76}$$

Since the shear stress is so small compared to the other stresses, the maximum stress

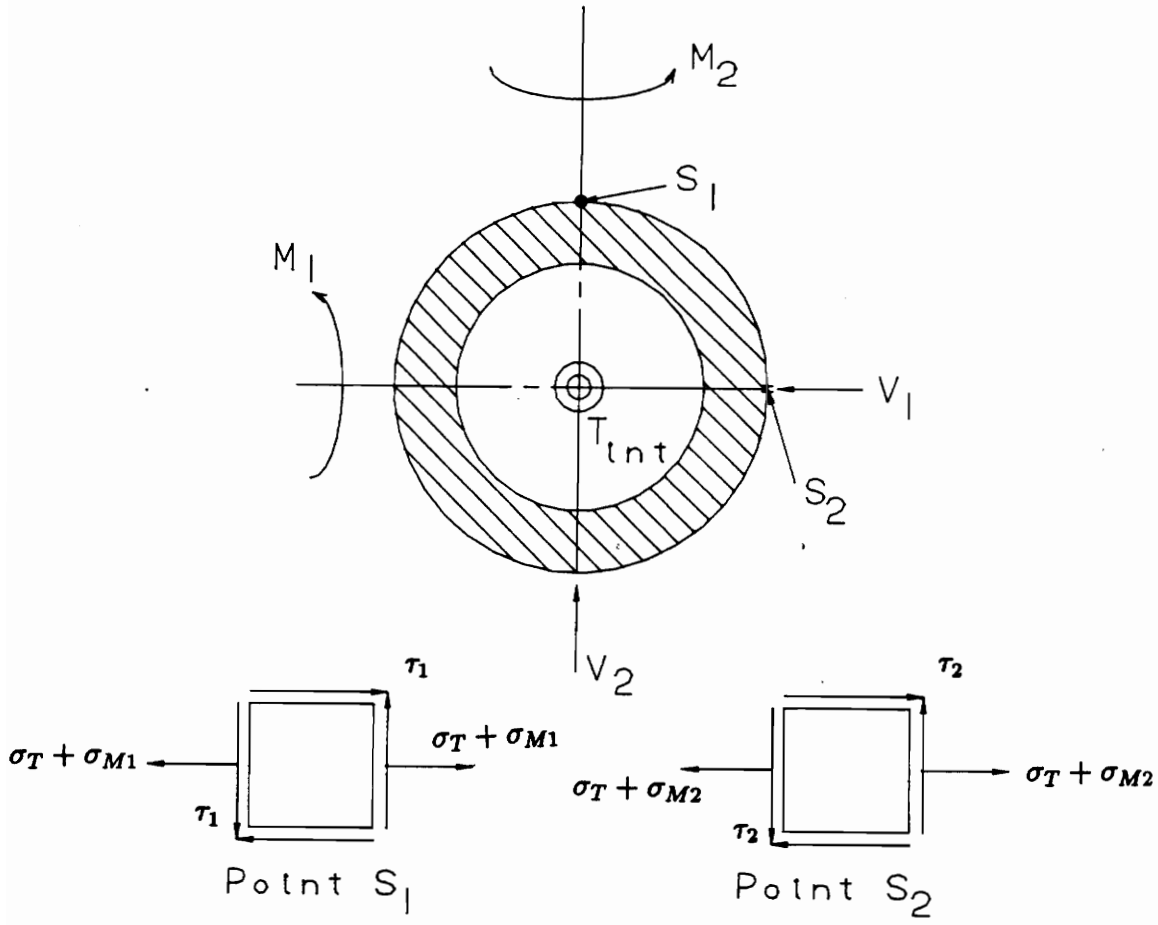

Figure 4.7: Cross Section of Truss Member Showing Internal Forces

occurs where the bending moment is a maximum. The internal stresses are found at the midpoint of each of the fixed-length members and at the cantilever connection of the leadscrew tube. The maximum stresses found during the total range of the motion are compared to the material yield stress to determine if the member will experience elastic failure. The members must also be checked for failure due to column buckling.

4.10 Buckling Analysis

The buckling analysis for the members of the spatial truss is the same as that presented in Section 2.8 for the members of the planar truss. The critical axial load, under which the member will experience buckling failure, is given as

$$P_{cr} = \frac{C\pi^2 EA}{(\frac{l}{k})^2} \tag{4.77}$$

where:

C is a constant based on the end conditions of the member
E is the modulus of elasticity of the material
A is the cross-sectional area of the member
l is the length of the member
k is the radius of gyration

For the fixed-length members of the truss, the ends are pinned and the end condition constant is 1.0. For the leadscrew tubes, where there is a pinned connection at one end and a cantilever connection at the other end, the end condition constant is also 1.0.

4.11 Actuator Model

The actuator model used for the spatial case is the same as that developed in Section 2.9 for the planar truss. However, instead of only one actuator, we have 3 in the spatial truss.

Each active member consists of a motor, gearhead, leadscrew, and supports. Each active member provides a pure kinematic input to the truss, consisting of a displacement, linear velocity, and linear acceleration of each of the leadscrew tubes. The motion of these leadscrew tubes then causes the motion of the truss. By solving for the forces at the connection of the leadscrew tube and the leadscrew, we can also find the linear force that the leadscrew applies to the leadscrew tube. This analysis was done in Section 4.7 above. Using the applied forces, we can calculate the torque requirements for each of the motors.

The torque provided by each motor must be enough to overcome the armature inertia and the friction force, and provide the linear force necessary to cause the specified displacement, velocity, and acceleration at each time step. The torque requirement for a single motor was defined in Section 2.9. The same DC voltage-controlled motor model used for the planar truss is used for the spatial truss.

Since the motor/torque relationships are the same for both the planar and the spatial trusses, the development of the actuator model will not be presented here again. The resulting equations specifying the motor torque, voltage, and current are repeated below, however, the reader is referred to Section 2.9 for the development of

the model.

The torque that must be developed by each motor to overcome the inertia of the armature, the friction in the leadscrew, and the force applied to the leadscrew tube (assuming the torque is acting against the load) is

$$T_m = J_m\ddot{\theta}_m + \frac{Fd_m}{2}\left(\frac{l + \pi\mu d_m sec\alpha}{\pi d_m - \mu l sec\alpha}\right) \tag{4.78}$$

The voltage required by each motor is

$$V_m = \frac{T_m R_a}{K_t} + K_b\dot{\theta}_m \tag{4.79}$$

Finally, the current required by each motor is

$$T_m = I_a K_t \tag{4.80}$$

4.12 Summary

In this chapter, we have extended the analysis of the planar, triangular truss to the case of a three dimensional, double-octahedral truss. The same basic approach was used for both cases, however, the increased number of components of the spatial truss resulted in a much larger model. A kinematic analysis was performed to get the position, velocity, and acceleration of all of the nodes in the truss. A set of equations defining the motion of all of the members of the truss was then developed and solved to find the forces acting on each member at each node. Using these nodal

forces and accelerations, the maximum stresses in the members were found. Finally, the motor parameters required to supply the specified motion were calculated.

The next step is to develop a simulation program that will implement the dynamic model for the spatial truss. The MATLAB command language was used to develop the simulation. The simulation and results are presented in the next chapter.

Chapter 5

Spatial Truss Simulation and Experimental Results

A simulation of the spatial truss model developed in Chapter 4 was implemented using MATLAB. The simulation program allows the user to input the dimensions of the truss members, material properties, top plane motion specifications, and simulation time. Gravity can be be applied to the truss and act either toward the base (for a truss standing up) of the truss or toward the top plane (for a truss hanging downward). In addition, the user can apply lumped masses or external forces to the nodes of the truss top plane. The listing of the simulation program is provided in Appendix C.

Two different methods were used to check the simulation program. First a static simulation was performed using the MATLAB code. The results of the static case were then checked against a finite-element analysis. The finite-element code used for this analysis was MSC/PAL. To check the dynamic solution of the MATLAB simulation code, a prototype truss available at VPI&SU was instrumented with strain gages. A slew maneuver was then performed on the truss; the resulting strains on sample members were recorded and compared to the values predicted by the MAT-

LAB code.

All simulations were performed using the dimensions, member properties, and active member characteristics of the VPI&SU prototype truss. These values are presented in Table 5.1.

Table 5.1: VPI&SU Prototype Truss Values

Fixed Triangle Member Length	0.9615 m
Cross Longeron Length	0.9144 m
Total Height of Truss	1.1218 m
Outer Diameter (all members)	0.01905 m
Inner Diameter (all members)	0.01600 m
Actuator Mass	3.0 kg.
Material	Aluminum
Density	2711.5 N/m^2
Motor Armature Inertia	$6.76\times10^{-6} Kg - m^2$
Motor Resistance	2 ohms
Motor Torque Constant	0.0398 $\frac{N-m}{A}$
Motor Back-emf Constant	0.0398 $\frac{V}{rad-s}$
Lead Screw Gain	$2.527\times10^{-6} \frac{m}{rad}$
Coefficient of Friction	0.1
Gearhead Gain	5.2

The simulation calculates a large number of variables. There are 117 nodal and reaction forces as well as 5 internal forces for each member. While the calculation of the nodal forces treats the active members as single links, the forces acting on the leadscrew tube cantilever connection are also found during the simulation. To reduce the number of variables presented in this section, we will only discuss the results for three representative members, located at different locations on the truss. The three members we will look at are: M6 - a fixed length member in the base cell,

M8 - a leadscrew tube in the active plane, and M19 - a fixed length member in the top plane.

5.1 Static Analysis

To perform the static analysis on the spatial truss, the MATLAB simulation was run with the active member lengths fixed. No gravity was applied to the truss, however, external loads in the vertical direction were applied to the top plane nodes (nodes N7, N8, and N9). The resulting reaction forces, nodal forces, and member stresses were compared to the results from the finite-element analysis. The prototype truss was also subjected to static loading and the strain in member M6 compared to the simulated value.

The static analysis was performed for three different configurations of the truss; the equilibrium position and two different slewed positions. The equilibrium position is the point where all three active member have the same length (1.1684 m). The first slewed position is the orientation where member M7/8 is a maximum (1.2289 m) and members M9/10 and M11/12 are a minimum (1.1077 m), and corresponds to the starting point for the dynamic simulation. The second slewed position is the point where member M7/8 is a minimum (1.1077 m) and members M9/10 and M11/12 are a maximum (1.2289 m). This position corresponds to the ending point for the dynamic simulation. These slewed positions were chosen because they represent the maximum slew that the prototype truss can achieve.

5.2 Finite Element Model

The finite-element analysis of the spatial truss was performed using MSC/PAL. The node locations, member dimensions, and material properties were taken directly from the MATLAB model. The members connecting each node were modelled as truss members allowing only axial loads to be applied to the members. No gravitational forces were applied to the truss model; external forces of 46.0 N were applied to each of the top plane nodes to provide the truss loading. The base plane nodes (N1, N2, and N3) were fixed so that no translation was allowed in any direction. The finite-element model file and the loading file are provided in Appendix D. The member and node identifiers for both the MATLAB and MSC/PAL models are identical and are as shown in Fig. 4.1.

5.3 Experimental Setup

The VPI&SU prototype truss was used to compare the simulation results to those found from a physical model. The VPI&SU truss is comprised of two double- octahedral bays and is attached to the ceiling hanging downward. In the simulation we are only dealing with a single double-octahedral bay, so one bay of the prototype truss must be static. For our experiment, the bay attached to the ceiling is fixed, and the second bay is actuated.

Several members of the first unit cell of the activated bay had strain gages that were attached during a previous experiment (not involved with this work). The existing gages were used to measure the strain in member M6. There were three

gages attached to the member; each was set parallel to the longitudinal axis of the member and was attached at the midpoint of the member. Each strain gage had a gage factor of 2.1. The gages were offset 120 degrees from each other about the diameter of the member. Three one-quarter-bridge circuits were used to measure the change in voltage of the gages. The voltage from the bridge circuits was amplified and then digitized using an analog-to-digital (A/D) conversion board resident in a PC. The digitized voltage from each gage was then multiplied by a conversion factor to convert it to microstrain and the value stored in a file. The three microstrain values were then averaged to find the tensile strain in the member. Calibration of each strain gage/bridge circuit was done by placing a known resistor (representing a known strain value) across the active leg of the bridge and recording the indicated strain.

5.4 Static Analysis Results

To check the results of the MATLAB simulation and the finite-element analysis for the static case, the reaction forces at the base nodes, the loads in the members, and the maximum von Mises stresses in the members were compared. The two models provided identical results for static loading. These results are presented in Table 5.2, Table 5.3, and Table 5.4. The tables provides only a small subset of the results. All member's results were compared and provided the same accuracy as that shown in the table.

The good comparison between the MATLAB model and the finite-element model provides confidence that the MATLAB model is correct for the static case. All other

Table 5.2: Comparison of MATLAB and MSC/PAL Static Results-Equilibrium Position

VALUES	MATLAB	MSC/PAL
Force Applied to Nodes N7, N8, and N9	46.0 N	46.0 N
Base Reaction at Node N1 - $\hat{n}_1$- Dir	-25.1882 N	-25.19 N
Base Reaction at Node N1 - $\hat{n}_2$- Dir	-14.5430 N	-14.54 N
Base Reaction at Node N1 - $\hat{n}_3$- Dir	-44.4980 N	-44.5 N
Axial Load in Member M6	35.2648 N (Tens.)	35.27 N (Tens.)
Axial Load in Member M19	16.7923 N (Comp.)	16.79 N (Comp.)
von Mises Stress in Member M6	3.9077 $\times 10^5 \frac{N}{m^2}$	3.908 $\times 10^5 \frac{N}{m^2}$
von Mises Stress in Member M19	1.8608 $\times 10^5 \frac{N}{m^2}$	1.861 $\times 10^5 \frac{N}{m^2}$

Table 5.3: Comparison of MATLAB and MSC/PAL Static Results-First Slewed Position

VALUES	MATLAB	MSC/PAL
Force Applied to Nodes N7, N8, and N9	46.0 N	46.0 N
Base Reaction at Node N1 - $\hat{n}_1$- Dir	-31.5663 N	-31.57 N
Base Reaction at Node N1 - $\hat{n}_2$- Dir	-11.0051 N	-11.00 N
Base Reaction at Node N1 - $\hat{n}_3$- Dir	-50.1415 N	-50.15 N
Axial Load in Member M6	41.0997 N (Tens.)	41.10 N (Tens.)
Axial Load in Member M19	24.6389 N (Comp.)	24.64 N (Comp.)
von Mises Stress in Member M6	4.5543 $\times 10^5 \frac{N}{m^2}$	4.554 $\times 10^5 \frac{N}{m^2}$
von Mises Stress in Member M19	2.7302 $\times 10^5 \frac{N}{m^2}$	2.731 $\times 10^5 \frac{N}{m^2}$

Table 5.4: Comparison of MATLAB and MSC/PAL Static Results-Second Slewed Position

VALUES	MATLAB	MSC/PAL
Force Applied to Nodes N7, N8, and N9	46.0 N	46.0 N
Base Reaction at Node N1 - $\hat{n}_1$- Dir	-19.6748 N	-19.68 N
Base Reaction at Node N1 - $\hat{n}_2$- Dir	-18.0142 N	-18.01 N
Base Reaction at Node N1 - $\hat{n}_3$- Dir	-38.9220 N	-38.92 N
Axial Load in Member M6	31.3596 N (Tens.)	31.36 N (Tens.)
Axial Load in Member M19	9.4169 N (Comp.)	9.419 N (Comp.)
von Mises Stress in Member M6	3.4750 $\times 10^5 \frac{N}{m^2}$	3.475 $\times 10^5 \frac{N}{m^2}$
von Mises Stress in Member M19	1.0435 $\times 10^5 \frac{N}{m^2}$	1.044 $\times 10^5 \frac{N}{m^2}$

member forces and von Mises stresses compared with the same precision as those listed in the tables.

The MATLAB simulation of the static case was also checked by comparing the calculated tensile strain in member M6 with the actual strain under a given load. The comparison with the prototype truss was performed for the same three configurations listed above. At each position, the strain gages on member M6 were set to zero with no external load applied. A known 46.0 N load was then applied to nodes N7, N8, and N9 and the resulting change in strain measured. Due to a large noise level and the small changes in strain, the procedure was repeated three times at each point and the resulting strains are the average of the measurements. The measured strain was then compared to the strain calculated by the MATLAB simulation. The results of this comparison is presented in Table 5.5.

The table shows good correlation between the simulation and the experimental strains. Possible sources of the error are discussed in Section 5.5.1.

Table 5.5: Comparison of MATLAB and Experimental Static Strain Results

VALUES	MATLAB	MEASURED	PERCENT ERROR
Applied Force	46.0 N	46.0 N	
Equilibrium Position			
Strain in Member M6	5.8631 $\mu\epsilon$ (Tens.)	5.9468 $\mu\epsilon$ (Tens.)	-1.41%
Slew Position 1			
Strain in Member M6	5.2094 $\mu\epsilon$ (Tens.)	5.1829 $\mu\epsilon$ (Tens.)	0.5%
Slew Position 2			
Strain in Member M6	6.8197 $\mu\epsilon$ (Tens.)	7.1210 $\mu\epsilon$ (Tens.)	-4.23%

After the static analysis was complete, the dynamic analysis was performed. In this, the results of the dynamic simulation were checked against the dynamic strains measured on the prototype truss.

5.5 Dynamic Analysis

Once the static simulation was performed and resulted in good comparisons, two dynamic simulations was performed. The objective of the dynamic simulations was to move the truss through some motion, measure the strain in a member, and compare that to the strain predicted by the simulation program. The active members of the prototype truss have a limited range of motion of +/- .0606 m from the equilibrium position. This limits the range of angles that we can rotate through to +/- 10 degrees. In addition, the gearhead and leadscrew gains are fixed, as are the motor parameters. These values do not allow for a fast motion which would generate large inertial loads.

Two different slewing experiments were performed and compared to the MATLAB simulation. The first experiment (referred to as Slewing Experiment 1) provided larger acceleration spikes and constant velocity during the maneuver. In the second experiment (referred to as Slewing Experiment 2), a sinusoidal voltage profile was input to the active member motors. The experiments and their results are presented below.

5.5.1 Slewing Experiment 1

The first dynamic analysis consisted of moving the truss from the slewed position of -10 degrees about the $\hat{n}_1$ axis to the slewed position of +10 degrees about the $\hat{n}_1$ axis. The was no rotation about the $\hat{n}_2$ axis. Constant external loads of 46.0 N, in the vertical direction, were applied to nodes N7, N8, and N9 during the motion. The active member lengths, linear velocities, and linear accelerations were saved in a file and used as inputs into the MATLAB simulation. The members were each subjected to virtually a constant velocity profile. There was a large acceleration spike at the beginning and the end of the motion with very little acceleration during the majority of the motion. This provided the maximum displacement of the active members and the maximum velocity. The motor amplifier was saturated during this maneuver. The measured position, velocity, and acceleration profile of member M7/8 are presented in Fig. 5.1, Fig. 5.2, and Fig. 5.3.

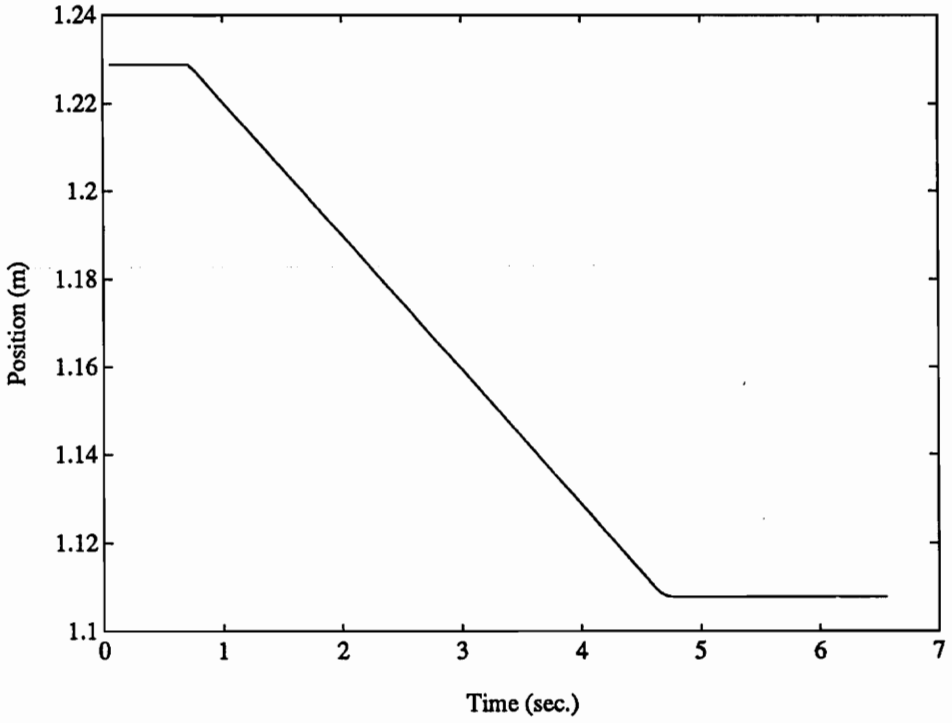

Figure 5.1: Displacement of Member M7/8 during Slewing Experiment 1

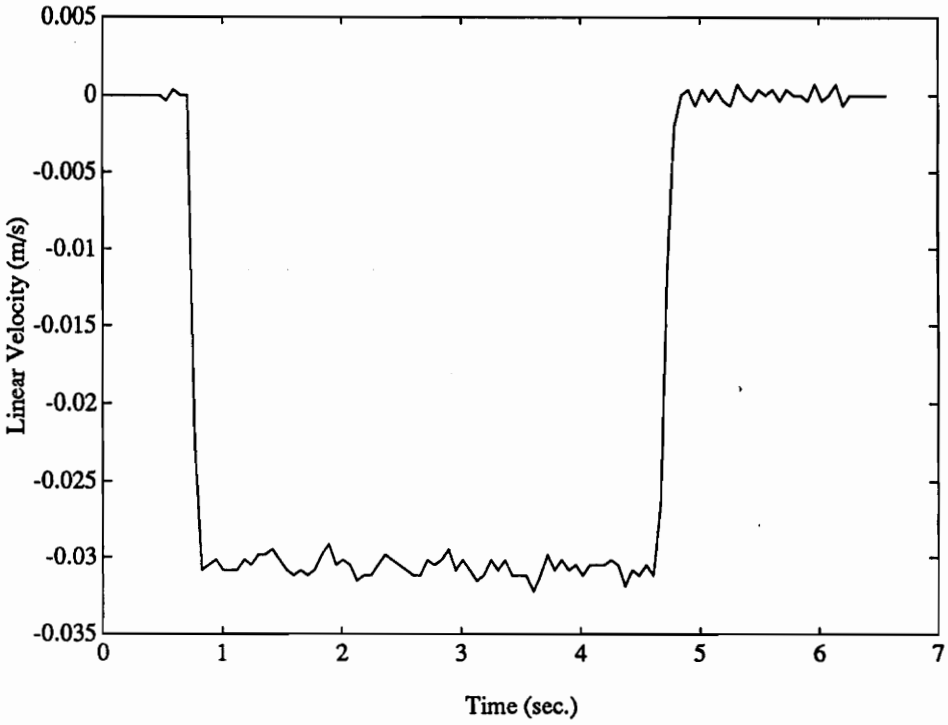

Figure 5.2: Velocity of Member M7/8 during Slewing Experiment 1

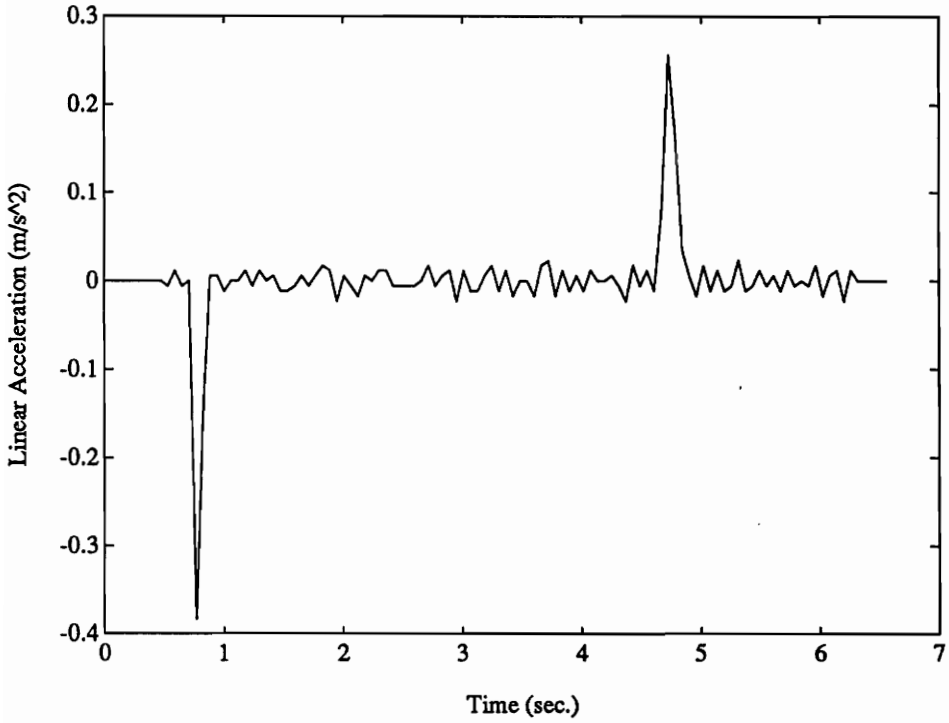

Figure 5.3: Acceleration of Member M7/8 during Slewing Experiment1

5.5.2 Slewing Experiment 2

The second slewing experiment involved moving the truss from an initial slewed position of -8 degrees about the $\hat{n}_1$ axis to the slewed position of +8 degrees about the $\hat{n}_1$ axis. In contrast to the first slewing experiment where the truss moved from one slewed position to another and then stopped, in the second slewing experiment the truss continuously moves from one position to the other until the motor voltage is cut. The voltage input to the motor was specified to be sinusoidal; this resulted in a cyclical velocity and acceleration profile of the active members. In addition, the input voltage level was selected so that the motor amplifier would not be saturated during the maneuver. As in the first slewing experiment, constant external vertical loads of 46.0 N were applied to nodes N7, N8, and N9, and the measured active member lengths, velocities, and accelerations were used as inputs to the MATLAB simulation. The measured position, velocity, and acceleration profile of member M7/8 are presented in Fig. 5.4, Fig. 5.5, and Fig. 5.6.

5.6 Comparison of Experimental and Simulated Results

For both experiments, the strain in member M6 was recorded throughout the motion to be compared to the strain calculated in the MATLAB simulation. Since the attached strain gages can only measure a change in strain, it was necessary to define a zero strain position. The zero strain position was chosen to be the equilibrium position with no external loads applied. The truss was first put at this position and the strain gage balance set to zero. The external forces of 46.0 N were then applied to nodes N7, N8, and N9 and the truss moved to the maneuver starting position (a

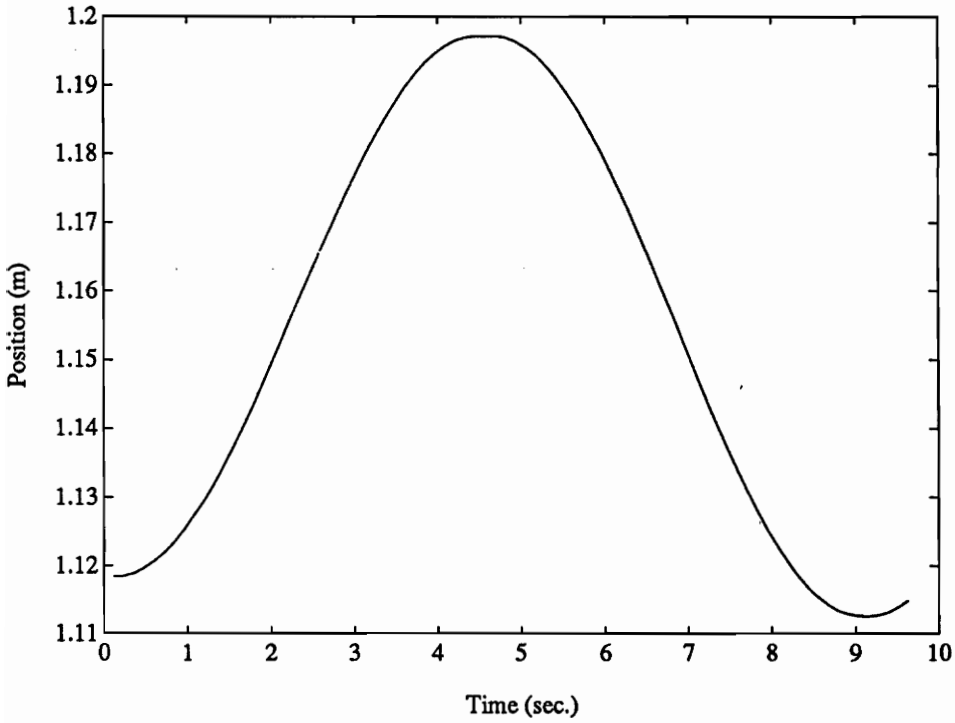

Figure 5.4: Displacement of Member M7/8 during Slewing Experiment 2

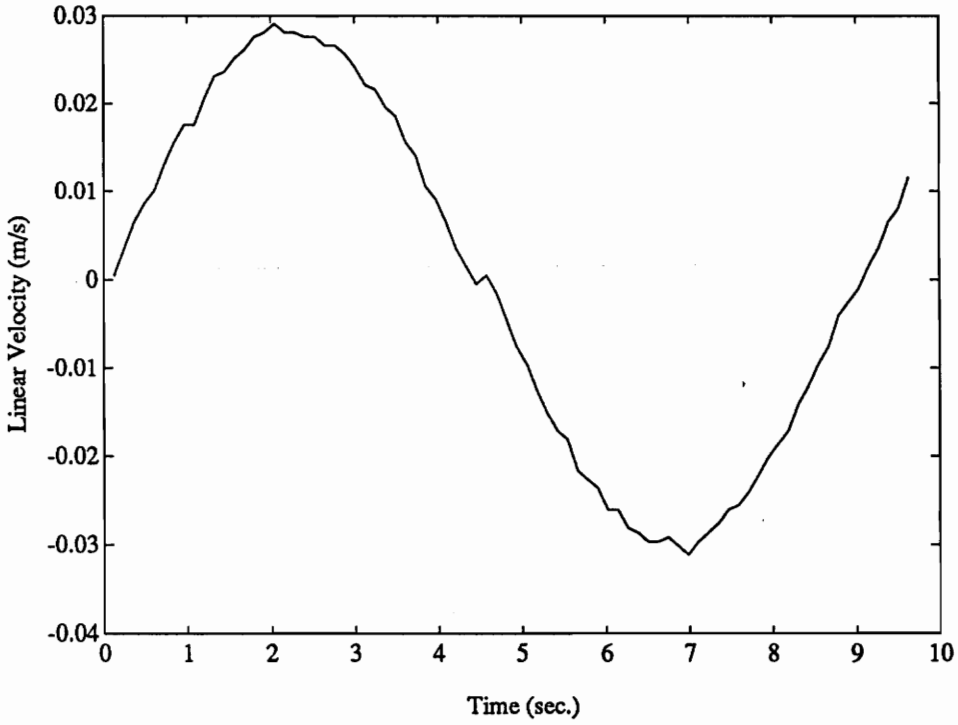

Figure 5.5: Velocity of Member M7/8 during Slewing Experiment 2

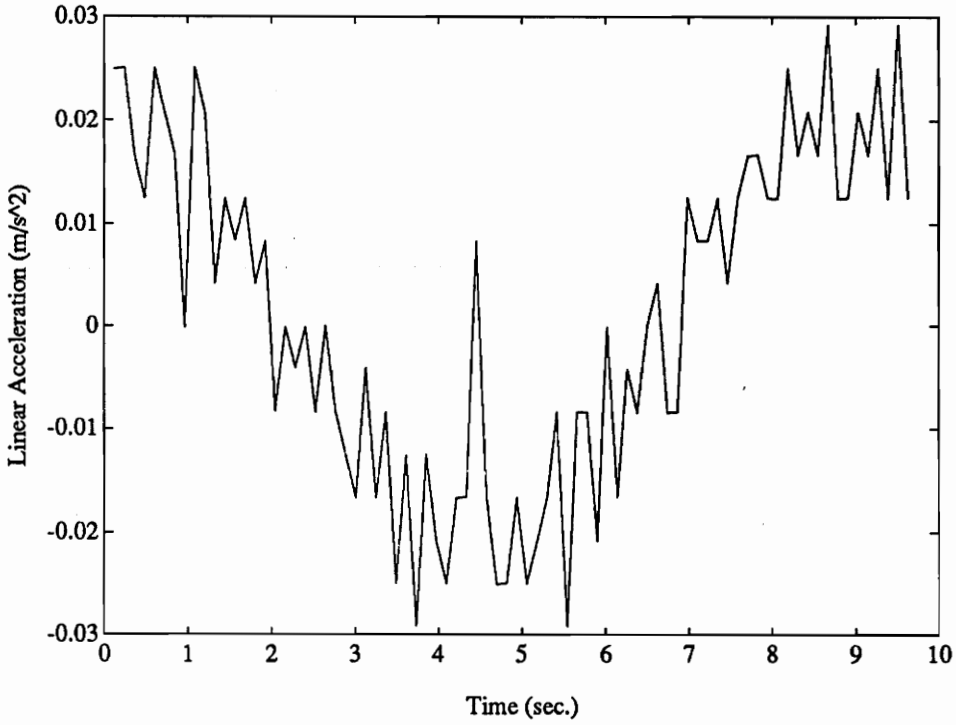

Figure 5.6: Acceleration of Member M7/8 during Slewing Experiment 2

-10 degree slew). At this point the strain recorder was turned on and the motion of the truss commenced. The first slewing experiment required approximately 5.5 seconds to complete; the second slewing experiment required approximately 10 seconds. For the first slewing experiment, the strains were recorded for a short period of time both before and after the motion to get a static baseline. The second slewing experiment consisted of a continuous motion of the truss – only one cycle of the motion is presented below. In both experiments, the recorded strain includes the effects of the applied loads and gravitational forces caused by motion away from the zero position.

After each physical maneuver was complete, the measured link histories were used as inputs to the MATLAB simulations. The simulations calculated the total tensile strain that the gages attached to member M6 should measure, including both gravitational and applied load effects. Therefore, it was necessary to modify the simulation results to have the same zero strain position as the experimental data. In each case, the simulation was first performed with both gravity and the external loads applied to the truss and the strain history for member M6 calculated. Next, the same simulation was performed neglecting the effects of gravity. The change in strain between the first simulation (with gravity) and the second simulation (without gravity) was then calculated. This was the strain due to gravity on the structure throughout the motion. Since the strain gages were set to zero at the equilibrium position under gravitational load, the equilibrium gravity strain was subtracted from the total gravitational strain history. The resulting values were the change in the gravitational strain of the member caused by the motion away from the equilibrium position. This strain curve was then added to the strain calculated by the simula-

tion in which gravity was not included. The final result is the change in strain of the member due to both the gravitational effects and the applied loads. This strain value was then compared with that found during the experiments.

5.6.1 Slewing Experiment 1

The results of the experimental and simulated strain for slewing experiment 1 are presented in Fig. 5.7. The strain gages used in the analysis were very noisy and tended to drift because the gage setup did not include temperature compensation. To compensate for the noise and the drift, the maneuver was performed three times. The strain from the three maneuvers was averaged to get the strain curve shown in the Fig. 5.7. During the maneuver, 110 data points were taken. The time step between each data point was $\Delta t = 0.0590$ sec. During the 0.059 sec. time step, 100 strain values were read from each gage and averaged to get the strain reading for the time step. The three strain readings for the time step were then averaged together to find the tensile strain in the member at that time step.

The simulated strain is higher than the experimental value for the entire curve. At the beginning of the motion, the error is approximately 13 percent. At the end of the motion, the error has decreased to within 5 percent. The two curves seem to be offset by some constant value. It is possible that the experimental strain is lower than it should be. We can check the strain in both the simulation and the experiment at the equilibrium position. This is the point where the static gravity effect was zeroed out of the experimental strain. When we performed the static strain experiment at the equilibrium position, we found the static strain in member

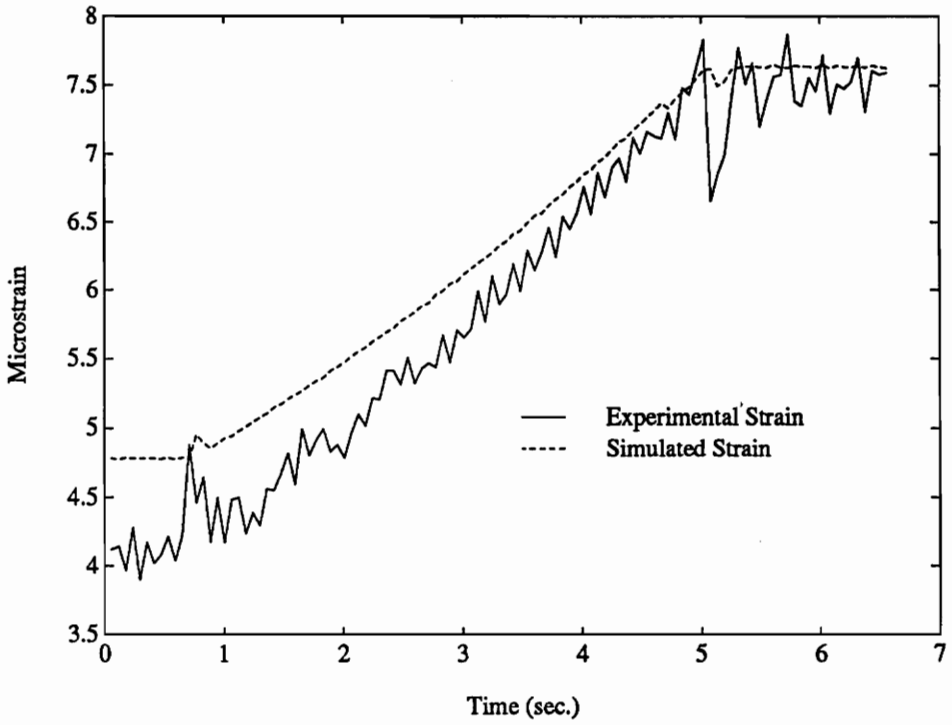

Figure 5.7: Comparison of Experimental and Simulated Strain during Slewing Experiment 1

M6 to be 5.9468 $\mu\epsilon$. Since the acceleration at this point is almost zero, the dynamic experiment should produce approximately the same value of strain when the truss moves through the equilibrium position (approximately 2.8 seconds). At this point, however, the measured strain is approximately 5.5 $\mu\epsilon$. The simulated dynamic strain is within 1 percent of the simulated static strain at the equilibrium position.

5.6.2 Slewing Experiment 2

The results of the experimental and simulated strain for slewing experiment 2 are presented in Fig. 5.8. As with the first experiment, the experiment was performed three times and the measured strain from each experiment was averaged to get the curve shown in the figure. The figure shows one cycle of the maneuver; during the cycle 80 data points were taken with a time step of $\Delta t = 0.121$ sec. During each 0.121 sec. time step, 100 strain readings were taken and averaged to get the strain value for the time step.

The simulated strain compares well with the measured strain except for a constant offset of approximately 0.5 $\mu\epsilon$. This offset results in an error of 9 percent. Except for the offset, the curves of the measured and simulated strain overlay each other. The good agreement provides us with confidence that the dynamic model of the truss is accurate. A discussion of the error between the measured and simulated strains is presented below.

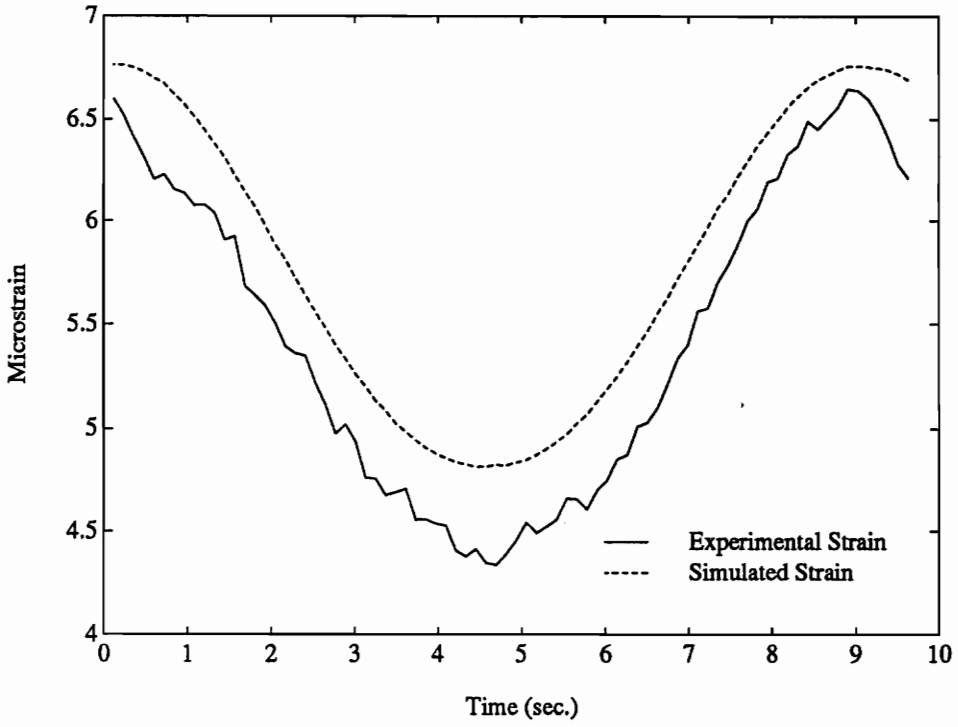

Figure 5.8: Comparison of Experimental and Simulated Strain during Slewing Experiment 2

5.6.3 Error Analysis

There are several ways in which errors can be introduced into the analysis. A major assumption made in the development of the force model was that the truss was an ideal one. The actual truss used for the experiment, however, is not ideal. The members are not connected with ideal pinned connections; moments are transmitted through the joints to the members. This will cause unmodelled forces to be applied to the member. Another source of error is in the development of the physical model's dimensions. All physical parameters of the prototype truss (member lengths, outer diameters, applied loads) that could be measured were measured. However, several important parameters could not be found directly and were estimated. The estimated values include the inner diameter of the members, density of the material, and the mass of the actuator (including motor, gearhead, leadscrew, support housing, etc.). The strain simulation is very sensitive to changes in member properties and dimensions. In addition, the kinematics are very sensitive to variations in the lengths of the members. The kinematics requires the positions of the ideal member connecting points, however, since the prototype truss uses offset joints, it was necessary to estimate the ideal connection point, and thus the lengths of the ideal members. As a final topic on the error between the experiment and the simulation, the strain gages were very noisy and tended to drift throughout the experiment. One of the reasons the strain gages were so noisy is that the levels of strain being measured were very low (maximum $7.9\mu\epsilon$). Every effort was made to zero the gages before running a maneuver and to average out the drifting and noise, however, there could be some error in the measured values.

Although there is a constant offset between the simulated and the experimental

strain curves, they do provide good correlation. To perform a more accurate comparison, it would be necessary to know the exact values of the truss properties and to use a better strain gage setup.

5.7 Simulation Results

All simulation results presented below were developed using the MATLAB simulation with both gravitational forces and external forces applied to the model. The values are absolute, i.e., they represent the total forces and stresses caused by gravity, the applied external loads, and the inertial loads. The link displacements, velocities, and accelerations were taken from the measured data for slewing experiment 1.

The maximum von Mises stresses occurring throughout the motion in members M6, M8, and M19 are presented in Fig. 5.9, Fig. 5.10, and Fig. 5.11. The maximum stress in members M6 and M19 occur at the midpoint of the member, while the maximum stress in member M8 occurs at the cantilever connection with the leadscrew. The maximum stress in member M6 is 1.294 $\times 10^6 \frac{N}{m^2}$. For member M8, the maximum stress is 3.57 $\times 10^7 \frac{N}{m^2}$. The maximum stress in M19 is 1.66 $\times 10^6 \frac{N}{m^2}$. For members M6 and M19, the maximum stress occurs at the end of the motion. The maximum stress in the leadscrew tube occurs at the beginning of the motion. The static stress is also shown for each of the members. As noted above, the motion is so slow that the stress occurring in the members during the motion is almost the same as the static stress. The maximum stresses for each member are due to the truss orientation rather than its motion.

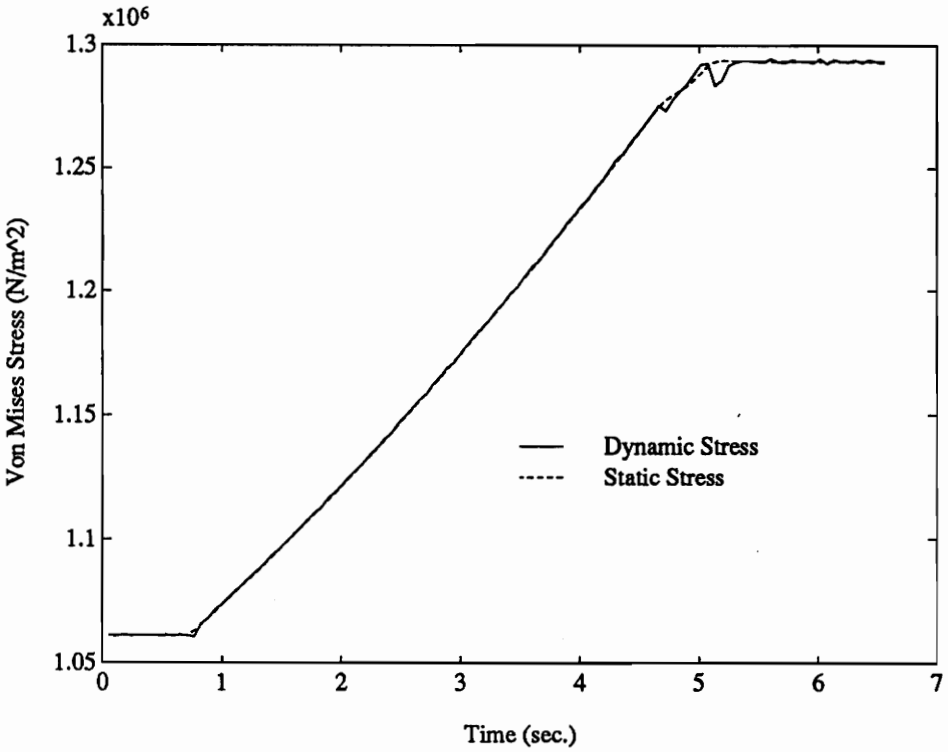

Figure 5.9: Maximum von Mises Stress in Member M6

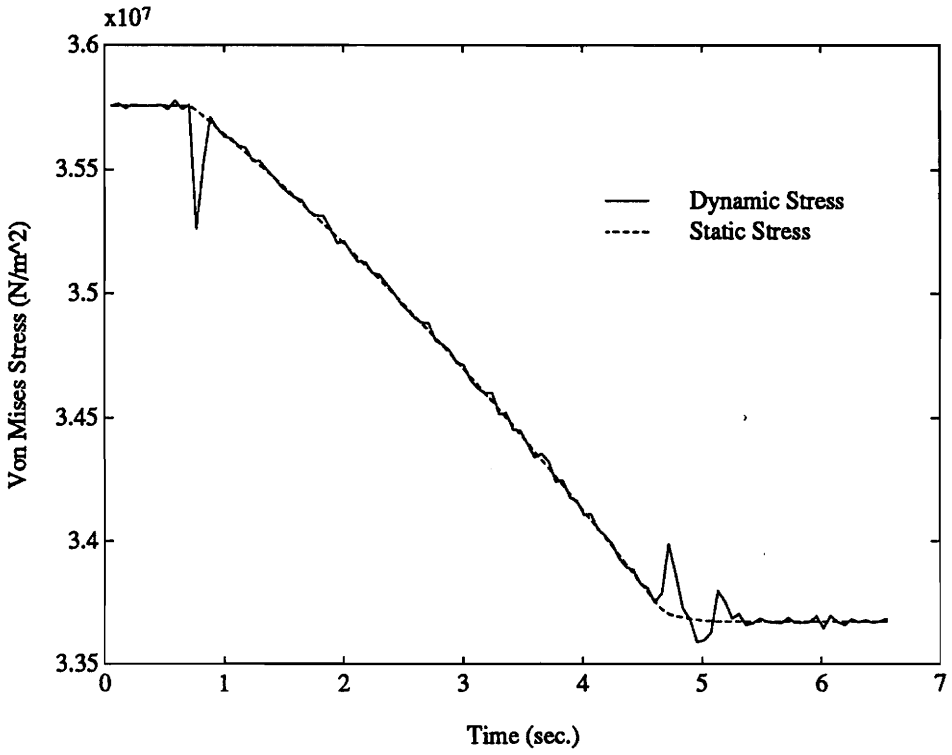

Figure 5.10: Maximum von Mises Stress in Member M8

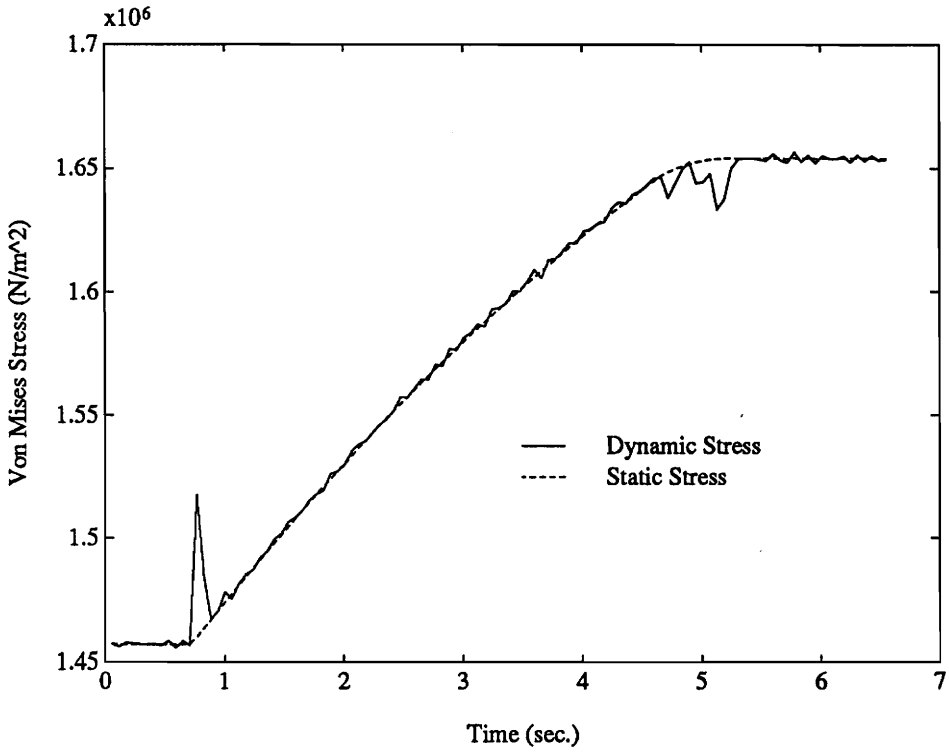

Figure 5.11: Maximum von Mises Stress in Member M19

The von Mises stress in each member is computed from the tensile stress, shear stresses, and bending moment stresses. Figure 5.12 shows the stresses in member M6 caused by the tensile load and the bending moments. The shear forces are negligible and are not shown in the figure.

In an ideal truss, with no member weight effects, the static truss member carries only axial loads and so any shear stresses or bending moment stresses will be caused only by the motion. However, when gravitational effects are included, the member has internal shear and bending moments that resist the weight of the member. This is in addition to the stresses caused by the motion. For the simulation being considered here, both weight effects and motion effects cause internal loads in the members. Since the motion is quite slow, the main cause of these internal loads is the weight of the member and the applied loads. As seen in Fig. 5.12, the largest portion of the stress in member M6 is caused by the tensile force. The maximum tensile stress is 8.4 $\times 10^5 \frac{N}{m^2}$. The bending moment stress in the $\hat{b}_2$ direction is the next largest contributor, with a maximum value of 4.5 $\times 10^5 \frac{N}{m^2}$. The bending moment stress in the $\hat{b}_1$ direction has a maximum value of 3.95 $\times 10^5 \frac{N}{m^2}$. The maximum shear stress is in the $\hat{b}_2$ direction and has a value of 230.8 $\frac{N}{m^2}$. In this simulation, the shear stresses are insignificant compared to the tensile and bending moment stresses. The shear stresses in the physical truss could be higher due to the non-ideal connections.

The force applied by the actuator onto the leadscrew tube depends on both the acceleration of the leadscrew tube and the orientation of the truss. The actuator force is required to provide the motion to the truss and is also required to provide the support force to keep the truss from collapsing inward. The force exerted by the

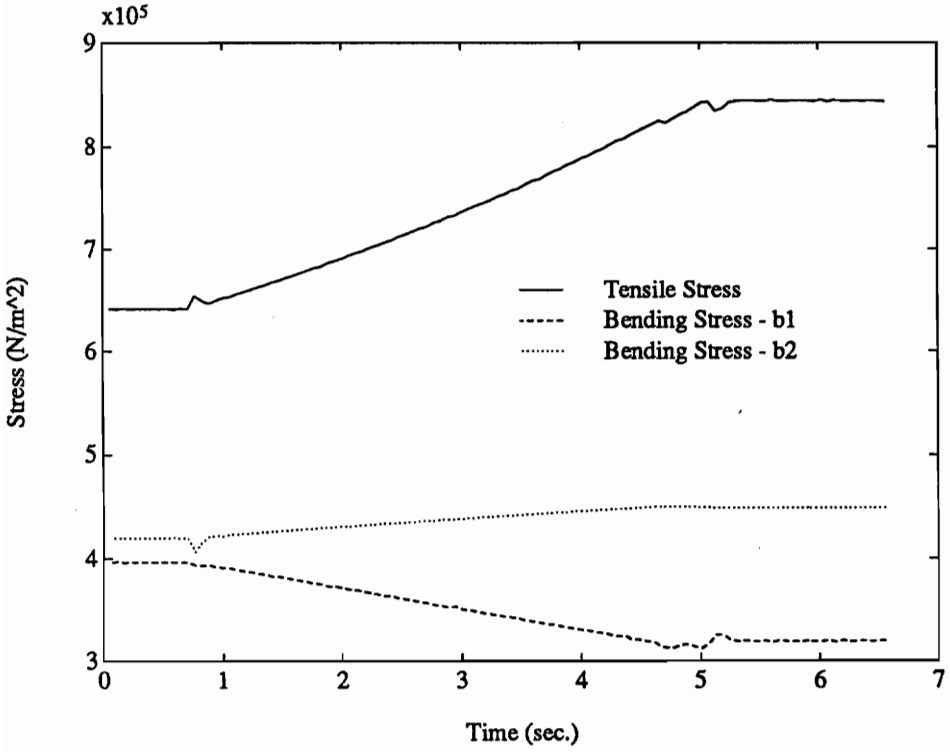

Figure 5.12: Components of Stress in Member M6

actuator on member M8 throughout the motion is presented in Fig. 5.13. The negative sign of the actuator force indicates that the leadscrew tube is in compression during the entire maneuver. The maximum compressive load is 74.5 N occurring at the beginning of the motion. The minimum compressive load is 29.5 N and occurs at the end of the motion. The compressive load is decreasing throughout the motion. In the first slewed position, member M8 is acting to keep nodes N4 and N5 apart since the static load is trying to drive them together. As the truss moves to the second slewed position, its orientation results in less force trying to drive nodes N7 and N8 together. Therefore, the compressive force decreases. If the truss could move farther along the same trajectory past the second slewed position, it would reach a point where member M8 changes over to tensile loading. At this point, the actuator force would become positive.

Once the force exerted on the leadscrew tube and the linear velocity of the actuator are known, we can use these values to back out the motor torque, voltage, and current histories required throughout the motion. We will present these values including leadscrew friction and neglecting it. The motor torque for member M7 is presented in Fig. 5.14. Since the actuator is providing a compressive force to prevent motion, as the actuator shortens, it is working with the load. At the end of the motion, and during the the static portion of the maneuver, the actuator works against the load. Since the force required by the actuator decreases during the motion, the torque also decreases. The torque required by the motor is much higher when the friction effects are included.

The current in the motor is directly related to the torque by the torque constant.

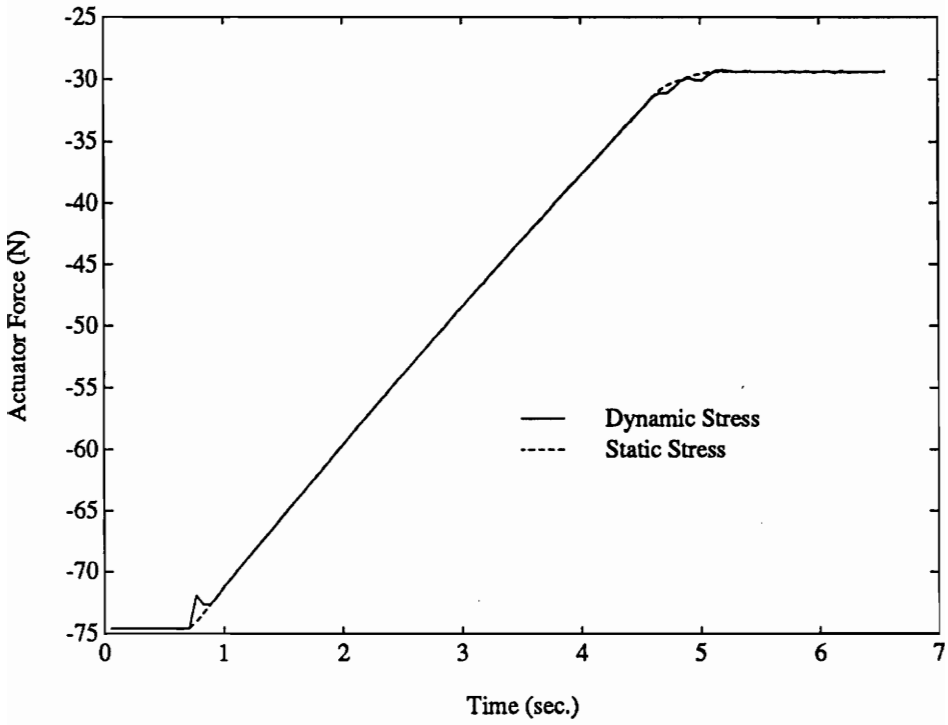

Figure 5.13: Actuator Force on Member M8

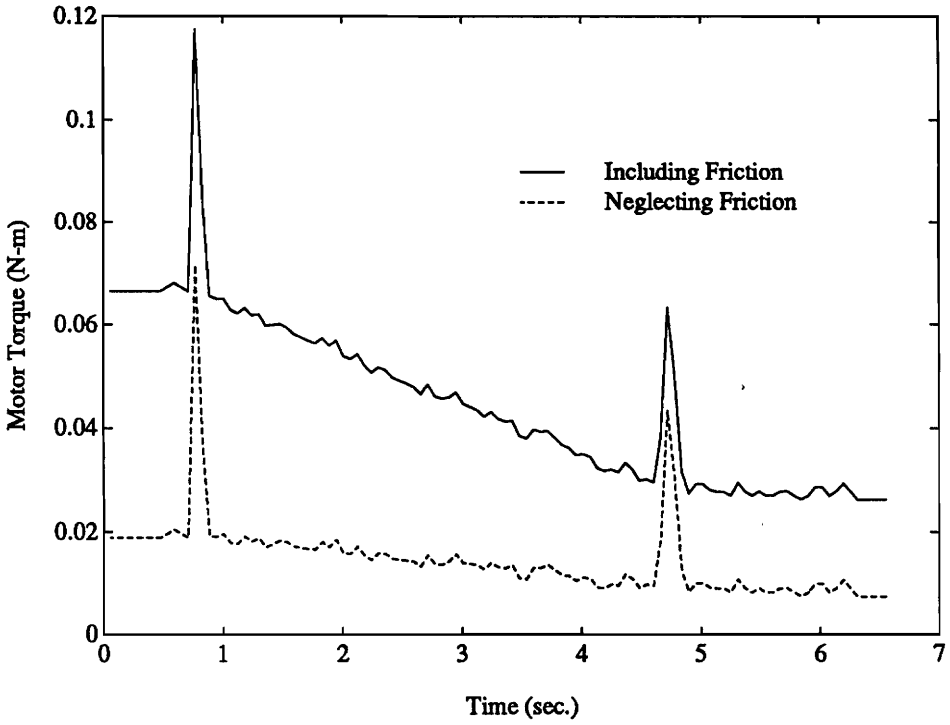

Figure 5.14: Motor Torque for Member M7

Once the torque produced by the motor is known, it is simple to calculate the current in the motor. The simulated current of the motor in member M7 is shown in Fig. 5.15.

The voltage for the motor is presented in Fig. 5.16. Once the motion begins, the voltage reaches a maximum value of 29 volts. The voltage required for the case of neglecting friction is a maximum of approximately 27 volts. The voltage model which includes friction is suspect since the maximum input voltage to the motor is 25 volts. One possible reason for the curve which includes friction being so large is that the coefficient of friction (0.1) is incorrect. In Section 2.8, we discussed the difficulty of finding a "good" value for friction coefficient. The voltage curve which includes friction has an offset before and after the motion occurs. This offset is approximately 3.8 V before the motion starts and 2.4 V after the motion ends. This voltage is due to the force on the leadscrew tube required to keep the active member constant length under a static structural load. In the actual truss, the self-locking characteristic of the leadscrew would provide the static force and the voltage requirement would be zero. Before motion of the truss starts, however, this static load must be overcome by the motor torque.

5.8 Buckling Analysis

The criteria for failure from buckling is that the applied compressive load be more than the allowable load. Since the prototype truss is hanging from the ceiling and the applied forces are acting downward, the only members that have a compressive load are member M8 and members M19, M20, and M21. The maximum compressive

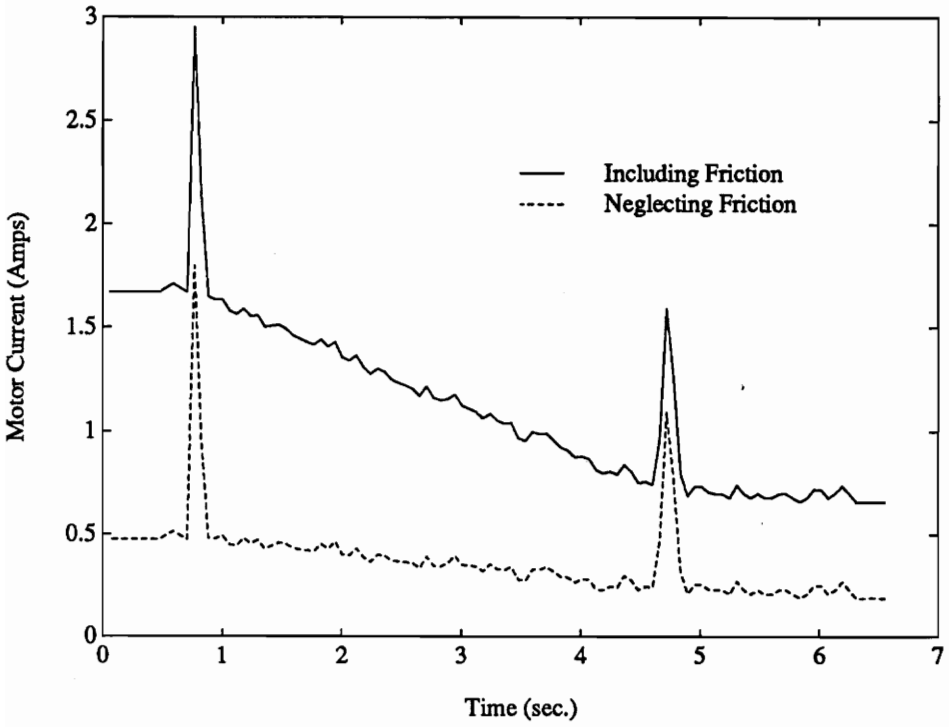

Figure 5.15: Motor Current for Member M7

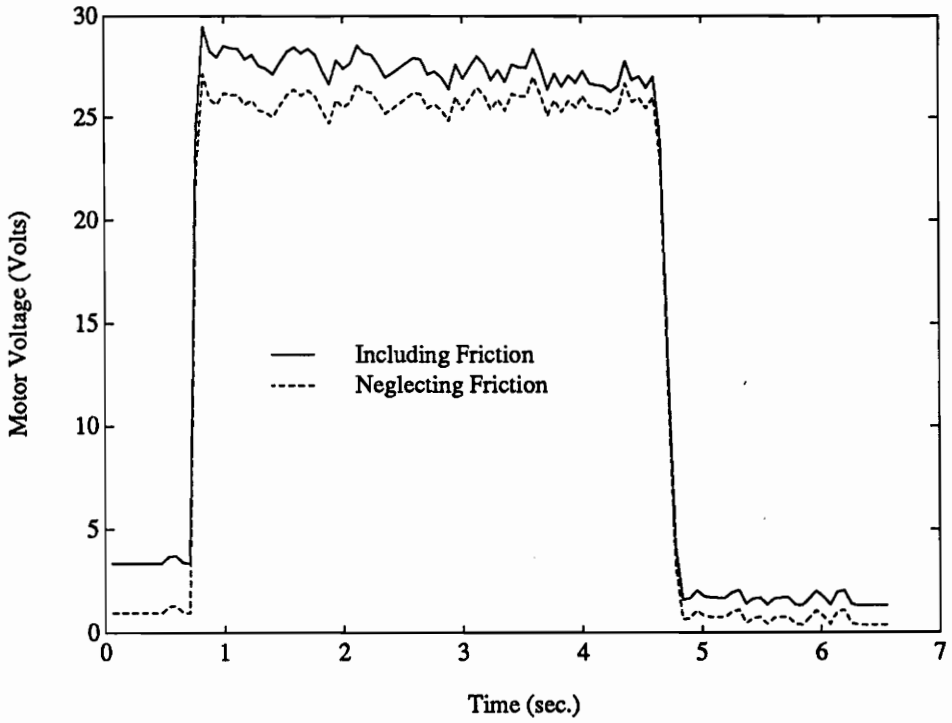

Figure 5.16: Motor Voltage for Member M7

load in M8 is 74.63 N. The maximum allowable load is 35258.0 N. Therefore, for the experimental maneuver, the member is not in danger of failure from buckling. The maximum allowable compressive load for the members of the top plane (M19, M20, and M21) is 3055.2 N. The maximum compressive loads are: M19 - 27.2 N, M20 - 22.6 N, and M21 - 22.7 N. The allowable load is so much higher for the leadscrew tube (M8) because it is much shorter (.254 m) than the fixed length members (.9615 m).

5.9 Summary

In this chapter, we have presented the simulation of the double-octahedral truss. The static forces calculated by the simulation compared with high accuracy (maximum error -4.23%) to those predicted by the finite-element model. The simulated forces were also compared to actual forces in a physical truss by comparing static and dynamic strains in member M6 during two different slewing experiments. The simulated strains had a maximum error of 15 percent from the actual dynamic strains in the first slewing experiment and a maximum error of 9 percent during the second slewing experiment. We discussed the possible sources of error in the model that could account for this difference. A small offset in the experimental values caused by strain gage drifting is a feasible explanation for the difference. We concluded by presenting results of the simulation program for different members of the truss and for one of the actuator motors.

The simulation presented in this chapter looked at only a single bay, double-octahedral truss. The analysis can be extended with minimal modification to take care of a chain of several double-octahedral bays. A discussion of the extension is

presented in Chapter 6.

Chapter 6

Analysis of Multiple Bay Trusses

The analysis developed in Chapter 4 for the spatial truss considered only a single bay. In some instances, however, a spatial truss may be composed of many double-octahedral bays connected together to form a chain. It is desirable to extend the kinematic and force analysis to deal with a multiple bay truss. Each bay can be analyzed separately using the model developed in Chapter 4. The kinematic analysis, however, must be modified to account for the multiple bays. In addition, the reaction forces at the base plane of a bay must be passed on to the next bay in the chain. The kinematic analysis will be developed first and then the application of the force analysis will discussed.

6.1 Kinematic Analysis of Multiple Bay Trusses

The kinematic analysis developed by Warrington is valid only for a single bay truss. All nodal positions, velocities, and accelerations are calculated relative to the base plane, which is fixed in space. This is necessary since we must have absolute, Newtonian values to develop the force relationships. However, for multiple bay trusses, only the base plane of the first bay is fixed. The base plane for all subsequent bays is moving in Newtonian space. Therefore, we must develop a method for finding

the Newtonian position, velocity, and acceleration of the nodes of subsequent bays. This method is outlined below.

To perform the kinematic analysis for the multiple bay truss, we use Warrington's kinematic analysis for a single bay. The motion that we are specifying is the motion of the top plane in Newtonian coordinates. For an n-bay truss, we break up the motion into n parts. For example, if we want a rotation of the top plane of a 3 bay truss to be 30 degrees about the $\hat{n}_1$ axis and 30 degrees about the $\hat{n}_2$ axis, we will perform Warrington's kinematic analysis for a 10 degree, 10 degree rotation on a single bay. The results of the kinematic analysis will provide the position, velocity, and acceleration of the nodes of each bay relative to the corresponding base plane. For the first bay, which is connected to ground, the values will be the Newtonian values. The kinematic analysis also provides the location, orientation, velocity, and acceleration of the top plane. The top plane of the first bay becomes the base plane of the second bay; the top plane of the second bay the base plane of the third bay, and so on up to the n^{th} bay. We will find the position, velocity, and acceleration of the multiple bays by working from the first bay sequentially up to the final bay. For each bay, we will have a bay-fixed coordinate system, identified as bi where the i represents the number of the bay away from the first bay. The first bay is fixed to ground; therefore $b1$ is the Newtonian coordinate system. The n^{th} bay will have the bn coordinate system. We will first show how to find the location of all of the nodes in each bay.

6.1.1 Position Analysis

The transformations for the nodes of the base plane of the second bay are simple, since those nodes are at the same location in space as the nodes of the top plane of the first bay. To find the location of the other 6 nodes, we describe a vector from the origin of the fixed base plane of the first bay to each node of the second bay. The Newtonian coordinate system is fixed at the centroid of the base of the first bay. A second coordinate system, $b2$, is attached to the centroid of the base plane of the second bay. The vector for the i^{th} node of the second bay is defined as

$$P_{i_2} = P_{base_2} + [\mathrm{C}^{b2/N}]P_{i/base} \tag{6.1}$$

where:

P_{i_2}=	the location of the i^{th} node of the second bay in Newtonian coordinates
P_{base_2}=	the vector from the origin of the Newtonian axes to the origin of the $b2$ coordinate system
$[\mathrm{C}^{b2/N}]$=	the rotation matrix relating the $b2$ coordinate system to the Newtonian axes
$P_{i/base}$=	the vector locating the i^{th} node in the $b2$ coordinate system

The vector $P_{i/base}$ is the x,y,z position of the node that we get from Warrington's analysis. The vector P_{base} is found from the kinematic analysis for the first bay, and is defined as

$$P_{base} = \frac{N7_1 + N8_1 + N9_1}{3} \tag{6.2}$$

where:

$N7_1, N8_1, N9_1$=	vectors containing the locations of the top plane nodes of the first bay

The rotation matrix, $[C^{b2/N}]$, is also provided by Warrington's kinematic analysis. The values provided by Warrington's kinematic analysis are all given with respect to the base plane. Therefore, they will be the same for each bay in the chain.

To find the location of the nodes in the n^{th} bay, we use a modified version of equation 6.1 above. This equation is

$$P_{i_n} = P_{base_n} + [C^{bn/N}]P_{i/base} \tag{6.3}$$

The $P_{i/base}$ value will be the same for the same node of each bay, however, the rotation matrix and the location of the P_{base_n} will be different. The rotation matrix, $[C^{bn/N}]$, is the product of the rotation matrices of all of the previous bays. It is calculated as

$$[C^{bn/N}] = [C^{b2/N}][C^{b3/b2}]...[C^{bn-1/bn-2}][C^{bn/bn-1}] \tag{6.4}$$

The rotation matrices are identical for each bay since the motion of each bay is identical with respect to its base plane. Therefore, the rotation matrix can also be written as

$$[C^{bn/N}] = [C^{b2/N}]^{n-1} \tag{6.5}$$

The vector position of P_{base} is the origin of the n^{th} coordinate system, bn, and is defined as

$$P_{base_n} = \frac{N7_{n-1} + N8_{n-1} + N9_{n-1}}{3} \tag{6.6}$$

where:

$N7_{n-1}, N8_{n-1}, N9_{n-1}$= vectors containing the Newtonian locations of the top plane nodes of the n-1 bay

Using these values, we can calculate all of the nodal positions for each of the bays. We must start at the first bay and work up to the last bay. Once we have found the position of the nodes of all of the bays in the truss, we next have to calculate their velocities.

6.1.2 Velocity Analysis

The nodal velocities that are calculated by Warrington's analysis are based on the base plane being fixed. For the first bay in the chain, this is true. However, for the rest of the bays, this is not true. Therefore, we must find a means of transforming the relative velocities provided by Warrington's analysis into Newtonian velocities. We use the 3-part velocity formula for finding the velocity of a point in a moving coordinate system. This is done in a manner similar to the position analysis. The velocity of the i^{th} node of the n^{th} bay is calculated as

$$V_{i_n} = V_{base_n} + V_{i/base} + \left\{\omega_{bn/N}\right\} \times P_{i/base}[11] \tag{6.7}$$

where:

$V_{i_n}=$	the velocity of node i of the n^{th} bay in Newtonian coordinates
$V_{base_n}=$	the velocity of the origin of the n^{th} coordinate system in Newtonian coordinates
$V_{i/base}=$	the velocity of the node with respect to the bn coordinate system(in bay-fixed coordinates)
$\{\omega_{bn/N}\}=$	the angular velocity vector of the bn coordinate system with respect to the Newtonian axes
$P_{i/base}=$	the position vector locating the node in bay-fixed coordinates

It is necessary that all of the components of the velocity equation be described in the same coordinate system. It is convenient to use the Newtonian coordinates system since we eventually will have to express the total velocity in Newtonian coordinates. To do this, we have to transform both $V_{i/base}$ and $P_{i/base}$ by premultiplying them by the rotation matrix, $[C^{bn/N}]$. The transformation matrix for each bay has been calculated during the position analysis.

To find V_{base_n}, we must have already performed the velocity analysis for the $n-1$ bay. The results of the $n-1$ bay analysis are then used to find V_{base_n} as

$$V_{base_n} = \frac{V7_{n-1} + V8_{n-1} + V9_{n-1}}{3} \tag{6.8}$$

where the nodal velocities are expressed in Newtonian coordinates.

We next need to calculate the angular velocity, $\{\omega_{bn/N}\}$. The angular velocity of the top plane of a bay with respect to its base plane is provided by Warrington's kinematic analysis. To find the angular velocity of the n^{th} coordinate system with respect to the Newtonian system, we simply add the angular velocities for each bay,

i.e.,

$$\left\{\omega_{bn/N}\right\} = \left\{\omega_{b2/N}\right\} + \left\{\omega_{b3/b2}\right\} + ... + \left\{\omega_{bn-1/bn-2}\right\} + \left\{\omega_{bn/bn-1}\right\} \qquad (6.9)$$

where:

$\left\{\omega_{b2/N}\right\}=$ the angular velocity of the $b2$ coordinate system with respect to the Newtonian system

$\left\{\omega_{b3/b2}\right\}=$ the angular velocity of the $b3$ coordinate system with respect to the $b2$ coordinate system

$\left\{\omega_{bn-1/bn-2}\right\}=$ the angular velocity of the $bn-1$ coordinate system with respect to the $bn-2$ system

$\left\{\omega_{bn/bn-1}\right\}=$ the angular velocity of the bn coordinate system with respect to the $bn-1$ system

Once again, all of the angular velocities must be expressed in the same coordinate system; it is easiest to use the Newtonian system. Therefore, all of the angular velocity vectors must be premultiplied by a rotation matrix. The rotation matrix must be the one that transforms the coordinates of the particular angular velocity back to Newtonian coordinates. Thus to find the Newtonian description of $\left\{\omega_{bn/bn-1}\right\}$, we must premultiply by $[\mathrm{C}^{bn/N}]$.

As an example, consider finding the velocity of the 5^{th} node of the 2^{nd} bay. The velocity equation becomes

$$V_{5_2} = \frac{V7_1 + V8_1 + V9_1}{3} + [\mathrm{C}^{b2/N}]V_{5/base} + \left\{\omega_{b2/N}\right\} \times [\mathrm{C}^{b2/N}]P_{i/base} \qquad (6.10)$$

Once we have the velocity of each of the nodes in each bay, we next have to find their acceleration.

6.1.3 Acceleration Analysis

The acceleration analysis is similar to the velocity analysis except that we now have to use the 5-part acceleration formula [11] to find the acceleration of a point that is accelerating in an accelerating coordinate system. We still have to take care to use the same coordinate systems when adding components or taking cross products. The 5-part acceleration formula can be written for the i^{th} node in the n^{th} bay as

$$A_{i_n} = A_{base_n} + A_{i/base} + \left\{\alpha_{bn/N}\right\} \times P_{i/base} + \left\{\omega_{bn/N}\right\} \times (\left\{\omega_{bn/N}\right\} \times P_{i/base}) + 2\omega_{bn/N} \times V_{i/base} \tag{6.11}$$

where:

$A_{i_n}=$	the acceleration of the i^{th} node
$A_{base_n}=$	the acceleration of the origin of the coordinate system attached to the n^{th} bay
$A_{i/base}=$	the acceleration of the i^{th} node with respect to the bay fixed coordinate system
$\left\{\alpha_{bn/N}\right\}=$	the angular acceleration of the coordinate system attached to the n^{th} bay
$\left\{\omega_{bn/N}\right\}=$	the angular velocity of the coordinate system attached to the n^{th} bay
$P_{i/base}=$	the location of the i^{th} node with respect to the bay fixed coordinate system
$V_{i/base}=$	the velocity of the i^{th} node with respect to the bay fixed coordinate system

All of the values in the above equation must be expressed in Newtonian coordinates. The rotation matrix relating the bay fixed coordinate system to the Newtonian coordinate system is provided in equation 6.4 for n bays. The method for finding the Newtonian coordinates for $P_{i/base}$, $\left\{\omega_{bn/N}\right\}$, and $V_{i/base}$ were also presented in Section 6.1.2. The only new values we must calculate are the acceleration terms, $A_{i/base}$ and A_{base}, and the angular acceleration, $\left\{\alpha_{bn/N}\right\}$.

We will use the same approach for finding the accelerations as we did to find the velocities. The acceleration of the node with respect to the bay fixed coordinate system ($A_{i/base}$) is provided by Warrington's kinematic analysis. To use it in the above equation, we only have to transform the vector into the Newtonian values. We use the rotation matrix relating the rotation of the body fixed coordinate system to the Newtonian coordinate system to get

$$A_{i/base} = [\mathrm{C}^{bn/N}]A_{i/base} = [\mathrm{C}^{b2/N}]^{n-1}A_{i/base} \tag{6.12}$$

We must solve the acceleration problem for each of the previous bays to find the Newtonian acceleration of the origin of the coordinate system, A_{base}. As with the velocity of the origin, the acceleration is given as

$$A_{base} = \frac{A7_{n-1} + A8_{n-1} + A9_{n-1}}{3} \tag{6.13}$$

where:

$A7_{n-1}, A8_{n-1}, A9_{n-1}$= vectors containing the Newtonian accelerations (in the $\hat{n}_1, \hat{n}_2$, and $\hat{n}_3$ directions) of the top plane nodes of the $n-1$ bay

Calculation of the angular accelerations is similar to that of the angular velocities. The angular acceleration of the coordinate system attached to the n^{th} bay is the same as the angular acceleration of the top plane of the $n-1$ bay. The angular acceleration of the top plane of any bay with respect to its base plane is provided by Warrington's kinematic analysis. The angular acceleration of the n^{th} coordinate system is the sum of all of the previous bays, calculated as

$$\left\{\alpha_{n/N}\right\} = \left\{\alpha_{b1/N}\right\} + \left\{\alpha_{b2/b1}\right\} + ... + \left\{\alpha_{n-1/n-2}\right\} + \left\{\alpha_{n/n-1}\right\} \tag{6.14}$$

Once again, it is necessary that each of the angular accelerations be expressed in the same coordinate system. Knowing these values, we can then calculate the Newtonian accelerations of the node points. These accelerations will be used to calculate the forces in each of the bays.

6.2 Force Analysis

Once we have the Newtonian position, velocity, and acceleration of each of the nodes of each bay, we can use the information to perform the force analysis of the members in each bay. We perform the force analysis in the opposite order that we performed the kinematic analysis, that is we start at the n^{th} bay and work backwards to the base.

The model developed in Chapter 4 for a single bay needs very little modification to be used for the multiple bay case. The method for performing the force analysis is as follows. We first look at the top bay in the truss. Since we know all of the accelerations, we can find all of the nodal forces. The only difference is that the base nodes (nodes N1, N2, and N3) are no longer fixed but have some velocity and acceleration. The model developed in Chapter 4 allows for motion of the base nodes; they are set to 0 for the single bay case where they are fixed to ground. In this analysis, they are no longer fixed but have some motion that was calculated above. The complete force analysis for the n^{th} bay is then performed using the simulation presented in Chapter 5.

Besides solving for the nodal forces, the force analysis will also provide the reaction forces at the base nodes (in the Newtonian coordinates). The resulting reaction forces of the n^{th} bay must be input as applied forces at the top plane nodes of the $n-1$ bay. The N1 node of the n^{th} bay corresponds with the N7 node of the $n-1$ bay; N2 of bay n corresponds with N9 of bay $n-1$; and N3 of bay n corresponds with N8 of bay $n-1$. As the analysis for each bay is completed, the reaction forces are then applied to the next bay down the chain. When the bottom bay of the chain is reached, the reaction forces are the forces exerted by the entire truss on the ground.

There are no other modifications necessary to the force analysis rather than feeding the reaction forces down the chain of bays.

6.3 Summary

In this chapter we have shown how the kinematics and force analysis model developed for a single bay truss can be extended to multiple bay trusses. This extension was not implemented in a simulation, however. The simulation routine developed for the single bay case was not modified to deal with the multiple bay case.

Chapter 7

Conclusions and Recommendations

7.1 Conclusions

The objective of this thesis was to find a method of analyzing the forces generated within an adaptive truss during some motion. A triangular truss was first analyzed since its geometry and kinematics are much less complex than three dimensional truss. The triangular truss was modelled and a simulation program was developed which predicted the internal forces in the truss members caused by both static and inertial loading. The static forces calculated by the simulation program compared exactly with the results of a finite-element analysis of the triangular truss. No experimental comparison was done for the triangular truss.

Using the insight gained from the triangular truss analysis, a three-dimensional truss consisting of a single double-octahedral bay was modelled. The equations of motion were set up for the truss and a simulation program was developed that completely analyzed the internal forces generated in the truss. The simulation calculates both static forces and dynamic forces. The static results were checked against the results

of a finite-element analysis and also with experiments on a prototype truss. The static forces matched exactly with the finite-element model. A maximum of 4.3 percent error was found when comparing the static simulation results to a static experiment using the prototype truss. Two dynamic simulations were then compared to dynamic experiments using the truss. In this case, there was a constant offset error between the experimental and simulated results for both experiments. One possible cause for the error is drifting in the strain gages during the experiment. Except for the constant offset, the simulated strains compare very well with the measured strains. The good comparison between the measured and simulated strains provides confidence that the model is accurate.

Once the model was developed for a single double-octahedral bay, the kinematic and force analysis process was extended to support a chain of n double-octahedral bays. The kinematics must be done from the fixed base to the end bay, while the force analysis must be started at the end bay and completed at the fixed base. It was shown that the kinematics solution developed by Warrington (25) can be used to find the position, velocity, and acceleration of each of the nodes in each bay in the chain in bay-fixed coordinates. A transformation between the bay-fixed coordinate system and the Newtonian reference coordinate system was defined and allows the accelerations to be presented in the Newtonian coordinates. The base reactions of each bay in the chain are passed on to the next bay as external loads applied at the top plane nodes. Little modification of the simulation program is required to support the chain truss analysis. Once the the existing, single-bay simulation program is modified to include the multiple bay effects, the forces generated in each member of the entire n-bay truss can calculated. The method was presented, however, it was

not implemented in a simulation.

One of the main motivations for the work presented in this thesis is the need to design adaptive trusses to be as light-weight as possible. To do this it is necessary to know the maximum loads (both static and dynamic) that each member will be subjected to. The force analysis presented in this thesis can be used as the basis for an adaptive truss design code. Based on design specifications such as type of motion, time of motion, and loads carried, the force analysis can be used to find the member dimensions required so that the truss members will not fail. The calculated motor parameters can be used to size the motors and select gearhead/leadscrew combinations that will provide the specified motion.

7.2 Recommendations

There are numerous areas for future research. The most important area is to further verify the results of the model. A new experimental truss should be built to provide higher speeds and thus induce larger inertial loads in the members. All members of the truss should be instrumented with strain gages set up so that the principle strains can be found. Experiments to correlate all internal forces (including bending moments and shear stresses) with those predicted by the model should be performed.

Another area for future work is to develop a more realistic model of the truss. This model should take into account the offsets in the truss joints and the non-ideal connections of the members. The more realistic model might alleviate some of the differences between the simulated and experimental strains.

A detailed modification of the simulation program to predict the loads occurring in a chain truss should be performed. It would also be beneficial if the experimental truss recommended above was capable of having multiple bays added to it.

List Of References

1. Beer, F. P., and E. R. Johnston,Jr.,Vector Mechanics for Engineers, McGraw-Hill Book Company, New York, NY, 1977.

2. Brennan, P. M., "Antagonistic Control for a Planar Variable Geometry Truss," Proposed Masters Thesis, Virginia Polytechnic Institute and State University, Blacksburg, VA, 1991.

3. Chalhoub, N. G., A. G. Ulsoy, "Dynamic Simulation of a Leadscrew Driven Flexible Robot Arm and Controller," *Journal of Dynamic Systems, Measurement, and Control*, Vol. 108, pp. 119-126, 1986.

4. Chalhoub, N. G., and A. G. Ulsoy, "Control of a Flexible Robot Arm: Experimental and Theoretical Results," *Journal of Dynamic Systems, Measurement, and Control*, Vol. 109, pp. 299-309, 1987.

5. Clark, W. W., B. Kimiavi, and H. H. Robertshaw,"Control of Flexible Beams Using a Free-Free Active Truss," *Proceedings of the 1989 ASME Winter Annual Meeting - Adaptive Structures Session*, pp. 61-68, 1989.

6. Gupta, V. K., "Dynamic Analysis of Multi-Rigid-Body Systems," *Journal of Engineering for Industry*, Vol. 96, pp. 886-892, 1974.

7. Kung, H. F., "Dynamics and Control of a Spatial Truss Actuator", Masters Thesis, Virginia Polytechnic Institute and State University, Blacksburg, VA, May, 1988.

8. Housner, J. M., et. al., LATDYN Users Manual - Preliminary, Prepared under NASA contract NAS1-18478, 1989.

9. Lovejoy, V. D., H. H. Robertshaw, W. N. Patten, and C. G. Horner,"Dynamics and Control of a Planar Truss Actuator," *Vibration Control and Active Vibration Suppression*, DE Vol-4, D. J. Inman and J. C. Simonis, Ed., ASME, New York, Dec. 1987.

10. MSC/PAL2 Advanced Stress and Vibration Analysis Reference Manual, MacNeal-Schwendler Corporation, Los Angeles, CA, 1989.

11. Meirovitch, L., Methods of Analytical Dynamics, McGraw-Hill Book Company, New York, NY, 1970.

12. Orin,S. Y., "Kinematic and Kinetic Analysis of Open-Chain Linkages Utilizing Newton-Euler Methods," *Mathematical Biosciences*, Vol. 43, 107-130, 1979.

13. Padmanhaban, B., V. Arun, and C. F. Reinholtz, "Closed-Form Inverse Kinematic Analysis of Variable Geometry Truss Manipulators," *Transactions of the ASME, Journal of Mechanical Design*, 1990.

14. Perry, C. C., and H. R. Lissner, The Strain Gage Primer, McGraw-Hill Book Company, New York, NY, 1962.

15. Reinholtz, C. F., P. Tidwell, H. H. Robertshaw, and C. G. Horner, "Kinematic Analysis of Generalized Adaptive Trusses," *Proceedings, First Joint US/JAPAN Conference on Adaptive Structures*, pp. 497-508, 1990.

16. Reinholtz, C. F., and D. Gokhale, "Design and Analysis of Variabe Geometry Truss Robots," *Proceedings of the 10th Applied Mechanisms Conference*, 1987.

17. Rhodes, M. D., and M. M. Mikulas, Jr., "Deployable Controllable Geometry Truss Beam," NASA Technical Memorandum, June 1985.

18. Robertshaw, H. H., R. H. Wynn, Jr., H. F. Kung, S. L. Hendricks, and W. W. Clark, "Dynamics and Control of a Spatial Active Truss Actuator," *Structures, Structural Dynamics, and Materials Conference, 30th*, April 1989, AIAA Paper no. 89-1328.

19. Shigley, J. E., and L. D. Mitchell, Mechanical Engineering Design, McGraw-Hill Book Company, New York, NY, 1983.

20. Stulce, J. R.,W. D. Burgos, S. G. Dhande, and C. F. Reinholtz, "Conceptual Design of a Multibody Passive-Legged Crawling Vehicle," *Proceedings of the 1990 ASME Mechanisms Conference*, 1990.

21. Utku, S., A. V. Ramesh, S. K. Das, B. K. Wada, and G. S. Chen, "Control of a Slow Moving Space Crane as an Adaptive Truss," *Structures, Structural*

Dynamics, and Materials Conference, 30th, April 1989, AIAA Paper no. 89-1286-CP.

22. Wada, B. K., "Adaptive Structures," *Structures, Structural Dynamics, and Materials Conference, 30th*, April 1989, AIAA Paper no. 89-1160-CP.

23. Warrington, T. J., and C. G. Horner, "Flexible Beam Control Using an Adaptive Truss," *Proceedings of the 1991 American Control Conference*, Vol. 1, pp. 430-439.

24. Warrington, T. J., W. W. Clark, H. H. Robertshaw, and C. G. Horner, "The Analysis and Large-Angle Control of a Flexible Beam Using an Adaptive Truss," *Structures, Structural Dynamics, and Materials Conference, 32th*, April, 1991.

25. Warrington, T. J., "Experiments and Simulations of Large Angle Flexible Beam Control Using an Adaptive Truss," Proposed Masters Thesis, Virginia Polytechnic Institute and State University, Blacksbuurg, VA, 1991.

26. Wynn, R. H.,Jr., H. H. Robertshaw, and C. G. Horner, "An Analytical Study of a Six Degree-of-Freedom Active Truss for Use in Vibration Control," *Structures, Structural Dynamics, and Materials Conference, 31st*, Long Beach, CA, April 1990, AIAA Paper no. 90-1164-CP.

Appendix A

Triangular Simulation Code

```
%-----------------------------------------------------------
%      PROGRAM: TRIAGSIM.M
%
%          This program calculates the time response of a triangular
%             truss with one active link
%
%                      a  /\
%                        X  \  b
%                  lpos /_ _ \
%                             d
%-----------------------------------------------------------

%   Read in input data

%l4=input('Enter the base length: ');
%l2=input('Enter length of actuated bar (w/o actuator): ');
%lstart=input('Enter starting actuator length: ');
%lend=input('Enter final actuator length: ');
%l1=input('Enter length of unactuated bar: ');
%mass1=input('Enter mass of unactuated bar: ');
%tt=input('Enter total time of motion: ');
%dt=input('Enter time step: ');
%numsteps=tt/dt;

l1=2.0;             % Length of unactuated bar (m)
l2=2.0;             % Length of lead screw tube (m)
l3min=.5;           % Minimum length of active base (m)
l3max=2.1;          % Overall length of actuator (m)
l4=2.0;             % Ground link length (m)
lstart=0;           % Starting actuator variable length (m)
```

```
lend=1.2;          % Ending actuator variable length (m)
cg1=l1/2;          % COG of unactuated member (m)
cg2=l2/2;          % COG of lead screw tube (m)
cg3=l3max/2;       % COG of active base (m)

tt=input('Enter Total Simulation Time: ');
dt=input('Enter Time Step: ');
numsteps=tt/dt ;    % Number of steps in the simulation

%  Values for Triangular Truss Simulation

id=.022225; % Inner diameter of unactuated
            %    and ls tubes (m) (7/8 in)
od=.0254    % Outer diameter of unactuated
            %    and ls tubes (m) (1 in)
area=pi*(od^2-id^2)/4;  % Cross sectional area of
                        %unactuated and ls tubes (m^2)
E=71.0e9;   % Modulus of Elasticity for aluminum (N/m^2)
G=26.2e9;   % Modulus of Rigidity for aluminum (N/m^2)
ro=2711.5;  % Density of aluminum (N/m^3)
c=od/2;     % Maximum bending moment - distance from
            %      neutral axis (m)
g=-9.81;    % Gravitational constant for truss members (m/s^2)
g1=-9.81;   % Gravitational constant for lumped mass (m/s^2)

mass1=ro*area*l1;        % Mass of unactuated tube
mass2=ro*area*l2;        % Mass of lead screw tube
mass3=2.4;               % Mass of actuator including motor and ls
%mass4=10;               % Lumped mass at the top
mass4=input('Enter Lumped Mass Value: ');
% Moment of Inertia of ls tube (about CG)
ic=mass2*(3*od^2+3*id^2+4*l2^2)/48;
% Moment of Inertia of ls tube (about CG)
io1=mass1*(3*od^2+3*id^2+4*l1^2)/48+mass1*l1^2/4;
io2=1/3*mass3*l3max^2;  % MOI of actuator about its base
ina=pi*(od^4-id^4)/64;  % MOI resisting bending moment

%    Input motor characteristics
Jm=6.76e-6;       % Motor armature inertia
Ra=2;             % Armature resistance
Ggh=5.2;          % Gear head gain
Gls=2.5266e-4;    % Leadscrew gain
Kt=0.0398;        % Motor Torque constant
Kb=0.0398;        % Back EMF constant
```

```
mu=0.3;          % Coefficient of Friction
dm=0.02;         % Leadscrew mean diameter

%   Initialize all calculated variables
phi=zeros(numsteps,1);
phidot=zeros(numsteps,1);
phiddot=zeros(numsteps,1);
theta=zeros(numsteps,1);
thetadot=zeros(numsteps,1);
thetaddot=zeros(numsteps,1);
xcoord=zeros(numsteps,1);
ycoord=zeros(numsteps,1);
lvel=zeros(numsteps,3);
lpos=zeros(numsteps,2);
aax=zeros(numsteps,1);
aay=zeros(numsteps,1);
abx=zeros(numsteps,1);
aby=zeros(numsteps,1);
acx=zeros(numsteps,1);
acy=zeros(numsteps,1);
adx=zeros(numsteps,1);
ady=zeros(numsteps,1);
fx1=zeros(numsteps,1);
fy1=zeros(numsteps,1);
fx2=zeros(numsteps,1);
fy2=zeros(numsteps,1);
fx3=zeros(numsteps,1);
fy3=zeros(numsteps,1);
rx1=zeros(numsteps,1);
rx2=zeros(numsteps,1);
ry1=zeros(numsteps,1);
ry2=zeros(numsteps,1);
m1=zeros(numsteps,1);
fx1st=zeros(numsteps,1);
fy1st=zeros(numsteps,1);
fx2st=zeros(numsteps,1);
fy2st=zeros(numsteps,1);
fx3st=zeros(numsteps,1);
fy3st=zeros(numsteps,1);
rx1st=zeros(numsteps,1);
rx2st=zeros(numsteps,1);
ry1st=zeros(numsteps,1);
ry2st=zeros(numsteps,1);
m1st=zeros(numsteps,1);
```

```
tension1=zeros(numsteps,7);
shear1=zeros(numsteps,7);
moment1=zeros(numsteps,7);
tension2=zeros(numsteps,7);
shear2=zeros(numsteps,7);
moment2=zeros(numsteps,7);
tstres1=zeros(numsteps,7);
mstres1=zeros(numsteps,7);
vstres1=zeros(numsteps,7);
tstres2=zeros(numsteps,7);
mstres2=zeros(numsteps,7);
vstres2=zeros(numsteps,7);
tstrain1=zeros(numsteps,7);
vstrain1=zeros(numsteps,7);
mstrain1=zeros(numsteps,7);

% Now set up third order transfer function model for actuator
% and simulate to get the link length and linear velocity
%    - the states of the plant
zeta=0.8;
ts=.9*tt;
wn=6/(zeta*ts);
rroot=2*wn;
ce1=poly(roots([1 2*zeta*wn wn^2]'));
ce2=poly(roots([1 rroot]'));
den=conv(ce1,ce2);
T=(0:dt:tt-dt)';
num=den(4);
U1=((lend-lstart))*ones(numsteps,1);
[lpos,lvel]=lsim(num,den,U1,T);
xvel=lvel(:,2)*num;
xacel=lvel(:,1)*num;

for i = 1:numsteps;

   %  Calculate angles, angular velocities, and accelerations
   %     at each time step
if i==1;
   xvel1(i)=0;
   xacel1(i)=0;
   phi(i)=acos((l1^2+l4^2-(l2+l3min+lpos(i))^2)/(2*l4*l1));
   theta(i)=acos(((l2+l3min+lpos(i))^2+l4^2-l1^2)/...
                        (2*l4*(l2+l3min+lpos(i))));
   thetadot(i)=0;
```

```
   phidot(i)=0;
   phiddot(i)=0;
  else
   xvel1(i)=(lpos(i)-lpos(i-1))/dt;
   xacel1(i)=(xvel(i)-xvel(i-1))/dt;
   phi(i)=acos((l1^2+l4^2-(l2+l3min+lpos(i))^2)/(2*l4*l1));
   theta(i)=acos(((l2+l3min+lpos(i))^2+l4^2-l1^2)/...
                        (2*l4*(l2+l3min+lpos(i))));
   thetadot(i)=(theta(i)-theta(i-1))/dt;
   thetaddot(i)=(thetadot(i)-thetadot(i-1))/dt;
   phidot(i)=(phi(i)-phi(i-1))/dt;
   phiddot(i)=(phidot(i)-phidot(i-1))/dt;
end

   xcoord(i)=l4-(l1*cos(phi(i)));
   ycoord(i)=l1*sin(phi(i));

% Calculate x and y accelerations of each point of
%  interest (node points and center of mass of each member)

   aax(i)=((cg1)*phidot(i)^2*cos(phi(i)))+...
                ((cg1)*phiddot(i)*sin(phi(i)));
   aay(i)=(-(cg1)*phidot(i)^2*sin(phi(i)))+...
                ((cg1)*phiddot(i)*cos(phi(i)));
   abx(i)=(l1*phidot(i)^2*cos(phi(i)))+...
                (l1*phiddot(i)*sin(phi(i)));
   aby(i)=(-l1*phidot(i)^2*sin(phi(i)))+...
                (l1*phiddot(i)*cos(phi(i)));
   acx(i)=(xacel(i)*cos(theta(i)))-(2*xvel(i)*...
                 thetadot(i)*sin(theta(i)))-...
                 ((lpos(i)+l3min+cg2)*thetaddot(i)...
                 *sin(theta(i)))-((lpos(i)+l3min+cg2)*...
                 thetadot(i)^2*cos(theta(i)));
   acy(i)=(xacel(i)*sin(theta(i)))+(2*xvel(i)*...
                 thetadot(i)*cos(theta(i)))+...
                 ((lpos(i)+l3min+cg2)*thetaddot(i)*...
                 cos(theta(i)))-((lpos(i)+l3min+cg2)*...
                 thetadot(i)^2*sin(theta(i)));
   adx(i)=(-cg3*thetaddot(i)*sin(theta(i)))-...
                 (cg3*thetadot(i)^2*cos(theta(i)));
   ady(i)=(cg3*thetaddot(i)*cos(theta(i)))-...
                 (cg3*thetadot(i)^2*sin(theta(i)));

   % Set up the system of Newtons equations and solve
```

```
   %       for the forces
   val1=(l2-cg2);
   val3=(lpos(i)+l3min)*sin(theta(i));
   val4=(lpos(i)+l3min)*cos(theta(i));
forcevect=[mass1*aax(i) mass1*aay(i)+mass1*g...
 io1*phiddot(i)+mass1*g*cg1*cos(phi(i)) mass2*acx(i)...
 mass2*acy(i)+mass2*g ic*thetaddot(i) mass3*adx(i)...
 mass3*ady(i)+mass3*g io2*thetaddot(i)+mass3*g*cg3*cos(theta(i))...
 mass4*abx(i) mass4*aby(i)+mass4*g1]';

   e1=[1 0 0 0 0 0 1 0 0 0 0];
   e2=[0 0 0 1 0 0 0 0 1 0 0];
   e3=[l1*sin(phi(i)) 0 0 l1*cos(phi(i)) 0 0 0 0 0 0 0];
   e4=[0 1 1 0 0 0 0 0 0 0 0];
   e5=[0 0 0 0 1 1 0 0 0 0 0];
   e6=[0 -val1*sin(theta(i)) cg2*sin(theta(i)) 0 val1*cos(theta(i))...
       -cg2*cos(theta(i)) 0 0 0 0 1];
   e7=[0 0 -1 0 0 0 0 1 0 0 0];
   e8=[0 0 0 0 0 -1 0 0 0 1 0];
   e9=[0 0 val3 0 0 -val4 0 0 0 0 -1];
   e10=[-1 -1 0 0 0 0 0 0 0 0 0];
   e11=[0 0 0 -1 -1 0 0 0 0 0 0];


   eqnmat=[e1;e2;e3;e4;e5;e6;e7;e8;e9;e10;e11];
   res=inv(eqnmat)*forcevect;

   %  The resulting forces for each time step are:

   fx1(i)=res(1);
   fx2(i)=res(2);
   fx3(i)=res(3);
   fy1(i)=res(4);
   fy2(i)=res(5);
   fy3(i)=res(6);
   rx1(i)=res(7);
   rx2(i)=res(8);
   ry1(i)=res(9);
   ry2(i)=res(10);
   m1(i)=res(11);

%    Now Calculate Static Forces at each timestep

statvect=[0 mass1*g mass1*g*cg1*cos(phi(i)) 0 mass2*g 0 0...
```

```
        mass3*g mass3*g*cg3*cos(theta(i)) 0 mass4*g1]';
fstat=inv(eqnmat)*statvect;
   fx1st(i)=fstat(1);
   fx2st(i)=fstat(2);
   fx3st(i)=fstat(3);
   fy1st(i)=fstat(4);
   fy2st(i)=fstat(5);
   fy3st(i)=fstat(6);
   rx1st(i)=fstat(7);
   rx2st(i)=fstat(8);
   ry1st(i)=fstat(9);
   ry2st(i)=fstat(10);
   m1st(i)=fstat(11);

%   Calculate the actuator force on the leadscrew tube
c1=[-sin(theta(i)) cos(theta(i)) 0;cos(theta(i)) ...
                                sin(theta(i)) 0;0 0 1];
f=[fx3(i) fy3(i) 0]';
actfor(:,i)=c1*f;

%   Calculate static load on leadscrew tube at each time step
fst=[fx3st(i) fy3st(i) 0]';
ffst(:,i)=c1*fst;

%    Determine whether motor torque is with or against load
sst=sign(ffst(2,i));
sact=sign(actfor(2,i));


%  Calculate motor torque
if sst==sact;          %  Signs are the same - force acting
                       %       against load
   Tm(i)=(Jm*abs(xacel(i))*Ggh/Gls)+(dm/2*actfor(2,i))*...
                ((2*pi*Gls+pi*mu*dm)/(pi*dm-mu*2*pi*Gls));
  else                 %  Signs are different - force acting
                       %       with load
   Tm(i)=(Jm*abs(xacel(i))*Ggh/Gls)+(dm/2*actfor(2,i))*...
               ((-2*pi*Gls+pi*mu*dm)/(pi*dm+mu*2*pi*Gls));
end
%  Calculate motor voltage
Vm(i)=Tm(i)*Ra/Kt+(Kb*Ggh*abs(xvel(i))/(Gls));
%  Calculate motor current
Ia(i)=Tm(i)/Kt;
```

```
%   Now find the internal forces due to inertia in each member

  for j = 1:7;

%     First for unactuated tube:
%
    ii=(l1/6)*(j-1);
    iz=(mass1*ii/l1)*(3*od^2+3*id^2+4*l1^2)/48;
    w=mass1*ii/l1*g;
    gamma=((pi/2)-phi(i));
    azx=((l1-ii/2)*phidot(i)^2*cos(phi(i)))+((l1-ii/2)*...
                                phiddot(i)*sin(phi(i)));
    azy=(-(l1-ii/2)*phidot(i)^2*sin(phi(i)))+((l1-ii/2)*...
                                 phiddot(i)*cos(phi(i)));
    mat1=[cos(phi(i)) -cos(gamma) 0;-sin(phi(i)) -sin(gamma)...
                                                  0;0 ii/2 -1];
    fvect1=[((mass1*ii/l1)*azx)-fx1(i);((mass1*ii/l1)*azy)-...
            fy1(i)+w;(iz*phiddot(i))-(fx1(i)*ii/2*sin(phi(i)))-...
            (fy1(i)*ii/2*cos(phi(i)))];
    ans1=inv(mat1)*fvect1;
    tension1(i,j)=ans1(1);
    shear1(i,j)=ans1(2);
    moment1(i,j)=ans1(3);
    tstres1(i,j)=tension1(i,j)/area;
    vstres1(i,j)=2*shear1(i,j)/area;
    mstres1(i,j)=moment1(i,j)*c/ina;
    vmstres1a(i,j)=sqrt(tstres1(i,j)^2+vstres1(i,j)^2);
    vmstres1b(i,j)=abs(mstres1(i,j))+abs(tstres1(i,j));
    tstrain1(i,j)=tstres1(i,j)/E;
    mstrain1(i,j)=mstres1(i,j)/E;
    vstrain1(i,j)=vstres1(i,j)/G;
    fstat1=[-fx1st(i);-fy1st(i)+w;-(fx1st(i)*ii/2*sin(phi(i)))-...
            (fy1st(i)*ii/2*cos(phi(i)))];
    ans1=inv(mat1)*fstat1;
    tstat1(i,j)=ans1(1);
    shstat1(i,j)=ans1(2);
    mstat1(i,j)=ans1(3);
    tstst1(i,j)=tstat1(i,j)/area;
    vstst1(i,j)=3*shstat1(i,j)/(2*area);
    mstst1(i,j)=mstat1(i,j)*c/ina;
    vmstst1a(i,j)=sqrt(tstst1(i,j)^2+vstst1(i,j)^2);
    vmstst1b(i,j)=abs(mstst1(i,j))+abs(tstst1(i,j));
    tststr1(i,j)=tstst1(i,j)/E;
    mststr1(i,j)=mstst1(i,j)/E;
```

```
vststr1(i,j)=vstst1(i,j)/G;

%     For lead screw tube:

kk=(l2/6)*(j-1);
is=(mass2*kk/l2)*(3*od^2+3*id^2+4*l2^2)/48;
w=mass2*kk/l2*g;
gamma=((pi/2)-theta(i));
asx=(xacel(i)*cos(theta(i)))-(2*xvel(i)*thetadot(i)*...
     sin(theta(i)))-((lpos(i)+l3min+l2-kk/2)*thetaddot(i)*...
     sin(theta(i)))-((lpos(i)+l3min+l2-kk/2)*thetadot(i)^2*...
     cos(theta(i)));
asy=(xacel(i)*sin(theta(i)))+(2*xvel(i)*thetadot(i)*...
     cos(theta(i)))+((lpos(i)+l3min+l2-kk/2)*thetaddot(i)*...
     cos(theta(i)))-((lpos(i)+l3min+l2-kk/2)*thetadot(i)^2*...
     sin(theta(i)));
mat2=[-cos(theta(i)) -cos(gamma) 0;-sin(theta(i)) sin(gamma)...
                                                  0;0 -kk/2 1];
fvect2=[((mass2*kk/l2)*asx)-fx2(i);((mass2*kk/l2)*asy)-...
        fy2(i)+w;(is*thetaddot(i))+(fx2(i)*kk/2*sin(theta(i)))-...
        (fy2(i)*kk/2*cos(theta(i)))];
ans2=inv(mat2)*fvect2;
tension2(i,j)=ans2(1);
shear2(i,j)=ans2(2);
moment2(i,j)=ans2(3);
tstres2(i,j)=tension2(i,j)/area;
vstres2(i,j)=2*shear2(i,j)/area;
mstres2(i,j)=moment2(i,j)*c/ina;
vmstres2a(i,j)=sqrt(tstres2(i,j)^2+vstres2(i,j)^2);
vmstres2b(i,j)=abs(mstres2(i,j))+abs(tstres2(i,j));
tstrain2(i,j)=tstres2(i,j)/E;
mstrain2(i,j)=mstres2(i,j)/E;
vstrain2(i,j)=vstres2(i,j)/G;
fstat2=[-fx2st(i);-fy2st(i)+w;(fx2st(i)*kk/2*sin(theta(i)))-...
        (fy2st(i)*kk/2*cos(theta(i)))];
ans1=inv(mat2)*fstat2;
tstat2(i,j)=ans1(1);
shstat2(i,j)=ans1(2);
mstat2(i,j)=ans1(3);
tstst2(i,j)=tstat2(i,j)/area;
vstst2(i,j)=3*shstat2(i,j)/(2*area);
mstst2(i,j)=mstat2(i,j)*c/ina;
vmstst2a(i,j)=sqrt(tstst2(i,j)^2+vstst2(i,j)^2);
vmstst2b(i,j)=abs(mstst2(i,j))+abs(tstst2(i,j));
```

```
   tststr2(i,j)=tstst2(i,j)/E;
   mststr2(1,j)=mstst2(i,j)/E;
   vststr2(i,j)=vstst2(i,j)/G;
  end;

end;

%       Now check buckling load for both members
%       Only need to use the maximum forces for buckling

%   For member 1
Pcr1=pi^2*E*ina/l1^2;
%   The maximum compressive load is the minimum tensile load found
%       from the analysis of internal forces
maxf1=max(abs(min(tension1)));

%   For member 2
Pcr2=pi^2*E*ina/l2^2;
%   The maximum compressive load is the minimum tensile load
maxf2=max(abs(min(tension2)));
```

Appendix B

Triangular Truss Finite-Element Model

```
%-------------------------------------------------------
%
%        MSC/PAL Model File
%
%-------------------------------------------------------
Title    Triangular Truss - Static Analysis
Nodal Point Locations 1
1    0.000000    0.000000  0.000000
2    0.312800    0.390300  0.000000
3    1.562500    1.951600  0.000000
4    2.000000    0.000000  0.000000
-- Blank Line --
Material Properties 71.0e9  26.2e9  2711.5  .3
Beam Type 3, .0254, .022225
release u,v
Connect 1 to 2
release u,v
Connect 3 to 2
beam type 1 1.1876e-4
Connect 3 to 4
 zero 1
ra of all
tz of all
-- Blank Line --
End Definition


%-------------------------------------------------------
%
%        MSC/PAL Loading File
%
%-------------------------------------------------------
```

```
displacements applied 1
tx 0 1 4
ty 0 1 4
--blank line--
forces and moments applied 1
fy -98.1 3
--blank line--
solve
```

Appendix C

Spatial Truss Simulation Code

```
%-----------------------------------------------------------------------
%
% PROGRAM:  SPATSIM.M
%
% This program is the simulation to model the spatial truss with
%         21 members
%
%      Written by:  David Lacy
%
% Run Tom's slewing simulation  - output is nodal position,
%       velocity and acceleration at each point in time
%-----------------------------------------------------------------------
%   Run Tom's Kinematics Solution
fwd1

numsteps=length(l1);          %Calculate number of steps in motion
tt=input('Enter Timestep: '); %Input delta t

%     Now load Tom's information in my format
%       First load position data
for i=1:numsteps;
N1(i,:)=[n1(1) -n1(3) n1(2)];
N2(i,:)=[n3(1) -n3(3) n3(2)];
N3(i,:)=[n2(1) -n2(3) n2(2)];
end;
N4=[n4(:,1) -n4(:,3) n4(:,2)];
N5=[n5(:,1) -n5(:,3) n5(:,2)];
N6=[n6(:,1) -n6(:,3) n6(:,2)];
N7=[n9(:,1) -n9(:,3) n9(:,2)];
N8=[n7(:,1) -n7(:,3) n7(:,2)];
```

```
N9=[n8(:,1) -n8(:,3) n8(:,2)];
%       Load velocity data
V1=zeros(numsteps,3);
V2=zeros(numsteps,3);
V3=zeros(numsteps,3);
V4=[nd4(1,:)' -nd4(3,:)' nd4(2,:)'];
V5=[nd5(1,:)' -nd5(3,:)' nd5(2,:)'];
V6=[nd6(1,:)' -nd6(3,:)' nd6(2,:)'];
V7=[nd9(1,:)' -nd9(3,:)' nd9(2,:)'];
V8=[nd7(1,:)' -nd7(3,:)' nd7(2,:)'];
V9=[nd8(1,:)' -nd8(3,:)' nd8(2,:)'];
%      Load Acceleration Data
A1=zeros(numsteps,3);
A2=zeros(numsteps,3);
A3=zeros(numsteps,3);
A4=[ndd4(1,:)' -ndd4(3,:)' ndd4(2,:)'];
A5=[ndd5(1,:)' -ndd5(3,:)' ndd5(2,:)'];
A6=[ndd6(1,:)' -ndd6(3,:)' ndd6(2,:)'];
A7=[ndd9(1,:)' -ndd9(3,:)' ndd9(2,:)'];
A8=[ndd7(1,:)' -ndd7(3,:)' ndd7(2,:)'];
A9=[ndd8(1,:)' -ndd8(3,:)' ndd8(2,:)'];

%  Now save only acceleration data, time step, total time
save node_dat N1 N2 N3 N4 N5 N6 N7 N8 N9 V1 V2 V3 V4 V5
              V6 V7 V8 V9 A1 A2 A3 A4 A5 A6 A7 A8 A9 numsteps tt dt
clear
load node_dat
t=[dt:dt:numsteps*dt]';

%  Now have only the nodal position, velocity, and acceleration, and the
%    total simulation time(tt), number of steps(numsteps), and
%    time step(dt)

%  Set up motor constants
Jm=6.76e-6;      % Motor armature inertia
Ra=2;            % Armature resistance
Ggh=5.2;         % Gear head gain
Gls=2.5266e-4;   % Leadscrew gain (1/16 in per rev)
Kt=0.0398;       % Motor Torque constant
Kb=0.0398;       % Back EMF constant
mu=.1            % Coefficient of friction
dm=0.5*.0254     % Mean diameter of leadscrew

%  Set up material constants and gravity
```

```
ro=2711.5;          % Mass per unit volume (total mass=ro*area*L)
E=71.0e9;         % Modulus of Elasticity

%    Gravitational Constant - positive 'g' acts toward
%                              fixed base of truss)
g=input('Enter Gravity Constant for
        members (positive is towards base plane): ');

%    Apply any External Loads or Lumped Masses
%        Lumped mass can have different gravitational constant if desired

resp=input('Do you want to add Lumped Masses to the Top Plane
               Nodes (y/n): ');
if resp=='y';
     lmass7=input('Enter Lumped Mass Value for Node 7: ');
     lmass8=input('Enter Lumped Mass Value for Node 8: ');
     lmass9=input('Enter Lumped Mass Value for Node 9: ');
     g1=input('Enter Gravity Constant for Lumped Mass (positive is toward
                base plane): ');
   else
     lmass7=0;
     lmass8=0;
     lmass9=0;
     g1=0;
end
resp=input('Do you want to add External Forces to the Top Plane
               Nodes (y/n): ');
if resp=='y'
   NN7x=input('Enter X-Direction External Load Applied to Node 7: ');
   NN7y=input('Enter Y-Direction External Load Applied to Node 7: ');
   NN7z=input('Enter Z-Direction External Load Applied to Node 7: ');
   NN8x=input('Enter X-Direction External Load Applied to Node 8: ');
   NN8y=input('Enter Y-Direction External Load Applied to Node 8: ');
   NN8z=input('Enter Z-Direction External Load Applied to Node 8: ');
   NN9x=input('Enter X-Direction External Load Applied to Node 9: ');
   NN9y=input('Enter Y-Direction External Load Applied to Node 9: ');
   NN9z=input('Enter Z-Direction External Load Applied to Node 9: ');
  else
   NN7x=0;
   NN7y=0;
   NN7z=0;
   NN8x=0;
   NN8y=0;
   NN8z=0;
```

```
   NN9x=0;
   NN9y=0;
   NN9z=0;
end;
cls
disp(' ')
disp(' ')
disp('          Calculating Dynamic Forces ')
disp(' ')
disp('                Please Wait')
disp(' ')

%  Set up constants for the active member (base and ls tube)

ls=.254;              % Length of Lead Screw Tube
odls=.75*2.54/100;    % Outer Diameter of lead screw tubes
idls=.63*2.54/100;    % Inner Diameter of lead screw tubes
areals=pi/4*(odls^2-idls^2);% Cross section area of lead screw tubes
massls=ro*areals*ls; % Mass of lead screw tube
iglsb1=massls/48*(3*odls^2+3*idls^2+4*ls^2); % Moment of Inertia of
                      % lead screw tube about its center of
                      % gravity (B1 direction)
iglsb2=iglsb1;       % MOI of ls tube about COG in B2 direction
massbase=3;          % Mass of Active base (including motor,gear head,
                     % lead screw, and supports)
lbase=.7;            % Overall length of active base
ibaseb1=massbase*lbase^2/12;  % MOI of active base in B1 direction
          % - active base MOI is modeled as member with no area only
          % mass - therefore constant density of the length
          % This is not actual case since motor, gear head,
          % lead screw masses are lumped at points along
          % the base length - CG of active base is length/2
ibaseb2=ibaseb1;     % MOI of active base in B2 direction

%  Set up fixed member constants

idf=.63*2.54/100; % Internal diameter of fixed length members
odf=.75*2.54/100; % Outer diameter of fixed length members
areaf=pi/4*(odf^2-idf^2);% Cross section area of fixed length members
cf=odf/2;                % Maximum distance from neutral axis
ina=pi/64*(odf^4-idf^4); % Area moment of inertia opposing bending moment

%  Set up fixed member lengths, masses, and moments of inertia
```

```
l1=sqrt((N6(1,:)-N2(1,:))*(N6(1,:)-N2(1,:))');
mass1=ro*areaf*l1;
i1b1=mass1/48*(3*odf^2+3*idf^2+4*l1^2);
i1b2=i1b1;

l2=sqrt((N6(1,:)-N1(1,:))*(N6(1,:)-N1(1,:))');
mass2=ro*areaf*l2;
i2b1=mass2/48*(3*odf^2+3*idf^2+4*l2^2);
i2b2=i2b1;

l3=sqrt((N4(1,:)-N1(1,:))*(N4(1,:)-N1(1,:))');
mass3=ro*areaf*l3;
i3b1=mass3/48*(3*odf^2+3*idf^2+4*l3^2);
i3b2=i3b1;

l4=sqrt((N4(1,:)-N3(1,:))*(N4(1,:)-N3(1,:))');
mass4=ro*areaf*l4;
i4b1=mass4/48*(3*odf^2+3*idf^2+4*l4^2);
i4b2=i4b1;

l5=sqrt((N5(1,:)-N3(1,:))*(N5(1,:)-N3(1,:))');
mass5=ro*areaf*l5;
i5b1=mass5/48*(3*odf^2+3*idf^2+4*l5^2);
i5b2=i5b1;

l6=sqrt((N5(1,:)-N2(1,:))*(N5(1,:)-N2(1,:))');
mass6=ro*areaf*l6;
i6b1=mass6/48*(3*odf^2+3*idf^2+4*l6^2);
i6b2=i6b1;

l13=sqrt((N9(1,:)-N6(1,:))*(N9(1,:)-N6(1,:))');
mass13=ro*areaf*l13;
i13b1=mass13/48*(3*odf^2+3*idf^2+4*l13^2);
i13b2=i13b1;

l14=sqrt((N7(1,:)-N6(1,:))*(N7(1,:)-N6(1,:))');
mass14=ro*areaf*l14;
i14b1=mass14/48*(3*odf^2+3*idf^2+4*l14^2);
i14b2=i14b1;

l15=sqrt((N7(1,:)-N4(1,:))*(N7(1,:)-N4(1,:))');
mass15=ro*areaf*l15;
i15b1=mass15/48*(3*odf^2+3*idf^2+4*l15^2);
i15b2=i15b1;
```

```
l16=sqrt((N8(1,:)-N4(1,:))*(N8(1,:)-N4(1,:))');
mass16=ro*areaf*l16;
i16b1=mass16/48*(3*odf^2+3*idf^2+4*l16^2);
i16b2=i16b1;

l17=sqrt((N8(1,:)-N5(1,:))*(N8(1,:)-N5(1,:))');
mass17=ro*areaf*l17;
i17b1=mass17/48*(3*odf^2+3*idf^2+4*l17^2);
i17b2=i17b1;

l18=sqrt((N9(1,:)-N5(1,:))*(N9(1,:)-N5(1,:))');
mass18=ro*areaf*l18;
i18b1=mass18/48*(3*odf^2+3*idf^2+4*l18^2);
i18b2=i18b1;

l19=sqrt((N9(1,:)-N7(1,:))*(N9(1,:)-N7(1,:))');
mass19=ro*areaf*l19;
i19b1=mass19/48*(3*odf^2+3*idf^2+4*l19^2);
i19b2=i19b1;

l20=sqrt((N8(1,:)-N7(1,:))*(N8(1,:)-N7(1,:))');
mass20=ro*areaf*l20;
i20b1=mass20/48*(3*odf^2+3*idf^2+4*l20^2);
i20b2=i20b1;

l21=sqrt((N9(1,:)-N8(1,:))*(N9(1,:)-N8(1,:))');
mass21=ro*areaf*l21;
i21b1=mass21/48*(3*odf^2+3*idf^2+4*l21^2);
i21b2=i21b1;


% Set up Newtonian Coordinate System
n1=[1 0 0];
n2=[0 1 0];
n3=[0 0 1];
nn=[n1;n2;n3];

%         Set up loop to calculate everything for each time step

for i=1:numsteps,

%    First set up the cosine matrices for each member
%   For Member 1
```

```
n1p=(N2(i,:)-N1(i,:))/sqrt((N2(i,:)-N1(i,:))*(N2(i,:)-N1(i,:))');
u13=(N6(i,:)-N2(i,:))/sqrt((N6(i,:)-N2(i,:))*(N6(i,:)-N2(i,:))');
u12=cross(u13,n1p);
u12=u12/sqrt(u12*u12');
u11=cross(u12,u13);
c1=[u11;u12;u13];

%  For Member 2
u23=(N6(i,:)-N1(i,:))/sqrt((N6(i,:)-N1(i,:))*(N6(i,:)-N1(i,:))');
u22=cross(u23,n1p);
u22=u22/sqrt(u22*u22');
u21=cross(u22,u23);
c2=[u21;u22;u23];

%  For Member 3
n1p=(N1(i,:)-N3(i,:))/sqrt((N1(i,:)-N3(i,:))*(N1(i,:)-N3(i,:))');
u33=(N4(i,:)-N1(i,:))/sqrt((N4(i,:)-N1(i,:))*(N4(i,:)-N1(i,:))');
u32=cross(u33,n1p);
u32=u32/sqrt(u32*u32');
u31=cross(u32,u33);
c3=[u31;u32;u33];

%  For Member 4
u43=(N4(i,:)-N3(i,:))/sqrt((N4(i,:)-N3(i,:))*(N4(i,:)-N3(i,:))');
u42=cross(u43,n1p);
 u42=u42/sqrt(u42*u42');
u41=cross(u42,u43);
c4=[u41;u42;u43];

%  For Member 5
n1p=(N2(i,:)-N3(i,:))/sqrt((N2(i,:)-N3(i,:))*(N2(i,:)-N3(i,:))');
u53=(N5(i,:)-N3(i,:))/sqrt((N5(i,:)-N3(i,:))*(N5(i,:)-N3(i,:))');
u52=cross(u53,n1p);
 u52=u52/sqrt(u52*u52');
u51=cross(u52,u53);
c5=[u51;u52;u53];

%  For Member 6
u63=(N5(i,:)-N2(i,:))/sqrt((N5(i,:)-N2(i,:))*(N5(i,:)-N2(i,:))');
u62=cross(u63,n1p);
 u62=u62/sqrt(u62*u62');
u61=cross(u62,u63);
c6=[u61;u62;u63];
```

```
%  For Member 7
u73=(N5(i,:)-N4(i,:))/sqrt((N5(i,:)-N4(i,:))*(N5(i,:)-N4(i,:))');
junk=(N6(i,:)-N4(i,:))/sqrt((N6(i,:)-N4(i,:))*(N6(i,:)-N4(i,:))');
u71=cross(junk,u73);
 u71=u71/sqrt(u71*u71');
u72=cross(u73,u71);
c7=[u71;u72;u73];

%  For Member 8
%c8=c7;

%  For Member 9
u93=(N5(i,:)-N6(i,:))/sqrt((N5(i,:)-N6(i,:))*(N5(i,:)-N6(i,:))');
junk=(N4(i,:)-N6(i,:))/sqrt((N4(i,:)-N6(i,:))*(N4(i,:)-N6(i,:))');
u91=cross(u93,junk);
 u91=u91/sqrt(u91*u91');
u92=cross(u93,u91);
c9=[u91;u92;u93];

%  For Member 10
%c10=c9;

%  For Member 11
u113=(N6(i,:)-N4(i,:))/sqrt((N6(i,:)-N4(i,:))*(N6(i,:)-N4(i,:))');
junk=(N5(i,:)-N4(i,:))/sqrt((N5(i,:)-N4(i,:))*(N5(i,:)-N4(i,:))');
u111=cross(u113,junk);
 u111=u111/sqrt(u111*u111');
u112=cross(u113,u111);
c11=[u111;u112;u113];

%  For Member 12
%c12=c11;

%  For Member 13
n1p=(N9(i,:)-N7(i,:))/sqrt((N9(i,:)-N7(i,:))*(N9(i,:)-N7(i,:))');
u133=(N9(i,:)-N6(i,:))/sqrt((N9(i,:)-N6(i,:))*(N9(i,:)-N6(i,:))');
u131=cross(n1p,u133);
 u131=u131/sqrt(u131*u131');
u132=cross(u133,u131);
c13=[u131;u132;u133];

%  For Member 14
u143=(N7(i,:)-N6(i,:))/sqrt((N7(i,:)-N6(i,:))*(N7(i,:)-N6(i,:))');
```

```
u141=cross(n1p,u143);
  u141=u141/sqrt(u141*u141');
u142=cross(u143,u141);
c14=[u141;u142;u143];

%  For Member 15
n1p=(N7(i,:)-N8(i,:))/sqrt((N7(i,:)-N8(i,:))*(N7(i,:)-N8(i,:))');
u153=(N7(i,:)-N4(i,:))/sqrt((N7(i,:)-N4(i,:))*(N7(i,:)-N4(i,:))');
u151=cross(n1p,u153);
 u151=u151/sqrt(u151*u151');
u152=cross(u153,u151);
c15=[u151;u152;u153];

%  For Member 16
u163=(N8(i,:)-N4(i,:))/sqrt((N8(i,:)-N4(i,:))*(N8(i,:)-N4(i,:))');
u161=cross(n1p,u163);
  u161=u161/sqrt(u161*u161');
u162=cross(u163,u161);
c16=[u161;u162;u163];

%  For Member 17
n1p=(N9(i,:)-N8(i,:))/sqrt((N9(i,:)-N8(i,:))*(N9(i,:)-N8(i,:))');
u173=(N8(i,:)-N5(i,:))/sqrt((N8(i,:)-N5(i,:))*(N8(i,:)-N5(i,:))');
u171=cross(n1p,u173);
 u171=u171/sqrt(u171*u171');
u172=cross(u173,u171);
c17=[u171;u172;u173];

%  For Member 18
u183=(N9(i,:)-N5(i,:))/sqrt((N9(i,:)-N5(i,:))*(N9(i,:)-N5(i,:))');
u181=cross(n1p,u183);
  u181=u181/sqrt(u181*u181');
u182=cross(u183,u181);
c18=[u181;u182;u183];

%  For Member 19
u193=(N9(i,:)-N7(i,:))/sqrt((N9(i,:)-N7(i,:))*(N9(i,:)-N7(i,:))');
 junk=(N8(i,:)-N7(i,:))/sqrt((N8(i,:)-N7(i,:))*(N8(i,:)-N7(i,:))');
u191=cross(u193,junk);
 u191=u191/sqrt(u191*u191');
u192=cross(u193,u191);
c19=[u191;u192;u193];

%  For Member 20
```

```
u203=(N7(i,:)-N8(i,:))/sqrt((N7(i,:)-N8(i,:))*(N7(i,:)-N8(i,:))');
 junk=(N9(i,:)-N8(i,:))/sqrt((N9(i,:)-N8(i,:))*(N9(i,:)-N8(i,:))');
u201=cross(u203,junk);
 u201=u201/sqrt(u201*u201');
u202=cross(u203,u201);
c20=[u201;u202;u203];

%  For Member 21
u213=(N9(i,:)-N8(i,:))/sqrt((N9(i,:)-N8(i,:))*(N9(i,:)-N8(i,:))');
u211=cross(u203,u213);
 u211=u211/sqrt(u211*u211');
u212=cross(u213,u211);
c21=[u211;u212;u213];

%  Now all cosine matrices are defined
%
%    Start setting up force matrices

%     Set up angular velocities and angular accelerations for
%        the plane 1-2-6 (in Global Coordinates)
%
d1=N6(i,:)-N1(i,:);
d2=N2(i,:)-N1(i,:);
mat=[0 d1(3) -d1(2);-d1(3) 0 d1(1);d2(2) -d2(1) 0];
wvect=[V6(i,1)-V1(i,1);V6(i,2)-V1(i,2);V2(i,3)-V1(i,3)];
wt126=inv(mat)*wvect;
b11=wt126(2)*(wt126(1)*d1(2)-wt126(2)*d1(1))
          -wt126(3)*(wt126(3)*d1(2)-wt126(1)*d1(3));
b21=wt126(3)*(wt126(2)*d1(3)-wt126(3)*d1(2))
          -wt126(1)*(wt126(1)*d1(2)-wt126(2)*d1(1));
b32=wt126(1)*(wt126(3)*d2(1)-wt126(1)*d2(3))
          -wt126(2)*(wt126(2)*d2(3)-wt126(3)*d2(2));
acvect=[A6(i,1)-A1(i,1)-b11;A6(i,2)-A1(i,2)-b21;A2(i,3)-A1(i,3)-b32];
alp126=inv(mat)*acvect;

%
%    For Member 1
mat1=[eye(3) eye(3);-c1(2,:) c1(2,:);c1(1,:) -c1(1,:)];
fmat1=[mat1 zeros(5,111)];
a1x(i)=(A2(i,1)+A6(i,1))/2;
a1y(i)=(A2(i,2)+A6(i,2))/2;
a1z(i)=(A2(i,3)+A6(i,3))/2;
alp11(i)=c1(1,:)*alp126;
alp12(i)=c1(2,:)*alp126;
```

```
i1b1=mass1*l1^3/12;
i1b2=i1b1;
inert1=[mass1*a1x(i) mass1*a1y(i) mass1*a1z(i)+mass1*g
                2/l1*i1b1*alp11(i) 2/l1*i1b2*alp12(i)];

%    For Member 2
mat2=[eye(3) eye(3);-c2(2,:) c2(2,:);c2(1,:) -c2(1,:)];
fmat2=[zeros(5,6) mat2 zeros(5,105)];
a2x(i)=(A1(i,1)+A6(i,1))/2;
a2y(i)=(A1(i,2)+A6(i,2))/2;
a2z(i)=(A1(i,3)+A6(i,3))/2;
alp21(i)=c2(1,:)*alp126;
alp22(i)=c2(2,:)*alp126;
i2b1=mass2*l2^3/12;
i2b2=i2b1;
inert2=[mass2*a2x(i) mass2*a2y(i) mass2*a2z(i)+mass2*g
                2/l2*i2b1*alp21(i) 2/l2*i2b2*alp22(i)];

%        General Angular velocity and accelration calculation for
%        the plane 3-1-4 (in Global Coordinates)
%
d1=N4(i,:)-N3(i,:);
d2=N1(i,:)-N3(i,:);
mat=[0 d1(3) -d1(2);-d1(3) 0 d1(1);d2(2) -d2(1) 0];
wvect=[V4(i,1)-V3(i,1);V4(i,2)-V3(i,2);V1(i,3)-V3(i,3)];
wt314=inv(mat)*wvect;
b11=wt314(2)*(wt314(1)*d1(2)-wt314(2)*d1(1))
        -wt314(3)*(wt314(3)*d1(2)-wt314(1)*d1(3));
b21=wt314(3)*(wt314(2)*d1(3)-wt314(3)*d1(2))
        -wt314(1)*(wt314(1)*d1(2)-wt314(2)*d1(1));
b32=wt314(1)*(wt314(3)*d2(1)-wt314(1)*d2(3))
        -wt314(2)*(wt314(2)*d2(3)-wt314(3)*d2(2));
acvect=[A4(i,1)-A3(i,1)-b11;A4(i,2)-A3(i,2)-b21;A1(i,3)-A3(i,3)-b32];
alp314 =inv(mat)*acvect;

%   For Member 3
mat3=[eye(3) eye(3);-c3(2,:) c3(2,:);c3(1,:) -c3(1,:)];
fmat3=[zeros(5,12) mat3 zeros(5,99)];
a3x(i)=(A1(i,1)+A4(i,1))/2;
a3y(i)=(A1(i,2)+A4(i,2))/2;
a3z(i)=(A1(i,3)+A4(i,3))/2;
alp31(i)=c3(1,:)*alp314;
alp32(i)=c3(2,:)*alp314;
i3b1=mass3*l3^3/12;
```

```
i3b2=i3b1;
inert3=[mass3*a3x(i) mass3*a3y(i) mass3*a3z(i)+mass3*g
                2/l3*i3b1*alp31(i) 2/l3*i3b2*alp32(i)];

%   For Member 4
mat4=[eye(3) eye(3);-c4(2,:) c4(2,:);c4(1,:) -c4(1,:)];
fmat4=[zeros(5,18) mat4 zeros(5,93)];
a4x(i)=(A3(i,1)+A4(i,1))/2;
a4y(i)=(A3(i,2)+A4(i,2))/2;
a4z(i)=(A3(i,3)+A4(i,3))/2;
alp41(i)=c4(1,:)*alp314;
alp42(i)=c4(2,:)*alp314;
i4b1=mass4*l4^3/12;
i4b2=i4b1;
inert4=[mass4*a4x(i) mass4*a4y(i) mass4*a4z(i)+mass4*g
                2/l4*i4b1*alp41(i) 2/l4*i4b2*alp42(i)];

%       General Angular velocity and accelration calculation for
%       the plane 2-3-5 (in Global Coordinates)
%
d1=N5(i,:)-N2(i,:);
d2=N3(i,:)-N2(i,:);
mat=[0 d1(3) -d1(2);-d1(3) 0 d1(1);d2(2) -d2(1) 0];
wvect=[V5(i,1)-V2(i,1);V5(i,2)-V2(i,2);V3(i,3)-V2(i,3)];
wt235=inv(mat)*wvect;
b11=wt235(2)*(wt235(1)*d1(2)-wt235(2)*d1(1))
      -wt235(3)*(wt235(3)*d1(2)-wt235(1)*d1(3));
b21=wt235(3)*(wt235(2)*d1(3)-wt235(3)*d1(2))
      -wt235(1)*(wt235(1)*d1(2)-wt235(2)*d1(1));
b32=wt235(1)*(wt235(3)*d2(1)-wt235(1)*d2(3))
      -wt235(2)*(wt235(2)*d2(3)-wt235(3)*d2(2));
acvect=[A5(i,1)-A2(i,1)-b11;A5(i,2)-A2(i,2)-b21;A3(i,3)-A2(i,3)-b32];
alp235=inv(mat)*acvect;

%   For Member 5
mat5=[eye(3) eye(3);-c5(2,:) c5(2,:);c5(1,:) -c5(1,:)];
fmat5=[zeros(5,24) mat5 zeros(5,87)];
a5x(i)=(A3(i,1)+A5(i,1))/2;
a5y(i)=(A3(i,2)+A5(i,2))/2;
a5z(i)=(A3(i,3)+A5(i,3))/2;
alp51(i)=c5(1,:)*alp235;
alp52(i)=c5(2,:)*alp235;
i5b1=mass5*l5^3/12;
i5b2=i5b1;
```

```
inert5=[mass5*a5x(i) mass5*a5y(i) mass5*a5z(i)+mass5*g
                2/l5*i5b1*alp51(i) 2/l5*i5b2*alp52(i)];

%   For Member 6
mat6=[eye(3) eye(3);-c6(2,:) c6(2,:);c6(1,:) -c6(1,:)];
fmat6=[zeros(5,30) mat6 zeros(5,81)];
a6x(i)=(A2(i,1)+A6(i,1))/2;
a6y(i)=(A2(i,2)+A6(i,2))/2;
a6z(i)=(A2(i,3)+A6(i,3))/2;
alp61(i)=c6(1,:)*alp235;
alp62(i)=c6(2,:)*alp235;
i6b1=mass6*l6^3/12;
i6b2=i6b1;
inert6=[mass6*a6x(i) mass6*a6y(i) mass6*a6z(i)+mass6*g
               2/l6*i6b1*alp61(i) 2/l6*i6b2*alp62(i)];

%   For Member 7
l7(i)=sqrt((N5(i,:)-N4(i,:))*(N5(i,:)-N4(i,:))');
mass7=massls+massbase;
cg7(i)=(massls*(l7(i)-ls/2)+massbase*lbase/2)/(massls+massbase);
d7(i)=l7(i)-cg7(i);
i7b1=iglsb1+ibaseb1+massls*(l7(i)-ls/2-cg7(i))^2
                    +massbase*(cg7(i)-lbase/2)^2;
i7b2=i7b1;
mat7=[eye(3) eye(3);cg7(i).*c7(2,:) -d7(i).*c7(2,:);
                    -cg7(i).*c7(1,:) d7(i).*c7(1,:)];
fmat7=[zeros(5,36) mat7 zeros(5,75)];
V4b=c7*V4(i,:)';
V5b=c7*V5(i,:)';
A4b=c7*A4(i,:)';
A5b=c7*A5(i,:)';
xdot7(i)=V5b(3)-V4b(3);
xddot7(i)=A5b(3)-A4b(3);
wt71=(V4b(2)-V5b(2))/l7(i);
wt72=(V5b(1)-V4b(1))/l7(i);
alp71(i)=(A4b(2)-A5b(2)-2*wt71*xdot7(i))/l7(i);
alp72(i)=(A5b(1)-A4b(1)-2*wt72*xdot7(i))/l7(i);
a7x(i)=A4(i,1)+cg7(i)*(A5(i,1)-A4(i,1))/l7(i);
a7y(i)=A4(i,2)+cg7(i)*(A5(i,2)-A4(i,2))/l7(i);
a7z(i)=A4(i,3)+cg7(i)*(A5(i,3)-A4(i,3))/l7(i);
inert7=[mass7*a7x(i) mass7*a7y(i) mass7*a7z(i)+mass7*g
                          i7b1*alp71(i) i7b2*alp72(i)];

%   For Member 9
```

```
l9(i)=sqrt((N6(i,:)-N5(i,:))*(N6(i,:)-N5(i,:))');
mass9=massls+massbase;
cg9=(massls*(l9(i)-ls/2)+massbase*lbase/2)/(massls+massbase);
d9=l9(i)-cg9;
i9b1=iglsb1+ibaseb1+massls*(l9(i)-ls/2-cg9)^2
                    +massbase*(cg9-lbase/2)^2;
i9b2=i7b1;
mat9=[eye(3) eye(3);cg9.*c9(2,:) -d9.*c9(2,:);
                    -cg9.*c9(1,:) d9.*c9(1,:)];
fmat9=[zeros(5,42) mat9 zeros(5,69)];
V5b=c9*V5(i,:)';
V6b=c9*V6(i,:)';
A5b=c9*A5(i,:)';
A6b=c9*A6(i,:)';
xdot9(i)=V5b(3)-V6b(3);
xddot9(i)=A5b(3)-A6b(3);
wt91=(V5b(2)-V6b(2))/l9(i);
wt92=(V6b(1)-V5b(1))/l9(i);
alp91(i)=(A5b(2)-A6b(2)-2*wt91*xdot9(i))/l9(i);
alp92(i)=(A6b(1)-A5b(1)-2*wt92*xdot9(i))/l9(i);
a9x(i)=A5(i,1)+cg9*(A6(i,1)-A5(i,1))/l9(i);
a9y(i)=A5(i,2)+cg9*(A6(i,2)-A5(i,2))/l9(i);
a9z(i)=A5(i,3)+cg9*(A6(i,3)-A5(i,3))/l9(i);
inert9=[mass9*a9x(i) mass9*a9y(i) mass9*a9z(i)+mass9*g
                        i9b1*alp91(i) i9b2*alp92(i)];

%   For Member 11
l11(i)=sqrt((N4(i,:)-N6(i,:))*(N4(i,:)-N6(i,:))');
mass11=massls+massbase;
cg11=(massls*(l11(i)-ls/2)+massbase*lbase/2)/(massls+massbase);
d11=l11(i)-cg11;
i11b1=iglsb1+ibaseb1+massls*(l11(i)-ls/2-cg11)^2
                       +massbase*(cg11-lbase/2)^2;
i11b2=i11b1;
mat11=[eye(3) eye(3);cg11.*c11(2,:) -d11.*c11(2,:);
                    -cg11.*c11(1,:) d11.*c11(1,:)];
fmat11=[zeros(5,48) mat11 zeros(5,63)];
V4b=c11*V4(i,:)';
V6b=c11*V6(i,:)';
A4b=c11*A4(i,:)';
A6b=c11*A6(i,:)';
xdot11(i)=V6b(3)-V4b(3);
xddot11(i)=A6b(3)-A4b(3);
wt111=(V6b(2)-V4b(2))/l11(i);
```

```
wt112=(V4b(1)-V6b(1))/l11(i);
alp111(i)=(A6b(2)-A4b(2)-2*wt111*xdot11(i))/l11(i);
alp112(i)=(A4b(1)-A6b(1)-2*wt112*xdot11(i))/l11(i);
a11x(i)=A4(i,1)+cg11*(A6(i,1)-A4(i,1))/l11(i);
a11y(i)=A4(i,1)+cg11*(A6(i,2)-A4(i,2))/l11(i);
a11z(i)=A4(i,1)+cg11*(A6(i,3)-A4(i,3))/l11(i);
inert11=[mass11*a11x(i) mass11*a11y(i) mass11*a11z(i)+mass11*g
                              i11b1*alp111(i) i11b2*alp112(i)];

%       General Angular velocity and accelration calculation for
%       the plane 6-7-9 (in Global Coordinates)
%
d1=N6(i,:)-N7(i,:);
d2=N9(i,:)-N7(i,:);
mat=[0 d1(3) -d1(2);-d1(3) 0 d1(1);d2(2) -d2(1) 0];
wvect=[V6(i,1)-V7(i,1);V6(i,2)-V7(i,2);V9(i,3)-V7(i,3)];
wt679=inv(mat)*wvect;
b11=wt679(2)*(wt679(1)*d1(2)-wt679(2)*d1(1))
        -wt679(3)*(wt679(3)*d1(2)-wt679(1)*d1(3));
b21=wt679(3)*(wt679(2)*d1(3)-wt679(3)*d1(2))
        -wt679(1)*(wt679(1)*d1(2)-wt679(2)*d1(1));
b32=wt679(1)*(wt679(3)*d2(1)-wt679(1)*d2(3))
        -wt679(2)*(wt679(2)*d2(3)-wt679(3)*d2(2));
acvect=[A6(i,1)-A7(i,1)-b11;A6(i,2)-A7(i,2)-b21;A9(i,3)-A7(i,3)-b32];
alp679=inv(mat)*acvect;

%   For Member 13
mat13=[eye(3) eye(3);c13(2,:) -c13(2,:);-c13(1,:) c13(1,:)];
fmat13=[zeros(5,54) mat13 zeros(5,57)];
a13x(i)=(A6(i,1)+A9(i,1))/2;
a13y(i)=(A6(i,2)+A9(i,2))/2;
a13z(i)=(A6(i,3)+A9(i,3))/2;
alp131(i)=c13(1,:)*alp679;
alp132(i)=c13(2,:)*alp679;
i13b1=mass13*l13^3/12;
i13b2=i13b1;
inert13=[mass13*a13x(i) mass13*a13y(i) mass13*a13z(i)+mass13*g
                  2/l13*i13b1*alp131(i) 2/l13*i13b2*alp132(i)];

%   For Member 14
mat14=[eye(3) eye(3);c14(2,:) -c14(2,:);-c14(1,:) c14(1,:)];
fmat14=[zeros(5,60) mat14 zeros(5,51)];
a14x(i)=(A6(i,1)+A7(i,1))/2;
a14y(i)=(A6(i,2)+A7(i,2))/2;
```

```
a14z(i)=(A6(i,3)+A7(i,3))/2;
alp141(i)=c14(1,:)*alp679;
alp142(i)=c14(2,:)*alp679;
i14b1=mass14*l14^3/12;
i14b2=i14b1;
inert14=[mass14*a14x(i) mass14*a14y(i) mass14*a14z(i)+mass14*g
                  2/l14*i14b1*alp141(i) 2/l14*i14b2*alp142(i)];

%       General Angular velocity and accelration calculation for
%       the plane 4-8-7 (in Global Coordinates)
%
d1=N4(i,:)-N8(i,:);
d2=N7(i,:)-N8(i,:);
mat=[0 d1(3) -d1(2);-d1(3) 0 d1(1);d2(2) -d2(1) 0];
wvect=[V4(i,1)-V8(i,1);V4(i,2)-V8(i,2);V7(i,3)-V8(i,3)];
wt478=inv(mat)*wvect;
b11=wt478(2)*(wt478(1)*d1(2)-wt478(2)*d1(1))
        -wt478(3)*(wt478(3)*d1(2)-wt478(1)*d1(3));
b21=wt478(3)*(wt478(2)*d1(3)-wt478(3)*d1(2))
        -wt478(1)*(wt478(1)*d1(2)-wt478(2)*d1(1));
b32=wt478(1)*(wt478(3)*d2(1)-wt478(1)*d2(3))
        -wt478(2)*(wt478(2)*d2(3)-wt478(3)*d2(2));
acvect=[A4(i,1)-A8(i,1)-b11;A4(i,2)-A8(i,2)-b21;A7(i,3)-A8(i,3)-b32];
alp478=inv(mat)*acvect;

%   For Member 15
mat15=[eye(3) eye(3);c15(2,:) -c15(2,:);-c15(1,:) c15(1,:)];
fmat15=[zeros(5,66) mat15 zeros(5,45)];
a15x(i)=(A4(i,1)+A7(i,1))/2;
a15y(i)=(A4(i,2)+A7(i,2))/2;
a15z(i)=(A4(i,3)+A7(i,3))/2;
alp151(i)=c15(1,:)*alp478;
alp152(i)=c15(2,:)*alp478;
i15b1=mass15*l15^3/12;
i15b2=i15b1;
inert15=[mass15*a15x(i) mass15*a15y(i) mass15*a15z(i)+mass15*g
                  2/l15*i15b1*alp151(i) 2/l15*i15b2*alp152(i)];

%   For Member 16
mat16=[eye(3) eye(3);c16(2,:) -c16(2,:);-c16(1,:) c16(1,:)];
fmat16=[zeros(5,72) mat16 zeros(5,39)];
a16x(i)=(A4(i,1)+A8(i,1))/2;
a16y(i)=(A4(i,2)+A8(i,2))/2;
a16z(i)=(A4(i,3)+A8(i,3))/2;
```

```
alp161(i)=c16(1,:)*alp478;
alp162(i)=c16(2,:)*alp478;
inert16=[mass16*a16x(i) mass16*a16y(i) mass16*a16z(i)+mass16*g
                  2/l16*i16b1*alp161(i) 2/l16*i16b2*alp162(i)];

%        General Angular velocity and accelration calculation for
%        the plane 5-8-9 (in Global Coordinates)
%
d1=N5(i,:)-N9(i,:);
d2=N8(i,:)-N9(i,:);
mat=[0 d1(3) -d1(2);-d1(3) 0 d1(1);d2(2) -d2(1) 0];
wvect=[V5(i,1)-V9(i,1);V5(i,2)-V9(i,2);V8(i,3)-V9(i,3)];
wt589=inv(mat)*wvect;
b11=wt589(2)*(wt589(1)*d1(2)-wt589(2)*d1(1))
         -wt589(3)*(wt589(3)*d1(2)-wt589(1)*d1(3));
b21=wt589(3)*(wt589(2)*d1(3)-wt589(3)*d1(2))
          -wt589(1)*(wt589(1)*d1(2)-wt589(2)*d1(1));
b32=wt589(1)*(wt589(3)*d2(1)-wt589(1)*d2(3))
          -wt589(2)*(wt589(2)*d2(3)-wt589(3)*d2(2));
acvect=[A5(i,1)-A9(i,1)-b11;A5(i,2)-A9(i,2)-b21;A8(i,3)-A9(i,3)-b32];
alp589=inv(mat)*acvect;

%   For Member 17
mat17=[eye(3) eye(3);c17(2,:) -c17(2,:);-c17(1,:) c17(1,:)];
fmat17=[zeros(5,78) mat17 zeros(5,33)];
a17x(i)=(A5(i,1)+A8(i,1))/2;
a17y(i)=(A5(i,2)+A8(i,2))/2;
a17z(i)=(A5(i,3)+A8(i,3))/2;
alp171(i)=c17(1,:)*alp589;
alp172(i)=c17(2,:)*alp589;
i17b1=mass17*l17^3/12;
i17b2=i17b1;
inert17=[mass17*a17x(i) mass17*a17y(i) mass17*a17z(i)+mass17*g
                  2/l17*i17b1*alp171(i) 2/l17*i17b2*alp172(i)];

%   For Member 18
mat18=[eye(3) eye(3);c18(2,:) -c18(2,:);-c18(1,:) c18(1,:)];
fmat18=[zeros(5,84) mat18 zeros(5,27)];
a18x(i)=(A5(i,1)+A9(i,1))/2;
a18y(i)=(A5(i,2)+A9(i,2))/2;
a18z(i)=(A5(i,3)+A9(i,3))/2;
alp181(i)=c18(1,:)*alp589;
alp182(i)=c18(2,:)*alp589;
i18b1=mass18*l18^3/12;
```

```
i18b2=i18b1;
inert18=[mass18*a18x(i) mass18*a18y(i) mass18*a18z(i)+mass18*g
                  2/l18*i18b1*alp181(i) 2/l18*i18b2*alp182(i)];


%     Set up angular velocities and angular accelerations for
%        the top plane (Plane 7-8-9)
%
d1=N8(i,:)-N9(i,:);
d2=N8(i,:)-N7(i,:);
topmat1=[0 d1(3) -d1(2);-d1(3) 0 d1(1);-d2(3) 0 d2(1)];
wvect1=[V8(i,1)-V9(i,1);V8(i,2)-V9(i,2);V8(i,2)-V7(i,2)];
wt789(:,i)=inv(topmat1)*wvect1;
b11=wt789(3,i)*(wt789(3,i)*d1(1)-wt789(1,i)*d1(3))
        -wt789(2,i)*(wt789(1,i)*d1(2)-wt789(2,i)*d1(1));
b21=wt789(1,i)*(wt789(1,i)*d1(2)-wt789(2,i)*d1(1))
        -wt789(3,i)*(wt789(2,i)*d1(3)-wt789(3,i)*d1(2));
b22=wt789(1,i)*(wt789(1,i)*d2(2)-wt789(2,i)*d2(1))
        -wt789(3,i)*(wt789(2,i)*d2(3)-wt789(3,i)*d2(2));
acvect=[A8(i,1)-A9(i,1)+b11;A8(i,2)-A9(i,2)+b21;A8(i,2)-A7(i,2)+b22];
alp789(:,i)=inv(topmat1)*acvect;


%   For Member 19
mat19=[eye(3) eye(3);c19(2,:) -c19(2,:);-c19(1,:) c19(1,:)];
fmat19=[zeros(5,90) mat19 zeros(5,21)];
a19x(i)=(A7(i,1)+A9(i,1))/2;
a19y(i)=(A7(i,2)+A9(i,2))/2;
a19z(i)=(A7(i,3)+A9(i,3))/2;
alp191(i)=c19(1,:)*alp789(:,i);
alp192(i)=c19(2,:)*alp789(:,i);
i19b1=mass19*l19^3/12;
i19b2=i19b1;
inert19=[mass19*a19x(i) mass19*a19y(i) mass19*a19z(i)+mass19*g
                  2/l19*i19b1*alp191(i) 2/l19*i19b2*alp192(i)];

%   For Member 20
mat20=[eye(3) eye(3);c20(2,:) -c20(2,:);-c20(1,:) c20(1,:)];
fmat20=[zeros(5,96) mat20 zeros(5,15)];
a20x(i)=(A7(i,1)+A8(i,1))/2;
a20y(i)=(A7(i,2)+A8(i,2))/2;
a20z(i)=(A7(i,3)+A8(i,3))/2;
alp201(i)=c20(1,:)*alp789(:,i);
alp202(i)=c20(2,:)*alp789(:,i);
```

```
i20b1=mass20*l20^3/12;
i20b2=i20b1;
inert20=[mass20*a20x(i) mass20*a20y(i) mass20*a20z(i)+mass20*g
                  2/l20*i20b1*alp201(i) 2/l20*i20b2*alp202(i)];

%   For Member 21
mat21=[eye(3) eye(3);c21(2,:) -c21(2,:);-c21(1,:) c21(1,:)];
fmat21=[zeros(5,102) mat21 zeros(5,9)];
a21x(i)=(A8(i,1)+A9(i,1))/2;
a21y(i)=(A8(i,2)+A9(i,2))/2;
a21z(i)=(A8(i,3)+A9(i,3))/2;
alp211(i)=c21(1,:)*alp789(:,i);
alp212(i)=c21(2,:)*alp789(:,i);
i21b1=mass21*l21^3/12;
i21b2=i21b1;
inert21=[mass21*a21x(i) mass21*a21y(i) mass21*a21z(i)+mass21*g
                 2/l21*i21b1*alp211(i) 2/l21*i21b2*alp212(i)];

%   Now Program in Constraint Equations
%   For Node 1
nmat1=[zeros(3,9) eye(3) zeros(3,3) eye(3) zeros(3,90) -eye(3) ...
          zeros(3,6)];

%   For Node 2
nmat2=[zeros(3,3) eye(3) zeros(3,27) eye(3) zeros(3,75) -eye(3) ...
          zeros(3,3)];

%   For Node 3
nmat3=[zeros(3,21) eye(3) zeros(3,3) eye(3) zeros(3,84) -eye(3)];

%   For Node 4
nmat4=[zeros(3,12) eye(3) zeros(3,3) eye(3) zeros(3,15) eye(3)...
       zeros(3,9) eye(3) zeros(3,15) eye(3) zeros(3,3) eye(3)...
        zeros(3,42)];

%   For Node 5
nmat5=[zeros(3,24) eye(3) zeros(3,3) eye(3) zeros(3,6) eye(3)...
       eye(3) zeros(3,33) eye(3) zeros(3,3) eye(3) zeros(3,30)];

%   For Node 6
nmat6=[eye(3) zeros(3,3) eye(3) zeros(3,36) eye(3) zeros(3,3)...
      eye(3) eye(3) zeros(3,3) eye(3) zeros(3,54)];

%   For Node 7
```

```
nmat7=[zeros(3,63) eye(3) zeros(3,3) eye(3) zeros(3,18) eye(3)...
       zeros(3,6) eye(3) zeros(3,15)];

%   For Node 8
nmat8=[zeros(3,75) eye(3) zeros(3,3) eye(3) zeros(3,12) eye(3)...
       zeros(3,3) eye(3) zeros(3,12)];

%   For Node 9
nmat9=[zeros(3,57) eye(3) zeros(3,27) eye(3) zeros(3,3) eye(3)...
        zeros(3,9) eye(3)  zeros(3,9)];

%   For the constraint equations: all resultant forces should
%     be zero unless external forces are applied or lumped
%     masses are placed at the nodes
coninert=[zeros(1,18) lmass7*A7(i,1)+NN7x lmass7*A7(i,2)+NN7y...
 lmass7*(A7(i,3)-g1)+NN7z lmass8*A8(i,1)+NN8x lmass8*A8(i,2)+NN8y...
 lmass8*(A8(i,3)-g1)+NN8z lmass9*A9(i,1)+NN9x lmass9*A9(i,2)+NN9y...
 lmass9*(A9(i,3)-g1)+NN9z];

%    Now we have all the components of the overall Force Matrix
%      Combine the elements to form the matrix

fmatrix=[fmat1;fmat2;fmat3;fmat4;fmat5;fmat6;fmat7;fmat9;fmat11;...
         fmat13;fmat14;fmat15;fmat16;fmat17;fmat18;fmat19;fmat20;...
         fmat21;nmat1;nmat2;nmat3;nmat4;nmat5;nmat6;nmat7;nmat8;nmat9];

%   Set up the inertial force matrix

finert=[inert1 inert2 inert3 inert4 inert5 inert6 inert7 inert9 ...
      inert11 inert13 inert14 inert15 inert16 inert17 inert18 ...
      inert19 inert20 inert21 coninert]';

%   Find the resulting forces by:   Forces = inv(fmatrix)*finert

forces(:,i)=inv(fmatrix)*finert;

%   Now Find internal forces on members 1 - 6
%    Only look at midpoint for our prototype truss with
%       strain gages already attached

%    For Dynamic Internal Forces


%   For Member 1
```

```
azx=A2(i,1)+3*(A6(i,1)-A2(i,1))/4;
azy=A2(i,2)+3*(A6(i,2)-A2(i,2))/4;
azz=A2(i,3)+3*(A6(i,3)-A2(i,3))/4;
azn=[azx;azy;azz];
azb1=c1(1,:)*azn;
azb2=c1(2,:)*azn;
azb3=c1(3,:)*azn;
ww=mass1*g/2;
f1=c1(1,:)*forces(1:3,i);
f2=c1(2,:)*forces(1:3,i);
f3=c1(3,:)*forces(1:3,i);
ibz=i1b1/16;
intmat=[-1 0 0 0 0;0 -1 0 0 0;0 0 -1 0 0;0 -l1/4 0 1 0;...
                                                l1/4 0 0 0 -1];
intfv=[mass1/2*azb1-f1+c1(1,3)*ww;mass1/2*azb2-f2+c1(2,3)*ww;...
       mass1/2*azb3-f3+c1(3,3)*ww;ibz*alp11(i)+l1/4*f2;...
                                              ibz*alp12(i)-l1/4*f1];
intfor=inv(intmat)*intfv;
shear11(i)=intfor(1);
shear21(i)=intfor(2);
tension1(i)=intfor(3);
moment11(i)=intfor(4);
moment21(i)=intfor(5);
tstr1(i)=tension1(i)/areaf;
mstr11(i)=moment11(i)*cf/ina;
mstr21(i)=moment21(i)*cf/ina;
vstr11(i)=2*shear11(i)/areaf;
vstr21(i)=2*shear21(i)/areaf;
vmst1a(i)=sqrt((abs(mstr11(i))+abs(tstr1(i)))^2+3*vstr11(i)^2);
vmst1b(i)=sqrt((abs(mstr21(i))+abs(tstr1(i)))^2+3*vstr21(i)^2);


%   For Member 2

azx=A1(i,1)+3*(A6(i,1)-A1(i,1))/4;
azy=A1(i,2)+3*(A6(i,2)-A1(i,2))/4;
azz=A1(i,3)+3*(A6(i,3)-A1(i,3))/4;
azn=[azx;azy;azz];
azb1=c2(1,:)*azn;
azb2=c2(2,:)*azn;
azb3=c2(3,:)*azn;
ww=mass2*g/2;
f1=c2(1,:)*forces(7:9,i);
```

```
f2=c2(2,:)*forces(7:9,i);
f3=c2(3,:)*forces(7:9,i);
ibz=i2b1/16;
intmat=[-1 0 0 0 0;0 -1 0 0 0;0 0 -1 0 0;0 -l2/4 0 1 0;...
                                          l2/4 0 0 0 -1];
intfv=[mass2/2*azb1-f1+c2(1,3)*ww;mass2/2*azb2-f2+c2(2,3)*ww;...
       mass2/2*azb3-f3+c2(3,3)*ww;ibz*alp21(i)+l2/4*f2;...
                                      ibz*alp22(i)-l2/4*f1];
intfor=inv(intmat)*intfv;
shear12(i)=intfor(1);
shear22(i)=intfor(2);
tension2(i)=intfor(3);
moment12(i)=intfor(4);
moment22(i)=intfor(5);
tstr2(i)=tension2(i)/areaf;
mstr12(i)=moment12(i)*cf/ina;
mstr22(i)=moment22(i)*cf/ina;
vstr12(i)=2*shear12(i)/areaf;
vstr22(i)=2*shear22(i)/areaf;
vmst2a(i)=sqrt((abs(mstr12(i))+abs(tstr2(i)))^2+3*vstr12(i)^2);
vmst2b(i)=sqrt((abs(mstr22(i))+abs(tstr2(i)))^2+3*vstr22(i)^2);

%   For Member 3

azx=A1(i,1)+3*(A4(i,1)-A1(i,1))/4;
azy=A1(i,2)+3*(A4(i,2)-A1(i,2))/4;
azz=A1(i,3)+3*(A4(i,3)-A1(i,3))/4;
azn=[azx;azy;azz];
azb1=c3(1,:)*azn;
azb2=c3(2,:)*azn;
azb3=c3(3,:)*azn;
ww=mass3*g/2;
f1=c3(1,:)*forces(13:15,i);
f2=c3(2,:)*forces(13:15,i);
f3=c3(3,:)*forces(13:15,i);
ibz=i3b1/16;
intmat=[-1 0 0 0 0;0 -1 0 0 0;0 0 -1 0 0;0 -l3/4 0 1 0;...
                                          l3/4 0 0 0 -1];
intfv=[mass3/2*azb1-f1+c3(1,3)*ww;mass3/2*azb2-f2+c3(2,3)*ww;...
       mass3/2*azb3-f3+c3(3,3)*ww;ibz*alp31(i)+l3/4*f2;...
                                      ibz*alp32(i)-l3/4*f1];
intfor=inv(intmat)*intfv;
shear13(i)=intfor(1);
shear23(i)=intfor(2);
```

```
tension3(i)=intfor(3);
moment13(i)=intfor(4);
moment23(i)=intfor(5);
tstr3(i)=tension3(i)/areaf;
mstr13(i)=moment13(i)*cf/ina;
mstr23(i)=moment23(i)*cf/ina;
vstr13(i)=2*shear13(i)/areaf;
vstr23(i)=2*shear23(i)/areaf;
vmst3a(i)=sqrt((abs(mstr13(i))+abs(tstr3(i)))^2+3*vstr13(i)^2);
vmst3b(i)=sqrt((abs(mstr23(i))+abs(tstr3(i)))^2+3*vstr23(i)^2);

%   For Member 4

azx=A3(i,1)+3*(A4(i,1)-A3(i,1))/4;
azy=A3(i,2)+3*(A4(i,2)-A3(i,2))/4;
azz=A3(i,3)+3*(A4(i,3)-A3(i,3))/4;
azn=[azx;azy;azz];
azb1=c4(1,:)*azn;
azb2=c4(2,:)*azn;
azb3=c4(3,:)*azn;
ww=mass4*g/2;
f1=c4(1,:)*forces(19:21,i);
f2=c4(2,:)*forces(19:21,i);
f3=c4(3,:)*forces(19:21,i);
ibz=i4b1/16;
intmat=[-1 0 0 0 0;0 -1 0 0 0;0 0 -1 0 0;0 -l4/4 0 1 0;...
                                          l4/4 0 0 0 -1];
intfv=[mass4/2*azb1-f1+c4(1,3)*ww;mass4/2*azb2-f2+c4(2,3)*ww;...
       mass4/2*azb3-f3+c4(3,3)*ww;ibz*alp41(i)+l4/4*f2;...
                                       ibz*alp42(i)-l4/4*f1];
intfor=inv(intmat)*intfv;
shear14(i)=intfor(1);
shear24(i)=intfor(2);
tension4(i)=intfor(3);
moment14(i)=intfor(4);
moment24(i)=intfor(5);
tstr4(i)=tension4(i)/areaf;
mstr14(i)=moment14(i)*cf/ina;
mstr24(i)=moment24(i)*cf/ina;
vstr14(i)=2*shear14(i)/areaf;
vstr24(i)=2*shear24(i)/areaf;
vmst4a(i)=sqrt((abs(mstr14(i))+abs(tstr4(i)))^2+3*vstr14(i)^2);
vmst4b(i)=sqrt((abs(mstr24(i))+abs(tstr4(i)))^2+3*vstr24(i)^2);
```

```
%   For Member 5

azx=A3(i,1)+3*(A5(i,1)-A3(i,1))/4;
azy=A3(i,2)+3*(A5(i,2)-A3(i,2))/4;
azz=A3(i,3)+3*(A5(i,3)-A3(i,3))/4;
azn=[azx;azy;azz];
azb1=c5(1,:)*azn;
azb2=c5(2,:)*azn;
azb3=c5(3,:)*azn;
ww=mass5*g/2;
f1=c5(1,:)*forces(25:27,i);
f2=c5(2,:)*forces(25:27,i);
f3=c5(3,:)*forces(25:27,i);
ibz=i5b1/16;
intmat=[-1 0 0 0 0;0 -1 0 0 0;0 0 -1 0 0;0 -l5/4 0 1 0;...
                                          l5/4 0 0 0 -1];
intfv=[mass5/2*azb1-f1+c5(1,3)*ww;mass5/2*azb2-f2+c5(2,3)*ww;...
       mass5/2*azb3-f3+c5(3,3)*ww;ibz*alp51(i)+l5/4*f2;...
                                        ibz*alp52(i)-l5/4*f1];
intfor=inv(intmat)*intfv;
shear15(i)=intfor(1);
shear25(i)=intfor(2);
tension5(i)=intfor(3);
moment15(i)=intfor(4);
moment25(i)=intfor(5);
tstr5(i)=tension5(i)/areaf;
mstr15(i)=moment15(i)*cf/ina;
mstr25(i)=moment25(i)*cf/ina;
vstr15(i)=2*shear15(i)/areaf;
vstr25(i)=2*shear25(i)/areaf;
vmst5a(i)=sqrt((abs(mstr15(i))+abs(tstr5(i)))^2+3*vstr15(i)^2);
vmst5b(i)=sqrt((abs(mstr25(i))+abs(tstr5(i)))^2+3*vstr25(i)^2);

%   For Member 6

azx=A2(i,1)+3*(A5(i,1)-A2(i,1))/4;
azy=A2(i,2)+3*(A5(i,2)-A2(i,2))/4;
azz=A2(i,3)+3*(A5(i,3)-A2(i,3))/4;
azn=[azx;azy;azz];
azb1=c6(1,:)*azn;
azb2=c6(2,:)*azn;
azb3=c6(3,:)*azn;
ww=mass6*g/2;
f1=c6(1,:)*forces(31:33,i);
```

```
f2=c6(2,:)*forces(31:33,i);
f3=c6(3,:)*forces(31:33,i);
ibz=i6b1/16;
intmat=[-1 0 0 0 0;0 -1 0 0 0;0 0 -1 0 0;0 -l6/4 0 1 0;...
                                              l6/4 0 0 0 -1];
intfv=[(mass6/2)*azb1-f1+(c6(1,3)*ww);mass6/2*azb2-f2+(c6(2,3)*ww);...
       mass6/2*azb3-f3+c6(3,3)*ww;ibz*alp61(i)+l6/4*f2;...
                                     ibz*alp62(i)-l6/4*f1];
intfor=inv(intmat)*intfv;
shear16(i)=intfor(1);
shear26(i)=intfor(2);
tension6(i)=intfor(3);
moment16(i)=intfor(4);
moment26(i)=intfor(5);
tstr6(i)=tension6(i)/areaf;
mstr16(i)=moment16(i)*cf/ina;
mstr26(i)=moment26(i)*cf/ina;
vstr16(i)=2*shear16(i)/areaf;
vstr26(i)=2*shear26(i)/areaf;
vmst6a(i)=sqrt((abs(mstr16(i))+abs(tstr6(i)))^2+3*vstr16(i)^2);
vmst6b(i)=sqrt((abs(mstr26(i))+abs(tstr6(i)))^2+3*vstr26(i)^2);

%   To find cantilever forces for Member 8

A4b=c7*A4(i,:)';
acg8b1=A4b(1)+alp72(i)*ls/2;
acg8b2=A4b(2)-alp71(i)*ls/2;
acg8b3=A4b(3)-(wt71^2+wt72^2)*ls/2;
acg8n=c7'*[acg8b1;acg8b2;acg8b3];
ww=massls*g;
f1=forces(37,i);
f2=forces(38,i);
f3=forces(39,i);
intmat=[1 0 0 0 0;0 1 0 0 0;0 0 1 0 0;ls/2.*c7(2,:) 1 0;...
                                        -l2/2*c7(1,:) 0 -1];
intfv=[massls*acg8n(1)-f1;massls*acg8n(2)-f2;massls*acg8n(3)-f3+ww;...
      iglsb1*alp71(i)+ls/2*c7(2,:)*[f1;f2;f3];...
      iglsb2*alp71(i)-ls/2*c7(1,:)*[f1;f2;f3]];
lsfor=inv(intmat)*intfv;
f8n1(i)=lsfor(1);          % n1 direction
                                force at leadscrew cantilever
f8n2(i)=lsfor(2);          % n2 direction
                                force at leadscrew cantilever
f8n3(i)=lsfor(3);          % n3 direction
```

```
                                    force at leadscrew cantilever
m81(i)=lsfor(4);            % moment about
                                    b1 axis at leadscrew cantilever
m82(i)=lsfor(5);            % moment about
                                    b2 axis at cantilever connection
afor8(:,i)=c7*[f8n1(i);f8n2(i);f8n3(i)];  % forces applied to
                                           leadscrew tube in b dir
actfor8(i)=afor8(3,i);      %linear force applied to leadscrew tube
tstr8(i)=actfor8(i)/areals; % tensile stress
mstr18(i)=m81(i)*cf/ina;
mstr28(i)=m82(i)*cf/ina;
vstr18(i)=2*afor8(1,i)/areals;
vstr28(i)=2*afor8(2,i)/areals;
vmst8a(i)=sqrt((abs(mstr18(i))+abs(tstr8(i)))^2+3*vstr18(i)^2);
vmst8b(i)=sqrt((abs(mstr28(i))+abs(tstr8(i)))^2+3*vstr28(i)^2);

%   To find cantilever forces for Member 10

A5b=c9*A5(i,:)';
acg10b1=A5b(1)+alp92(i)*ls/2;
acg10b2=A5b(2)-alp91(i)*ls/2;
acg10b3=A5b(3)-(wt91^2+wt92^2)*ls/2;
acg10n=c9'*[acg10b1;acg10b2;acg10b3];
ww=massls*g;
f1=forces(43,i);
f2=forces(44,i);
f3=forces(45,i);
intmat=[1 0 0 0 0;0 1 0 0 0;0 0 1 0 0;ls/2.*c9(2,:) 1 0;...
                                        -l2/2*c9(1,:) 0 -1];
intfv=[massls*acg10n(1)-f1;massls*acg10n(2)-f2;massls*acg10n(3)-f3+ww;...
                iglsb1*alp91(i)+ls/2*c9(2,:)*[f1;f2;f3];...
                iglsb2*alp91(i)-ls/2*c9(1,:)*[f1;f2;f3]];
lsfor=inv(intmat)*intfv;
f10n1(i)=lsfor(1);
f10n2(i)=lsfor(2);
f10n3(i)=lsfor(3);
m101(i)=lsfor(4);
m102(i)=lsfor(5);
afor10(:,i)=c9*[f10n1(i);f10n2(i);f10n3(i)];
actfor10(i)=afor10(3,i);
tstr10(i)=actfor10(i)/areals;
mstr110(i)=m101(i)*cf/ina;
mstr210(i)=m102(i)*cf/ina;
vstr110(i)=2*afor10(1,i)/areals;
```

```
vstr210(i)=2*afor10(2,i)/areals;
vmst10a(i)=sqrt((abs(mstr110(i))+abs(tstr10(i)))^2+3*vstr110(i)^2);
vmst10b(i)=sqrt((abs(mstr210(i))+abs(tstr10(i)))^2+3*vstr210(i)^2);

%   To find cantilever forces for Member 12

A6b=c11*A6(i,:)';
acg12b1=A6b(1)+alp112(i)*ls/2;
acg12b2=A6b(2)-alp111(i)*ls/2;
acg12b3=A6b(3)-(wt111^2+wt112^2)*ls/2;
acg12n=c11'*[acg12b1;acg12b2;acg12b3];
ww=massls*g;
f1=forces(52,i);
f2=forces(53,i);
f3=forces(54,i);
intmat=[1 0 0 0 0;0 1 0 0 0;0 0 1 0 0;ls/2.*c9(2,:) 1 0;...
                                        -l2/2*c9(1,:) 0 -1];
intfv=[massls*acg12n(1)-f1;massls*acg12n(2)-f2;massls*acg12n(3)-f3+ww;...
          iglsb1*alp111(i)+ls/2*c11(2,:)*[f1;f2;f3];...
          iglsb2*alp111(i)-ls/2*c11(1,:)*[f1;f2;f3]];
lsfor=inv(intmat)*intfv;
f12n1(i)=lsfor(1);
f12n2(i)=lsfor(2);
f12n3(i)=lsfor(3);
m121(i)=lsfor(4);
m122(i)=lsfor(5);
afor12(:,i)=c11*[f12n1(i);f12n2(i);f12n3(i)];
actfor12(i)=afor12(3,i);
tstr12(i)=actfor12(i)/areals;
mstr112(i)=m121(i)*cf/ina;
mstr212(i)=m122(i)*cf/ina;
vstr112(i)=2*afor12(1,i)/areals;
vstr212(i)=2*afor12(2,i)/areals;
vmst12a(i)=sqrt((abs(mstr112(i))+abs(tstr12(i)))^2+3*vstr112(i)^2);
vmst12b(i)=sqrt((abs(mstr212(i))+abs(tstr12(i)))^2+3*vstr212(i)^2);


%   For Member 13

azx=A6(i,1)+3*(A9(i,1)-A6(i,1))/4;
azy=A6(i,2)+3*(A9(i,2)-A6(i,2))/4;
azz=A6(i,3)+3*(A9(i,3)-A6(i,3))/4;
azn=[azx;azy;azz];
azb1=c13(1,:)*azn;
```

```
azb2=c13(2,:)*azn;
azb3=c13(3,:)*azn;
ww=mass13*g/2;
f1=c13(1,:)*forces(58:60,i);
f2=c13(2,:)*forces(58:60,i);
f3=c13(3,:)*forces(58:60,i);
ibz=i13b1/16;
intmat=[-1 0 0 0 0;0 -1 0 0 0;0 0 -1 0 0;0 -l13/4 0 1 0;...
                                          l13/4 0 0 0 -1];
intfv=[(mass13/2)*azb1-f1+(c13(1,3)*ww);mass13/2*azb2-f2+(c13(2,3)*ww);...
               mass13/2*azb3-f3+c13(3,3)*ww;ibz*alp131(i)+l13/4*f2;...
               ibz*alp132(i)-l13/4*f1];
intfor=inv(intmat)*intfv;
shear113(i)=intfor(1);
shear213(i)=intfor(2);
tension13(i)=intfor(3);
moment113(i)=intfor(4);
moment213(i)=intfor(5);
tstr13(i)=tension13(i)/areaf;
mstr113(i)=moment113(i)*cf/ina;
mstr213(i)=moment213(i)*cf/ina;
vstr113(i)=2*shear113(i)/areaf;
vstr213(i)=2*shear213(i)/areaf;
vmst13a(i)=sqrt((abs(mstr113(i))+abs(tstr13(i)))^2+3*vstr113(i)^2);
vmst13b(i)=sqrt((abs(mstr213(i))+abs(tstr13(i)))^2+3*vstr213(i)^2);

%   For Member 14

azx=A6(i,1)+3*(A7(i,1)-A6(i,1))/4;
azy=A6(i,2)+3*(A7(i,2)-A6(i,2))/4;
azz=A6(i,3)+3*(A7(i,3)-A6(i,3))/4;
azn=[azx;azy;azz];
azb1=c14(1,:)*azn;
azb2=c14(2,:)*azn;
azb3=c14(3,:)*azn;
ww=mass14*g/2;
f1=c14(1,:)*forces(64:66,i);
f2=c14(2,:)*forces(64:66,i);
f3=c14(3,:)*forces(64:66,i);
ibz=i14b1/16;
intmat=[-1 0 0 0 0;0 -1 0 0 0;0 0 -1 0 0;0 -l14/4 0 1 0;...
                                          l14/4 0 0 0 -1];
intfv=[(mass14/2)*azb1-f1+(c14(1,3)*ww);mass14/2*azb2-f2+(c14(2,3)*ww);...
            mass14/2*azb3-f3+c14(3,3)*ww;ibz*alp141(i)+l14/4*f2;...
```

```
            ibz*alp142(i)-l14/4*f1];
intfor=inv(intmat)*intfv;
shear114(i)=intfor(1);
shear214(i)=intfor(2);
tension14(i)=intfor(3);
moment114(i)=intfor(4);
moment214(i)=intfor(5);
tstr14(i)=tension14(i)/areaf;
mstr114(i)=moment114(i)*cf/ina;
mstr214(i)=moment214(i)*cf/ina;
vstr114(i)=2*shear114(i)/areaf;
vstr214(i)=2*shear214(i)/areaf;
vmst14a(i)=sqrt((abs(mstr114(i))+abs(tstr14(i)))^2+3*vstr114(i)^2);
vmst14b(i)=sqrt((abs(mstr214(i))+abs(tstr14(i)))^2+3*vstr214(i)^2);

%   For Member 15

azx=A4(i,1)+3*(A7(i,1)-A4(i,1))/4;
azy=A4(i,2)+3*(A7(i,2)-A4(i,2))/4;
azz=A4(i,3)+3*(A7(i,3)-A4(i,3))/4;
azn=[azx;azy;azz];
azb1=c15(1,:)*azn;
azb2=c15(2,:)*azn;
azb3=c15(3,:)*azn;
ww=mass15*g/2;
f1=c15(1,:)*forces(70:72,i);
f2=c15(2,:)*forces(70:72,i);
f3=c15(3,:)*forces(70:72,i);
ibz=i15b1/16;
intmat=[-1 0 0 0 0;0 -1 0 0 0;0 0 -1 0 0;0 -l15/4 0 1 0;...
                                          l15/4 0 0 0 -1];
intfv=[(mass15/2)*azb1-f1+(c15(1,3)*ww);mass15/2*azb2-f2+(c15(2,3)*ww);...
              mass15/2*azb3-f3+c15(3,3)*ww;ibz*alp151(i)+l15/4*f2;...
              ibz*alp152(i)-l15/4*f1];
intfor=inv(intmat)*intfv;
shear115(i)=intfor(1);
shear215(i)=intfor(2);
tension15(i)=intfor(3);
moment115(i)=intfor(4);
moment215(i)=intfor(5);
tstr15(i)=tension15(i)/areaf;
mstr115(i)=moment115(i)*cf/ina;
mstr215(i)=moment215(i)*cf/ina;
vstr115(i)=2*shear115(i)/areaf;
```

```
vstr215(i)=2*shear215(i)/areaf;
vmst15a(i)=sqrt((abs(mstr115(i))+abs(tstr15(i)))^2+3*vstr115(i)^2);
vmst15b(i)=sqrt((abs(mstr215(i))+abs(tstr15(i)))^2+3*vstr215(i)^2);


%   For Member 16

azx=A4(i,1)+3*(A8(i,1)-A4(i,1))/4;
azy=A4(i,2)+3*(A8(i,2)-A4(i,2))/4;
azz=A4(i,3)+3*(A8(i,3)-A4(i,3))/4;
azn=[azx;azy;azz];
azb1=c16(1,:)*azn;
azb2=c16(2,:)*azn;
azb3=c16(3,:)*azn;
ww=mass16*g/2;
f1=c16(1,:)*forces(76:78,i);
f2=c16(2,:)*forces(76:78,i);
f3=c16(3,:)*forces(76:78,i);
ibz=i16b1/16;
intmat=[-1 0 0 0 0;0 -1 0 0 0;0 0 -1 0 0;0 -l16/4 0 1 0;...
                                                l16/4 0 0 0 -1];
intfv=[(mass16/2)*azb1-f1+(c16(1,3)*ww);mass16/2*azb2-f2+(c16(2,3)*ww);...
                mass16/2*azb3-f3+c16(3,3)*ww;ibz*alp161(i)+l16/4*f2;...
                ibz*alp162(i)-l16/4*f1];
intfor=inv(intmat)*intfv;
shear116(i)=intfor(1);
shear216(i)=intfor(2);
tension16(i)=intfor(3);
moment116(i)=intfor(4);
moment216(i)=intfor(5);
tstr16(i)=tension16(i)/areaf;
mstr116(i)=moment116(i)*cf/ina;
mstr216(i)=moment216(i)*cf/ina;
vstr116(i)=2*shear116(i)/areaf;
vstr216(i)=2*shear216(i)/areaf;
vmst16a(i)=sqrt((abs(mstr116(i))+abs(tstr16(i)))^2+3*vstr116(i)^2);
vmst16b(i)=sqrt((abs(mstr216(i))+abs(tstr16(i)))^2+3*vstr216(i)^2);


%   For Member 17

azx=A5(i,1)+3*(A8(i,1)-A5(i,1))/4;
azy=A5(i,2)+3*(A8(i,2)-A5(i,2))/4;
azz=A5(i,3)+3*(A8(i,3)-A5(i,3))/4;
```

```
azn=[azx;azy;azz];
azb1=c17(1,:)*azn;
azb2=c17(2,:)*azn;
azb3=c17(3,:)*azn;
ww=mass17*g/2;
f1=c17(1,:)*forces(82:84,i);
f2=c17(2,:)*forces(82:84,i);
f3=c17(3,:)*forces(82:84,i);
ibz=i17b1/16;
intmat=[-1 0 0 0 0;0 -1 0 0 0;0 0 -1 0 0;0 -l17/4 0 1 0;...
                                          l17/4 0 0 0 -1];
intfv=[(mass17/2)*azb1-f1+(c17(1,3)*ww);mass17/2*azb2-f2+(c17(2,3)*ww);...
            mass17/2*azb3-f3+c17(3,3)*ww;ibz*alp171(i)+l17/4*f2;...
            ibz*alp172(i)-l17/4*f1];
intfor=inv(intmat)*intfv;
shear117(i)=intfor(1);
shear217(i)=intfor(2);
tension17(i)=intfor(3);
moment117(i)=intfor(4);
moment217(i)=intfor(5);
tstr17(i)=tension17(i)/areaf;
mstr117(i)=moment117(i)*cf/ina;
mstr217(i)=moment217(i)*cf/ina;
vstr117(i)=2*shear117(i)/areaf;
vstr217(i)=2*shear217(i)/areaf;
vmst17a(i)=sqrt((abs(mstr117(i))+abs(tstr17(i)))^2+3*vstr117(i)^2);
vmst17b(i)=sqrt((abs(mstr217(i))+abs(tstr17(i)))^2+3*vstr217(i)^2);


%   For Member 18

azx=A5(i,1)+3*(A9(i,1)-A5(i,1))/4;
azy=A5(i,2)+3*(A9(i,2)-A5(i,2))/4;
azz=A5(i,3)+3*(A9(i,3)-A5(i,3))/4;
azn=[azx;azy;azz];
azb1=c18(1,:)*azn;
azb2=c18(2,:)*azn;
azb3=c18(3,:)*azn;
ww=mass18*g/2;
f1=c18(1,:)*forces(88:90,i);
f2=c18(2,:)*forces(88:90,i);
f3=c18(3,:)*forces(88:90,i);
ibz=i18b1/16;
intmat=[-1 0 0 0 0;0 -1 0 0 0;0 0 -1 0 0;0 -l18/4 0 1 0;...
```

```
                                                  l18/4 0 0 0 -1];
intfv=[(mass18/2)*azb1-f1+(c18(1,3)*ww);mass18/2*azb2-f2+(c18(2,3)*ww);...
                  mass18/2*azb3-f3+c18(3,3)*ww;ibz*alp181(i)+l18/4*f2;...
                  ibz*alp182(i)-l18/4*f1];
intfor=inv(intmat)*intfv;
shear118(i)=intfor(1);
shear218(i)=intfor(2);
tension18(i)=intfor(3);
moment118(i)=intfor(4);
moment218(i)=intfor(5);
tstr18(i)=tension18(i)/areaf;
mstr118(i)=moment118(i)*cf/ina;
mstr218(i)=moment218(i)*cf/ina;
vstr118(i)=2*shear118(i)/areaf;
vstr218(i)=2*shear218(i)/areaf;
vmst18a(i)=sqrt((abs(mstr118(i))+abs(tstr18(i)))^2+3*vstr118(i)^2);
vmst18b(i)=sqrt((abs(mstr218(i))+abs(tstr18(i)))^2+3*vstr218(i)^2);

%   For Member 19

azx=A7(i,1)+3*(A9(i,1)-A7(i,1))/4;
azy=A7(i,2)+3*(A9(i,2)-A7(i,2))/4;
azz=A7(i,3)+3*(A9(i,3)-A7(i,3))/4;
azn=[azx;azy;azz];
azb1=c19(1,:)*azn;
azb2=c19(2,:)*azn;
azb3=c19(3,:)*azn;
ww=mass19*g/2;
f1=c19(1,:)*forces(94:96,i);
f2=c19(2,:)*forces(94:96,i);
f3=c19(3,:)*forces(94:96,i);
ibz=i19b1/16;
intmat=[-1 0 0 0 0;0 -1 0 0 0;0 0 -1 0 0;0 -l19/4 0 1 0;...
                                                  l19/4 0 0 0 -1];
intfv=[(mass19/2)*azb1-f1+(c19(1,3)*ww);mass19/2*azb2-f2+(c19(2,3)*ww);...
                mass19/2*azb3-f3+c19(3,3)*ww;ibz*alp191(i)+l19/4*f2;...
                ibz*alp192(i)-l19/4*f1];
intfor=inv(intmat)*intfv;
shear119(i)=intfor(1);
shear219(i)=intfor(2);
tension19(i)=intfor(3);
moment119(i)=intfor(4);
moment219(i)=intfor(5);
tstr19(i)=tension19(i)/areaf;
```

```
mstr119(i)=moment119(i)*cf/ina;
mstr219(i)=moment219(i)*cf/ina;
vstr119(i)=2*shear119(i)/areaf;
vstr219(i)=2*shear219(i)/areaf;
vmst19a(i)=sqrt((abs(mstr119(i))+abs(tstr19(i)))^2+3*vstr119(i)^2);
vmst19b(i)=sqrt((abs(mstr219(i))+abs(tstr19(i)))^2+3*vstr219(i)^2);

%   For Member 20

azx=A7(i,1)+3*(A8(i,1)-A7(i,1))/4;
azy=A7(i,2)+3*(A8(i,2)-A7(i,2))/4;
azz=A7(i,3)+3*(A8(i,3)-A7(i,3))/4;
 azn=[azx;azy;azz];
azb1=c20(1,:)*azn;
azb2=c20(2,:)*azn;
azb3=c20(3,:)*azn;
ww=mass20*g/2;
f1=c20(1,:)*forces(100:102,i);
f2=c20(2,:)*forces(100:102,i);
f3=c20(3,:)*forces(100:102,i);
ibz=i20b1/16;
intmat=[-1 0 0 0 0;0 -1 0 0 0;0 0 -1 0 0;0 -l20/4 0 1 0;...
                                          l20/4 0 0 0 -1];
intfv=[(mass20/2)*azb1-f1+(c20(1,3)*ww);mass20/2*azb2-f2+(c20(2,3)*ww);...
               mass20/2*azb3-f3+c20(3,3)*ww;ibz*alp201(i)+l20/4*f2;...
               ibz*alp202(i)-l20/4*f1];
intfor=inv(intmat)*intfv;
shear120(i)=intfor(1);
shear220(i)=intfor(2);
tension20(i)=intfor(3);
moment120(i)=intfor(4);
moment220(i)=intfor(5);
tstr20(i)=tension20(i)/areaf;
mstr120(i)=moment120(i)*cf/ina;
mstr220(i)=moment220(i)*cf/ina;
vstr120(i)=2*shear120(i)/areaf;
vstr220(i)=2*shear220(i)/areaf;
vmst20a(i)=sqrt((abs(mstr120(i))+abs(tstr20(i)))^2+3*vstr120(i)^2);
vmst20b(i)=sqrt((abs(mstr220(i))+abs(tstr20(i)))^2+3*vstr220(i)^2);

%   For Member 21

azx=A8(i,1)+3*(A9(i,1)-A8(i,1))/4;
azy=A8(i,2)+3*(A9(i,2)-A8(i,2))/4;
```

```
azz=A8(i,3)+3*(A9(i,3)-A8(i,3))/4;
azn=[azx;azy;azz];
azb1=c21(1,:)*azn;
azb2=c21(2,:)*azn;
azb3=c21(3,:)*azn;
ww=mass21*g/2;
f1=c21(1,:)*forces(106:108,i);
f2=c21(2,:)*forces(106:108,i);
f3=c21(3,:)*forces(106:108,i);
ibz=i21b1/16;
intmat=[-1 0 0 0 0;0 -1 0 0 0;0 0 -1 0 0;0 -l21/4 0 1 0;...
                                          l21/4 0 0 0 -1];
intfv=[(mass21/2)*azb1-f1+(c21(1,3)*ww);mass21/2*azb2-f2+(c21(2,3)*ww);...
                mass21/2*azb3-f3+c21(3,3)*ww;ibz*alp211(i)+l21/4*f2;...
                ibz*alp212(i)-l21/4*f1];
intfor=inv(intmat)*intfv;
shear121(i)=intfor(1);
shear221(i)=intfor(2);
tension21(i)=intfor(3);
moment121(i)=intfor(4);
moment221(i)=intfor(5);
tstr21(i)=tension21(i)/areaf;
mstr121(i)=moment121(i)*cf/ina;
mstr221(i)=moment221(i)*cf/ina;
vstr121(i)=2*shear121(i)/areaf;
vstr221(i)=2*shear221(i)/areaf;
vmst21a(i)=sqrt((abs(mstr121(i))+abs(tstr21(i)))^2+3*vstr121(i)^2);
vmst21b(i)=sqrt((abs(mstr221(i))+abs(tstr21(i)))^2+3*vstr221(i)^2);

%      Now need to check buckling for fixed members
%        Define Buckling Failure Load
Pcrf=pi^2*E*ina/l1^2;

%     Now find maximum compressive load in each member
%      For Member 1
maxf1=abs(min(tension1));
%      For Member 2
maxf2=abs(min(tension2));
%      For Member 3
maxf3=abs(min(tension3));
%      For Member 4
maxf4=abs(min(tension4));
%      For Member 5
maxf5=abs(min(tension5));
```

```
%       For Member 6
maxf6=abs(min(tension6));
%       For Member 13
maxf13=abs(min(tension13));
%       For Member 14
maxf14=abs(min(tension14));
%       For Member 15
maxf15=abs(min(tension15));
%       For Member 16
maxf16=abs(min(tension16));
%       For Member 17
maxf17=abs(min(tension17));
%       For Member 18
maxf18=abs(min(tension18));
%       For Member 19
maxf19=abs(min(tension19));
%       For Member 20
maxf20=abs(min(tension20));
%       For Member 21
maxf21=abs(min(tension21));

%        Now Calculate static values
%  Set up static inertial force vectors for each member
stinert1=[0 0 mass1*g 0 0];
stinert2=[0 0 mass2*g 0 0];
stinert3=[0 0 mass3*g 0 0];
stinert4=[0 0 mass4*g 0 0];
stinert5=[0 0 mass5*g 0 0];
stinert6=[0 0 mass6*g 0 0];
stinert7=[0 0 mass7*g 0 0];
stinert9=[0 0 mass9*g 0 0];
stinert11=[0 0 mass11*g 0 0];
stinert13=[0 0 mass13*g 0 0];
stinert14=[0 0 mass14*g 0 0];
stinert15=[0 0 mass15*g 0 0];
stinert16=[0 0 mass16*g 0 0];
stinert17=[0 0 mass17*g 0 0];
stinert18=[0 0 mass18*g 0 0];
stinert19=[0 0 mass19*g 0 0];
stinert20=[0 0 mass20*g 0 0];
stinert21=[0 0 mass21*g 0 0];
%   Set up static constraint equations
stconin=[zeros(1,18) NN7x NN7y -lmass7*g1+NN7z NN8x NN8y...
            -lmass8*g1+NN8z NN9x NN9y -lmass9*g1+NN9z];
```

```
%   Set up the static inertial force matrix
stfinert=[stinert1 stinert2 stinert3 stinert4 stinert5 stinert6 ...
      stinert7 stinert9 stinert11 stinert13 stinert14 stinert15...
      stinert16 stinert17 stinert18 stinert19 stinert20 stinert21...
      stconin]';

%   Find the resulting forces by:   StForces = inv(fmatrix)*stfinert
Stforces(:,i)=inv(fmatrix)*stfinert;

%   Now have nodal forces for static model at each time step
%     Need to find internal forces at each time step

%    Static Internal Forces

%   For member 1
ww=mass1*g/2;
f1=c1(1,:)*Stforces(1:3,i);
f2=c1(2,:)*Stforces(1:3,i);
f3=c1(3,:)*Stforces(1:3,i);
intmat=[-1 0 0 0 0;0 -1 0 0 0;0 0 -1 0 0;0 -l1/4 0 1 0;l1/4 0 0 0 -1];
intfv=[-f1+c1(1,3)*ww;-f2+c1(2,3)*ww;-f3+c1(3,3)*ww;l1/4*f2;-l1/4*f1];
intfor=inv(intmat)*intfv;
shear11s(i)=intfor(1);
shear21s(i)=intfor(2);
tension1s(i)=intfor(3);
moment11s(i)=intfor(4);
moment21s(i)=intfor(5);
tstr1s(i)=tension1s(i)/areaf;
mstr11s(i)=moment11s(i)*cf/ina;
mstr21s(i)=moment21s(i)*cf/ina;
vstr11s(i)=2*shear11s(i)/areaf;
vstr21s(i)=2*shear21s(i)/areaf;
vmst1as(i)=sqrt((abs(mstr11s(i))+abs(tstr1s(i)))^2+3*vstr11s(i)^2);
vmst1bs(i)=sqrt((abs(mstr21s(i))+abs(tstr1s(i)))^2+3*vstr21s(i)^2);

%  For member 2
ww=mass2*g/2;
f1=c2(1,:)*Stforces(7:9,i);
f2=c2(2,:)*Stforces(7:9,i);
f3=c2(3,:)*Stforces(7:9,i);
intmat=[-1 0 0 0 0;0 -1 0 0 0;0 0 -1 0 0;0 -l2/4 0 1 0;l2/4 0 0 0 -1];
intfv=[-f1+c2(1,3)*ww;-f2+c2(2,3)*ww;-f3+c2(3,3)*ww;l2/4*f2;-l2/4*f1];
intfor=inv(intmat)*intfv;
shear12s(i)=intfor(1);
```

```
shear22s(i)=intfor(2);
tension2s(i)=intfor(3);
moment12s(i)=intfor(4);
moment22s(i)=intfor(5);
tstr2s(i)=tension2s(i)/areaf;
mstr12s(i)=moment12s(i)*cf/ina;
mstr22s(i)=moment22s(i)*cf/ina;
vstr12s(i)=2*shear12s(i)/areaf;
vstr22s(i)=2*shear22s(i)/areaf;
vmst2as(i)=sqrt((abs(mstr12s(i))+abs(tstr2s(i)))^2+3*vstr12s(i)^2);
vmst2bs(i)=sqrt((abs(mstr22s(i))+abs(tstr2s(i)))^2+3*vstr22s(i)^2);

%  For member 3
ww=mass3*g/2;
f1=c3(1,:)*Stforces(13:15,i);
f2=c3(2,:)*Stforces(13:15,i);
f3=c3(3,:)*Stforces(13:15,i);
intmat=[-1 0 0 0 0;0 -1 0 0 0;0 0 -1 0 0;0 -13/4 0 1 0;13/4 0 0 0 -1];
intfv=[-f1+c3(1,3)*ww;-f2+c3(2,3)*ww;-f3+c3(3,3)*ww;13/4*f2;-13/4*f1];
intfor=inv(intmat)*intfv;
shear13s(i)=intfor(1);
shear23s(i)=intfor(2);
tension3s(i)=intfor(3);
moment13s(i)=intfor(4);
moment23s(i)=intfor(5);
tstr3s(i)=tension3s(i)/areaf;
mstr13s(i)=moment13s(i)*cf/ina;
mstr23s(i)=moment23s(i)*cf/ina;
vstr13s(i)=2*shear13s(i)/areaf;
vstr23s(i)=2*shear23s(i)/areaf;
vmst3as(i)=sqrt((abs(mstr13s(i))+abs(tstr3s(i)))^2+3*vstr13s(i)^2);
vmst3bs(i)=sqrt((abs(mstr23s(i))+abs(tstr3s(i)))^2+3*vstr23s(i)^2);

%  For member 4
ww=mass4*g/2;
f1=c4(1,:)*Stforces(19:21,i);
f2=c4(2,:)*Stforces(19:21,i);
f3=c4(3,:)*Stforces(19:21,i);
intmat=[-1 0 0 0 0;0 -1 0 0 0;0 0 -1 0 0;0 -14/4 0 1 0;14/4 0 0 0 -1];
intfv=[-f1+c4(1,3)*ww;-f2+c4(2,3)*ww;-f3+c4(3,3)*ww;14/4*f2;-14/4*f1];
intfor=inv(intmat)*intfv;
shear14s(i)=intfor(1);
shear24s(i)=intfor(2);
tension4s(i)=intfor(3);
```

```
moment14s(i)=intfor(4);
moment24s(i)=intfor(5);
tstr4s(i)=tension4s(i)/areaf;
mstr14s(i)=moment14s(i)*cf/ina;
mstr24s(i)=moment24s(i)*cf/ina;
vstr14s(i)=2*shear14s(i)/areaf;
vstr24s(i)=2*shear24s(i)/areaf;
vmst4as(i)=sqrt((abs(mstr14s(i))+abs(tstr4s(i)))^2+3*vstr14s(i)^2);
vmst4bs(i)=sqrt((abs(mstr24s(i))+abs(tstr4s(i)))^2+3*vstr24s(i)^2);

%  For member 5
ww=mass5*g/2;
f1=c5(1,:)*Stforces(25:27,i);
f2=c5(2,:)*Stforces(25:27,i);
f3=c5(3,:)*Stforces(25:27,i);
intmat=[-1 0 0 0 0;0 -1 0 0 0;0 0 -1 0 0;0 -15/4 0 1 0;15/4 0 0 0 -1];
intfv=[-f1+c5(1,3)*ww;-f2+c5(2,3)*ww;-f3+c5(3,3)*ww;15/4*f2;-15/4*f1];
intfor=inv(intmat)*intfv;
shear15s(i)=intfor(1);
shear25s(i)=intfor(2);
tension5s(i)=intfor(3);
moment15s(i)=intfor(4);
moment25s(i)=intfor(5);
tstr5s(i)=tension5s(i)/areaf;
mstr15s(i)=moment15s(i)*cf/ina;
mstr25s(i)=moment25s(i)*cf/ina;
vstr15s(i)=2*shear15s(i)/areaf;
vstr25s(i)=2*shear25s(i)/areaf;
vmst5as(i)=sqrt((abs(mstr15s(i))+abs(tstr5s(i)))^2+3*vstr15s(i)^2);
vmst5bs(i)=sqrt((abs(mstr25s(i))+abs(tstr5s(i)))^2+3*vstr25s(i)^2);

%   For member 6

ww=mass6*g/2;
f1=c6(1,:)*Stforces(31:33,i);
f2=c6(2,:)*Stforces(31:33,i);
f3=c6(3,:)*Stforces(31:33,i);
intmat=[-1 0 0 0 0;0 -1 0 0 0;0 0 -1 0 0;0 -16/4 0 1 0;16/4 0 0 0 -1];
intfv=[-f1+(c6(1,3)*ww);-f2+(c6(2,3)*ww);-f3+c6(3,3)*ww;16/4*f2;-16/4*f1];
intfor=inv(intmat)*intfv;
shear16s(i)=intfor(1);
shear26s(i)=intfor(2);
tension6s(i)=intfor(3);
moment16s(i)=intfor(4);
```

```
moment26s(i)=intfor(5);
tstr6s(i)=tension6s(i)/areaf;
mstr16s(i)=moment16s(i)*cf/ina;
mstr26s(i)=moment26s(i)*cf/ina;
vstr16s(i)=2*shear16s(i)/areaf;
vstr26s(i)=2*shear26s(i)/areaf;
vmst6as(i)=sqrt((abs(mstr16s(i))+abs(tstr6s(i)))^2+3*vstr16s(i)^2);
vmst6bs(i)=sqrt((abs(mstr26s(i))+abs(tstr6s(i)))^2+3*vstr26s(i)^2);

%   For member 8

ww=massls*g;
f1=Stforces(37,i);
f2=Stforces(38,i);
f3=Stforces(39,i);
intmat=[1 0 0 0 0;0 1 0 0 0;0 0 1 0 0;ls/2.*c7(2,:) 1 0;...
                                        -l2/2*c7(1,:) 0 -1];
intfv=[-f1;-f2;-f3+ww;ls/2*c7(2,:)*[f1;f2;f3];...
                                    -ls/2*c7(1,:)*[f1;f2;f3]];
lsfor=inv(intmat)*intfv;
f8n1s(i)=lsfor(1);
f8n2s(i)=lsfor(2);
f8n3s(i)=lsfor(3);
m81s(i)=lsfor(4);
m82s(i)=lsfor(5);
afor8s(:,i)=c7*[f8n1s(i);f8n2s(i);f8n3s(i)];
actfor8s(i)=afor8s(3,i);
tstr8s(i)=actfor8s(i)/areals;
mstr18s(i)=m81s(i)*cf/ina;
mstr28s(i)=m82s(i)*cf/ina;
vstr18s(i)=2*afor8s(1,i)/areals;
vstr28s(i)=2*afor8s(2,i)/areals;
vmst8as(i)=sqrt((abs(mstr18s(i))+abs(tstr8s(i)))^2+3*vstr18s(i)^2);
vmst8bs(i)=sqrt((abs(mstr28s(i))+abs(tstr8s(i)))^2+3*vstr28s(i)^2);

%       For member 10
ww=massls*g;
f1=Stforces(43,i);
f2=Stforces(44,i);
f3=Stforces(45,i);
intmat=[1 0 0 0 0;0 1 0 0 0;0 0 1 0 0;ls/2.*c9(2,:) 1 0;...
                                        -l2/2*c9(1,:) 0 -1];
intfv=[-f1;-f2;-f3+ww;ls/2*c9(2,:)*[f1;f2;f3];...
                                    -ls/2*c9(1,:)*[f1;f2;f3]];
```

```
lsfor=inv(intmat)*intfv;
f10n1s(i)=lsfor(1);
f10n2s(i)=lsfor(2);
f10n3s(i)=lsfor(3);
m101s(i)=lsfor(4);
m102s(i)=lsfor(5);
afor10s(:,i)=c9*[f10n1s(i);f10n2s(i);f10n3s(i)];
actfor10s(i)=afor10s(3,i);
tstr10s(i)=actfor10s(i)/areals;
mstr110s(i)=m101s(i)*cf/ina;
mstr210s(i)=m102s(i)*cf/ina;
vstr110s(i)=2*afor10s(1,i)/areals;
vstr210s(i)=2*afor10s(2,i)/areals;
vmst10as(i)=sqrt((abs(mstr110s(i))+abs(tstr10s(i)))^2+3*vstr110s(i)^2);
vmst10bs(i)=sqrt((abs(mstr210s(i))+abs(tstr10s(i)))^2+3*vstr210s(i)^2);

%   To find cantilever forces for Member 12

ww=massls*g;
f1=Stforces(52,i);
f2=Stforces(53,i);
f3=Stforces(54,i);
intmat=[1 0 0 0 0;0 1 0 0 0;0 0 1 0 0;ls/2.*c9(2,:) 1 0;...
                                        -l2/2*c9(1,:) 0 -1];
intfv=[-f1;-f2;-f3+ww;ls/2*c11(2,:)*[f1;f2;f3];...
                                    -ls/2*c11(1,:)*[f1;f2;f3]];
lsfor=inv(intmat)*intfv;
f12n1s(i)=lsfor(1);
f12n2s(i)=lsfor(2);
f12n3s(i)=lsfor(3);
m121s(i)=lsfor(4);
m122s(i)=lsfor(5);
afor12s(:,i)=c11*[f12n1s(i);f12n2s(i);f12n3s(i)];
actfor12s(i)=afor12s(3,i);
tstr12s(i)=actfor12s(i)/areals;
mstr112s(i)=m121s(i)*cf/ina;
mstr212s(i)=m122s(i)*cf/ina;
vstr112s(i)=2*afor12s(1,i)/areals;
vstr212s(i)=2*afor12s(2,i)/areals;
vmst12as(i)=sqrt((abs(mstr112s(i))+abs(tstr12s(i)))^2+3*vstr112s(i)^2);
vmst12bs(i)=sqrt((abs(mstr212s(i))+abs(tstr12s(i)))^2+3*vstr212s(i)^2);


%   For Member 13
```

```
ww=mass13*g/2;
f1=c13(1,:)*Stforces(58:60,i);
f2=c13(2,:)*Stforces(58:60,i);
f3=c13(3,:)*Stforces(58:60,i);
intmat=[-1 0 0 0 0;0 -1 0 0 0;0 0 -1 0 0;0 -l13/4 0 1 0;...
                                                    l13/4 0 0 0 -1];
intfv=[-f1+(c13(1,3)*ww);-f2+(c13(2,3)*ww);-f3+c13(3,3)*ww;...
                                                   l13/4*f2;-l13/4*f1];
intfor=inv(intmat)*intfv;
shear113s(i)=intfor(1);
shear213s(i)=intfor(2);
tension13s(i)=intfor(3);
moment113s(i)=intfor(4);
moment213s(i)=intfor(5);
tstr13s(i)=tension13s(i)/areaf;
mstr113s(i)=moment113s(i)*cf/ina;
mstr213s(i)=moment213s(i)*cf/ina;
vstr113s(i)=2*shear113s(i)/areaf;
vstr213s(i)=2*shear213s(i)/areaf;
vmst13as(i)=sqrt((abs(mstr113s(i))+abs(tstr13s(i)))^2+3*vstr113s(i)^2);
vmst13bs(i)=sqrt((abs(mstr213s(i))+abs(tstr13s(i)))^2+3*vstr213s(i)^2);

%   For Member 14

ww=mass14*g/2;
f1=c14(1,:)*Stforces(64:66,i);
f2=c14(2,:)*Stforces(64:66,i);
f3=c14(3,:)*Stforces(64:66,i);
intmat=[-1 0 0 0 0;0 -1 0 0 0;0 0 -1 0 0;0 -l14/4 0 1 0;l14/4 0 0 0 -1];
intfv=[-f1+(c14(1,3)*ww);-f2+(c14(2,3)*ww);-f3+c14(3,3)*ww;...
                                                    l14/4*f2;-l14/4*f1];
intfor=inv(intmat)*intfv;
shear114s(i)=intfor(1);
shear214s(i)=intfor(2);
tension14s(i)=intfor(3);
moment114s(i)=intfor(4);
moment214s(i)=intfor(5);
tstr14s(i)=tension14s(i)/areaf;
mstr114s(i)=moment114s(i)*cf/ina;
mstr214s(i)=moment214s(i)*cf/ina;
vstr114s(i)=2*shear114s(i)/areaf;
vstr214s(i)=2*shear214s(i)/areaf;
vmst14as(i)=sqrt((abs(mstr114s(i))+abs(tstr14s(i)))^2+3*vstr114s(i)^2);
```

```
vmst14bs(i)=sqrt((abs(mstr214s(i))+abs(tstr14s(i)))^2+3*vstr214s(i)^2);


%   For Member 15

ww=mass15*g/2;
f1=c15(1,:)*Stforces(70:72,i);
f2=c15(2,:)*Stforces(70:72,i);
f3=c15(3,:)*Stforces(70:72,i);
intmat=[-1 0 0 0 0;0 -1 0 0 0;0 0 -1 0 0;0 -l15/4 0 1 0;l15/4 0 0 0 -1];
intfv=[-f1+(c15(1,3)*ww);-f2+(c15(2,3)*ww);-f3+c15(3,3)*ww;...
                                          l15/4*f2;-l15/4*f1];
intfor=inv(intmat)*intfv;
shear115s(i)=intfor(1);
shear215s(i)=intfor(2);
tension15s(i)=intfor(3);
moment115s(i)=intfor(4);
moment215s(i)=intfor(5);
tstr15s(i)=tension15s(i)/areaf;
mstr115s(i)=moment115s(i)*cf/ina;
mstr215s(i)=moment215s(i)*cf/ina;
vstr115s(i)=2*shear115s(i)/areaf;
vstr215s(i)=2*shear215s(i)/areaf;
vmst15as(i)=sqrt((abs(mstr115s(i))+abs(tstr15s(i)))^2+3*vstr115s(i)^2);
vmst15bs(i)=sqrt((abs(mstr215s(i))+abs(tstr15s(i)))^2+3*vstr215s(i)^2);


%   For Member 16

ww=mass16*g/2;
f1=c16(1,:)*Stforces(76:78,i);
f2=c16(2,:)*Stforces(76:78,i);
f3=c16(3,:)*Stforces(76:78,i);
intmat=[-1 0 0 0 0;0 -1 0 0 0;0 0 -1 0 0;0 -l16/4 0 1 0;l16/4 0 0 0 -1];
intfv=[-f1+(c16(1,3)*ww);-f2+(c16(2,3)*ww);-f3+c16(3,3)*ww;...
                                          l16/4*f2;-l16/4*f1];
intfor=inv(intmat)*intfv;
shear116s(i)=intfor(1);
shear216s(i)=intfor(2);
tension16s(i)=intfor(3);
moment116s(i)=intfor(4);
moment216s(i)=intfor(5);
tstr16s(i)=tension16s(i)/areaf;
mstr116s(i)=moment116s(i)*cf/ina;
```

```
mstr216s(i)=moment216s(i)*cf/ina;
vstr116s(i)=2*shear116s(i)/areaf;
vstr216s(i)=2*shear216s(i)/areaf;
vmst16as(i)=sqrt((abs(mstr116s(i))+abs(tstr16s(i)))^2+3*vstr116s(i)^2);
vmst16bs(i)=sqrt((abs(mstr216s(i))+abs(tstr16s(i)))^2+3*vstr216s(i)^2);


%   For Member 17

ww=mass17*g/2;
f1=c17(1,:)*Stforces(82:84,i);
f2=c17(2,:)*Stforces(82:84,i);
f3=c17(3,:)*Stforces(82:84,i);
intmat=[-1 0 0 0 0;0 -1 0 0 0;0 0 -1 0 0;0 -l17/4 0 1 0;l17/4 0 0 0 -1];
intfv=[-f1+(c17(1,3)*ww);-f2+(c17(2,3)*ww);-f3+c17(3,3)*ww;...
                                            l17/4*f2;-l17/4*f1];
intfor=inv(intmat)*intfv;
shear117s(i)=intfor(1);
shear217s(i)=intfor(2);
tension17s(i)=intfor(3);
moment117s(i)=intfor(4);
moment217s(i)=intfor(5);
tstr17s(i)=tension17s(i)/areaf;
mstr117s(i)=moment117s(i)*cf/ina;
mstr217s(i)=moment217s(i)*cf/ina;
vstr117s(i)=2*shear117s(i)/areaf;
vstr217s(i)=2*shear217s(i)/areaf;
vmst17as(i)=sqrt((abs(mstr117s(i))+abs(tstr17s(i)))^2+3*vstr117s(i)^2);
vmst17bs(i)=sqrt((abs(mstr217s(i))+abs(tstr17s(i)))^2+3*vstr217s(i)^2);


%   For Member 18

ww=mass18*g/2;
f1=c18(1,:)*Stforces(88:90,i);
f2=c18(2,:)*Stforces(88:90,i);
f3=c18(3,:)*Stforces(88:90,i);
intmat=[-1 0 0 0 0;0 -1 0 0 0;0 0 -1 0 0;0 -l18/4 0 1 0;l18/4 0 0 0 -1];
intfv=[-f1+(c18(1,3)*ww);-f2+(c18(2,3)*ww);-f3+c18(3,3)*ww;...
                                            l18/4*f2;-l18/4*f1];
intfor=inv(intmat)*intfv;
shear118s(i)=intfor(1);
shear218s(i)=intfor(2);
tension18s(i)=intfor(3);
```

```
moment118s(i)=intfor(4);
moment218s(i)=intfor(5);
tstr18s(i)=tension18s(i)/areaf;
mstr118s(i)=moment118s(i)*cf/ina;
mstr218s(i)=moment218s(i)*cf/ina;
vstr118s(i)=2*shear118s(i)/areaf;
vstr218s(i)=2*shear218s(i)/areaf;
vmst18as(i)=sqrt((abs(mstr118s(i))+abs(tstr18s(i)))^2+3*vstr118s(i)^2);
vmst18bs(i)=sqrt((abs(mstr218s(i))+abs(tstr18s(i)))^2+3*vstr218s(i)^2);

%   For Member 19

ww=mass19*g/2;
f1=c19(1,:)*Stforces(94:96,i);
f2=c19(2,:)*Stforces(94:96,i);
f3=c19(3,:)*Stforces(94:96,i);
intmat=[-1 0 0 0 0;0 -1 0 0 0;0 0 -1 0 0;0 -l19/4 0 1 0;l19/4 0 0 0 -1];
intfv=[-f1+(c19(1,3)*ww);-f2+(c19(2,3)*ww);-f3+c19(3,3)*ww;...
                                          l19/4*f2;-l19/4*f1];
intfor=inv(intmat)*intfv;
shear119s(i)=intfor(1);
shear219s(i)=intfor(2);
tension19s(i)=intfor(3);
moment119s(i)=intfor(4);
moment219s(i)=intfor(5);
tstr19s(i)=tension19s(i)/areaf;
mstr119s(i)=moment119s(i)*cf/ina;
mstr219s(i)=moment219s(i)*cf/ina;
vstr119s(i)=2*shear119s(i)/areaf;
vstr219s(i)=2*shear219s(i)/areaf;
vmst19as(i)=sqrt((abs(mstr119s(i))+abs(tstr19s(i)))^2+3*vstr119s(i)^2);
vmst19bs(i)=sqrt((abs(mstr219s(i))+abs(tstr19s(i)))^2+3*vstr219s(i)^2);
%   For Member 20

ww=mass20*g/2;
f1=c20(1,:)*Stforces(100:102,i);
f2=c20(2,:)*Stforces(100:102,i);
f3=c20(3,:)*Stforces(100:102,i);
intmat=[-1 0 0 0 0;0 -1 0 0 0;0 0 -1 0 0;0 -l20/4 0 1 0;l20/4 0 0 0 -1];
intfv=[-f1+(c20(1,3)*ww);-f2+(c20(2,3)*ww);-f3+c20(3,3)*ww;...
                                          l20/4*f2;-l20/4*f1];
intfor=inv(intmat)*intfv;
shear120s(i)=intfor(1);
shear220s(i)=intfor(2);
```

```
tension20s(i)=intfor(3);
moment120s(i)=intfor(4);
moment220s(i)=intfor(5);
tstr20s(i)=tension20s(i)/areaf;
mstr120s(i)=moment120s(i)*cf/ina;
mstr220s(i)=moment220s(i)*cf/ina;
vstr120s(i)=2*shear120s(i)/areaf;
vstr220s(i)=2*shear220s(i)/areaf;
vmst20as(i)=sqrt((abs(mstr120s(i))+abs(tstr20s(i)))^2+3*vstr120s(i)^2);
vmst20bs(i)=sqrt((abs(mstr220s(i))+abs(tstr20s(i)))^2+3*vstr220s(i)^2);

%   For Member 21

ww=mass21*g/2;
f1=c21(1,:)*Stforces(106:108,i);
f2=c21(2,:)*Stforces(106:108,i);
f3=c21(3,:)*Stforces(106:108,i);
intmat=[-1 0 0 0 0;0 -1 0 0 0;0 0 -1 0 0;0 -l21/4 0 1 0;l21/4 0 0 0 -1];
intfv=[-f1+(c21(1,3)*ww);-f2+(c21(2,3)*ww);-f3+c21(3,3)*ww;...
                                              l21/4*f2;-l21/4*f1];
intfor=inv(intmat)*intfv;
shear121s(i)=intfor(1);
shear221s(i)=intfor(2);
tension21s(i)=intfor(3);
moment121s(i)=intfor(4);
moment221s(i)=intfor(5);
tstr21s(i)=tension21s(i)/areaf;
mstr121s(i)=moment121s(i)*cf/ina;
mstr221s(i)=moment221s(i)*cf/ina;
vstr121s(i)=2*shear121s(i)/areaf;
vstr221s(i)=2*shear221s(i)/areaf;
vmst21as(i)=sqrt((abs(mstr121s(i))+abs(tstr21s(i)))^2+3*vstr121s(i)^2);
vmst21bs(i)=sqrt((abs(mstr221s(i))+abs(tstr21s(i)))^2+3*vstr221s(i)^2);

%       Now need to check buckling for fixed members
%       For Member 1
maxf1s=abs(min(tension1s));
%       For Member 2
maxf2s=abs(min(tension2s));
%       For Member 3
maxf3s=abs(min(tension3s));
%       For Member 4
maxf4s=abs(min(tension4s));
%       For Member 5
```

```
maxf5s=abs(min(tension5s));
%       For Member 6
maxf6s=abs(min(tension6s));
%       For Member 13
maxf13s=abs(min(tension13s));
%       For Member 14
maxf14s=abs(min(tension14s));
%       For Member 15
maxf15s=abs(min(tension15s));
%       For Member 16
maxf16s=abs(min(tension16s));
%       For Member 17
maxf17s=abs(min(tension17s));
%       For Member 18
maxf18s=abs(min(tension18s));
%       For Member 19
maxf19s=abs(min(tension19s));
%       For Member 20
maxf20s=abs(min(tension20s));
%       For Member 21
maxf21s=abs(min(tension21s));

%    Now calculate motor torque, voltage, and current
%  Now find motor values for member M8
%  Check if torque is acting with or against load
sact1=sign(actfor8(i));
sst1=sign(actfor8s(i));
%   Calculate values with leadscerw friction
if sst1==sact1;
  Tm1(i)=(Jm*abs(xddot7(i))*Ggh/Gls)
   + (dm/2*abs(actfor8(i)))*((2*pi*Gls+pi*mu*dm)/(pi*dm-mu*2*pi*Gls));
 else
  Tm1(i)=(Jm*abs(xddot7(i))*Ggh/Gls)
   +(dm/2*abs(actfor8(i)))*((-2*pi*Gls+pi*mu*dm)/(pi*dm+mu*2*pi*Gls));
end
Vm1(i)=Tm1(i)*Ra/Kt+(Kb*Ggh*abs(xdot7(i))/Gls);     % motor voltage
Im1(i)=Tm1(i)/Kt;                                   % motor current

%  Now calculate without friction
Tm1n(i)=(Jm*abs(xddot7(i))*Ggh/Gls)+abs(actfor8(i))*Gls;% motor torque
Vm1n(i)=Tm1n(i)*Ra/Kt+(Kb*Ggh*abs(xdot7(i))/Gls);  % motor voltage
Im1n(i)=Tm1n(i)/Kt;                                % motor current

%  Now find motor values for member M10
```

```
sact2=sign(actfor10(i));
sst2=sign(actfor10s(i));
if sst2==sact2;
 Tm2(i)=(Jm*abs(xddot9(i))*Ggh/Gls)
  +(dm/2*abs(actfor10(i)))*((2*pi*Gls+pi*mu*dm)/(pi*dm-mu*2*pi*Gls));
 else
  Tm2(i)=(Jm*abs(xddot9(i))*Ggh/Gls)
   +(dm/2*abs(actfor10(i)))*((-2*pi*Gls+pi*mu*dm)/(pi*dm+mu*2*pi*Gls));
end
Vm2(i)=Tm2(i)*Ra/Kt+(Kb*Ggh*abs(xdot9(i))/Gls); % motor voltage
Im2(i)=Tm2(i)/Kt;                               % motor current
Tm2n(i)=(Jm*abs(xddot9(i))*Ggh/Gls)+abs(actfor10(i))*Gls;% motor torque
Vm2n(i)=Tm2n(i)*Ra/Kt+(Kb*Ggh*abs(xdot9(i))/Gls);  % motor voltage
Im2n(i)=Tm2n(i)/Kt;                                % motor current

%  Now find motor values for member M12
sact3=sign(actfor12(i));
sst3=sign(actfor12s(i));
if sst3==sact3;
 Tm3(i)=(Jm*abs(xddot11(i))*Ggh/Gls)
  +(dm/2*abs(actfor12(i)))*((2*pi*Gls+pi*mu*dm)/(pi*dm-mu*2*pi*Gls));
 else
 Tm3(i)=(Jm*abs(xddot11(i))*Ggh/Gls)
  +(dm/2*abs(actfor12(i)))*((-2*pi*Gls+pi*mu*dm)/(pi*dm+mu*2*pi*Gls));
end
Vm3(i)=Tm3(i)*Ra/Kt+(Kb*Ggh*abs(xdot11(i))/Gls);  % motor voltage
Im3(i)=Tm3(i)/Kt;                                 % motor current
Tm3n(i)=(Jm*abs(xddot11(i))*Ggh/Gls)+abs(actfor12(i))*Gls; % motor torque
Vm3n(i)=Tm3n(i)*Ra/Kt+(Kb*Ggh*abs(xdot11(i))/Gls); % motor voltage
Im3n(i)=Tm3n(i)/Kt;                                % motor current

end;
disp('               Done ')
clear fmatrix;
```

```
%-------------------------------------------------------
%  PROGRAM: FWD1
%      Written by: Tom Warrington
%
%    This program solves the forward problem of the
%      double-octahedral truss.  It requires the following
%      function files: kinang1, part1, link1, cross, and cross1.
%      Time histories of the link lengths are generated
%      from a file called motpro.m.
%-------------------------------------------------------

%base=input('Enter Fixed Base Length: ');
%cl=input('Enter Cross Longeron Length: ');
%d=input('Enter Height of Truss: ');

%    For Robert's Truss
base=48*2.54/100;
cl=36*2.54/100;
d=47*2.54/100;

motpro1
load lnk_inf.mat

k=length(l1);

% Define the geometric constants of the truss

xl=tan(.52359)*base/2;
rim=sqrt(cl^2-(base/2)^2);
uu1=[1,0,0];
uu2=[-.5,0,-.866];
uu3=[-.5,0,.866];
oo1=[0,0,xl];
oo2=[.866*xl,0,-xl/2];
oo3=[-.866*xl,0,-xl/2];
rr1=rim*oo1/sqrt(sum(oo1.*oo1));
rr2=rim*oo2/sqrt(sum(oo2.*oo2));
rr3=rim*oo3/sqrt(sum(oo3.*oo3));

% Start the iteration loop to solve for gamma and beta
thtaold=[0.5 0.5 0.5]';

for p=1:k
    L=[l1(p) l2(p) l3(p)]';
```

```
    theta=kinang1(L,thtaold,base,cl);
    thtaold=theta;

% Define trig id's %

    theta1=theta(1);
    theta2=theta(2);
    theta3=theta(3);
    vt1=1-cos(theta1);
    vt2=1-cos(theta2);
    vt3=1-cos(theta3);
    st1=sin(theta1);
    st2=sin(theta2);
    st3=sin(theta3);
    ct1=cos(theta1);
    ct2=cos(theta2);
    ct3=cos(theta3);

% Define Bi's %

 b1=[(uu1(1)^2*vt1+ct1)*rr1(1)+(uu1(1)*uu1(3)*vt1)*rr1(3)+oo1(1),...
   -((uu1(3)*st1)*rr1(1)-(uu1(1)*st1)*rr1(3)),...
    (uu1(1)*uu1(3)*vt1)*rr1(1)+(uu1(3)^2*vt1+ct1)*rr1(3)+oo1(3)];
 b2=[(uu2(1)^2*vt2+ct2)*rr2(1)+(uu2(1)*uu2(3)*vt2)*rr2(3)+oo2(1),...
   -((uu2(3)*st2)*rr2(1)-(uu2(1)*st2)*rr2(3)),...
    (uu2(1)*uu2(3)*vt2)*rr2(1)+(uu2(3)^2*vt2+ct2)*rr2(3)+oo2(3)];
 b3=[(uu3(1)^2*vt3+ct3)*rr3(1)+(uu3(1)*uu3(3)*vt3)*rr3(3)+oo3(1),...
   -((uu3(3)*st3)*rr3(1)-(uu3(1)*st3)*rr3(3)),...
    (uu3(1)*uu3(3)*vt3)*rr3(1)+(uu3(3)^2*vt3+ct3)*rr3(3)+oo3(3)];

% This part of the program solves for the euler angles of
% the top plane of the truss by determining the height of
% the unit cell, r, and sequentially solving for the height
% of the truss, d, and then the unit vector, U2, to the
% top plane. From this unit vector U2 the euler angles gamma
% and beta are determined.

     u0=[0 1 0];
     u1=cross1((b1-b2)',(b1-b3)');
     u1=u1/sqrt(sum(u1.*u1));
     r=(sum(b1.*u1))/(sum(u0.*u1));
     d=2*sum(u1.*u0)*r;
     u2=d/r*u1-u0;
     u2=u2/sqrt(sum(u2.*u2));
```

```
     gma(p)=atan2(-u2(1),sqrt(u2(2)^2+u2(3)^2));
     bta(p)=atan2(u2(3),u2(2));

%********* Calculate Nodal Positions ************

pe=r*(u0+u2);
R_vec(:,p)=pe';
ll=sqrt(sum(pe.*pe));

% Unit vector normal to mid-transverse plane %
u=pe/ll;

n1=oo1-uu1*(base/2);
n2=oo2+uu2*(base/2);
n3=oo2-uu2*(base/2);
n4(p,:)=b3;
n5(p,:)=b2;
n6(p,:)=b1;

rot=[1-(u2(1)^2/(1+u2(2))) u2(1) -u2(1)*u2(3)/(1+u2(2));...
      -u2(1) u2(2) -u2(3);-u2(1)*u2(3)/(1+u2(2)) u2(3) ...
      1-(u2(3)^2/(1+u2(2)))];
da=[base/2,0,xl];
db=[0,0,-2*xl];
dc=[-base/2,0,xl];

n7(p,:)=(pe'+rot*db')';
n8(p,:)=(pe'+rot*da')';
n9(p,:)=(pe'+rot*dc')';

%******** Calculate Nodal Velocities ************

l45=l1(p);
l46=l2(p);
l56=l3(p);

p14=(n1-n4(p,:))/cl;
p24=(n2-n4(p,:))/cl;
p45=(n4(p,:)-n5(p,:))/l45;
p25=(n2-n5(p,:))/cl;
p46=(n4(p,:)-n6(p,:))/l46;
p35=(n3-n5(p,:))/cl;
p36=(n3-n6(p,:))/cl;
p56=(n5(p,:)-n6(p,:))/l56;
```

```
p16=(n1-n6(p,:))/cl;

bigP1=[-p14 0 0 0 0 0 0;-p24 0 0 0 0 0 0;p45 -p45 0 0 0;...
       0 0 0 -p25 0 0 0;p46 0 0 0 -p46;0 0 0 -p35 0 0 0;...
       0 0 0 p56 -p56;0 0 0 0 0 0 -p36;0 0 0 0 0 0 -p16];

p47=(n4(p,:)-n7(p,:))/cl;
p57=(n5(p,:)-n7(p,:))/cl;
p78=(n7(p,:)-n8(p,:))/base;
p58=(n5(p,:)-n8(p,:))/cl;
p89=(n8(p,:)-n9(p,:))/base;
p68=(n6(p,:)-n8(p,:))/cl;
p69=(n6(p,:)-n9(p,:))/cl;
p49=(n4(p,:)-n9(p,:))/cl;
p79=(n7(p,:)-n9(p,:))/base;

bigR2=[p47 zeros(1,6);0 0 0 p57 0 0 0;zeros(1,9);0 0...
     0 p58 0 0 0;zeros(1,9);zeros(1,6) p68; zeros(1,9);...
     zeros(1,6) p69;p49 zeros(1,6)];
bigP2=[-p47 zeros(1,6);-p57 zeros(1,6);p78 -p78 0 0 0;...
     0 0 0 -p58 0 0 0;p79 0 0 0 -p79 ;0 0 0 -p68 0...
     0 0; 0 0 0 p89 -p89 ;zeros(1,6) -p69;zeros(1,6) -p49];

%  Find nodal velocities

l45d=l1d(p);
l46d=l2d(p);
l56d=l3d(p);


ND456=inv(bigP1)*[0 0 l45d 0 l46d 0 l56d 0 0]';
ND789=-inv(bigP2)*(bigR2*ND456);

nd4(:,p)=ND456(1:3);
nd5(:,p)=ND456(4:6);
nd6(:,p)=ND456(7:9);
nd7(:,p)=ND789(1:3);
nd8(:,p)=ND789(4:6);
nd9(:,p)=ND789(7:9);

ped_x(p)=(nd7(1,p)+nd8(1,p)+nd9(1,p))/3;
ped_y(p)=(nd7(2,p)+nd8(2,p)+nd9(2,p))/3;
ped_z(p)=(nd7(3,p)+nd8(3,p)+nd9(3,p))/3;
```

```
% Find euler angular velocities of the top plane
r79=n7(p,:)-n9(p,:);
r78=n7(p,:)-n8(p,:);

rtop1=[0 r78(3) -r78(2);-r78(3) 0 r78(1);0, r79(3),-r79(2)];
rtop2=[0 r78(3) -r78(2);-r78(3) 0 r78(1);-r79(3), 0, r79(1)];
rtop3=[0 r78(3) -r78(2);-r78(3) 0 r78(1);r79(2), -r79(1),0];
omega(:,p)=inv(rtop2)*[nd7(1:2,p)-nd8(1:2,p);nd7(2,p)-nd9(2,p)];

btad(p)=omega(1,p);
gmad(p)=omega(3,p);
alphad(p)=omega(2,p);

%********** Calculate Nodal Accelerations ***************

q14=sum(nd4(:,p).*nd4(:,p))/cl;
q24=sum(nd4(:,p).*nd4(:,p))/cl;
q45=(sum((nd4(:,p)-nd5(:,p)).*(nd4(:,p)-nd5(:,p)))-l45d^2)/l45;
q25=sum(nd5(:,p).*nd5(:,p))/cl;
q46=(sum((nd4(:,p)-nd6(:,p)).*(nd4(:,p)-nd6(:,p)))-l46d^2)/l46;
q35=sum(nd5(:,p).*nd5(:,p))/cl;
q56=(sum((nd5(:,p)-nd6(:,p)).*(nd5(:,p)-nd6(:,p)))-l56d^2)/l56;
q36=sum(nd6(:,p).*nd6(:,p))/cl;
q16=sum(nd6(:,p).*nd6(:,p))/cl;

Q1=[q14 q24 q45 q25 q46 q35 q56 q36 q16]';

q47=(sum((nd4(:,p)-nd7(:,p)).*(nd4(:,p)-nd7(:,p))))/cl;
q57=(sum((nd5(:,p)-nd7(:,p)).*(nd5(:,p)-nd7(:,p))))/cl;
q78=(sum((nd7(:,p)-nd8(:,p)).*(nd7(:,p)-nd8(:,p))))/base;
q58=(sum((nd5(:,p)-nd8(:,p)).*(nd5(:,p)-nd8(:,p))))/cl;
q79=(sum((nd7(:,p)-nd9(:,p)).*(nd7(:,p)-nd9(:,p))))/base;
q68=(sum((nd6(:,p)-nd8(:,p)).*(nd6(:,p)-nd8(:,p))))/cl;
q89=(sum((nd8(:,p)-nd9(:,p)).*(nd8(:,p)-nd9(:,p))))/base;
q69=(sum((nd6(:,p)-nd9(:,p)).*(nd6(:,p)-nd9(:,p))))/cl;
q49=(sum((nd4(:,p)-nd9(:,p)).*(nd4(:,p)-nd9(:,p))))/cl;

Q2=[q47 q57 q78 q58 q79 q68 q89 q69 q49]';

l45dd=l1dd(p);
l46dd=l2dd(p);
l56dd=l3dd(p);

NDD456=inv(bigP1)*([0 0 l45dd 0 l46dd 0 l56dd 0 0]'-Q1);
```

```
NDD789=inv(bigP2)*(-Q2-bigR2*NDD456);

ndd4(:,p)=NDD456(1:3);
ndd5(:,p)=NDD456(4:6);
ndd6(:,p)=NDD456(7:9);
ndd7(:,p)=NDD789(1:3);
ndd8(:,p)=NDD789(4:6);
ndd9(:,p)=NDD789(7:9);

pedd_x(p)=(ndd7(1,p)+ndd8(1,p)+ndd9(1,p))/3;
pedd_y(p)=(ndd7(2,p)+ndd8(2,p)+ndd9(2,p))/3;
pedd_z(p)=(ndd7(3,p)+ndd8(3,p)+ndd9(3,p))/3;
arow1=ndd7(1,p)-ndd8(1,p)-(alphad(p)*(btad(p)*r78(2)...
   -alphad(p)*r78(1))-gmad(p)*(gmad(p)*r78(1)-btad(p)*r78(3)));
arow2=ndd7(2,p)-ndd8(2,p)-(gmad(p)*(alphad(p)*r78(3)...
   -gmad(p)*r78(2))-btad(p)*(btad(p)*r78(2)-alphad(p)*r78(1)));
arow3=ndd7(2,p)-ndd9(2,p)-(gmad(p)*(alphad(p)*r79(3)...
   -gmad(p)*r79(2))-btad(p)*(btad(p)*r79(2)-alphad(p)*r79(1)));
omegad(:,p)=inv(rtop2)*[arow1 arow2 arow3]';

btadd(p)=omegad(1,p);
gmadd(p)=omegad(3,p);
alphadd(p)=omegad(2,p);

   end
end
```

```
%-----------------------------------------------------
%      PROGRAM: MOTPRO1
%        Written by: T. Warrington
%
%  This program calculates the link profile for an
%  exponential input voltage profile. It also calculates
%  the link velocities and link accelerations.
%-----------------------------------------------------
linkst=linkstrt(base,cl,d);
lnot1=linkst(1);
lnot2=linkst(2);
lnot3=linkst(3);
links=linkfnd1(base,cl,d);

% These are the required link positions to attain the specified
% truss orientation
l1ref=links(1);
l2ref=links(2);
l3ref=links(3);

%  define constants and motor variables
gl= 0.0159*2.54/100; % gain of lead screw(m/rad)
gr=1/5.1;        % gear reduction
ka=1.8;          % amplifier gain(A/V)
kt=0.1;          % torque constant(N-m/A)
J=1.22e-4;       % armature inertia(kg-m^2)
tm=.0164;        % time constant of motor(sec)
Vp=5;            % maximum input voltage(volts)
l0=1.412;        % nominal length position(meters)
Tfric=.6;        % motor friction(N-m)
thetamx=400;     % max velocity of motor (rad/s)

%  determine acceleration profile of link1
te=input('Enter total simulation time: ');
dt=input('Enter time step: ');
t=0:dt:te;
nn=te/dt;
c1=kt*ka*gl*gr/J;
A=[0 1;0 0];
B=[0;c1];
%  Now set up eigenvalues based on simulation time
zeta=0.8;
ts=.9*te;
wn=6/(zeta*ts);
```

```
rroots=[-wn*zeta+wn*sin(acos(zeta))*j;-wn*zeta-...
       wn*sin(acos(zeta))*j];
K=place(A,B,rroots);
cls
disp(' ')
disp(' ')
disp('            Calculating Forward Kinematics')
disp(' ')
disp('                  Please Wait')

[phi,gamma]=c2d(A,B,dt);
l1dd(1)=0;
x1(:,1)=[lnot1;0];
for j=1:nn
   x1(:,j+1)=phi*x1(:,j)-gamma*K*(x1(:,j)-[l1ref;0]);
   l1dd(j+1)=-c1*K*(x1(:,j)-[l1ref;0]);
end

l2dd(1)=0;
x2(:,1)=[lnot2;0];
for j=1:nn
   x2(:,j+1)=phi*x2(:,j)-gamma*K*(x2(:,j)-[l2ref;0]);
   l2dd(j+1)=-c1*K*(x2(:,j)-[l2ref;0]);
end

l3dd(1)=0;
x3(:,1)=[lnot3;0];
for j=1:nn
   x3(:,j+1)=phi*x3(:,j)-gamma*K*(x3(:,j)-[l3ref;0]);
   l3dd(j+1)=-c1*K*(x3(:,j)-[l3ref;0]);
end

x1=x1';
x2=x2';
x3=x3';
l1(:)=x1(:,1);
l1d(:)=x1(:,2);
l2(:)=x2(:,1);
l2d(:)=x2(:,2);
l3(:)=x3(:,1);
l3d(:)=x3(:,2);
l1dd=l1dd';
l2dd=l2dd';
l3dd=l3dd';
```

```
save lnk_inf l1 l1d l1dd l2 l2d l2dd l3 l3d l3dd
end
```

```
%-------------------------------------------------
%   PROGRAM: LINKSTRT.M
%       Written by: D. Lacy
%
%    This program will find the beginning link lengths
%    for the truss positioned at the equilibrium point
%
%-------------------------------------------------
function[Links]=linkstrt(base,cl,d)
% Define fixed dimensions of truss %

x=tan(.52359)*base/2;
rim=sqrt(cl^2-(base/2)^2);
gamma=0;
beta=0;

% Create unit vectors on fixed plane(0) and moving plane(2) %
u0=[0,1,0];
u2=[-sin(gamma), cos(beta)*cos(gamma), sin(beta)*cos(gamma)];
r=d/(sqrt(2*(1+sum(u0.*u2))));

% Vector describing origin of top plate %
pe=r*(u0+u2);

% Unit vector normal to mid-transverse plane %
l=sqrt(sum(pe.*pe));
u=pe/l;

% Define fixed plane-vectors %
uu1=[1,0,0];
uu2=[-.5,0,-.866];
uu3=[-.5,0,.866];
oo1=[0,0,x];
oo2=[.866*x,0,-x/2];
oo3=[-.866*x,0,-x/2];
rr1=rim*oo1/sqrt(sum(oo1.*oo1));
rr2=rim*oo2/sqrt(sum(oo2.*oo2));
rr3=rim*oo3/sqrt(sum(oo3.*oo3));

% Define k's %

k11=(rr1(1)-uu1(1)^2*rr1(1)-uu1(1)*uu1(3)*rr1(3))*u(1)+...
    (rr1(3)-uu1(3)^2*rr1(3)-uu1(1)*uu1(3)*rr1(1))*u(3);
k12=(uu1(3)*rr1(1)-uu1(1)*rr1(3))*u(2);
```

```
k13=(uu1(1)^2*rr1(1)+uu1(1)*uu1(3)*rr1(3)+oo1(1))*u(1)+...
    (uu1(3)^2*rr1(3)+uu1(3)*uu1(1)*rr1(1)+oo1(3))*u(3)-1/2;
k21=(rr2(1)-uu2(1)^2*rr2(1)-uu2(1)*uu2(3)*rr2(3))*u(1)+...
    (rr2(3)-uu2(3)^2*rr2(3)-uu2(1)*uu2(3)*rr2(1))*u(3);
k22=(uu2(3)*rr2(1)-uu2(1)*rr2(3))*u(2);
k23=(uu2(1)^2*rr2(1)+uu2(1)*uu2(3)*rr2(3)+oo2(1))*u(1)+...
    (uu2(3)^2*rr2(3)+uu2(3)*uu2(1)*rr2(1)+oo2(3))*u(3)-1/2;
k31=(rr3(1)-uu3(1)^2*rr3(1)-uu3(1)*uu3(3)*rr3(3))*u(1)+...
    (rr3(3)-uu3(3)^2*rr3(3)-uu3(1)*uu3(3)*rr3(1))*u(3);
k32=(uu3(3)*rr3(1)-uu3(1)*rr3(3))*u(2);
k33=(uu3(1)^2*rr3(1)+uu3(1)*uu3(3)*rr3(3)+oo3(1))*u(1)+...
    (uu3(3)^2*rr3(3)+uu3(3)*uu3(1)*rr3(1)+oo3(3))*u(3)-1/2;

% Define ti's %

t1=-(-k12-sqrt(k12^2-k13^2+k11^2))/(k13-k11);
t2=-(-k22-sqrt(k22^2-k23^2+k21^2))/(k23-k21);
t3=-(-k32-sqrt(k32^2-k33^2+k31^2))/(k33-k31);

% Define the theta's %

theta1=2*atan(t1);
theta2=2*atan(t2);
theta3=2*atan(t3);

% Define trig id's %

vt1=1-cos(theta1);
vt2=1-cos(theta2);
vt3=1-cos(theta3);

st1=sin(theta1);
st2=sin(theta2);
st3=sin(theta3);

ct1=cos(theta1);
ct2=cos(theta2);
ct3=cos(theta3);

% Define Bi's %

b1=[(uu1(1)^2*vt1+ct1)*rr1(1)+(uu1(1)*uu1(3)*vt1)*rr1(3)+oo1(1),...
 -((uu1(3)*st1)*rr1(1)-(uu1(1)*st1)*rr1(3)),...
  (uu1(1)*uu1(3)*vt1)*rr1(1)+(uu1(3)^2*vt1+ct1)*rr1(3)+oo1(3)];
```

```
b2=[(uu2(1)^2*vt2+ct2)*rr2(1)+(uu2(1)*uu2(3)*vt2)*rr2(3)+oo2(1),...
 -((uu2(3)*st2)*rr2(1)-(uu2(1)*st2)*rr2(3)),...
  (uu2(1)*uu2(3)*vt2)*rr2(1)+(uu2(3)^2*vt2+ct2)*rr2(3)+oo2(3)];
b3=[(uu3(1)^2*vt3+ct3)*rr3(1)+(uu3(1)*uu3(3)*vt3)*rr3(3)+oo3(1),...
 -((uu3(3)*st3)*rr3(1)-(uu3(1)*st3)*rr3(3)),...
  (uu3(1)*uu3(3)*vt3)*rr3(1)+(uu3(3)^2*vt3+ct3)*rr3(3)+oo3(3)];
bd1=b2-b3;
bd2=b3-b1;
bd3=b1-b2;

% Obtain link lengths %

l1=sqrt(sum(bd1.*bd1));
l2=sqrt(sum(bd2.*bd2));
l3=sqrt(sum(bd3.*bd3));

Links=[l1 l2 l3]';
end
```

```
%-------------------------------------------
%     PROGRAM: KINANG1.M
%       Written by: T. Warrington
%
%  This program solves for the thetas(theta) of the
%  three "kinematically equivalent" links given the
%  desired link lengths, L,(3X1) and an initial guess
%  at the thetas, theta,(3x1).
%-------------------------------------------
 function[theta]=kinang1(L,theta,base,cl)

error=1;
 while error>0.01;

   l=link1(theta,base,cl);
   err=L-l;
   err1=L(1,1)-l(1,1);
   err2=L(2,1)-l(2,1);
   err3=L(3,1)-l(3,1);
   error=abs(err1)+abs(err2)+abs(err3);

   p=part1(theta,base,cl);
   theta=theta-p*err;

   if theta(1,1)>pi
      theta(1,1)=theta(1,1)-pi;
     elseif theta(1,1)<0
      theta(1,1)=theta(1,1)+pi;
   end

   if theta(2,1)>pi
      theta(2,1)=theta(2,1)-pi;
     elseif theta(2,1)<0
      theta(2,1)=theta(2,1)+pi;
   end

   if theta(3,1)>pi
      theta(3,1)=theta(3,1)-pi;
     elseif theta(3,1)<0
      theta(3,1)=theta(3,1)+pi;
   end
end
```

```
%-----------------------------------------------
%      PROGRAM: LINKFND1.M
%        Written by: T. Warrington
%
%     This program finds the link lengths for the end
%       angle of theta
%-----------------------------------------------

x=tan(.52359)*base/2;
rim=sqrt(cl^2-(base/2)^2);
beta=input('  Enter the angle of rotation about
                  the x-axis, beta, in degrees  ')
gamma=input(' Enter the angle of rotation about
                  the z-axis, gamma, in degrees  ')
gamma=gamma*pi/180;
beta=beta*pi/180;

% Create unit vectors on fixed plane(0) and moving plane(2) %
u0=[0,1,0];
u2=[-sin(gamma), cos(beta)*cos(gamma), sin(beta)*cos(gamma)];
r=d/(sqrt(2*(1+sum(u0.*u2))));

% Vector describing origin of top plate %
pe=r*(u0+u2);

% Unit vector normal to mid-transverse plane %
l=sqrt(sum(pe.*pe));
u=pe/l;

% Define fixed plane-vectors %
uu1=[1,0,0];
uu2=[-.5,0,-.866];
uu3=[-.5,0,.866];
oo1=[0,0,x];
oo2=[.866*x,0,-x/2];
oo3=[-.866*x,0,-x/2];
rr1=rim*oo1/sqrt(sum(oo1.*oo1));
rr2=rim*oo2/sqrt(sum(oo2.*oo2));
rr3=rim*oo3/sqrt(sum(oo3.*oo3));

% Define k's %

k11=(rr1(1)-uu1(1)^2*rr1(1)-uu1(1)*uu1(3)*rr1(3))*u(1)+...
    (rr1(3)-uu1(3)^2*rr1(3)-uu1(1)*uu1(3)*rr1(1))*u(3);
```

```
k12=(uu1(3)*rr1(1)-uu1(1)*rr1(3))*u(2);
k13=(uu1(1)^2*rr1(1)+uu1(1)*uu1(3)*rr1(3)+oo1(1))*u(1)+...
   (uu1(3)^2*rr1(3)+uu1(3)*uu1(1)*rr1(1)+oo1(3))*u(3)-1/2;
k21=(rr2(1)-uu2(1)^2*rr2(1)-uu2(1)*uu2(3)*rr2(3))*u(1)+...
    (rr2(3)-uu2(3)^2*rr2(3)-uu2(1)*uu2(3)*rr2(1))*u(3);
k22=(uu2(3)*rr2(1)-uu2(1)*rr2(3))*u(2);
k23=(uu2(1)^2*rr2(1)+uu2(1)*uu2(3)*rr2(3)+oo2(1))*u(1)+...
    (uu2(3)^2*rr2(3)+uu2(3)*uu2(1)*rr2(1)+oo2(3))*u(3)-1/2;
k31=(rr3(1)-uu3(1)^2*rr3(1)-uu3(1)*uu3(3)*rr3(3))*u(1)+...
    (rr3(3)-uu3(3)^2*rr3(3)-uu3(1)*uu3(3)*rr3(1))*u(3);
k32=(uu3(3)*rr3(1)-uu3(1)*rr3(3))*u(2);
k33=(uu3(1)^2*rr3(1)+uu3(1)*uu3(3)*rr3(3)+oo3(1))*u(1)+...
    (uu3(3)^2*rr3(3)+uu3(3)*uu3(1)*rr3(1)+oo3(3))*u(3)-1/2;

% Define ti's %

t1=-(-k12-sqrt(k12^2-k13^2+k11^2))/(k13-k11);
t2=-(-k22-sqrt(k22^2-k23^2+k21^2))/(k23-k21);
t3=-(-k32-sqrt(k32^2-k33^2+k31^2))/(k33-k31);

% Define the theta's %

theta1=2*atan(t1);
theta2=2*atan(t2);
theta3=2*atan(t3);

% Define trig id's %
vt1=1-cos(theta1);
vt2=1-cos(theta2);
vt3=1-cos(theta3);
st1=sin(theta1);
st2=sin(theta2);
st3=sin(theta3);
ct1=cos(theta1);
ct2=cos(theta2);
ct3=cos(theta3);

% Define Bi's %
b1=[(uu1(1)^2*vt1+ct1)*rr1(1)+(uu1(1)*uu1(3)*vt1)*rr1(3)+oo1(1),...
   -((uu1(3)*st1)*rr1(1)-(uu1(1)*st1)*rr1(3)),...
   (uu1(1)*uu1(3)*vt1)*rr1(1)+(uu1(3)^2*vt1+ct1)*rr1(3)+oo1(3)];
b2=[(uu2(1)^2*vt2+ct2)*rr2(1)+(uu2(1)*uu2(3)*vt2)*rr2(3)+oo2(1),...
   -((uu2(3)*st2)*rr2(1)-(uu2(1)*st2)*rr2(3)),...
   (uu2(1)*uu2(3)*vt2)*rr2(1)+(uu2(3)^2*vt2+ct2)*rr2(3)+oo2(3)];
```

```
b3=[(uu3(1)^2*vt3+ct3)*rr3(1)+(uu3(1)*uu3(3)*vt3)*rr3(3)+oo3(1),...
   -((uu3(3)*st3)*rr3(1)-(uu3(1)*st3)*rr3(3)),...
   (uu3(1)*uu3(3)*vt3)*rr3(1)+(uu3(3)^2*vt3+ct3)*rr3(3)+oo3(3)];
bd1=b2-b3;
bd2=b3-b1;
bd3=b1-b2;

% Obtain link lengths %

l1=sqrt(sum(bd1.*bd1));
l2=sqrt(sum(bd2.*bd2));
l3=sqrt(sum(bd3.*bd3));
Links=[l1 l2 l3]';
end
```

```
%-------------------------------------------------
%      PROGRAM: LINK1.M
%         Written by: T. Warrington
%
%  This program solves for the link lengths, length(1X3),
%  for inputs of the three thetas, theta(3x1).
%
%-------------------------------------------------
function[length]=link1(theta,base,cl)
r=sqrt(3)/2*base;          % height of base triangle(m)
dd=sqrt(cl^2-(base/2)^2);% length of "kin. equiv." link(m)

t1=theta(1,1);
t2=theta(2,1);
t3=theta(3,1);

p4=[base/4+sqrt(3)/2*dd*cos(t2),-1/6*r-1/2*dd*cos(t2),dd*sin(t2)];
p5=[-base/4-sqrt(3)/2*dd*cos(t3),-1/6*r-1/2*dd*cos(t3),dd*sin(t3)];
p6=[0,1/3*r+dd*cos(t1),dd*sin(t1)];

l1=sqrt((p4(1,1)-p5(1,1))^2+(p4(1,2)-p5(1,2))^2+(p4(1,3)-p5(1,3))^2);
l2=sqrt((p5(1,1)-p6(1,1))^2+(p5(1,2)-p6(1,2))^2+(p5(1,3)-p6(1,3))^2);
l3=sqrt((p6(1,1)-p4(1,1))^2+(p6(1,2)-p4(1,2))^2+(p6(1,3)-p4(1,3))^2);
length=[l1;l2;l3];

end
```

```
%-----------------------------------------------
%    PROGRAM: PART1.M
%      Written by: T. Warrington
%
%   This program calculates the Jacobian of the actruated link lengths
%   to the thetas(partial of theta wrt L). It calculates the partials at
%   the current operating point.
%-----------------------------------------------
function[part]=part1(theta,base,cl)
l=link1(theta,base,cl);
dt=.05;
dtheta1=theta+[dt; 0; 0];
lt1=link1(dtheta1,base,cl);
part(:,1)=(lt1-l)/dt;
dtheta2=theta+[0; dt; 0];
lt2=link1(dtheta2,base,cl);
part(:,2)=(lt2-l)/dt;
dtheta3=theta+[0; 0; dt];
lt3=link1(dtheta3,base,cl);
part(:,3)=(lt3-l)/dt;
part=-inv(part);
end
```

```
%-----------------------------------------------
%    PROGRAM: CROSS.M
%       Written by: D. Lacy
%
%     This program evaluates the cross product of two
%      vectors
%-----------------------------------------------
function Y = cross(data1,data2)
crossmat=[0 -data1(3) data1(2);data1(3) 0 -data1(1);-data1(2) data1(1) 0];
Y=(crossmat*data2')';



%-----------------------------------------------
%    PROGRAM: CROSS1.M
%       Written by: T. Warrington
%
%     This program evaluates the cross product of two
%      vectors
%-----------------------------------------------
function[v]=cross(p,q)
v(1)=p(2)*q(3)-p(3)*q(2);
v(2)=p(3)*q(1)-p(1)*q(3);
v(3)=p(1)*q(2)-p(2)*q(1);
end

æ
```

Appendix D

Spatial Truss Finite-Element Model

```
%-----------------------------------------------------
%
%        MSC/PAL Model File
%
%-----------------------------------------------------
Title    Spatial Truss - Equilibrium Position
Nodal Point Locations 1
1  -0.6096   -0.3519         0
2   0.6096   -0.3519         0
3  -0.0000    0.7039         0
4  -0.5842    0.3373    0.5609
5   0.5842    0.3373    0.5609
6        0   -0.6746    0.5609
7  -0.6096   -0.3519    1.1218
8        0    0.7039    1.1218
9   0.6096   -0.3519    1.1218
-- Blank Line --
Material Properties 71.e9  26.2e9  2711.5
Beam Type 1, 9.0245e-5
Connect 2 to 6
Connect 1 to 6
Connect 1 to 4
Connect 3 to 4
Connect 3 to 5
Connect 2 to 5
Connect 4 to 5
Connect 5 to 6
Connect 4 to 6
Connect 6 to 9
Connect 6 to 7
Connect 4 to 7
Connect 4 to 8
```

```
Connect 5 to 8
Connect 5 to 9
Connect 7 to 9
Connect 8 to 7
Connect 8 to 9
zero 1
ra of all
-- Blank Line --
End Definition
```

```
%-------------------------------------------------
%
%        MSC/PAL Loading File
%
%-------------------------------------------------
displacements applied 1
tx 0 1 2 3
ty 0 1 2 3
tz 0 1 2 3
--blank line--
forces and moments applied 1
fz 46 7 8 9
--blank line--
solve
```

VITA

The author was born in Richmond, VA on December 6, 1959 and grew up in Ashland, VA. He received his Bachelor of Science degree in Mechanical Engineering from Virginia Polytechnic Institute and State University in June, 1984. After travelling throughout the United States and Europe for a year, he settled down and took an engineering job in northern Virginia. In 1988, he was married to Joyce Kolb. He returned to Virgina Tech to pursue a Master of Science in Mechanical Engineering in 1989. This thesis represents the culmination of that program. He now returns to the world of the practicing engineer, where he will work as a mechanical design engineer for Allied-Signal, Inc.