

INFORMATION TO USERS

This manuscript has been reproduced from the microfilm master. UMI films the text directly from the original or copy submitted. Thus, some thesis and dissertation copies are in typewriter face, while others may be from any type of computer printer.

The quality of this reproduction is dependent upon the quality of the copy submitted. Broken or indistinct print, colored or poor quality illustrations and photographs, print bleedthrough, substandard margins, and improper alignment can adversely affect reproduction.

In the unlikely event that the author did not send UMI a complete manuscript and there are missing pages, these will be noted. Also, if unauthorized copyright material had to be removed, a note will indicate the deletion.

Oversize materials (e.g., maps, drawings, charts) are reproduced by sectioning the original, beginning at the upper left-hand corner and continuing from left to right in equal sections with small overlaps. Each original is also photographed in one exposure and is included in reduced form at the back of the book.

Photographs included in the original manuscript have been reproduced xerographically in this copy. Higher quality 6" x 9" black and white photographic prints are available for any photographs or illustrations appearing in this copy for an additional charge. Contact UMI directly to order.

U·M·I

University Microfilms International
A Bell & Howell Information Company
300 North Zeeb Road, Ann Arbor, MI 48106-1346 USA
313/761-4700 800/521-0600

Order Number 9321229

Dynamic modeling of multibody flexible structures

Xu, Jiechi, Ph.D.

Iowa State University, 1993

U·M·I
300 N. Zeeb Rd.
Ann Arbor, MI 48106

Dynamic modeling of multibody flexible structures

by

Jiechi Xu

A Dissertation Submitted to the
Graduate Faculty in Partial Fulfillment of the
Requirements for the Degree of
DOCTOR OF PHILOSOPHY

Department: Mechanical Engineering
Major: Mechanical Engineering

Approved:

Signature was redacted for privacy.

In Charge of Major Work

Signature was redacted for privacy.

For the Major Department

Signature was redacted for privacy.

For the Graduate College

Iowa State University
Ames, Iowa
1993

Copyright © Jiechi Xu, 1993. All rights reserved.

TABLE OF CONTENTS

ACKNOWLEDGMENTS	xvi
CHAPTER 1. GENERAL INTRODUCTION	1
1.1 Overview of Research History	2
1.2 Analysis of Structure Vibration	11
1.3 Stability Study of a Rigid Body System	12
1.4 Dynamic Modeling of a Flexible System	13
CHAPTER 2. FUNDAMENTAL STRUCTURE ANALYSIS	17
2.1 Analysis of Inertial Forces	19
2.2 Structural Vibration Analysis	23
2.2.1 Structure configuration and material properties	25
2.2.2 Finite element model	26
2.2.3 Angular velocity profile and inertial force	26
2.2.4 NASTRAN run	29
2.2.5 Estimation of elastic deformation	31
CHAPTER 3. MODELING OF FLEXIBLE BODIES	37
3.1 Introduction	37
3.2 Previous Work Review	38
3.3 Current Approach	44

3.4	Finite Element Modeling	47
3.4.1	Displacement function	48
3.4.2	Geometric boundary conditions	48
3.4.3	Structural stiffness matrix	51
3.5	Local Level Motion Equations	53
3.5.1	Position and velocity vectors	55
3.5.2	Velocity squared term	59
3.5.3	Kinetic energy	61
3.5.4	Potential energy	62
3.5.5	Motion equations	64
3.5.6	Derivation of $\frac{1}{2}\partial(\mathbf{m}_i\dot{\mathbf{q}}_i)/\partial\mathbf{q}_i^T$	64
3.6	Global Level Motion Equations	68
3.6.1	Compatibility matrix	69
3.6.2	Assembly of motion equations	69
3.7	Summary	70
CHAPTER 4. MODELING OF RIGID BODIES		73
4.1	Introduction	73
4.2	Coordinate Systems	74
4.2.1	Three successive rotating angles	77
4.2.2	Rigid body angular velocity	79
4.3	Tank Dynamics	80
4.3.1	Geometric constraints	81
4.3.2	Tank position and velocity vectors	84
4.3.3	Tank angular velocity	85

4.3.4	Tank kinetic energy	87
4.3.5	Tank potential energy	88
4.3.6	Tank equations of motion	88
4.4	Rigid Beam Dynamics	89
4.4.1	Rigid beam velocity vector	89
4.4.2	Inertia dyadic and angular velocity	91
4.4.3	Rigid beam equations of motion	92
4.5	Dynamics of a Bar-Shaft Assembly	92
4.6	System Equations of Motion	93
4.6.1	Generalized global and local coordinates	94
4.6.2	Compatibility matrices	95
4.6.3	Assembly of equations	97
CHAPTER 5. DEVELOPMENT OF NUMERICAL ALGORITHMS		98
5.1	Introduction	99
5.2	Current Approach	101
5.3	Dynamic Equations	102
5.3.1	System mapping	103
5.3.2	Subsystem partition	105
5.4	Algorithm Development	107
5.4.1	Implicit phase	108
5.4.2	Explicit phase	111
5.5	Illustration of Numerical Results	112
5.6	Summary	118

CHAPTER 6. SIMULATION AND MEASUREMENT RESULTS	120
6.1 Run Case 1: Stable Spin-up with Collar Up	127
6.2 Run Case 2: Stable Spin-up with Collar Down	128
6.3 Run Case 3: General Motion with Initial Tilt	129
6.4 Run Case 4: General Motion with Excitation	131
6.5 Run Case 5: Fluid-structure Interaction	133
6.6 Experimental Measurements	135
CHAPTER 7. GENERAL CONCLUSIONS	206
REFERENCES	210
APPENDIX A. TRANSFORMATION MATRICES	220
APPENDIX B. SKEW-SYMMETRIC MATRICES	223
B.1 Skew-Symmetric Matrix	223
B.2 Special Matrix Operators	224
B.3 Matrix Partial Derivatives	224
APPENDIX C. CONSTANT MATRICES DEFINED IN CHAPTER 3	226
APPENDIX D. COMPUTER SIMULATION PROGRAM	228

LIST OF TABLES

Table 2.1:	Material and geometry properties of each individual beam . .	25
Table 2.2:	Computed real eigenvalues of the finite element model using NASTRAN normal modes solution	30
Table 6.1:	List of cross-sectional shape and size	121
Table 6.2:	List of cross-sectional area	121
Table 6.3:	List of link length	122
Table 6.4:	List of mass density	122
Table 6.5:	List of Young's modulus	123
Table 6.6:	List of second moment of area	124
Table 6.7:	Case 1 conditions and parameters	127
Table 6.8:	Case 2 conditions and parameters	129
Table 6.9:	Case 3 conditions and parameters	130
Table 6.10:	Case 4 conditions and parameters	132
Table 6.11:	Case 5 conditions and parameters	134

LIST OF FIGURES

Figure 1.1:	Flow chart of satellite project research history and personnel	3
Figure 1.2:	A first working version of the satellite simulator	5
Figure 1.3:	A modified final version of the satellite test rig	7
Figure 2.1:	Accelerations in the x - z plane	20
Figure 2.2:	Accelerations in the x - y plane	21
Figure 2.3:	Accelerations in the y - z plane	22
Figure 2.4:	A schematic drawing of structure configuration (dynamic part only)	24
Figure 2.5:	Finite element model of an axisymmetric half of the upper assembly	27
Figure 2.6:	A set of sinusoidal spin-up profiles: The solid line denotes spin-up velocity, $\dot{\lambda}_3(= \omega)$; The dashed line denotes spin-up acceleration, $\ddot{\lambda}_3(= \dot{\omega})$	28
Figure 2.7:	Translational displacements at node 41 (from NASTRAN run)	31
Figure 2.8:	Angular displacements at node 41 (from NASTRAN run) . .	32
Figure 2.9:	Translational velocity at node 41 (from NASTRAN run) . . .	32
Figure 2.10:	Angular velocity at node 41 (from NASTRAN run)	33
Figure 2.11:	A cantilever beam model of beam 1	34

Figure 3.1:	Functions of transverse deflection of a cantilever beam for three cases with different external loadings	49
Figure 3.2:	Sign conventions of nodal displacements	50
Figure 3.3:	Sign conventions of nodal forces	52
Figure 3.4:	Coordinate systems and elastic deformation for a generic g^{th} element in an arbitrary i^{th} flexible beam	54
Figure 4.1:	A schematic drawing of the test rig (dynamic part only) . . .	75
Figure 4.2:	Coordinate systems of the test rig	76
Figure 4.3:	Three successive rotating angles between the inertial and moving coordinates	78
Figure 4.4:	Tank assembly and its associated structures	82
Figure 4.5:	Schematic of a rigid beam	90
Figure 5.1:	Case 1: Response comparison of a rigid body model with a flexible model using initial tilt angle of $\lambda_1 = 1$ degree and base time of $t_0 = 3$ seconds	114
Figure 5.2:	Case 2: Confirmation of “numerical damping” effect with $\gamma = 0.6$, $\beta = 0.303$ and $\gamma = 0.5$, $\beta = 0.25$ by applying an impulse of $1N$ as an exciting force	116
Figure 6.1:	Lower shaft spin-up profiles where the solid line denotes spin velocity, $\dot{\lambda}_3 (= \omega)$, and the dashed line denotes spin acceleration, $\ddot{\lambda}_3 (= \dot{\omega})$	125
Figure 6.2:	A test rig schematic drawing	126

Figure 6.3:	Trajectory of a tank center relative to an inertial frame during spin-up with collar up: $t_0 = 3$ seconds, $\omega = 60rpm$	138
Figure 6.4:	Trace of X and Z inertial coordinates of a tank center in the first one second during spin-up with collar up: $t_0 = 3$ seconds, $\omega = 60rpm$	139
Figure 6.5:	Time history of tank center velocity relative to an inertial frame during spin-up with collar up: $t_0 = 3$ seconds, $\omega = 60rpm$	140
Figure 6.6:	Time history of tank center acceleration relative to an inertial frame during spin-up with collar up: $t_0 = 3$ seconds, $\omega = 60rpm$	141
Figure 6.7:	Time history of bending deformation of beams 1 and 2 in the corresponding local $x - y$ and $y - z$ planes during spin-up with collar up: $t_0 = 3$ seconds, $\omega = 60rpm$	142
Figure 6.8:	Time history of bending deflections of beams 3 and 4 in the local $x - z$ plane during spin-up with collar up: $t_0 = 3$ seconds, $\omega = 60rpm$	144
Figure 6.9:	Time history of bending deformation of beams 1 and 2 in the corresponding local $x - z$ planes during spin-up with collar up: $t_0 = 3$ seconds, $\omega = 60rpm$	146
Figure 6.10:	Time history of rigid body nutating angles with collar down and no initial tilt: $t_0 = 3$ seconds, $\omega = 60rpm$	148
Figure 6.11:	Trajectory of a tank center in an inertial frame with collar down and no initial tilt: $t_0 = 3$ seconds, $\omega = 60rpm$	150
Figure 6.12:	Time history of rigid body nutating angles with collar down and initial tilt: $t_0 = 3$ seconds, $\omega = 60rpm$, $\lambda_1 = 1$ degree . .	152

- Figure 6.13: Time history of rigid body angular velocity 1, $\dot{\lambda}_1$, with collar
down and initial tilt: $t_0 = 3$ seconds, $\omega = 60rpm$, $\lambda_1 = 1$ degree 153
- Figure 6.14: Time history of rigid body angular velocity 2, $\dot{\lambda}_2$, with collar
down and initial tilt: $t_0 = 3$ seconds, $\omega = 60rpm$, $\lambda_1 = 1$ degree 155
- Figure 6.15: Time history of rigid body angular acceleration 1, $\ddot{\lambda}_1$, with
collar down and initial tilt: $t_0 = 3$ seconds, $\omega = 60rpm$,
 $\lambda_1 = 1$ degree 156
- Figure 6.16: Time history of rigid body angular acceleration 2, $\ddot{\lambda}_2$, with
collar down and initial tilt: $t_0 = 3$ seconds, $\omega = 60rpm$,
 $\lambda_1 = 1$ degree 158
- Figure 6.17: Beam 1 tangential deflection without initial tilt or with tilt of
 $\lambda_1 = 1$ degree: collar down, $t_0 = 3$ seconds, $\omega = 60rpm$ 160
- Figure 6.18: Beam 3 tangential deflection without initial tilt or with tilt of
 $\lambda_1 = 1$ degree: collar down, $t_0 = 3$ seconds, $\omega = 60rpm$ 162
- Figure 6.19: Beam 1 vertical deflection without initial tilt or with tilt of
 $\lambda_1 = 1$ degree: collar down, $t_0 = 3$ seconds, $\omega = 60rpm$ 164
- Figure 6.20: Beam 3 radial deflection without initial tilt or with tilt of
 $\lambda_1 = 1$ degree: collar down, $t_0 = 3$ seconds, $\omega = 60rpm$ 166
- Figure 6.21: Beams 1 and 5 tangential deflections with initial tilt and collar
down: $t_0 = 3$ seconds, $\omega = 60rpm$, $\lambda_1 = 1$ degree 168
- Figure 6.22: Beams 1 and 5 vertical deflections with initial tilt and collar
down: $t_0 = 3$ seconds, $\omega = 60rpm$, $\lambda_1 = 1$ degree 170
- Figure 6.23: Beams 3 and 4 radial deflections with initial tilt and collar
down: $t_0 = 3$ seconds, $\omega = 60rpm$, $\lambda_1 = 1$ degree 172

Figure 6.24: Beams 3 and 7 tangential deflections with initial tilt and collar down: $t_0 = 3$ seconds, $\omega = 60rpm$, $\lambda_1 = 1$ degree	174
Figure 6.25: Beams 3 and 7 radial deflections with initial tilt and collar down: $t_0 = 3$ seconds, $\omega = 60rpm$, $\lambda_1 = 1$ degree	176
Figure 6.26: Time history of rigid body nutating angles with impulse of 1 Newton and no initial tilt: $t_0 = 1$ second, $\omega = 60rpm$	178
Figure 6.27: Time history of rigid body angular velocities with impulse of 1 Newton and no initial tilt: $t_0 = 1$ second, $\omega = 60rpm$. . .	180
Figure 6.28: Time history of rigid body angular accelerations with impulse of 1 Newton and no initial tilt: $t_0 = 1$ second, $\omega = 60rpm$. .	182
Figure 6.29: Trajectory of tank 1 center position in an inertial frame with impulse of 1 Newton and no initial tilt: $t_0 = 1$ second, $\omega =$ $60rpm$	184
Figure 6.30: Trajectory of tank 2 center position in an inertial frame with impulse of 1 Newton and no initial tilt: $t_0 = 1$ second, $\omega =$ $60rpm$	186
Figure 6.31: Z coordinates of tanks 1 and 2 center positions in an inertial frame with impulse of 1 Newton and no initial tilt: $t_0 = 1$ second, $\omega = 60rpm$	188
Figure 6.32: Beam local tangential deflections at distal ends with impulse of 1 Newton and no initial tilt: $t_0 = 1$ second, $\omega = 60rpm$. .	190
Figure 6.33: Beam local vertical deflections at distal ends with impulse of 1 Newton and no initial tilt: $t_0 = 1$ second, $\omega = 60rpm$. . .	192

Figure 6.34: Beam local radial deflections at distal ends with impulse of 1 Newton and no initial tilt: $t_0 = 1$ second, $\omega = 60rpm$	193
Figure 6.35: Beams 1 and 5 rotations about local Y axes at distal ends with impulse of 1 Newton and no initial tilt: $t_0 = 1$ second, $\omega = 60rpm$	195
Figure 6.36: Beams 1 and 5 rotations about local Z axes at distal ends with impulse of 1 Newton and no initial tilt: $t_0 = 1$ second, $\omega = 60rpm$	196
Figure 6.37: Beams 3 and 7 rotations about local Y axes at distal ends with impulse of 1 Newton and no initial tilt: $t_0 = 1$ second, $\omega = 60rpm$	197
Figure 6.38: Comparison of local radial deflections at the tank center for non-interactive and interactive modes with collar up: $t_0 = 0.5$ second, $\omega = 30 rpm$, $\gamma = 0.6$, $\beta = 0.303$	198
Figure 6.39: Comparison of local tangential deflections at the tank center for non-interactive and interactive modes with collar up: $t_0 = 0.5$ second, $\omega = 30rpm$, $\gamma = 0.6$, $\beta = 0.303$	201
Figure 6.40: Lower shaft spin velocity profiles: measured case and ideal case	204
Figure 6.41: Overall radial deflection at tank center for a stable spin-up case with collar up: experimental and computational results .	205

Nomenclature

Bold letters denote matrices or vectors

Overscores

$\vec{}$	Geometric vector
$\hat{}$	Cartesian frame
\sim	3×3 skew-symmetric matrix or predicted value in Chapter 5
$\dot{}$	First derivative with respect to time
$\ddot{}$	Second derivative with respect to time

Superscript

T	Matrix or vector transpose
-1	Matrix inverse
i	i^{th} iteration
$*$	Effective matrix

Subscripts

i	i^{th} beam in a system
g	g^{th} element in i^{th} beam
e	Inertial frame
o	Moving frame
\mathbf{r}	Rigid body
\mathbf{e}	Elastic body
t	Time

Symbols

$\hat{\mathbf{i}}_o \hat{\mathbf{j}}_o \hat{\mathbf{k}}_o, \hat{\mathbf{i}}_i \hat{\mathbf{j}}_i \hat{\mathbf{k}}_i$	Moving and reference Cartesian frames, respectively
$\hat{\mathbf{e}}_o, \hat{\mathbf{e}}_i$	Unit vectors of moving and reference frames, respectively
$\vec{r}_{ig}, \mathbf{r}_{ig}$	Absolute position vectors measured in moving frame
\vec{R}_i, \mathbf{R}_i	Position vectors of reference frame measured in moving frame
$\hat{\rho}_{ig}, \{\rho\}_{ig}$	Local position vectors measured in reference frame
\mathbf{R}	A position vector (different from \mathbf{R}_i)

R_{i1}, R_{i2}, R_{i3}	Three scalar components in \mathbf{R}_i
$\rho_{igs}, v_{ig}, w_{ig}$	Three scalar components in $\{\rho\}_{ig}$
R_1, R_2, R_3	Three scalar components in \mathbf{R}
\mathbf{T}_{eo}	Rotation matrix transferring inertial to moving frame
\mathbf{T}_{oi}	Rotation matrix transferring moving to reference frame
Ω	System angular velocity vector
Λ	Vector of generalized rigid body coordinates
\mathbf{N}	Coefficient matrix of system angular velocity vector
$\mathbf{N}_1, \mathbf{N}_2, \mathbf{N}_3$	Three column vectors in \mathbf{N} matrix
\mathbf{q}_i, \mathbf{q}	Local and global generalized coordinate vectors
\mathbf{d}_{ig}	Vector of generalized elastic coordinates
d_{igy}, d_{igz}	Elastic deflections in Y and Z directions, respectively
ϕ_{igy}, ϕ_{igz}	Elastic rotations about Y and Z axes, respectively
\mathbf{s}	Generalized axial coordinate vector of an element
$\mathbf{Y}_i, \mathbf{Z}_i$	Constant matrices defined in displacement functions
$\Theta_{i\lambda}, \Theta_{igd}, \Phi_i$	Compatibility matrices
G, \vec{G}	Scalar gravity and gravity vector
ρ_i, A_i, m_i	Mass density, cross-sectional area, and mass, respectively
KE_i, PE_i	Kinetic and potential energies, respectively
U_{bi}, U_{ei}	Body force potential energy and elastic strain energy
\mathbf{Q}_i	External nonconservative force vector
\mathbf{h}_i	Force vector due to elastic deflection
V_i	Rigid body potential function
\mathbf{k}_s	Structural stiffness matrix
$\mathbf{m}_i, \mathbf{c}_i, \mathbf{k}_i, \mathbf{f}_i$	Local level mass, damping, and stiffness matrices and force vector, respectively
$\mathbf{M}, \mathbf{C}, \mathbf{K}, \mathbf{F}$	Global level mass, damping, and stiffness matrices and force vector, respectively
$[1], [2], [3]$	Symbolic matrices
$[X_1], [X_2], [X_3]$	Symbolic matrices
\mathbf{a}, \mathbf{b}	Intermediate matrices
$\mathbf{a}_1, \mathbf{a}_2, \mathbf{a}_3$	Submatrices in \mathbf{a}
$\mathbf{b}_1, \mathbf{b}_2, \mathbf{b}_3$	Row vectors in \mathbf{b}
$\mathbf{a}_I, \mathbf{b}_I$	Constant vectors

$[\hat{\mathbf{b}}]$	Skew-symmetric matrix associated with \mathbf{b} matrix
$\mathbf{G}_{i1}, \mathbf{G}_{i2}$	Intermediate coefficient matrices
$\mathbf{H}_{i1}, \mathbf{H}_{i2}, \mathbf{H}_{i3}$	Intermediate coefficient matrices
$\mathbf{B}_{\alpha\beta}$	Intermediate submatrices
$(\alpha, \beta = 1, 2, 3)$	
$\mathbf{D}_{\alpha\beta}$	Intermediate submatrices
$(\alpha, \beta = 1, 2, 3)$	
$\mathbf{E}_1, \mathbf{E}_2, \mathbf{E}_3$	Intermediate submatrices
Δt	Time increment
β, γ	Newmark parameters
\mathbf{M}^*	Effective inertia matrix
$\Delta \ddot{\mathbf{q}}_e$	Increment of elastic acceleration vector
$\Delta \mathbf{f}$	Increment of effective force vector
a_1^r, a_2^r, a_3^r	Centrifugal (radial) accelerations
a_1^t, a_2^t, a_3^t	Tangential accelerations
$\lambda_1, \lambda_2, \lambda_3$	Rigid body angles
$\omega_1, \omega_2, \omega$	Rigid body angular velocities
I, J	Second moments of area
\vec{I}	Inertia dyadic
\mathbf{I}	Inertia matrix or identity matrix
P, F, W	Forces
T	Torque
\mathbf{T}	Rotational transformation matrix
δ	Elastic deflection
θ	Elastic rotation
ϕ	Elastic twist
E	Young's modulus of elasticity
G_i	Torsional modulus of elasticity

ACKNOWLEDGMENTS

Very special thanks are given to my major adviser, Dr. Joseph R. Baumgarten, and co-adviser, Dr. Donald R. Flugrad, Jr., for their consistent encouragement and help in conducting research and in seeking financial aid during the course of this study. Thanks are also due to Dr. Jeffrey C. Huston, Dr. Bion L. Pierson, and Dr. Patrick Kavanagh for their willingness to serve as my committee members.

I would like to say thank you to my fellow friends with whom I feel my life full of pleasure and a lot easier here at Iowa State University.

Finally, This dissertation is dedicated to my father, Mr. Shouheng Xu, and my family for their patient supporting and caring about my study.

CHAPTER 1. GENERAL INTRODUCTION

Launchings of several communication satellites with liquid stores on board have demonstrated an unstable coning motion of the satellite when boosted from the Space Shuttle into a geosynchronous Earth orbit. It is believed that an axial thrust from the motor at burnout of the satellite's power assist module (PAM) gives rise to the initiation of the sloshing motion of the liquid stores in the vehicle. A pressure field within the fluid is disturbed due to this initial liquid sloshing in the container. In addition, the presence of an interface between the liquid and gas will influence the motion of the tank containing liquid stores, hence the orbital motion of the satellite. If there is no internal energy dissipation, the spin-stabilized satellite will still be able to regain its stability in rotation. However, a large liquid propellant mass fraction contained in many communication satellites causes a significant amount of internal kinetic energy dissipation by viscous friction along the tank wall. The satellite, designed to be spin-stabilized about its axis of minimum moment of inertia, has a constant angular momentum when in orbit. It will attempt to conserve its angular momentum and begin to reorient its spin about an axis associated with a lower energy state. If this coning motion is not controlled, the spacecraft will eventually enter an unstable spin. Unfortunately, the constraints in spacecraft design cause many space vehicles to be configured with fairly flexible structures because added

weight for stiffening is costly in terms of payload reduction. This tends to aggravate the stability problem.

1.1 Overview of Research History

As shown in Figure 1.1, a flow chart of research history and personnel, the satellite project has already lasted one decade during which four faculty members and eleven graduate students actively participated. As a result, five Ph.D. degrees and four M.S. degrees were awarded, and fifteen technical papers and numerous annual reports [7] [8] [9] [10] were published. The project experienced three major phases from its beginning to its ending. Initiated in June 1982, exploration of research work of the first phase was completed in 1986. The possible cause and mechanism of instability of spin-stabilized spacecraft with sloshing fluid stores were identified during the period. The successful research initiation resulted in obtaining an award of an AFOSR grant for the next three years. From 1986 to 1989, a test rig, which is capable of simulating the coning motion of a satellite containing liquid stores, was designed and built. A rigid body model of the test rig dynamics was developed and initial computer simulation was accomplished. In the meantime, a first computational fluid dynamics (CFD) model was built up to represent the fluid sloshing motion. As a result of grant extension for an additional three years, a much more complicated and more realistic multibody flexible structure model was developed. The new structure model has the capability to accommodate CFD input and accounts for structure-fluid interaction while the CFD model is further modified and improved to be more effective and efficient. Stability analysis of the test rig was performed extensively using the rigid body model. Computer simulation of the structure-fluid interaction

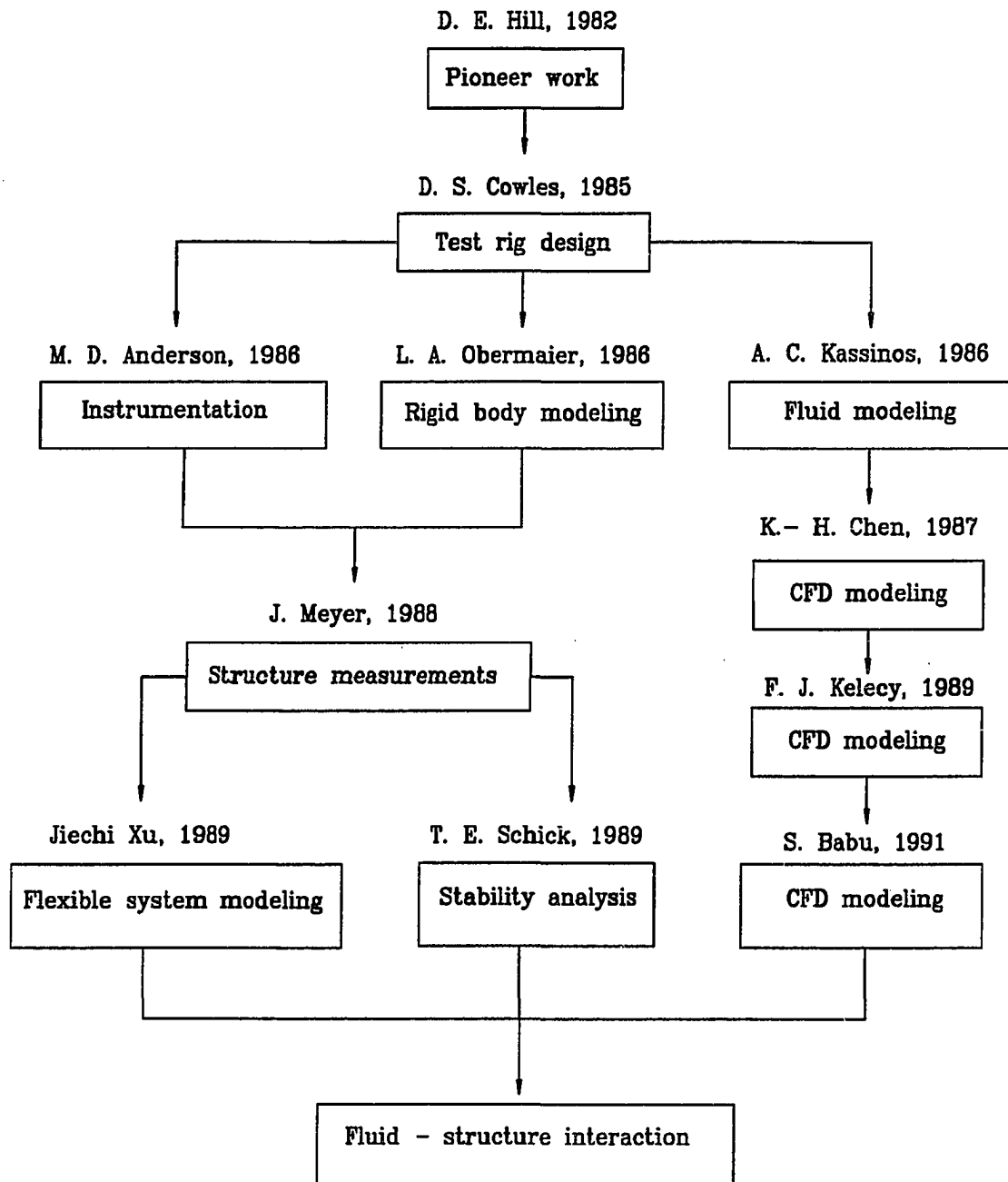


Figure 1.1: Flow chart of satellite project research history and personnel

was initiated and the results were generally acceptable. The last phase of the project is expected to be completed by the end of year 1992.

Hill [41] [42] [43], under supervision of Baumgarten, first pioneered and initiated research work on the project. He studied the dynamic response of sloshing fuel stores in the satellites and concluded that sloshing motion of the liquid stores in the vehicle, excited by the axial thrust, was the mechanism for creating the nutation to the spacecraft. The goal of his study was the development of a closed loop control law which may be applied to a spin-stabilized spacecraft with sloshing fluid stores without baffling and changing the design of the spacecraft. He successfully developed a linear optimal feedback control system, using an equivalent spherical pendulum to model fluid motion. This control system included the first mode of fluid oscillation, which employed state variable representation. The control law was shown to be stable for a wide variation in fluid level and could also be used for pointing maneuvers. It was implemented by sensing only the main body angular rates and attitude.

The desire to have a bench test device to duplicate the relative fluid motion in a spinning-nutating structure motivated the study of Cowles [35]. Under direction of Flugrad, Cowles designed and built the first working version of the satellite simulator shown in Figure 1.2. The test rig consisted of a vertical spin shaft with a horizontal crossbar. Two plastic spheres were supported by the horizontal bar at equal radial distances from the vertical spin axis. A two degree of freedom Hooke's type universal joint was located just below the horizontal beam in the vertical shaft to allow the spin axis of the horizontal beam structure to cone. A yoked sleeve, hand-actuated by a straight-line motion four-bar mechanism was utilized to cover the universal joint to give initial stability and rigidity to the system during spin-up. A D.C. motor was

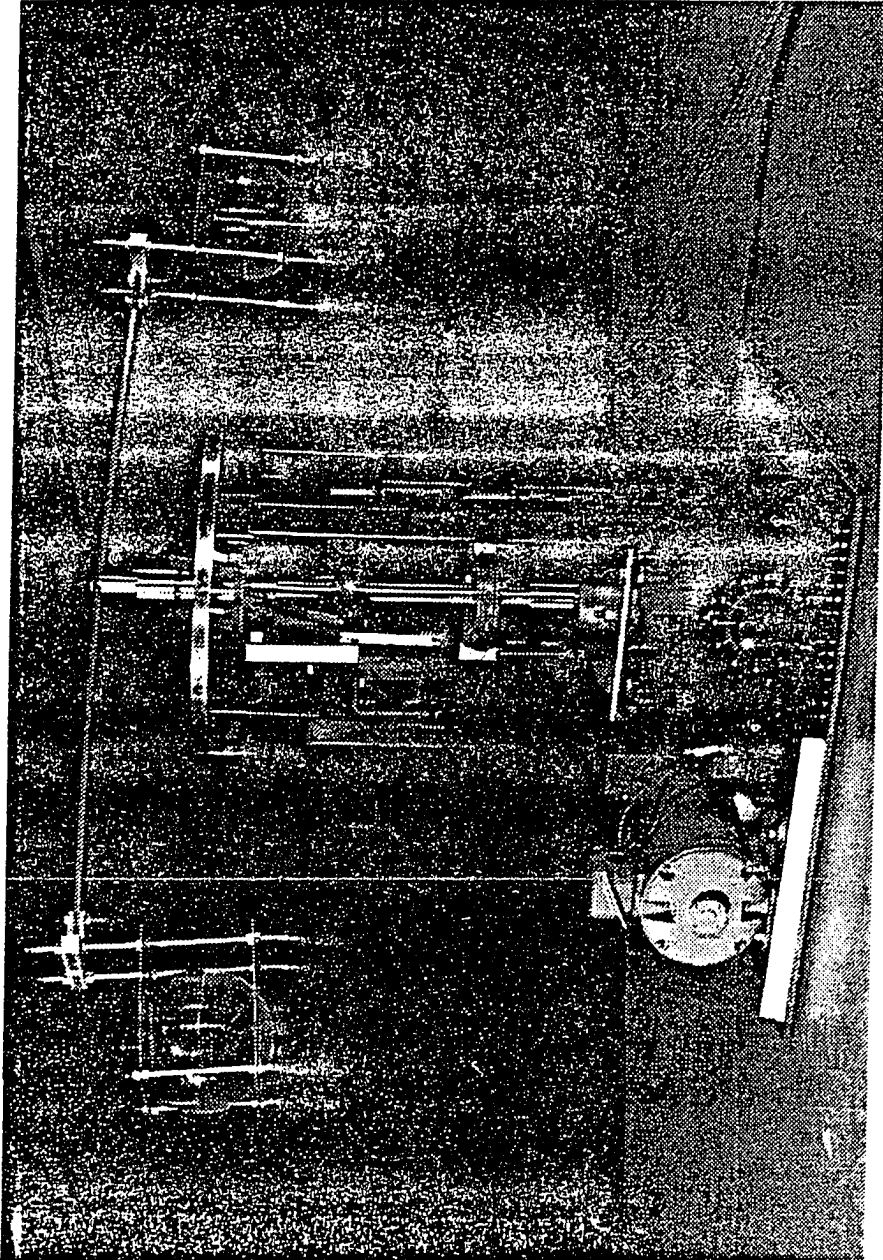


Figure 1.2: A first working version of the satellite simulator

selected to drive the system. Operation of the initial design of the completed test rig resulted in a very unstable motion. An analysis of the system inertias indicated that the spin axis was the axis of intermediate inertia.

Anderson [3] [38] improved the operation performance of the initial test rig and instrumented it to measure the response of the sloshing fluid in the nutating structure. He worked under the direction of Flugrad, reconfigured the test rig, instrumented the fluid tanks and the two degree of freedom universal joint, and set up a data collection system. A modified test rig (see Figure 1.3) with major change in the rig structure allowed for instrumentation of additional configurations. A new upper collar was designed to restrict the cone angle and to prevent damage to the unit under unstable operating conditions while providing the ability to restabilize the rig during spin. Instrumentation was developed to study the effects of liquid motion on the test rig dynamics. All runs for spin about an axis of minimum moment of inertia were found to be unstable and those for spin about an axis of maximum moment of inertia were stable. It was also found that small products of inertia can have a strong influence on the dynamics of the test rig. In conclusion, the flexibility of the structure greatly affected the coning motion and stability of the precessing simulator. Large beam deflections and rotations were observed during the runs due to the effect of centrifugal inertial force acting on the liquid mass.

Obermaier [75] [39] initiated analysis to guide further experimental work, to study the fundamental dynamic behavior of the test rig, and to predict stability characteristics of the test rig with arbitrary configuration and liquid filling percentage. Her thesis, directed by Flugrad, successfully developed a rigid body dynamic model with symmetric liquid tanks and equivalent pendulums representing sloshing liquid.

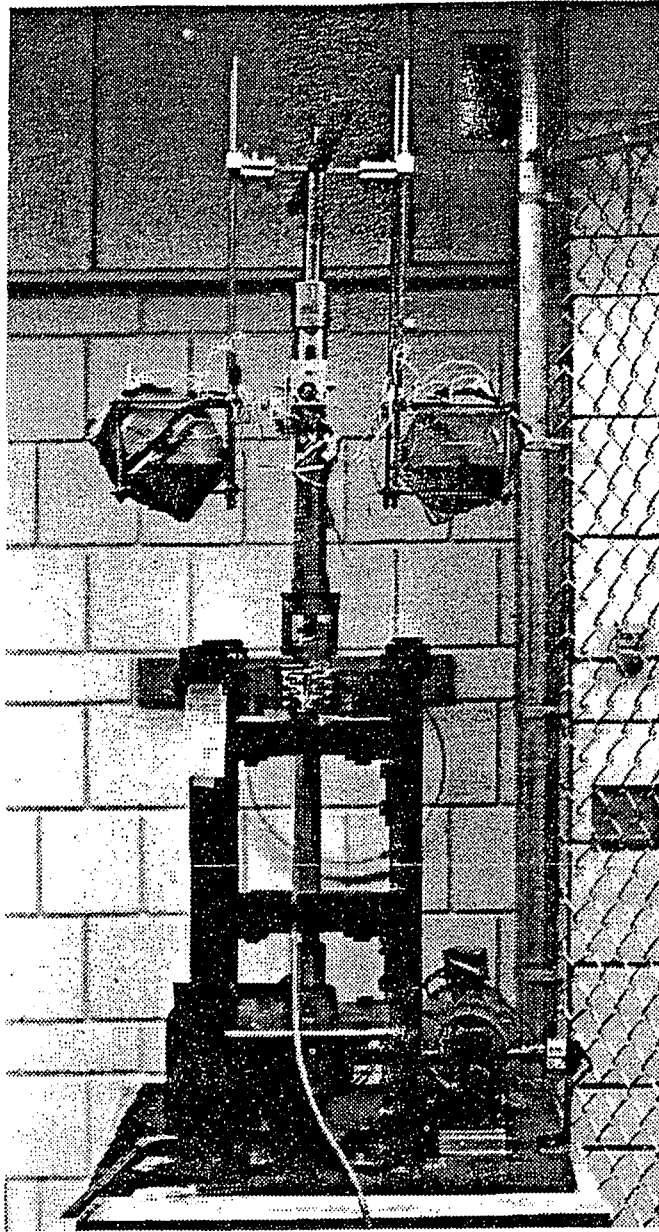


Figure 1.3: A modified final version of the satellite test rig

She used a Lagrangian formulation in company with state variables to best match the quantities being measured by Anderson's instrumentation. A computer program, SATELL, based on her mathematical model, was written and interfaced with a locally developed time integration program, DIFFEQ. Simulation results of SATELL were compared with the results of CAMS, a commercialized rigid body dynamics software package, and the corresponding measurements. Agreement between the output of SATELL and the results of CAMS was very close. Reasonable match between the SATELL runs and experimental data was also indicated. Difficulty in modeling the experimental setup, however, arose in determining values for mass moments of inertia of the test rig and damping effects of the equivalent pendulums. It was realized that it is necessary and important to develop a comprehensive multibody flexible dynamic model which couples the terms of gross rigid body motion and elastic vibration in order to more precisely describe and predict the motion and stability characteristics of the test rig.

Under direction of Baumgarten, Meyer [7] studied the fluid and structure interaction during the sloshing and nutating phase of the simulator motion. She installed a strain gauge measurement system on the test rig. One strain gauge was mounted on each side of the long vertical beams which connect the horizontal bar to two massive tanks on each side of the test rig. The voltage signal of a full-bridge was amplified by an op-amp and the output was to be recorded using a computerized data acquisition system. The work was interrupted after one year by her order to report to her first U.S. Air Force duty station.

The need to replace the pendulum model of the sloshing liquid in its spherical tanks with a computational fluid dynamic modeling of the free surface liquid moti-

vated the work of Kassinos [51] [52]. Under direction of Prusa, Kassinos formulated a primitive variable computer program for analysis of the dynamic fluid response. The liquid sloshing was characterized as incompressible laminar viscous flow and was modeled using Navier-Stokes equations. The general motion of the spherical tanks was transferred to a body fixed coordinate system with implementation of several successive axis rotations and translations. The pressure distribution within the fluid was determined by solving Poisson's equation. A free surface tracking technique was presented to identify the motion of the free surface. A type of fractional step method was adopted to obtain the flow field solution.

CFD modeling of liquid sloshing was further investigated by Chen [27] [28] [29] [30] under supervision of Pletcher. Chen adopted Kassinos's general formulation of the governing equations of incompressible sloshing flow and developed a new numerical method. This method solved a full set of governing equations simultaneously and eliminated the need to derive a pressure Poisson equation. A free surface fitting approach was employed in dealing with the free surface between the liquid and gas. Calculations were carried out in a transformed coordinate system that conforms to the shape of the free surface. An artificial compressibility method was approached in solving primitive variables in a strongly implicit manner. Elastic deflections and rotations were considered by adding more degrees of freedom. A first version CFD modeling program, SLOSH3D, was developed. Extensive computer simulation for the axisymmetric spin-up case was performed and the results of final steady state free surface positions agreed very well with the analytical solutions. Asymmetric spin-up cases were also investigated and minor difficulty in numerical integration was encountered.

Tremendously long computing times were needed to solve incompressible Navier-Stokes equations governing sloshing flow in one tank with the free surface fitting approach. Difficulty arose in handling sloshing flow at the high Reynolds numbers using the first version of the SLOSH3D program. Kelecy [53], directed by Pletcher, attempted to explore a brand new approach to simplify the complicated sloshing problem. It was hoped that a new CFD model would be more straight forward, more efficient, and more effective, and hence would largely reduce CPU time and accommodate input of a computerized structure model. Kelecy used a surface capturing approach along with a fractional step method to solve unknown primitive variables. The surface capturing approach employed a fixed grid in physical space in contrast to a moving grid in physical space with the surface fitting approach. Consequently, it was possible to use simpler coordinate systems which led to simpler forms of the governing equations. A numerical algorithm was developed and a computer code was written. The new CFD program was tested for an accelerating cubic tank case and a broken dam case [9]. The numerical results were encouraging and the agreement between the results of computer simulation and the experimental data was excellent. Future work was planned and additional efforts were spurred to refine the methodology. Final application of the method to the satellite propellant sloshing problem is expected to be accomplished soon.

Under work of Babu, supervised by Pletcher, the first version SLOSH3D code was revised. More features were added and some critical parts of the code which consumed most of computer execution time were vectorized. It was realized that the use of a finer grid could overcome numerical divergent problems when the code was executed beyond a certain range of high Reynolds numbers. A major difficulty associated with

grid refinement was that the time taken for the computations began to grow out of control. It was then decided to vectorize the code on a supercomputer, a Cray Y-MP with large memory size. The initial code was tailored to suit the capability of workstations available on campus which had very limited memory. The code was based on repetitive generation of the same sets of numbers rather than generating them just once and storing them in large arrays. The enhanced memory on large computers permitted switching to large storage and fewer calculations. In addition, a three-dimensional, coupled, strongly implicit procedure algorithm was identified as a critical part of the overall calculations. The technique consumed a large fraction of the computing time due to high data dependence of the implicit procedure and the consequent time consuming scalar execution loops. The algorithm was vectorized along surfaces of constant index sums, and the three-dimensional calculations were therefore converted to two dimensions [8]. The overall execution speed of the code was increased to about sixteen times the original speed.

1.2 Analysis of Structure Vibration

It is important to investigate the fundamental characteristics of a structure prior to developing a dynamic model for a system with flexible multibodies. Attention must be focused on two major concerns, first, gaining insight into the flexibility of each individual structure member and second, identifying the lower natural frequencies of the test rig. It is desirable to emphasize only the most significant degrees of freedom in terms of their quantities for the relatively flexible members in order to simplify the complexity and reduce the number of dynamic equations. Detailed fundamental structure analysis and results are developed and explained in the next chapter.

Finite element modeling best fits the needs of basic structural analysis in this study. A type of beam element was chosen in the discretization of the continuous system. A comprehensive commercial finite element package, MSC/NASTRAN [60], was employed to determine the stiffnesses and the eigenvalues of the test rig structure. By taking advantage of the symmetry about the spin axis of the test rig, only half of the system was considered and modeled. Besides, the finite element model was constrained such that the system was only allowed to spin about the vertical axis for the rigid body motion. Such an assumption reflects a real spin-up case when the collar is in its upper position covering the universal joint of the test rig. A realistic spin velocity profile with sinusoidal function was specified. The loadings considered included the tank weight, the torque due to the tank weight, and the centrifugal force induced during the spin of the test rig.

Two different types of NASTRAN runs [8] were accomplished, in which both two and ten elements were used to model each structure member in order to test the sensitivity to the number of elements chosen. A *Normal Modes Solution* was run to evaluate the real eigenvalues of the model. A *Direct Transient Response Solution* was then sought to obtain the response of the transient system vibration and the steady-state values. In conclusion, it was found that (a) relatively high natural frequencies were observed, (b) bending motions were recognized as the most significant elastic degrees of freedom, and (c) the most flexible members were identified.

1.3 Stability Study of a Rigid Body System

As a part of the overall study of dynamic behavior of the test rig, Schick [76] [77] further investigated stability characteristics of the test rig. His work was based on the

early work done by Obermaier. A rigid body model was established and the computer simulation program, SATELL, was modified. Schick was able to verify that the test rig does respond as an actual satellite thrusting in orbit for at least one configuration of the system by comparing the experimental data with the results of SATELL runs under conditions of zero gravity. The stability rules of a system of rigid bodies were developed and were verified by the simulation results and the experimental measurements. Instability ranges of the test rig with the different configurations were identified. The physical phenomenon of reorientation of the test rig during spin was successfully demonstrated by the simulation data and was also verified by the experimental tests. This further proved the effectiveness of the test rig modeled as a precessing satellite simulator.

1.4 Dynamic Modeling of a Flexible System

Flexible structure modeling, including the effects of elastic deflections and rotations, ultimately explore the natural behavior of the test rig more precisely. Extensive research work on flexible modeling has been conducted over the past decade. Very few investigations have dealt with the most complicated dynamic response for such a flexible system in which the overall rigid body motion of the system is strongly coupled with the unknown elastic deformation of each individual member of the system. However, this is the exact situation encountered in dealing with the modeling of the current test rig. During the course of this part of the study, the emphasis was concentrated on developing new theories and exploring new approaches which would eventually lead to the development of a systematic procedure to establish dynamic equations of motion for the test rig. Four major steps were developed in deriving

the equations of motion, in resolving numerical algorithms, and in accomplishing computation in this study [93] [94] [95] [96].

Firstly, the spatial elastic members which were identified during the analysis of the structural vibration using the NASTRAN package were discretized and modeled by using predefined finite elements. Assumptions were made to deal with three dimensional spatial deformation of each structure member. A type of beam element was also selected in the finite element modeling in order to reflect the structure bending with both deflection and rotation. A third order polynomial function was adopted as the element displacement function in a conventional stiffness method approach. Both the geometrical boundary conditions and the force boundary conditions were proposed based on agreement with the real configurations of the test rig and the external loadings acting on the system, respectively. For each individual structure member, the elastic degrees of freedom were determined and the structure stiffness matrix was then formulated.

Secondly, Lagrange's approach was employed in the derivation of dynamic equations of motion. Elastic strain energy of the flexible structure members was accommodated in the potential energy term of the Lagrangian formulation. The degrees of freedom from both the rigid body motion and the elastic deformation were considered as unknown degrees of freedom of the entire system. It was therefore no longer necessary to assume the prescribed gross rigid body motion which is usually specified in most situations. In consequence, two kinds of motion, the large rigid body motion and the small elastic vibration, which are mutually coupled and influence each other, ultimately determine the inherent nature of the motion of the test rig. Nonlinear coupling terms were completely taken into account and were derived explicitly using

a matrix formulation. The equations of motion of the system on the global level were obtained by performing the assembly of the equations of motion for each individual structure member on the local coordinate level with implementation of the geometrical boundary conditions by means of a compatibility matrix. Liquid sloshing within the spherical container was modeled using computational fluid dynamics (CFD) [9] under a separate effort.

Thirdly, the system equations of motion in matrix form were characterized as a second order nonlinear ordinary differential equation system. The coefficient matrices of the equations included the time-varying rigid and elastic variables. Implicit integration algorithms tend to be numerically stable, permitting large time steps. Explicit algorithms, on the other hand, tend to be effective for nonlinear systems with low natural frequencies in assuring the numerical stability which depends on the highest natural frequency of the system. However, neither class, implicit nor explicit, seemed optimally suited and efficient by itself in dealing with systems in which both linear and nonlinear properties are involved. In this study, a sequential implicit-explicit integration algorithm with predictor-corrector schemes arising from a Newmark method was developed in which an attempt was made to achieve the benefits of both classes of algorithms. The matrix equations were rearranged and partitioned into two sets of coupled equations which were solved sequentially. A Newmark method provided the primary algorithm in the time integration of the system equations.

Finally, a general computer code was written using the new numerical techniques developed in this study. An arbitrary number of finite elements can be specified for each elastic member. All the material properties, the geometrical configuration of

the test rig, and the numerical parameters of the algorithm were treated as input information stored in a data file. The simulation results matched closely with the experimental data as indicated in the simulation chapter. In particular, close attention was paid to the most sensitive and comprehensive overall transverse deflection of the spherical tank mounted on each side of the test rig. The difference between the analytical and experimental results was within 5% in their qualitative patterns and their magnitudes. The lower natural frequencies identified in the NASTRAN runs were also observed in simulation. In general, the results were very favorable and acceptable.

CHAPTER 2. FUNDAMENTAL STRUCTURE ANALYSIS

From previous work, investigators showed that a mathematical model based on rigid body dynamics can no longer precisely describe and predict the stability characteristics for the satellite simulator. A visible structural deflection was observed in a video tape recording of the precessing simulator. To gain further insight into the dynamic response of the test rig, a flexible structure model associated with deformable body dynamics must be developed. A finite element method was chosen in the current study to discretize some of the most flexible members under consideration.

Primarily, there are two fields of published research dealing with the modeling of flexible structures. One is in the area of kinematic response of flexible mechanisms and the other one is in the robotics area. A common fact is noticed in both fields that either the joint rigid body motions of a mechanism are specified or the trajectories of robot arms are prescribed. To date, almost all investigators have used the same approach to obtain dynamic response by superposing the elastic motion upon a known rigid body motion. Very few studies have discussed a motion coupled method. To address this problem a complete method is needed to solve the rigid body motion and elastic motion simultaneously. Such a method will consider the effect of flexible deflection and rotation in succeeding links on the current link, and will thus proceed down the chain. Understandably, this method is extremely difficult to develop. Two

major difficulties arise. First, detailed and tedious symbolic coordinate transformations and differentiation must be performed in the procedure to derive the governing equations of motion. Second, a special technique must be developed to numerically integrate the resulting nonlinear differential equations having (a) large-valued position variables (rigid body motion) and (b) small-valued displacement parameters (elastic motion). Unfortunately, the gross rigid body motion is unknown in the present problem. This would indicate an approach be based on the motion coupled method. However, an alternative approach still seems possible. First, a rigid body model is employed, and the gross rigid body motion, in turn, can then be predicted from this model. Second, a flexible structure model is built up using a superposition method. For each time step, the rigid body motion can be predicted from the first model and the total motion is then obtained from superposing the two motions. According to Huang [44], a superposition model can sometimes fail to describe the nonlinear coupling behavior. Therefore, a potential risk would exist in some situations in the prediction of highly nonlinear characteristics.

In this chapter, efforts are concentrated on studying structural fundamental behavior of the test rig. Orbital spin-up motion of the structure system is considered as the primary investigation case in this chapter. Only the axisymmetric part of the structures about the axis of the vertical shaft is modeled, due to the symmetry of beam deflections on each side of the tank arm of the test rig. Centrifugal and tangential forces acting on one tank during spin of the system are transformed to the corresponding external forces and torques on the system in the formation of a pseudo vibration analysis. A solid finite element model using beam elements representing flexible beams is developed. The model was adapted to suit the solution structure

of a commercialized finite element software package, MSC/NASTRAN [60]. Several runs were successfully executed and the findings were recorded.

In the finite element method, each node possesses six degrees of freedom. By considering all these coordinates for each individual element, the final set of system dynamic equations will be very large. In order to reduce the number of equations, it is necessary that some degrees of freedom of fairly rigid structures must be neglected, and emphasis must be focused on the most significant ones. A thorough knowledge of fundamental vibration behavior of the structures must be known. MSC/NASTRAN is a comprehensive and powerful finite element package which is currently available in the Department of Mechanical Engineering at Iowa State University. It was successfully employed in determining the stiffness and modes of the test rig structure in preparation for development of a discretized dynamic model.

2.1 Analysis of Inertial Forces

It has been shown that the configuration of the structure system is symmetric about the vertical axis of the lower shaft. This does not imply, however, the symmetry of inertial forces which induce elastic deflection and rotation. Figures 2.1, 2.2, and 2.3 depict the active centripetal and tangential accelerations in three orthogonal planes, x - z , x - y , and y - z . The major concerns here are the beam transverse deflections induced by the inertial forces and external loading. Figure 2.1 shows the active accelerations in the x - z plane. All of three rotations will affect the beam deflections in this plane. Besides, the tangential accelerations, \vec{a}_2^T , acting on tank 1 and tank 2 are non-symmetric with the presence of other accelerations. Hence, the corresponding deflections in the x - z plane are not symmetric either. In Figure 2.2, it can be

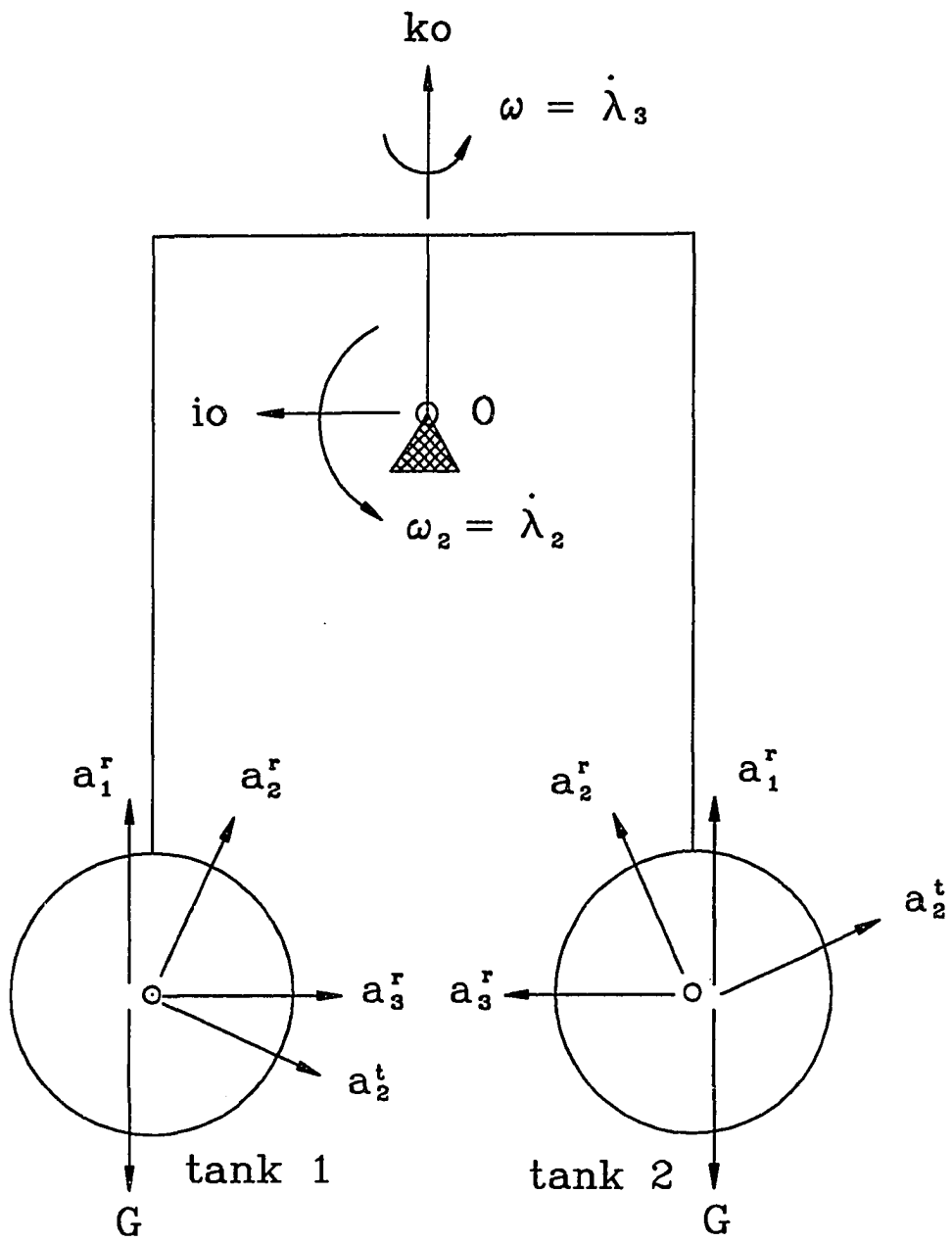
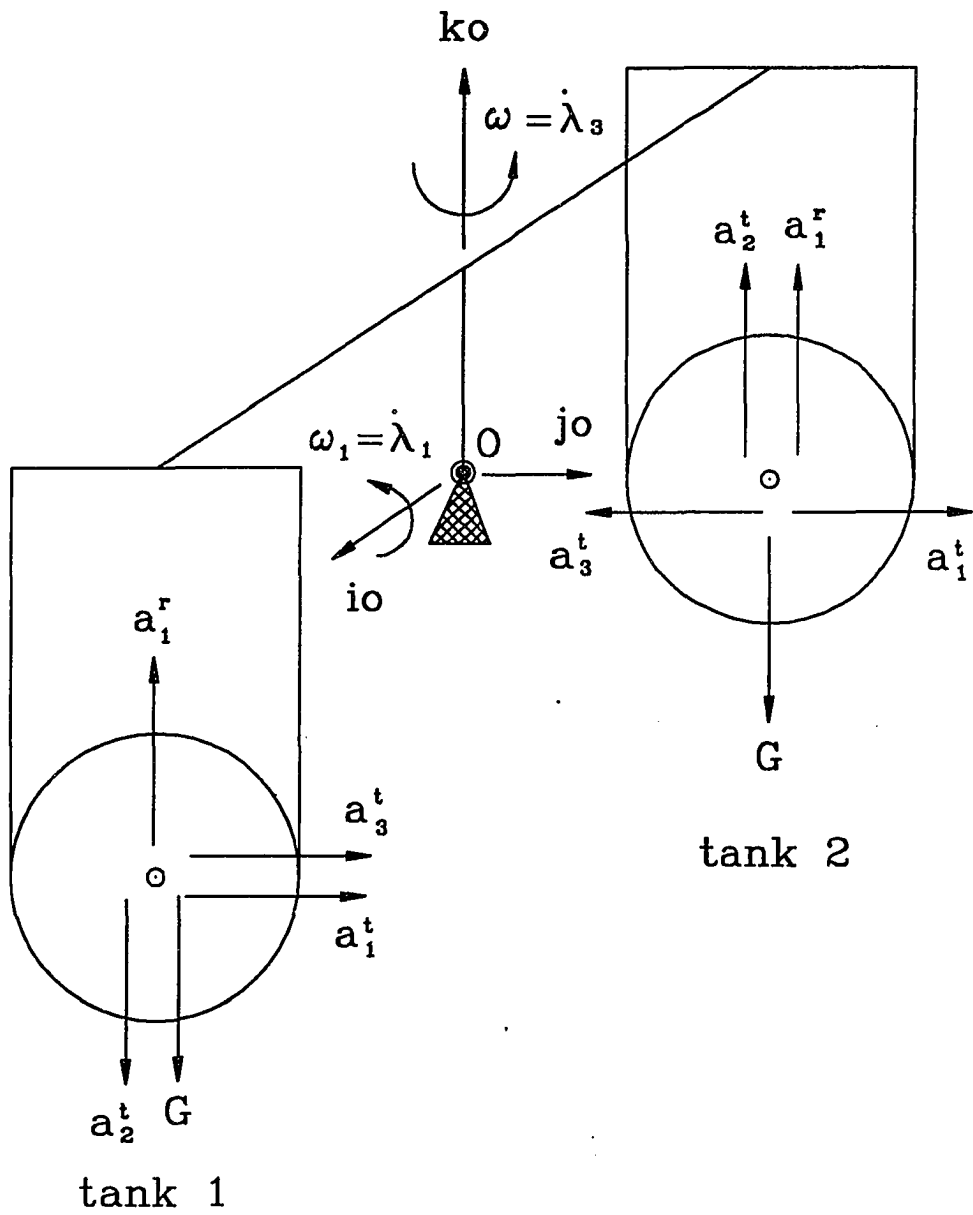
Figure 2.1: Accelerations in the x - z plane

Figure 2.2: Accelerations in the x - y plane

Figure 2.3: Accelerations in the y - z plane

found that the pair of tangential accelerations, \vec{a}_1^t and \vec{a}_3^t , are not symmetric, nor are the deflections in the x - y plane. In the y - z plane, as shown in Figure 2.3, there also exists another asymmetric pair of tangential accelerations, \vec{a}_2^t and \vec{a}_3^t . In summary, the tangential inertial forces will induce non-symmetric elastic beam transverse deflections. It was concluded therefore that the different elastic generalized coordinates must be defined for each elastic beam in spite of the geometric symmetry of the structure configurations.

2.2 Structural Vibration Analysis

The main purpose of performing structural vibration analysis using the finite element method in this chapter is to identify the most flexible structures in the system. A vibration model for the structural system can be built up by applying to the system the external and inertial forces which induce the elastic deformation. The inertial force can be found once the kinematic motion of the system is specified by assuming a realistic spin velocity profile for the system. For a stable case when the universal joint is covered by the collar, it is known that the upper assembly, a rotating structure fully supported by the universal joint, spins about the vertical axis of the lower shaft without nutation. Two different types of accelerations, centripetal and tangential, are produced and involved. From a statics point of view, the corresponding centrifugal and tangential inertial forces essential to the study of structural vibration can be loaded on the structure. A quasi-static kind of problem is then formed.

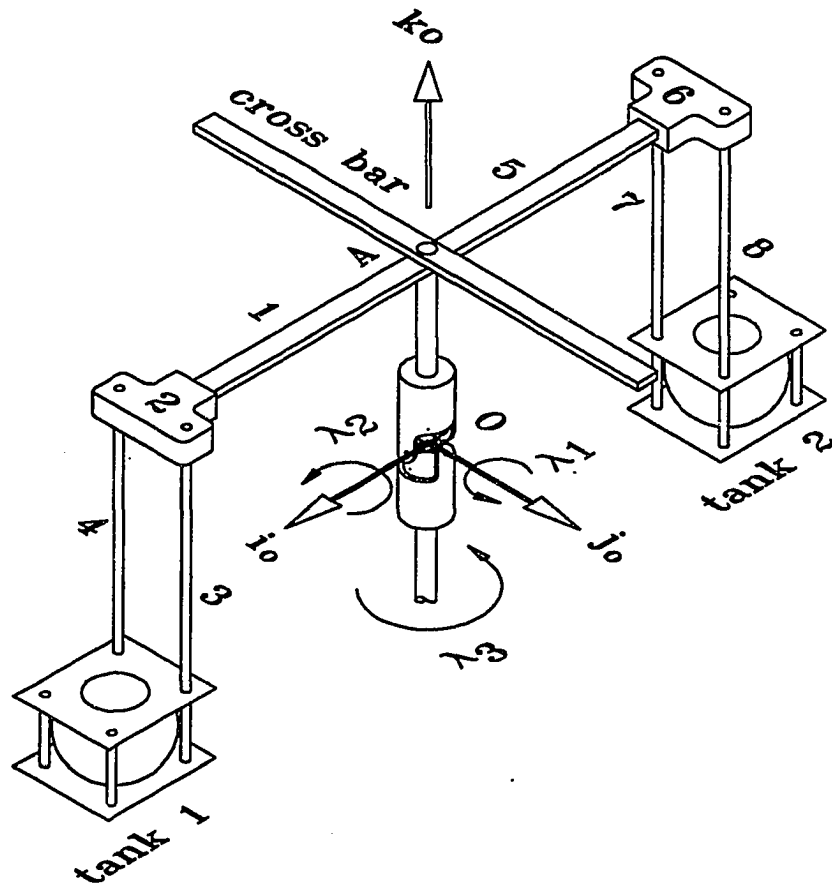


Figure 2.4: A schematic drawing of structure configuration (dynamic part only)

Table 2.1: Material and geometry properties of each individual beam

Beam (or Bar)	1	5	Bar	2	6	3	4	7	8
Material	Steel			Aluminum		Steel			
Cross-sectional area (mm^2) & shape	72 Rectangle			361 Square		38.5 Circle			
Second moment of area (mm^4)	$I_y = 216$ $I_z = 864$ $J = 593.6$			$I_x = 10860$ $I_z = 10860$ $J = 18375.3$		$I_x = 117.9$ $I_y = 117.9$ $J = 235.7$			
Young's modulus (GPa)	210			70		210			
Poisson ratio	0.3								

2.2.1 Structure configuration and material properties

A schematic drawing (dynamic part only) of the test rig is shown in Figure 2.4. Point O represents a universal joint connecting an upper shaft and a lower shaft. The upper shaft is constrained except for spinning about the vertical axis of the lower shaft for this stable case. The geometry of the entire upper assembly is axisymmetric about the axis of the upper shaft. Beams 1 and 5, the cross bar, and the upper shaft are rigidly connected at point A . Each adjacent beam is also rigidly connected. Two spherical containers are partially filled with the liquid which represents the free surface fuel load.

Material properties and individual beam (or bar) geometry are listed in Table 2.1. Each tank assembly (or tank for simplicity) has a weight of 15 Newtons (3.3 lbs) including the weight of the liquid and the weight of the structures in the tank assembly. All the beams and the cross bar are made of steel, except for beams 2 and 6 which are made of aluminum.

2.2.2 Finite element model

A finite element model is introduced in this section to assist the analysis of structural vibration. Figure 2.5 illustrates a solid finite element model with ten beam elements for each beam of a symmetric half of the upper assembly. Loadings $P = 7.5(N)$ and $T = 600(N - mm)$ are the corresponding half tank weight and external torque due to the tank weight. Forces F_t and F_r are the primary tangential inertial and radial forces, respectively, due to the tank weight. These forces depend on a function of applied spin angular velocity, $\omega(t)$. Beam 1 is constrained at node 1 (or point A) in all six degrees of freedom. Each adjacent beam is connected through clamped joints.

2.2.3 Angular velocity profile and inertial force

A realistic function of sinusoidal profile is specified as a spin angular velocity profile (see Figure 2.6), $\omega(t)$. The first derivatives of the profile with respect to time, t , at both the starting and ending points are zero, and the value of the corresponding angular acceleration function maintains finite value throughout the range. The velocity profile is determined as a piece-wise continuous function with the following expression

$$\omega(t) = \begin{cases} \frac{1}{2}\omega_0 \left[1 - \cos\left(\frac{\pi t}{t_0}\right) \right] & (0 \leq t \leq t_0) \\ \omega_0 & (t \geq t_0). \end{cases}$$

Where $t_0 = 3$ seconds and $\omega_0 = 360$ deg/sec (see Figure 2.6). Differentiation of the angular velocity with respect to time results in an angular acceleration function as

$$\dot{\omega}(t) = \begin{cases} \frac{1}{2t_0}\pi\omega_0 \sin\left(\frac{\pi t}{t_0}\right) & (0 \leq t \leq t_0) \\ 0 & (t \geq t_0). \end{cases}$$

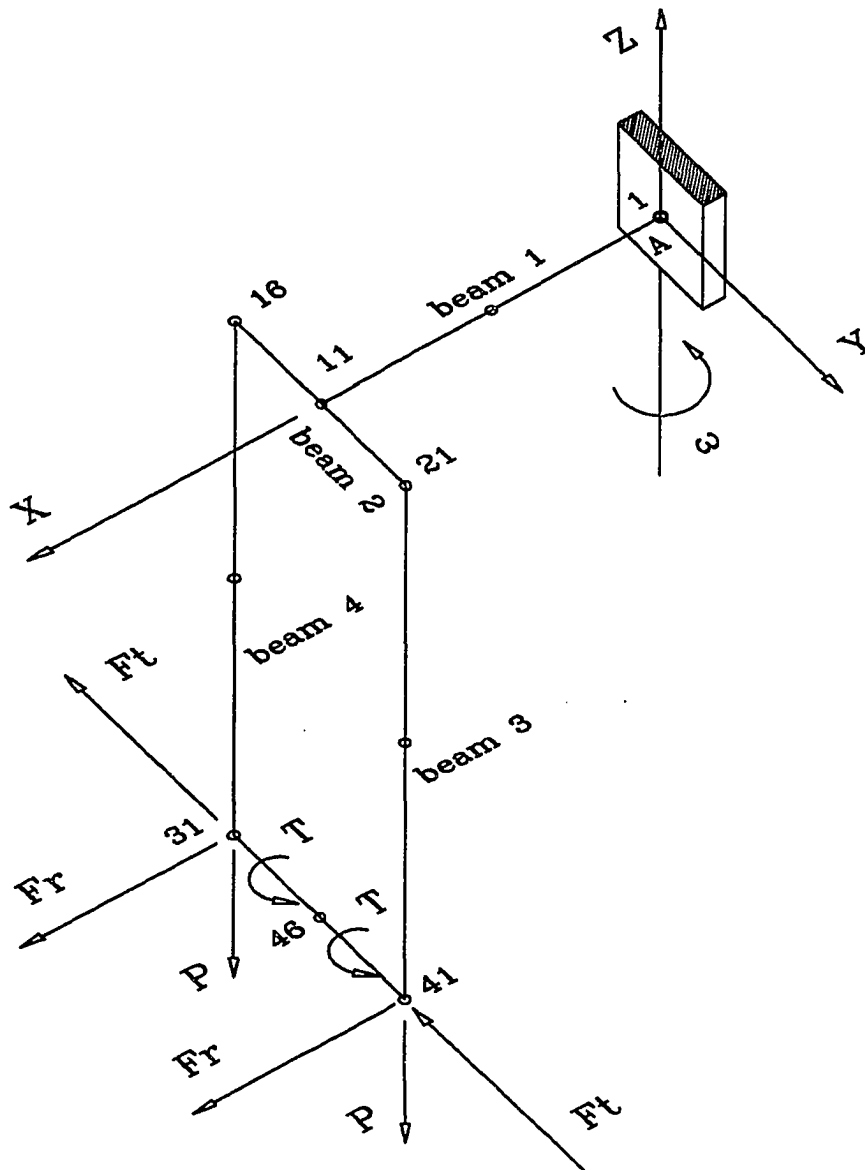


Figure 2.5: Finite element model of an axisymmetric half of the upper assembly

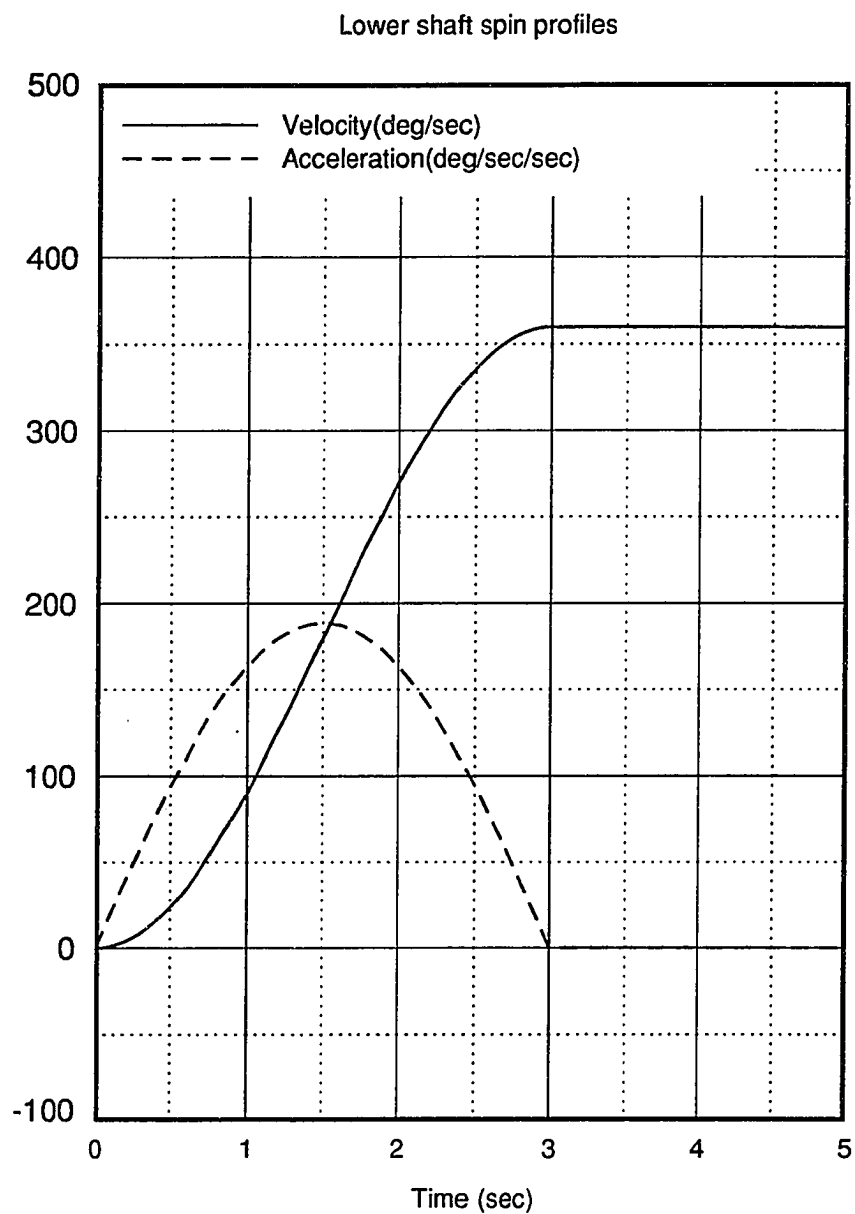


Figure 2.6: A set of sinusoidal spin-up profiles: The solid line denotes spin-up velocity, $\dot{\lambda}_3(=\omega)$; The dashed line denotes spin-up acceleration, $\ddot{\lambda}_3(=\dot{\omega})$

This angular acceleration satisfies the finite value condition. Let \vec{e}_r be a unit vector in radial direction of positive x , and \vec{e}_t be a unit vector in tangential direction of positive y , consistent with the direction of positive angular velocity, ω . The inertial forces acting on the center of the tank become

$$\begin{aligned}\vec{F}_i &= mL(\omega^2\vec{e}_r - \dot{\omega}\vec{e}_t) \\ &= \begin{cases} \frac{1}{8}mL\omega_0^2 \left[3 + \cos\left(\frac{2\pi t}{t_0}\right) - 4\cos\left(\frac{\pi t}{t_0}\right) \right] \vec{e}_r \\ - \frac{1}{2t_0}\pi\omega_0 mL \sin\left(\frac{\pi t}{t_0}\right) \vec{e}_t & (0 \leq t \leq t_0) \\ \omega_0^2 mL \vec{e}_r & (t \geq t_0) \end{cases}\end{aligned}$$

where $m = 1.5kg$ is the mass of the tank and L is the radial distance in the X direction from point A to the tank center. Thus, the magnitudes of the inertial forces acting on node 31 and node 41 (see Figure 2.5) are found to be

$$F_r = \begin{cases} \frac{1}{16}mL\omega_0^2 \left[3 + \cos\left(\frac{2\pi t}{t_0}\right) - 4\cos\left(\frac{\pi t}{t_0}\right) \right] & (0 \leq t \leq t_0) \\ \frac{1}{2}mL\omega_0^2 & (t \geq t_0) \end{cases}$$

and

$$F_t = \begin{cases} -\frac{1}{4t_0}\pi\omega_0 mL \sin\left(\frac{\pi t}{t_0}\right) & (0 \leq t \leq t_0) \\ 0 & (t \geq t_0). \end{cases}$$

where F_r is the centrifugal (radial) force and F_t is the tangential inertial force.

2.2.4 NASTRAN run

Two different types of NASTRAN runs were accomplished. A Normal Modes solution was run first to evaluate the real eigenvalues of the system. A Direct Transient Response solution was then sought to obtain a system transient vibration response to the external loadings with both static and dynamic forces. A moderate 0.2 system damping ratio was chosen in the transient response run. Due to the relatively

Table 2.2: Computed real eigenvalues of the finite element model using NASTRAN normal modes solution

MODE NO.	EXTRACTION ORDER	EIGENVALUE	RADIANS/SEC	CYCLES/SEC
1	1	2.166460E+04	1.471890E+02	2.342586E+01
2	4	3.008446E+04	1.734487E+02	2.760522E+01
3	2	2.146677E+05	4.633224E+02	7.374005E+01
4	5	4.415662E+05	6.645045E+02	1.057592E+02
5	7	1.003559E+06	1.001778E+03	1.594379E+02
6	3	2.523443E+06	1.588535E+03	2.528232E+02
7	8	2.983474E+06	1.727273E+03	2.749041E+02
8	6	3.465876E+06	1.861686E+03	2.962966E+02
9	9	6.159099E+06	2.481753E+03	3.949833E+02
10	10	7.351627E+06	2.711388E+03	4.315308E+02
11	12	1.961671E+07	4.429075E+03	7.049092E+02
12	11	2.111069E+07	4.594637E+03	7.312592E+02

high natural frequencies tabulated in Table 2.2, the transient response decayed very rapidly, and a steady-state value appeared after approximately two seconds. Node 41 stated in this section and in Figures 2.7, 2.8, 2.9, and 2.10 are all referred to Figure 2.5. Figure 2.7 shows the translational displacements at node 41 at which the largest deflection occurs. The X component is the most significant one among three components. The angular displacements at node 41, illustrated in Figure 2.8, clearly show a large steady-state value in the Y direction. Therefore, it can be concluded that the bending of either beam 3 or beam 4 in the X - Z plane is the most significant bending of the beams while the corresponding structural stiffness is very soft. The time responses of a typical translational velocity and an angular velocity at node 41 were plotted in Figure 2.9 and Figure 2.10, respectively. A large transient amplitude of the translational velocity in the X component can be observed in Figure 2.9. A

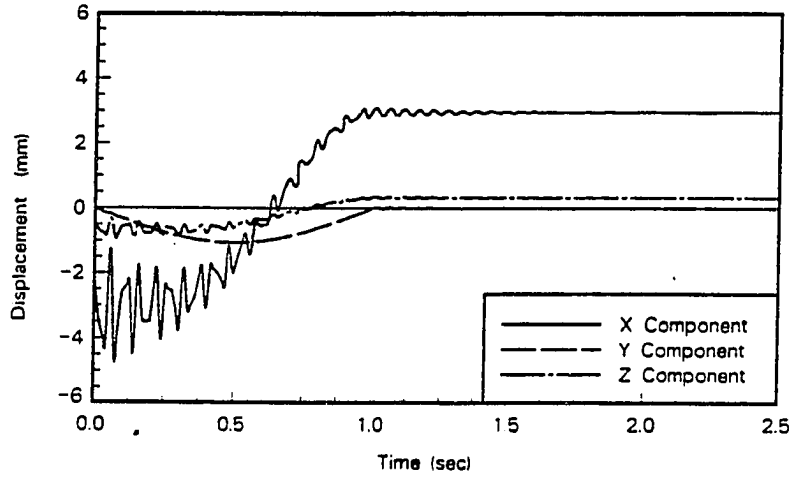


Figure 2.7: Translational displacements at node 41 (from NASTRAN run)

sudden jump in angular velocity was also observed near one second in Figure 2.10. This velocity jump was due to a combination of the form of the profile of the lower shaft input angular velocity, $\omega(t)$, and a special treatment of the radial force, F_r , in order to compromise with the rigid format for loading functions in NASTRAN code.

2.2.5 Estimation of elastic deformation

In this section, the elastic deformation of the flexible structures (see Figure 2.5) are calculated by using classic elasticity theory to verify the results obtained by running MSC/NASTRAN code. It is also attempted to identify some relatively small deformations among the beam transverse deflection, rotation, torsion, and axial displacement to further confirm the findings from MSC/NASTRAN code. Figure 2.11 shows a cantilever beam model of beam 1 shown in Figure 2.5. The dimensions are

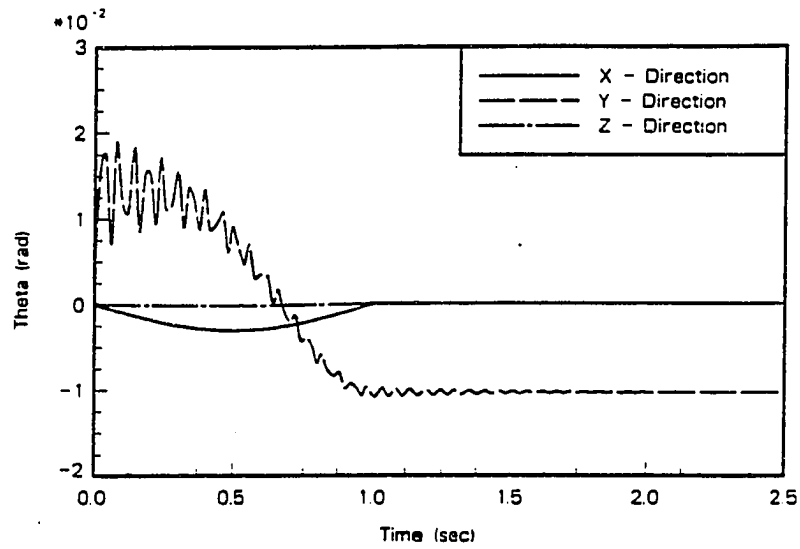


Figure 2.8: Angular displacements at node 41 (from NASTRAN run)

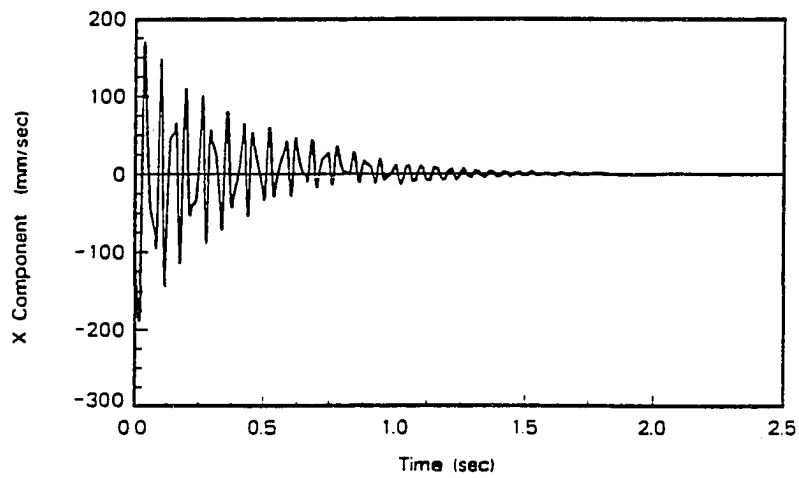


Figure 2.9: Translational velocity at node 41 (from NASTRAN run)

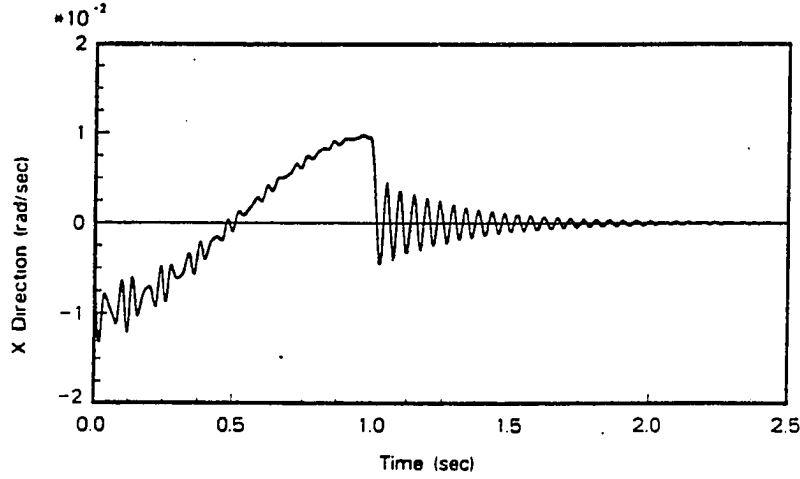


Figure 2.10: Angular velocity at node 41 (from NASTRAN run)

$L_1 = 150(mm)$, $a_1 = 12(mm)$, and $b_1 = 6(mm)$. The loadings acting on beam 1 using the spin profiles specified in the previous section are $W = 15(N)$, $F_1 = 13.6(N)$, $T_1 = 220(N-mm)$, $M_s = 1200(N-mm)$, and $M_d = 4100(N-mm)$, where W is the weight of the tank, F_1 is the axial loading resulting from the centrifugal inertia, T_1 is the torque resulting from the tangential inertia, M_s is the static bending moment due to the tank weight, and M_d is the dynamic bending moment due to the centrifugal inertia.

The maximum transverse deflection in the Z direction at the free end due to bending is

$$\begin{aligned}\delta_t &= \frac{(M_d - M_s)L_1^2}{2EI_1} - \frac{WL_1^3}{3EI_1} \\ &= 0.35 (mm)\end{aligned}$$

where $I_1 = 216 (mm^4)$ is the *second moment of the cross-sectional area* of beam 1

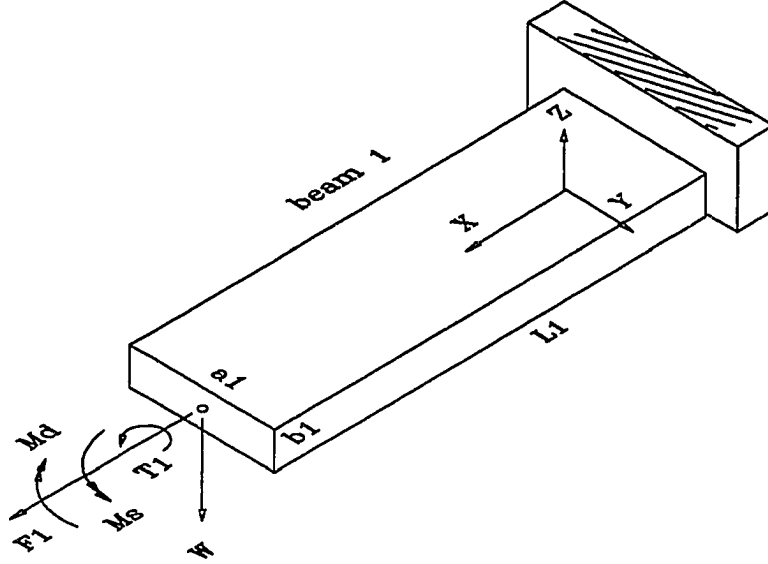


Figure 2.11: A cantilever beam model of beam 1

against the bending in the X - Z plane. The corresponding rotation about the Y axis at the free end is

$$\begin{aligned}\theta &= \frac{(M_s - M_d)L_1}{EI_1} + \frac{WL_1^2}{2EI_1} \\ &= -5.84 \times 10^{-3} \text{ (rad)}\end{aligned}$$

The longitudinal displacement in the axial direction of beam 1 is

$$\begin{aligned}\delta_a &= \frac{F_1 L_1}{a_1 b_1 E} \\ &= 1.35 \times 10^{-4} \text{ (mm)}\end{aligned}$$

The maximum angle of twist about the X axis due to torsion is

$$\begin{aligned}\phi &= \frac{T_1 L_1}{G_t k_1 a_1 b_1^3} \\ &= 7.7 \times 10^{-4} \text{ (rad)}\end{aligned}$$

where $G_t = 80 \text{ (GPa)}$ is the torsional modulus of elasticity and $k_1 = 0.229$ is a coefficient for a rectangular cross-sectional shape [18] [19].

It is therefore concluded that the axial displacement, which is much smaller in magnitude than the transverse deflection, and the twisting, which is also smaller in magnitude than the rotation, can be neglected in terms of calculating the elastic deformation of beam 1. It also can be shown that the entire elastic deformation of beam2 (see Figure 2.5) and the twisting and axial displacement of beams 3 and 4 are also negligible in this study.

On the basis of the findings, the rigid assumption for beam 2 and negligible twisting and axial displacement for all flexible beams, the overall displacement in the $X - Z$ plane at node 41 (see Figure 2.5) due to the corresponding bending can be found as

$$\begin{aligned}
 \delta_x &= (\delta_x)_{static} + (\delta_x)_{dynamic} \\
 &= - \left[L_3 \left(\frac{M_s L_1}{EI_1} + \frac{W L_1^2}{2EI_1} \right) + \frac{M_s L_3^2}{4EI_3} \right] + \left[L_3 \frac{M_d L_1}{EI_1} + \frac{F_1 L_3^3}{3E \cdot 2I_3} \right] \\
 &= -3.4 \text{ (mm)} + 6.53 \text{ (mm)} \\
 &= 3.13 \text{ (mm)} \\
 \delta_z &= (\delta_z)_{static} + (\delta_z)_{dynamic} \\
 &= - \left(\frac{M_s L_1^2}{2EI_1} + \frac{W L_1^3}{3EI_1} \right) + \frac{M_d L_1^2}{2EI_1} \\
 &= -0.67 \text{ (mm)} + 1.01 \text{ (mm)} \\
 &= 0.34 \text{ (mm)} \\
 \theta &= \theta_{static} + \theta_{dynamic} \\
 &= \left[\left(\frac{M_s L_1}{EI_1} + \frac{W L_1^2}{2EI_1} \right) + \frac{M_s L_3}{2EI_3} \right] - \left[\frac{M_d L_1}{EI_1} + \frac{F_1 L_3^2}{2E \cdot 2I_3} \right] \\
 &= 0.015 \text{ (rad)} - 0.026 \text{ (rad)} \\
 &= -0.011 \text{ (rad)}
 \end{aligned}$$

where $L_3 = 300 \text{ (mm)}$ is the length of beam 3, $I_3 = 118 \text{ (mm}^4\text{)}$ is the second moment

of area of beam 3, δ_x is the X component of the translational displacement at node 41, δ_z is the Z component of the translational displacement at node 41, and Θ is the angle of twist (angular displacement) about the Y axis at node 41. These three values match the corresponding values resulting from the MSC/NASTRAN run (see Figures 2.7 and 2.8). Similarly, other elastic deformations can also be verified.

From the above analysis it was concluded that the rigidity assumption of beam 2 was valid, and both the axial and torsional deformation can be neglected in simplifying the dynamic motion analysis without affecting the accuracy in predicting the overall elastic deformation.

CHAPTER 3. MODELING OF FLEXIBLE BODIES

3.1 Introduction

Flexible dynamic modeling has been an attractive but difficult topic for a long time. Severely restricted by the lack of computer processing speed in the early years and the complexity of the mathematical formulation, traditional designs in robots, mechanisms, and other relatively flexible structures have been limited to the realm of rigid body dynamics. However, increasing demands for higher operating speeds and better performance result in a desire for light weight structures. A by-product of the flexibility effect is now recognized as a critical issue. It becomes impossible to implement the required time-consuming numerical integration without solid support of sophisticated modern computers with high processing speed capability.

The past decade has seen significant advances in dynamic analysis for flexible multibody systems. Extensive work, analytically and experimentally, has been conducted in dealing with flexible modeling. Most investigators, however, employ a common approach in which the elastic deformation is superimposed on the gross rigid body motion due to the nature of their specific problems. The application of that method is severely limited due to the need for predefined rigid body motion. It is therefore very desirable to investigate a new approach in which all the degrees of freedom (DOF) of a system, elastic as well as rigid, are treated as unknown gener-

alized coordinates. This enables analysis of situations where the rigid body motion needs to be predicted, and the relationship between two motions affects the system stability.

The purpose of this study is to explore a general modeling technique to develop such a systematic procedure for establishing dynamic equations of motion of a flexible system with mutually dependent rigid body and elastic motions. In addition, the formulation procedure is to be optimized and simplified so as to accommodate the needs of numerical analysis and computer programming.

3.2 Previous Work Review

The first to exploit the advantages of the finite element analysis (FEA) with Lagrangian mechanics were Sunada and Dubowsky[86][87]. Their model incorporated a Denavit-Hartenberg [36] representation of the kinematic rigid body transformation excluding kinematic coupling. The degrees of freedom of the discretized system were reduced by means of *component mode synthesis* (CMS) [48] [49] [37] [81]. The equations of motion of all links were assembled using a Compatibility Matrix routine. In their illustrative examples, a set of first order equations was solved numerically for a special case in which the mechanism's nominal speeds and accelerations are much smaller than the component elastic coordinate velocities and accelerations. In their later extended work, the assembly of dynamic equations was performed in symbolic form due to the special form of matrix terms. The final system equations were solved using a Newmark-Beta integration algorithm. Their approach is applicable for problems where nominal rigid body motion is specified by kinematic constraints.

Early works by Naganathan and Soni [62][63][64][65][66] developed a fully non-

linear model employing a kinematic representation with rigid link based reference. The three-dimensional model was constructed by accounting for axial, torsional, and lateral deformations. A Galerkin method was used with linear shape functions to represent the elasticity of the links. Link level matrices were transformed by time-varying compatibility matrices and cascaded into global matrices. The gross rigid body motion was specified at the revolute joints, and, subsequently, the element matrices were assumed constant at each time step in the numerical integration.

Simo and Vu-Quoc [83][84] presented a different problem which arose in simulating dynamic response of a flexible plane beam subject to large overall motions. Two orthogonal coordinates, measured in an inertial frame, were defined to account for the large overall rigid body motion and small elastic deformation. Hamilton's dynamics associated with a Galerkin spatial discretization were employed in the formulation, in which the use of finite strain rod theories capable of treating finite rotations was essential. The inherent nonlinear character of the problem was transferred to the stiffness part of the equations of motion, which resulted in the possibility of numerical implementation by means of any commercial finite element code able to analyze nonlinear structural dynamics.

An automated procedure in the study of large constrained flexible structures undergoing large specified motions was developed by Amirouche and Huston [1] [2]. The governing equations are derived using Kane's method [50]. The accommodation of the constraint equations is based on the use of orthogonal complement arrays. The flexibility and oscillations of the bodies are modeled using finite segment modeling, structure analysis, and scaling techniques. The generalized active forces are uncoupled in order to evaluate the corresponding stiffness matrix which is based upon the

local elastic deformation associated with equivalent beam deflections.

In the category of inverse dynamics, Bayo *et al* [11] [12] [13] extended the recursive Newton-Euler formulation for the inverse dynamics of flexible structural systems with open-loop chain. By their definition, the inverse dynamics is the finding of the driving input forces necessary to move the end effector of an open-loop system along a prescribed trajectory, avoiding any oscillations. The formulation is complete in the sense that it includes all the nonlinear terms due to the large rotations of the links. The Timoshenko beam theory is used to model the elastic characteristics, and the resulting equations of motion are discretized using the finite element method. An iterative scheme is proposed that relies on local linearization of the problem. The solution of each linearization is carried out in the frequency domain. The performance and capabilities of the technique were experimentally tested on a single-link flexible robot. In the category of forward dynamics, finding the kinematic properties providing given driving forces, Serna and Bayo [78] [79] [80] presented a series of penalty methods which incorporate the constraint equations in the Hamilton's principle by adding penalty functions to lead to a set of ordinary differential equations. This penalty method avoids the usual differential-algebraic equations as other methods using the same *one pass* coordinate representation, modeling both the large motions and small elastic deformations of the links simultaneously. The proposed methods formulate the equations of motion with respect to a floating frame that follows the rigid body motion of the links. The elastic links are discretized using a finite element method. The numerical implementation of the penalty methods was also presented and tested for a planar four-bar mechanism with specified rigid body motion.

Low and Vidyasagar [57] [58] [59] developed a systematic procedure for deriving

symbolic equations of motion for manipulators containing both rigid and flexible links. Hamilton's principle is employed in the derivation of the system equations which are expressed in the nonlinear and integro-differential scalar form. The formulation is based on expressing the kinetic and potential energies of the manipulator system in terms of generalized coordinates. The derivation proceeds for the cases of rigid and elastic links individually by means of defining rigid body coordinates and elastic deformation coordinates separately. In case of flexible links, the mass distribution and flexibility are taken into account. A simplified form of the elastic strain energy of the flexible links is formulated using the Euler-Bernoulli theory by assuming a slenderness ratio of the links and hence neglecting the effect of transverse shear and rotary inertia. An expression of the generalized force representing nonconservative forces is derived using Kane's method with a concept of partial velocity. The final resulting equations consist of the terms for inertia, Coriolis, centrifugal, gravitational, and exerted forces. In their numerical schemes, the partial differential equations are transferred into ordinary differential equations by the implementation of Galerkin's method.

A recursive Lagrangian approach was adopted by Book [15] [16] [17] in developing nonlinear equations of motion for flexible manipulator arms consisting of rotary joints that connect pairs of flexible links. Kinematics of both the rotary-joint motion and the link deformation are described by 4×4 transformation matrices. The elastic deflection is assumed small and the corresponding transformation is represented in terms of a summation of modal shapes. The equations are free from assumptions of a nominal motion and account for the interaction of angular rates and elastic deflections. In the assembly of link dynamic equations of motion, it is required to

take the inverse of the inertia matrix which can only be evaluated numerically.

In the work accomplished by Sadler and Yang [97][98][99], a total mechanism displacement was defined to reflect the large gross rigid body motion and small elastic deformation in the dynamic modeling. Example problems were demonstrated in two different categories: planar multi-link mechanisms and spatial robot manipulators. The effects of Rayleigh damping were introduced. In the mechanism applications, the authors claimed that the method could be adopted in the forward, as well as the inverse dynamic analyses if either the input forcing functions or the crank motion are specified. The link orientation angle must be related to a total unknown displacement in the formulation, which is possible for the mechanisms with one rigid body degree of freedom.

Turcic and Midha [89] [90] [91] developed the generalized equations of motion for elastic mechanism systems with given specified rigid body motion by utilizing general finite element theory. The derivation and final form of the equations of motion provide the capability to model a general two- or three-dimensional complex elastic mechanism, to include the nonlinear rigid-body and elastic motion coupling terms in a general representation, and to allow any finite element type to be used in the model. The equations of motion are first developed for a single finite element, then for a single link of the mechanism and finally for the entire mechanism system. The transition of the equations of motion from a lower level to an upper level is implemented by matrix assembly and transformation to ensure the displacement, velocity, and acceleration compatibility of all degrees-of-freedom composing the mechanism system. An orthogonal viscous damping is assumed and added to the coefficient matrix associated with the generalized velocity vector. The method is applied to

a four-bar mechanism with a geometrically complex follower link which is modeled with quadrilateral finite elements. The analytical results are verified by experimental measurement on the same mechanism.

More recently Nagarajan and Turcic [67] [68] [69] approached a new method to derive equations of motion for elastic mechanism systems. Both the rigid body and the elastic degrees of freedom were considered as generalized coordinates in the derivation. The equations of motion were first formulated based on element level coordinate systems in which elastic nodal displacements are measured. The equations were then transformed to a reference coordinate system to ensure compatibility of the displacement, velocity, and acceleration of the degrees of freedom that are common to two or more links during the assembly of the system equations of motion. Because the authors attempted to make the procedure general in their work, the equations at element and system levels are complicated, and the transformation from element level to system level takes a great amount of effort.

A concept of an equivalent rigid link system (ERLS) was presented by Chang and Hamilton [22] [23] in the dynamic analysis of robotic manipulators with flexible links. The basic idea of this hypothetical system is to separate the rigid-body dynamics and structural vibration. The global motion of the flexible link system is thereby separated into a large motion with a superimposed small motion. The large motion is represented by the ERLS and the small motion is due to the deviations with respect to the ERLS. A dynamic model was developed by means of finite element method and Lagrange's formulation. In the derivation of the equations of motion, the generalized coordinates are selected to represent the total motion as a nonlinear large motion and a linear small motion. The final global level system dynamic equations are grouped

into two sets of coupled nonlinear equations in which the equations representing small elastic motion are linear with respect to the small motion variables.

Cleghorn *et al* [31] [33] presented a procedure for determining the governing equations of a mechanism with physically undamped flexible links and distributed mass, operating at a prescribed input rotational speed. The Lagrangian equation is used in the derivation of the set of governing equations for the deflections about the rigid body nominal motion. The elastic members are discretized by the finite element method. The elemental level equations are transformed and assembled to generate a smaller set of global equations using the concept of connection compatibility for the adjacent members. The procedure is illustrated by using a planar four-bar angular function generating mechanism modeled with one finite element per link.

A literature survey of flexible modeling of multibody systems was completed by Cleghorn [40]. It was observed that the most effective model is one which incorporates Lagrange's equation with the finite element method. This produces a generalized element for easy application to flexible systems.

3.3 Current Approach

In the current study, a method combining Lagrangian dynamics with finite element analysis is developed in the modeling of dynamic response of multibody flexible structures. Lagrange's approach is used to develop the system dynamic equations. A finite element analysis associated with direct stiffness method is employed to discretize the elastic members in the system and to determine elastic degrees of freedom and the structural stiffness matrix which is required in finding the elastic strain energy. Each flexible beam is assumed to be slender and is therefore modeled using

beam elements. The generalized coordinates of an entire system reflect both the parameters from the nominal rigid body motion and the components of elastic displacements. The nonlinear coupling terms in all the coefficient matrices and the generalized force vectors are completely defined and formulated mathematically in detail. For an individual beam, the Lagrangian equation in matrix form [6] can be expressed as

$$\frac{d}{dt} \left(\frac{\partial KE_i}{\partial \dot{\mathbf{q}}_i^T} \right) - \frac{\partial KE_i}{\partial \mathbf{q}_i^T} + \frac{\partial PE_i}{\partial \mathbf{q}_i^T} = \mathbf{Q}_i \quad (3.1)$$

where KE_i and PE_i are the kinetic and potential energies of the beam, \mathbf{Q}_i are the nonconservative forces, and \mathbf{q}_i are the local generalized coordinates which reflect the degrees of freedom of the beam. A general expression of the i^{th} elastic beam kinetic energy modeled by finite elements can be written as

$$KE_i = \frac{1}{2} \sum_{g=1}^{N_i} \int_0^{l_g} \rho_i A_i \vec{V}_{ig} \cdot \vec{V}_{ig} ds \quad (3.2)$$

where N_i is the total number of finite elements, l_g is the length of the g^{th} element, ρ_i and A_i are the mass density and cross sectional area of the beam, and \vec{V}_{ig} is a generic velocity vector in element g . The above equation clearly shows that the velocity squared term plays a major role in kinetic energy. On the other hand, potential energy, consisting of body force potential energy as well as the structural strain energy, can be written as

$$PE_i = \frac{1}{2} \mathbf{q}_i^T \mathbf{K}_{si} \mathbf{q}_i + V_i(G) \quad (3.3)$$

where the first term is the elastic strain energy and the second term is a potential function which accounts for the beam elevation in the gravity field which takes into account both the macro rigid body motion and the micro elastic vibration. By

differentiation of the kinetic and potential energy terms and substitution of the results into Eq. 3.1, it can be shown that the equations of motion of the i^{th} beam take the following matrix form,

$$\mathbf{m}_i(\mathbf{q}_i)\ddot{\mathbf{q}}_i + \mathbf{c}_i(\mathbf{q}_i, \dot{\mathbf{q}}_i)\dot{\mathbf{q}}_i + \mathbf{k}_i(\mathbf{q}_i)\mathbf{q}_i = \mathbf{f}_i(\mathbf{q}_i) \quad (3.4)$$

In general, the mass matrix, \mathbf{m}_i , is a function of the generalized coordinates, \mathbf{q}_i ; the damping matrix, \mathbf{c}_i , which depends upon the *Coriolis* and *centrifugal* accelerations, is a function of the generalized coordinates and velocities; the stiffness matrix, \mathbf{k}_i , including the conventional structural stiffness, is a function of \mathbf{q}_i only; and the generalized force vector, \mathbf{f}_i , involving the external nonconservative forces acting on the beam, is also a function of \mathbf{q}_i only.

A set of global generalized coordinates, \mathbf{q} , is defined first. These coordinates are chosen from the local generalized coordinates, \mathbf{q}_i , such that every coordinate in \mathbf{q} is independent of every other. The relationship between the global and the local generalized coordinates is then determined by the following equation,

$$\mathbf{q}_i = \Phi_i \mathbf{q} \quad (3.5)$$

where Φ_i is a compatibility matrix which is in general a function of time. By differentiation of the above equation with respect to time followed by the substitution and pre-multiplication of Φ_i in Eq. 3.4, the system equations of motion can be obtained in the following form,

$$\mathbf{M}\ddot{\mathbf{q}} + \mathbf{C}\dot{\mathbf{q}} + \mathbf{K}\mathbf{q} = \mathbf{F} \quad (3.6)$$

where \mathbf{M} is a global mass matrix, \mathbf{C} is a global damping matrix, \mathbf{K} is a global stiffness matrix, and \mathbf{F} is a global force vector. In the following sections, more

detailed procedures and formulations are developed step by step. A demonstrative example is illustrated in Chapter 4 in which the simulation results are verified by experimental data.

3.4 Finite Element Modeling

Each elastic beam is modeled using several conventional predefined beam elements. There are a maximum of six degrees of freedom for each node in an element. This includes two orthogonal transverse deflections and two corresponding rotations, one longitudinal displacement, and one twist about the element axis. In order to achieve relatively simple modeling, only transverse deflections and rotations are allowed at each node. The contributions of the other two displacements are neglected in most cases (refer to Low [57] [58] [59] for a complete modeling). The following assumptions are therefore proposed for each element:

- Elementary beam theory applies and elastic flexure obeys Hooke's law;
- Each beam undergoes two uncoupled orthogonal deflections and rotations;
- Longitudinal displacement and axial twist are neglected.

Following a conventional direct stiffness method [56] [34] [100], a polynomial displacement function is presumed with knowledge of the external loadings. The boundary conditions are applied followed by direct application of the strain/stress relationships with sign conventions of the bending moments and shear forces. A structural stiffness matrix is obtained by comparing the relationship between the nodal forces and the nodal displacements.

3.4.1 Displacement function

It is indicated from classic elasticity theory [88] that a polynomial function of the static transverse deflection for a cantilever beam can be determined, depending on the type of external loads acting on the beam as shown in Figure 3.1. With no distributed loading, the highest order of the polynomial function is of order three, that is

$$y = a_0 + a_1x + a_2x^2 + a_3x^3 \quad (3.7)$$

where x denotes the axial coordinate of the beam, y is the corresponding transverse deflection, and $a_i (i = 0, 1, 2, 3)$ are the constant coefficients. The above formula is then adopted as a displacement function for each beam element.

3.4.2 Geometric boundary conditions

As illustrated in Figure 3.2, four geometric boundary conditions are proposed for each element as follows:

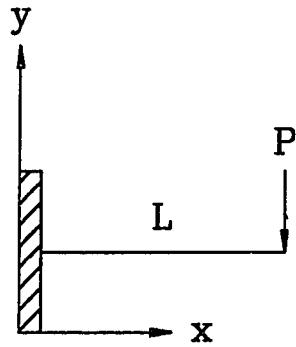
$$s = 0 \quad \text{deflection} = d_1, \quad \text{and} \quad \text{slope} = \phi_1$$

$$s = l \quad \text{deflection} = d_2, \quad \text{and} \quad \text{slope} = \phi_2$$

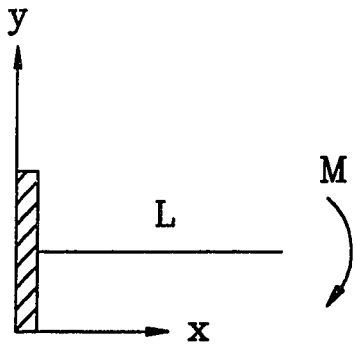
where s is the local axial coordinate in an undeformed element segment, d_i , and $\phi_i (i = 1, 2)$ are the transverse deflections and slopes at the corresponding nodes, respectively, and l is the length of the element. By applying the above four geometric boundary conditions to Eq. 3.7, it can be demonstrated that the final displacement functions in matrix form in each orthogonal plane are of the following forms,

$$v(s) = \mathbf{d}^T \mathbf{Y} \mathbf{s} = \mathbf{s}^T \mathbf{Y}^T \mathbf{d} \quad (3.8)$$

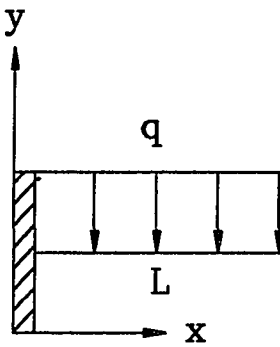
$$w(s) = \mathbf{d}^T \mathbf{Z} \mathbf{s} = \mathbf{s}^T \mathbf{Z}^T \mathbf{d} \quad (3.9)$$



$$y = - \frac{Px^2}{6EI} (3L - x)$$



$$y = - \frac{Mx^2}{2EI}$$



$$y = - \frac{qx^2}{24EI} (x^2 - 4Lx + 6L^2)$$

Figure 3.1: Functions of transverse deflection of a cantilever beam for three cases with different external loadings

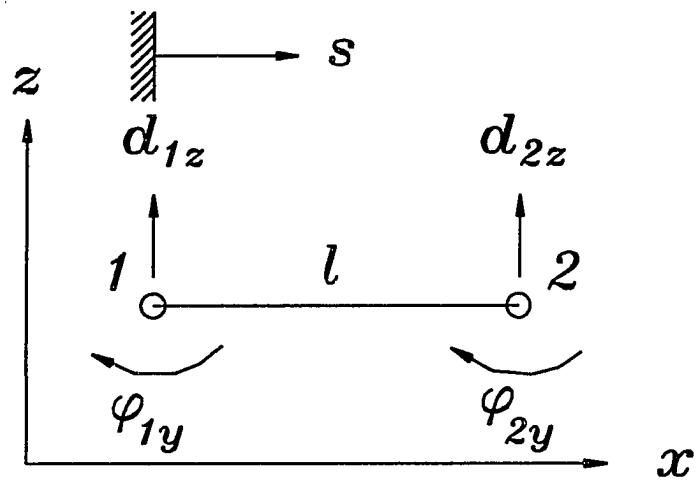
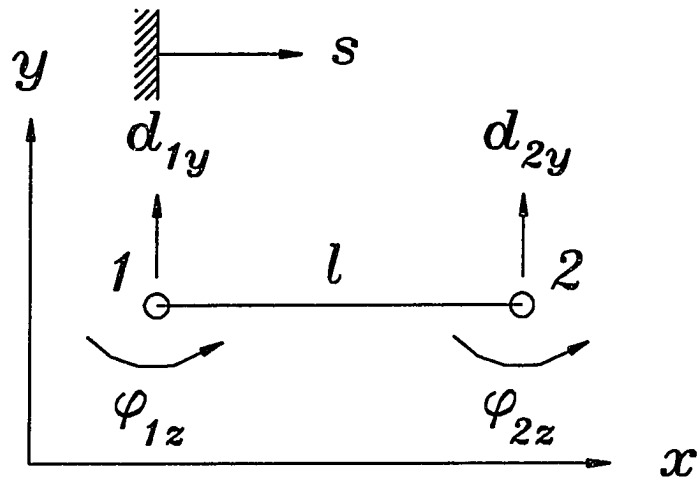


Figure 3.2: Sign conventions of nodal displacements

where \mathbf{Y} and \mathbf{Z} are the constant coefficient matrices (see Appendix C), and $v(s)$ and $w(s)$ are the displacement functions in the $X - Y$ and $X - Z$ planes, respectively. Also \mathbf{d} and \mathbf{s} are the generalized nodal coordinates of the element under consideration and a generalized function vector, respectively, which accordingly are defined as

$$\begin{aligned}\mathbf{d} &= \{d_{1y}\phi_{1z}d_{1z}\phi_{1y}d_{2y}\phi_{2z}d_{2z}\phi_{2y}\}^T \\ \mathbf{s} &= \{1 \ s \ s^2 \ s^3\}^T\end{aligned}\tag{3.10}$$

where d_{iy} and $\phi_{iz}(i = 1, 2)$ are the deflections and slopes in the $X - Y$ plane while d_{iz} and $\phi_{iy}(i = 1, 2)$ are the deflections and slopes in the $X - Z$ plane. It is noted that $\mathbf{s}^T\mathbf{Y}^T$ and $\mathbf{s}^T\mathbf{Z}^T$ in Eqs. 3.8 and 3.9 are the conventional shape functions of two orthogonal bendings.

3.4.3 Structural stiffness matrix

For small elastic deflection, the formulas for the bending moments and shear forces are found to be

$$M(s) = EI \frac{\partial^2 u(s)}{\partial s^2}\tag{3.11}$$

$$V(s) = EI \frac{\partial^3 u(s)}{\partial s^3}\tag{3.12}$$

where E is Young's modulus, I is the second moment of area, and $u(s)$ is a transverse deflection function (either $v(s)$ or $w(s)$). According to Eq. 3.10, a corresponding vector of generalized nodal forces to the vector of generalized nodal coordinates, \mathbf{d} , is defined as

$$\mathbf{f} = \{f_{1y}m_{1z}f_{1z}m_{1y}f_{2y}m_{2z}f_{2z}m_{2y}\}^T\tag{3.13}$$

where f_i and $m_i(i = 1, 2)$ are the nodal forces and moments, respectively, as shown in Figure 3.3. With respect to geometric boundary conditions, four force boundary

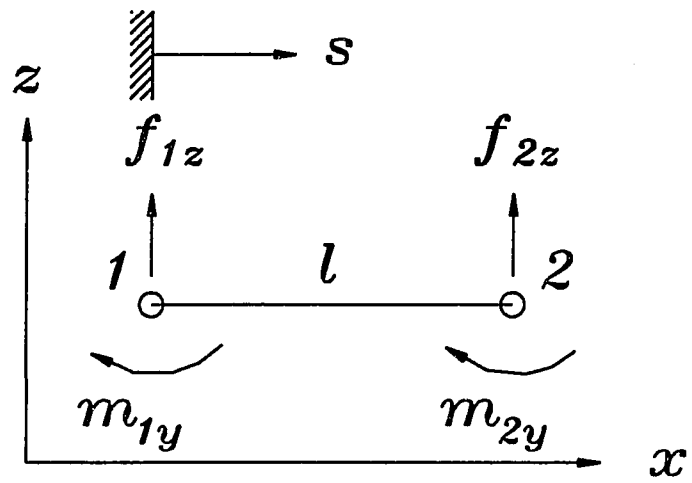
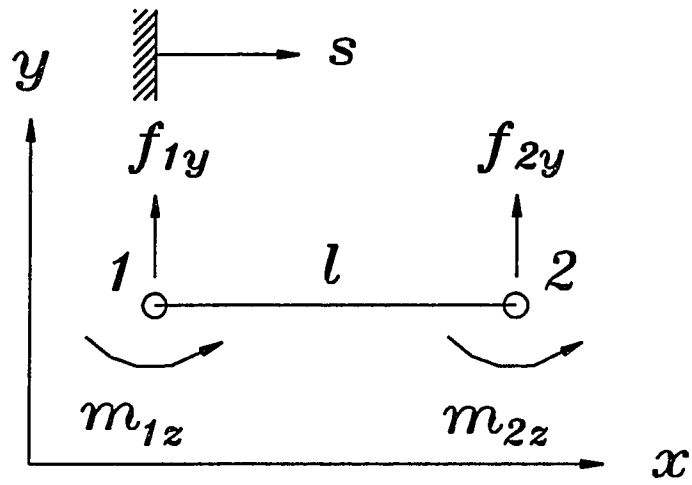


Figure 3.3: Sign conventions of nodal forces

conditions are accordingly determined as

$$s = 0 \quad \text{bending moment} = m_1, \quad \text{and} \quad \text{shear force} = f_1$$

$$s = l \quad \text{bending moment} = m_2, \quad \text{and} \quad \text{shear force} = f_2$$

By applying these four force boundary conditions for each orthogonal bending coordinate to Eq. 3.7 and arranging the results in the following standard form as

$$\mathbf{f} = \mathbf{k}_s \mathbf{d}, \quad (3.14)$$

it can be found that the structural stiffness matrix, \mathbf{k}_s , takes the following form,

$$\mathbf{k}_s = \frac{EI_a}{l^3} \left(C_z [\beta_1]^T \mathbf{k}_1 [\beta_1] + C_y [\beta_2]^T \mathbf{k}_2 [\beta_2] \right) \quad (3.15)$$

where \mathbf{k}_1 and \mathbf{k}_2 are the symmetric stiffness matrices, l is the length of the element, $I_a = (I_y + I_z)/2$ is the average second moment of area, $C_y = I_y/I_a$ and $C_z = I_z/I_a$ are the constant ratios, and $[\beta_1]$ and $[\beta_2]$ are the constant matrices (see Appendix C). The structural stiffness matrix is used in formulating the structural strain energy which is part of the potential energy of an elastic beam with kinematic motion.

3.5 Local Level Motion Equations

In the present chapter emphasis is placed on studying the dynamic response of spherical unconstrained structural systems. All the vectors defined in this dissertation are column vectors except where mentioned. Figure 3.4 shows a generic finite element, the g^{th} element, of an arbitrary elastic beam, the i^{th} beam, in such a structural system. In the rest of this chapter, the subscript i denotes the i^{th} beam and the subscript g denotes the g^{th} element in the i^{th} beam. Two sets of Cartesian coordinates are set up to assist the representation of the rigid body motion and elastic deformation. Set

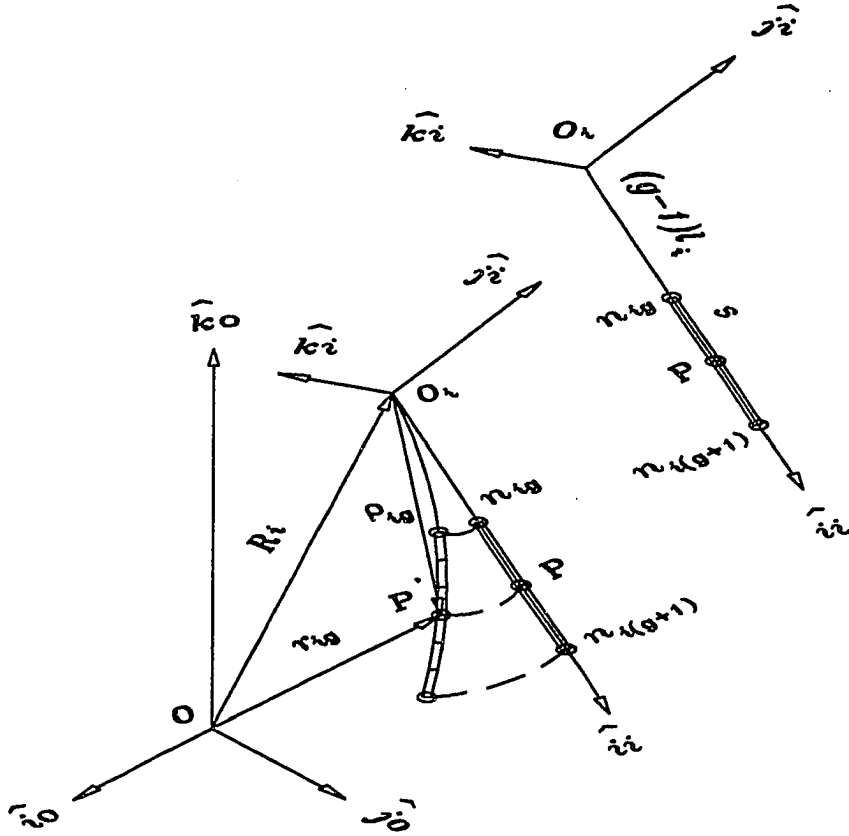


Figure 3.4: Coordinate systems and elastic deformation for a generic g^{th} element in an arbitrary i^{th} flexible beam

$(\hat{\mathbf{i}}_o, \hat{\mathbf{j}}_o, \hat{\mathbf{k}}_o)$ constitutes a floating (moving) frame of which the origin O is located at a spherical universal joint with three rotations. Set $(\hat{\mathbf{i}}_i, \hat{\mathbf{j}}_i, \hat{\mathbf{k}}_i)$, which accommodates the arbitrary elastic beam, is a reference frame which is relative to the moving frame. Vector \vec{R}_i is a position vector which indicates the position of the origin of the reference frame under consideration relative to the moving frame. This vector is considered as a rigid body position vector which describes the rigid body motion of the elastic beam. Vector $\vec{\rho}_{ig}$ is a local position vector measured in the reference frame for an arbitrary point P' in element g after its deformation. This $\vec{\rho}_{ig}$ vector includes both the rigid body motion of point P' relative to the moving frame and the elastic motion relative to the reference frame. Vector \vec{r}_{ig} , measured in the moving frame, is an absolute position vector which consists of the rigid body and elastic motions of point P' .

3.5.1 Position and velocity vectors

Referring to Figure 3.4 again, we can write the absolute position vector of point P' in the moving frame as

$$\begin{aligned}\vec{r}_{ig} &= \vec{R}_i + \vec{\rho}_{ig} \\ &= \hat{\mathbf{e}}_o^T (\mathbf{R}_i + \mathbf{T}_{oi} \{\rho\}_{ig})\end{aligned}\tag{3.16}$$

where $\hat{\mathbf{e}}_o = \{\hat{\mathbf{i}}_o, \hat{\mathbf{j}}_o, \hat{\mathbf{k}}_o\}^T$, a unit direction vector of the moving frame, \mathbf{R}_i and $\{\rho\}_{ig}$ are the reference and local position vectors in matrix form, and \mathbf{T}_{oi} denotes a 3×3 transformation matrix transferring the moving frame to the reference frame. That is, $\hat{\mathbf{e}}_o = \mathbf{T}_{oi} \hat{\mathbf{e}}_i$ where $\hat{\mathbf{e}}_i$ is a unit direction vector of the reference frame. A corresponding position equation in matrix form formulated in the moving frame, $(\hat{\mathbf{i}}_o, \hat{\mathbf{j}}_o, \hat{\mathbf{k}}_o)$, takes

the following form

$$\mathbf{r}_{ig} = \mathbf{R}_i + \mathbf{T}_{oi} \{\rho\}_{ig} \quad (3.17)$$

All the vectors in the following sections are also expressed relative to the same moving frame except where mentioned. Differentiation of Eq. 3.17 with respect to time gives a velocity formula which can be written as

$$\dot{\mathbf{r}}_{ig} = \tilde{\boldsymbol{\Omega}}(\mathbf{R}_i + \mathbf{T}_{oi} \{\rho\}_{ig}) + \dot{\mathbf{R}}_i + \mathbf{T}_{oi} \{\dot{\rho}\}_{ig} + \dot{\mathbf{T}}_{oi} \{\rho\}_{ig} \quad (3.18)$$

where $\dot{\mathbf{r}}_{ig}$ denotes $d\mathbf{r}_{ig}/dt$, $\dot{\mathbf{R}}_i$, $\{\dot{\rho}\}_{ig}$, and $\dot{\mathbf{T}}_{oi}$ are the time rates of the corresponding vectors and the transformation matrix, and $\tilde{\boldsymbol{\Omega}}$ is a skew-symmetric matrix (see Appendix B) derived from a rigid body system angular velocity, $\boldsymbol{\Omega}$ (see Chapter 4 for details), which can be expressed as

$$\boldsymbol{\Omega}^T = \dot{\mathbf{A}}^T \mathbf{N}^T = \dot{\mathbf{A}}^T [\mathbf{N}_1 \ \mathbf{N}_2 \ \mathbf{N}_3] \quad (3.19)$$

where $\dot{\mathbf{A}}$ is a generalized angular velocity vector containing the time rates of three rotating angles about the spherical universal joint, and \mathbf{N} is a 3×3 time-varying coefficient matrix which can be partitioned as $[\mathbf{N}_1 \mathbf{N}_2 \mathbf{N}_3]^T$. The rigid body system angular velocity, $\boldsymbol{\Omega}$, governs the angular motion of the moving frame, $(\hat{\mathbf{i}}_o, \hat{\mathbf{j}}_o, \hat{\mathbf{k}}_o)$, which is relative to an inertial frame, $(\hat{\mathbf{e}}_1, \hat{\mathbf{e}}_2, \hat{\mathbf{e}}_3)$. The origin of this inertial frame is also located at the universal joint, point O . Position vectors \mathbf{R}_i and $\{\rho\}_{ig}$ can also be partitioned and written as

$$\mathbf{R}_i^T = \{R_{i1} \ R_{i2} \ R_{i3}\}^T \quad (3.20)$$

$$\{\rho\}_{ig} = \begin{Bmatrix} \rho_{igs} \\ v_{ig} \\ w_{ig} \end{Bmatrix} = \begin{Bmatrix} (g-1)l_i + s \\ v_{ig} \\ w_{ig} \end{Bmatrix} \quad (3.21)$$

where g is an index of the g^{th} element, l_i is the element length, s is the local axial coordinate, R_{i1} , R_{i2} , and R_{i3} are three rigid body components of vector \mathbf{R}_i , ρ_{igs} is a rigid body component of vector $\{\rho\}_{ig}$, and v_{ig} and w_{ig} are two elastic components corresponding to two orthogonal deflections as shown in Eqs. 3.8 and 3.9. Therefore the time rates of the corresponding position vectors are found to be

$$\dot{\mathbf{R}}_i = \mathbf{0} \quad (3.22)$$

$$\{\dot{\rho}\}_{ig}^T = \{0 \ \dot{v}_{ig} \ \dot{w}_{ig}\} \quad (3.23)$$

For the cases with no revolute joint between elastic beams, the last term in Eq. 3.18 can be eliminated. A set of generalized coordinates for the i^{th} elastic beam is defined in terms of three rotation angles and generalized nodal displacements of each element. Thus,

$$\begin{aligned} \mathbf{q}_i^T &= \left\{ \Lambda^T d_{i1y} \phi_{i1z} d_{i1z} \phi_{i1y} d_{i2y} \phi_{i2z} d_{i2z} \phi_{i2y} \right. \\ &\quad \cdots d_{igy} \phi_{igz} d_{igz} \phi_{igy} d_{i(g+1)y} \phi_{i(g+1)z} d_{i(g+1)z} \phi_{i(g+1)y} \cdots \\ &\quad \left. d_{iN_iy} \phi_{iN_iz} d_{iN_iz} \phi_{iN_iy} d_{i(N_i+1)y} \phi_{i(N_i+1)z} d_{i(N_i+1)z} \phi_{i(N_i+1)y} \right\} \\ &= \left\{ \Lambda^T \mathbf{d}_{i1}^T \mathbf{d}_{i2}^T \cdots \mathbf{d}_{ig}^T \cdots \mathbf{d}_{iN_i}^T \right\} \end{aligned} \quad (3.24)$$

where N_i is the total number of elements of the i^{th} elastic beam. Each \mathbf{d}_{ig} ($g = 1, 2, \dots, N_i$) contains eight components as defined in Eq. 3.10. Attention must be paid that four nodal displacements are shared in the adjacent elements. That is,

$$\mathbf{d}_{i(g-1)}^T = \left\{ d_{i(g-1)y} \phi_{i(g-1)z} d_{i(g-1)z} \phi_{i(g-1)y} d_{igy} \phi_{igz} d_{igz} \phi_{igy} \right\}$$

and

$$\mathbf{d}_{ig}^T = \left\{ d_{igy} \phi_{igz} d_{igz} \phi_{igy} d_{i(g+1)y} \phi_{i(g+1)z} d_{i(g+1)z} \phi_{i(g+1)y} \right\}$$

The relationships between vectors Λ , \mathbf{d}_{ig} and \mathbf{q}_i are then established as

$$\Lambda = \Theta_{i\lambda} \mathbf{q}_i \quad (3.25)$$

$$\mathbf{d}_{ig} = \Theta_{igd} \mathbf{q}_i \quad (3.26)$$

where $\Theta_{i\lambda}$ and Θ_{igd} are types of linear compatibility matrices which can be derived by comparing Eqs. 3.25 and 3.26 with Eq. 3.24. A more compatible form of the velocity vector can be expressed as a function of the time rate of the generalized coordinates, $\dot{\mathbf{q}}$, by substituting Eqs. 3.8-3.10 and 3.19-3.26 into Eq. 3.18. After rearrangement, this produces

$$\begin{aligned} \dot{\mathbf{r}}_{ig} = & \begin{bmatrix} (R_{i3} + T_{31}\rho_{igs})\mathbf{N}_2 - (R_{i2} + T_{21}\rho_{igs})\mathbf{N}_3 \\ (R_{i1} + T_{11}\rho_{igs})\mathbf{N}_3 - (R_{i3} + T_{31}\rho_{igs})\mathbf{N}_1 \\ (R_{i2} + T_{21}\rho_{igs})\mathbf{N}_1 - (R_{i1} + T_{11}\rho_{igs})\mathbf{N}_2 \end{bmatrix} \Theta_{i\lambda} \dot{\mathbf{q}}_i + \\ & \begin{bmatrix} \mathbf{q}_i^T \Theta_{igd}^T ((T_{32}\mathbf{Y}_i + T_{33}\mathbf{Z}_i)\mathbf{sN}_2 - (T_{22}\mathbf{Y}_i + T_{23}\mathbf{Z}_i)\mathbf{sN}_3) \\ \mathbf{q}_i^T \Theta_{igd}^T ((T_{12}\mathbf{Y}_i + T_{13}\mathbf{Z}_i)\mathbf{sN}_3 - (T_{32}\mathbf{Y}_i + T_{33}\mathbf{Z}_i)\mathbf{sN}_1) \\ \mathbf{q}_i^T \Theta_{igd}^T ((T_{22}\mathbf{Y}_i + T_{23}\mathbf{Z}_i)\mathbf{sN}_1 - (T_{12}\mathbf{Y}_i + T_{13}\mathbf{Z}_i)\mathbf{sN}_2) \end{bmatrix} \Theta_{i\lambda} \dot{\mathbf{q}}_i + \\ & \begin{bmatrix} \mathbf{s}^T (\mathbf{Y}_i^T T_{12} + \mathbf{Z}_i^T T_{13}) \\ \mathbf{s}^T (\mathbf{Y}_i^T T_{22} + \mathbf{Z}_i^T T_{23}) \\ \mathbf{s}^T (\mathbf{Y}_i^T T_{32} + \mathbf{Z}_i^T T_{33}) \end{bmatrix} \Theta_{igd} \dot{\mathbf{q}}_i \end{aligned} \quad (3.27)$$

where $T_{\alpha\beta}(\alpha, \beta = 1, 2, 3)$ are the elements of the transformation matrix \mathbf{T}_{oi} (see Appendix A) and \mathbf{Y}_i and \mathbf{Z}_i are the constant matrices for the i^{th} beam defined in Eqs. 3.8 and 3.9.

3.5.2 Velocity squared term

The purpose of formulating the velocity squared term is to find the beam kinetic energy which is defined as

$$K E_i = \frac{1}{2} \sum_{g=1}^{N_i} \int_0^{l_g} \vec{V}_{ig} \cdot \vec{V}_{ig} dm_{ig} \quad (3.28)$$

where N_i is the total number of elements, l_g is the length of the g^{th} element in the i^{th} beam, and \vec{V}_{ig} is a velocity vector at an arbitrary point in the element. After substitution of $\rho_i A_i ds$ for dm_{ig} and $\dot{\mathbf{r}}_{ig}^T \hat{\mathbf{e}}_o$ or $\hat{\mathbf{e}}_o^T \dot{\mathbf{r}}_{ig}$ for \vec{V}_{ig} the above kinetic energy equation becomes

$$K E_i = \frac{1}{2} \rho_i A_i \sum_{g=1}^{N_i} \int_0^{l_g} \dot{\mathbf{r}}_{ig}^T \dot{\mathbf{r}}_{ig} ds \quad (3.29)$$

where ρ_i and A_i are the mass density and cross sectional area of the beam, respectively. Eq. 3.29 indicates that finding the velocity squared term should be accomplished prior to finding the kinetic energy. As shown in Eqs. 3.10 and 3.21, the velocity vector in Eq. 3.27 is also a function of the local axial coordinate, s . This complicates the problem since the velocity squared term must be formulated properly such that the integration in Eq. 3.29 can be carried out analytically. For simplicity, Eq. 3.27 is reformulated in symbolic fashion,

$$\dot{\mathbf{r}}_{ig} = [\mathbf{1}] \Theta_{i\lambda} \dot{\mathbf{q}}_i + [\mathbf{3}] \Theta_{i\lambda} \dot{\mathbf{q}}_i + [\mathbf{2}] \Theta_{igd} \dot{\mathbf{q}}_i \quad (3.30)$$

where matrices $[\mathbf{1}]$, $[\mathbf{2}]$, and $[\mathbf{3}]$ represent the corresponding coefficient matrices in Eq. 3.27 in the same order. Premultiplication of the velocity vector in Eq. 3.30 by its transpose vector will result in the velocity squared term as

$$\dot{\mathbf{r}}_{ig}^T \dot{\mathbf{r}}_{ig} = \dot{\mathbf{q}}_i^T \left\{ \Theta_{i\lambda}^T \left([\mathbf{1}]^T [\mathbf{1}] + [\mathbf{3}]^T [\mathbf{3}] + [\mathbf{1}]^T [\mathbf{3}] + [\mathbf{3}]^T [\mathbf{1}] \right) \Theta_{i\lambda} + \right.$$

$$\begin{aligned}
& \Theta_{igd}^T [\mathbf{2}]^T [\mathbf{2}] \Theta_{igd} + \Theta_{i\lambda}^T ([\mathbf{1}]^T + [\mathbf{3}]^T) [\mathbf{2}] \Theta_{igd} + \\
& \Theta_{igd}^T [\mathbf{2}]^T ([\mathbf{1}] + [\mathbf{3}]) \Theta_{i\lambda} \} \dot{\mathbf{q}}_i
\end{aligned} \tag{3.31}$$

By defining the following terms

$$\begin{aligned}
\mathbf{a} &= \begin{bmatrix} \mathbf{a}_1 \\ \mathbf{a}_2 \\ \mathbf{a}_3 \end{bmatrix} = \begin{bmatrix} T_{12} \mathbf{Y}_i + T_{13} \mathbf{Z}_i \\ T_{22} \mathbf{Y}_i + T_{23} \mathbf{Z}_i \\ T_{32} \mathbf{Y}_i + T_{33} \mathbf{Z}_i \end{bmatrix} \\
\mathbf{b} &= \begin{bmatrix} \mathbf{b}_1 \\ \mathbf{b}_2 \\ \mathbf{b}_3 \end{bmatrix} = \begin{bmatrix} \mathbf{s}^T \mathbf{a}_1^T \\ \mathbf{s}^T \mathbf{a}_2^T \\ \mathbf{s}^T \mathbf{a}_3^T \end{bmatrix} \\
\mathbf{R} &= \begin{bmatrix} R_1 \\ R_2 \\ R_3 \end{bmatrix} = \begin{bmatrix} R_{i1} + T_{11} \rho_{igs} \\ R_{i2} + T_{21} \rho_{igs} \\ R_{i3} + T_{31} \rho_{igs} \end{bmatrix}
\end{aligned}$$

where \mathbf{a}_1 , \mathbf{a}_2 , and \mathbf{a}_3 are the submatrices in the \mathbf{a} matrix, \mathbf{b}_1 , \mathbf{b}_2 , and \mathbf{b}_3 are the *row* vectors in the \mathbf{b} matrix, and \mathbf{R} is a type of position vector (which is different from \mathbf{R}_i). It can be demonstrated that the symbolic matrices $[\mathbf{1}]$, $[\mathbf{2}]$, and $[\mathbf{3}]$ in Eq. 3.30 become

$$\begin{aligned}
[\mathbf{1}] &= \tilde{\mathbf{R}}^T \mathbf{N} \\
[\mathbf{2}] &= \mathbf{b} \\
[\mathbf{3}] &= -[\tilde{\mathbf{b}}] \odot \Gamma(\Theta_{igd} \mathbf{q}_i) \mathbf{N}
\end{aligned}$$

where $\tilde{\mathbf{R}}^T$ is a transposed skew-symmetric matrix derived from the \mathbf{R} vector; $[\tilde{\mathbf{b}}]$ is a skew-symmetric matrix associated with the \mathbf{b} matrix; and Γ and \odot are the special

matrix operators defined in Appendix B. Thus,

$$\begin{aligned}
[1]^T [1] &= \mathbf{N}^T \tilde{\mathbf{R}} \tilde{\mathbf{R}}^T \mathbf{N} \\
[2]^T [2] &= \mathbf{b}^T \mathbf{b} \\
[3]^T [3] &= \mathbf{N}^T \Gamma \left(\mathbf{q}_i^T \boldsymbol{\Theta}_{igd}^T \right) \otimes \left([\tilde{\mathbf{b}}]^T [\tilde{\mathbf{b}}] \right) \otimes \Gamma \left(\boldsymbol{\Theta}_{igd} \mathbf{q}_i \right) \mathbf{N} \\
[1]^T [3] &= \mathbf{N}^T \tilde{\mathbf{R}}^T [\tilde{\mathbf{b}}] \otimes \Gamma \left(\boldsymbol{\Theta}_{igd} \mathbf{q}_i \right) \mathbf{N} \\
[1]^T [2] &= \mathbf{N}^T \tilde{\mathbf{R}} \mathbf{b} \\
[2]^T [3] &= \mathbf{b}^T [\tilde{\mathbf{b}}] \otimes \Gamma \left(\boldsymbol{\Theta}_{igd} \mathbf{q}_i \right) \mathbf{N}
\end{aligned}$$

By substituting above expressions into Eq. 3.31, it can be found that the velocity squared term is

$$\begin{aligned}
\dot{\mathbf{r}}_{ig}^T \dot{\mathbf{r}}_{ig} &= \dot{\mathbf{q}}_i^T \left\{ \boldsymbol{\Theta}_{i\lambda}^T \mathbf{N}^T \left[\Gamma \left(\mathbf{q}_i^T \boldsymbol{\Theta}_{igd}^T \right) \otimes \left([\tilde{\mathbf{b}}]^T [\tilde{\mathbf{b}}] \right) \otimes \Gamma \left(\boldsymbol{\Theta}_{igd} \mathbf{q}_i \right) + \right. \right. \\
&\quad \left. \tilde{\mathbf{R}} \tilde{\mathbf{R}}^T + \left(\tilde{\mathbf{R}}^T [\tilde{\mathbf{b}}] + [\tilde{\mathbf{b}}] \tilde{\mathbf{R}}^T \right) \otimes \Gamma \left(\boldsymbol{\Theta}_{igd} \mathbf{q}_i \right) \right] \mathbf{N} \boldsymbol{\Theta}_{i\lambda} + \\
&\quad \boldsymbol{\Theta}_{i\lambda}^T \mathbf{N}^T \left[\tilde{\mathbf{R}} \mathbf{b} + \Gamma^T \left(\boldsymbol{\Theta}_{igd} \mathbf{q}_i \right) \otimes [\tilde{\mathbf{b}}]^T \mathbf{b} \right] \boldsymbol{\Theta}_{igd} + \\
&\quad \boldsymbol{\Theta}_{igd}^T \left[\mathbf{b}^T \tilde{\mathbf{R}}^T + \mathbf{b}^T [\tilde{\mathbf{b}}] \otimes \Gamma \left(\boldsymbol{\Theta}_{igd} \mathbf{q}_i \right) \right] \mathbf{N} \boldsymbol{\Theta}_{i\lambda} + \\
&\quad \left. \boldsymbol{\Theta}_{igd}^T \mathbf{b}^T \mathbf{b} \boldsymbol{\Theta}_{igd} \right\} \dot{\mathbf{q}}_i
\end{aligned} \tag{3.32}$$

3.5.3 Kinetic energy

Substitution of Eq. 3.32 into the kinetic energy equation, Eq. 3.29, yields a more compact form of the kinetic energy as

$$K E_i = \frac{1}{2} \dot{\mathbf{q}}_i^T \mathbf{m}_i \dot{\mathbf{q}}_i \tag{3.33}$$

where \mathbf{m}_i is a symmetric mass matrix formulated as

$$\mathbf{m}_i = \mathbf{m}_{ic} + \boldsymbol{\Theta}_{i\lambda}^T \mathbf{N}^T (\mathbf{G}_{i1} + \mathbf{H}_{i1} + \mathbf{H}_{i2}) \mathbf{N} \boldsymbol{\Theta}_{i\lambda} +$$

$$\begin{aligned}
& \Theta_{i\lambda}^T \mathbf{N}^T (\mathbf{G}_{i2} + \mathbf{H}_{i3}) \Theta_{igd} + \\
& \Theta_{igd}^T (\mathbf{G}_{i2}^T + \mathbf{H}_{i3}^T) \mathbf{N} \Theta_{i\lambda}
\end{aligned} \tag{3.34}$$

with

$$\begin{aligned}
\mathbf{m}_{ic} &= \rho_i A_i \sum_{g=1}^{N_i} \Theta_{igd}^T \left(\int_0^{l_i} \mathbf{b}^T \mathbf{b} \, ds \right) \Theta_{igd} \\
\mathbf{G}_{i1} &= \rho_i A_i \sum_{g=1}^{N_i} \int_0^{l_i} \tilde{\mathbf{R}} \tilde{\mathbf{R}}^T \, ds \\
\mathbf{G}_{i2} &= \rho_i A_i \sum_{g=1}^{N_i} \int_0^{l_i} \tilde{\mathbf{R}} \mathbf{b} \, ds \\
\mathbf{H}_{i1} &= \rho_i A_i \sum_{g=1}^{N_i} \Gamma^T (\Theta_{igd} \mathbf{q}_i) \otimes \left(\int_0^{l_i} [\tilde{\mathbf{b}}]^T [\tilde{\mathbf{b}}] \, ds \right) \otimes \Gamma (\Theta_{igd} \mathbf{q}_i) \\
\mathbf{H}_{i2} &= \rho_i A_i \sum_{g=1}^{N_i} \int_0^{l_i} (\tilde{\mathbf{R}}^T [\tilde{\mathbf{b}}] + [\tilde{\mathbf{b}}] \tilde{\mathbf{R}}^T) \, ds \otimes \Gamma (\Theta_{igd} \mathbf{q}_i) \\
\mathbf{H}_{i3} &= \rho_i A_i \sum_{g=1}^{N_i} \Gamma^T (\Theta_{igd} \mathbf{q}_i) \otimes \int_0^{l_i} [\tilde{\mathbf{b}}]^T \mathbf{b} \, ds
\end{aligned} \tag{3.35}$$

where \mathbf{m}_{ic} is a constant symmetric mass matrix, \mathbf{G}_{i1} is a constant symmetric coefficient matrix, \mathbf{G}_{i2} is a constant rectangular matrix, \mathbf{H}_{i1} and \mathbf{H}_{i2} are the time-varying symmetric matrices, and \mathbf{H}_{i3} is a time-varying rectangular matrix.

3.5.4 Potential energy

The total potential energy of an elastic beam is defined as the summation of the body force potential energy and the elastic strain energy. The former is defined as the negative work done by gravity, *i.e.*

$$\begin{aligned}
U_{bi} &= \sum_{g=1}^{N_i} U_{big} \\
&= \sum_{g=1}^{N_i} \int_V -dm_i \vec{G} \cdot \vec{r}_{ig}
\end{aligned} \tag{3.36}$$

where $dm_i = \rho_i A_i ds$, $\vec{G} = -G\hat{e}_3$ in which \hat{e}_3 is a unit vector in the positive direction of a vertical axis of the inertial frame, $(\hat{e}_1, \hat{e}_2, \hat{e}_3)$, and \vec{r}_{ig} is a position vector as defined in the previous sections. Substitution of Eqs. 3.8, 3.9, 3.16, 3.20, 3.21, and 3.26 into Eq. 3.36 will result in a compact form of the body force potential energy in matrix form,

$$U_{bi} = V_i + \mathbf{h}_i^T \mathbf{q}_i \quad (3.37)$$

where V_i is a potential function which represents the rigid body potential energy, and \mathbf{h}_i^T is a force vector due to the elastic deflection. These two terms can further be formulated as

$$V_i = m_i G \mathbf{b}^T \mathbf{T}_{eo} \left(\mathbf{R}_i + \frac{L_i}{2} \mathbf{T}_{oi} \mathbf{a}_I \right)$$

$$\mathbf{h}_i^T = \frac{m_i G}{L_i} \mathbf{b}_I^T \mathbf{T}_{eo} \mathbf{T}_{oi} \int_0^{L_i} \begin{bmatrix} \mathbf{0} \\ \mathbf{s}^T \mathbf{Y}_i^T \\ \mathbf{s}^T \mathbf{Z}_i^T \end{bmatrix} ds \sum_{g=1}^{N_i} \Theta_{igd}$$

where m_i is the mass of the beam, L_i is the length of the beam, \mathbf{T}_{eo} is a time-varying transformation matrix between the inertial frame and the moving frame (see Appendix A), $\mathbf{a}_I^T = \{1 \ 0 \ 0\}$, and $\mathbf{b}_I^T = \{0 \ 0 \ 1\}$. The elastic strain energy is defined as

$$\begin{aligned} U_{ei} &= \sum_{g=1}^{N_i} U_{eig} \\ &= \frac{1}{2} \sum_{g=1}^{N_i} \mathbf{q}_i^T \left(\Theta_{igd}^T \mathbf{k}_{sig} \Theta_{igt} \right) \mathbf{q}_i \end{aligned} \quad (3.38)$$

where \mathbf{k}_{sig} is a structural stiffness matrix of the g^{th} element in the i^{th} elastic beam as shown in Eq. 3.15. Therefore, the total potential energy can be found as

$$PE_i = \frac{1}{2} \mathbf{q}_i^T \mathbf{k}_{si} \mathbf{q}_i + V_i + \mathbf{h}_i^T \mathbf{q}_i \quad (3.39)$$

with $\mathbf{k}_{si} = \sum_{g=1}^{N_i} \mathbf{\Theta}_{igd}^T \mathbf{k}_{sig} \mathbf{\Theta}_{igd}$, a constant symmetric stiffness matrix of the beam.

3.5.5 Motion equations

By substituting the formulas of kinetic and potential energies in Eqs. 3.33 and 3.39 into the Lagrange's equation, Eq. 3.1, it is found that the equations of motion of an arbitrary free elastic beam at the local level are

$$\mathbf{m}_i \ddot{\mathbf{q}}_i + \left[\dot{\mathbf{m}}_i - \frac{1}{2} \frac{\partial(\mathbf{m}_i \dot{\mathbf{q}}_i)}{\partial \mathbf{q}_i^T} \right] \dot{\mathbf{q}}_i + \left(\mathbf{k}_{si} + \frac{\partial \mathbf{h}_i}{\partial \mathbf{q}_i^T} \right) \mathbf{q}_i = \mathbf{Q}_i - \mathbf{h}_i - \frac{\partial V_i}{\partial \mathbf{q}_i^T} \quad (3.40)$$

The above equations clearly show the nonlinearity involved in the time-varying coefficient matrices. Referring to Appendix B, some of the partial derivatives can be derived immediately in the following forms,

$$\frac{\partial \mathbf{h}_i}{\partial \mathbf{q}_i^T} = \frac{m_i G}{L_i} \frac{\partial(\mathbf{T}_{eo}^T \mathbf{b}_I)}{\partial \mathbf{q}_i^T} \mathbf{T}_{oi} \int_0^{l_i} \begin{bmatrix} \mathbf{0} \\ \mathbf{s}^T \mathbf{Y}_i^T \\ \mathbf{s}^T \mathbf{Z}_i^T \end{bmatrix} ds \sum_{g=1}^{N_i} \mathbf{\Theta}_{igd} \quad (3.41)$$

$$\frac{\partial V_i}{\partial \mathbf{q}_i^T} = m_i G \frac{\partial(\mathbf{T}_{eo}^T \mathbf{b}_I)}{\partial \mathbf{q}_i^T} \left(\mathbf{R}_i + \frac{L_i}{2} \mathbf{T}_{oi} \mathbf{a}_I \right) \quad (3.42)$$

The partial derivatives on the right hand sides of the above equations can be carried out analytically by substitution of the specific transformation matrix for \mathbf{T}_{eo} .

3.5.6 Derivation of $\frac{1}{2} \partial(\mathbf{m}_i \dot{\mathbf{q}}_i) / \partial \mathbf{q}_i^T$

This is one of the most difficult and challenging tasks addressed in this and the following chapters. It is necessary to derive the mathematical formula representing nonlinear coupling terms between the rigid body motion and elastic vibration in the procedures of the derivation of the equations of motion for the elastic and rigid bodies.

Referring to Eq. 3.33, the formula for kinetic energy of the i^{th} beam is rearranged, as

$$\begin{aligned} KE_i &= \frac{1}{2} \dot{\mathbf{q}}_i^T \mathbf{m}_i \dot{\mathbf{q}}_i \\ &= KE_{ic} + KE_{i1} + 2KE_{i2} \end{aligned} \quad (3.43)$$

The constant part of the kinetic energy in the above equation is written as

$$KE_{ic} = \frac{1}{2} \dot{\mathbf{q}}_i^T \mathbf{m}_{ic} \dot{\mathbf{q}}_i \quad (3.44)$$

The last two terms in Eq. 3.43 account for the time-varying part of the kinetic energy and are written as

$$KE_{i1} = \frac{1}{2} \dot{\mathbf{q}}_i^T \Theta_{i\lambda}^T \mathbf{N}^T (\mathbf{G}_{i1} + \mathbf{H}_{i1} + \mathbf{H}_{i2}) \mathbf{N} \Theta_{i\lambda} \dot{\mathbf{q}}_i \quad (3.45)$$

$$KE_{i2} = \frac{1}{2} \dot{\mathbf{q}}_i^T \Theta_{i\lambda}^T \mathbf{N}^T (\mathbf{G}_{i2} + \mathbf{H}_{i3}) \Theta_{igd} \dot{\mathbf{q}}_i \quad (3.46)$$

Partial differentiation of Eq. 3.43 with respect to \mathbf{q}_i^T gives

$$\begin{aligned} \frac{\partial KE_i}{\partial \mathbf{q}_i^T} &= \frac{1}{2} \frac{\partial (\mathbf{m}_i \dot{\mathbf{q}}_i)}{\partial \mathbf{q}_i^T} \dot{\mathbf{q}}_i \\ &= \frac{\partial KE_{ic}}{\partial \mathbf{q}_i^T} + \frac{\partial KE_{i1}}{\partial \mathbf{q}_i^T} + 2 \frac{\partial KE_{i2}}{\partial \mathbf{q}_i^T} \end{aligned} \quad (3.47)$$

where the first term vanishes because \mathbf{m}_{ic} in Eq. 3.44 is a constant matrix. Matrices \mathbf{H}_{i1} , \mathbf{H}_{i2} , and \mathbf{H}_{i3} , referred to Eq. 3.35, are defined as

$$\mathbf{H}_{i1} = [\mathbf{H}_{i1,1} \mathbf{H}_{i1,2} \mathbf{H}_{i1,3}] = \Gamma^T(\mathbf{q}_i) \otimes \begin{bmatrix} \mathbf{B}_{11} & \mathbf{B}_{12} & \mathbf{B}_{13} \\ \mathbf{B}_{21} & \mathbf{B}_{22} & \mathbf{B}_{23} \\ \mathbf{B}_{31} & \mathbf{B}_{32} & \mathbf{B}_{33} \end{bmatrix} \otimes \Gamma(\mathbf{q}_i) \quad (3.48)$$

$$\mathbf{H}_{i2} = [\mathbf{H}_{i2,1} \mathbf{H}_{i2,2} \mathbf{H}_{i2,3}] = \begin{bmatrix} \mathbf{D}_{11} & \mathbf{D}_{12} & \mathbf{D}_{13} \\ \mathbf{D}_{21} & \mathbf{D}_{22} & \mathbf{D}_{23} \\ \mathbf{D}_{31} & \mathbf{D}_{32} & \mathbf{D}_{33} \end{bmatrix} \otimes \Gamma(\mathbf{q}_i) \quad (3.49)$$

$$\mathbf{H}_{i3} = [\mathbf{H}_{i3,1}\mathbf{H}_{i3,2}\mathbf{H}_{i3,3}]^T = \begin{bmatrix} \mathbf{q}_i^T \mathbf{E}_1^T \\ \mathbf{q}_i^T \mathbf{E}_2^T \\ \mathbf{q}_i^T \mathbf{E}_3^T \end{bmatrix} \quad (3.50)$$

By substituting Eqs. 3.48 and 3.49 into Eq. 3.45, the second term in Eq. 3.47 can be written as

$$\begin{aligned} \frac{\partial K E_{i1}}{\partial \mathbf{q}_i^T} &= \frac{\partial(\mathbf{N}\Theta_{i\lambda}\dot{\mathbf{q}}_i)}{\partial \mathbf{q}_i^T} (\mathbf{G}_{i1} + \mathbf{H}_{i1} + \mathbf{H}_{i2}) \mathbf{N}\Theta_{i\lambda}\dot{\mathbf{q}}_i + \\ &\quad \frac{1}{2} \left[\begin{array}{l} \frac{\partial(\mathbf{H}_{i1,1} + \mathbf{H}_{i2,1})}{\partial \mathbf{q}_i^T} \mathbf{N}\Theta_{i\lambda}\dot{\mathbf{q}}_i \mid \\ \mid \frac{\partial(\mathbf{H}_{i1,2} + \mathbf{H}_{i2,2})}{\partial \mathbf{q}_i^T} \mathbf{N}\Theta_{i\lambda}\dot{\mathbf{q}}_i \mid \\ \mid \frac{\partial(\mathbf{H}_{i1,3} + \mathbf{H}_{i2,3})}{\partial \mathbf{q}_i^T} \mathbf{N}\Theta_{i\lambda}\dot{\mathbf{q}}_i \end{array} \right] \mathbf{N}\Theta_{i\lambda}\dot{\mathbf{q}}_i \\ &= [\mathbf{X}_1] + [\mathbf{X}_2] \end{aligned} \quad (3.51)$$

where the first symbolic matrix, $[\mathbf{X}_1]$, corresponds to the first term, and the second symbolic matrix, $[\mathbf{X}_2]$, represents the second term. By substituting the matrix partition form of \mathbf{N} in Eq. 3.19 into Eq. 3.51, these two symbolic matrices can be formulated as

$$[\mathbf{X}_1] = \left[\frac{\partial \mathbf{N}_1}{\partial \mathbf{q}_i^T} \Theta_{i\lambda} \dot{\mathbf{q}}_i \mid \frac{\partial \mathbf{N}_2}{\partial \mathbf{q}_i^T} \Theta_{i\lambda} \dot{\mathbf{q}}_i \mid \frac{\partial \mathbf{N}_3}{\partial \mathbf{q}_i^T} \Theta_{i\lambda} \dot{\mathbf{q}}_i \right] \cdot (\mathbf{G}_{i1} + \mathbf{H}_{i1} + \mathbf{H}_{i2}) \mathbf{N}\Theta_{i\lambda}\dot{\mathbf{q}}_i$$

$$\begin{aligned} [\mathbf{X}_2] &= \frac{1}{2} \sum_{\beta=1}^3 \frac{\partial(\mathbf{H}_{i1,\beta} + \mathbf{H}_{i2,\beta})}{\partial \mathbf{q}_i^T} \mathbf{N}\Theta_{i\lambda} \dot{\mathbf{q}}_i \mathbf{N}_\beta^T \Theta_{i\lambda} \dot{\mathbf{q}}_i \\ &= \frac{1}{2} \sum_{\beta=1}^3 [\mathbf{X}_3] \mathbf{N}\Theta_{i\lambda} \dot{\mathbf{q}}_i \mathbf{N}_\beta^T \Theta_{i\lambda} \dot{\mathbf{q}}_i \end{aligned}$$

where $[\mathbf{X}_3]$ is a symbolic matrix which represents a partial derivative term in $[\mathbf{X}_2]$ and is written as

$$\begin{aligned} [\mathbf{X}_3] &= \frac{\partial}{\partial \mathbf{q}_i^T} \left(\begin{Bmatrix} \mathbf{q}_i^T \mathbf{B}_{1\beta} \mathbf{q}_i \\ \mathbf{q}_i^T \mathbf{B}_{2\beta} \mathbf{q}_i \\ \mathbf{q}_i^T \mathbf{B}_{3\beta} \mathbf{q}_i \end{Bmatrix} + \begin{Bmatrix} \mathbf{D}_{1\beta} \mathbf{q}_i \\ \mathbf{D}_{2\beta} \mathbf{q}_i \\ \mathbf{D}_{3\beta} \mathbf{q}_i \end{Bmatrix} \right) \\ &= [(\mathbf{B}_{1\beta} + \mathbf{B}_{\beta 1}) \mathbf{q}_i + \mathbf{D}_{1\beta}^T | (\mathbf{B}_{2\beta} + \mathbf{B}_{\beta 2}) \mathbf{q}_i + \mathbf{D}_{2\beta}^T \\ &\quad | (\mathbf{B}_{3\beta} + \mathbf{B}_{\beta 3}) \mathbf{q}_i + \mathbf{D}_{3\beta}^T] \end{aligned}$$

with $\mathbf{B}_{\alpha\beta}^T = \mathbf{B}_{\beta\alpha}$ ($\alpha = 1, 2, 3$) and $\mathbf{D}_{\alpha\beta} = \mathbf{D}_{\beta\alpha}$ ($\alpha = 1, 2, 3$). Substitution of the expression of $[\mathbf{X}_3]$ into $[\mathbf{X}_2]$ gives

$$[\mathbf{X}_2] = \frac{1}{2} \sum_{\beta=1}^3 \sum_{\alpha=1}^3 \left\{ (\mathbf{B}_{\alpha\beta} + \mathbf{B}_{\beta\alpha}) \mathbf{q}_i + \mathbf{D}_{\alpha\beta}^T \right\} \dot{\mathbf{q}}_i^T \boldsymbol{\Theta}_{i\lambda}^T \mathbf{N}_\alpha \mathbf{N}_\beta^T \boldsymbol{\Theta}_{i\lambda} \dot{\mathbf{q}}_i$$

Substitution of the expressions of $[\mathbf{X}_1]$ and $[\mathbf{X}_2]$ into Eq. 3.51 yields

$$\begin{aligned} \frac{\partial K E_{i1}}{\partial \mathbf{q}_i^T} &= \left[\frac{\partial \mathbf{N}_1}{\partial \mathbf{q}_i^T} \boldsymbol{\Theta}_{i\lambda} \dot{\mathbf{q}}_i \mid \frac{\partial \mathbf{N}_2}{\partial \mathbf{q}_i^T} \boldsymbol{\Theta}_{i\lambda} \dot{\mathbf{q}}_i \mid \frac{\partial \mathbf{N}_3}{\partial \mathbf{q}_i^T} \boldsymbol{\Theta}_{i\lambda} \dot{\mathbf{q}}_i \right] \cdot \\ &\quad (\mathbf{G}_{i1} + \mathbf{H}_{i1} + \mathbf{H}_{i2}) \mathbf{N} \boldsymbol{\Theta}_{i\lambda} \dot{\mathbf{q}}_i + \\ &\quad \frac{1}{2} \sum_{\beta=1}^3 \sum_{\alpha=1}^3 \left\{ (\mathbf{B}_{\alpha\beta} + \mathbf{B}_{\beta\alpha}) \mathbf{q}_i + \mathbf{D}_{\alpha\beta}^T \right\} \cdot \\ &\quad \dot{\mathbf{q}}_i^T \boldsymbol{\Theta}_{i\lambda}^T \mathbf{N}_\alpha \mathbf{N}_\beta^T \boldsymbol{\Theta}_{i\lambda} \dot{\mathbf{q}}_i \end{aligned} \quad (3.52)$$

By substituting Eq. 3.50 into Eq. 3.46, the last term in Eq. 3.47 can be written as

$$\begin{aligned} 2 \frac{\partial K E_{i2}}{\partial \mathbf{q}_i^T} &= \frac{\partial (\mathbf{N} \boldsymbol{\Theta}_{i\lambda} \dot{\mathbf{q}}_i)}{\partial \mathbf{q}_i^T} (\mathbf{G}_{i2} + \mathbf{H}_{i3}) \boldsymbol{\Theta}_{igd} \dot{\mathbf{q}}_i + \\ &\quad \frac{\partial \{(\mathbf{G}_{i2} + \mathbf{H}_{i3}) \boldsymbol{\Theta}_{igd} \dot{\mathbf{q}}_i\}}{\partial \mathbf{q}_i^T} \mathbf{N} \boldsymbol{\Theta}_{i\lambda} \dot{\mathbf{q}}_i \\ &= \left[\frac{\partial \mathbf{N}_1}{\partial \mathbf{q}_i^T} \boldsymbol{\Theta}_{i\lambda} \dot{\mathbf{q}}_i \mid \frac{\partial \mathbf{N}_2}{\partial \mathbf{q}_i^T} \boldsymbol{\Theta}_{i\lambda} \dot{\mathbf{q}}_i \mid \frac{\partial \mathbf{N}_3}{\partial \mathbf{q}_i^T} \boldsymbol{\Theta}_{i\lambda} \dot{\mathbf{q}}_i \right] (\mathbf{G}_{i2} + \mathbf{H}_{i3}) \boldsymbol{\Theta}_{igd} \dot{\mathbf{q}}_i \\ &\quad + [\mathbf{E}_1^T \boldsymbol{\Theta}_{igd} \dot{\mathbf{q}}_i \mid \mathbf{E}_2^T \boldsymbol{\Theta}_{igd} \dot{\mathbf{q}}_i \mid \mathbf{E}_3^T \boldsymbol{\Theta}_{igd} \dot{\mathbf{q}}_i] \mathbf{N} \boldsymbol{\Theta}_{i\lambda} \dot{\mathbf{q}}_i \end{aligned} \quad (3.53)$$

Finally, substitution of Eqs. 3.52 and 3.53 into Eq. 3.47 will result in the following expression for the matrix partial derivatives, as

$$\begin{aligned}
\frac{1}{2} \frac{\partial(\mathbf{m}_i \dot{\mathbf{q}}_i)}{\partial \mathbf{q}_i^T} &= \left[\frac{\partial \mathbf{N}_1}{\partial \mathbf{q}_i^T} \boldsymbol{\Theta}_{i\lambda} \dot{\mathbf{q}}_i \mid \frac{\partial \mathbf{N}_2}{\partial \mathbf{q}_i^T} \boldsymbol{\Theta}_{i\lambda} \dot{\mathbf{q}}_i \mid \frac{\partial \mathbf{N}_3}{\partial \mathbf{q}_i^T} \boldsymbol{\Theta}_{i\lambda} \dot{\mathbf{q}}_i \right] \cdot \\
&\quad \{(\mathbf{G}_{i1} + \mathbf{H}_{i1} + \mathbf{H}_{i2}) \mathbf{N} \boldsymbol{\Theta}_{i\lambda} + (\mathbf{G}_{i2} + \mathbf{H}_{i3}) \boldsymbol{\Theta}_{igd}\} + \\
&\quad \left[\mathbf{E}_1^T \boldsymbol{\Theta}_{igd} \dot{\mathbf{q}}_i \mid \mathbf{E}_2^T \boldsymbol{\Theta}_{igd} \dot{\mathbf{q}}_i \mid \mathbf{E}_3^T \boldsymbol{\Theta}_{igd} \dot{\mathbf{q}}_i \right] \mathbf{N} \boldsymbol{\Theta}_{i\lambda} + \\
&\quad \frac{1}{2} \sum_{\beta=1}^3 \sum_{\alpha=1}^3 \left\{ (\mathbf{B}_{\alpha\beta} + \mathbf{B}_{\beta\alpha}) \mathbf{q}_i + \mathbf{D}_{\alpha\beta}^T \right\} \cdot \\
&\quad \dot{\mathbf{q}}_i^T \boldsymbol{\Theta}_{i\lambda}^T \mathbf{N}_\alpha \mathbf{N}_\beta^T \boldsymbol{\Theta}_{i\lambda} \dot{\mathbf{q}}_i
\end{aligned} \tag{3.54}$$

where matrices $\mathbf{B}_{\alpha\beta}$, $\mathbf{D}_{\alpha\beta}$, and \mathbf{E}_α are defined by comparing Eqs. 3.48, 3.49, and 3.50 with Eq. 3.35.

3.6 Global Level Motion Equations

In the previous sections, the local level equations of motion were derived for an arbitrary elastic beam with no kinematic constraints. In order that those generalized coordinates at the common connecting boundaries are consistent for the adjacent beams, it is necessary to include the kinematic constraints in the equations of motion. The concept of the compatibility matrix (as defined in the following section) is adopted in the assembly process so that the coefficient matrices and the generalized force vectors of each subsystem are compatible. A set of global generalized coordinates is selected among the local generalized coordinates of each subsystem such that the global generalized coordinates are independent of each other.

3.6.1 Compatibility matrix

A matrix which linearly relates the local generalized coordinates to the global generalized coordinates is called the *compatibility matrix*. For a system with n global generalized coordinates, \mathbf{q} , the local generalized coordinates, \mathbf{q}_i , with m components can be expressed as

$$\mathbf{q}_i = \Phi_i \mathbf{q} \quad (i = 1, 2, \dots, N) \quad (3.55)$$

Where N is the total number of subsystems under consideration. The compatibility matrix, Φ_i , is an $m \times n$ matrix which is in general a time-varying function of the rigid body generalized coordinates. It is apparent that the compatibility matrix contains the information of the geometric boundary conditions which describe the kinematic constraints for the adjacent subsystems.

3.6.2 Assembly of motion equations

Differentiation of Eq. 3.55 with respect to time leads to

$$\dot{\mathbf{q}}_i = \Phi_i \dot{\mathbf{q}} + \dot{\Phi}_i \mathbf{q} \quad (3.56)$$

$$\ddot{\mathbf{q}}_i = \Phi_i \ddot{\mathbf{q}} + 2\dot{\Phi}_i \dot{\mathbf{q}} + \ddot{\Phi}_i \mathbf{q} \quad (3.57)$$

By rearranging the local level equations of motion, it can be shown that Eq. 3.40 takes the following standard form

$$\mathbf{m}_i \ddot{\mathbf{q}}_i + \mathbf{c}_i \dot{\mathbf{q}}_i + \mathbf{k}_i \mathbf{q}_i = \mathbf{f}_i \quad (3.58)$$

where \mathbf{c}_i is a damping matrix, \mathbf{k}_i is a stiffness matrix, and \mathbf{f}_i is a generalized force vector. These matrices are formulated as

$$\mathbf{c}_i = \dot{\mathbf{m}}_i - \frac{1}{2} \frac{\partial(\mathbf{m}_i \dot{\mathbf{q}}_i)}{\partial \mathbf{q}_i^T}$$

$$\begin{aligned} \mathbf{k}_i &= \mathbf{k}_{si} + \frac{\partial \mathbf{h}_i}{\partial \mathbf{q}_i^T} \\ \mathbf{f}_i &= \mathbf{Q}_i - \mathbf{h}_i - \frac{\partial V_i}{\partial \mathbf{q}_i^T} \end{aligned}$$

Substitution of Eqs. 3.56 and 3.57 into Eq. 3.58 followed by pre-multiplication of Eq. 3.58 by the transpose compatibility matrix, Φ_i^T , will result in the following global equations of motion

$$\mathbf{M} \ddot{\mathbf{q}} + \mathbf{C} \dot{\mathbf{q}} + \mathbf{K} \mathbf{q} = \mathbf{F} \quad (3.59)$$

where the global mass, damping, and stiffness matrices and the global generalized force vector are formulated as

$$\begin{aligned} \mathbf{M} &= \sum_{i=1}^N \Phi_i^T \mathbf{m}_i \Phi_i \\ \mathbf{C} &= \sum_{i=1}^N \left(\Phi_i^T \mathbf{c}_i \Phi_i + 2\Phi_i^T \mathbf{m}_i \dot{\Phi}_i \right) \\ \mathbf{K} &= \sum_{i=1}^N \left(\Phi_i^T \mathbf{k}_i \Phi_i + \Phi_i^T \mathbf{c}_i \dot{\Phi}_i + \Phi_i^T \mathbf{m}_i \ddot{\Phi}_i \right) \\ \mathbf{F} &= \sum_{i=1}^N \Phi_i^T \mathbf{f}_i \end{aligned} \quad (3.60)$$

Structural and fluid viscous damping terms of each subsystem can be added in each \mathbf{c}_i matrix. The internal connecting force terms in each local level generalized force vector, \mathbf{f}_i , vanish automatically during the process of matrix assembly.

3.7 Summary

A systematic procedure of mathematical modeling of an arbitrary elastic beam in a multibody system was fully developed in the present chapter. The natural characteristics of mutually coupled rigid body and elastic motions were revealed by

including the unknown rigid body degrees of freedom in the global generalized coordinates. The significant complexity involved in mathematical formulation arose because of the involvement of the unknown rigid body degrees of freedom. Nonlinear coupling terms due to Coriolis and centrifugal forces, which were neglected historically, were completely taken into account and were derived explicitly in matrix form. The conventional finite element analysis associated with the direct stiffness method was used in the discretization of the elastic members. A third order polynomial function was adopted in the finite element shape function in order to exclude the negligible effect of the longitudinal displacement and axial twisting which usually consist of higher order terms compared with the other deformation. The Lagrange's equation was employed in which both the rigid body and the elastic degrees of freedom were treated as unknown generalized coordinates of the system. The elastic deformation of every element in each elastic beam were measured in the local reference frame so that they are compatible at the local level. The position vector as well as the velocity vector were formulated in terms of the moving frame instead of the usual inertial frame. This resulted in simple mathematical operations in finding kinetic and potential energies.

The final form of the system dynamic equations of motion was expressed in an analytical form which shows high nonlinearity and strong contributions of the coupling terms in the time-varying coefficient matrices and generalized force terms. The procedure and methodology developed herein are applicable to the dynamic modeling of planar mechanisms, as well as the unconstrained spatial structures. The application of the theory presented in the current chapter will be demonstrated and implemented in Chapter 4. The extensive simulation results and the experimental

data are included in the following chapters. Numerical techniques which resolve the difficulty in solving nonlinear differential equations involving mixed rigid body variables with large overall motion and elastic variables with small vibration are investigated. The details are presented in Chapter 5 in which Newmark predictor-corrector integration schemes are developed.

CHAPTER 4. MODELING OF RIGID BODIES

The application of the systematic procedure in the derivation of the equations of motion developed in Chapter 3 of this work is demonstrated and implemented in detail in this chapter. The equations of motion for each subsystem are derived individually and are assembled under the concept of compatibility between the local kinematic properties of the elastic degrees of freedom of the connected adjacent elastic members. A specific structure system under consideration is characterized as an open loop system with spherical unconstrained chains capable of rotating about a Hooke's type universal joint. The rigid body motion with three unknown rotations and the elastic degrees of freedom are mutually coupled and influence each other. A traditional motion superposition approach is no longer applicable. Numerical examples for several cases are presented in the following chapters. The simulation results are compared with experimental data and good agreement is indicated.

4.1 Introduction

Chapter 3 of this work presents the development of the equations of motion for an arbitrary elastic beam in a flexible structural system containing both rigid and elastic bodies. In this chapter the theories developed in Chapter 3 are applied to a specific problem. The structural system is characterized as an open loop system with

spherical unconstrained chains capable of rotational motion. The equations of motion for the rigid bodies in the system are derived in a fashion similar to the derivation of the equations of motion for the elastic beams. The influence of the elastic deformation on the rigid body is considered. The strategy in the derivation is first to obtain the local level dynamic equations for each subsystem and then to assemble the equations at the global level to obtain the system equations of motion of the structure. The geometric boundary conditions are implemented to ensure compatibility between the local displacement, velocity and acceleration of the elastic degrees of freedom that are common to two or more members.

Figure 1.3 in Chapter 1 shows a satellite simulator, a test rig, built by Cowles and Anderson at Iowa State University. A corresponding schematic drawing of the dynamic part of the test rig is shown in Figure 4.1. A lower shaft, constrained to rotate about its own spin axis only, supports an upper rotating structure and is driven by a D.C. motor through a drive train. An upper shaft is connected to the lower shaft by a Hooke's type universal joint which allows the upper shaft to rotate in three dimensions. A cross bar is fixed on the top of the upper shaft to balance the coning motion. An upper assembly is defined as those parts of the structure that are supported by the universal joint, except for the upper shaft and the cross bar. The configuration of the entire rotating upper assembly is axisymmetric about the spin axis of the upper shaft.

4.2 Coordinate Systems

Figure 4.2 shows the coordinate systems associated with the structural model of the test rig to assist analyzing the dynamic response of the structural system.

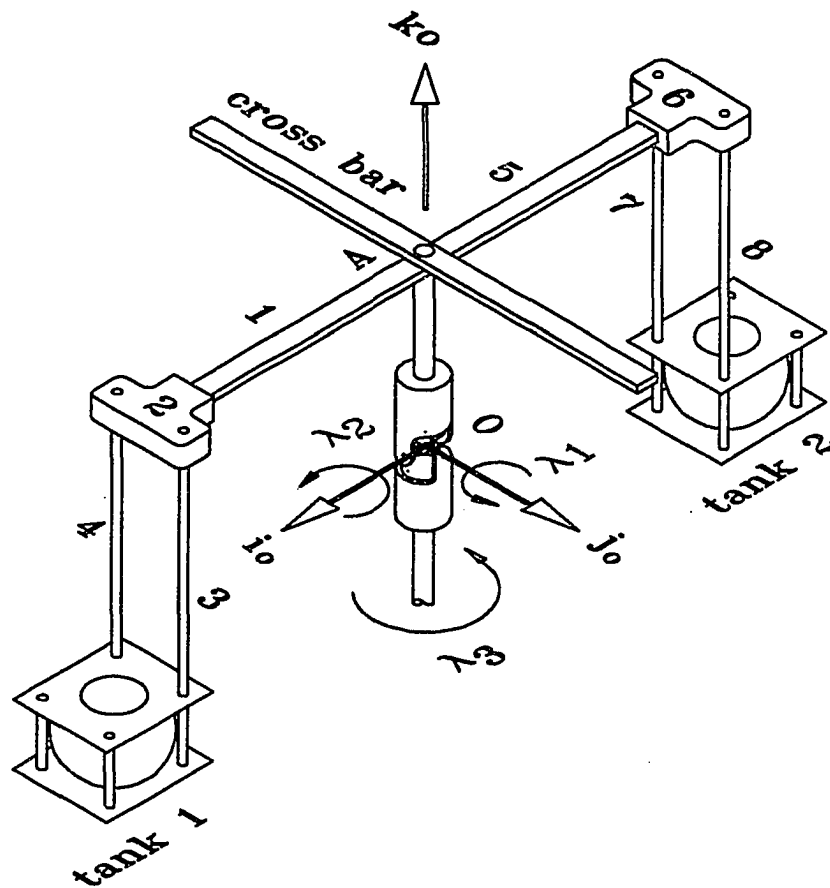


Figure 4.1: A schematic drawing of the test rig (dynamic part only)

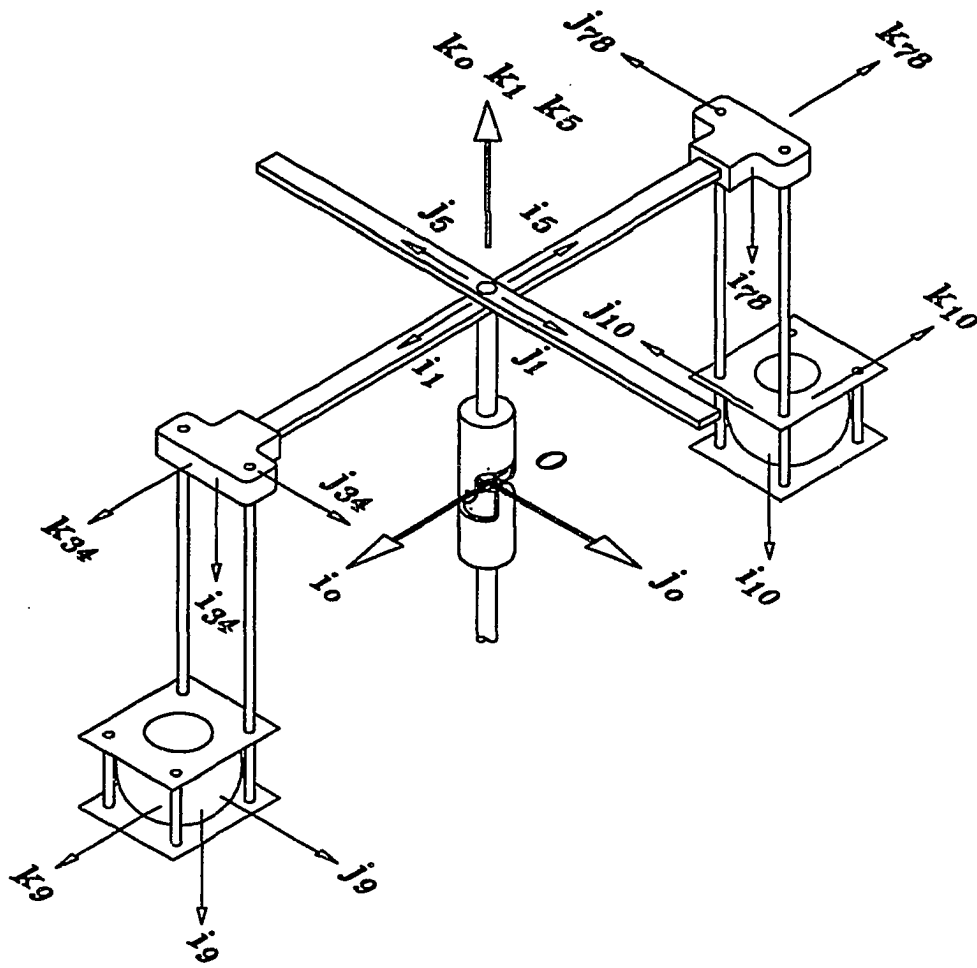


Figure 4.2: Coordinate systems of the test rig

Point O represents the center of the universal joint at which the origin of a set of inertial coordinates, $(\hat{\mathbf{e}}_1, \hat{\mathbf{e}}_2, \hat{\mathbf{e}}_3)$, is located. In addition, a set of moving coordinates, $(\hat{\mathbf{i}}_o, \hat{\mathbf{j}}_o, \hat{\mathbf{k}}_o)$, is also located at point O . These moving coordinates are initially aligned with the inertial coordinates before spin but are attached to the upper assembly and thus rotate with the assembly. Three successive rotating angles are defined between these two sets of coordinates, the inertial coordinates and the moving coordinates. One set of coordinates is defined for each beam, elastic or rigid, and each tank, with the corresponding origins located at each proximal end of the beams and at each geometric center of the tanks, respectively. All of the coordinates are locally defined and are with respect to the rigid body system with no elastic deformation. In particular, the $\hat{\mathbf{i}}$ coordinates for both the rigid and elastic beams are defined such that they coincide with the center lines of the undeformed beams. For the tanks, however, $\hat{\mathbf{i}}$ coordinates are pointed to the opposite direction of one of the moving coordinates, $\hat{\mathbf{k}}_o$, in the initial state with no spin.

4.2.1 Three successive rotating angles

A set of three successive rotating angles about the universal joint is defined between the inertial and moving coordinates as shown in Figure 4.3. First, the upper shaft spins about the $\hat{\mathbf{e}}_3$ axis of the inertial coordinates with an angle of λ_3 to reach a first intermediate system, $(\hat{\mathbf{i}}''_o, \hat{\mathbf{j}}''_o, \hat{\mathbf{k}}''_o)$. Second, the upper assembly nutates about the $\hat{\mathbf{j}}''_o$ axis of the first intermediate system with an angle of λ_1 to reach a second intermediate system, $(\hat{\mathbf{i}}'_o, \hat{\mathbf{j}}'_o, \hat{\mathbf{k}}'_o)$. Finally, the upper assembly rotates through an angle of λ_2 about the $\hat{\mathbf{i}}'_o$ axis of the second intermediate system to reach the final moving coordinates, $(\hat{\mathbf{i}}_o, \hat{\mathbf{j}}_o, \hat{\mathbf{k}}_o)$. These three successive rotating angles constitute the

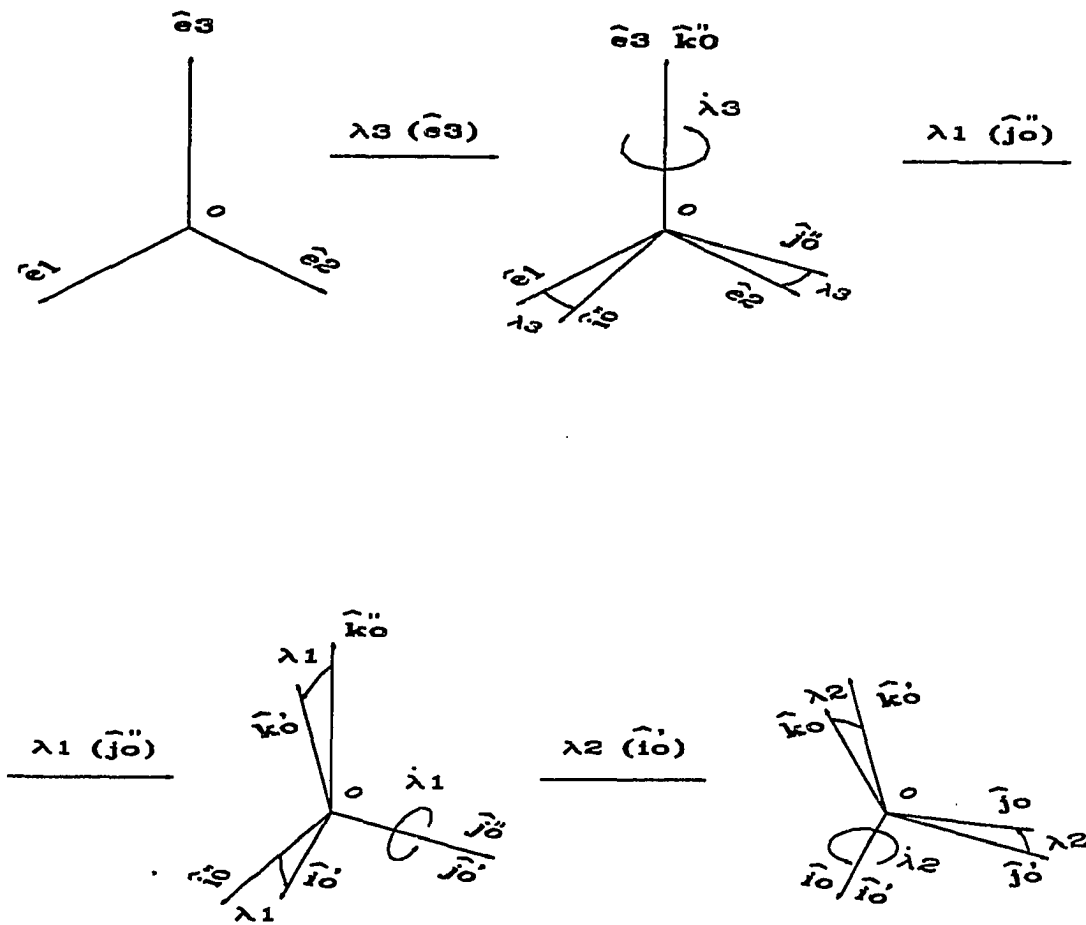


Figure 4.3: Three successive rotating angles between the inertial and moving coordinates

base of the gross rigid body motion which is then coupled with the elastic deflections and rotations in the analysis of dynamic motion of the test rig.

4.2.2 Rigid body angular velocity

Referring to Figure 4.3, the rigid body angular velocity, $\vec{\Omega}$, defined as the angular velocity of the moving frame, can be found by means of the superposition principle of angular velocity, as

$$\vec{\Omega} = \dot{\lambda}_3 \hat{\mathbf{e}}_3 + \dot{\lambda}_1 \hat{\mathbf{j}}''_o + \dot{\lambda}_2 \hat{\mathbf{i}}'_o \quad (4.1)$$

where $\dot{\lambda}_1$, $\dot{\lambda}_2$, and $\dot{\lambda}_3$ are the corresponding time rates of three successive rotating angles λ_1 , λ_2 , and λ_3 , respectively. By observing the rotations, it is found that the following unit vectors are identical, that is

$$\begin{aligned} \hat{\mathbf{e}}_3 &= \hat{\mathbf{k}}''_o \\ \hat{\mathbf{j}}''_o &= \hat{\mathbf{j}}'_o \\ \hat{\mathbf{i}}'_o &= \hat{\mathbf{i}}_o \end{aligned}$$

Substitution of three identities shown above into Eq. 4.1 gives

$$\vec{\Omega} = \dot{\lambda}_3 \hat{\mathbf{k}}''_o + \dot{\lambda}_1 \hat{\mathbf{j}}'_o + \dot{\lambda}_2 \hat{\mathbf{i}}_o \quad (4.2)$$

The objective is to transfer $\hat{\mathbf{k}}''_o$ and $\hat{\mathbf{j}}'_o$ coordinates into the $(\hat{\mathbf{i}}_o, \hat{\mathbf{j}}_o, \hat{\mathbf{k}}_o)$ system. From Appendix A, this can easily be resolved, and the final expression can be written as

$$\vec{\Omega} = \hat{\mathbf{e}}_o^T \mathbf{\Omega} \quad (4.3)$$

where $\hat{\mathbf{e}}_o^T = \{\hat{\mathbf{i}}_o, \hat{\mathbf{j}}_o, \hat{\mathbf{k}}_o\}$, a set of unit vectors of the moving coordinates, and $\mathbf{\Omega}$ is an angular velocity vector which can be written as

$$\mathbf{\Omega} = \mathbf{N} \dot{\mathbf{\Lambda}} \quad (4.4)$$

with $\dot{\mathbf{A}} = \{\dot{\lambda}_1 \dot{\lambda}_2 \dot{\lambda}_3\}^T$. The matrix, \mathbf{N} , in the above equation is a coefficient matrix containing the information of three successive rotations and is defined as

$$\mathbf{N} = \begin{bmatrix} \mathbf{N}_1^T \\ \mathbf{N}_2^T \\ \mathbf{N}_3^T \end{bmatrix} = \begin{bmatrix} 0 & 1 & -\sin \lambda_1 \\ \cos \lambda_2 & 0 & \cos \lambda_1 \sin \lambda_2 \\ -\sin \lambda_2 & 0 & \cos \lambda_1 \cos \lambda_2 \end{bmatrix} \quad (4.5)$$

where \mathbf{N}_1 , \mathbf{N}_2 , and \mathbf{N}_3 are three component vectors in \mathbf{N} . In the previous chapters, it was shown that the configurations of the structure are axisymmetric about the spin axis of the upper shaft. This does not imply, however, that the inertial forces which induce elastic deformation are symmetric about the same spin axis. From an analysis of quasi-static forces acting on the structures, it was found that the tangential inertial forces are non-symmetric about the spin axis of the upper shaft in three orthogonal Cartesian planes. Each elastic beam, therefore, must be discretized using unique elastic generalized coordinates.

4.3 Tank Dynamics

A tank assembly, shown in Figure 4.1, is considered a rigid body system. It is constructed of a spherical plastic container, liquid within the container, two clamping steel plates which hold the spherical tank and clamping bolts. One tank is placed on each side of the test rig axisymmetrically. Liquid sloshing motion within the tank is modeled using computational fluid dynamics (CFD) techniques [28]. The interaction mechanisms between the structure and liquid are investigated in a joint effort considering the flexible structure model and the CFD model. The results are to be published in a separate paper [53]. Due to the special construction of the tank assembly, some elastic degrees of freedom of the supporting flexible beams connecting the

tank assembly are constrained. The geometric boundary conditions are established explicitly prior to defining tank generalized coordinates. *Tank equations of motion are derived in such a way that one model accommodates two tanks in terms of proper substitutions of the corresponding transformation matrices for the appropriate coordinates.* A position vector of the tank is formulated followed by the derivation of a velocity vector at the mass center of the tank assembly by differentiation of the position vector with respect to time. The vector expressions are all relative to the moving coordinates as stated earlier. The tank translational kinetic energy is found using a standard formula which involves a velocity squared term. The tank angular velocity and the inertia dyadic about the mass center are formulated prior to calculating tank rotational kinetic energy. Gravity is the only external loading considered. An instantaneous free surface liquid shape and its orientation within the tank are supplied by the output of the CFD model. The liquid inertia dyadic is updated so as to update the tank kinetic energy. The mass center of the sloshing liquid is calculated and located relative to the geometric center of the tank. The coefficient mass, damping, and stiffness matrices and the generalized force vectors are formulated by applying Lagrange's equation. Derivation of the tank dynamic equations of motion are thereby determined.

4.3.1 Geometric constraints

Figure 4.4 depicts the tank assembly and its associated structures. Beams a and b are two flexible beams which connect the rigid tank assembly at the clamped points, D_a and D_b . The transverse deflections of these two beams in the $\hat{\mathbf{j}}_{ab}$ direction are equal to each other. The rotation of beam a about the $\hat{\mathbf{j}}_{ab}$ axis at point D_a is

equal to the corresponding rotation of beam b about the same axis at point D_b . The rotations of the beams about the $\hat{\mathbf{k}}_{ab}$ axis at points D_a and D_b are zero because these points are clamped on the tank. Therefore, the following four geometric boundary conditions at points D_a and D_b apply,

$$\begin{aligned} d_{amy} &= d_{bmy} \\ \phi_{amy} &= \phi_{bmy} \\ \phi_{amz} &= 0 \\ \phi_{bmz} &= 0 \end{aligned} \quad (4.6)$$

where the second subscript, m , denotes the last finite element node (for both beams) which coincides with either the point D_a or the point D_b . Terms d_{amy} and d_{bmy} are the deflections of beams a and b in the $\hat{\mathbf{j}}_{ab}$ direction, ϕ_{amy} and ϕ_{bmy} are the rotations about the $\hat{\mathbf{j}}_{ab}$ axis, and ϕ_{amz} and ϕ_{bmz} are the rotations about the $\hat{\mathbf{k}}_{ab}$ axis.

The local generalized coordinates of the tank assembly include the rigid body degrees of freedom, which result from three rotations about the universal joint, and the elastic degrees of freedom, which are due to the elastic deformation at the distal end, B_i , of beam i and the elastic deformation of beams a and b at the points D_a and D_b . By applying the four geometric boundary conditions stated in Eq. 4.6, a set of local generalized coordinates of the tank assembly is defined as

$$\mathbf{q}_j = \{\mathbf{\Lambda}^T \mathbf{d}_j^T\}^T \quad (4.7)$$

where

$$\mathbf{d}_j^T = \{d_{imy} \phi_{imz} d_{imz} \phi_{imy} d_{amy} d_{amz} \phi_{amy} d_{bmz}\} \quad (4.8)$$

Thus,

$$\mathbf{\Lambda} = \mathbf{\Theta}_{j\lambda} \mathbf{q}_j$$

$$\mathbf{d}_j = \mathbf{\Theta}_{jd} \mathbf{q}_j \quad (4.9)$$

where the compatibility matrices $\mathbf{\Theta}_{j\lambda}$ and $\mathbf{\Theta}_{jd}$ can easily be found by their definitions.

4.3.2 Tank position and velocity vectors

As shown in Figure 4.4, point T_j is the mass center of the corresponding tank assembly, and the vector \vec{r}_{Tj} is the position vector of the mass center. Thus,

$$\begin{aligned} \vec{r}_{Tj} = & L_{us} \hat{\mathbf{k}}_o + L_i \hat{\mathbf{i}}_i + d_{imy} \hat{\mathbf{j}}_i + d_{imz} \hat{\mathbf{k}}_i + \\ & L_a \hat{\mathbf{i}}_{ab} + d_{amy} \hat{\mathbf{j}}_{ab} + \frac{1}{2} (d_{amz} + d_{bmz}) \hat{\mathbf{k}}_{ab} + \\ & r_{t1} \hat{\mathbf{i}}_j + r_{t3} \hat{\mathbf{k}}_j + \Delta_i \hat{\mathbf{i}}_j + \Delta_j \hat{\mathbf{j}}_j + \Delta_k \hat{\mathbf{k}}_j \end{aligned} \quad (4.10)$$

where the Δ 's are three relative coordinates of the instantaneous mass center displaced from the initial position during the tank motion. It should be noted that the first four terms in the above equation determine the position vector of point B_i after elastic deflections of beam i , the next four terms relate a relative position vector of point E_j to point B_i , and the last five terms establish a relative position vector of the mass center T_j to point E_j on the tank. After transferring the local coordinates to the moving coordinates, it can be shown that Eq. 4.10 in matrix form becomes

$$\begin{aligned} \mathbf{r}_{Tj} = & \begin{Bmatrix} 0 \\ 0 \\ L_{us} \end{Bmatrix} + \mathbf{T}_{oi} \begin{Bmatrix} L_i \\ d_{imy} \\ d_{imz} \end{Bmatrix} + \\ & \mathbf{T}_{oab} \begin{Bmatrix} L_a \\ d_{amy} \\ \frac{1}{2} (d_{amz} + d_{bmz}) \end{Bmatrix} + \end{aligned}$$

$$\mathbf{T}_{oab}\mathbf{T}_{abj}\begin{Bmatrix} r_{t1} + \Delta_i \\ \Delta_j \\ r_{t3} + \Delta_k \end{Bmatrix} \quad (4.11)$$

where, from Figure 4.2, the matrices $\mathbf{T}_{oi}(i = 1 \text{ or } 5)$, $\mathbf{T}_{oab}(ab = 34 \text{ or } 78)$, and $\mathbf{T}_{abj}(abj = 342 \text{ or } 786)$ are the rotational transformation matrices (see Appendix B) wherein \mathbf{T}_{oi} is a constant matrix and \mathbf{T}_{abj} is a time-varying matrix. Physically, beam i is much stiffer than beams a and b , and hence the \mathbf{T}_{oab} matrix is considered as a constant matrix approximately. Differentiation of the above equation with respect to time yields a velocity vector as

$$\begin{aligned} \dot{\mathbf{r}}_{Tj} = & \mathbf{T}_{oi}\begin{Bmatrix} 0 \\ \dot{d}_{imy} \\ \dot{d}_{imz} \end{Bmatrix} + \mathbf{T}_{oab}\begin{Bmatrix} 0 \\ \dot{d}_{amy} \\ \frac{1}{2}(\dot{d}_{amz} + \dot{d}_{bmz}) \end{Bmatrix} + \\ & \mathbf{T}_{oab}\dot{\mathbf{T}}_{abj}\begin{Bmatrix} r_{t1} + \Delta_i \\ \Delta_j \\ r_{t3} + \Delta_k \end{Bmatrix} + \mathbf{T}_{oab}\mathbf{T}_{abj}\begin{Bmatrix} \dot{\Delta}_i \\ \dot{\Delta}_j \\ \dot{\Delta}_k \end{Bmatrix} + \tilde{\boldsymbol{\Omega}} \mathbf{r}_{Tj} \end{aligned} \quad (4.12)$$

where the $\dot{\Delta}'$ s are the relative velocity components of the tank mass center, which are small compared with the overall tank motion. The terms associated with these components can then be neglected in the above velocity equation. $\tilde{\boldsymbol{\Omega}}$ is a skew-symmetric matrix derived from the corresponding rigid body angular velocity, $\tilde{\boldsymbol{\Omega}}$.

4.3.3 Tank angular velocity

The tank angular velocity, $\tilde{\boldsymbol{\Omega}}_j$, is a vector summation of the following three angular velocities,

1. $\vec{\Omega}$, a rigid body angular velocity ($\hat{\mathbf{e}}_o$ relative to $\hat{\mathbf{e}}_e$)
2. $\vec{\omega}_{ab}$, a rigid beam angular velocity ($\hat{\mathbf{e}}_{ab}$ relative to $\hat{\mathbf{e}}_o$)
3. $\vec{\omega}_j$, a tank local angular velocity ($\hat{\mathbf{e}}_j$ relative to $\hat{\mathbf{e}}_{ab}$)

that is,

$$\vec{\Omega}_j = \vec{\Omega} + \vec{\omega}_{ab} + \vec{\omega}_j \quad (4.13)$$

From Figure 4.4 it can be derived that the tank angular velocity, $\vec{\Omega}_j$, takes the following matrix form,

$$\vec{\Omega}_j = \hat{\mathbf{e}}_o^T \left(\boldsymbol{\Omega} + \mathbf{T}_{oi} \begin{Bmatrix} \dot{\phi}_{imy} \dot{\phi}_{imz} \\ \dot{\phi}_{imy} \\ \dot{\phi}_{imz} \end{Bmatrix} + \mathbf{T}_{oab} \begin{Bmatrix} 0 \\ \dot{\phi}_{amy} \\ 0 \end{Bmatrix} \right) \quad (4.14)$$

The local inertia dyadic of the tank assembly about the mass center of the tank can be written as

$$\vec{I}_j = \hat{\mathbf{e}}_j^T \mathbf{I}_j \hat{\mathbf{e}}_j \quad (4.15)$$

where \mathbf{I}_j is a local inertia matrix about the local tank coordinates. Substitution of the transformation matrices between the moving frame and the local tank frame, $(\hat{\mathbf{i}}_j, \hat{\mathbf{j}}_j, \hat{\mathbf{k}}_j)$ into the above equation yields

$$\vec{I}_j = \hat{\mathbf{e}}_o^T \mathbf{I}_{ij} \hat{\mathbf{e}}_o \quad (4.16)$$

where the tank inertia matrix expressed in the moving frame is

$$\mathbf{I}_{ij} = \mathbf{T}_{oab} \mathbf{T}_{abj} \mathbf{I}_j \mathbf{T}_{abj}^T \mathbf{T}_{oab}^T \quad (4.17)$$

4.3.4 Tank kinetic energy

The tank kinetic energy, translational and rotational, can be formulated as

$$KE_j = \frac{1}{2} m_j \dot{\mathbf{r}}_{Tj}^T \dot{\mathbf{r}}_{Tj} + \frac{1}{2} \vec{\Omega}_j \cdot \vec{I}_j \cdot \vec{\Omega}_j \quad (4.18)$$

By substituting Eqs. 4.9, 4.12-4.16, and the expression for $\mathbf{\Omega}$ into Eq. 4.18, it can be found that the kinetic energy of the tank assembly becomes

$$KE_j = \frac{1}{2} \dot{\mathbf{q}}_j^T \mathbf{m}_j \dot{\mathbf{q}}_j \quad (4.19)$$

where the mass matrix, \mathbf{m}_j , is

$$\begin{aligned} \mathbf{m}_j = & m_j \mathbf{\Theta}_{j\lambda}^T \mathbf{N}^T \left(\mathbf{G}_{j1} + \mathbf{H}_{j1} + \mathbf{H}_{j2} + \frac{1}{m_j} \mathbf{I}_{ij} \right) \mathbf{N} \mathbf{\Theta}_{j\lambda} + \\ & m_j \mathbf{\Theta}_{jd}^T \left(\mathbf{G}_{j2} + \mathbf{H}_{j3} + \frac{1}{m_j} \mathbf{H}_{j4} \right) \mathbf{N} \mathbf{\Theta}_{j\lambda} + \\ & m_j \mathbf{\Theta}_{j\lambda}^T \mathbf{N}^T \left(\mathbf{G}_{j2}^T + \mathbf{H}_{j3}^T + \frac{1}{m_j} \mathbf{H}_{j4}^T \right) \mathbf{\Theta}_{jd} + \\ & \mathbf{\Theta}_{jd}^T \mathbf{G}_{j3} \mathbf{\Theta}_{jd} + \mathbf{m}_{jc} \end{aligned} \quad (4.20)$$

where m_j is the total mass of the tank, \mathbf{m}_{jc} is an instantaneous constant mass matrix, \mathbf{G}_{j1} , \mathbf{G}_{j2} , and \mathbf{G}_{j3} are instantaneous constant coefficient matrices, \mathbf{H}_{j1} and \mathbf{H}_{j2} are time-varying symmetric matrices, and \mathbf{H}_{j3} and \mathbf{H}_{j4} are time-varying rectangular matrices. All these matrices are similar to the corresponding ones derived in Chapter 3. Partial differentiation of Eq. 4.19 with respect to \mathbf{q}_j , and partial differentiation of Eq. 4.19 with respect to $\dot{\mathbf{q}}_j$ followed by differentiation of the result with respect to time will result in the following damping matrix,

$$\mathbf{c}_j = \dot{\mathbf{m}}_j - \frac{1}{2} \frac{\partial(\mathbf{m}_j \dot{\mathbf{q}}_j)}{\partial \mathbf{q}_j^T} \quad (4.21)$$

where the partial differentiation can be carried out following the same development procedure demonstrated in Chapter 3.

4.3.5 Tank potential energy

The tank potential energy includes only the potential energy induced by tank elevation in the gravitational field. By following the same procedure in Chapter 3, it can be found that the potential energy takes a similar form,

$$PE_j = V_j + \mathbf{h}_j^T \mathbf{q}_j \quad (4.22)$$

where the potential function, V_j , and the force coefficient vector \mathbf{h}_j are functions of the local generalized coordinates, \mathbf{q}_j . Partial differentiation of the above potential energy equation with respect to \mathbf{q}_j yields the following generalized force vector and stiffness matrix,

$$\begin{aligned} \mathbf{f}_j &= -\frac{\partial V_j}{\partial \mathbf{q}_j^T} - \mathbf{h}_j \\ \mathbf{k}_j &= \frac{\partial \mathbf{h}_j}{\partial \mathbf{q}_j^T} \end{aligned} \quad (4.23)$$

where the matrix partial derivatives can be obtained in a manner similar to that demonstrated in Chapter 3.

4.3.6 Tank equations of motion

The local level tank equations of motion can now be written in a standard form as

$$\mathbf{m}_j \ddot{\mathbf{q}}_j + \mathbf{c}_j \dot{\mathbf{q}}_j + \mathbf{k}_j \mathbf{q}_j = \mathbf{f}_j \quad (4.24)$$

where all the coefficient matrices and the generalized force vectors were derived in the previous sections. Particularly, the mass matrix, \mathbf{m}_j , the stiffness matrix, \mathbf{k}_j , and the generalized force vector, \mathbf{f}_j , are functions of the generalized coordinates, \mathbf{q}_j , only

while the damping matrix, \mathbf{c}_j , is a function of both the generalized coordinates, \mathbf{q}_j , and velocities, $\dot{\mathbf{q}}_j$.

4.4 Rigid Beam Dynamics

Beams 2 and 6 (see Figure 4.1) are modeled as rigid bodies because they are much more rigid in resisting deflections than the other beams. Following the same concept in the tank dynamics, only one model is developed to accommodate two rigid beams.

4.4.1 Rigid beam velocity vector

As shown in Figure 4.5, point B_i is the mass center of the rigid beam after considering the deflections and rotations of the preceding elastic beam. Point O is the universal joint at which the moving frame is located. A position vector of the mass center of the rigid beam can then be written as

$$\vec{r}_{Bi} = L_{us}\hat{\mathbf{k}}_o + L_i\hat{\mathbf{i}}_i + d_{imy}\hat{\mathbf{j}}_i + d_{imz}\hat{\mathbf{k}}_i \quad (4.25)$$

where L_{us} is the length of the upper shaft, L_i is the length of the preceding elastic beam, and d_{imy} and d_{imz} are the elastic deflections at the distal end of the preceding elastic beam. The vector equation above can be rewritten in matrix form as

$$\mathbf{r}_{Bi} = \begin{Bmatrix} 0 \\ 0 \\ L_{us} \end{Bmatrix} + \mathbf{T}_{oi} \begin{Bmatrix} L_i \\ d_{imy} \\ d_{imz} \end{Bmatrix} \quad (4.26)$$

The local generalized coordinates of the rigid beam are defined as

$$\mathbf{q}_i^T = \left\{ \Lambda^T \mid \mathbf{d}_i^T \right\} \quad (4.27)$$

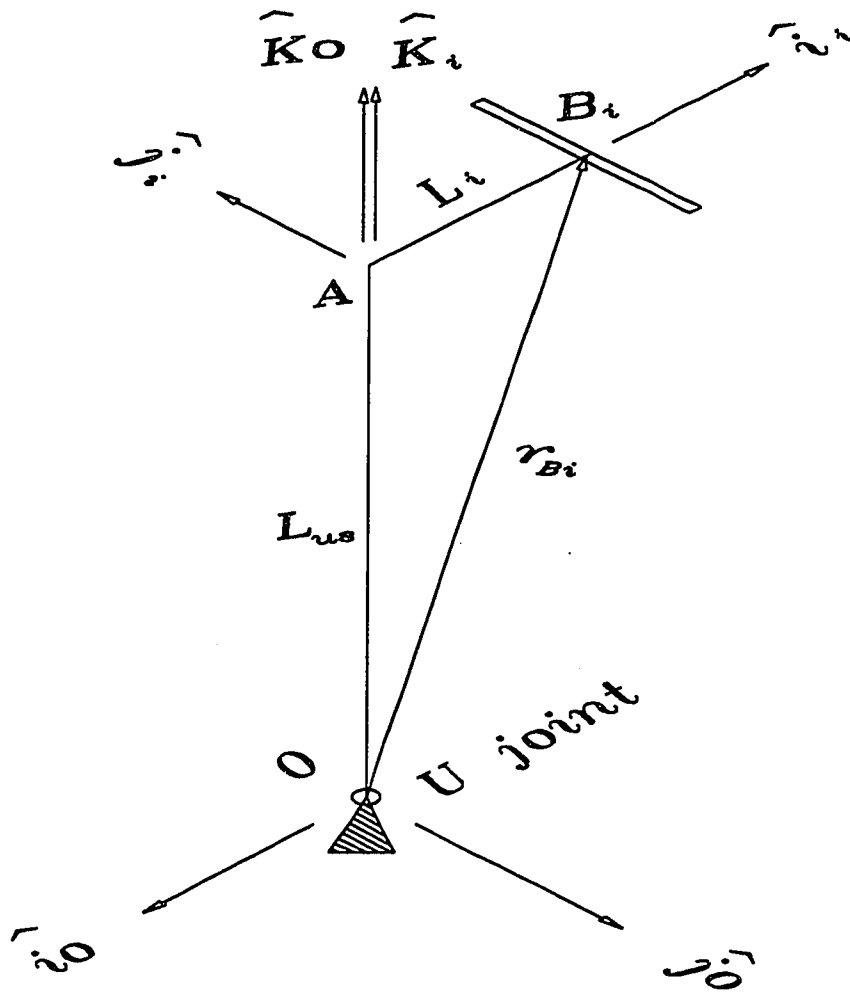


Figure 4.5: Schematic of a rigid beam

where \mathbf{d}_i , the elastic generalized coordinates of the preceding elastic beam, are defined as

$$\mathbf{d}_i^T = \{d_{imy} \ \phi_{imz} \ d_{imz} \ \phi_{imy}\} \quad (4.28)$$

Differentiation of Eq. 4.26 with respect to time yields the following velocity vector,

$$\dot{\mathbf{r}}_{Bi} = \mathbf{T}_{oi} \begin{Bmatrix} 0 \\ \dot{d}_{imy} \\ \dot{d}_{imz} \end{Bmatrix} + \tilde{\boldsymbol{\Omega}} \mathbf{r}_{Bi} \quad (4.29)$$

where $\tilde{\boldsymbol{\Omega}}$ is a skew-symmetric matrix derived from the system angular velocity, $\boldsymbol{\Omega}$, \dot{d}_{imy} is a time derivative of the elastic deflection, d_{imy} , and \dot{d}_{imz} is a time derivative of the elastic deflection, d_{imz} .

4.4.2 Inertia dyadic and angular velocity

The inertia dyadic of the rigid beam about its mass center, B_i , is formulated in the following as

$$\vec{I}_i = \hat{\mathbf{e}}_o^T \mathbf{T}_{oi} \mathbf{I}_i \mathbf{T}_{oi}^T \hat{\mathbf{e}}_o \quad (4.30)$$

where \mathbf{I}_i is a local inertia matrix about the $(\hat{\mathbf{i}}_i, \hat{\mathbf{j}}_i, \hat{\mathbf{k}}_i)$ coordinates. The angular velocity of the rigid beam can be written as

$$\begin{aligned} \vec{\Omega}_i &= \vec{\Omega} + \vec{\omega}_i \\ &= \hat{\mathbf{e}}_o^T \left(\boldsymbol{\Omega} + \mathbf{T}_{oi} \begin{Bmatrix} \phi_{imy} \dot{\phi}_{imz} \\ \dot{\phi}_{imy} \\ \dot{\phi}_{imz} \end{Bmatrix} \right) \end{aligned} \quad (4.31)$$

where $\vec{\omega}_i$ is a local angular velocity accounting for the elastic rotations, ϕ_{imy} and ϕ_{imz} , of the preceding elastic beam.

4.4.3 Rigid beam equations of motion

By following the same procedures stated in the previous sections, the kinetic and potential energies can be formulated in terms of Eqs. 4.27 – 4.31. The mass, damping, and stiffness matrices and the generalized force vectors are derived by differentiating the kinetic energy and potential energy terms with respect to the corresponding quantities in the Lagrange's formula. Finally, the equations of motion of the rigid beam can be expressed in the following form,

$$\mathbf{m}_i \ddot{\mathbf{q}}_i + \mathbf{c}_i \dot{\mathbf{q}}_i + \mathbf{k}_i \mathbf{q}_i = \mathbf{f}_i \quad (4.32)$$

where \mathbf{m}_i , \mathbf{k}_i , and \mathbf{f}_i are functions of \mathbf{q}_i only, and \mathbf{c}_i is a function of \mathbf{q}_i and $\dot{\mathbf{q}}_i$. These matrices and vectors are similar to those derived in the tank dynamics.

4.5 Dynamics of a Bar-Shaft Assembly

The cross bar, the lower shaft, and the upper shaft (see Figure 4.1) constitute a bar-shaft assembly. Following the same procedures in Lagrange's approach, it can be shown that the kinetic and potential energies of the bar-shaft assembly take the following forms,

$$\begin{aligned} KE_o &= \frac{1}{2} \left(I_{ls} \dot{\lambda}_3^2 + \dot{\mathbf{\Lambda}}^T \mathbf{m}_{us} \dot{\mathbf{\Lambda}} + \dot{\mathbf{\Lambda}}^T \mathbf{m}_{cb} \dot{\mathbf{\Lambda}} \right) \\ PE_o &= -m_{ls} G L_{ls} + \left(\frac{1}{2} m_{us} + m_{cb} \right) G L_{us} T_{eo33} \end{aligned} \quad (4.33)$$

where I_{ls} is the moment of inertia of the lower shaft about its spin axis, \mathbf{m}_{us} and \mathbf{m}_{cb} are the mass matrices of the upper shaft and the cross bar, respectively, m_{ls} , m_{us} , and m_{cb} are the corresponding masses of the lower shaft, the upper shaft, and the cross bar, L_{ls} is the length between the universal joint and the mass center of the

lower shaft, and T_{eo33} is an element of the transformation matrix, \mathbf{T}_{eo} , which relates the inertial frame to the moving frame. Substitution of the above equations into Lagrange's formula gives the following dynamic equations for the bar-shaft assembly,

$$(\mathbf{m}_{oc} + \mathbf{m}_{ov}) \ddot{\mathbf{\Lambda}} + \mathbf{c}_o \dot{\mathbf{\Lambda}} = \mathbf{f}_o - \left(\frac{1}{2} m_{us} + m_{cb} \right) GL_{us} \frac{\partial T_{eo33}}{\partial \mathbf{\Lambda}^T} \quad (4.34)$$

where \mathbf{m}_{oc} is a 3×3 null matrix except for the element at the 3rd row and the 3rd column with the value of I_{ls} . \mathbf{f}_o is a zero force vector except for the 3rd component which reflects the unknown input torque about the vertical axis, $\hat{\mathbf{e}}_3$, applied on the lower shaft. The time-varying mass matrix, \mathbf{m}_{ov} , and damping matrix, \mathbf{c}_o , can be further expressed as

$$\begin{aligned} \mathbf{m}_{ov} &= \mathbf{m}_{us} + \mathbf{m}_{cb} \\ \mathbf{c}_o &= \dot{\mathbf{m}}_{ov} - \frac{1}{2} \frac{\partial(\mathbf{m}_o \dot{\mathbf{\Lambda}})}{\partial \mathbf{\Lambda}^T} \end{aligned} \quad (4.35)$$

where $\mathbf{m}_o = \mathbf{m}_{oc} + \mathbf{m}_{ov}$.

4.6 System Equations of Motion

The equations of motion for the elastic beams were derived in Chapter 3. The total number of degrees of freedom of the structure system is nineteen if each elastic beam is modeled by one finite element. Three of the generalized coordinates, $\mathbf{\Lambda}$, result from the rigid body motion and the rest are due to the elastic deformation. If each elastic beam is modeled by two elements, the total number of degrees of freedom will increase to forty-three.

4.6.1 Generalized global and local coordinates

By using one element for each elastic beam, the global generalized coordinates are established as

$$\mathbf{q} = \left\{ \Lambda^T \mid d_{1my}\phi_{1mz}d_{1mz}\phi_{1my} \mid \right. \\ \left. \mid d_{5my}\phi_{5mz}d_{5mz}\phi_{5my} \mid \right. \\ \left. \mid d_{3my}d_{3mz}\phi_{3my}d_{4mz} \mid \right. \\ \left. \mid d_{7my}d_{7mz}\phi_{7my}d_{8mz} \right\}^T \quad (4.36)$$

where the first subscripts of each elastic variable denote the corresponding elastic beam, and the second subscript, m , denotes the last node. Accordingly, the local generalized coordinates are defined as follows:

$$\begin{aligned} \mathbf{q}_0^T &= \{ \Lambda^T \} \\ \mathbf{q}_1^T &= \{ \Lambda^T \mid \mathbf{0} \mid d_{1my}\phi_{1mz}d_{1mz}\phi_{1my} \} \\ \mathbf{q}_2^T &= \{ \Lambda^T \mid d_{1my}\phi_{1mz}d_{1mz}\phi_{1my} \} \\ \mathbf{q}_3^T &= \{ \Lambda^T \mid \mathbf{0} \mid d_{3my}\phi_{3mz}d_{3mz}\phi_{3my} \} \\ \mathbf{q}_4^T &= \{ \Lambda^T \mid \mathbf{0} \mid d_{4my}\phi_{4mz}d_{4mz}\phi_{4my} \} \\ \mathbf{q}_5^T &= \{ \Lambda^T \mid \mathbf{0} \mid d_{5my}\phi_{5mz}d_{5mz}\phi_{5my} \} \\ \mathbf{q}_6^T &= \{ \Lambda^T \mid d_{5my}\phi_{5mz}d_{5mz}\phi_{5my} \} \\ \mathbf{q}_7^T &= \{ \Lambda^T \mid \mathbf{0} \mid d_{7my}\phi_{7mz}d_{7mz}\phi_{7my} \} \\ \mathbf{q}_8^T &= \{ \Lambda^T \mid \mathbf{0} \mid d_{8my}\phi_{8mz}d_{8mz}\phi_{8my} \} \\ \mathbf{q}_9^T &= \{ \Lambda^T \mid d_{1my}\phi_{1mz}d_{1mz}\phi_{1my} \mid d_{3my}d_{3mz}\phi_{3my}d_{4mz} \} \\ \mathbf{q}_{10}^T &= \{ \Lambda^T \mid d_{5my}\phi_{5mz}d_{5mz}\phi_{5my} \mid d_{7my}d_{7mz}\phi_{7my}d_{8mz} \} \end{aligned} \quad (4.37)$$

where $\mathbf{q}_1, \mathbf{q}_3, \mathbf{q}_4, \mathbf{q}_5, \mathbf{q}_7$, and \mathbf{q}_8 are the generalized coordinates for the corresponding elastic beams, \mathbf{q}_0 are for the bar-shaft assembly, \mathbf{q}_2 and \mathbf{q}_6 are for the rigid beams, and \mathbf{q}_9 and \mathbf{q}_{10} are for the tanks. The null vector, $\mathbf{0}$, containing four components, appears in each set of the generalized coordinates for the elastic beams because the proximal ends are all clamped in this particular structure.

4.6.2 Compatibility matrices

By comparison of Eq. 4.37 with Eq. 4.36, the corresponding compatibility matrices for each subsystem are found as

$$\begin{aligned}\Phi_0 &= [\mathbf{I}_\lambda \ \mathbf{0}] \\ \Phi_1 &= \begin{bmatrix} \mathbf{I}_\lambda & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{I}_d & \mathbf{0} \end{bmatrix} \\ \Phi_2 &= \begin{bmatrix} \mathbf{I}_\lambda & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{I}_d & \mathbf{0} \end{bmatrix} \\ \Phi_3 &= \begin{bmatrix} \mathbf{I}_\lambda & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{I}'_d & \mathbf{0} \end{bmatrix} \\ \Phi_4 &= \begin{bmatrix} \mathbf{I}_\lambda & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{I}''_d & \mathbf{0} \end{bmatrix}\end{aligned}$$

$$\begin{aligned}
\Phi_5 &= \begin{bmatrix} \mathbf{I}_\lambda & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{I}_d & \mathbf{0} \end{bmatrix} \\
\Phi_6 &= \begin{bmatrix} \mathbf{I}_\lambda & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{I}_d & \mathbf{0} \end{bmatrix} \\
\Phi_7 &= \begin{bmatrix} \mathbf{I}_\lambda & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{I}'_d \end{bmatrix} \\
\Phi_8 &= \begin{bmatrix} \mathbf{I}_\lambda & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{I}''_d \end{bmatrix} \\
\Phi_9 &= \begin{bmatrix} \mathbf{I}_\lambda & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{I}_d & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{I}_d & \mathbf{0} \end{bmatrix} \\
\Phi_{10} &= \begin{bmatrix} \mathbf{I}_\lambda & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{I}_d & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{I}_d \end{bmatrix} \tag{4.38}
\end{aligned}$$

where $\mathbf{0}$'s are null matrices, \mathbf{I}_λ is a 3×3 identity matrix, \mathbf{I}_d is a 4×4 identity matrix, and \mathbf{I}'_d and \mathbf{I}''_d are defined as

$$\mathbf{I}'_d = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

$$\mathbf{I}_d'' = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix} \quad (4.39)$$

4.6.3 Assembly of equations

The time rates of the compatibility matrices are zero because they are constant. By following the same procedure developed in Chapter 3, it can be concluded that the global mass, damping, and stiffness matrices and generalized force vectors can be written as

$$\begin{aligned} \mathbf{M} &= \sum_{i=0}^{10} \Phi_i^T \mathbf{m}_i \Phi_i \\ \mathbf{C} &= \sum_{i=0}^{10} \Phi_i^T \mathbf{c}_i \Phi_i \\ \mathbf{K} &= \sum_{i=0}^{10} \Phi_i^T \mathbf{k}_i \Phi_i \\ \mathbf{F} &= \sum_{i=0}^{10} \Phi_i^T \mathbf{f}_i \end{aligned} \quad (4.40)$$

These global coefficient matrices can be implemented directly in the computer program and do not have to be expanded mathematically in detail. The system equations of motion are therefore given as

$$\mathbf{M} \ddot{\mathbf{q}} + \mathbf{C} \dot{\mathbf{q}} + \mathbf{K} \mathbf{q} = \mathbf{F} \quad (4.41)$$

The solution of the system dynamic response can be obtained by numerically integrating the system equations above. A detailed numerical integration technique is developed and addressed in Chapter 5. Computer simulation results and experimental measurements are presented in Chapter 6.

CHAPTER 5. DEVELOPMENT OF NUMERICAL ALGORITHMS

The dynamic equations of motion obtained in the modeling of flexible structural systems with unknown gross rigid body motion are often highly nonlinear and possess time-varying coefficient matrices. The inherent characteristics of large overall nonlinear rigid body motion and small linear vibrations are also involved in the system equations. Neither an implicit nor an explicit algorithm seems optimally suited and efficient by itself in dealing with these kinds of equations. This chapter, therefore, presents a sequential implicit-explicit method in which an attempt is made to achieve the benefits of both classes of algorithms. The equation system expressed in matrix form is first mapped to a subsystem in which the specified generalized coordinates are eliminated. The subsystem is then partitioned into two sets of coupled equations. One set of equations, describing the elastic motion, is linear with respect to the elastic generalized coordinates and is integrated implicitly. The other set of equations, governing the rigid body motion, contains the highly nonlinear coupling terms and is integrated explicitly with back substitution of the elastic kinematic properties already calculated in solving the first set of equations. A Newmark algorithm [4] [5] [61] [70] [71] [72] [73] [92] is employed to integrate the second order system of differential equations directly. A predictor-corrector scheme also from the Newmark algorithm is applied to the explicit integration. The procedures developed in the current chapter

are applied to solve system equations derived in the previous chapters in simulating dynamic response of a complicated flexible system with mutually dependent unconstrained rigid body spherical motion and small elastic deformation. Some examples for illustration are presented in validation of the numerical algorithms developed in this chapter.

5.1 Introduction

Traditionally, the dynamic modeling of flexible systems involving elastic bodies focuses on problems in which the gross, or nominal, rigid body motion is predefined or can be derived. The resulting system equations, therefore, only include the elastic generalized coordinates. The mutually coupled terms between the rigid body and elastic motions are usually neglected by assuming them small with negligible effects on a system. However, for those problems with unknown rigid body motion, the corresponding rigid body degrees of freedom must also be included in the system generalized coordinates. These two motions, therefore, influence and are dependent on each other. Consequently, difficulties arise in the numerical analysis. The inherent kinematic facts, reflecting the large overall nonlinear rigid body motion and small linear vibration, need to be accounted for at each time step in the integration. Basically, there are two classes of time integration algorithms for dynamic problems: implicit and explicit. Implicit methods are usually stable numerically, permitting large time steps, and are effective for linear systems. Explicit methods, on the other hand, tend to be effective for nonlinear systems with low natural frequencies. This assures the numerical stability which depends on the highest natural frequency of the system. However, neither class seems very efficient by itself in dealing with systems

with mixed properties arising from nonlinear and linear motions.

For the type of problems under investigation in this work, many methods have been developed in which an attempt is made to *simultaneously* achieve the attributes of both classes of methods in a *single* algorithm. In the time integration of structure-media problems, Belytschko *et al* [14] have presented three techniques for enhancing computational efficiency: explicit-explicit (E-E) partitions, explicit-implicit (E-I) partitions, and implicit-implicit (I-I) partitions. The mesh resulting from the discretization in space by the finite element method is subdivided into two subdomains in which each domain is integrated by a different method. The nodes are partitioned into two groups, explicit and implicit; and the elements are partitioned into three groups, explicit, implicit, and interface, accordingly. In the E-I partitions, the explicit subdomain is integrated first, and the results are subsequently used as boundary conditions for the integration of the implicit subdomain. In the E-E and I-I partitions, either interpolation or extrapolation must be performed, respectively.

Hughes and Liu [45] [46] introduced a simplified method in which the mesh is grouped into explicit and implicit elements only. The notions of interface elements and node categories are avoided. It is claimed that the improved implicit-explicit algorithms are amenable to stability and accuracy analysis, and, at the same time, are simply and concisely implemented. The stability analyses are also carried out for the implicit, explicit, and implicit-explicit algorithms. In their formulation, the Newmark family of methods is used to define the implicit method. A predictor-corrector scheme, constructed from the Newmark family, is employed in defining the explicit method. The developments described in their papers are restricted to linear structural dynamics. In a later paper published by Hughes *et al* [47], the

implicit-explicit finite element concept is extended to nonlinear transient analysis. An effective static problem is formed in the iterative procedures in terms of the unknown displacement, which is in turn linearized. A predictor-multicorrector scheme is proposed to achieve second order accuracy.

In an effort contributed by Chang and Hamilton [20] [21], a method for simulating systems with two inertially coupled motions, a slow motion and a fast motion, is presented. The concept of an implicit-explicit algorithm is applied to integrate the coupled system in a sequential fashion. The fast motion equations are integrated first by the implicit method in which an effective static problem is also formed in terms of displacement. By assuming negligible changes for variables of slow motions for each time step, the time-varying coefficient matrices are replaced by the corresponding ones at the previous time step. The slow motion is updated by integrating the nonlinear equations explicitly, in which a predictor-corrector scheme is employed.

5.2 Current Approach

A sequential implicit-explicit time integration method is proposed in the current chapter. This is designed to simulate systems with mutually coupled large overall nonlinear rigid body motion and small linear elastic motion arising in the dynamic modeling of flexible structural systems. The original differential equation system, which is capable of handling the forward and inverse dynamic analyses, is mapped into a subsystem by eliminating the specified rigid body degrees of freedom in a forward like dynamic analysis. The subsystem is then partitioned into two equation groups, suggested by the inherent characteristics of the flexible dynamic motion. One group is defined to describe the linear elastic motion, and the other group is derived

by including the nonlinear rigid body motion and the coupling terms. The Newmark implicit algorithm is applied to the first set of equations to integrate the elastic motion. Two distinct schemes, direct and iterative integrations, are introduced. The direct integration leads to a direct substitution of the displacement and velocity in the equations in terms of the acceleration, and the values of the coefficients at $(t + \Delta t)$ are replaced by the predicted values based on the current time. The iterative integration, on the other hand, leads to an effective linear problem in terms of the acceleration, which is in turn linearized. A predictor-multicorrector scheme is adopted to achieve second order accuracy without an adverse effect on the stability condition. The explicit algorithm, incorporated with a single pass predictor-corrector scheme, is proposed to integrate the second set of nonlinear rigid body equations of motion. The elastic quantities involved in the coupling terms are back substituted by the values calculated from the first set of elastic equations of motion. The rigid body variables at the future time step are substituted by the predicted values and are corrected using the same Newmark algorithm. The method developed in this chapter possesses improved implementation properties and is intended to be applicable to any dynamic systems with mixed rigid body and elastic degrees of freedom.

5.3 Dynamic Equations

A standard representation of the structural dynamic equations can be written in the following matrix form,

$$\mathbf{M}(\mathbf{p})\ddot{\mathbf{p}} + \mathbf{C}(\mathbf{p}, \dot{\mathbf{p}})\dot{\mathbf{p}} + \mathbf{K}(\mathbf{p})\mathbf{p} = \mathbf{F}(\mathbf{p}) \quad (5.1)$$

where the mass matrix, \mathbf{M} , is usually a symmetric matrix and is a function of the generalized coordinates, \mathbf{p} , which include the rigid body and elastic degrees of freedom. The damping matrix, \mathbf{C} , resulting from the *Coriolis* and *centrifugal* accelerations, is a nonsymmetric matrix and is a function of both the generalized coordinates and their time rates, $\dot{\mathbf{p}}$. The stiffness matrix, \mathbf{K} , is a nonsymmetric matrix and is a function of the generalized coordinates only. The generalized force vector, \mathbf{F} , is also a function of generalized coordinates in general and includes the external loadings which initiate the motion and drive the system. In an inverse dynamic analysis the driving forces are specified and the rigid body motion is to be determined. The above equations of motion need not be modified because the force terms appear on the right hand side of the equations, and the number of generalized coordinates is equal to the number of equations. In a forward-like dynamic analysis, however, the rigid body degrees of freedom are partially or totally specified, and the corresponding driving forces become unknown. The number of unknown generalized coordinates is less than the number of equations though the total number of unknown variables still equals the number of the equations. The equation system cannot be integrated directly and must be restructured for direct integration.

5.3.1 System mapping

By dividing the system generalized coordinates into three groups: unknown rigid body coordinates, known coordinates (rigid and/or elastic), and unknown elastic coordinates, the standard structural dynamic equations, Eq. 5.1, can be rewritten in

the following sub-matrix form,

$$\begin{aligned}
 & \begin{bmatrix} M_{11} & M_{12} & M_{13} \\ M_{21} & M_{22} & M_{23} \\ M_{31} & M_{32} & M_{33} \end{bmatrix} \begin{Bmatrix} \ddot{p}_1 \\ \ddot{p}_2 \\ \ddot{p}_3 \end{Bmatrix} + \\
 & \begin{bmatrix} C_{11} & C_{12} & C_{13} \\ C_{21} & C_{22} & C_{23} \\ C_{31} & C_{32} & C_{33} \end{bmatrix} \begin{Bmatrix} \dot{p}_1 \\ \dot{p}_2 \\ \dot{p}_3 \end{Bmatrix} + \\
 & \begin{bmatrix} K_{11} & K_{12} & K_{13} \\ K_{21} & K_{22} & K_{23} \\ K_{31} & K_{32} & K_{33} \end{bmatrix} \begin{Bmatrix} p_1 \\ p_2 \\ p_3 \end{Bmatrix} = \begin{Bmatrix} F_1 \\ F_2 \\ F_3 \end{Bmatrix} \quad (5.2)
 \end{aligned}$$

where p_1 and p_3 represents the unknown rigid and elastic coordinates respectively. Here p_2 are the generalized sub-coordinates and are supposed to be the specified degrees of freedom of the rigid body motion. The unknown driving forces are included in F_2 . It can easily be shown that Eq. 5.2 can be mapped into the following system which includes two sets of equations,

$$\begin{aligned}
 & \begin{bmatrix} M_{11} & M_{13} \\ M_{31} & M_{33} \end{bmatrix} \begin{Bmatrix} \ddot{p}_1 \\ \ddot{p}_3 \end{Bmatrix} + \\
 & \begin{bmatrix} C_{11} & C_{13} \\ C_{31} & C_{33} \end{bmatrix} \begin{Bmatrix} \dot{p}_1 \\ \dot{p}_3 \end{Bmatrix} + \begin{bmatrix} K_{11} & K_{13} \\ K_{31} & K_{33} \end{bmatrix} \begin{Bmatrix} p_1 \\ p_3 \end{Bmatrix} \\
 & = \begin{Bmatrix} F_1 - M_{12}\ddot{p}_2 - C_{12}\dot{p}_2 - K_{12}p_2 \\ F_3 - M_{32}\ddot{p}_2 - C_{32}\dot{p}_2 - K_{32}p_2 \end{Bmatrix} \quad (5.3)
 \end{aligned}$$

and

$$F_2 = \sum_{i=1}^3 (M_{2i}\ddot{p}_i + C_{2i}\dot{p}_i + K_{2i}p_i) \quad (5.4)$$

Eq. 5.3 can be solved first by the proposed integration algorithms in the following sections. The results are subsequently used for the vectors $\ddot{\mathbf{p}}_i$, $\dot{\mathbf{p}}_i$, and \mathbf{p}_i ($i = 1, 2, 3$) in Eq. 5.4 to determine the unknown driving forces involved in the force vector, \mathbf{F}_2 .

5.3.2 Subsystem partition

In Lagrange's approach, the formula for kinetic energy can be written in a standard matrix form as

$$KE = \frac{1}{2} \dot{\mathbf{p}}^T \mathbf{M} \dot{\mathbf{p}} \quad (5.5)$$

where \mathbf{M} is a symmetric mass matrix and $\dot{\mathbf{p}}$ is a vector resulting from the derivatives of the generalized coordinates with respect to time. The damping matrix can be derived from the kinetic energy formula, Eq. 5.5, and can be expressed in a summation of two sub-matrices as

$$\mathbf{C} = \dot{\mathbf{M}} + \hat{\mathbf{M}} \quad (5.6)$$

where $\dot{\mathbf{M}}$ are the time rates of the mass matrix, \mathbf{M} , and $\hat{\mathbf{M}}$ is a nonsymmetric matrix defined as

$$\hat{\mathbf{M}} = -\frac{1}{2} \frac{\partial}{\partial \mathbf{p}^T} (\mathbf{M} \dot{\mathbf{p}}) \quad (5.7)$$

In general, the damping matrix, \mathbf{C} , is a nonsymmetric matrix while $\dot{\mathbf{M}}$ is a symmetric matrix. A viscous damping matrix can be added to the $\dot{\mathbf{M}}$ matrix. In analogy to Eq. 5.3, the system equations can be partitioned in the following form,

$$\begin{bmatrix} \mathbf{M}_{rr} & \mathbf{M}_{re} \\ \mathbf{M}_{er} & \mathbf{M}_{ee} \end{bmatrix} \begin{Bmatrix} \ddot{\mathbf{q}}_r \\ \ddot{\mathbf{q}}_e \end{Bmatrix} + \begin{bmatrix} \dot{\mathbf{M}}_{rr} + \hat{\mathbf{M}}_{rr} & \dot{\mathbf{M}}_{re} + \hat{\mathbf{M}}_{re} \\ \dot{\mathbf{M}}_{er} + \hat{\mathbf{M}}_{er} & \dot{\mathbf{M}}_{ee} + \hat{\mathbf{M}}_{ee} \end{bmatrix} \begin{Bmatrix} \dot{\mathbf{q}}_r \\ \dot{\mathbf{q}}_e \end{Bmatrix} +$$

$$\begin{bmatrix} \mathbf{0} & \mathbf{K}_{re} \\ \mathbf{0} & \mathbf{K}_{ee} \end{bmatrix} \begin{Bmatrix} \mathbf{q}_r \\ \mathbf{q}_e \end{Bmatrix} = \begin{Bmatrix} \mathbf{Q}_r \\ \mathbf{Q}_e \end{Bmatrix} \quad (5.8)$$

where \mathbf{q}_r are the rigid body generalized coordinates, \mathbf{q}_e are the elastic generalized coordinates, and \mathbf{K}_{ee} is a symmetric structural stiffness matrix. The sub-matrices associated with the rigid body generalized coordinates in the system stiffness matrix are null because there is no stiffness associated with the rigid body motion. The equations above can be separated into two sets of equations as shown below,

$$\mathbf{M}_{rr}\ddot{\mathbf{q}}_r + \mathbf{M}_{re}\ddot{\mathbf{q}}_e = \mathbf{f}_r \quad (5.9)$$

$$\mathbf{M}_{er}\ddot{\mathbf{q}}_r + \mathbf{M}_{ee}\ddot{\mathbf{q}}_e + \dot{\mathbf{M}}_{ee}\dot{\mathbf{q}}_e + \mathbf{K}_{ee}\mathbf{q}_e = \mathbf{f}_e \quad (5.10)$$

where two force sub-vectors, \mathbf{f}_r and \mathbf{f}_e , are defined as

$$\begin{aligned} \mathbf{f}_r &= \mathbf{Q}_r - (\dot{\mathbf{M}}_{rr} + \hat{\mathbf{M}}_{rr})\dot{\mathbf{q}}_r \\ &\quad - (\dot{\mathbf{M}}_{re} + \hat{\mathbf{M}}_{re})\dot{\mathbf{q}}_e - \mathbf{K}_{re}\mathbf{q}_e \\ \mathbf{f}_e &= \mathbf{Q}_e - (\dot{\mathbf{M}}_{er} + \hat{\mathbf{M}}_{er})\dot{\mathbf{q}}_r - \hat{\mathbf{M}}_{ee}\dot{\mathbf{q}}_e \end{aligned} \quad (5.11)$$

By solving Eq. 5.9 for $\ddot{\mathbf{q}}_r$, it is found that

$$\ddot{\mathbf{q}}_r = \mathbf{M}_{rr}^{-1}(\mathbf{f}_r - \mathbf{M}_{re}\ddot{\mathbf{q}}_e) \quad (5.12)$$

Substitution of Eq. 5.12 into Eq. 5.10 for $\ddot{\mathbf{q}}_r$ gives

$$\mathbf{M}_{es}\ddot{\mathbf{q}}_e + \dot{\mathbf{M}}_{ee}\dot{\mathbf{q}}_e + \mathbf{K}_{ee}\mathbf{q}_e = \mathbf{f}_{es} \quad (5.13)$$

where

$$\mathbf{M}_{es} = \mathbf{M}_{ee} - \mathbf{M}_{er}\mathbf{M}_{rr}^{-1}\mathbf{M}_{re} \quad (5.14)$$

and

$$\mathbf{f}_{es} = \mathbf{f}_e - \mathbf{M}_{er}\mathbf{M}_{rr}^{-1}\mathbf{f}_r \quad (5.15)$$

A modified system compared with Eqs. 5.9 and 5.10 therefore takes the following form,

$$\mathbf{M}_{rr}\ddot{\mathbf{q}}_r + \mathbf{M}_{re}\ddot{\mathbf{q}}_e = \mathbf{f}_r \quad (5.16)$$

$$\mathbf{M}_{es}\ddot{\mathbf{q}}_e + \dot{\mathbf{M}}_{ee}\dot{\mathbf{q}}_e + \mathbf{K}_{ee}\mathbf{q}_e = \mathbf{f}_{es} \quad (5.17)$$

where \mathbf{M}_{es} and \mathbf{f}_{es} are defined in Eq. 5.14 and Eq. 5.15. In general, the mass submatrices, \mathbf{M}_{rr} , \mathbf{M}_{re} , and \mathbf{M}_{es} , are nonlinear functions of \mathbf{q}_r and \mathbf{q}_e ; matrix $\dot{\mathbf{M}}_{ee}$ is a nonlinear function of \mathbf{q}_e and $\dot{\mathbf{q}}_e$; the structural stiffness matrix, \mathbf{K}_{ee} , is a constant matrix; the generalized force vectors, \mathbf{f}_r and \mathbf{f}_{es} , include not only the external loading but also the Coriolis and centrifugal effects and are nonlinear functions of \mathbf{q}_r , \mathbf{q}_e , $\dot{\mathbf{q}}_r$, and $\dot{\mathbf{q}}_e$. These two sets of equations above are coupled through the inertia matrix, \mathbf{M}_{es} , and the force terms. Eq. 5.16, which governs the rigid body motion, is nonlinear with respect to \mathbf{q}_r and \mathbf{q}_e while Eq. 5.17, which governs the elastic motion, is linear with respect to \mathbf{q}_e .

5.4 Algorithm Development

In the following sections a sequential implicit-explicit time integration algorithm is developed to solve a system with second order coupled nonlinear ordinary differential equations as expressed in Eqs. 5.16 and 5.17. Eq. 5.17 is numerically integrated first by an implicit method to find the kinematic values of the elastic motion. The results are subsequently used to integrate Eq. 5.16 to update the rigid body motion.

5.4.1 Implicit phase

Two implicit algorithms, direct and iterative, are demonstrated in the following two sections. In the direct method the initial values of the displacement and velocity at the future time step are replaced by the predicted values, and Eq. 5.17 is integrated in terms of the acceleration. The displacement and velocity at the future time step are in turn corrected by a Newmark algorithm once the acceleration at the future time step are found, but the acceleration remains the same as it is predicted. In the iterative method the values of the displacement and velocity are predicted first. Eq. 5.17 is then integrated by forming an effective linear problem in terms of the acceleration. All the variables at the future time step are finally corrected also by the Newmark algorithm. Multiple iterations can be performed to increase the accuracy.

Direct method: The Newmark algorithms [70] [71] [72] [73] in terms of acceleration can be written in the following forms as

$$\mathbf{d}_{t+\Delta t} = \tilde{\mathbf{d}}_{t+\Delta t} + \Delta t^2 \beta \ddot{\mathbf{d}}_{t+\Delta t} \quad (5.18)$$

$$\dot{\mathbf{d}}_{t+\Delta t} = \tilde{\dot{\mathbf{d}}}_{t+\Delta t} + \Delta t \gamma \ddot{\mathbf{d}}_{t+\Delta t} \quad (5.19)$$

and

$$\tilde{\mathbf{d}}_{t+\Delta t} = \mathbf{d}_t + \Delta t \dot{\mathbf{d}}_t + \frac{1}{2} \Delta t^2 (1 - 2\beta) \ddot{\mathbf{d}}_t \quad (5.20)$$

$$\tilde{\dot{\mathbf{d}}}_{t+\Delta t} = \dot{\mathbf{d}}_t + \Delta t (1 - \gamma) \ddot{\mathbf{d}}_t \quad (5.21)$$

where Δt is the size of the time step; the subscripts t and $t + \Delta t$ denote the current and future time, β and γ are two Newmark parameters; \mathbf{d} , $\dot{\mathbf{d}}$, and $\ddot{\mathbf{d}}$ are the displacement, velocity, and acceleration vectors, respectively; and $\tilde{\mathbf{d}}$ and $\tilde{\dot{\mathbf{d}}}$ are the predicted displacement and velocity vectors. The values of the matrices, \mathbf{M}_{es} and $\dot{\mathbf{M}}_{ee}$, and

the vector, \mathbf{f}_{es} , in Eq. 5.17 at the future time $t + \Delta t$ can be evaluated by substitution of the predicted values as shown in Eqs. 5.20 and 5.21. Substitution of Eqs. 5.18 and 5.19 into Eq. 5.17 results in the following equation,

$$\mathbf{M}_{e_I, t+\Delta t} \ddot{\mathbf{q}}_{e, t+\Delta t} = \mathbf{f}_{e_I, t+\Delta t} \quad (5.22)$$

where \mathbf{M}_{e_I} and \mathbf{f}_{e_I} are the effective inertia matrix and the effective force vector, respectively, defined as

$$\begin{aligned} \mathbf{M}_{e_I, t+\Delta t} &= \tilde{\mathbf{M}}_{es, t+\Delta t} + \Delta t \gamma \tilde{\mathbf{M}}_{ee, t+\Delta t} \\ &\quad + \Delta t^2 \beta \mathbf{K}_{ee, t+\Delta t} \end{aligned} \quad (5.23)$$

$$\begin{aligned} \mathbf{f}_{e_I, t+\Delta t} &= \tilde{\mathbf{f}}_{es, t+\Delta t} - \tilde{\mathbf{M}}_{ee, t+\Delta t} \tilde{\ddot{\mathbf{q}}}_{e, t+\Delta t} \\ &\quad - \mathbf{K}_{ee, t+\Delta t} \tilde{\mathbf{q}}_{e, t+\Delta t} \end{aligned} \quad (5.24)$$

Once the acceleration vector is found from Eq. 5.22, the displacement and velocity vectors can be corrected by Eqs. 5.18 and 5.19. This leads to updating the rigid body motion by explicitly integrating Eq. 5.16, and then the procedure advances to the next time step.

Iterative method: The accuracy can be improved using the iterative method with the trade off of performing iterations. A superscript notation, (i) , is used in the following quantities to denote the iteration. The same Newmark algorithms are employed in the development of a predictor-multicorrector scheme.

Before iteration ($i = 0$), the predicted values of the displacement and velocity are assigned as initial values for the future time while the initial acceleration is assigned as zero, *i.e.*

$$i = 0 \quad (5.25)$$

$$\mathbf{q}_{e,t+\Delta t}^{(i)} = \tilde{\mathbf{q}}_{e,t+\Delta t} \quad (5.26)$$

$$\dot{\mathbf{q}}_{e,t+\Delta t}^{(i)} = \tilde{\dot{\mathbf{q}}}_{e,t+\Delta t} \quad (5.27)$$

$$\ddot{\mathbf{q}}_{e,t+\Delta t}^{(i)} = \mathbf{0} \quad (5.28)$$

Substitution of Eqs. 5.18 and 5.19 into Eq. 5.17 yields

$$\begin{aligned} \mathbf{M}^* \tilde{\mathbf{q}}_{e,t+\Delta t}^{(i)} &= \mathbf{f}_{es,t+\Delta t}^{(i)} - \dot{\mathbf{M}}_{ee,t+\Delta t}^{(i)} \tilde{\dot{\mathbf{q}}}_{e,t+\Delta t} \\ &\quad - \mathbf{K}_{ee,t+\Delta t}^{(i)} \tilde{\mathbf{q}}_{e,t+\Delta t} \end{aligned} \quad (5.29)$$

where \mathbf{M}^* is an effective inertia matrix which can be expressed as

$$\begin{aligned} \mathbf{M}^* &= \mathbf{M}_{es,t+\Delta t}^{(i)} + \Delta t \gamma \dot{\mathbf{M}}_{ee,t+\Delta t}^{(i)} \\ &\quad + \Delta t^2 \beta \mathbf{K}_{ee,t+\Delta t}^{(i)} \end{aligned} \quad (5.30)$$

Let $\Delta \ddot{\mathbf{q}}_e$ be an acceleration increment during each iteration, *i.e.*

$$\Delta \ddot{\mathbf{q}}_e = \ddot{\mathbf{q}}_{e,t+\Delta t}^{(i)} - \ddot{\mathbf{q}}_{e,t+\Delta t}^{(i-1)} \quad (5.31)$$

and let $\Delta \mathbf{f}$ be an effective force increment during the same interval of iteration, *i.e.*

$$\begin{aligned} \Delta \mathbf{f} &= \mathbf{f}_{es,t+\Delta t}^{(i)} - \mathbf{M}_{es,t+\Delta t}^{(i)} \ddot{\mathbf{q}}_{e,t+\Delta t}^{(i)} - \\ &\quad \dot{\mathbf{M}}_{ee,t+\Delta t}^{(i)} \dot{\mathbf{q}}_{e,t+\Delta t}^{(i)} - \mathbf{K}_{ee,t+\Delta t}^{(i)} \mathbf{q}_{e,t+\Delta t}^{(i)} \end{aligned} \quad (5.32)$$

By substituting Eqs. 5.30, 5.31, and 5.32 into Eq. 5.29, an effective inertia equation can be derived and approximately expressed as

$$\mathbf{M}^* \Delta \ddot{\mathbf{q}}_e = \Delta \mathbf{f} \quad (5.33)$$

Solution of Eq. 5.33 gives the values of the acceleration increment, $\Delta \ddot{\mathbf{q}}_e$. The results are then subsequently used to find the corrected values of the displacement, velocity,

and acceleration,

$$\ddot{\mathbf{q}}_{\mathbf{e},t+\Delta t}^{(i+1)} = \ddot{\mathbf{q}}_{\mathbf{e},t+\Delta t}^{(i)} + \Delta \ddot{\mathbf{q}}_{\mathbf{e}} \quad (5.34)$$

$$\dot{\mathbf{q}}_{\mathbf{e},t+\Delta t}^{(i+1)} = \tilde{\dot{\mathbf{q}}}_{\mathbf{e},t+\Delta t} + \Delta t \gamma \ddot{\mathbf{q}}_{\mathbf{e},t+\Delta t}^{(i+1)} \quad (5.35)$$

$$\mathbf{q}_{\mathbf{e},t+\Delta t}^{(i+1)} = \tilde{\mathbf{q}}_{\mathbf{e},t+\Delta t} + \Delta t^2 \beta \ddot{\mathbf{q}}_{\mathbf{e},t+\Delta t}^{(i+1)} \quad (5.36)$$

In summary, Eqs. 5.25-5.28 constitute a predictor phase, Eqs. 5.30, 5.32, and 5.33 form an effective linear problem, and Eqs. 5.34-5.36 establish a corrector phase. If additional iterations are to be performed, i is replaced by $i + 1$, and calculations resume with Eq. 5.30. Either a fixed number of iterations may be performed, or iterating may be terminated when $\Delta \ddot{\mathbf{q}}_{\mathbf{e}}$ or $\Delta \mathbf{f}$ satisfy preassigned convergence conditions. When the iterative phase is completed, the solution at the future time, $t + \Delta t$, is defined by the last iterated values. At this point, the current time t is replaced by the future time $t + \Delta t$, and calculations for the next time step may begin.

5.4.2 Explicit phase

After performing the implicit integration of Eq. 5.17, the kinematic values of the elastic displacement, velocity, and acceleration at the future time are obtained, and the results can be substituted into Eq. 5.16. The rigid body displacement and velocity vectors at the future time can be predicted using the following formulas,

$$\tilde{\mathbf{q}}_{\mathbf{r},t+\Delta t} = \mathbf{q}_{\mathbf{r},t} + \Delta t \dot{\mathbf{q}}_{\mathbf{r},t} + \frac{1}{2} \Delta t^2 (1 - 2\beta) \ddot{\mathbf{q}}_{\mathbf{r},t} \quad (5.37)$$

$$\tilde{\dot{\mathbf{q}}}_{\mathbf{r},t+\Delta t} = \dot{\mathbf{q}}_{\mathbf{r},t} + \Delta t (1 - \gamma) \ddot{\mathbf{q}}_{\mathbf{r},t} \quad (5.38)$$

It is noted that Eqs. 5.37 and 5.38 are analogous to Eqs. 5.20 and 5.21. By substitution of the above predictor vectors along with the results from solving Eq. 5.17,

the acceleration vector of the rigid body motion in Eq. 5.16 can be found from the following equations,

$$\mathbf{M}_{\mathbf{r}\mathbf{r},t+\Delta t}\ddot{\mathbf{q}}_{\mathbf{r},t+\Delta t} = \mathbf{f}_{\mathbf{r}\mathbf{E},t+\Delta t} \quad (5.39)$$

where

$$\mathbf{f}_{\mathbf{r}\mathbf{E},t+\Delta t} = \mathbf{f}_{\mathbf{r},t+\Delta t} - \mathbf{M}_{\mathbf{r}\mathbf{e},t+\Delta t}\ddot{\mathbf{q}}_{\mathbf{e},t+\Delta t} \quad (5.40)$$

Once again, the rigid body displacement and velocity vectors are ready to be corrected as follows,

$$\mathbf{q}_{\mathbf{r},t+\Delta t} = \tilde{\mathbf{q}}_{\mathbf{r},t+\Delta t} + \Delta t^2\beta\ddot{\mathbf{q}}_{\mathbf{r},t+\Delta t} \quad (5.41)$$

$$\dot{\mathbf{q}}_{\mathbf{r},t+\Delta t} = \tilde{\dot{\mathbf{q}}}_{\mathbf{r},t+\Delta t} + \Delta t\gamma\ddot{\mathbf{q}}_{\mathbf{r},t+\Delta t} \quad (5.42)$$

where the acceleration vector, $\ddot{\mathbf{q}}_{\mathbf{r},t+\Delta t}$, is found from solving Eq. 5.39. The procedures in the explicit phase include predicting the values through Eqs. 5.37 and 5.38, solving Eq. 5.39 for the acceleration vector, and correcting the values through Eqs. 5.41 and 5.42.

Thus, the sequential implicit-explicit time integration algorithms introduced to solve the equation system, Eqs. 5.16 and 5.17, are completely derived. Solution of the original dynamic equations, Eq. 5.1, are therefore obtained.

5.5 Illustration of Numerical Results

A Fortran computer code has been written to simulate the dynamic response of a spatial structure system with the implementation of the numerical algorithms developed in this work. In the analyses to follow, the direct method, rather than the more accurate iterative method, is employed in the implicit phase because of limitations in computing storage and time. Illustrated in Figure 4.1, the dynamic

part of the structure under consideration in the model is supported by a Hooke's type universal joint at point O . The lower shaft connected to the joint, driven by a D.C. motor, spins vertically about its own central axis. The structure rotates about the joint with two unknown rigid body rotating angles λ_1 and λ_2 . Tanks 1 and 2 are two rigid body assemblies which contain sloshing liquid. Beams 1, 3, 4, 5, 7, and 8 are modeled as elastic bodies while beams 2 and 6 and the cross bar are treated as rigid bodies. More detailed modeling and application details were published by Xu and Baumgarten [93][94][95][96].

A modal analysis for the structure model has been accomplished using the MSC/NASTRAN finite element package. The natural frequencies range from $23Hz$ to over $1000Hz$. The critical size of the time step with γ equal to 0.5 is about 0.0002 seconds, if using the explicit integration method only (see Hughes and Liu [45]). By considering the accuracy in showing the effect of the highest natural frequency in the model, the time step size could be as small as 0.0001 seconds. Based on the sequential implicit-explicit time integration method, a time interval of 0.005 seconds was chosen for integration. The simulations were performed on a networked DECstation 3100 workstation using MIPS Fortran 77 compiler running under RISC-based ULTRIX 4.1. Approximately 5.21 seconds of CPU time was required for one real-time step. The total number of the degrees of freedom of the model is equal to nineteen, in which each elastic beam is modeled by one beam element with a third order polynomial shape function.

A sinusoidal function (see Figure 2.6) was used as a spin profile for the lower shaft in the simulation. Starting from zero, the angular velocity, $\dot{\lambda}_3$, increased gradually and reached $60rpm$ over the time base, t_o . In Case 1, an initial tilt of $\lambda_1 = 1$ degree

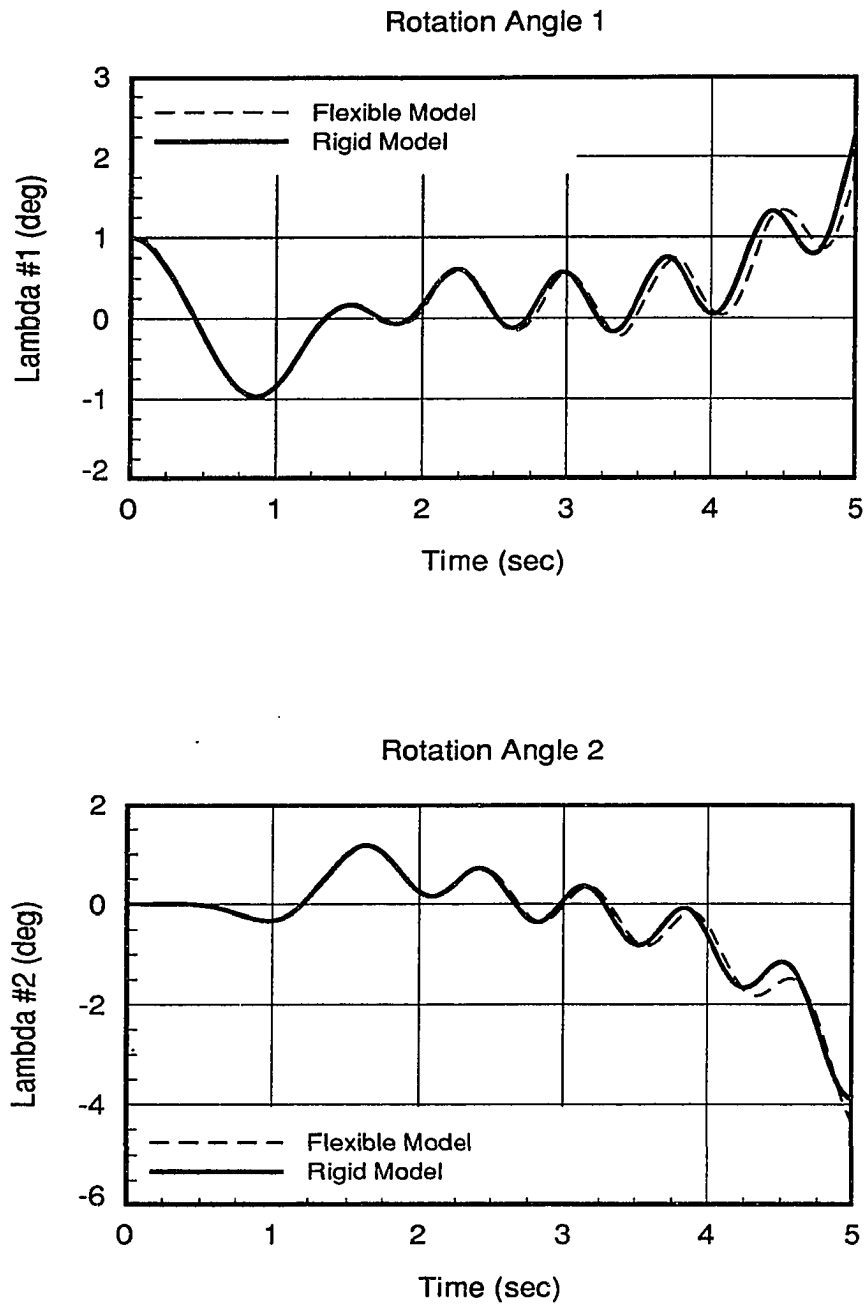


Figure 5.1: Case 1: Response comparison of a rigid body model with a flexible model using initial tilt angle of $\lambda_1 = 1$ degree and base time of $t_0 = 3$ seconds

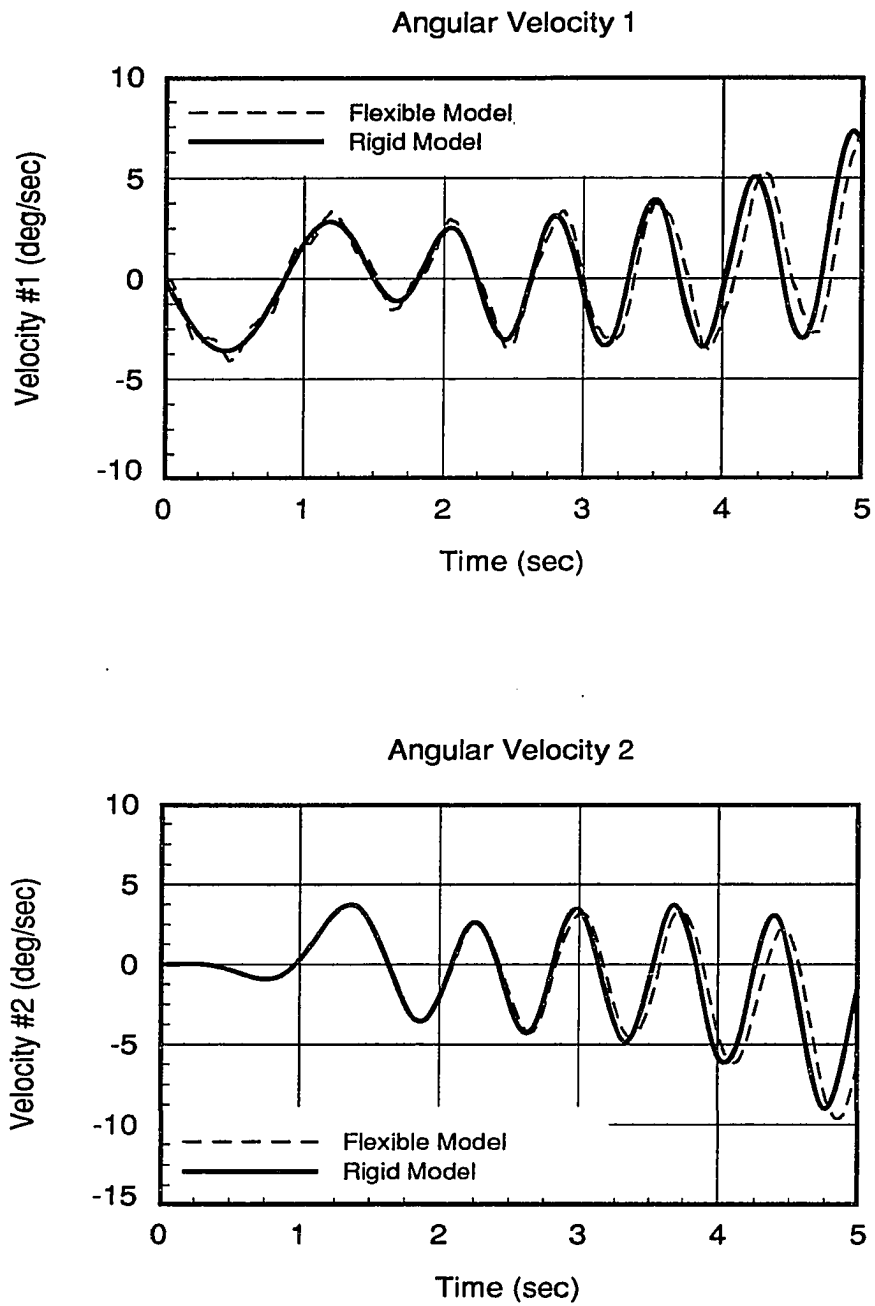


Figure 5.1 (Continued)

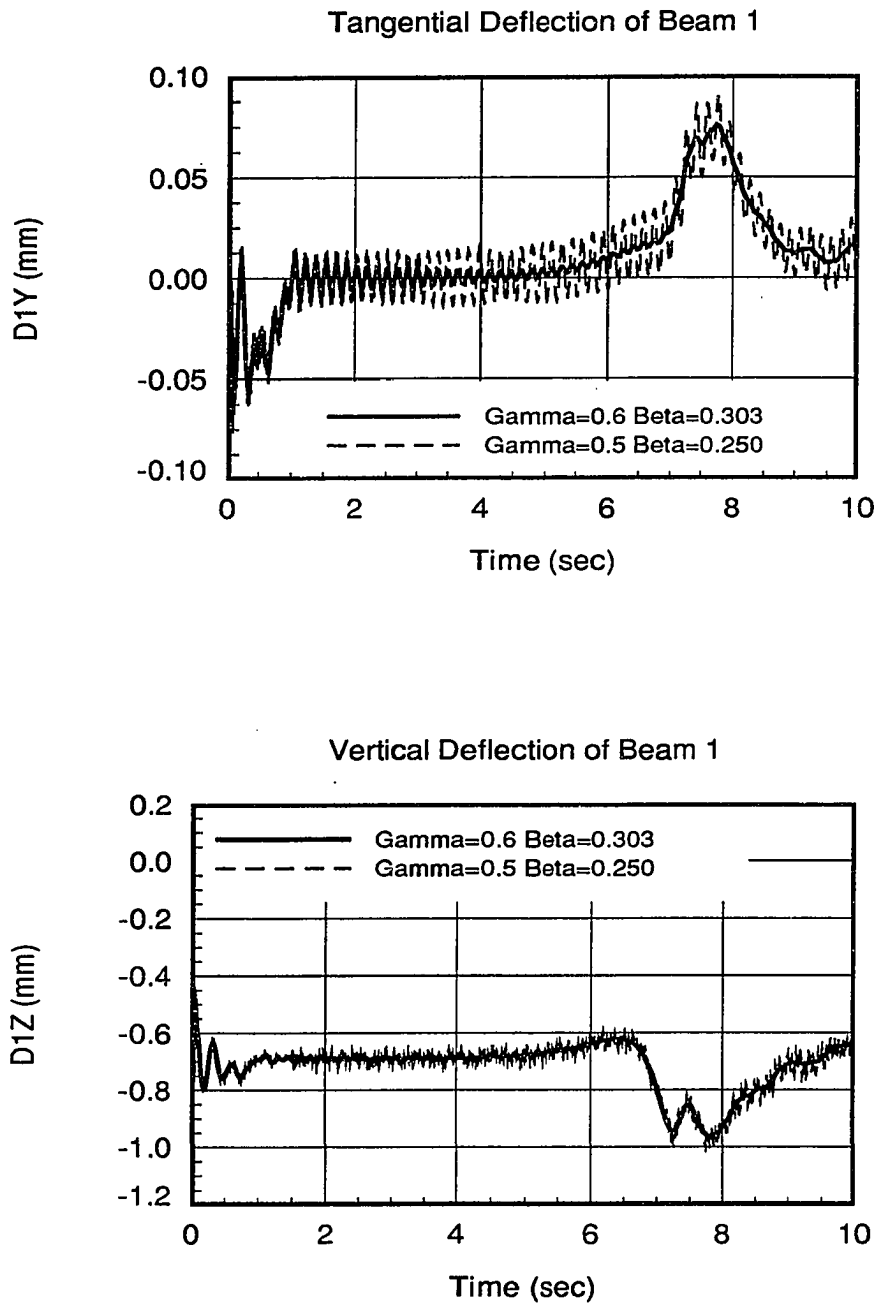
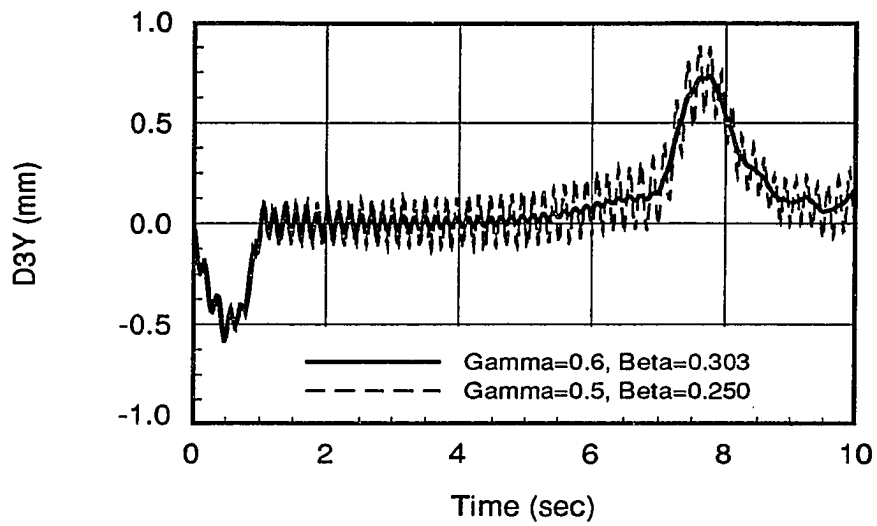


Figure 5.2: Case 2: Confirmation of “numerical damping” effect with $\gamma = 0.6$, $\beta = 0.303$ and $\gamma = 0.5$, $\beta = 0.25$ by applying an impulse of $1N$ as an exciting force

Tangential Deflection of Beam 3



Radial Deflection of Beam 3

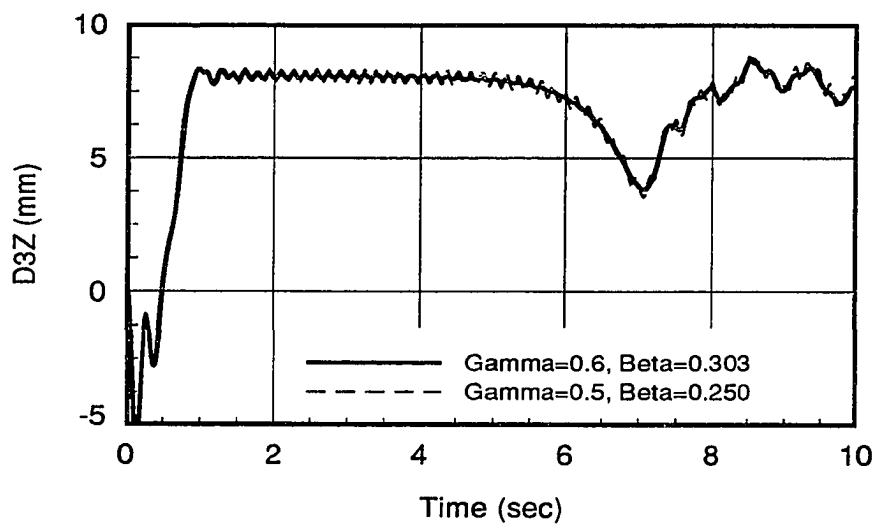


Figure 5.2 (Continued)

was set to induce off balance rotation, and t_o was set to 3 seconds in the spin profile. The numerical results of two rigid body rotating angles and velocities were compared for the rigid and flexible models. As shown in Figure 5.1, the values of the flexible model (dashed line) deviate significantly from the corresponding values of the rigid body model (solid line) after a few seconds. Another run, Case 2, with an impulse acting on one of the tanks but with no initial tilt was performed. Here, $t_o = 1$ second was used in the spin profile. The impulse was applied in the vertical direction after 1.5 seconds with a magnitude of 1 Newton. This run lasted for 10 seconds so that the peak value of off balance motion was developed thoroughly. The solid lines represent the results with Newmark parameters of $\gamma = 0.6$ and $\beta = 0.303$ while the dashed lines are for $\gamma = 0.5$ and $\beta = 0.25$. A phenomenon of “numerical damping” is reconfirmed in the plots as shown in Figure 5.2. By increasing γ to 0.6, the high frequencies engendered by the stiff components are damped out. $D1Y$ and $D3Y$ are the circumferential (tangential) deflections of beam 1 and beam 3 at the distal ends, respectively (note the different scales used in the plots). $D1Z$ is the vertical deflection of beam 1 while $D3Z$ is the radial deflection of beam 3. The initial elastic deformation for each flexible beam in all the runs were set to zero to avoid possible over or under estimation of elastic deflections and rotations in the simulation.

5.6 Summary

A sequential implicit-explicit time integration algorithm has been developed in the present chapter. The method is intended to solve second order nonlinear ordinary differential equations derived from the modeling of flexible structural systems with mutually dependent rigid body and elastic motions. The original dynamic equations

are transferred to a subsystem which is composed of two coupled sets of motion equations. One set of equations governs the nonlinear rigid body motion while another set of equations is defined to describe the linear elastic vibration. Two algorithms, implicit and explicit, are proposed to integrate the subsystem, in which the elastic vibration is solved first during the implicit phase, and the rigid body motion is then updated subsequently during the explicit phase. The Newmark algorithm family is employed in both the implicit and explicit integrations in which a multiple pass predictor-corrector scheme is used in the implicit method while a single pass predictor-corrector scheme is used in the explicit method. Two illustrative examples are presented in simulating dynamic response of a spatial system with unknown rigid body motion. The numerical integrations are carried out, and the results are compared for a rigid body model and a flexible model. In the second run case an impulse is applied to the structure to excite the elastic beam oscillations in which the higher frequencies can be damped out by increasing the value of the Newmark parameter, γ . The computational efficiency is demonstrated using the current method. The accuracy (which is at most second order as discussed here) can be further improved by introducing higher order predictor-corrector schemes.

CHAPTER 6. SIMULATION AND MEASUREMENT RESULTS

The following structure configurations and material properties are used in the simulation of dynamic response of the satellite test rig. The ISO unit system (metric system) is selected as a primary unit system. The corresponding values in the English system are also supplied in parentheses following the ISO values. The specific values for each of the structure members are listed in Tables 6.1 - 6.6.

The input spin velocity of the lower shaft and its corresponding angular acceleration profile are shown in Figure 6.1. A sinusoidal function is assumed for the angular velocity profile in which the speed of the lower shaft increases gradually from zero to $\omega_o = 60 \text{ rpm}$ over a time base, t_o . In cases 1, 2, and 3 the time base of $t_o = 3$ seconds is used while in case 4 the time base of $t_o = 1.5$ seconds is selected. For convenience, the test rig schematic drawing is illustrated in Figure 6.2 again. All the physical structures, dimensions, and locations dealt with in the following simulation cases are referred to in the figure. The following parameters and conditions in each case are either optional or varying:

upper shaft tilt, λ_1	(to set initial tilt angle)
external impulse	(to excite the structures)
steady-state spin velocity, ω_o	(to limit maximum speed)
spin time base, t_o	
total simulation time, τ	
integration time size, Δt	
Newmark parameters, γ, β	

Table 6.1: List of cross-sectional shape and size

Beam 1	rectangle	$6mm \times 12mm(1/4'' \times 1/2'')$
Beam 2	square	$19mm \times 19mm(3/4'' \times 3/4'')$
Beam 3	circle	$d_3 = 6mm(5/16'')$
Beam 4	circle	$d_4 = 6mm(5/16'')$
Beam 5	rectangle	$6mm \times 12mm(1/4'' \times 1/2'')$
Beam 6	square	$19mm \times 19mm(3/4'' \times 3/4'')$
Beam 7	circle	$d_7 = 6mm(5/16'')$
Beam 8	circle	$d_8 = 6mm(5/16'')$
Cross bar	rectangle	$6.35mm \times 25.4mm(1/4'' \times 1'')$
Upper shaft	circle	$d_{us} = 25.4mm(1'')$
Lower shaft	circle	$d_{ls} = 25.4mm(1'')$
Tank	cube	$165mm \times 165mm \times 114mm(6.5'' \times 6.5'' \times 4.5'')$

Table 6.2: List of cross-sectional area

Beam 1	$A_1 = 72 mm^2$	$(0.125 in^2)$
Beam 2	$A_2 = 361 mm^2$	$(0.56 in^2)$
Beam 3	$A_3 = 28.3 mm^2$	$(4.39 \times 10^{-2} in^2)$
Beam 4	$A_4 = 28.3 mm^2$	$(4.39 \times 10^{-2} in^2)$
Beam 5	$A_5 = 72 mm^2$	$(0.125 in^2)$
Beam 6	$A_6 = 361 mm^2$	$(0.56 in^2)$
Beam 7	$A_7 = 28.3 mm^2$	$(4.39 \times 10^{-2} in^2)$
Beam 8	$A_8 = 28.3 mm^2$	$(4.39 \times 10^{-2} in^2)$
Cross bar	$A_{cb} = 161.3 mm^2$	$(0.25 in^2)$
Upper shaft	$A_{us} = 506.7 mm^2$	$(0.785 in^2)$
Lower shaft	$A_{ls} = 506.7 mm^2$	$(0.785 in^2)$

Table 6.3: List of link length

Beam 1	$L_1 = 165 \text{ mm}$	(6.5")
Beam 2	$L_2 = 104 \text{ mm}$	(4")
Beam 3	$L_3 = 290 \text{ mm}$	(11.4")
Beam 4	$L_4 = 290 \text{ mm}$	(11.4")
Beam 5	$L_5 = 165 \text{ mm}$	(6.5")
Beam 6	$L_6 = 104 \text{ mm}$	(4")
Beam 7	$L_7 = 290 \text{ mm}$	(11.4")
Beam 8	$L_8 = 290 \text{ mm}$	(11.4")
Cross bar	$L_{cb} = 1219 \text{ mm}$	(48")
Upper shaft	$L_{us} = 127 \text{ mm}$	(5")
Lower shaft	$L_{ls} = 940 \text{ mm}$	(37")

Table 6.4: List of mass density

Beam 1	$\rho_1 = 7.833 \times 10^3 \text{ kg/m}^3$	(15.18slug/ft ³)
Beam 2	$\rho_2 = 2.707 \times 10^3 \text{ kg/m}^3$	(5.25slug/ft ³)
Beam 3	$\rho_3 = 7.833 \times 10^3 \text{ kg/m}^3$	(15.18slug/ft ³)
Beam 4	$\rho_4 = 7.833 \times 10^3 \text{ kg/m}^3$	(15.18slug/ft ³)
Beam 5	$\rho_5 = 7.833 \times 10^3 \text{ kg/m}^3$	(15.18slug/ft ³)
Beam 6	$\rho_6 = 2.707 \times 10^3 \text{ kg/m}^3$	(5.25slug/ft ³)
Beam 7	$\rho_7 = 7.833 \times 10^3 \text{ kg/m}^3$	(15.18slug/ft ³)
Beam 8	$\rho_8 = 7.833 \times 10^3 \text{ kg/m}^3$	(15.18slug/ft ³)
Cross bar	$\rho_{cb} = 7.833 \times 10^3 \text{ kg/m}^3$	(15.18slug/ft ³)
Upper shaft	$\rho_{us} = 7.833 \times 10^3 \text{ kg/m}^3$	(15.18slug/ft ³)
Lower shaft	$\rho_{ls} = 7.833 \times 10^3 \text{ kg/m}^3$	(15.18slug/ft ³)

Table 6.5: List of Young's modulus

Beam 1	$E_1 = 210 \text{ GPa}$	$(30 \times 10^6 \text{ psi})$
Beam 2	$E_2 = 70 \text{ GPa}$	$(10 \times 10^6 \text{ psi})$
Beam 3	$E_3 = 210 \text{ GPa}$	$(30 \times 10^6 \text{ psi})$
Beam 4	$E_4 = 210 \text{ GPa}$	$(30 \times 10^6 \text{ psi})$
Beam 5	$E_5 = 210 \text{ GPa}$	$(30 \times 10^6 \text{ psi})$
Beam 6	$E_6 = 70 \text{ GPa}$	$(10 \times 10^6 \text{ psi})$
Beam 7	$E_7 = 210 \text{ GPa}$	$(30 \times 10^6 \text{ psi})$
Beam 8	$E_8 = 210 \text{ GPa}$	$(30 \times 10^6 \text{ psi})$
Cross bar	$E_{cb} = 210 \text{ GPa}$	$(30 \times 10^6 \text{ psi})$
Upper shaft	$E_{us} = 210 \text{ GPa}$	$(30 \times 10^6 \text{ psi})$
Lower shaft	$E_{ls} = 210 \text{ GPa}$	$(30 \times 10^6 \text{ psi})$

Gravity is considered the only external loading acting on the test rig. In the first four cases, liquid contained in the tanks is modeled as a fixed mass concentrated at the tank geometric centers. Additionally, internal energy dissipation and instantaneous liquid free surface profile are considered. In the last run case, the liquid sloshing flow is modeled by the computational fluid dynamics techniques. Instantaneous liquid energy loss due to internal friction, liquid mass center relative to the tank geometric center, liquid shape, and liquid orientation are computed using the CFD modeling and are implemented in the structure modeling. In return, tank position, velocity, acceleration, and orientation are computed by the structure computer code and are implemented in the CFD code. Fluid-structure interaction mechanisms are then investigated. No relevant publications addressing the interaction mechanisms linking two completely developed models have been found to date. Initial results for the pure spin-up cases are encouraging and are acceptable in general.

Table 6.6: List of second moment of area

Beam 1	$I_{1y} = 216mm^4$	$(5.19 \times 10^{-4}in^4)$
	$I_{1z} = 864mm^4$	$(2.08 \times 10^{-3}in^4)$
Beam 2	$I_{2y} = 10860mm^4$	$(2.61 \times 10^{-2}in^4)$
	$I_{2z} = 10860mm^4$	$(2.61 \times 10^{-2}in^4)$
Beam 3	$I_{3y} = 63.6mm^4$	$(1.53 \times 10^{-4}in^4)$
	$I_{3z} = 63.6mm^4$	$(1.53 \times 10^{-4}in^4)$
Beam 4	$I_{4y} = 63.6mm^4$	$(1.53 \times 10^{-4}in^4)$
	$I_{4z} = 63.6mm^4$	$(1.53 \times 10^{-4}in^4)$
Beam 5	$I_{5y} = 216mm^4$	$(5.19 \times 10^{-4}in^4)$
	$I_{5z} = 864mm^4$	$(2.08 \times 10^{-3}in^4)$
Beam 6	$I_{6y} = 10860mm^4$	$(2.61 \times 10^{-2}in^4)$
	$I_{6z} = 10860mm^4$	$(2.61 \times 10^{-2}in^4)$
Beam 7	$I_{7y} = 63.6mm^4$	$(1.53 \times 10^{-4}in^4)$
	$I_{7z} = 63.6mm^4$	$(1.53 \times 10^{-4}in^4)$
Beam 8	$I_{8y} = 63.6mm^4$	$(1.53 \times 10^{-4}in^4)$
	$I_{8z} = 63.6mm^4$	$(1.53 \times 10^{-4}in^4)$
Cross bar	$I_{cby} = 542mm^4$	$(1.30 \times 10^{-3}in^4)$
	$I_{cbz} = 8672mm^4$	$(2.08 \times 10^{-2}in^4)$
Upper shaft	$I_{usy} = 20432mm^4$	$(4.90 \times 10^{-2}in^4)$
	$I_{usz} = 20432mm^4$	$(4.90 \times 10^{-2}in^4)$
Lower shaft	$I_{lsy} = 20432mm^4$	$(4.90 \times 10^{-2}in^4)$
	$I_{lsz} = 20432mm^4$	$(4.90 \times 10^{-2}in^4)$

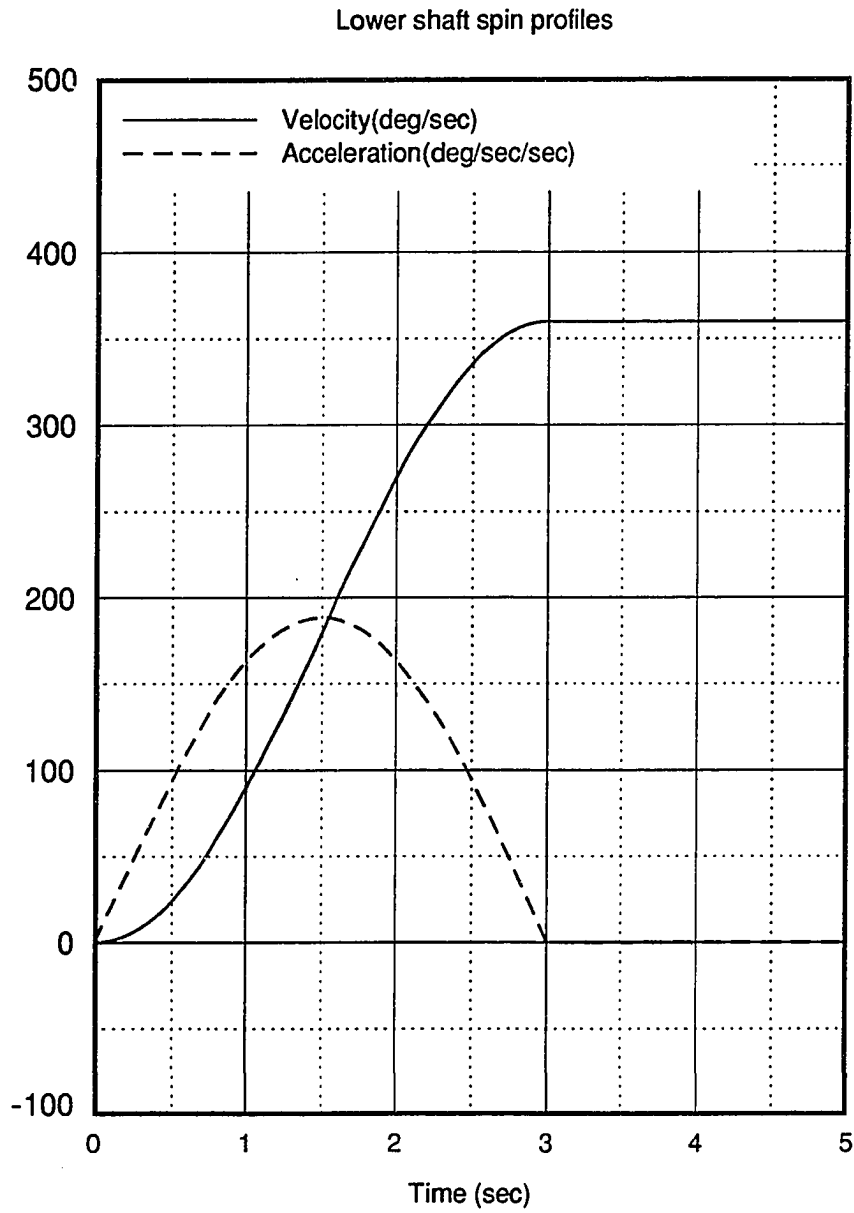


Figure 6.1: Lower shaft spin-up profiles where the solid line denotes spin velocity, $\dot{\lambda}_3 (= \omega)$, and the dashed line denotes spin acceleration, $\ddot{\lambda}_3 (= \dot{\omega})$

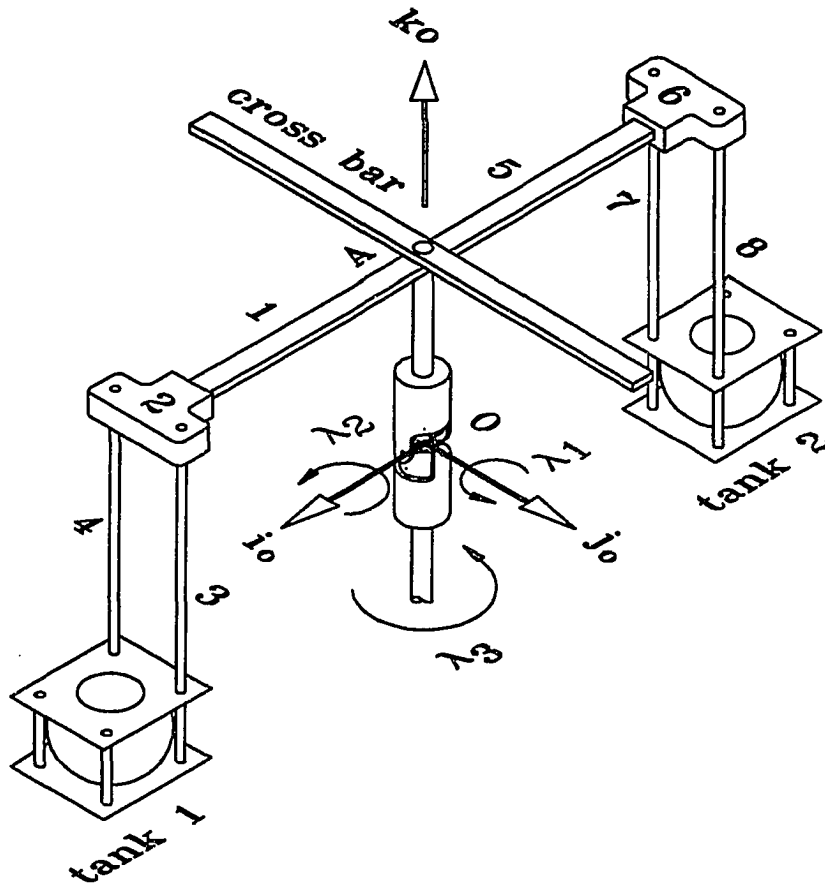


Figure 6.2: A test rig schematic drawing

6.1 Run Case 1: Stable Spin-up with Collar Up

Simulation conditions and parameters for Case 1 are listed in Table 6.7. The collar which covers the universal joint in the test rig is always up in this case so that two rigid body nutation angles are always zero. This is a pure spin-up case for which the simulation results are shown from Figure 6.3 to Figure 6.9.

Table 6.7: Case 1 conditions and parameters

Tilt, λ_1	(degree)	(none)
Impulse	(Newton)	(none)
Velocity, ω_o	(rpm)	60
Time base, t_o	(second)	3
Total time, τ	(second)	5
Step size, Δt	(second)	0.005
Parameter, γ		0.5
Parameter, β		0.25

The time histories of the tank center position, velocity, and acceleration relative to an inertial frame during spin-up are plotted in Figures 6.3, 6.4, 6.5, and 6.6. The effect of elastic deformation is clearly evident in Figure 6.4 showing the X and Z coordinates of the tank center position during the first second. The tangential deflections of beams 1 and 3 at the distal ends are compared in Figure 6.7. The tangential deflection of beam 1 is much smaller than the total tangential deflection of beam 3 which includes not only beam 1 tangential deflection but also its own local tangential deflection. In addition, there is a phase shift of about 45 degrees to 90 degrees between these two deflections as shown in Figure 6.7. The radial deflections of beams 3 and 4 are plotted in Figure 6.8. Initial deflections are set to zero. During the

first one second the gravity dominates the forces acting on the tank and overpowers the centrifugal force term. As a consequence, the radial deflections are negative relative to the local coordinates. As spin velocity increases during the next few seconds, the centrifugal force overcomes the gravity, and the radial deflections become positive and reach the steady-state values as the spin velocity reaches a constant. In addition, there is a twist phenomenon between beam 3 and beam 4 during the gradual spin-up period of the first three seconds as evidenced by the difference of the radial deflections shown in Figure 6.8. The twist is further evidenced during the constant spin period of the last two seconds by the phase shift shown in Figure 6.8. The rotations, due to the vertical or radial deflection, about the corresponding Y axis of beams 1 and 3 are compared in Figure 6.9. In Figure 6.9, the vertical deflection of beam 1 shows a negative value (referred to the left axis) while the corresponding rotation of beam 1 about the Y_1 axis shows a positive value (referred to the right axis). This is exactly what would be expected. The signs of the radial deflection of beam 3 (see Figure 6.8) and the corresponding rotation of beam 3 about the Y_{34} axis (see Figure 6.9) are also opposite.

6.2 Run Case 2: Stable Spin-up with Collar Down

The only difference between Case 1 and Case 2 is that the collar is down in Case 2 while the collar is up in Case 1. Therefore, there is no constraint against rigid body nutation. Any asymmetric elastic deformation of the flexible structures will cause rigid body nutation. Two nutating angles, induced by the elastic deformation, clearly show up in Figure 6.10 though the magnitudes are still small during the first five seconds simulated. The time history of the tank center position relative to an

inertial frame is plotted in Figure 6.11(note the different scales used). All the elastic deflections and rotations are very much the same as those in Case 1 because the magnitudes of two nutating angles are small. Therefore, they would not induce large inertial forces.

Table 6.8: Case 2 conditions and parameters

Tilt, λ_1	(degree)	0
Impulse	(Newton)	(none)
Velocity, ω_o	(rpm)	60
Time base, t_o	(second)	3
Total time, τ	(second)	5
Step size, Δt	(second)	0.005
Parameter, γ		0.5
Parameter, β		0.25

6.3 Run Case 3: General Motion with Initial Tilt

In case 3, one of the initial rigid body tilt angles, λ_1 , is set to 1 degree while the other rigid body tilt angle, λ_2 , is set to zero. No constraint on the universal joint is applied. The main purpose of this run is to verify the effects of rigid body nutation on the structural elastic deformation. Also, the results of a rigid body model are compared with the results of a flexible model. A fairly large value of the Young's modulus is used in the rigid body model so that the structures are fictitiously stiffened. Thus, the elastic deflections and rotations are negligible. The simulation conditions and parameters of Case 3 are listed in Table 6.9:

A comparison of the results of the flexible and rigid models are plotted from Figure 6.12 to Figure 6.16. Rigid body nutating angles are shown in Figure 6.12 in

Table 6.9: Case 3 conditions and parameters

Tilt, λ_1	(degree)	1
Impulse	(Newton)	(none)
Velocity, ω_o	(rpm)	60
Time base, t_o	(second)	3
Total time, τ	(second)	5
Step size, Δt	(second)	0.005
Parameter, γ		0.5
Parameter, β		0.25

which the λ_2 angles of both the rigid and the flexible models increase to four degrees while the λ_1 angles increase to only two degrees. The periodic spin-up frequency of 1 *Hz* affects the rigid body nutation as evidenced by that frequency appearing in the plot. Figures 6.13 and 6.14 illustrate the rigid body angular velocities (nutating rates), $\dot{\lambda}_1$ and $\dot{\lambda}_2$, over a five second simulation period. Velocity 2 also grows faster than velocity 1 as does rigid body angle 1, λ_1 , shown in Figure 6.12. The plots of the rigid body angular accelerations, $\ddot{\lambda}_1$ and $\ddot{\lambda}_2$, are shown in Figures 6.15 and 6.16. The typical patterns of the rigid body model and the flexible model are observed and compared in Figure 6.15. The instantaneous value of the flexible model oscillates around the value of the rigid body model. More modes and a large peak value of $\dot{\lambda}_1$ also show up in this flexible model plot.

Elastic deflections and rotations of each flexible structure with or without initial rigid body tilt are compared in Figures 6.17 - 6.20. The tangential deflections of beams 1 and 3 in the corresponding local *Y* direction are shown in Figures 6.17 and 6.18. The tangential deflections with initial rigid body tilt of $\lambda_1 = 1$ deviate from the corresponding counterparts with no initial tilt (see Figures 6.17 and 6.18) as the rigid

body nutating angles grow dramatically after four seconds (see Figure 6.12). In the next four plots with or without tilt as shown in Figures 6.19 and 6.20, comparisons of beam 1 vertical deflections and beam 3 radial deflections are illustrated. The same ‘run-away’ phenomenon is also expected and observed in Figures 6.19 and 6.20. The deflections of beam 1 in the local Y and Z directions and their axisymmetric counterparts of beam 5 are compared in Figures 6.21 and 6.22. The values of beam 1 and beam 5 start to separate, run away, and head for the opposite directions after approximately four seconds (see Figures 6.21 and 6.22). The twist phenomenon of beams 3 and 4 are also observed in this run case (see Figure 6.23). Instead of remaining twisted in run Case 1 with no initial tilt, the radial deflections of beams 3 and 4 are in phase in this case once the spin velocity reaches steady state as shown in Figure 6.23. The tangential and radial deflections of beam 3 in the local coordinates are compared with the corresponding deflections of the axisymmetric counterpart, beam 7, in Figures 6.24 and 6.25. The tangential deflections are in phase (see Figure 6.24) while the radial deflections are out of phase (see Figure 6.25).

6.4 Run Case 4: General Motion with Excitation

In this run case, an impulse of 1 *Newton* is applied on one of the tanks in an upward direction after the lower shaft spins for 1.5 seconds. The spin profiles are the same as those used in the previous cases except that it takes only 1 second to drive the lower shaft from zero to 60 *rpm*. The running conditions and parameters are listed in Table 6.10.

During the period of increasing rigid body angular velocity, from 0 to 1 second, the collar is set in its up position and no nutation is allowed. After 1 second the

Table 6.10: Case 4 conditions and parameters

Tilt, λ_1	(degree)	0
Impulse	(Newton)	1
Velocity, ω_o	(rpm)	60
Time base, t_o	(second)	1
Total time, τ	(second)	10
Step size, Δt	(second)	0.005
Parameter, γ		0.5, 0.6
Parameter, β		0.25, 0.3025

collar is suddenly dropped and the universal joint is free to nutate. At the time of 1.5 seconds an impulse with the magnitude of 1 *Newton* suddenly acts on a tank to initiate excitation of the structures. The upper shaft experiences a relatively large nutation before it regains stability as evidenced by the large variations of the rigid body rotating angles of λ_1 and λ_2 as shown in Figure 6.26. The corresponding rigid body velocities and accelerations are plotted in Figures 6.27 and 6.28. The time histories of tanks 1 and 2 center positions are shown in Figures 6.29 - 6.31 in which the Z coordinates of tank 1 center and tank 2 center are compared in Figure 6.31. The influence of the rigid body motion on the elastic deformation of the structures is very significant as shown in Figures 6.32 - 6.34. Large transient values are also observed in the beam tangential, vertical, and radial deflection profiles. These values will not disappear unless there is some kind of damping in the system. The change of the Newmark parameters from $\gamma = 0.5$ and $\beta = 0.25$ to $\gamma = 0.6$ and $\beta = 0.3025$ will artificially add numerical damping to the system in the time integration. This effect of numerical damping is verified in Figures 6.35 - 6.37. Beams 1 and 5 are much stiffer than beams 3 and 7 so that the transient values of rotation for beams

1 and 5 are much larger than the transient values of beams 3 and 7 also shown in Figures 6.35 - 6.37.

6.5 Run Case 5: Fluid-structure Interaction

In this interaction run case, fluid sloshing motion is modeled using the CFD technique. The fluid-structure interaction mechanisms are investigated under a joint effort of the CFD modeling and the *flexible system dynamics* (FSD) modeling. As part of the research on the satellite project, two computer codes have been developed: the FSD code computes the overall test rig dynamics and the CFD code calculates the sloshing motion of the fluid in the tank. A master program has been written to call these two codes and to control the way of transferring information between them after every time step during the execution. Three different interaction modes are defined in the following according to the method of information transfer:

- Non-interaction: No information is transferred between the codes. Two codes are executed separately.
- One-way interaction: Only the information from the CFD code is transferred to the FSD code. There is no information input to the CFD code from the FSD code.
- Two-way interaction: Information goes back and forth between the two codes.

In a non-interaction mode, The FSD code assumes the fluid to be a solid mass which is lumped at the initial location of the mass center of the tank. Similarly, the CFD code assumes the tank to undergo a simple rotary motion about the spin axis. In an interaction mode, on the other hand, the FSD code gets the location of the fluid mass center and the six components of moments of inertia as input at the

Table 6.11: Case 5 conditions and parameters

Tilt, λ_1	(degree)	(none)
Impulse	(Newton)	(none)
Velocity, ω_o	(rpm)	30
Time base, t_o	(second)	0.5
Total time, τ	(second)	2
Step size, Δt	(second)	0.001
Parameter, γ		0.6
Parameter, β		0.303

beginning of every time step. This helps correlate the effects of the liquid sloshing on the structure. In return, the FSD code passes on the instantaneous positions, velocities, and accelerations of the tank to the CFD code at the end of every time step, resulting in an accurate kinematic representation of the tank.

In this section, study is focused on investigating the effects of liquid sloshing on the structural deflections for a stable spin-up case. The specific conditions and the values of parameters are listed in Table 6.11.

In Figure 6.38, radial deflections at the tank center for three modes are compared. The results of the one-way and two-way interactions are so close in the entire simulation period that they are almost identical. Three curves remain nearly identical in the increasing spin-up period (see Figure 6.38) but the curve of the non-interaction mode runs away from the other two once the spin-up speed reaches its constant value. A different steady-state value of the radial deflection at the tank center for the non-interaction and interaction modes is found in the simulation (see Figure 6.38). The tangential deflections at the tank center for three modes are plotted in Figure 6.39. Three curves are kept close during the increasing spin-up period as well as the con-

stant velocity period. The peak values of the non-interaction mode are larger than the counterparts of the interaction modes. This is expected because in the interaction modes the information of the liquid damping is passed on from the CFD code to the FSD code. During this initial interaction run, it was noticed that it is difficult for the computation sensitive CFD code to handle elastic deformation with large transient values. It was therefore decided to introduce numerical damping using Newmark parameters of $\gamma = 0.6$ and $\beta = 0.303$ to minimize the transient values. This somehow decreases the sense of the effect of liquid damping. However, it was observed from the results that the numerical damping is more effective than the liquid damping in computation as there is very little difference between the non-interaction and interaction modes in identifying transient values from the plots.

6.6 Experimental Measurements

The configuration of the satellite test rig (dynamic part) for the experimental measurements is shown in Figure 6.2. The physical representation of each individual structure is set to identical dimensions as the computational simulation model. Both tanks are half filled with glycerin which was chosen as the test liquid in order to correlate CFD modeling. The dynamic part of the test rig system is powered by a DC drive motor through a drive train. The motor is controlled manually rather than automatically. The collar which can cover the universal joint could be in its up or down position depending on the needs.

One arm (involving beams 3 and 4) of the test rig is instrumented using four strain gauges mounted on the front and rear sides of each of beams 3 and 4. The strain gauges are circuited as a full bridge from which the output signal (voltage) is

amplified using an operational amplifier. This bridge system is powered using two sets of DC battery with 10 V each, and is connected to a slip ring with multi-leads for output. The rotational speed of the lower shaft is measured by a tachometer connected to the motor drive train. All the measurement outputs are connected to an IBM PS/2 model 50 computer outfitted with a National Instruments' MIO-16 data acquisition board. This hardware is currently configured to accept 8 channels of bipolar voltage signals (± 10 V), and is capable of a maximum data acquisition rate of 90,000 samples per second.

Prior to performing the experiments with the test rig, the bridge system was calibrated to obtain its sensitivity (voltage versus beam deflection relationship) within the linear range. It was determined that in a reasonable range of beam deflection the sensitivity of the bridge system remained constant. This calibration was accomplished by applying known deflection of the beam and recording the output voltage from the bridge system.

It was decided to run the experimental tests for a stable spin-up case with the collar in its up position to restrain the rigid body nutation at the universal joint. Due to the fact that the CFD modeling is still in development to handle the general motion cases, the experimental tests are therefore also restricted. Since the speed of the test rig's DC motor is controlled manually using a transformer, the transient variation and the profile of the rotational speed could not be made repeatable. It was decided, therefore, to perform several runs by manually varying the drive motor from 0 to 60 rpm over a time interval of approximately 3 seconds. The "best" profile would then be selected as the input speed profile to be read by the FSD code, and the corresponding numerical results are computed. Figure 6.40 shows the spin

velocity profiles in which the dotted one is the measured one and the solid curve is the ideal sinusoidal profile. For the current mounting of the strain gauges, they can detect not only the local beam deformation but also the elastic deformation of the beams preceding the current beam. The most critical, sensitive, and significant elastic deformation, the overall radial deflection at the tank center, is measured, which is then compared with the computational result as shown in Figure 6.41. The results are very close both in their patterns and in their magnitudes for the transient and steady-state values. The difference between the computational result and experimental data is within 5%. The computed curve oscillates about the measured curve during the initial spin period (within one second) as shown in Figure 6.41. This is because the initial elastic deformations for the flexible beams in the computer simulation are set to zero. In other words, the structural system is undeformed initially. as a consequence, there is a sudden force acting on the tank due to its heavy weight. On the other hand, the same overall radial deflection at the tank center is measured when the structural system is in its deformed and hence statically balanced configuration.

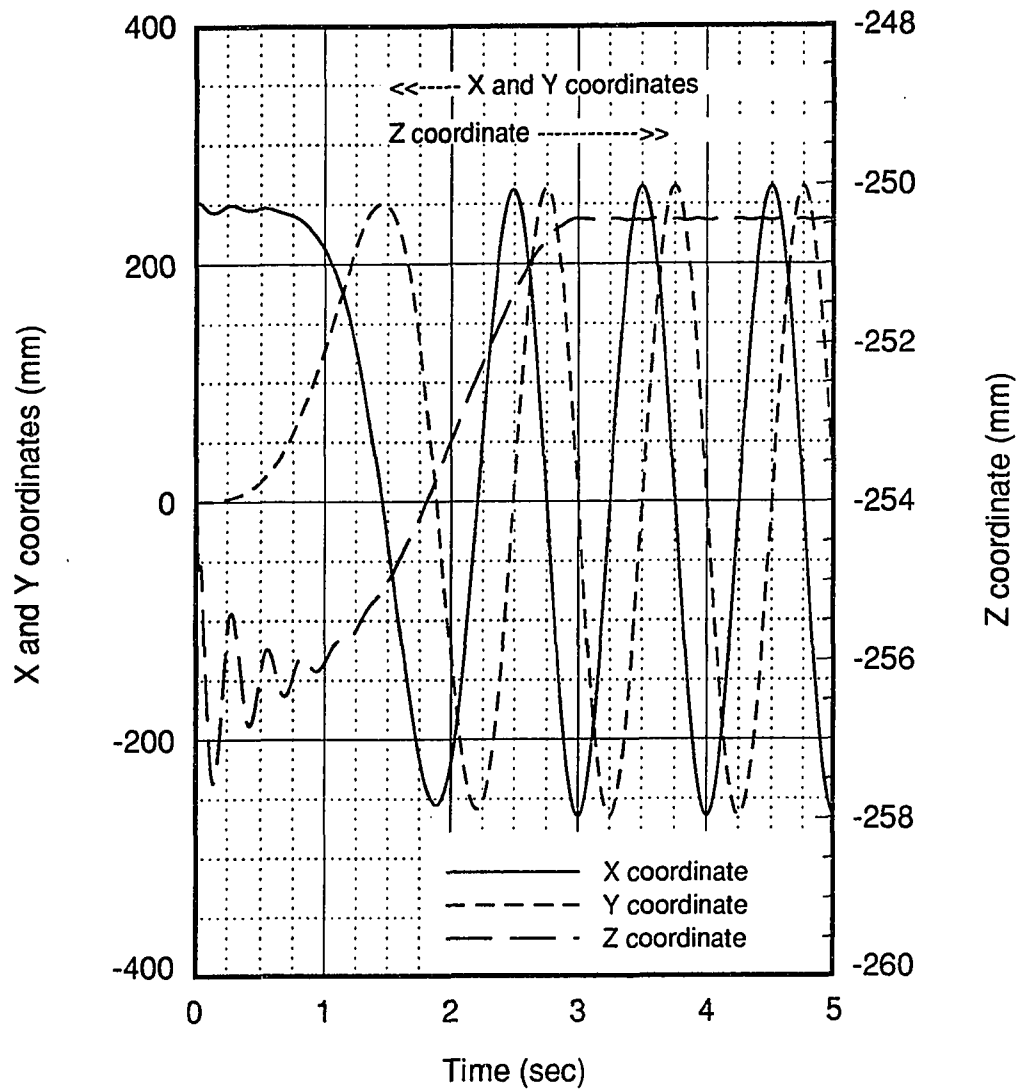


Figure 6.3: Trajectory of a tank center relative to an inertial frame during spin-up with collar up: $t_0 = 3$ seconds, $\omega = 60rpm$

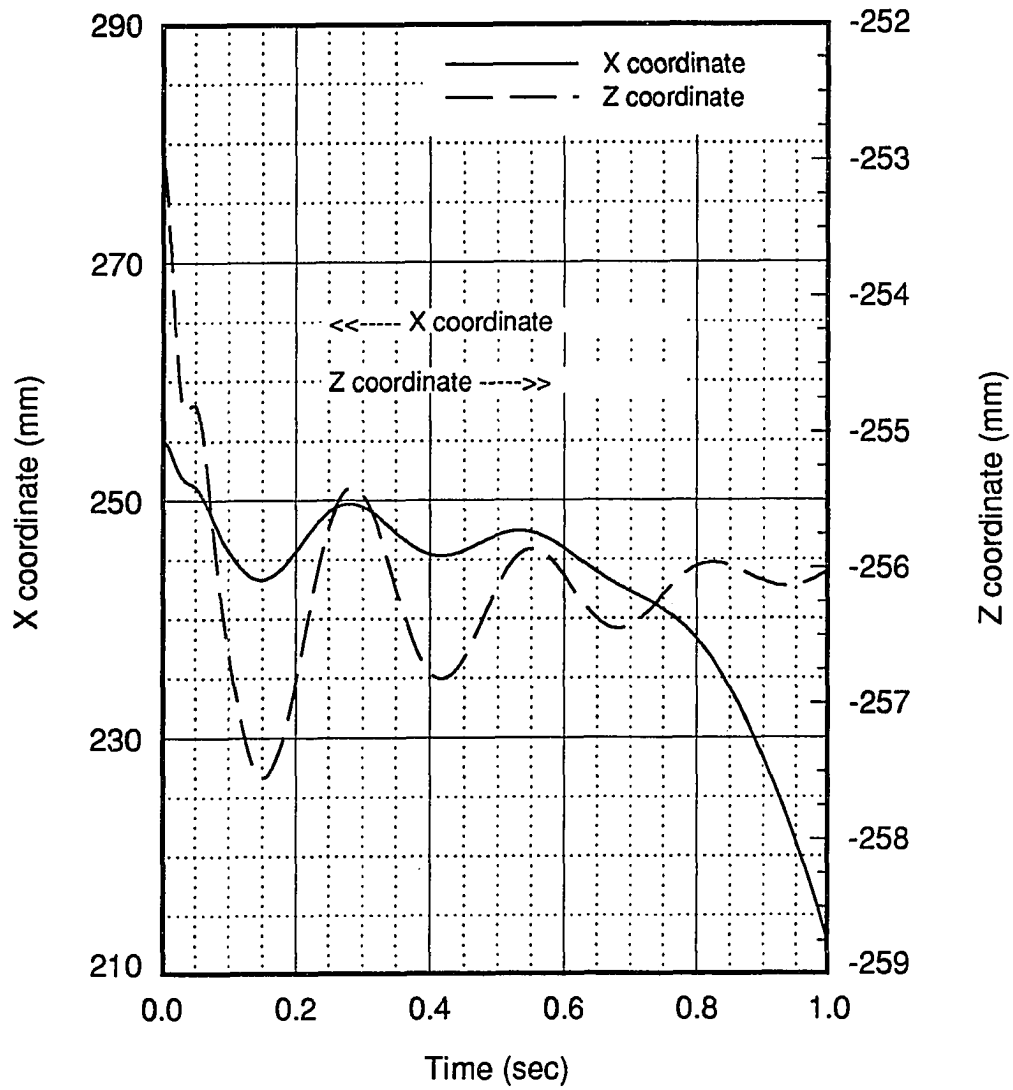


Figure 6.4: Trace of X and Z inertial coordinates of a tank center in the first one second during spin-up with collar up: $t_0 = 3$ seconds, $\omega = 60rpm$

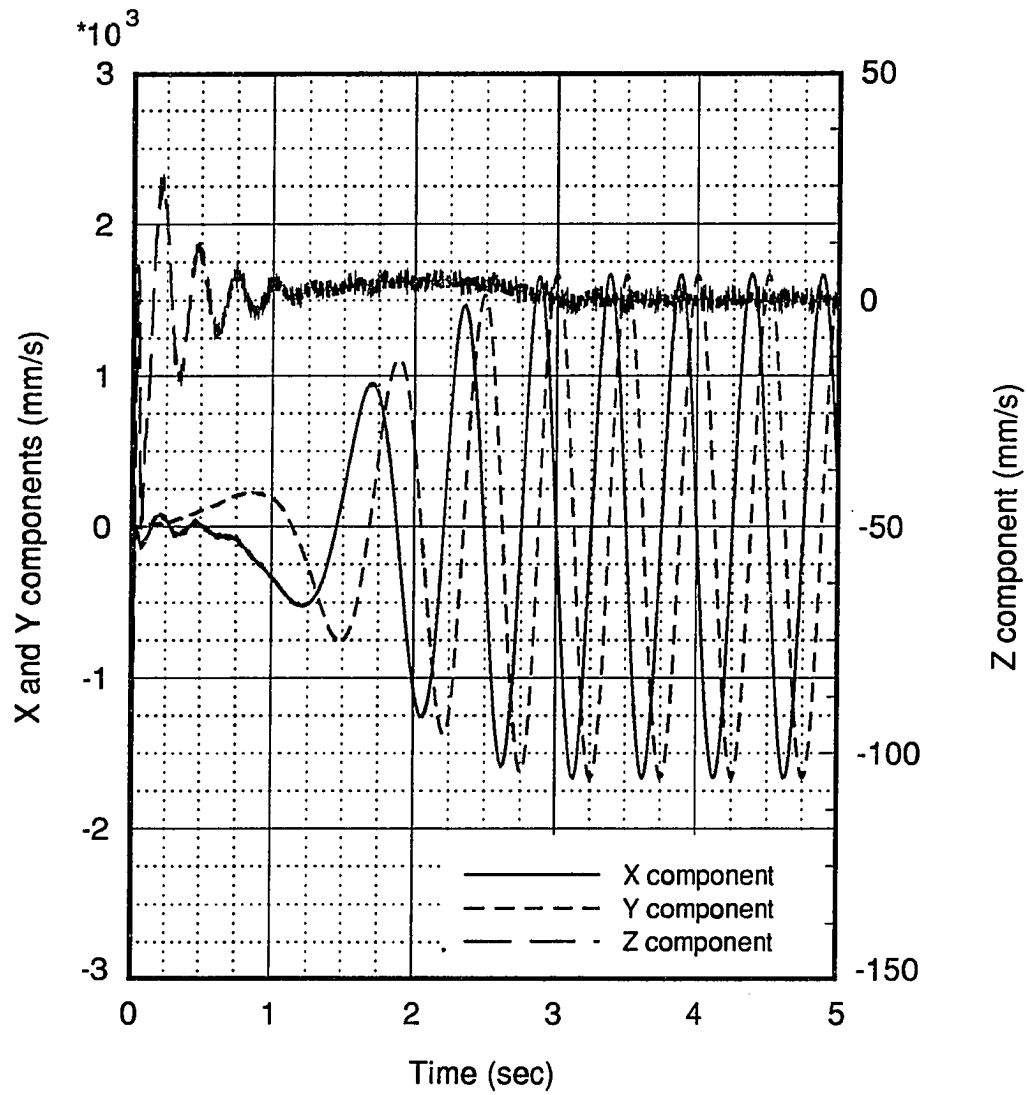


Figure 6.5: Time history of tank center velocity relative to an inertial frame during spin-up with collar up: $t_0 = 3$ seconds, $\omega = 60rpm$

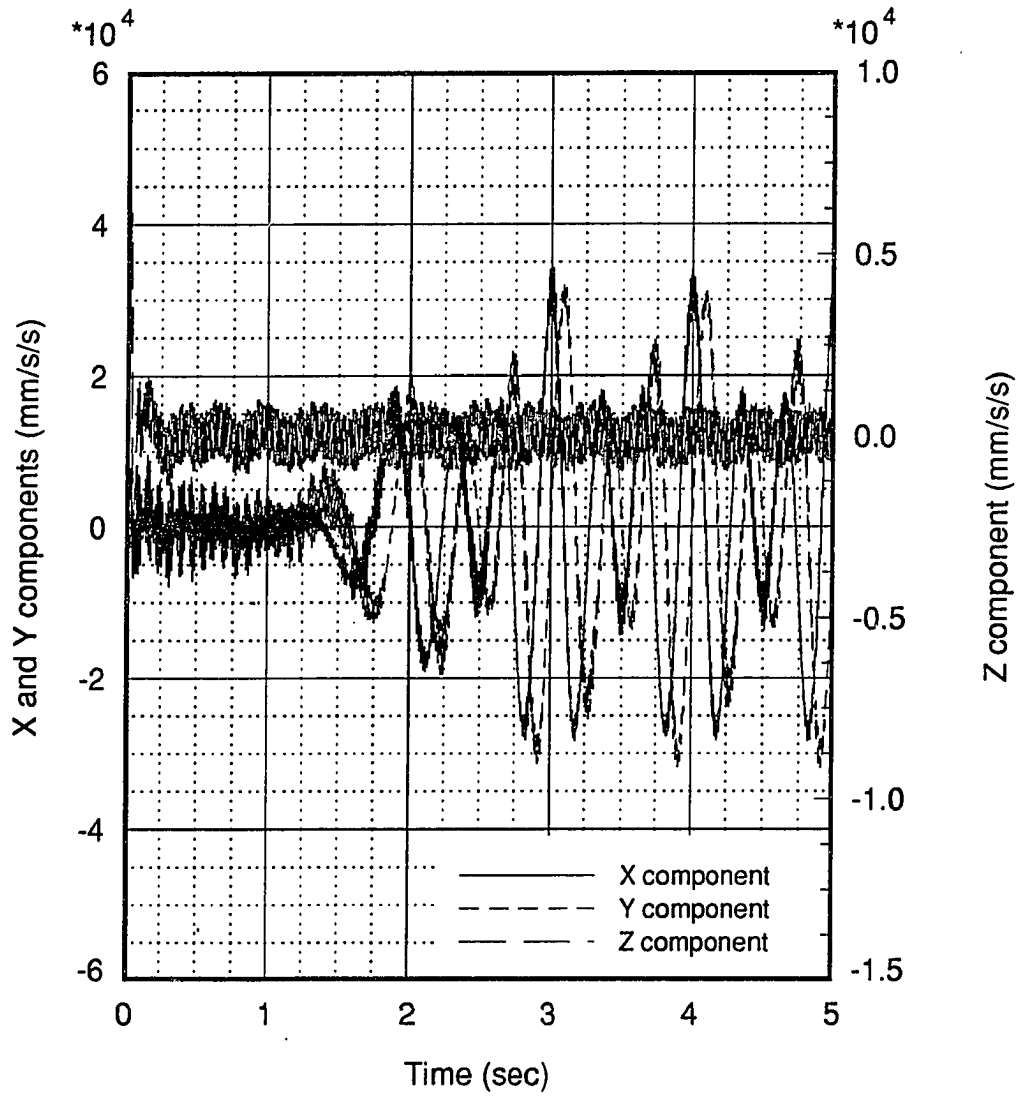


Figure 6.6: Time history of tank center acceleration relative to an inertial frame during spin-up with collar up: $t_0 = 3$ seconds, $\omega = 60rpm$

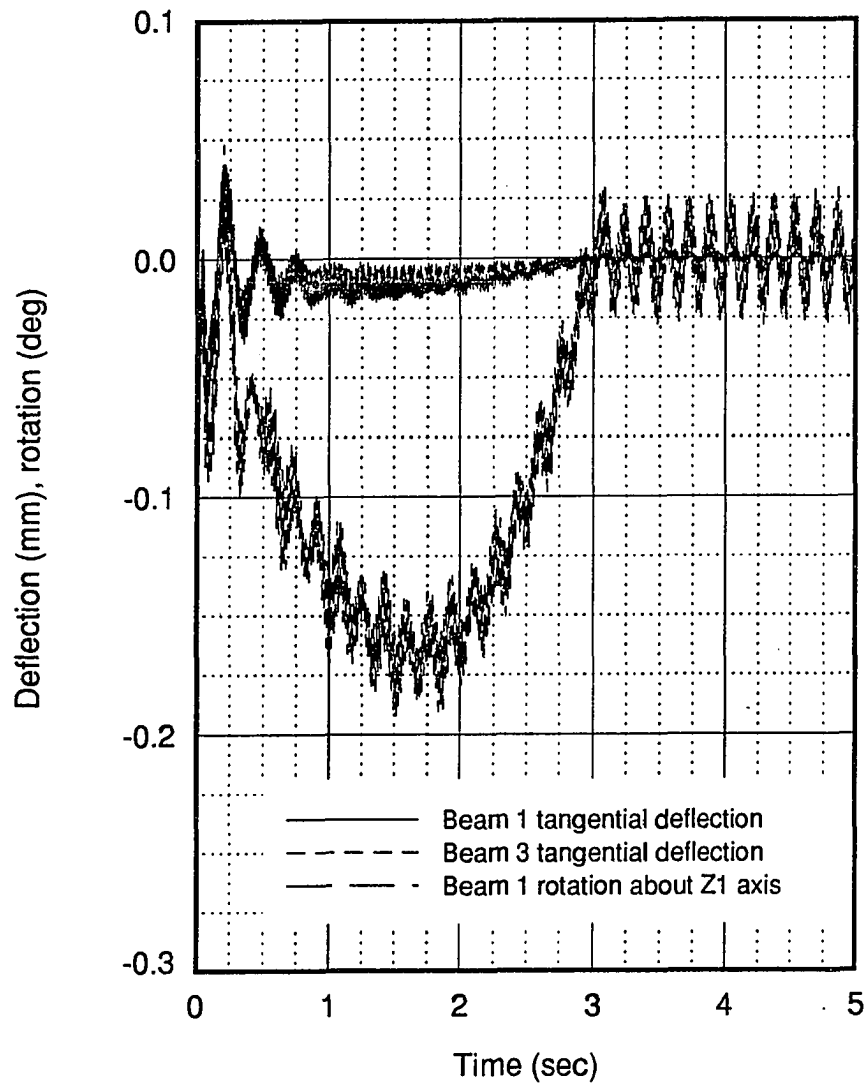


Figure 6.7: Time history of bending deformation of beams 1 and 2 in the corresponding local $x - y$ and $y - z$ planes during spin-up with collar up: $t_0 = 3$ seconds, $\omega = 60rpm$

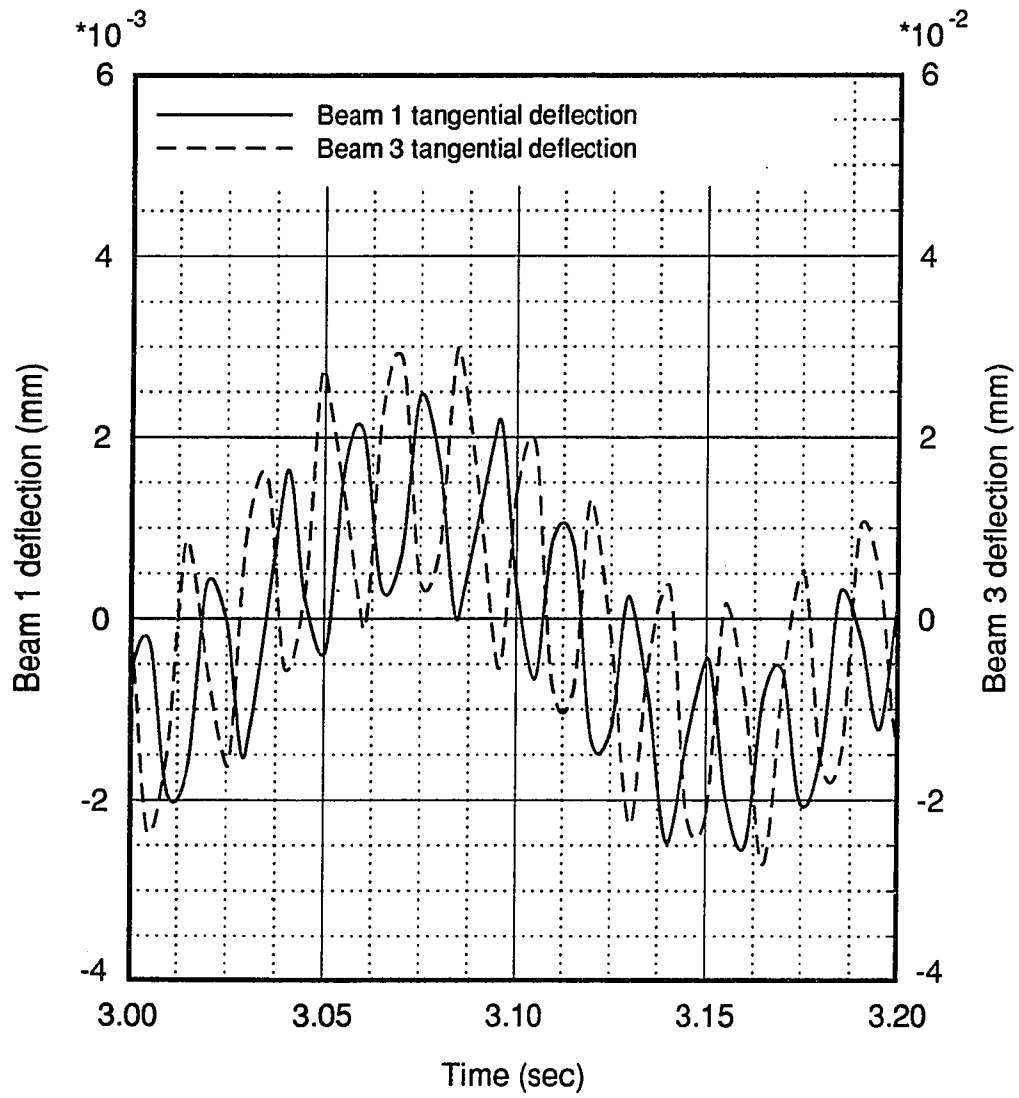


Figure 6.7 (Continued)

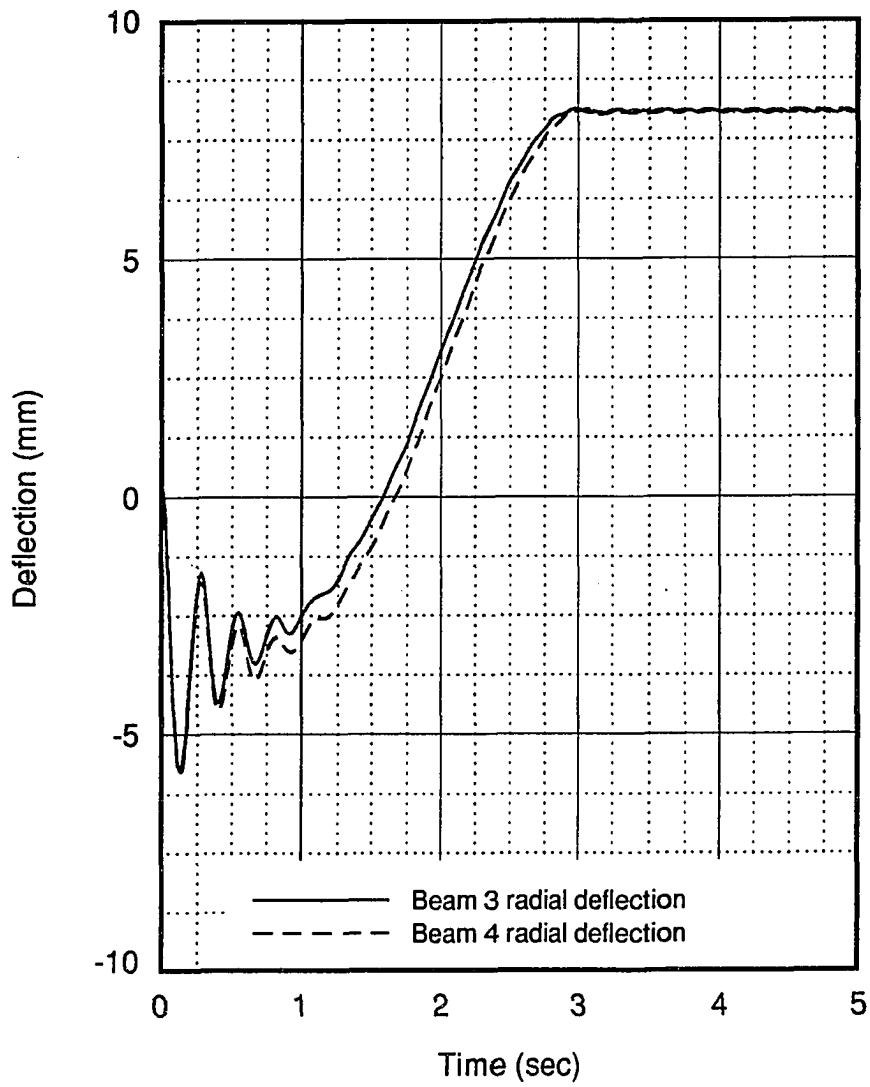


Figure 6.8: Time history of bending deflections of beams 3 and 4 in the local $x - z$ plane during spin-up with collar up: $t_0 = 3$ seconds, $\omega = 60rpm$

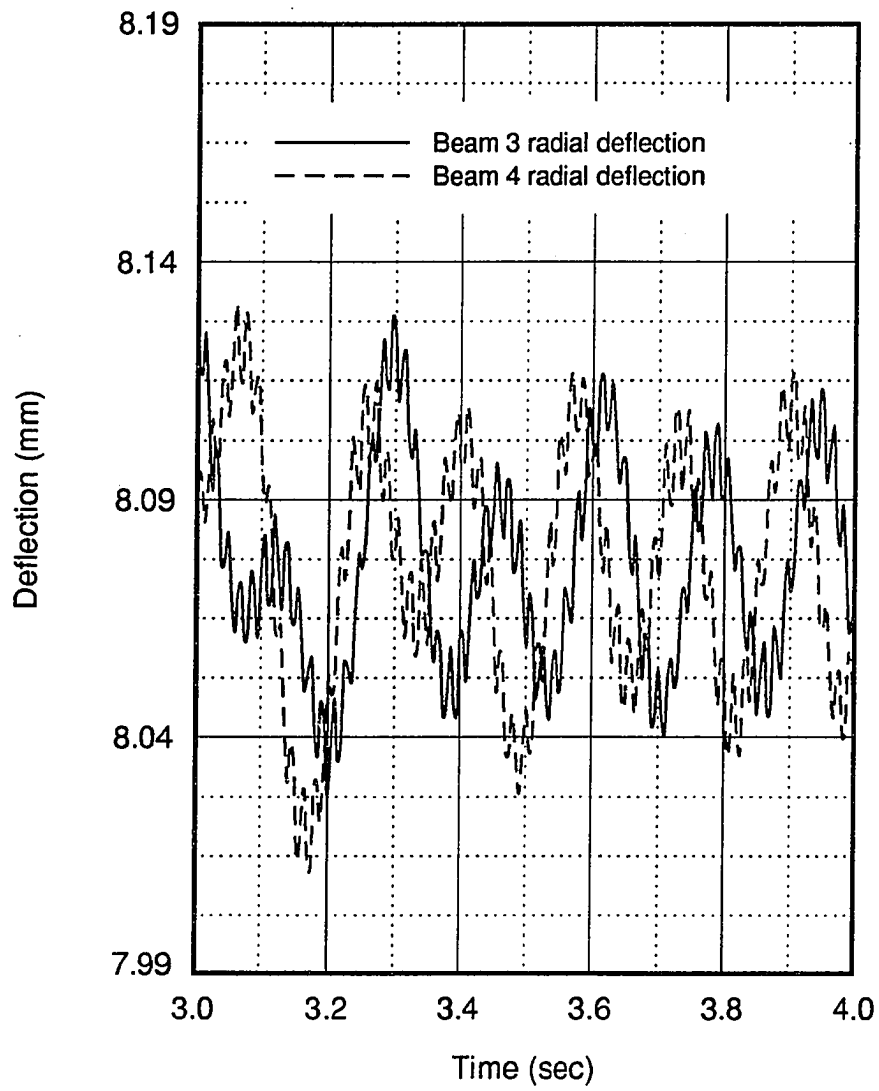


Figure 6.8 (Continued)

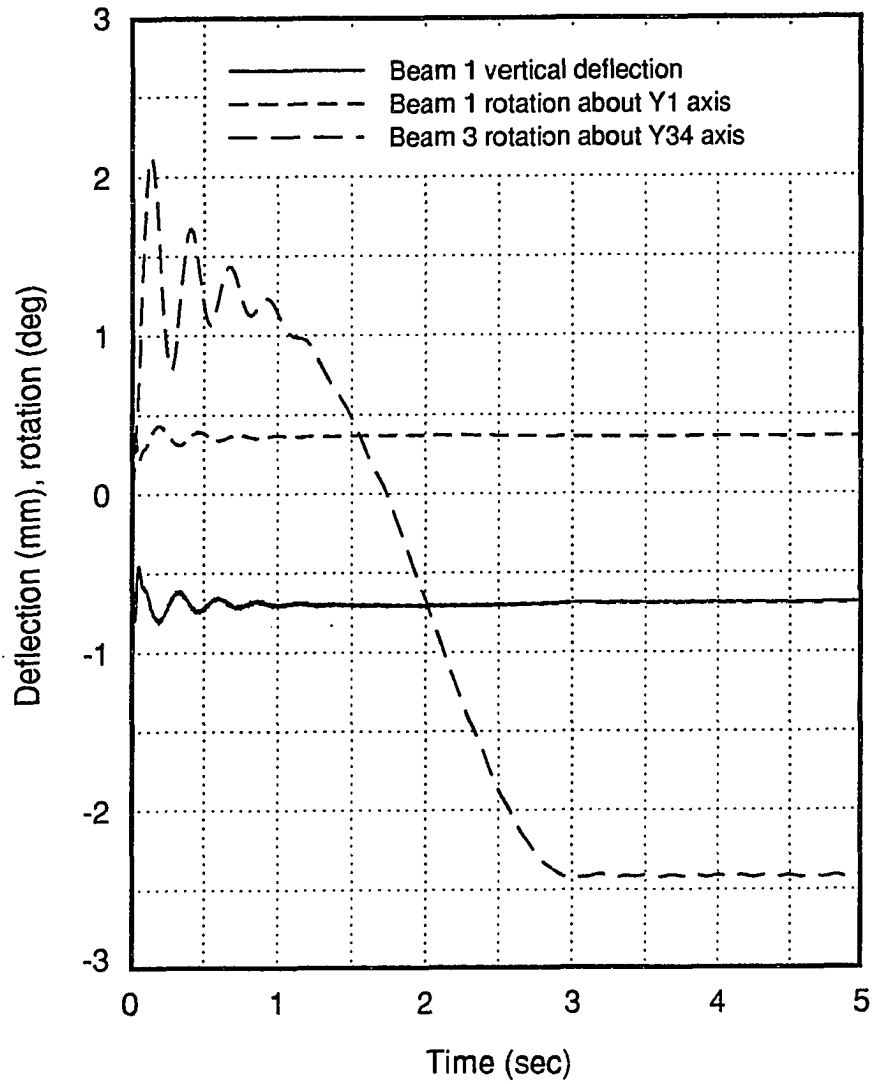


Figure 6.9: Time history of bending deformation of beams 1 and 2 in the corresponding local $x - z$ planes during spin-up with collar up: $t_0 = 3$ seconds, $\omega = 60rpm$

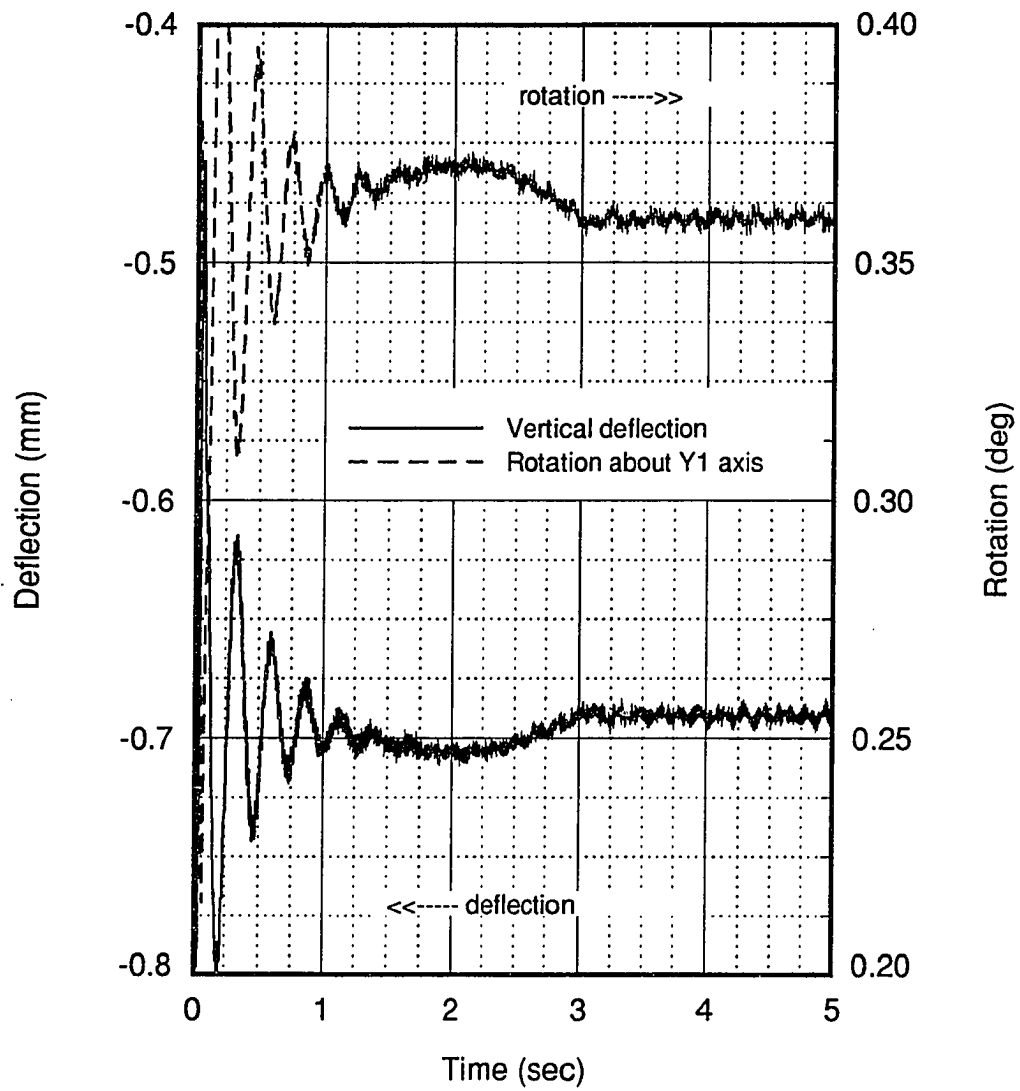


Figure 6.9 (Continued)

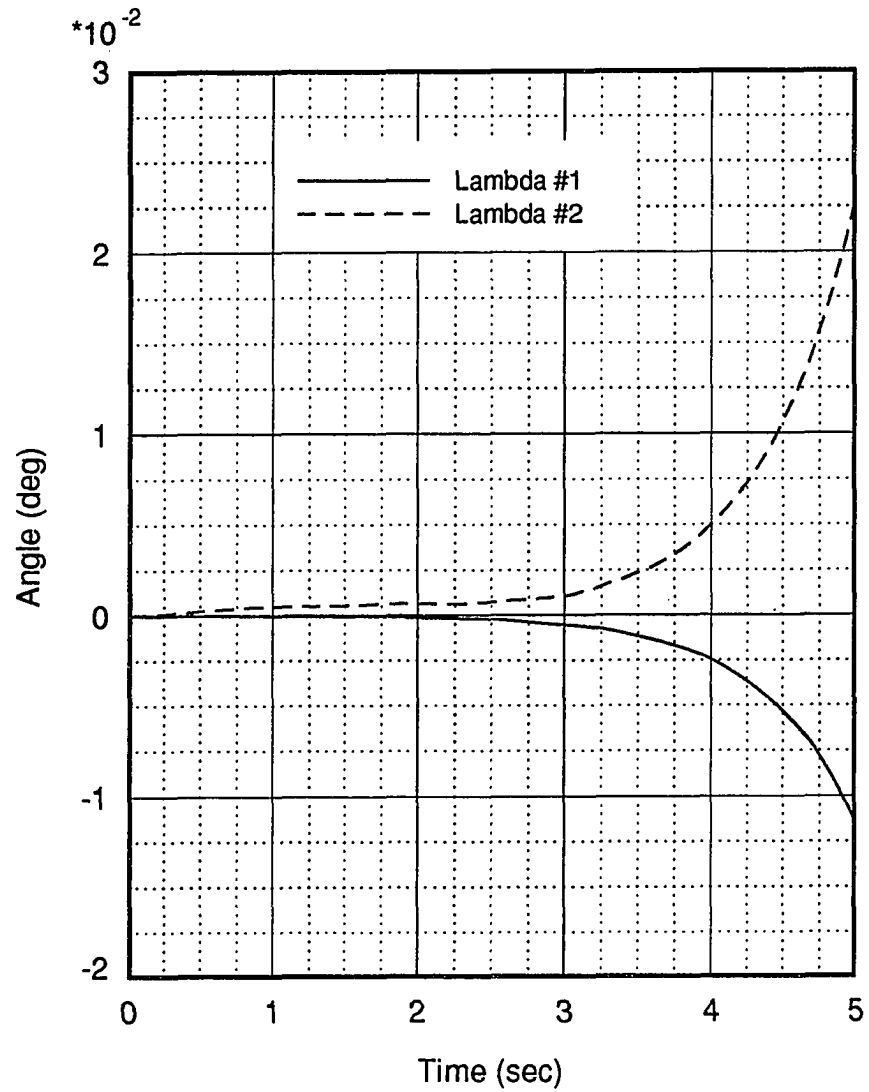


Figure 6.10: Time history of rigid body nutating angles with collar down and no initial tilt: $t_0 = 3$ seconds, $\omega = 60rpm$

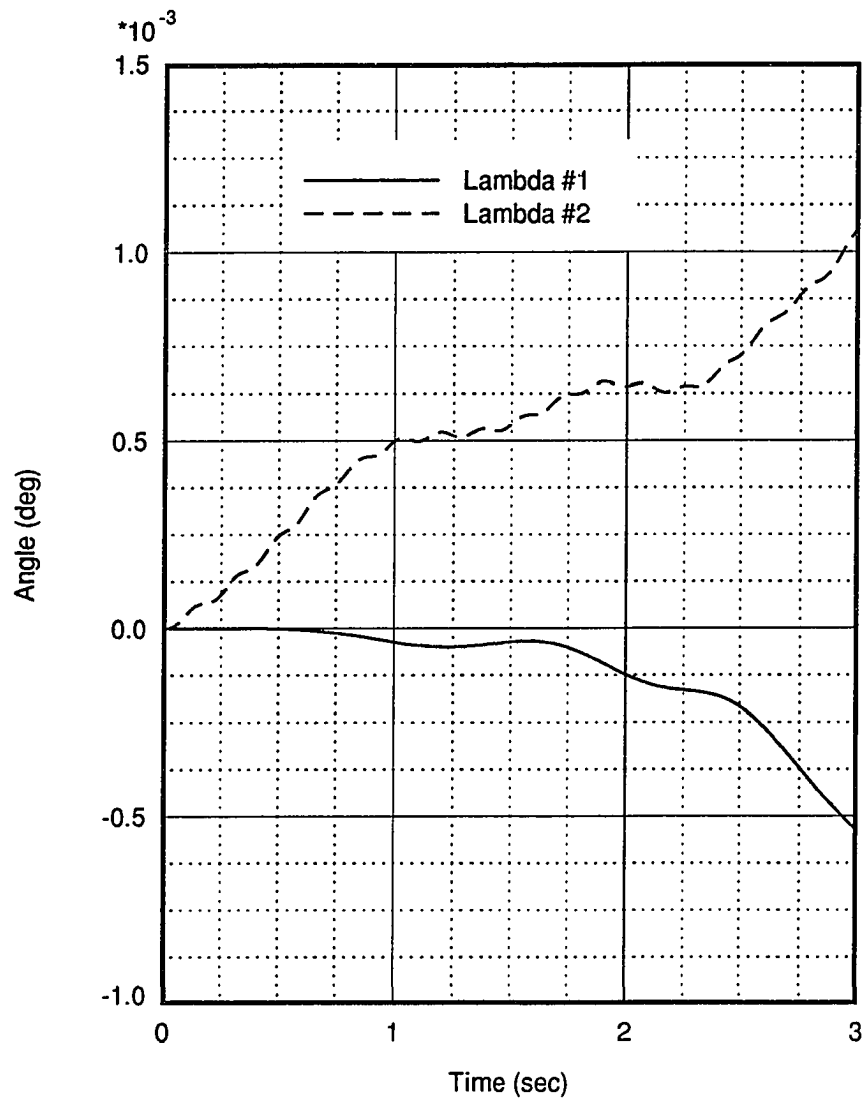


Figure 6.10 (Continued)

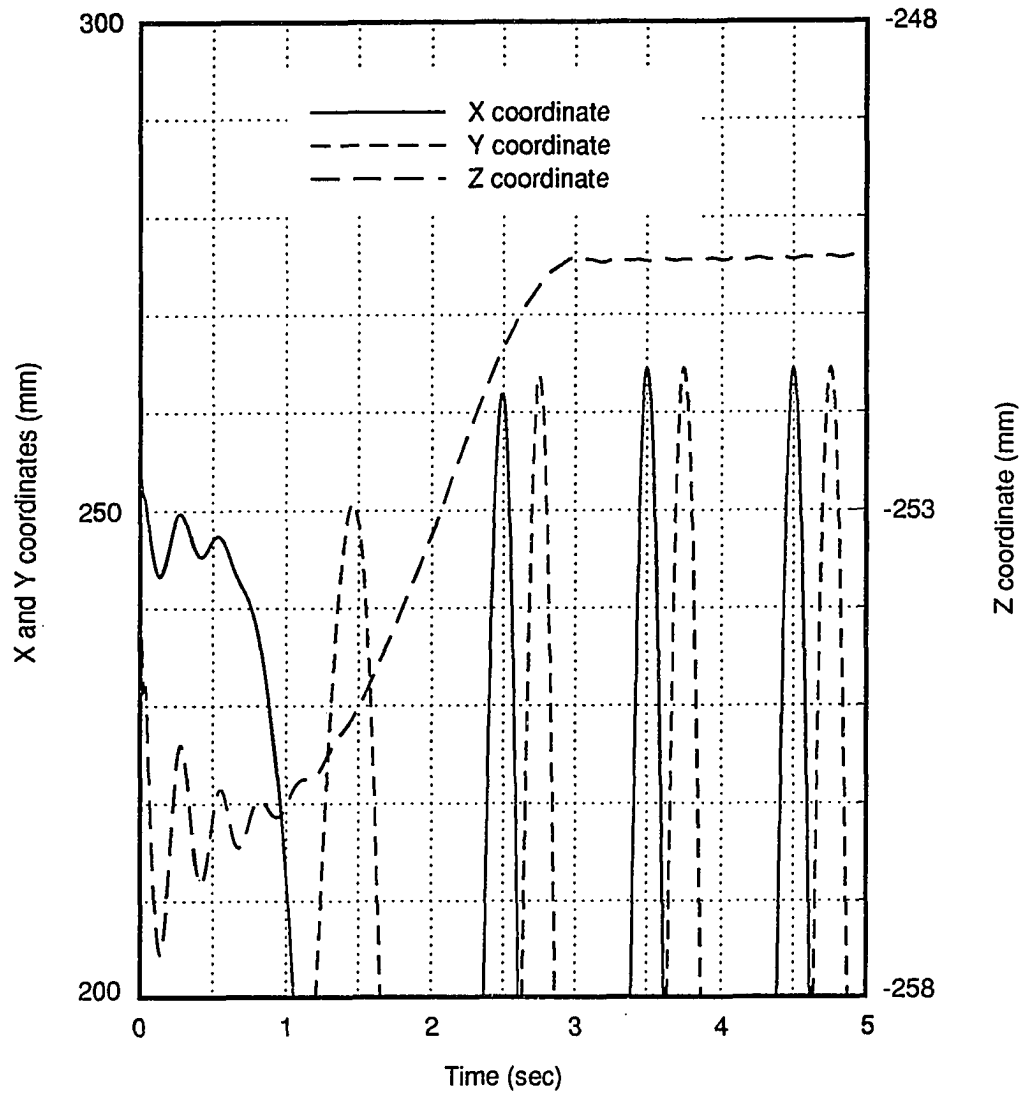


Figure 6.11: Trajectory of a tank center in an inertial frame with collar down and no initial tilt: $t_0 = 3$ seconds, $\omega = 60rpm$

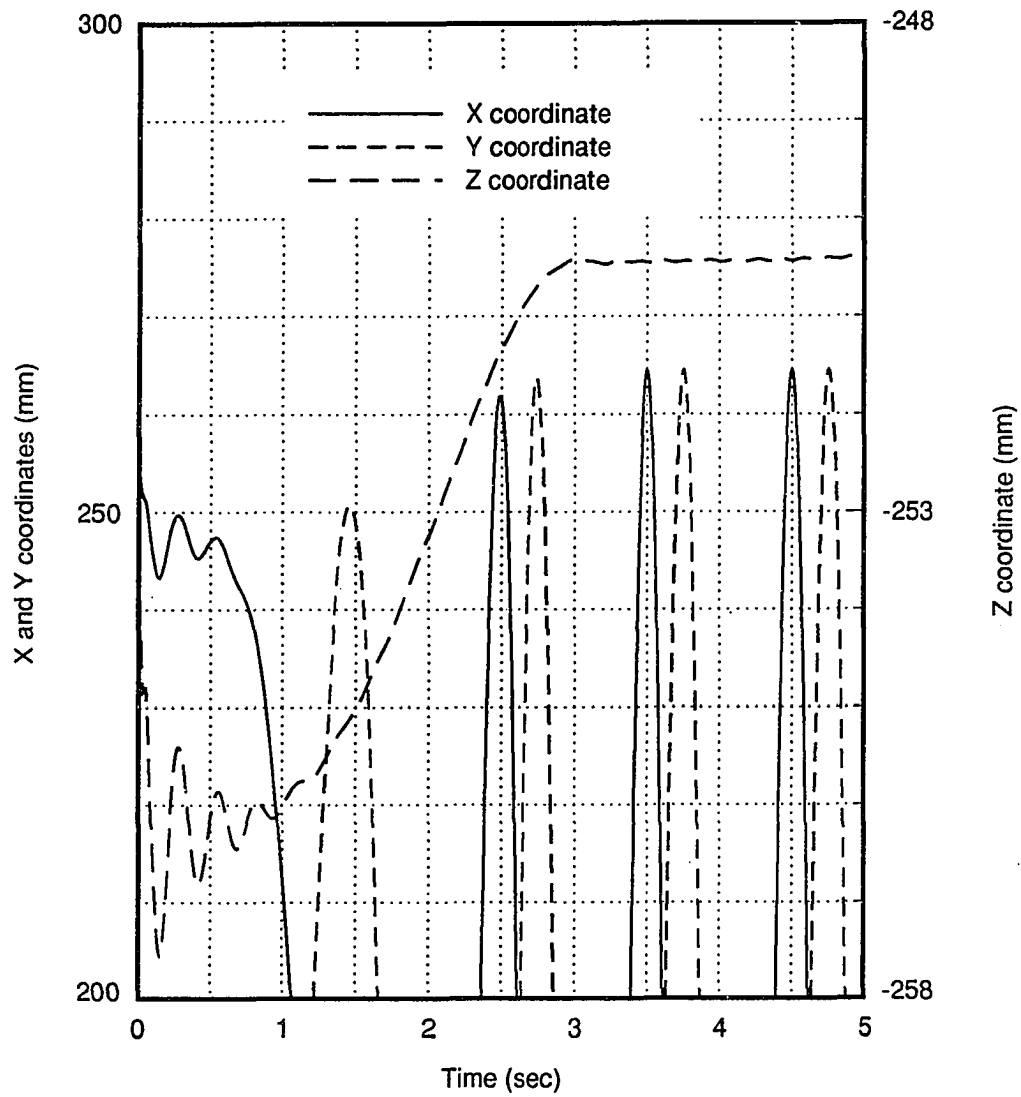


Figure 6.11 (Continued)

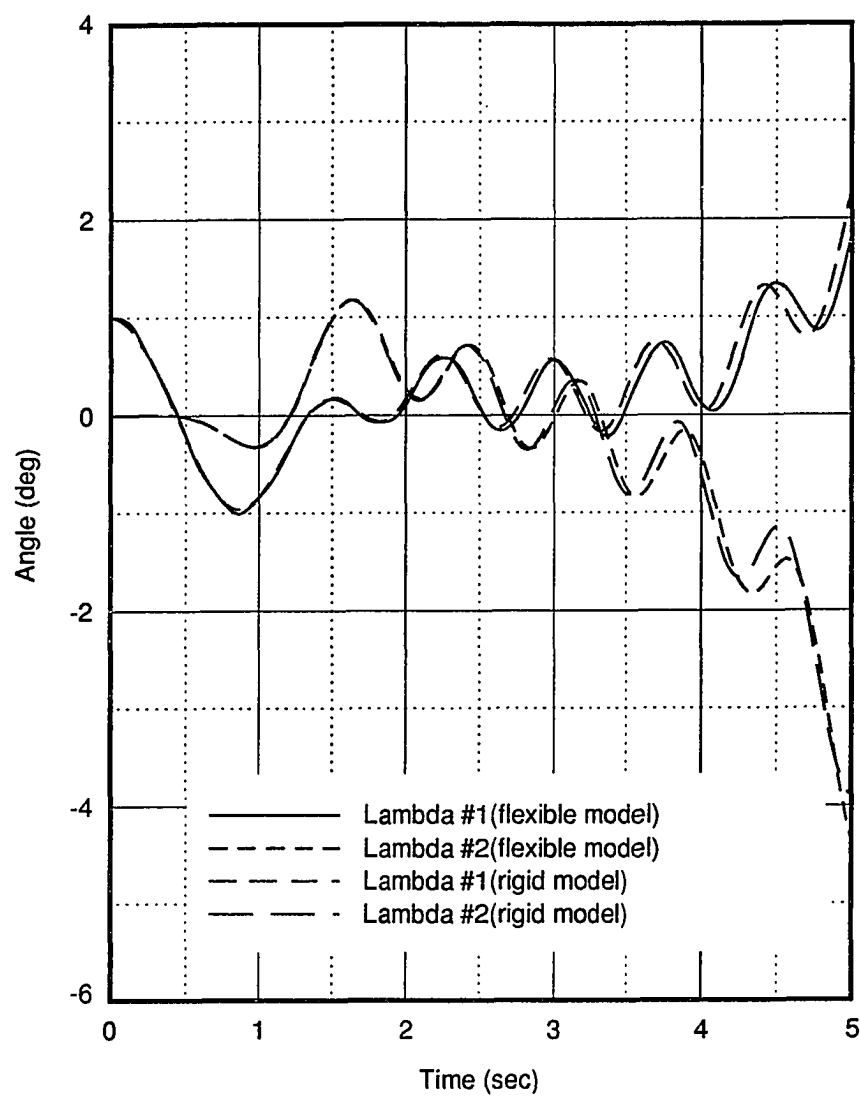


Figure 6.12: Time history of rigid body nutating angles with collar down and initial tilt: $t_0 = 3$ seconds, $\omega = 60rpm$, $\lambda_1 = 1$ degree

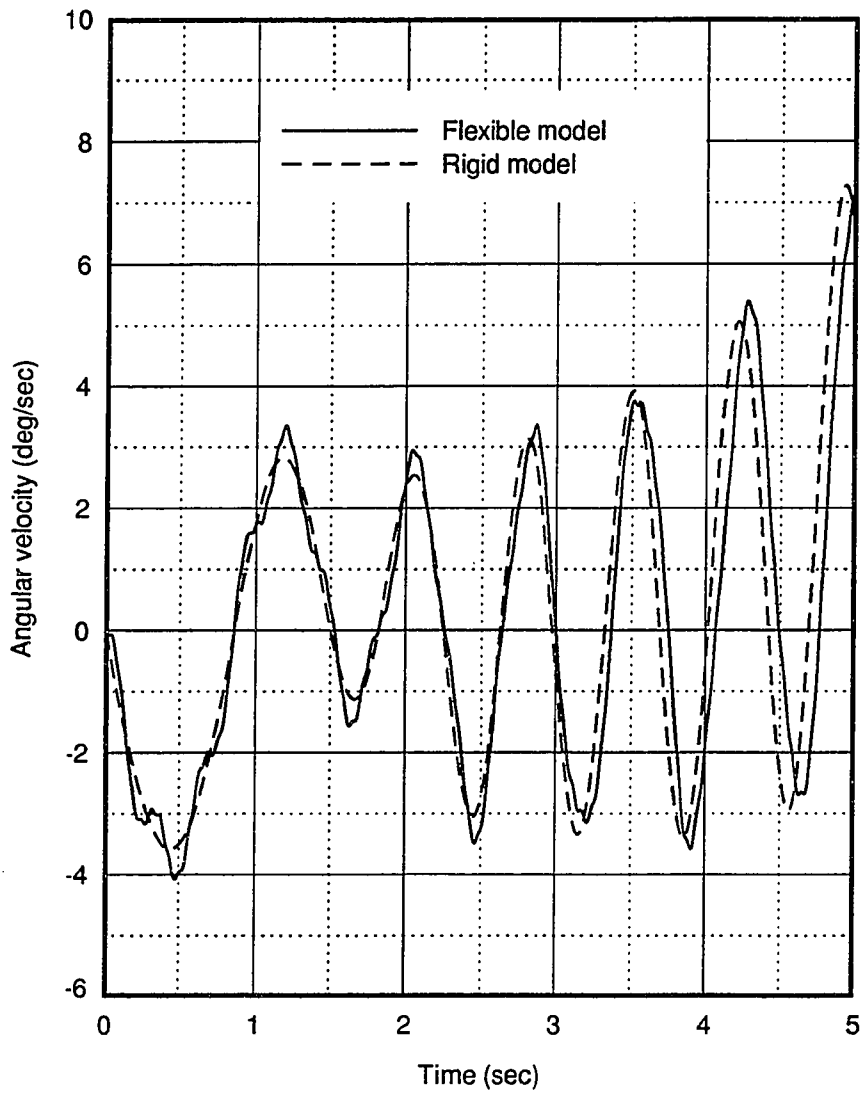


Figure 6.13: Time history of rigid body angular velocity 1, $\dot{\lambda}_1$, with collar down and initial tilt: $t_0 = 3$ seconds, $\omega = 60rpm$, $\lambda_1 = 1$ degree

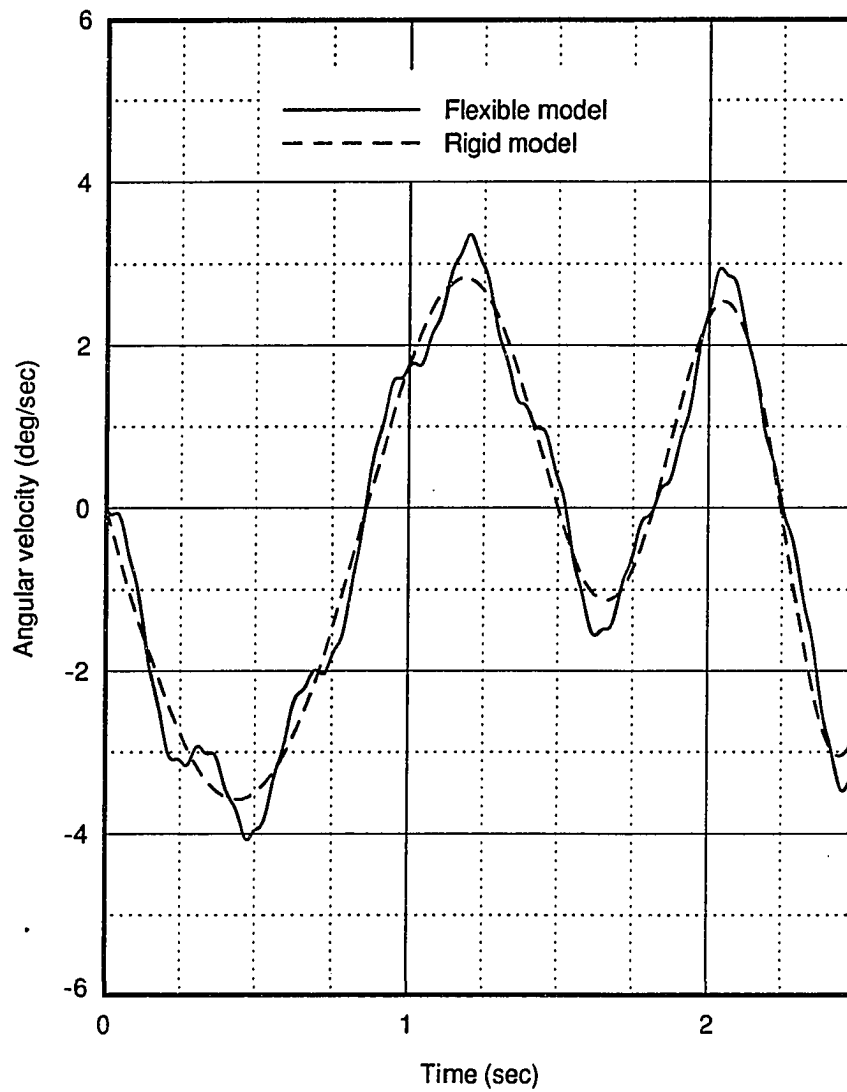


Figure 6.13 (Continued)

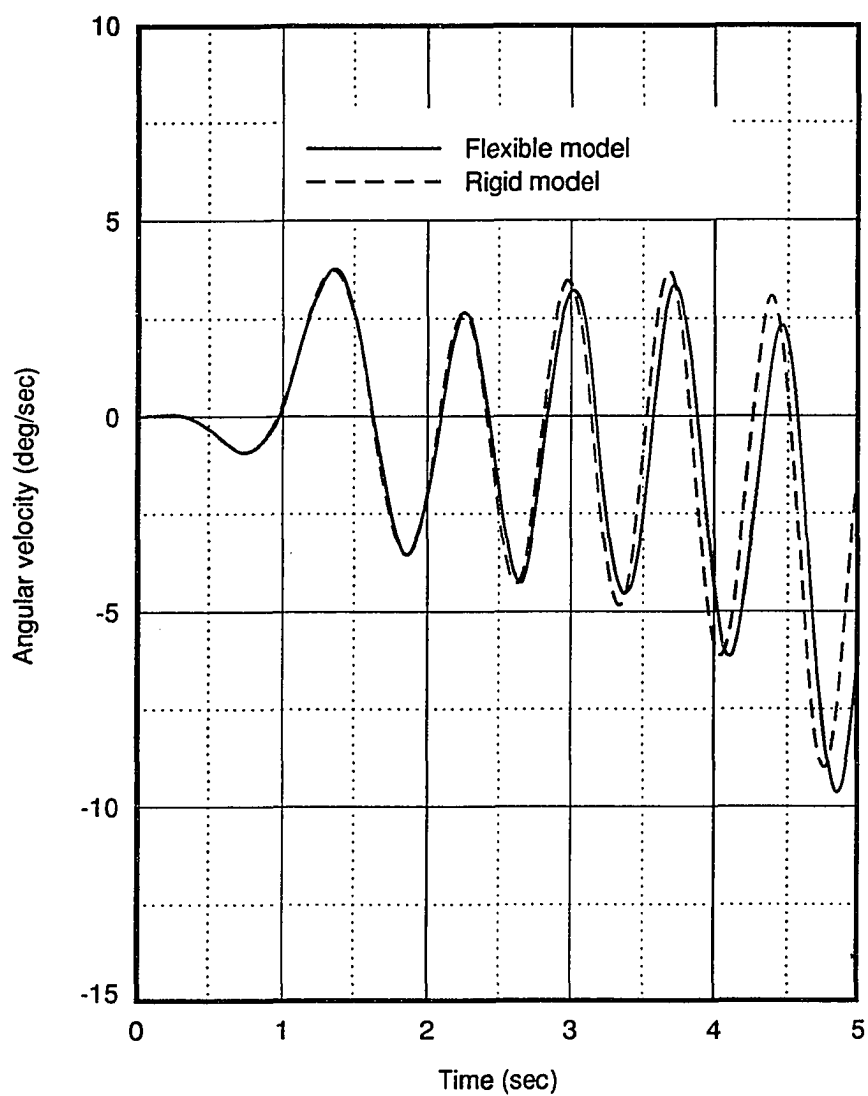


Figure 6.14: Time history of rigid body angular velocity $2, \dot{\lambda}_2$, with collar down and initial tilt: $t_0 = 3$ seconds, $\omega = 60rpm$, $\lambda_1 = 1$ degree

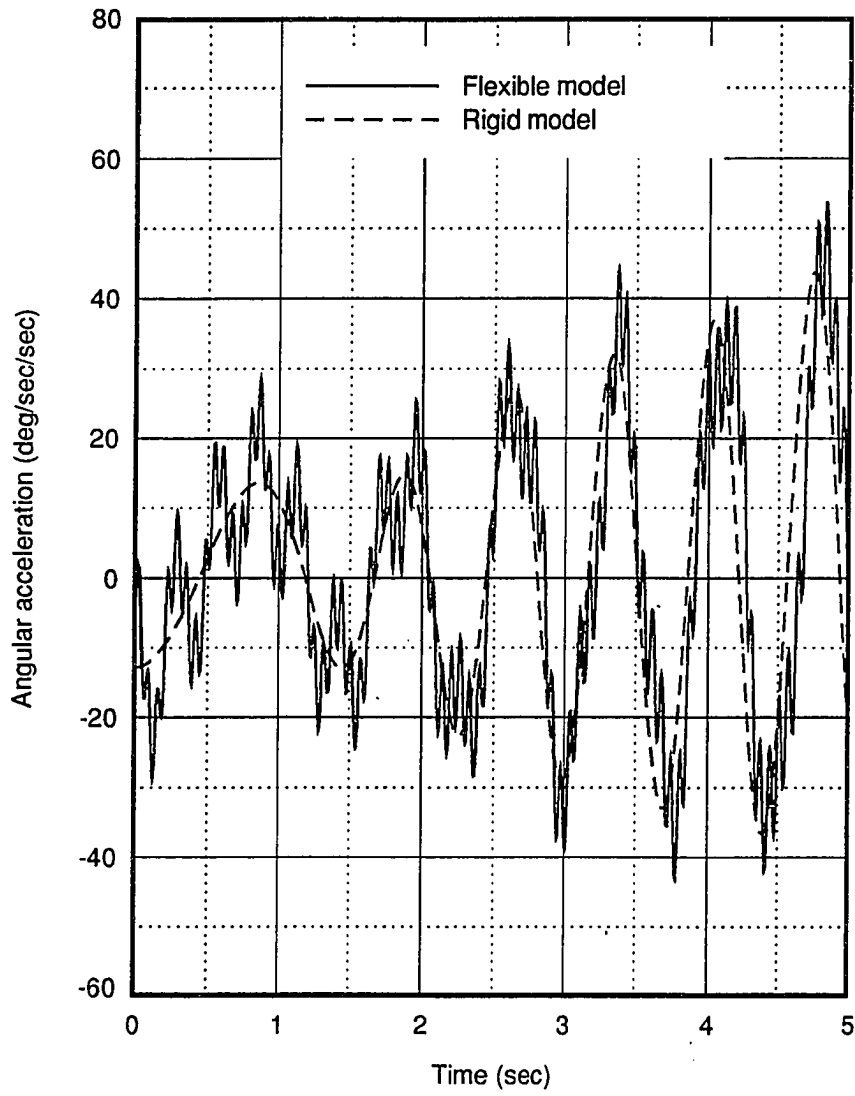


Figure 6.15: Time history of rigid body angular acceleration 1, $\ddot{\lambda}_1$, with collar down and initial tilt: $t_0 = 3$ seconds, $\omega = 60rpm$, $\lambda_1 = 1$ degree

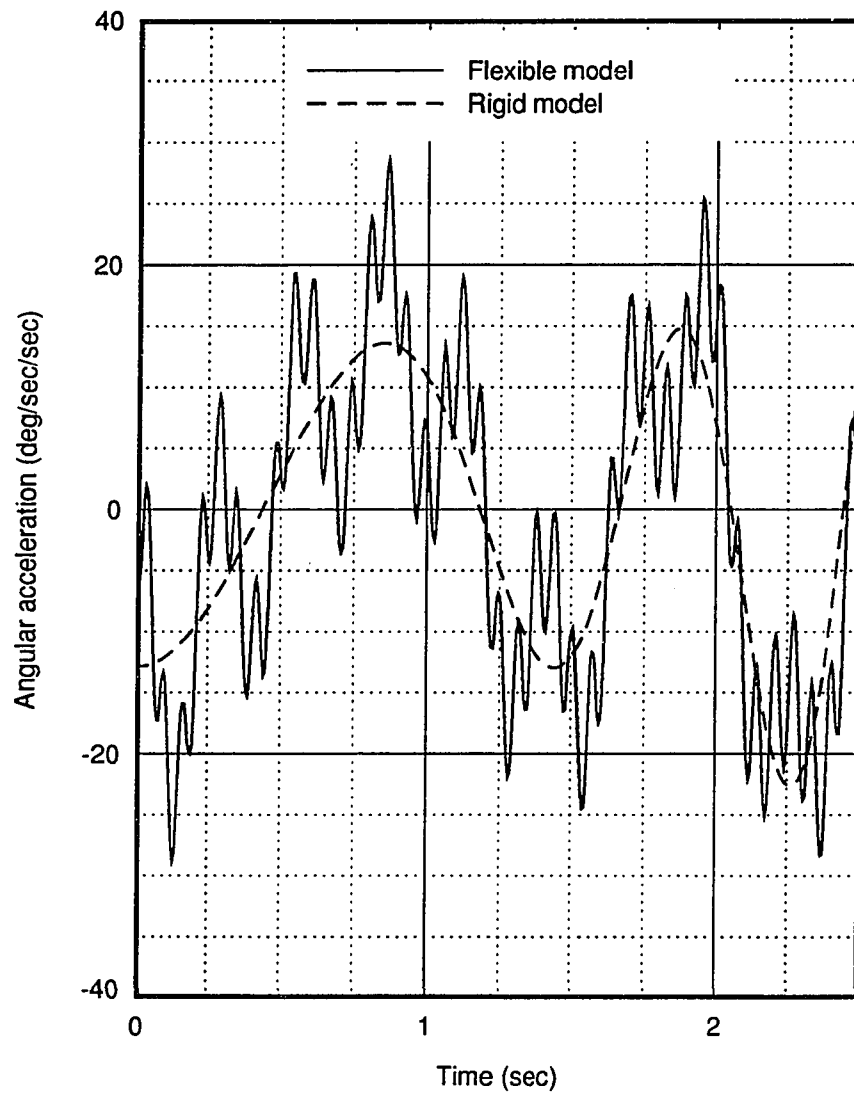


Figure 6.15 (Continued)

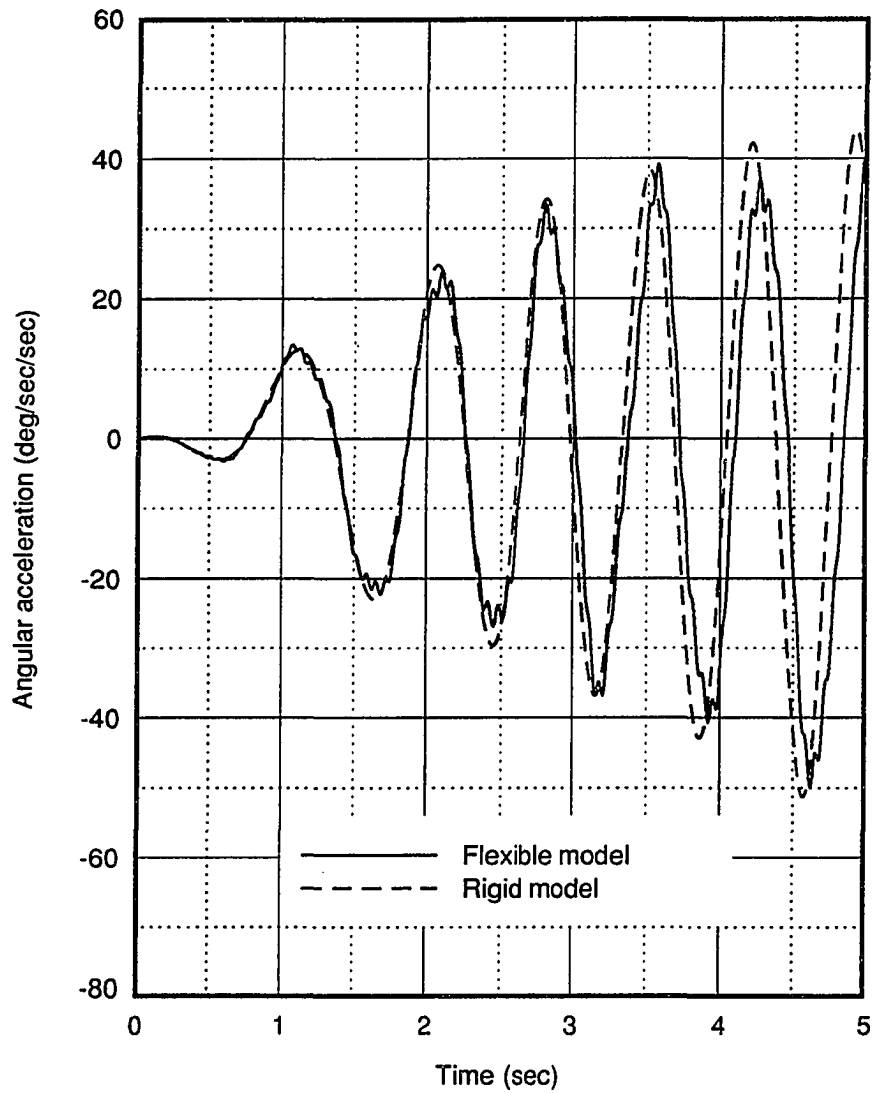


Figure 6.16: Time history of rigid body angular acceleration 2, $\ddot{\lambda}_2$, with collar down and initial tilt: $t_0 = 3$ seconds, $\omega = 60rpm$, $\lambda_1 = 1$ degree

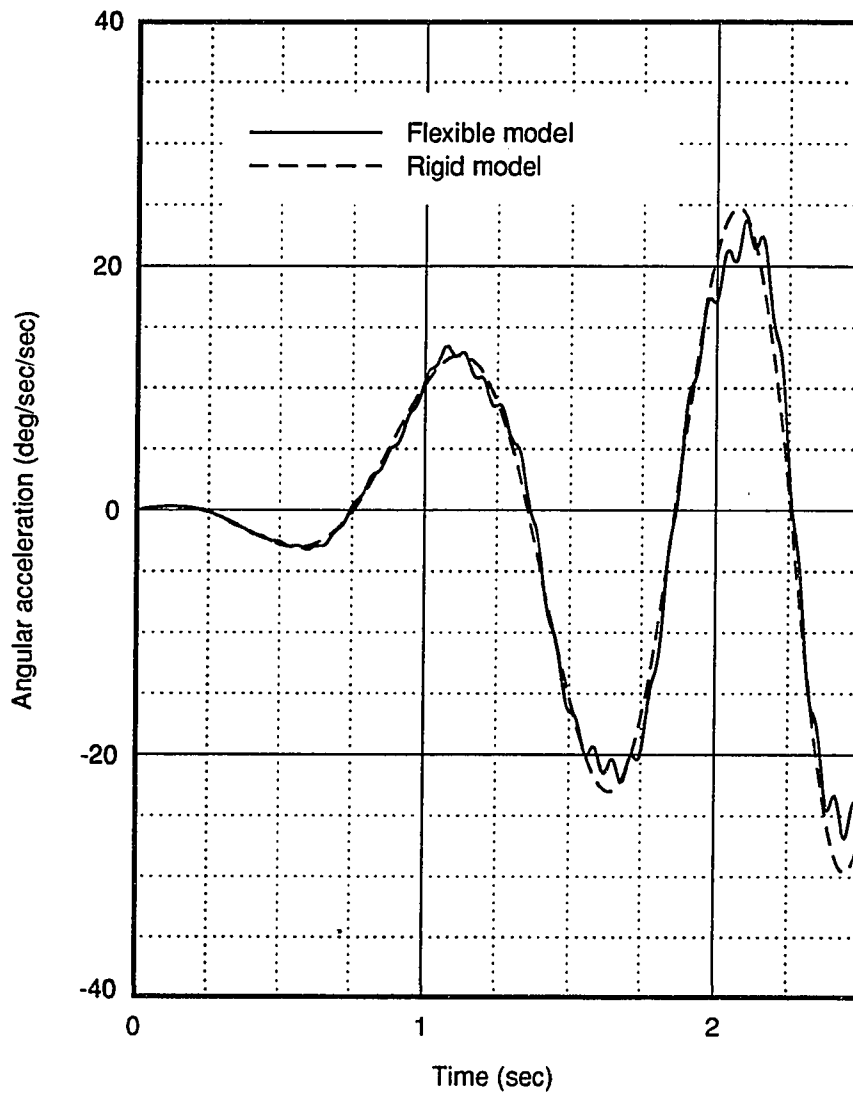


Figure 6.16 (Continued)

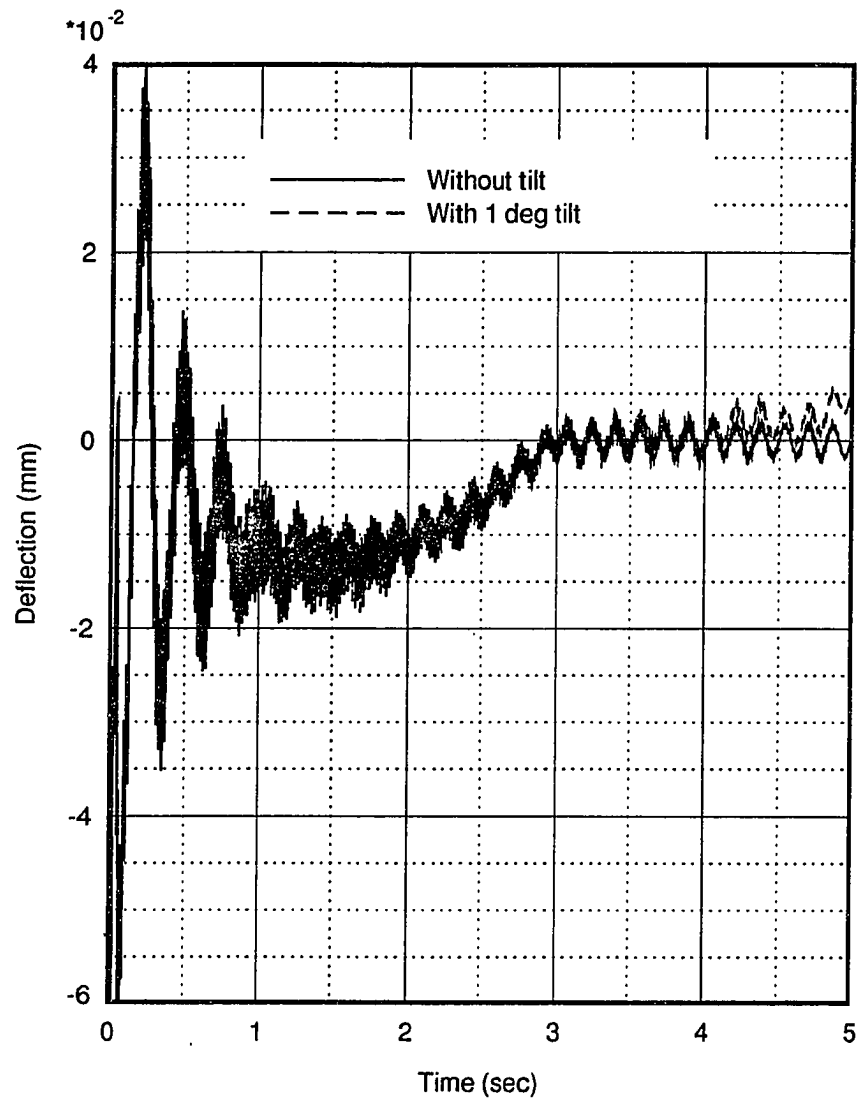


Figure 6.17: Beam 1 tangential deflection without initial tilt or with tilt of $\lambda_1 = 1$ degree: collar down, $t_0 = 3$ seconds, $\omega = 60rpm$

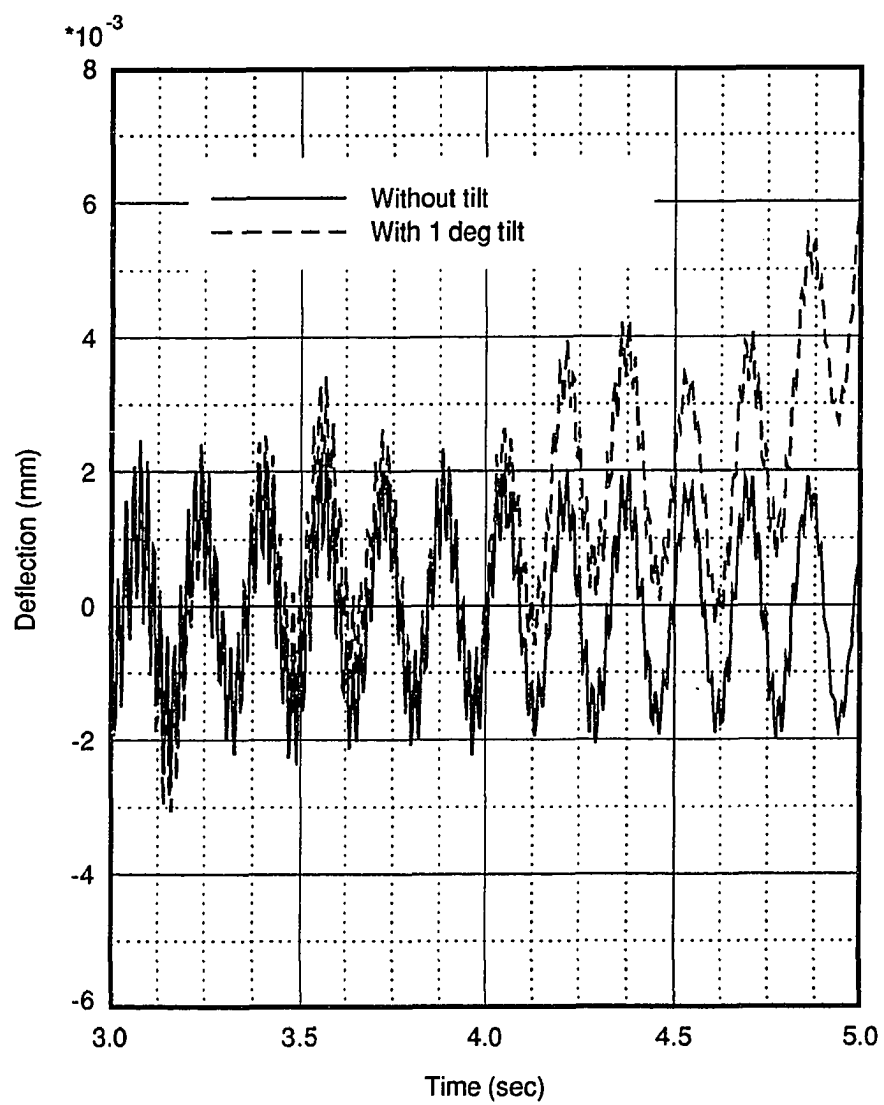


Figure 6.17 (Continued)

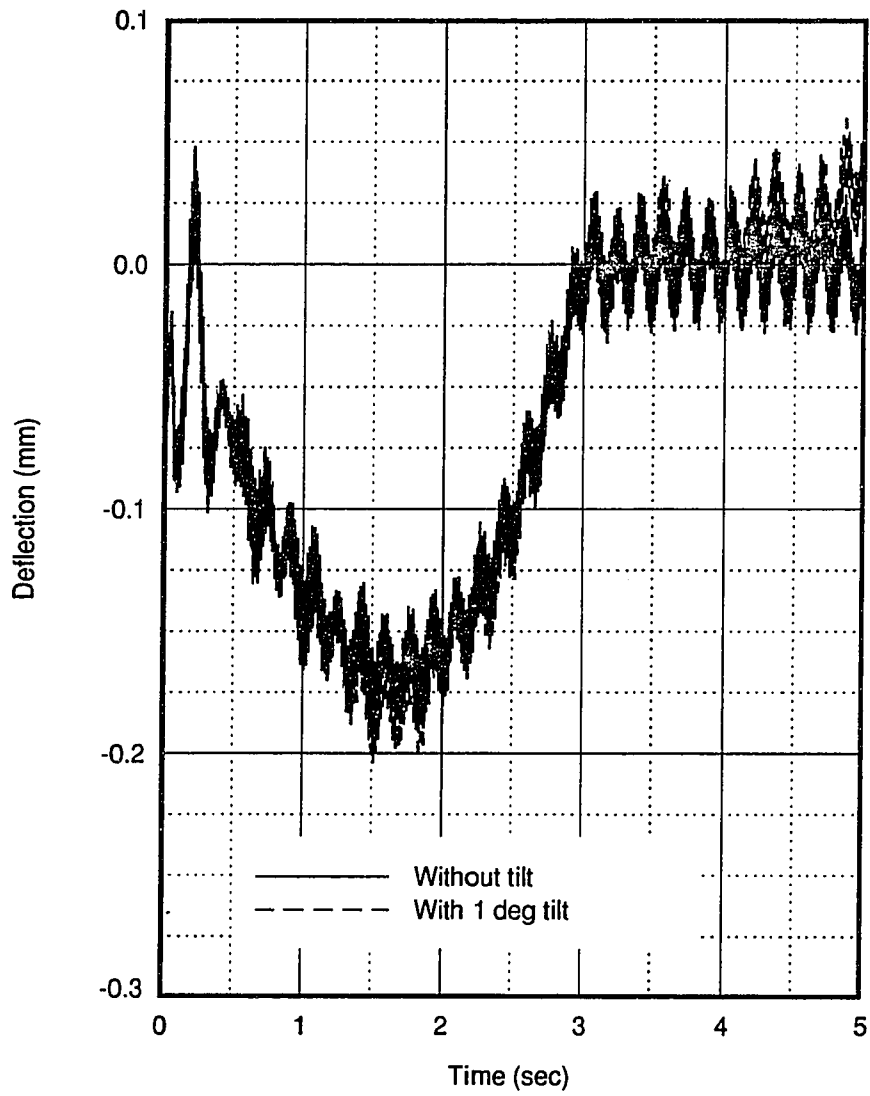


Figure 6.18: Beam 3 tangential deflection without initial tilt or with tilt of $\lambda_1 = 1$ degree: collar down, $t_0 = 3$ seconds, $\omega = 60rpm$

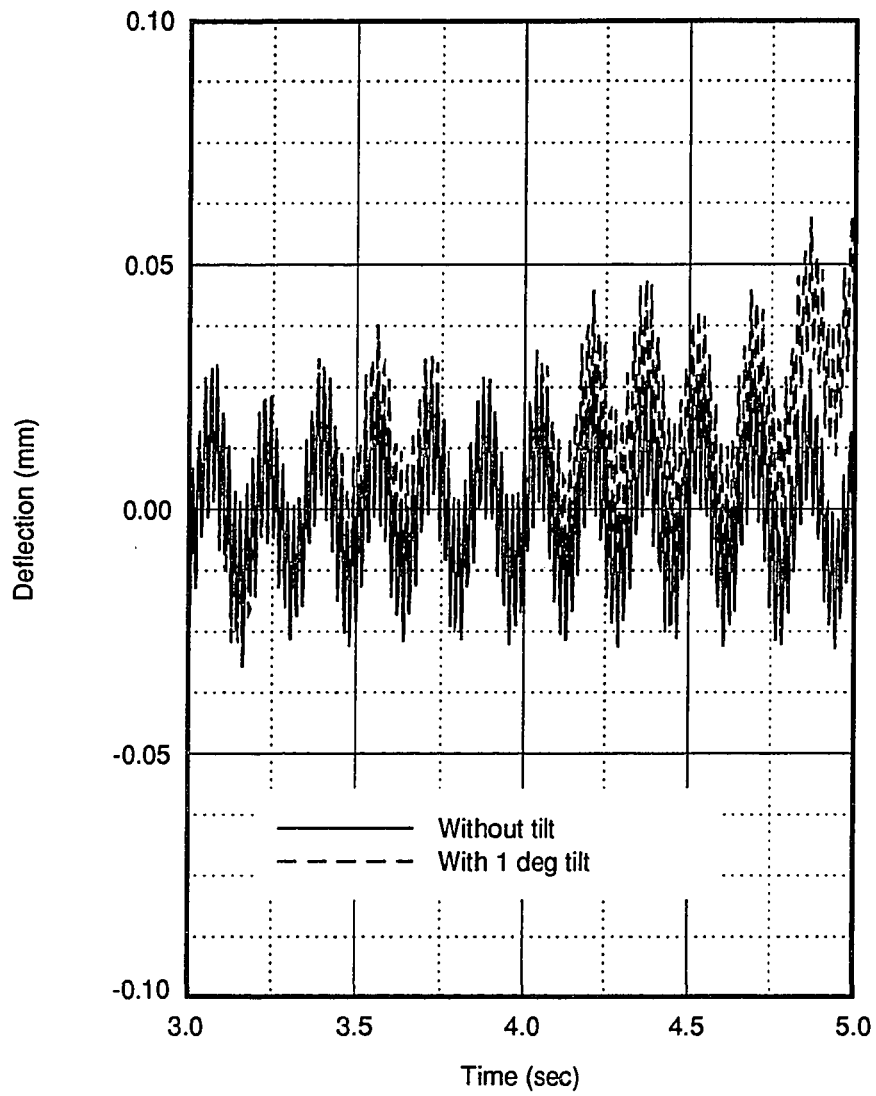


Figure 6.18 (Continued)

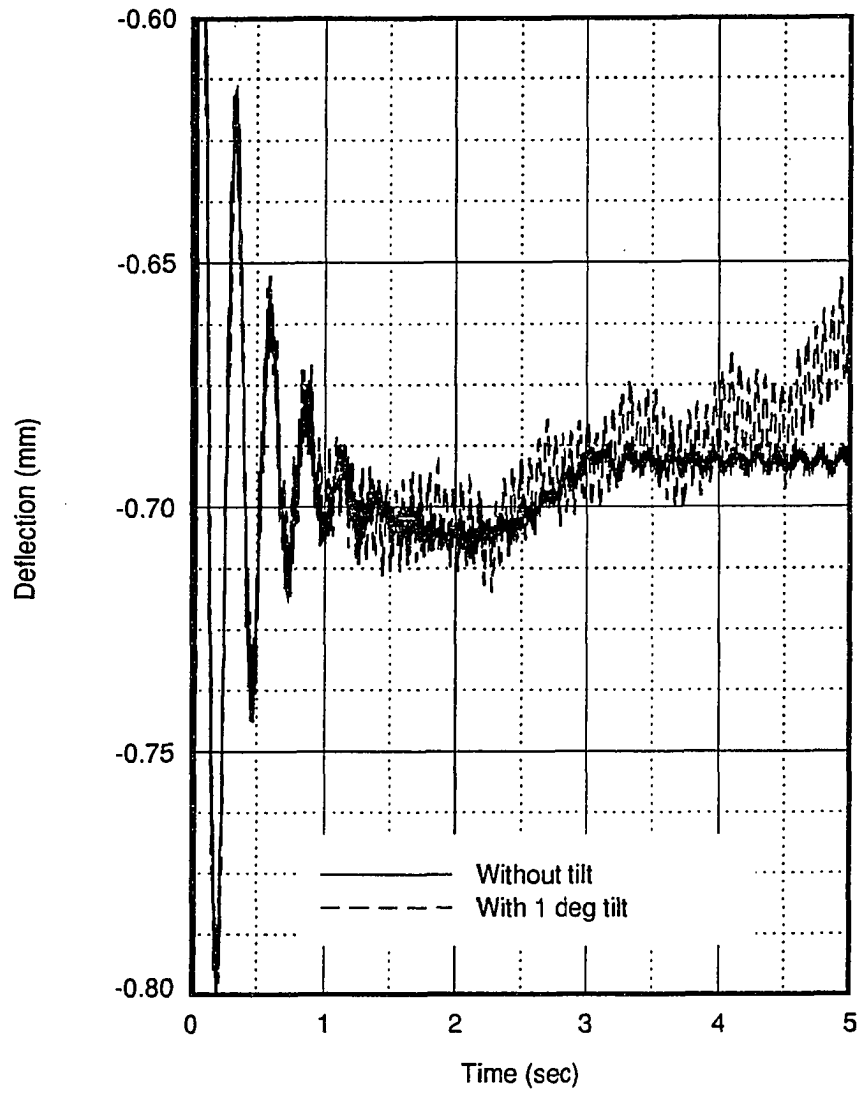


Figure 6.19: Beam 1 vertical deflection without initial tilt or with tilt of $\lambda_1 = 1$ degree: collar down, $t_0 = 3$ seconds, $\omega = 60rpm$

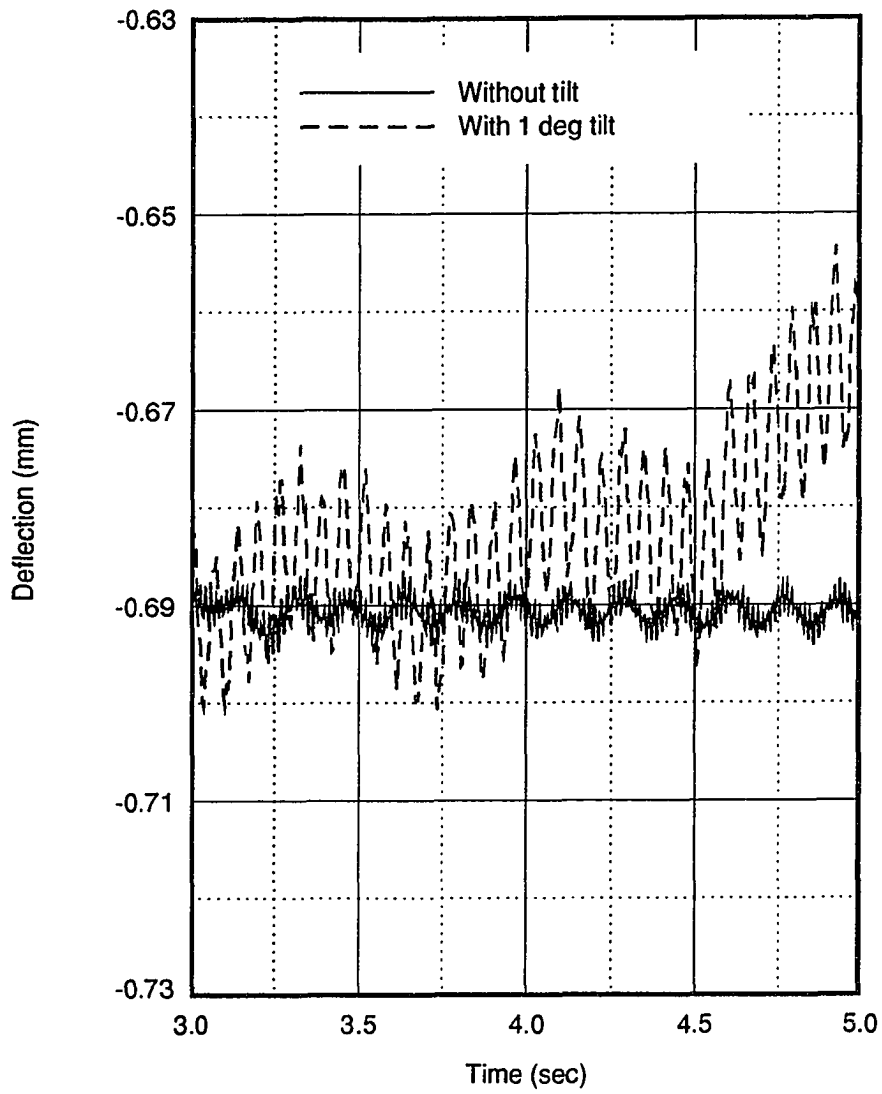


Figure 6.19 (Continued)

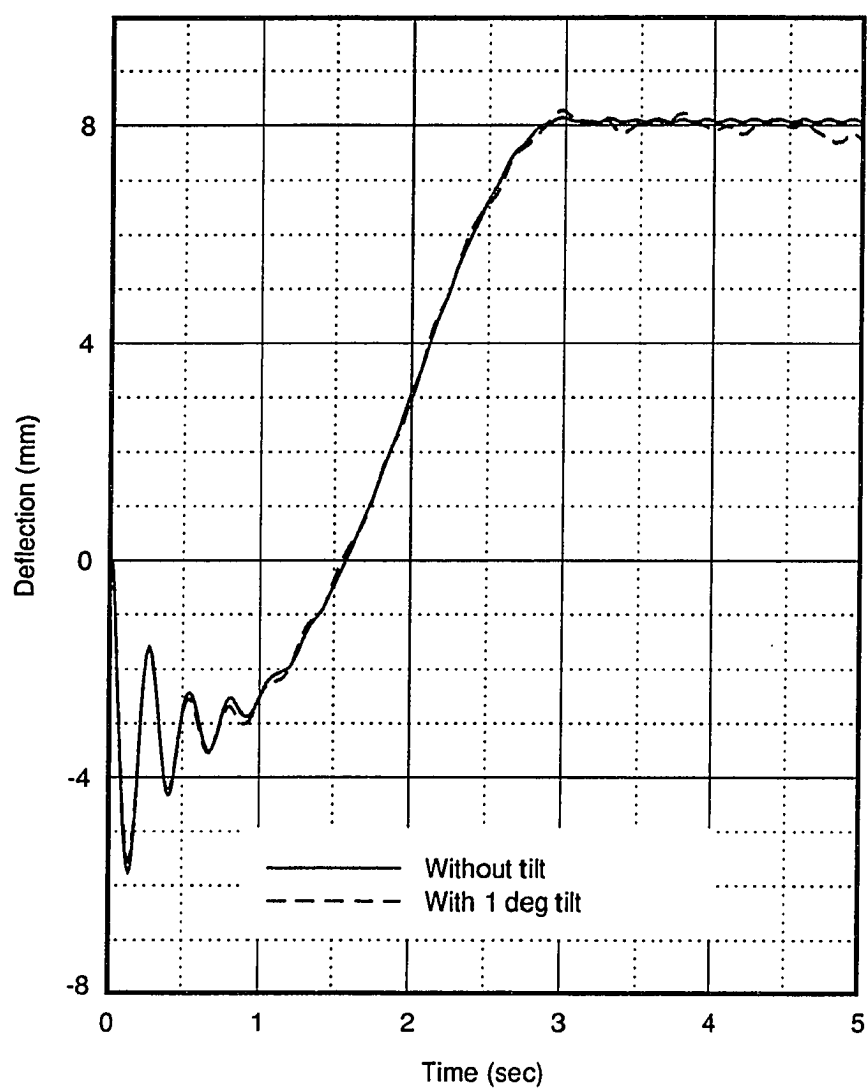


Figure 6.20: Beam 3 radial deflection without initial tilt or with tilt of $\lambda_1 = 1$ degree: collar down, $t_0 = 3$ seconds, $\omega = 60rpm$

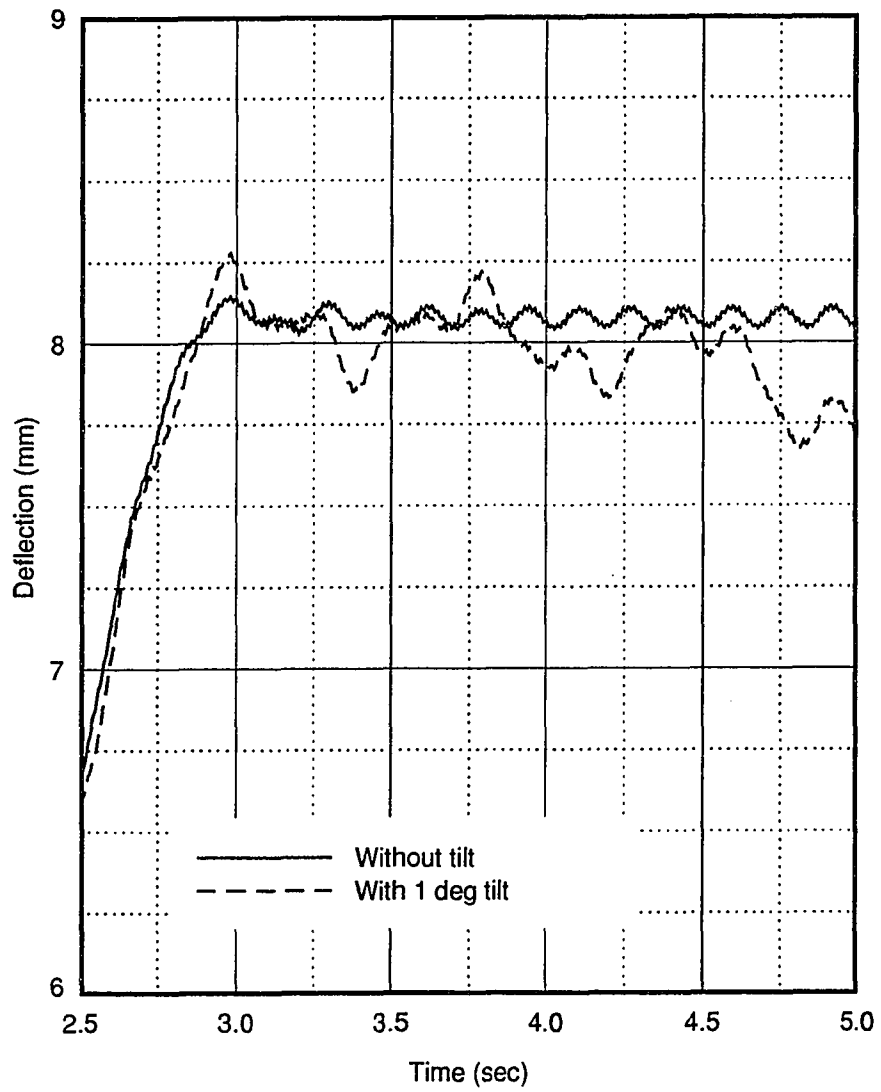


Figure 6.20 (Continued)

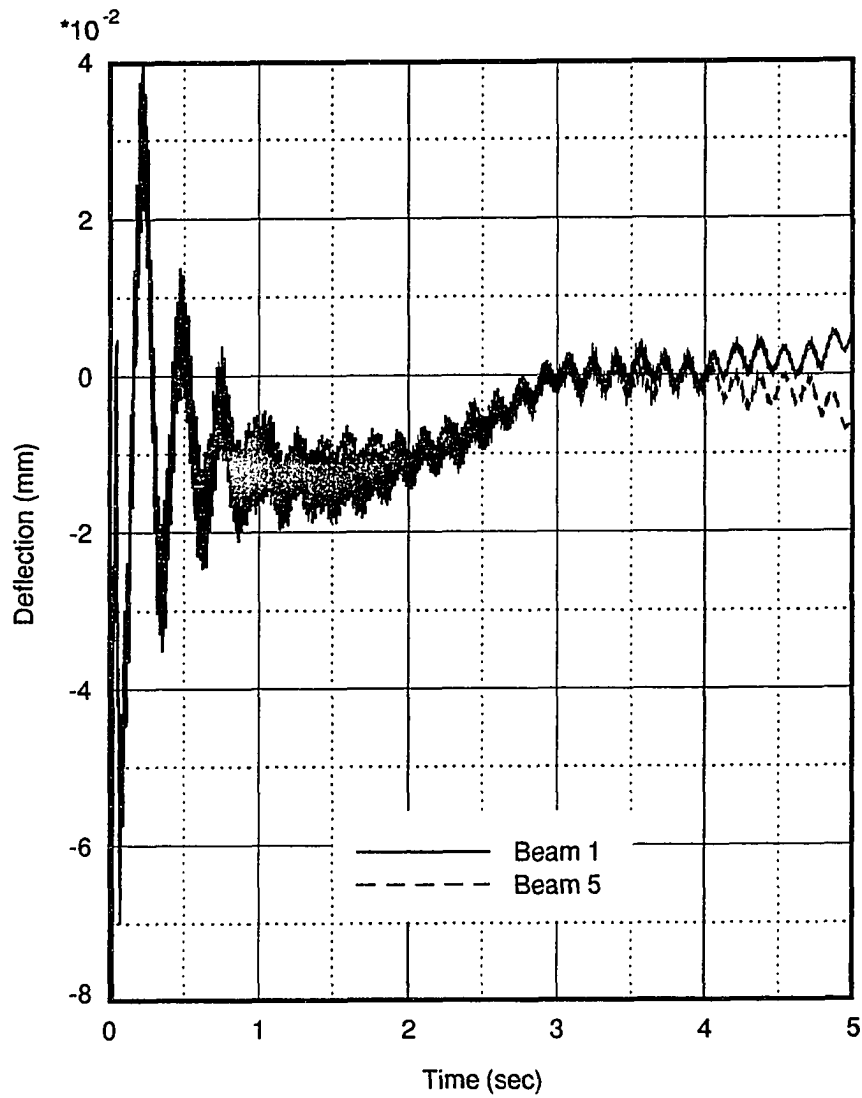


Figure 6.21: Beams 1 and 5 tangential deflections with initial tilt and collar down:
 $t_0 = 3$ seconds, $\omega = 60rpm$, $\lambda_1 = 1$ degree

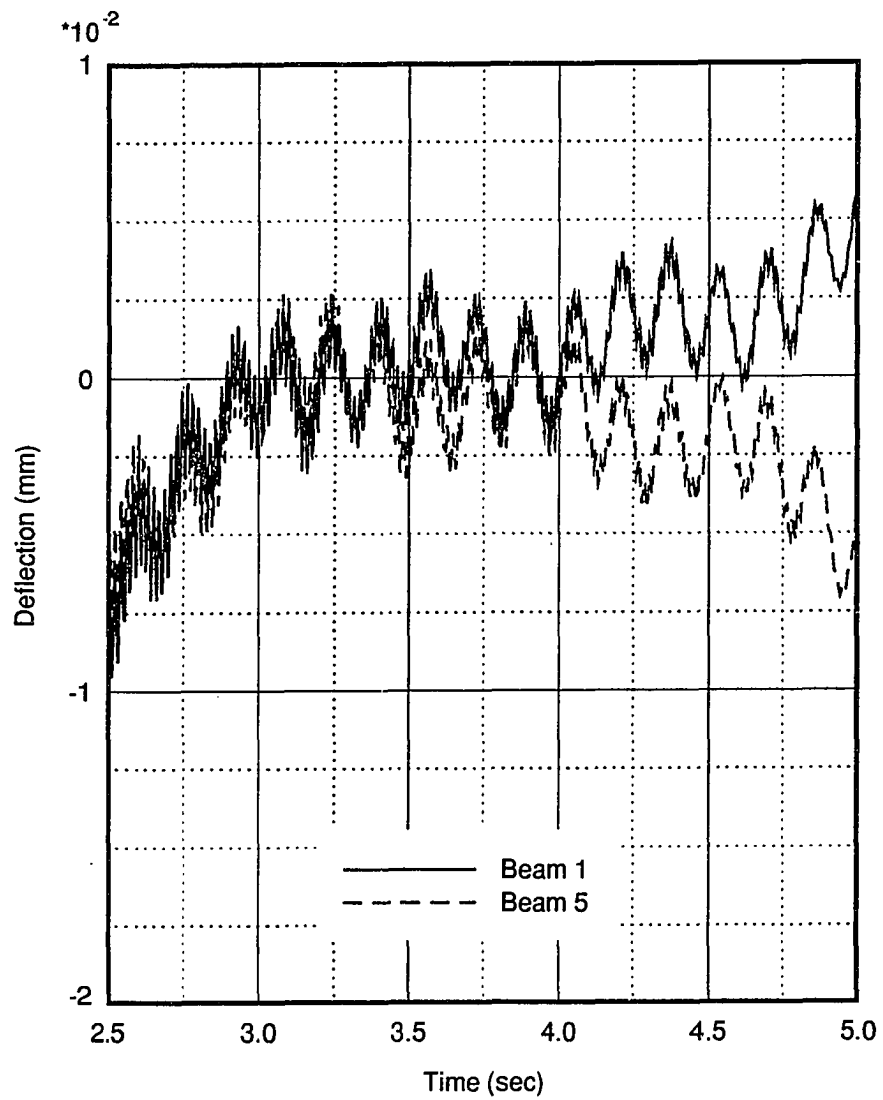


Figure 6.21 (Continued)

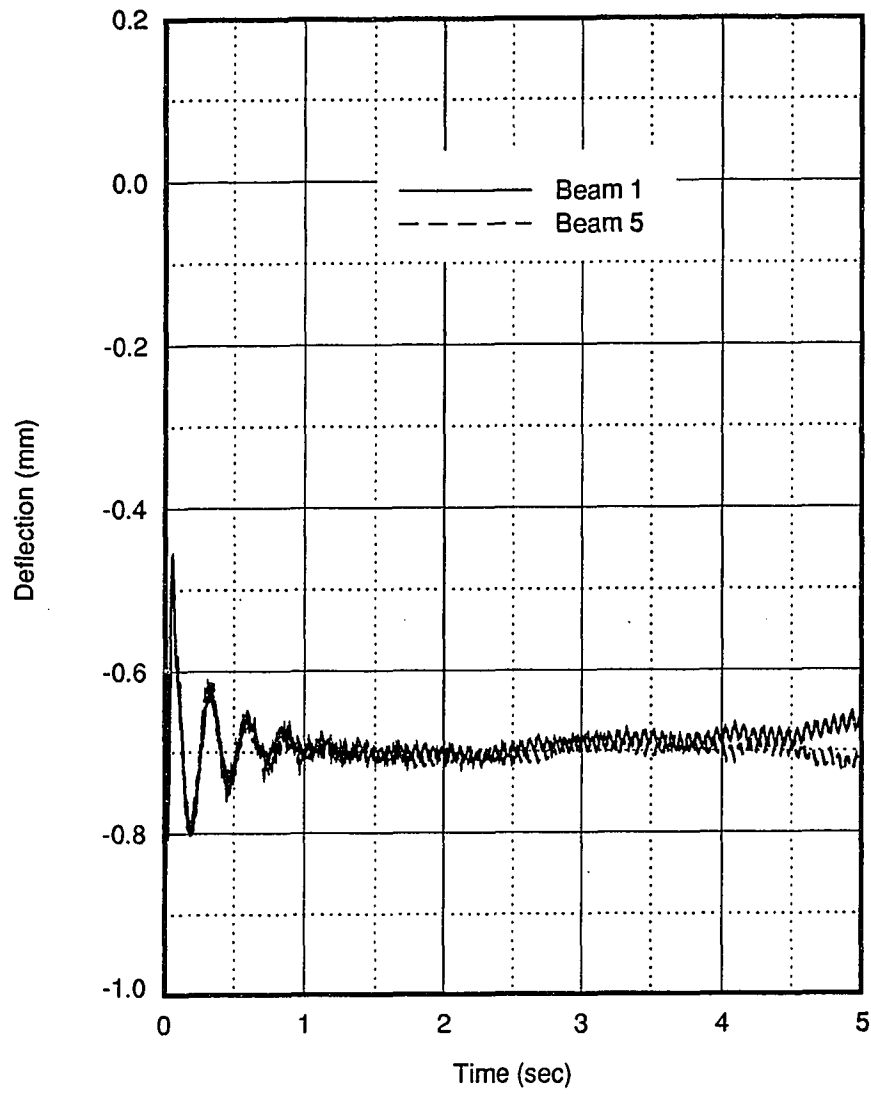


Figure 6.22: Beams 1 and 5 vertical deflections with initial tilt and collar down:
 $t_0 = 3$ seconds, $\omega = 60rpm$, $\lambda_1 = 1$ degree

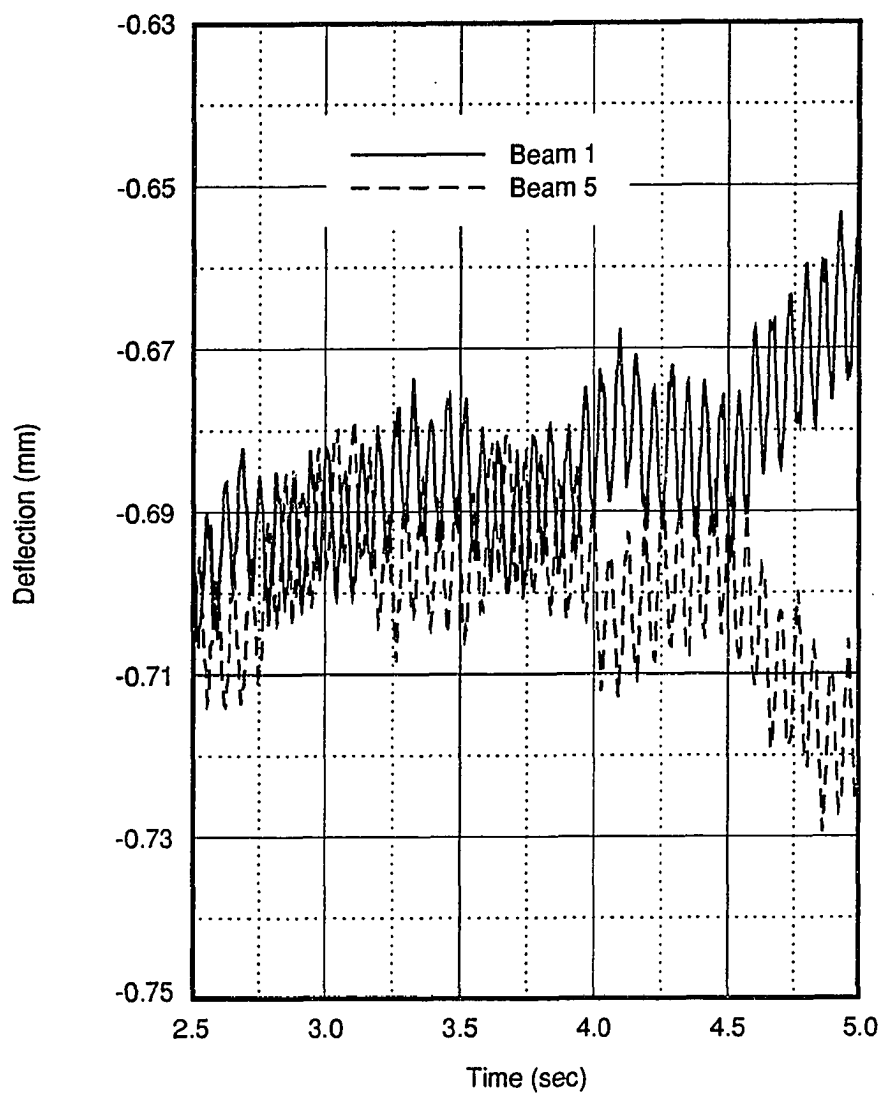


Figure 6.22 (Continued)

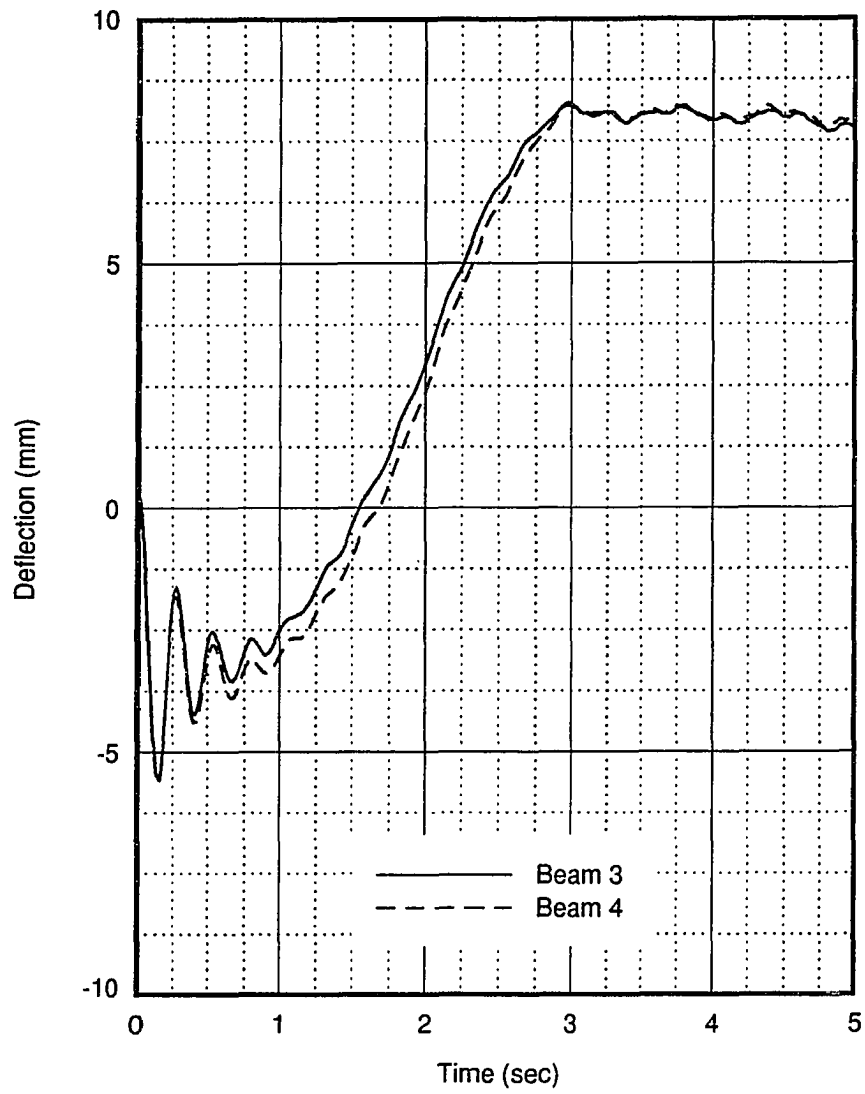


Figure 6.23: Beams 3 and 4 radial deflections with initial tilt and collar down: $t_0 = 3$ seconds, $\omega = 60rpm$, $\lambda_1 = 1$ degree

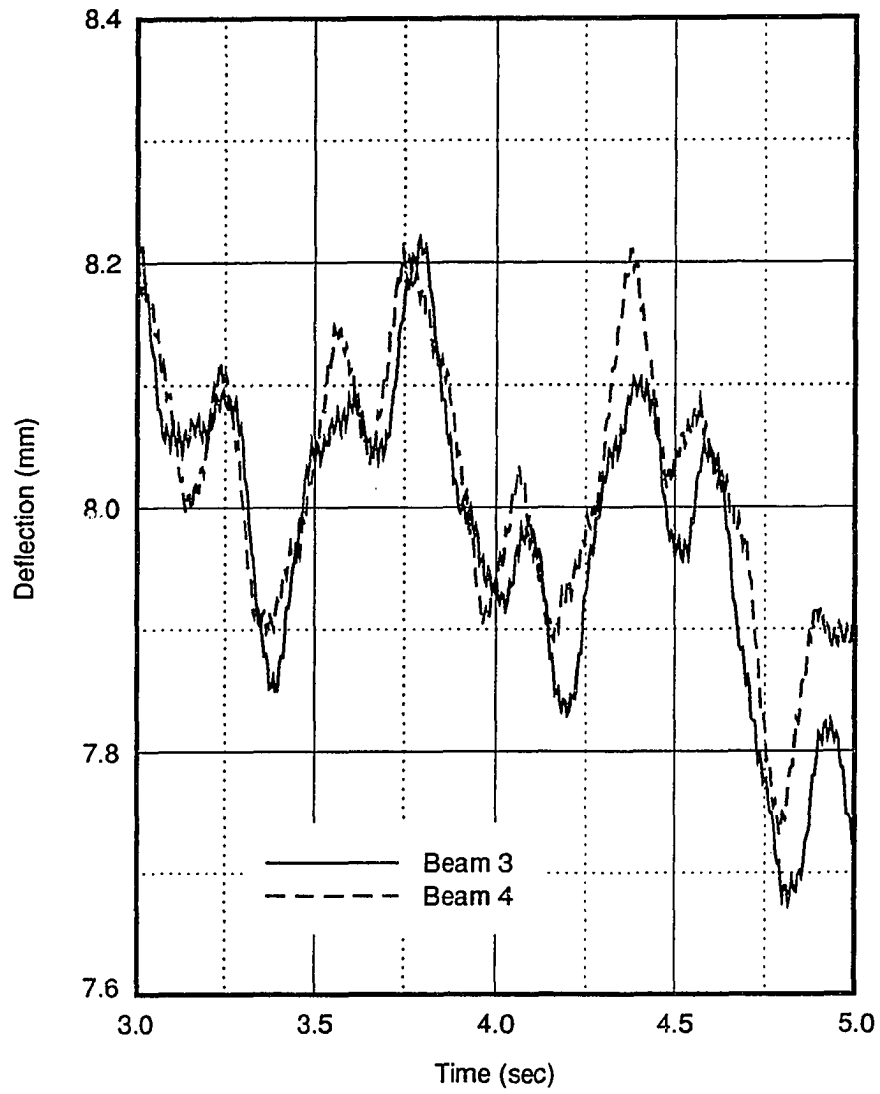


Figure 6.23 (Continued)

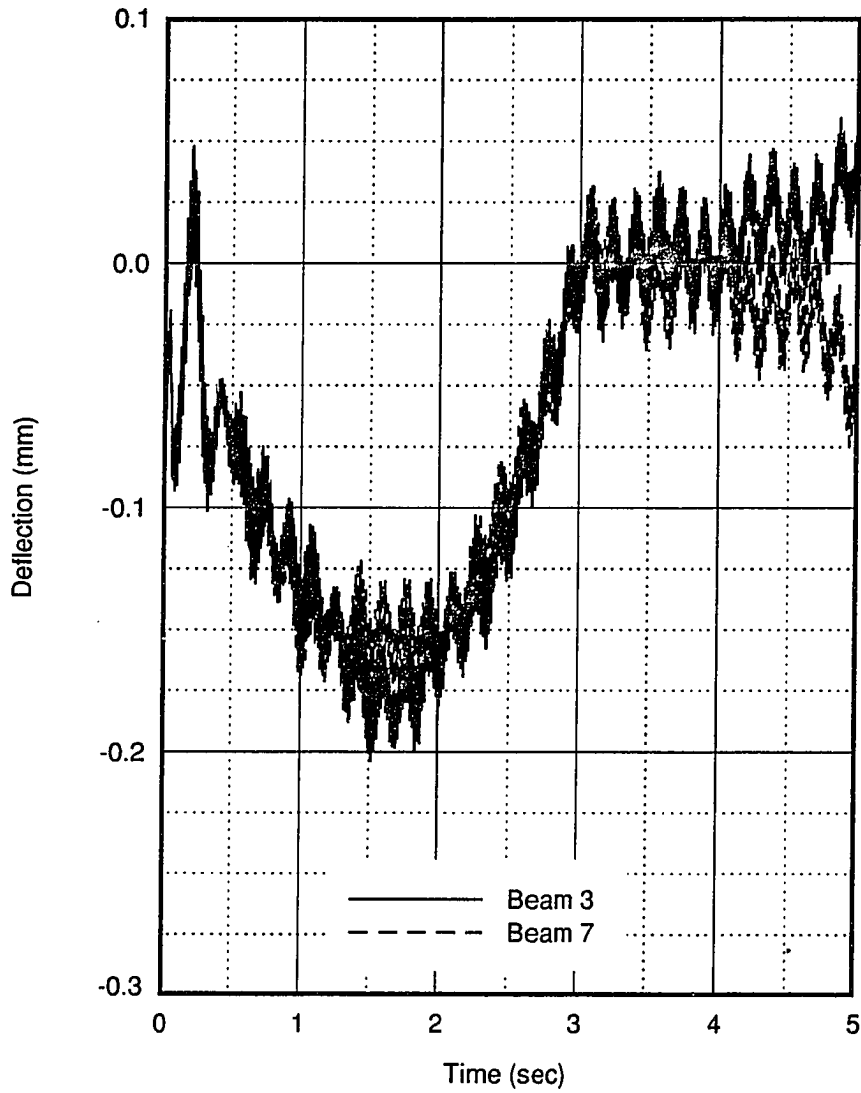


Figure 6.24: Beams 3 and 7 tangential deflections with initial tilt and collar down: $t_0 = 3$ seconds, $\omega = 60rpm$, $\lambda_1 = 1$ degree

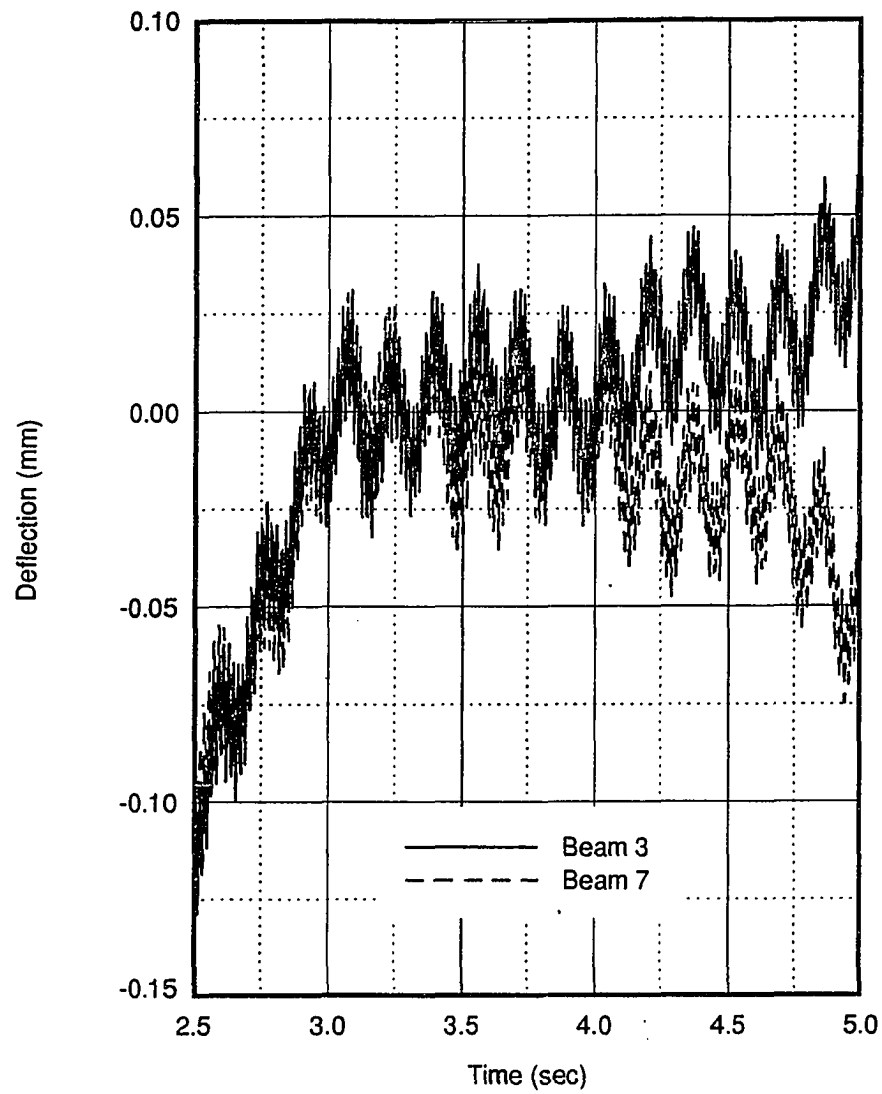


Figure 6.24 (Continued)

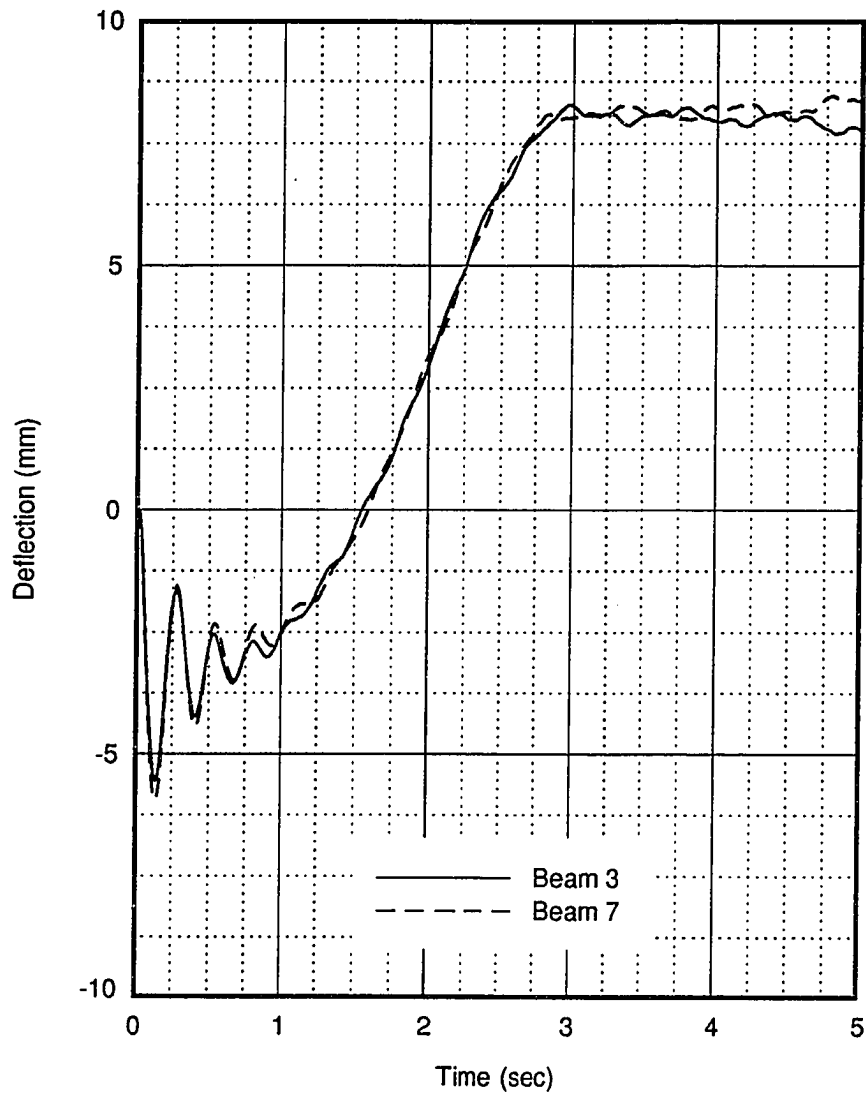


Figure 6.25: Beams 3 and 7 radial deflections with initial tilt and collar down: $t_0 = 3$ seconds, $\omega = 60rpm$, $\lambda_1 = 1$ degree

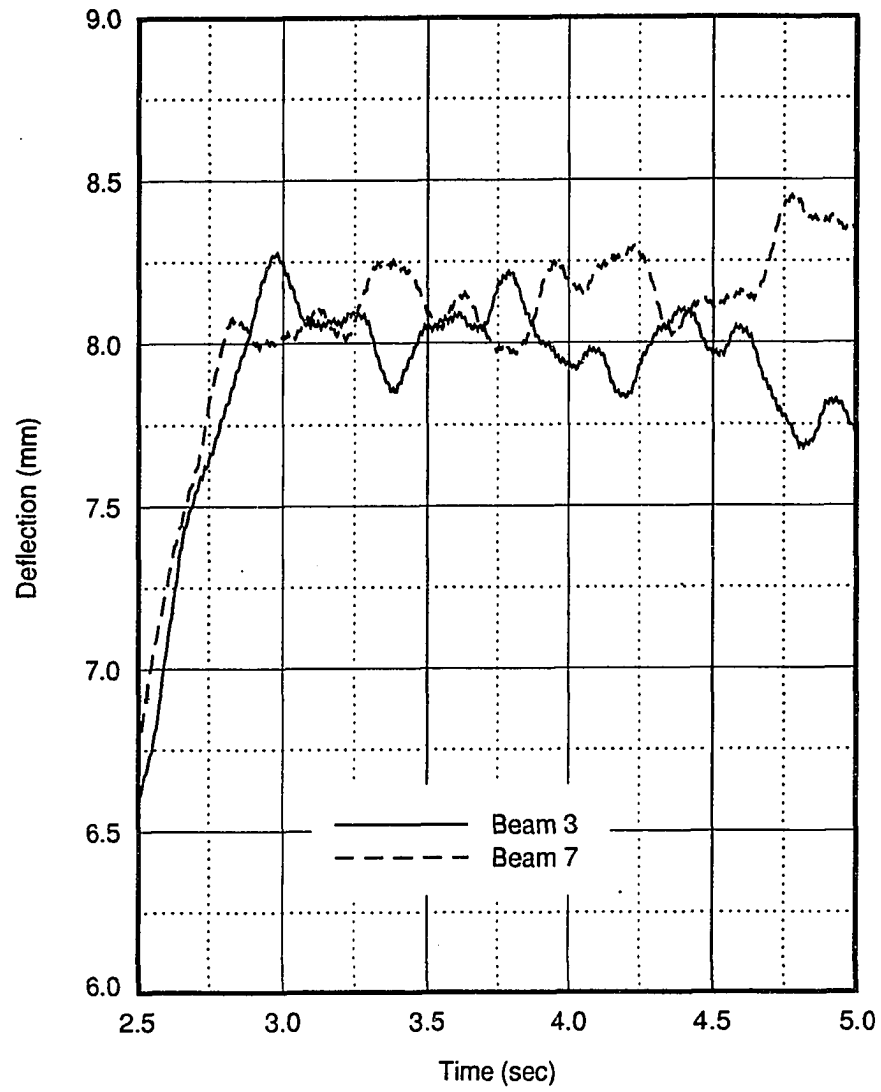


Figure 6.25 (Continued)

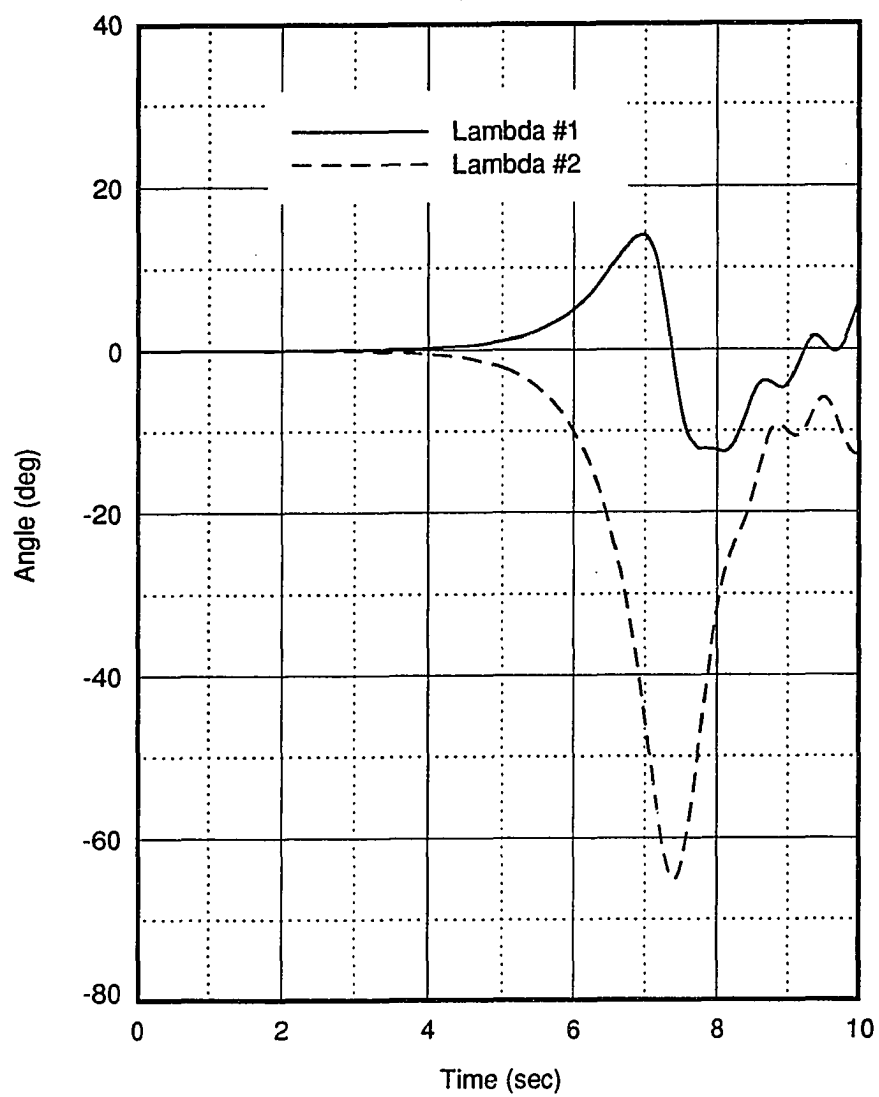


Figure 6.26: Time history of rigid body nutating angles with impulse of 1 Newton and no initial tilt: $t_0 = 1$ second, $\omega = 60rpm$

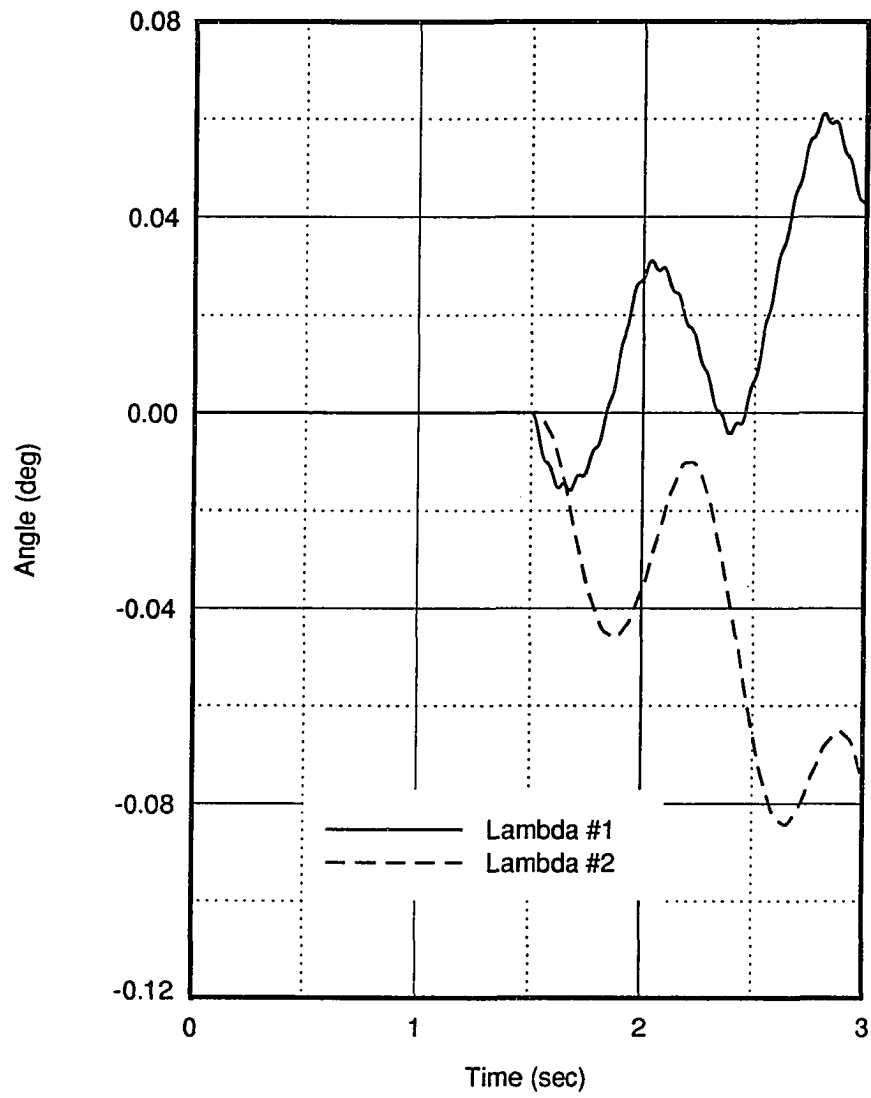


Figure 6.26 (Continued)

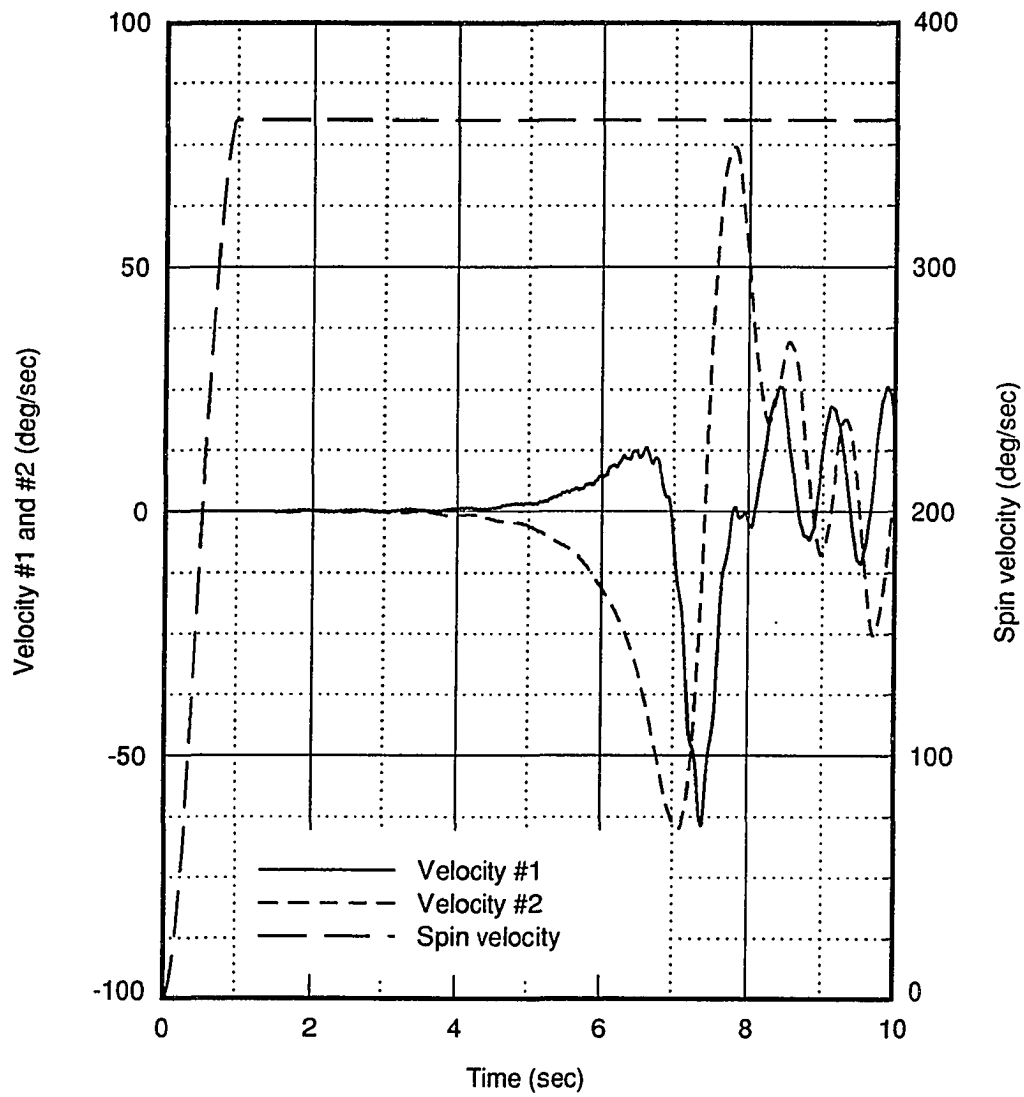


Figure 6.27: Time history of rigid body angular velocities with impulse of 1 Newton and no initial tilt: $t_0 = 1$ second, $\omega = 60rpm$

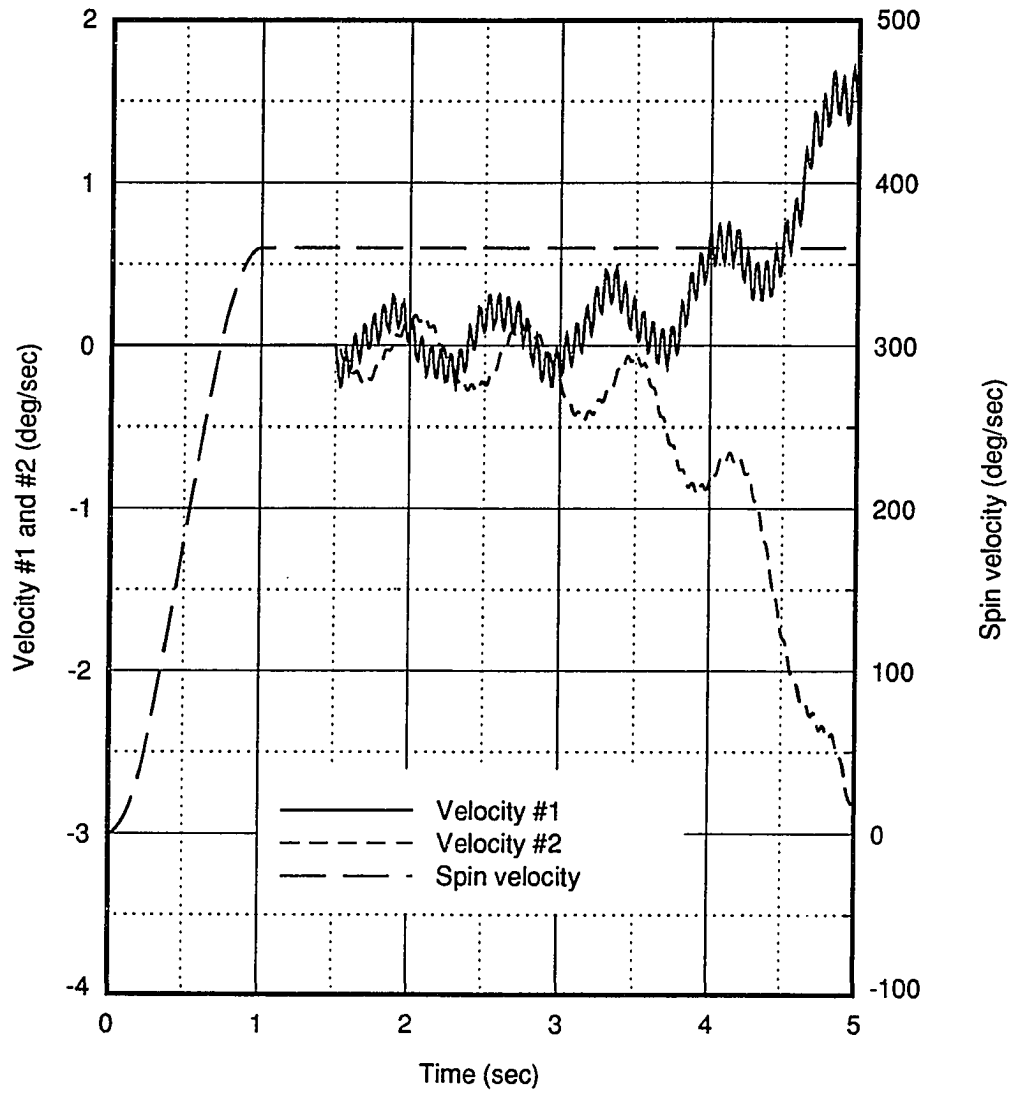


Figure 6.27 (Continued)

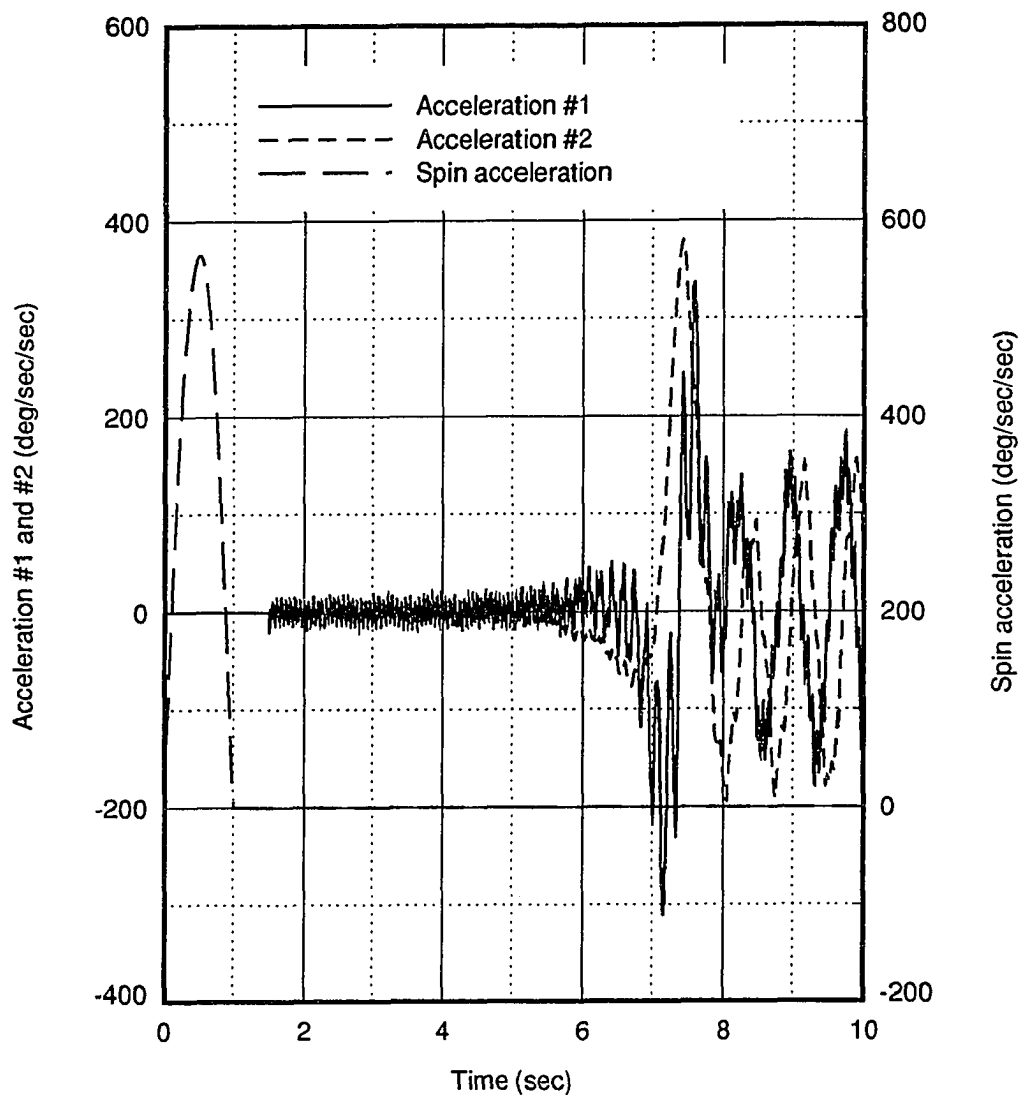


Figure 6.28: Time history of rigid body angular accelerations with impulse of 1 Newton and no initial tilt: $t_0 = 1$ second, $\omega = 60rpm$

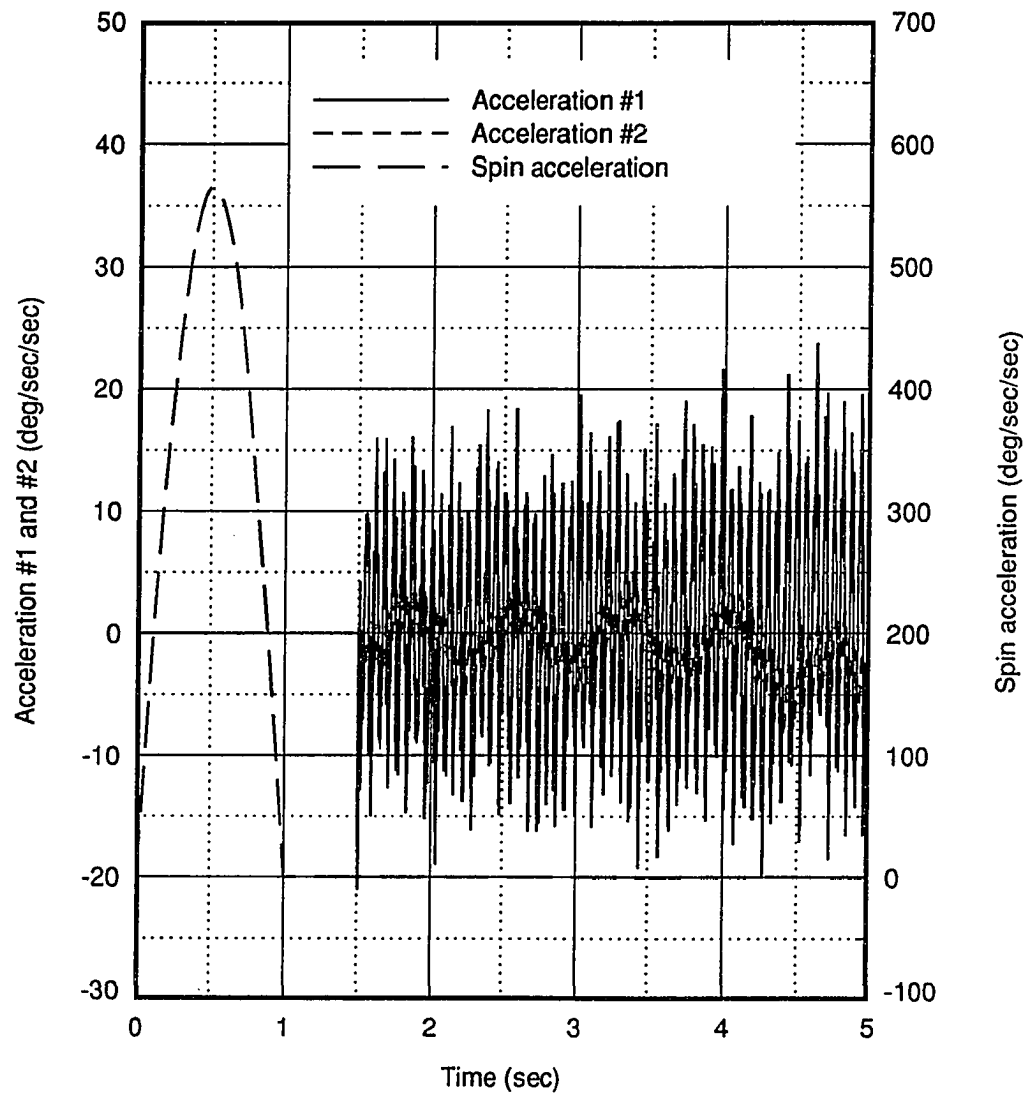


Figure 6.28 (Continued)

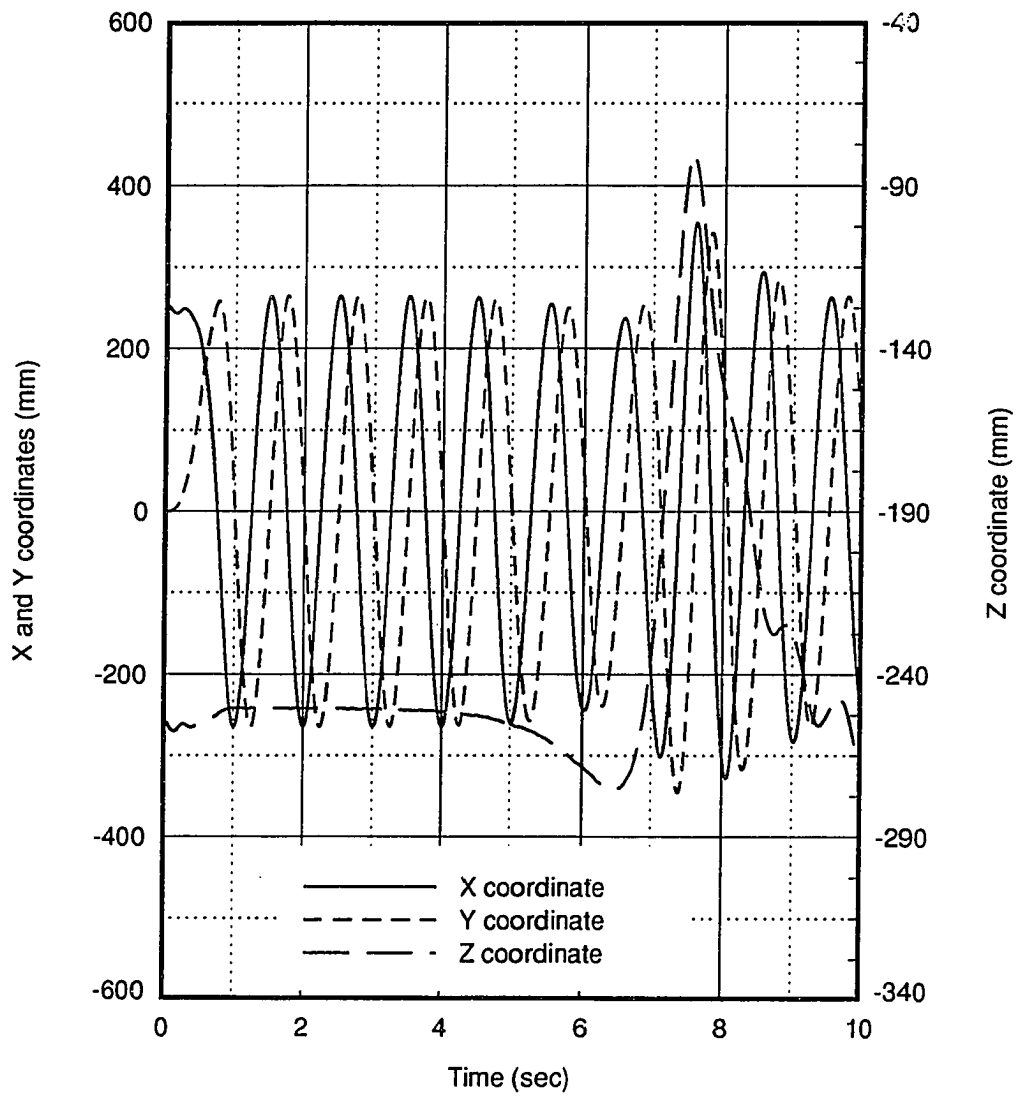


Figure 6.29: Trajectory of tank 1 center position in an inertial frame with impulse of 1 Newton and no initial tilt: $t_0 = 1$ second, $\omega = 60rpm$

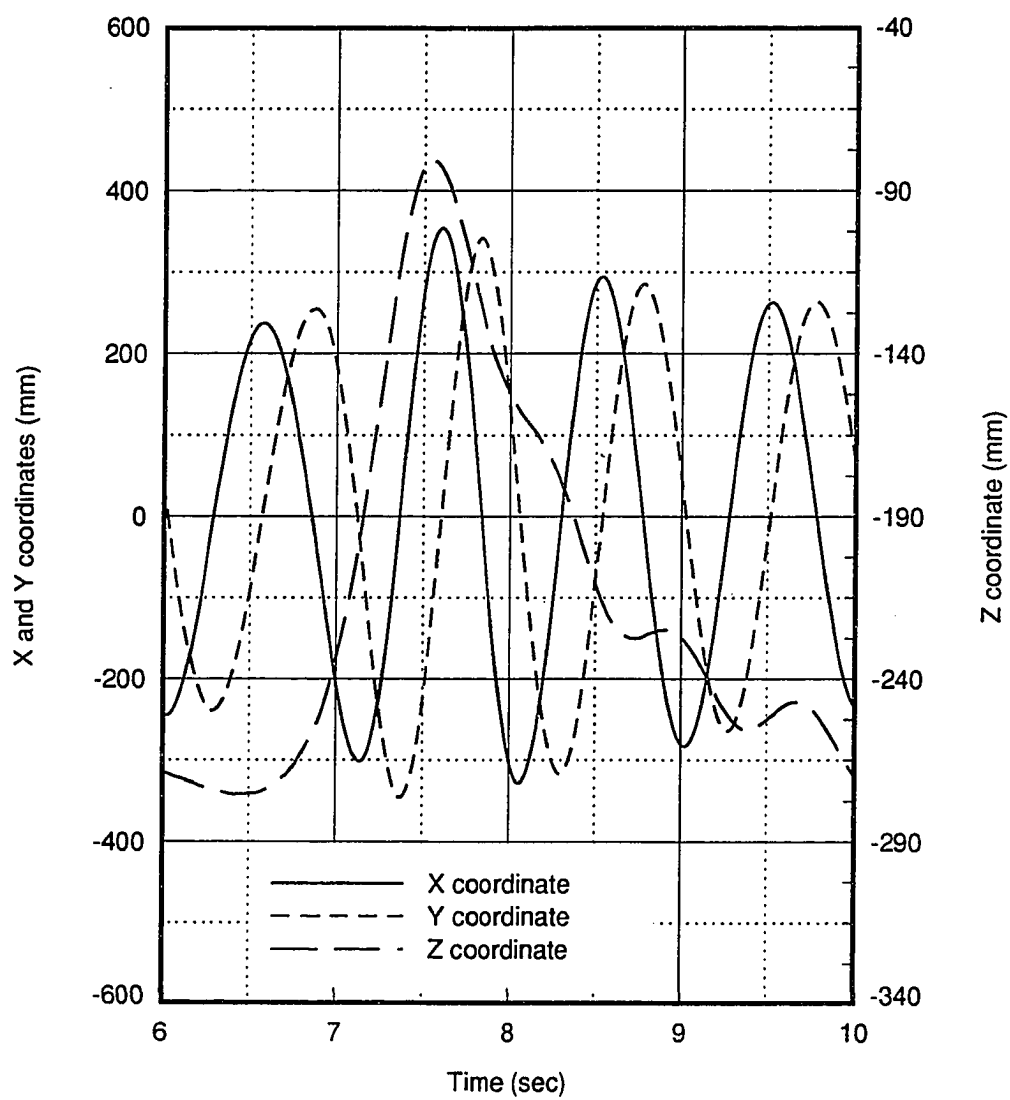


Figure 6.29 (Continued)

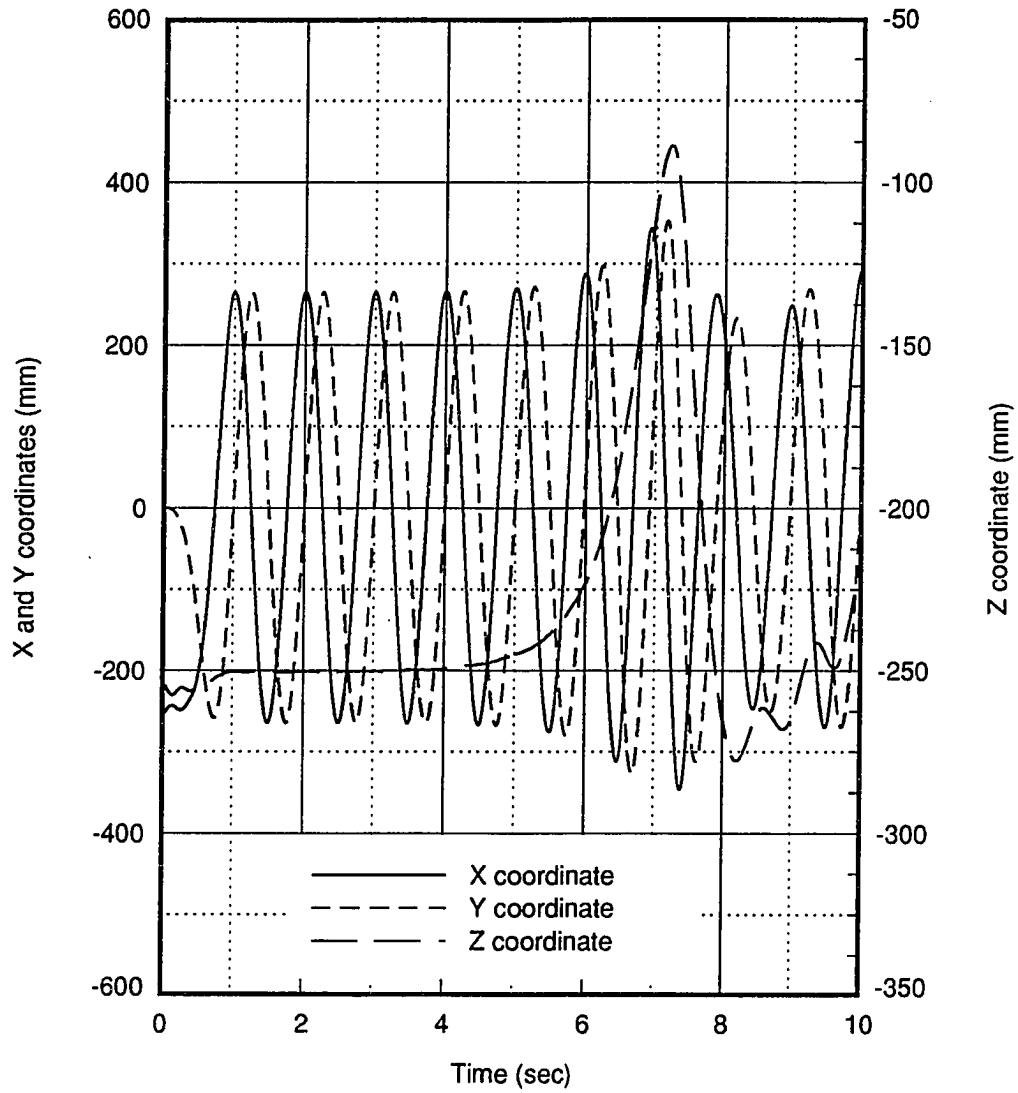


Figure 6.30: Trajectory of tank 2 center position in an inertial frame with impulse of 1 Newton and no initial tilt: $t_0 = 1$ second, $\omega = 60rpm$

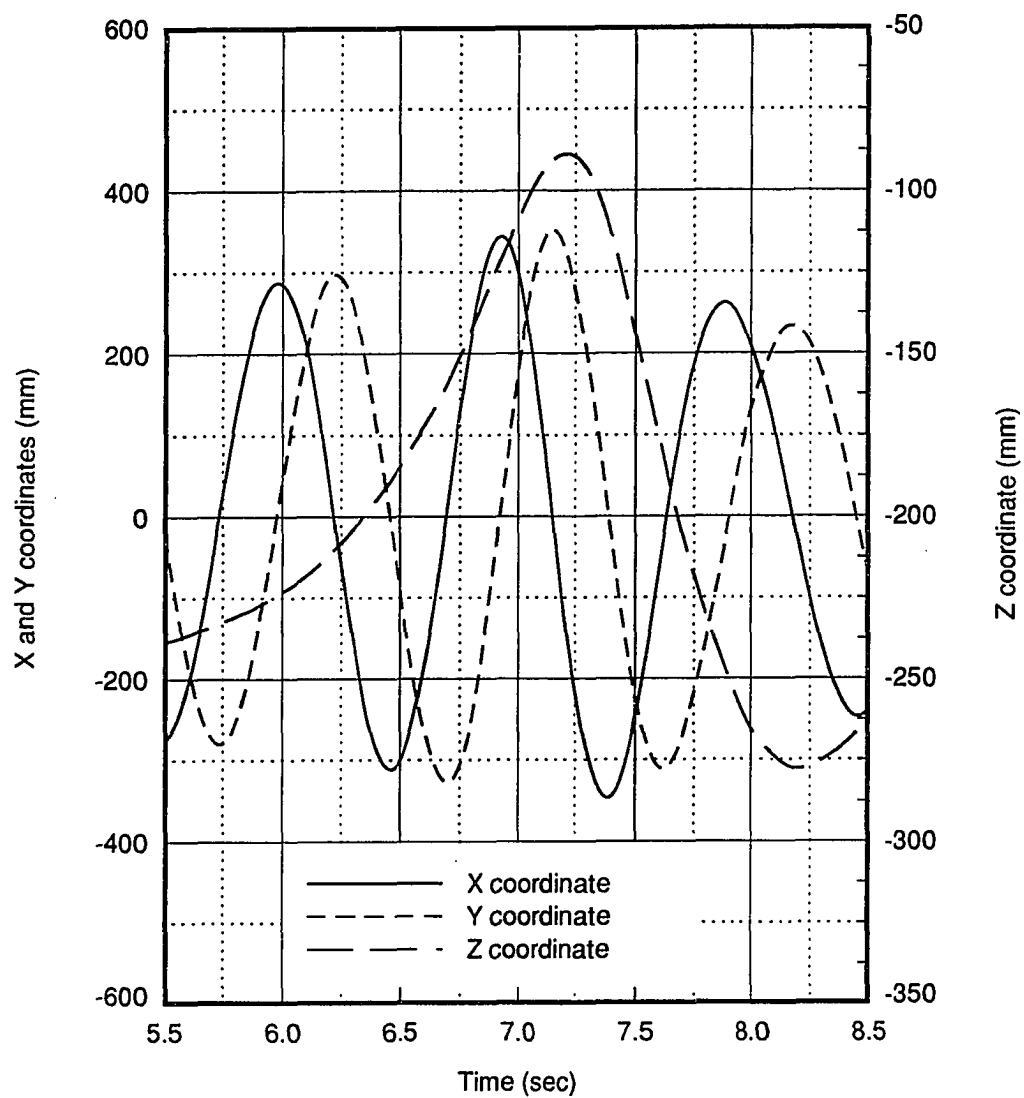


Figure 6.30 (Continued)

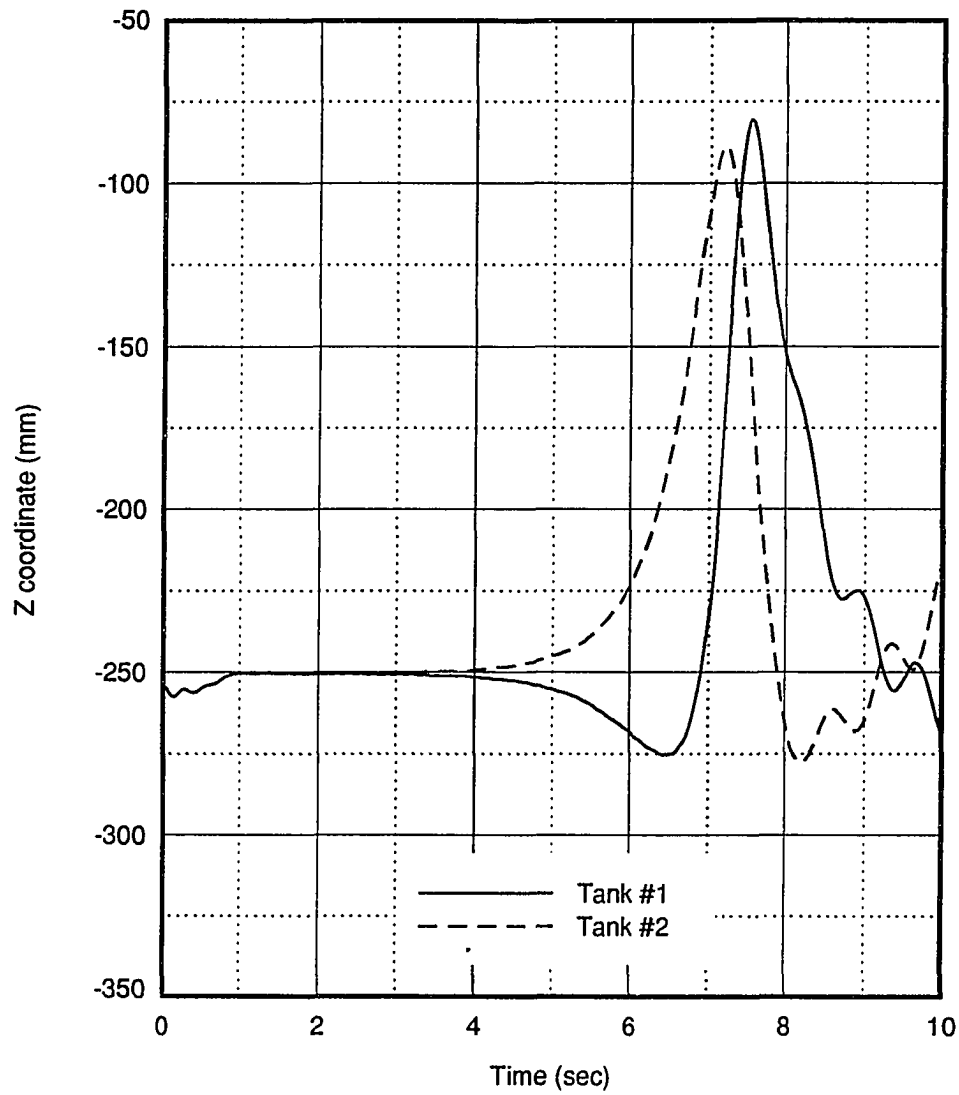


Figure 6.31: Z coordinates of tanks 1 and 2 center positions in an inertial frame with impulse of 1 Newton and no initial tilt: $t_0 = 1$ second, $\omega = 60rpm$

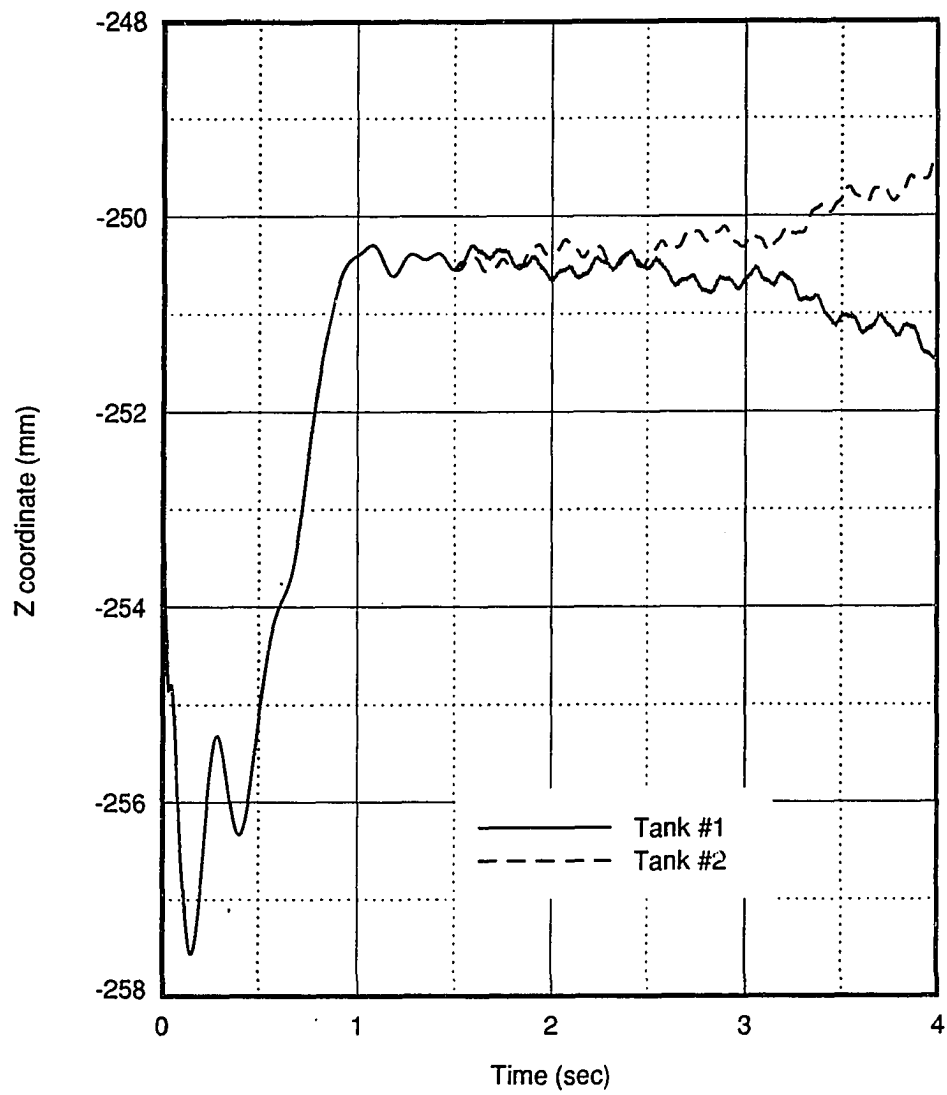


Figure 6.31 (Continued)

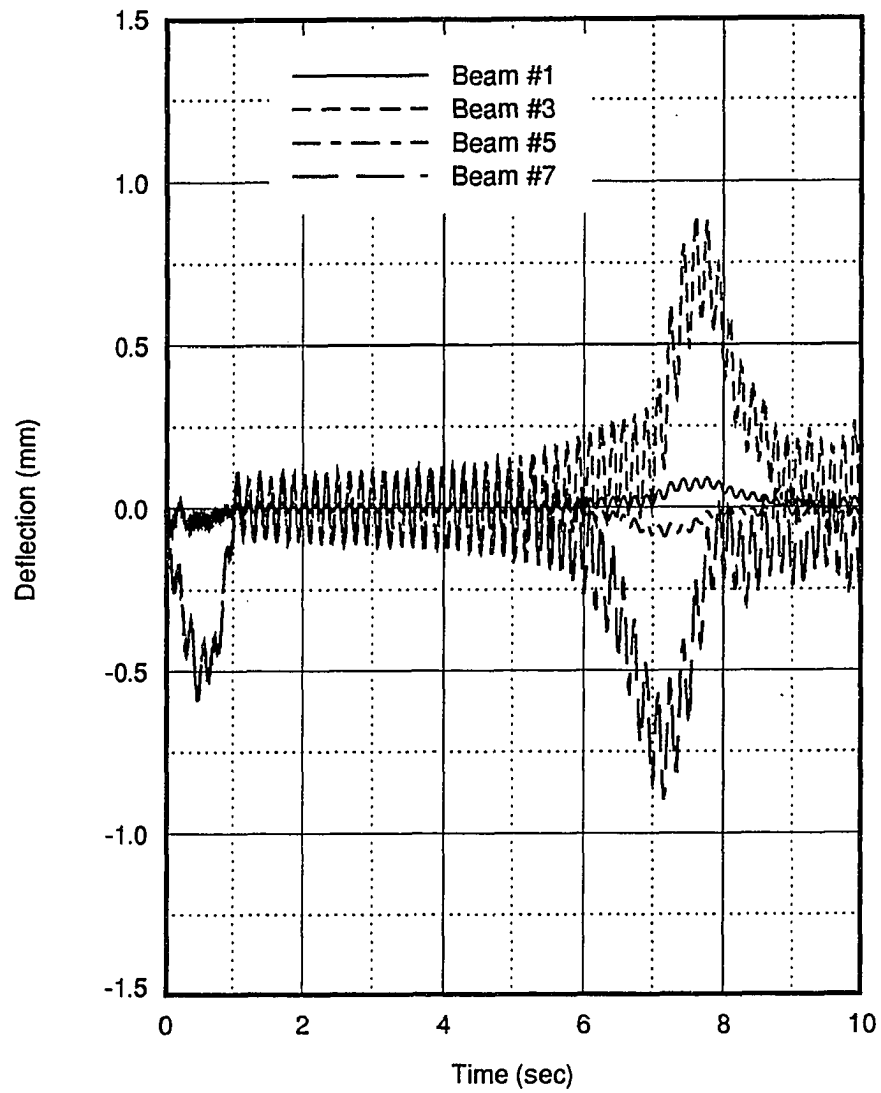


Figure 6.32: Beam local tangential deflections at distal ends with impulse of 1 Newton and no initial tilt: $t_0 = 1$ second, $\omega = 60rpm$

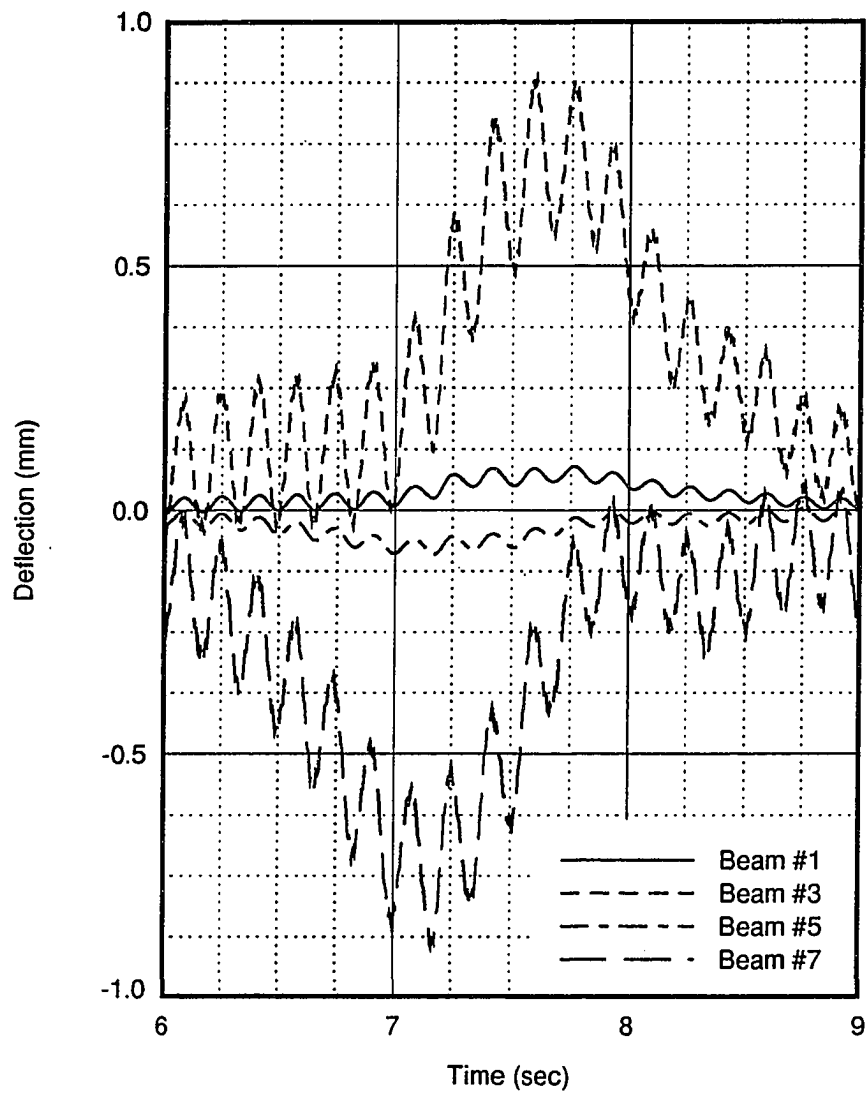


Figure 6.32 (Continued)

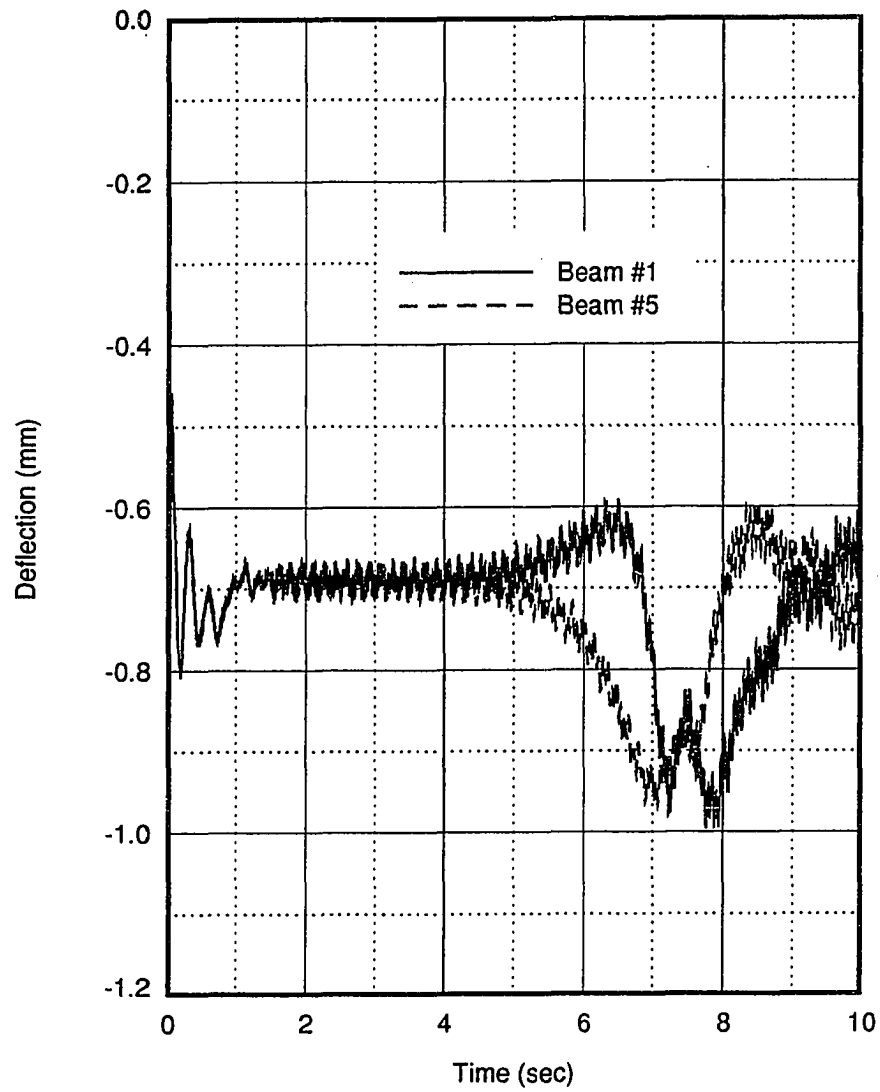


Figure 6.33: Beam local vertical deflections at distal ends with impulse of 1 Newton and no initial tilt: $t_0 = 1$ second, $\omega = 60rpm$

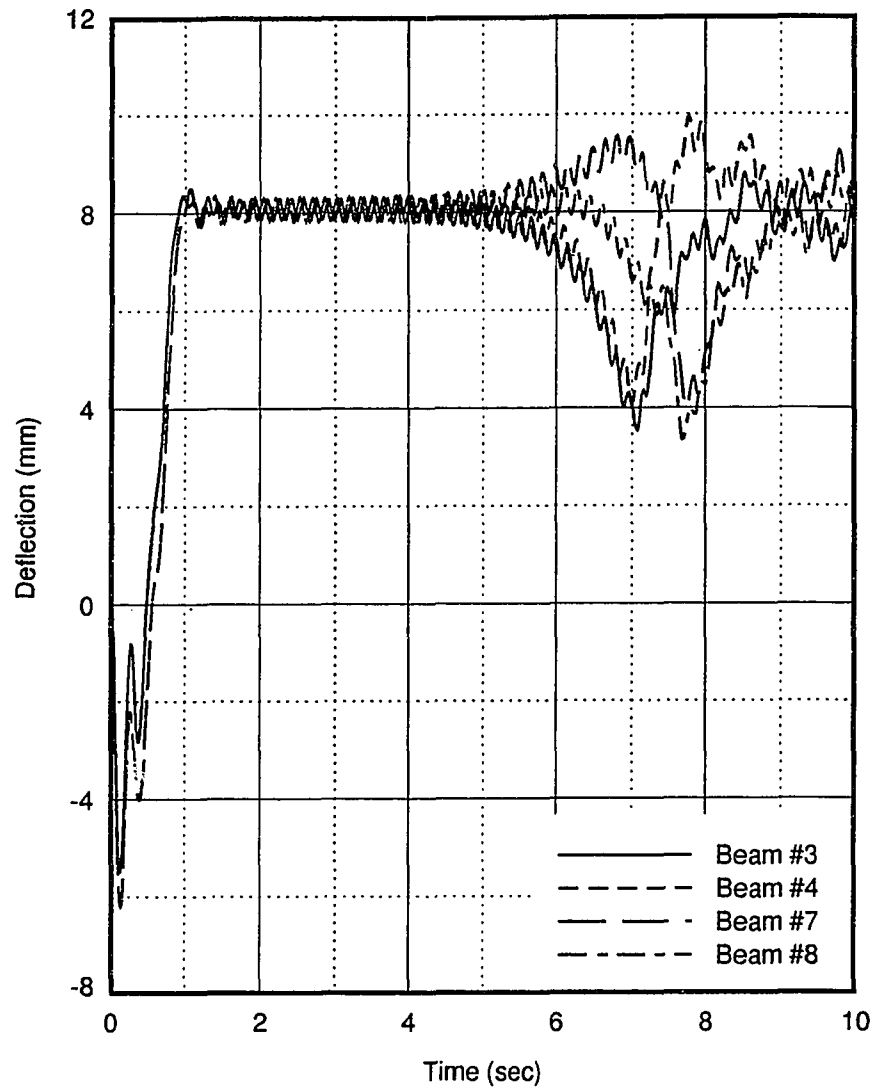


Figure 6.34: Beam local radial deflections at distal ends with impulse of 1 Newton and no initial tilt: $t_0 = 1$ second, $\omega = 60rpm$

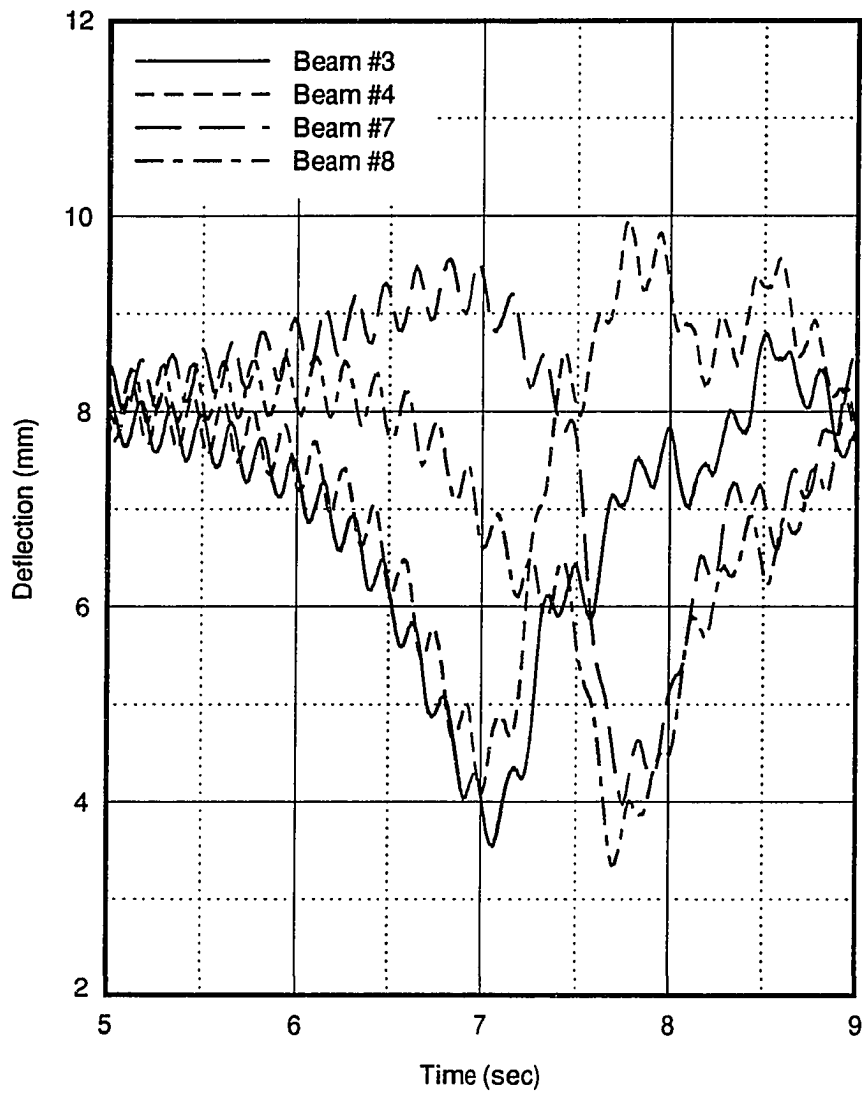


Figure 6.34 (Continued)

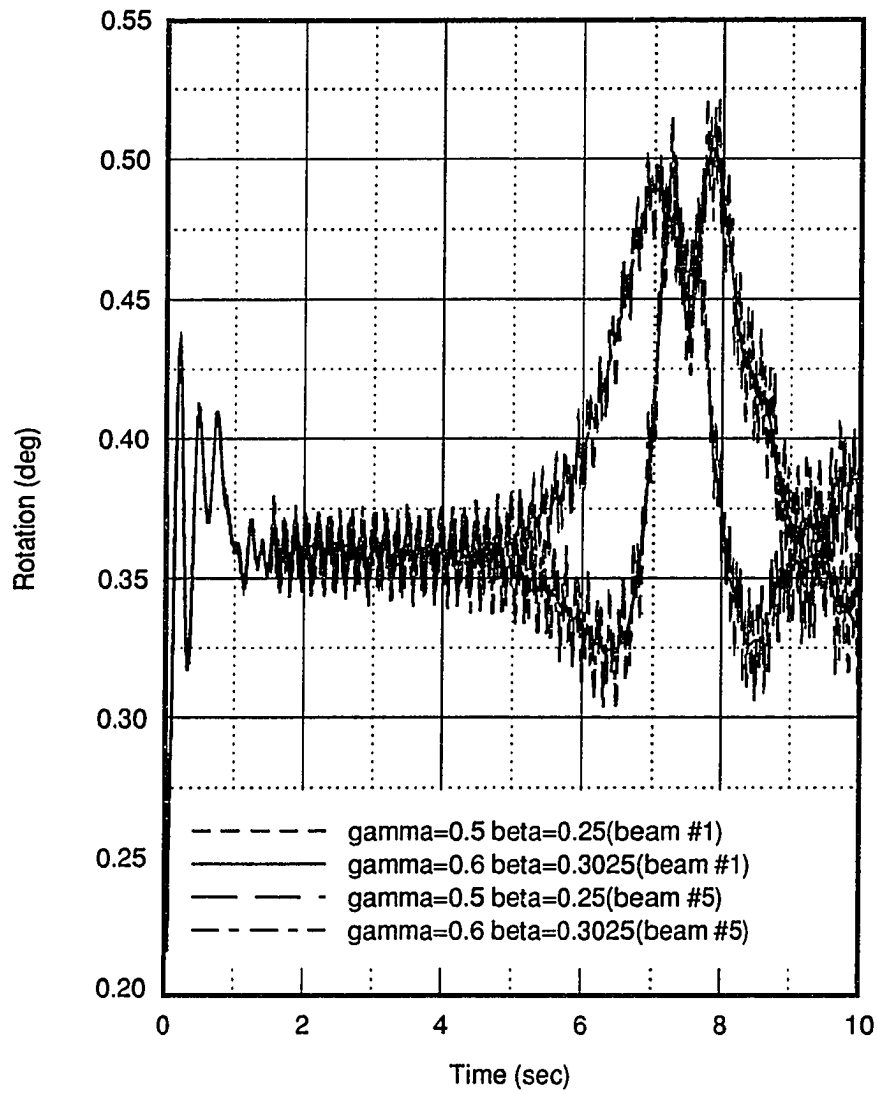


Figure 6.35: Beams 1 and 5 rotations about local Y axes at distal ends with impulse of 1 Newton and no initial tilt: $t_0 = 1$ second, $\omega = 60rpm$

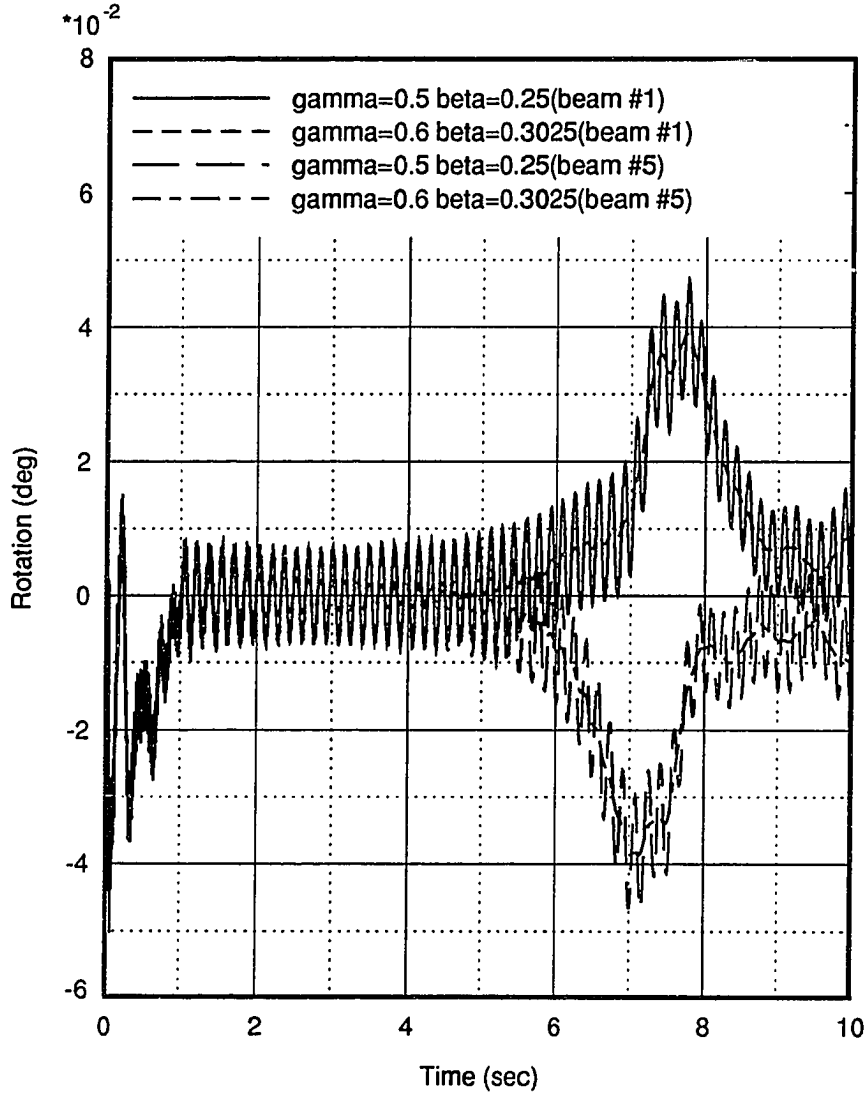


Figure 6.36: Beams 1 and 5 rotations about local Z axes at distal ends with impulse of 1 Newton and no initial tilt: $t_0 = 1$ second, $\omega = 60rpm$

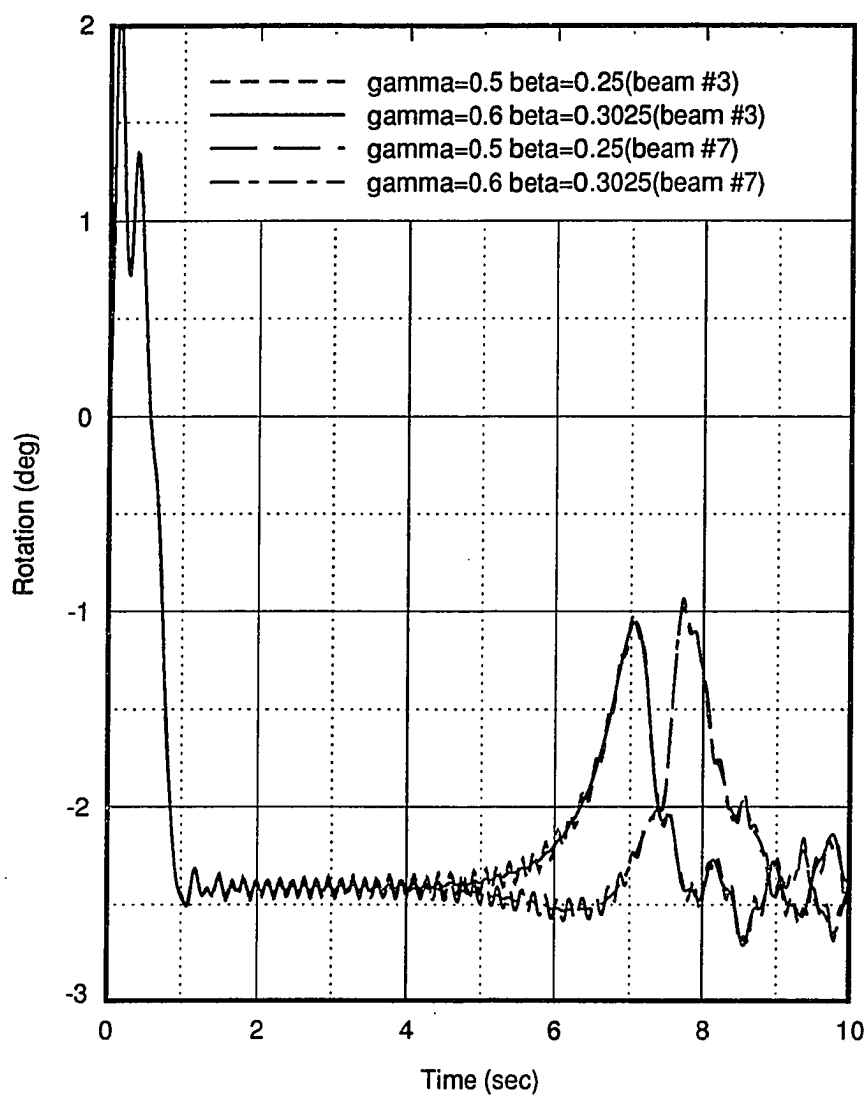


Figure 6.37: Beams 3 and 7 rotations about local Y axes at distal ends with impulse of 1 Newton and no initial tilt: $t_0 = 1$ second, $\omega = 60rpm$

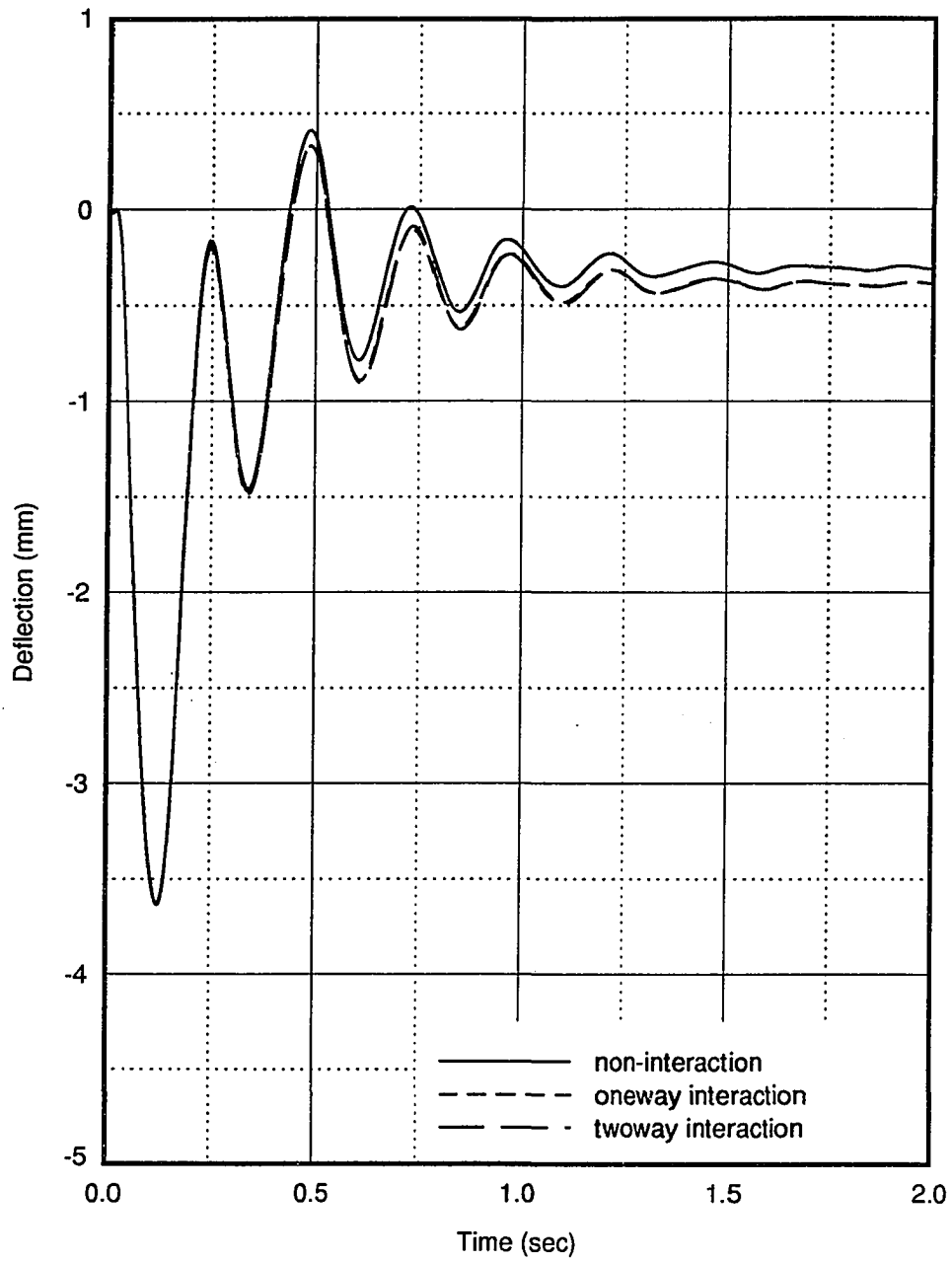


Figure 6.38: Comparison of local radial deflections at the tank center for non-interactive and interactive modes with collar up: $t_0 = 0.5$ second, $\omega = 30$ rpm, $\gamma = 0.6$, $\beta = 0.303$

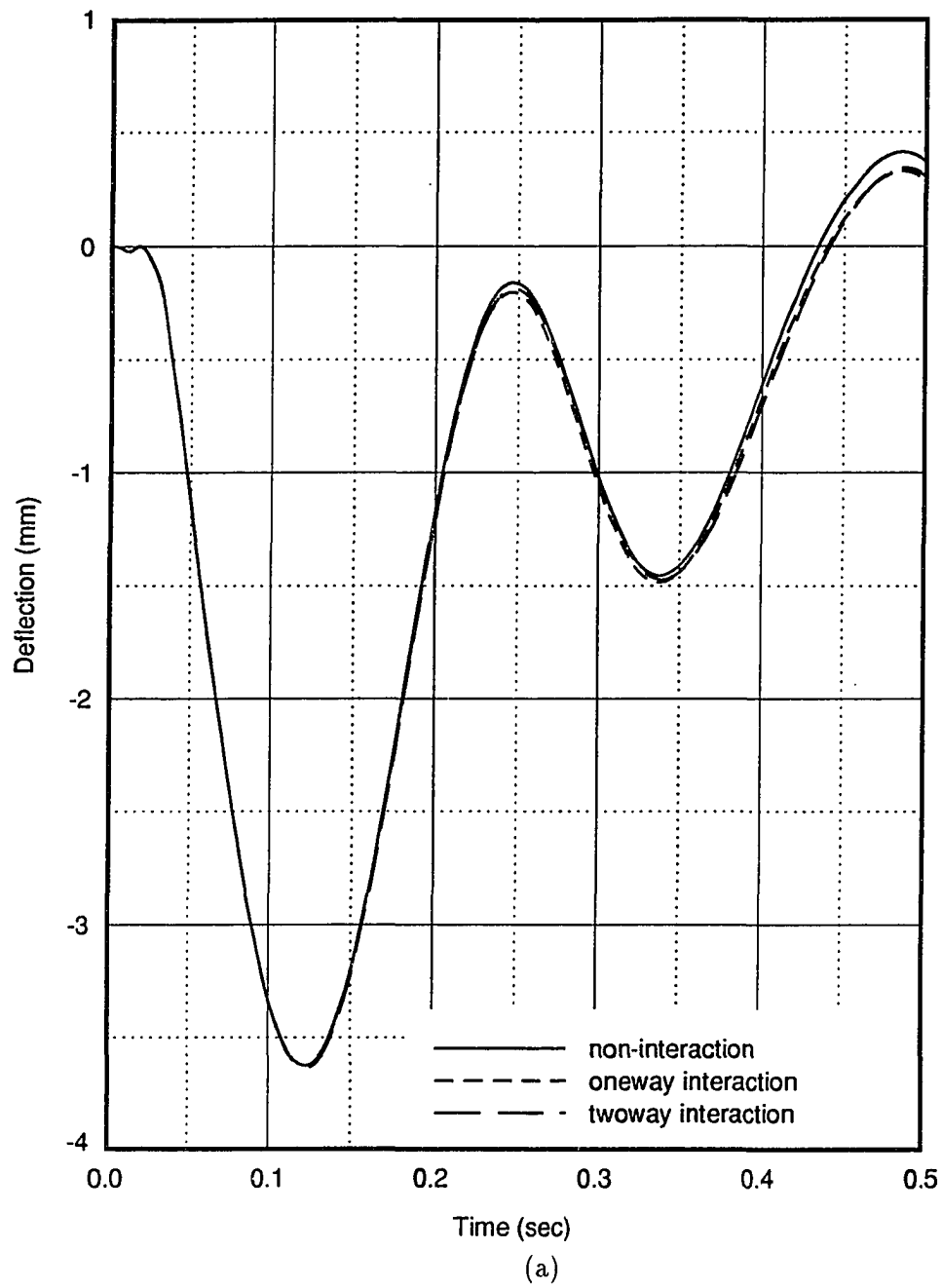


Figure 6.38 (Continued)

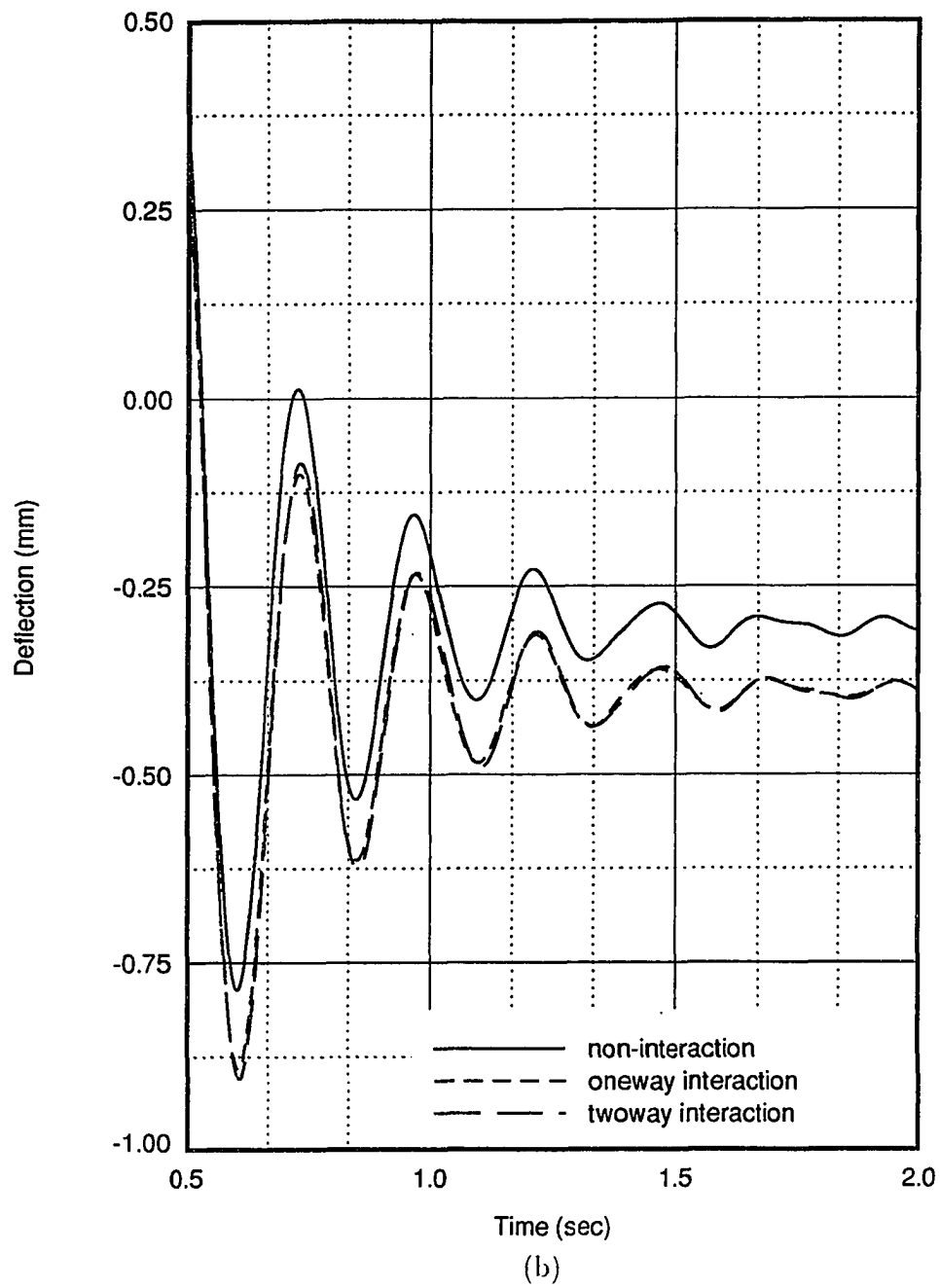


Figure 6.38 (Continued)

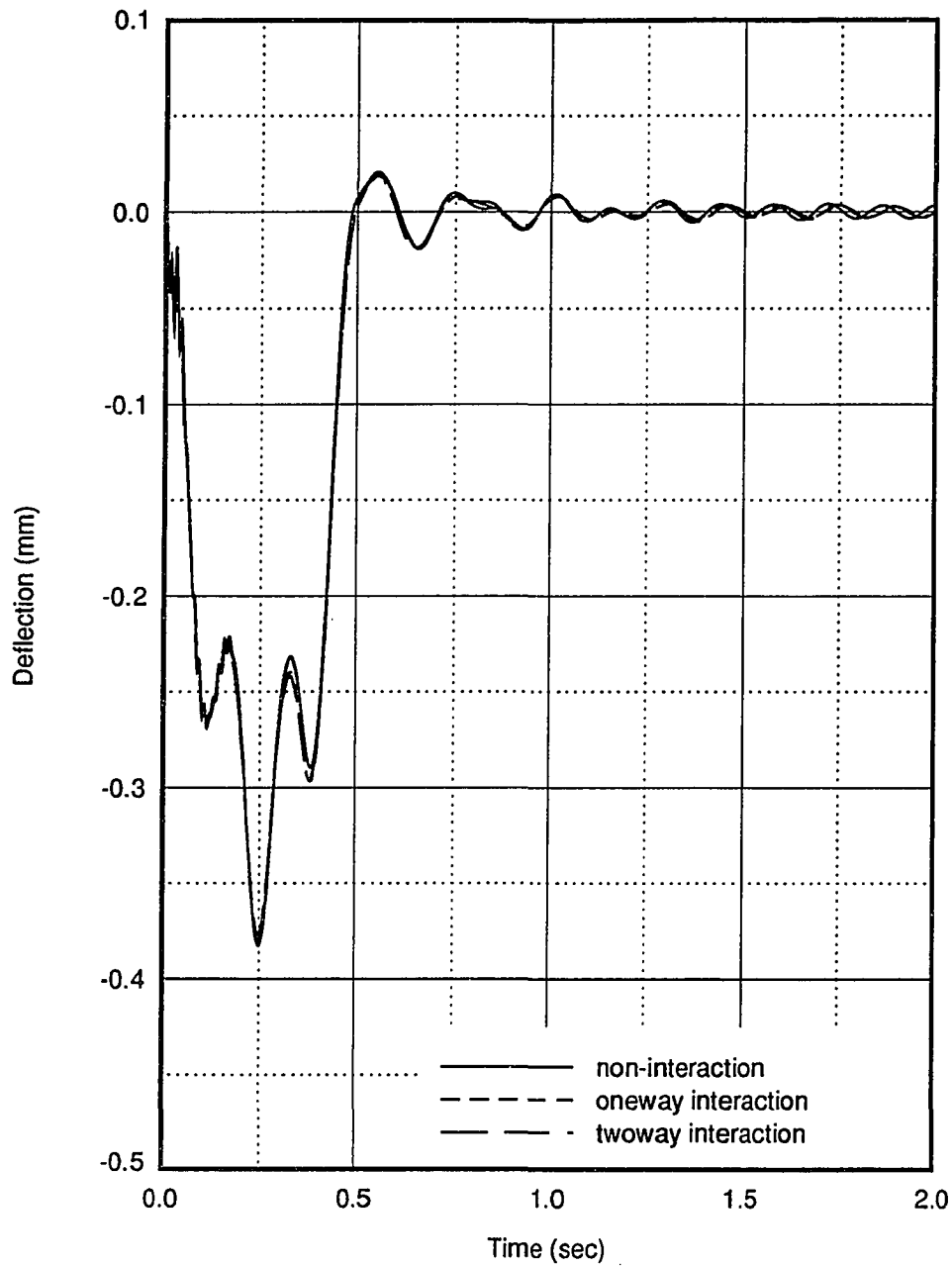


Figure 6.39: Comparison of local tangential deflections at the tank center for non-interactive and interactive modes with collar up: $t_0 = 0.5$ second, $\omega = 30rpm$, $\gamma = 0.6$, $\beta = 0.303$

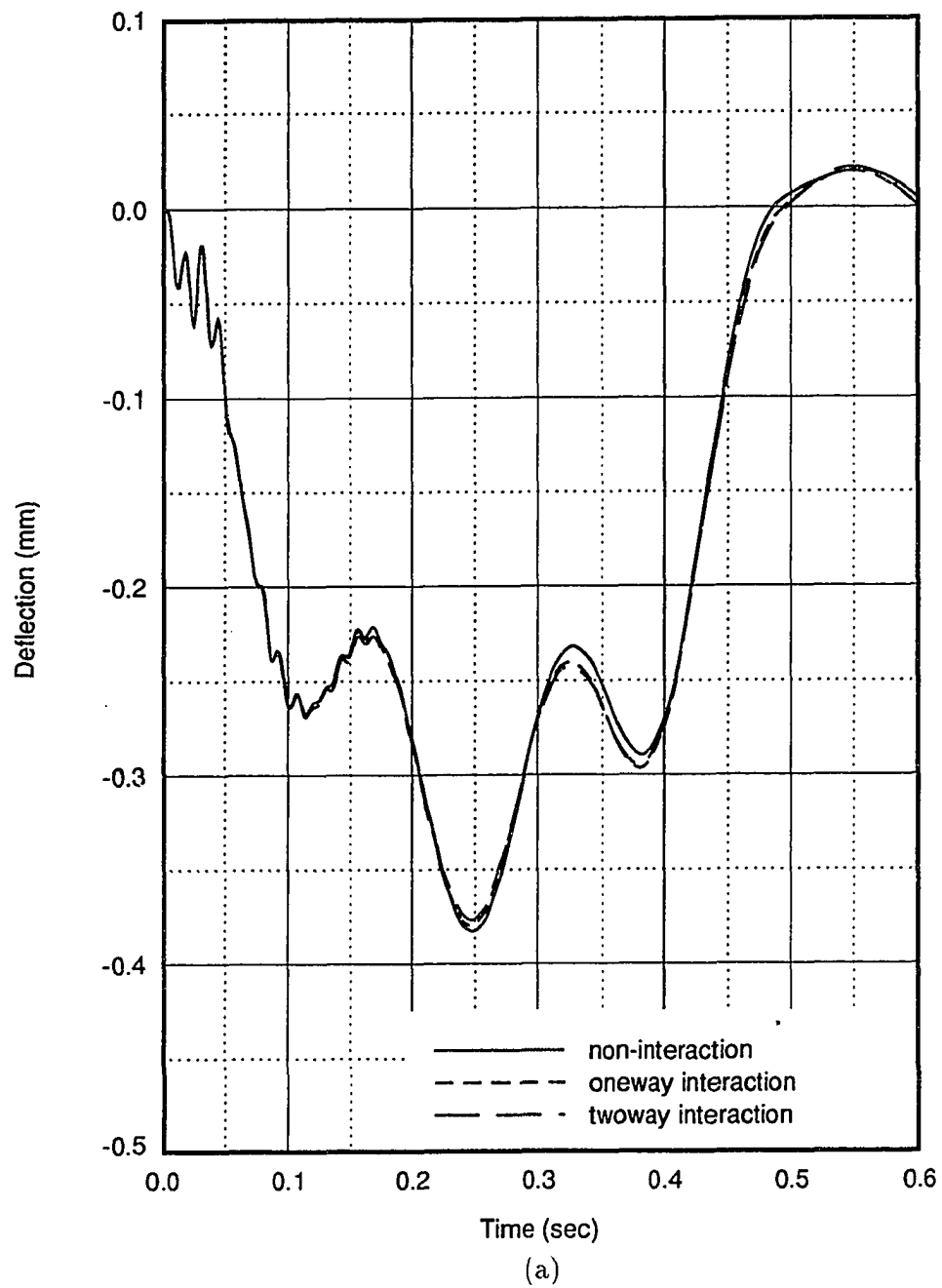


Figure 6.39 (Continued)

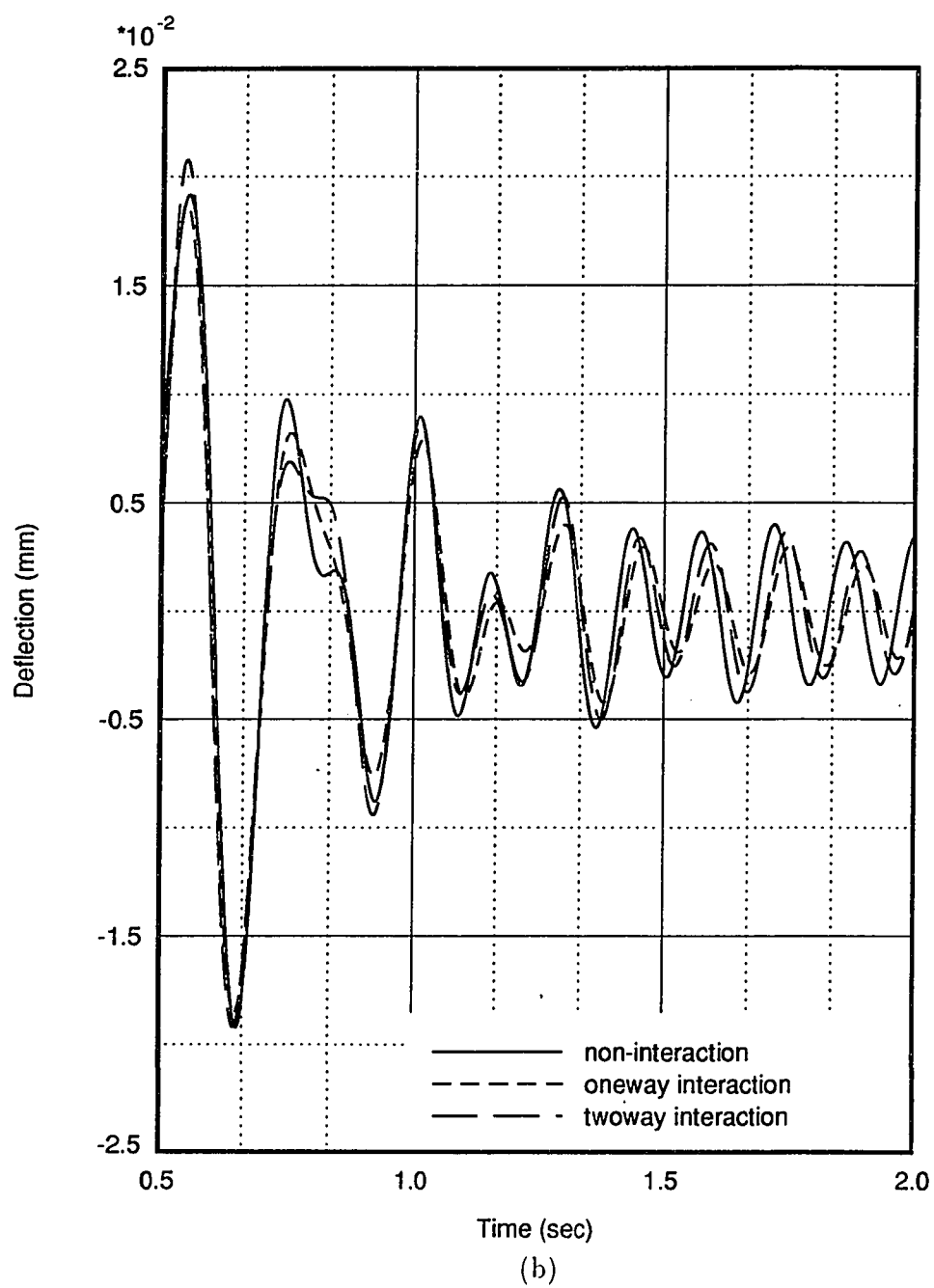


Figure 6.39 (Continued)

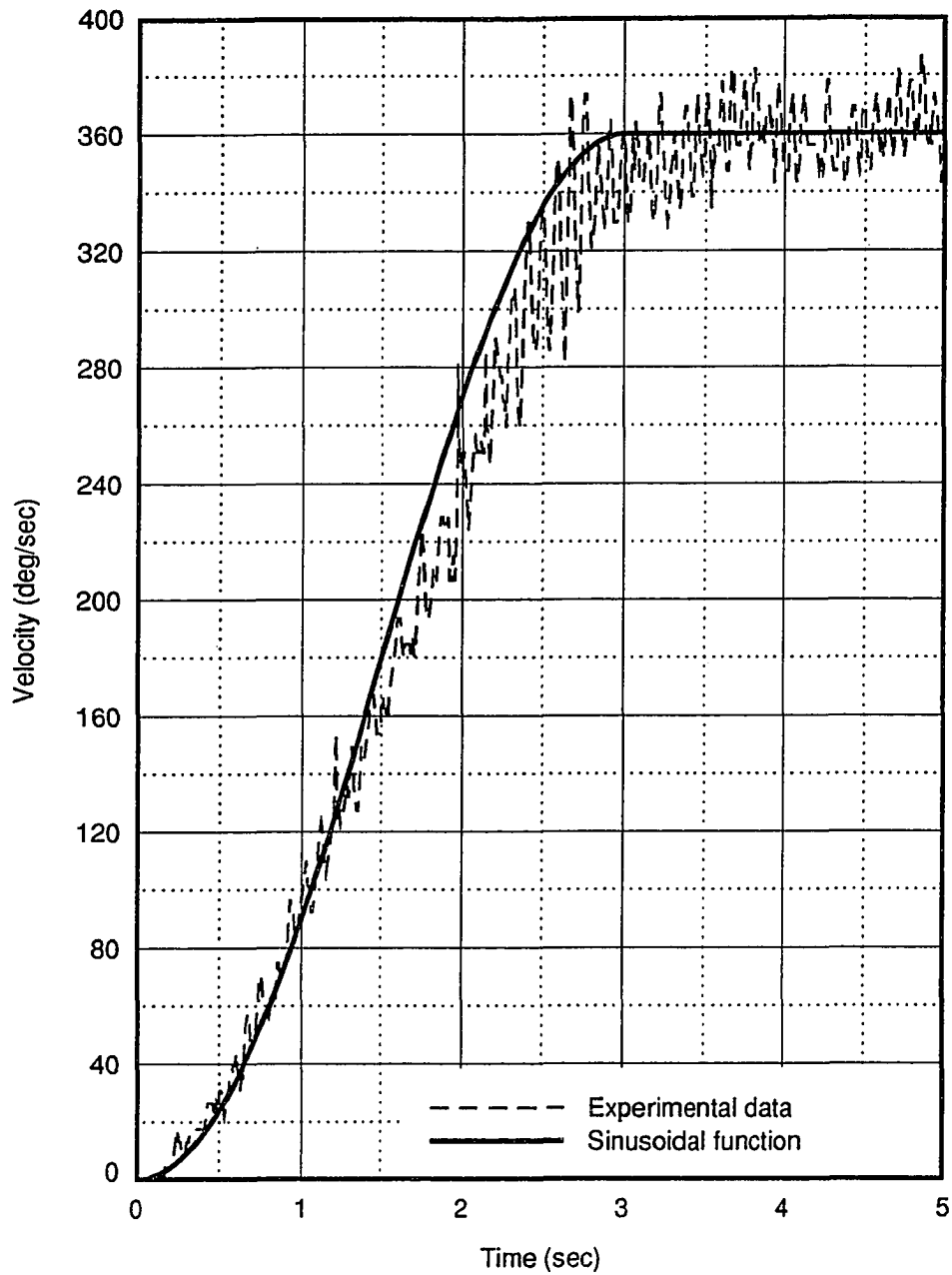


Figure 6.40: Lower shaft spin velocity profiles: measured case and ideal case

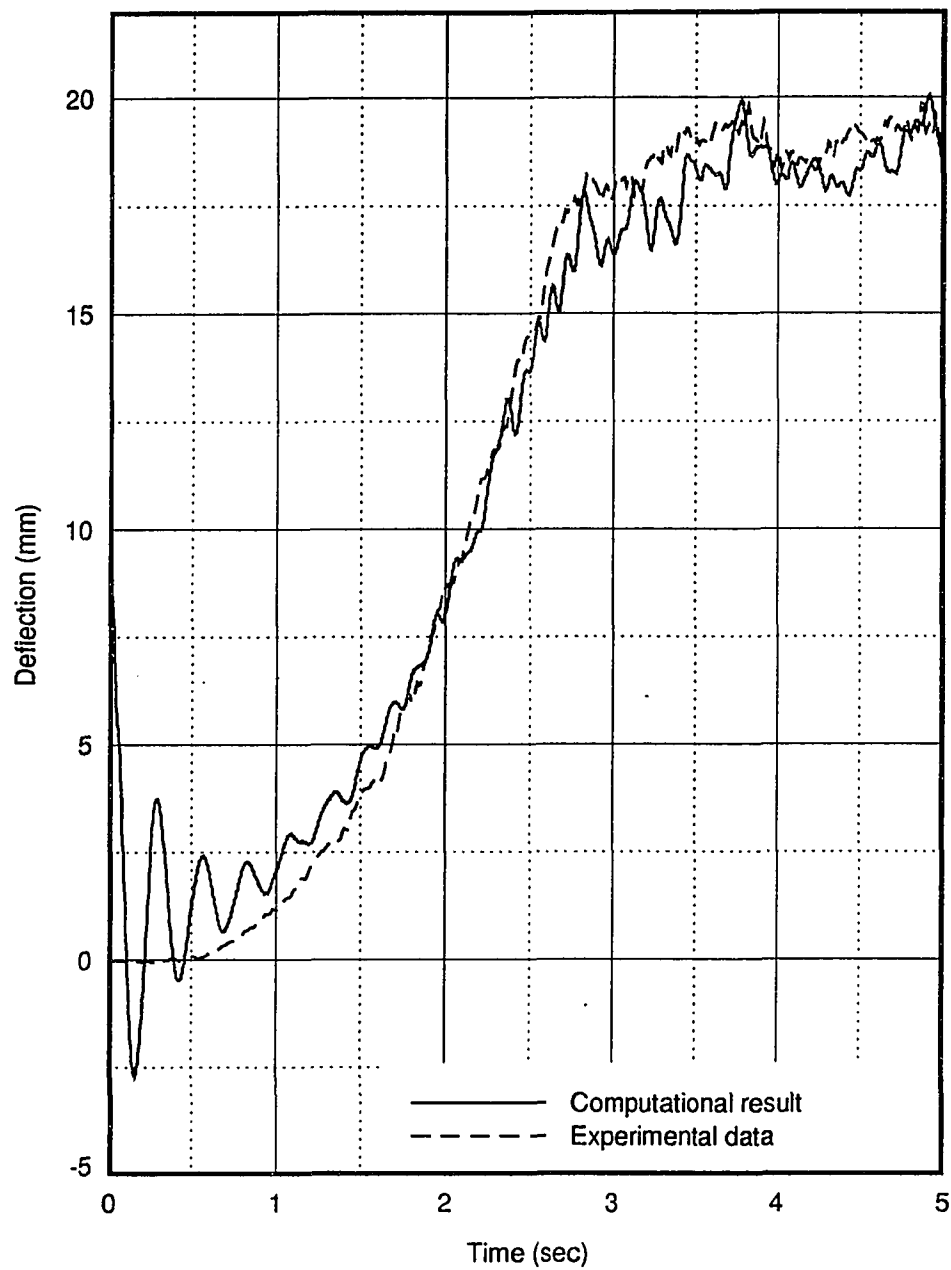


Figure 6.41: Overall radial deflection at tank center for a stable spin-up case with collar up: experimental and computational results

CHAPTER 7. GENERAL CONCLUSIONS

A systematic mathematical model which predicts the test rig dynamic response, and accounts for the mutual influence between the nonlinear rigid body motion and the linear elastic deformation, has been fully developed during the course of this study. The present model has a unique character which differs from and stands out from the traditional flexible system model. This characteristic is that the unknown global generalized coordinates include the elastic degrees of freedom, as well as the rigid body degrees of freedom. These two motions affect one another in the modeling, which then ultimately reflects the inherent nature of the problem under consideration. The nonlinear coupling terms in the dynamic equations of motion are completely derived in matrix form. A special numerical integration procedure is developed to solve the dynamic equations of motion with large valued rigid body motion and small valued elastic deformation. A sequential implicit-explicit time integration scheme is adopted in which a multiple predictor-corrector algorithm from Newmark family is used. A general FSD (*flexible structural dynamics*) computer code is written in which the CFD (*computational fluid dynamics*) modeling of the sloshing motion of the liquid is accommodated. The numerical simulation for several run cases has been performed. The results in general agree with observations. Measurement instrumentation has been set up, and the experimental test run has been performed

for a stable spin-up case. The measured data are compared with the numerical results for the corresponding case, and good agreement is indicated. A fluid-structure interaction analysis has been initiated for the stable spin-up case. The results of the non-interactive mode and the interactive mode are very close. The ultimate goals of this satellite project are to simulate satellite orbital motion using the test rig and to predict the instability range under certain circumstances using the FSD and CFD models. Therefore, future work is recommended as follows:

- Passive interaction mechanisms in terms of inertia coupling and force coupling should be further investigated.
- Extensive interaction runs linking the FSD and CFD computer codes should be performed for three different interaction modes, non-interaction, one-way interaction, and two-way interaction.
- Stability analysis of dynamic response for multibody flexible structures might be studied to predict the instability range and to guide the interaction simulation.
- Further experimental measurements corresponding to specific interaction simulations should be performed in order to validate the interaction mechanisms.

Major contributions of this research have been achieved in the investigation of the dynamic response of a flexible structural system in terms of motion coupling of unknown rigid body motion and elastic deformation, three-dimensional analysis of the elastic member discretization, and development of a numerical technique for solving the totally coupled dynamic equation system. Due to the nature of the test rig structure under consideration for this study, rigid body degrees of freedom of

the structure at the spherical universal joint are unknown and cannot be specified. Instead, these two rotations must be calculated through the modeling. Therefore, the generalized coordinates of the entire structure include not only the elastic deformation coordinates but also the unknown rigid body coordinates. As a consequence, the unknown motion coupling terms involve the elastic-rigid coupling, elastic-elastic coupling in the presence of the unknown rigid body motion, and the rigid-rigid coupling in the presence of the unknown elastic deformation. The formulation of the governing equations of motion is much more complicated in determining these nonlinear coupling terms which are neglected historically. To date, most of the study of the dynamics of flexible structural systems is limited in the investigation of the link elastic deformation and its effects on specified nominal rigid body motion in the area of mechanisms. In the area of robotics, investigators have studied the cause of oscillation of the actuator hand moving along a described trajectory due to the elastic deflections and rotations of the preceding flexible arms. Under this current study, however, it is desired to compute the elastic deformation of the individual flexible members of the structure undergoing unknown rigid body motion, as well as to predict this unknown rigid body motion accounting for the effects of the elastic deformation and the mutually dependent interaction between the rigid body motion and elastic deformation.

The current work has also extended the finite element modeling of elastic members involved in the flexible system dynamics to a three-dimensional spatial problem which further increases the complexity of the modeling. In order to reduce the elastic degrees of freedom and hence the number of equations, some assumptions regarding elastic deformation are proposed to complete the formulation of the finite element

discretization for continuous elastic members. *Two phase* displacement definition is employed instead of *one phase* representation of the coordinates [82] [54] [55] [24] [25] [26] [85] in defining the system generalized coordinates which include the rigid body motion and the elastic deformation.

The numerical technique developed in this study is designed specially for integrating structural dynamic equations with large valued rigid body variables and small valued elastic variables. The key step in the procedure of numerical integration is to separate the equation system into two groups. One of the groups represents the nonlinear rigid body motion and the other group represents the linear elastic motion. The linear equation system is integrated first using an implicit method with a Newmark predictor-corrector scheme. The nonlinear equation system is then integrated using an explicit method where the elastic kinematic properties are backward substituted.

REFERENCES

- [1] Amirouche, F.M.L., 1987, "Equations of Motion for Two-Arm Manipulators Including the Effects of Flexibility," *International Journal of Robotics and Automation*, Vol. 2, No. 2, pp. 77-84.
- [2] Amirouche, F.M.L., and Huston, R.L., 1988, "Dynamics of Large Constrained Flexible Structures," *ASME Journal of Dynamic Systems, Measurement, and Control*, Vol. 110, pp. 78-83.
- [3] Anderson, M.D., 1988, "Instrumentation of A Spin-Stabilized Spacecraft Simulator with Liquid Fuel Stores," M.S. thesis, Iowa State University, Ames, Iowa.
- [4] Bathe, K.J., 1982, *Finite Element Procedures in Engineering Analysis*, Prentice-Hall, Englewood Cliffs, New Jersey.
- [5] Bathe, K.J., and Wilson, E.L., 1984, *Numerical Methods in Finite Element Analysis*, Prentice-Hall, Englewood Cliffs, New Jersey.
- [6] Battin, R.H., 1987, *An Introduction to The Mathematics and Methods of ASTRODYNAMICS*, AIAA Education Series, New York, New York.
- [7] Baumgarten, J.R., Flugrad, D.R., and Prusa, J.M., 1989, "Investigation of Liquid Sloshing in Spin-Stabilized Satellites," ISU-ERI-Ames-90401, Iowa State University, Ames, Iowa.
- [8] Baumgarten, J.R., Flugrad, D.R., and Pletcher, R.H., 1990, "Investigation of Liquid Sloshing in Spin-Stabilized Satellites," ISU-ERI-Ames-90410, Iowa State University, Ames, Iowa.
- [9] Baumgarten, J.R., Flugrad, D.R., and Pletcher, R.H., 1991, "Investigation of Liquid Sloshing in Spin-Stabilized Satellites," ISU-ERI-Ames-92400, Iowa State University, Ames, Iowa.

- [10] Baumgarten, J.R., Flugrad, D.R., and Pletcher, R.H., 1992, "Investigation of Liquid Sloshing in Spin-Stabilized Satellites," Iowa State University, Ames, Iowa.
- [11] Bayo, E., 1987, "A Finite Element Approach to Control the End-Point Motion of a Single-Link Flexible Robot," *Journal of Robotics Systems*, Vol. 4, No. 1, pp. 63-75.
- [12] Bayo, E., *et al*, 1988, "Inverse Dynamics of a Single-Link Flexible Robot: Analytical and Experimental Results," *International Journal of Robotics and Automation*, Vol. 3, No. 3, pp. 150-157.
- [13] Bayo, E., and Serna, M.A., 1989, "Inverse Dynamics and Kinematics of Multi-Link Elastic Robots: An Iterative Frequency Domain Approach," *International Journal of Robotics Research*, M.I.T., Vol. 8, No. 6, pp. 49-62.
- [14] Belytschko, T., Yen, H.J., and Mullen, R. 1979, "Mixed Method for Time Integration," *Computer Method in Applied Mechanics and Engineering*, Vol. 17/18, pp. 259-275.
- [15] Book, W.J., 1974, "Modeling, Design and Control of Flexible Manipulator Arms," Ph.D. dissertation, Massachusetts Institute of Technology, Cambridge, Massachusetts.
- [16] Book, W.J., 1979, "Analysis of Massless Elastic Chains with Servo Controlled Joints," *ASME Journal of Dynamic Systems, Measurement, and Control*, Vol. 101, pp. 187-192.
- [17] Book, W.J., 1984, "Recursive Lagrangian Dynamics of Flexible Manipulator Arms," *International Journal of Robotics Research*, M.I.T., Vol. 3, No. 3, pp. 87-101.
- [18] Boresi, A.P., and Lynn, P.P., 1974, *Elasticity in Engineering Mechanics*, 2nd Ed., Prentice-Hall, Englewood Cliffs, New Jersey.
- [19] Boresi, A.P., and Sidebottom, O.M., 1978, *Advanced Mechanics of Materials*, 3rd Ed., John Wiley & Sons, New York, New York.
- [20] Chang, L.-W., and Hamilton, J.F., 1988, "A Sequential Integration Method," *ASME Journal of Dynamic Systems, Measurement, and Control*, Vol. 110, pp. 382-388.
- [21] Chang, L.-W., and Hamilton, J.F., 1991, "Simulation of Flexible-Link Manipulators Applying Sequential Integration Method," *ASME Journal of Dynamic Systems, Measurement, and Control*, Vol. 113, pp. 175-178.

- [22] Chang, L.-W., and Hamilton, J.F., 1991, "The Kinematics of Robotic Manipulators with Flexible Links Using an Equivalent Rigid Link System (ERLS) Model," *ASME Journal of Dynamic Systems, Measurement, and Control*, Vol. 113, pp. 48-52.
- [23] Chang, L.-W., and Hamilton, J.F., 1991, "Dynamics of Robotic Manipulators with Flexible Links," *ASME Journal of Dynamic Systems, Measurement, and Control*, Vol. 113, pp. 54-59.
- [24] Chang, C.W., and Shabana, A.A., 1988, "Dynamic Motion of High Cyclic Rate Flexible Mechanisms," *ASME Trends and Developments in Mechanisms, Machines, and Robotics*, Vol. 2, pp. 387-396.
- [25] Chang, C.W., and Shabana, A.A., 1990, "Spatial Dynamics of Deformable Multi-body Systems with Variable Kinematic Structure: Part 1-Dynamic Model," *ASME Journal of Mechanical Design*, Vol. 112, pp. 153-159.
- [26] Chang, C.W., and Shabana, A.A., 1990, "Spatial Dynamics of Deformable Multi-body Systems with Variable Kinematic Structure: Part 2-Velocity Transformation," *ASME Journal of Mechanical Design*, Vol. 112, pp. 160-167.
- [27] Chen, K.-H., 1990, "A Primitive Variable, Strongly Implicit Calculation Procedure For Two and Three-Dimensional Unsteady Viscous Flows: Application to Compressible and Incompressible Flows Including Flows with Free Surfaces," Ph.D. dissertation, Iowa State University, Ames, Iowa.
- [28] Chen, K.-H., and Pletcher, R.H., 1990, "A Primitive Variable, Strongly Implicit Calculation Procedure for Viscous Flows at All Speeds," AIAA-90-1521, *AIAA 21th Fluid Dynamics, Plasma Dynamics, and Lasers Conference*, June 18-20, Seattle, Washington.
- [29] Chen, K.-H., and Pletcher, R.H., 1991, "Simulation of Three-Dimensional Liquid Sloshing Flows Using a Strongly Implicit Calculation Procedure," AIAA-91-1661.
- [30] Chen, K.-H., Kelecý, F.J., and Pletcher, R.H., 1992, "A Numerical and Experimental Study of Three-Dimensional Liquid Sloshing in a Rotating Spherical Container," AIAA-92-0829, presented at the *30th Aerospace Sciences Meeting*, January 6-9, Reno, Nevada.
- [31] Cleghorn, W.L., Fenton, R.G., and Tabarrok, B., 1981, "Finite Element Analysis of High-Speed Flexible Mechanisms," *Mechanism and Machine Theory*, Vol. 16, No. 4, pp. 407-424.

- [32] Cleghorn, W.L., Fenton, R.G., and Tabarrok, B., 1984, "Steady-State Vibrational Response of High-Speed Flexible Mechanisms," *Mechanism and Machine Theory*, Vol. 19, No. 4/5, pp. 417-423.
- [33] Cleghorn, W.L., and Chao, K.c., 1988, "Kineto-Elastodynamic Modeling of Mechanisms Employing Linearly Tapered Beam Finite Elements," *Mechanism and Machine Theory*, Vol. 23, No.5, pp. 333-342.
- [34] Cook, R.D., 1981, *Concepts and Applications of Finite Element Analysis*, Wiley, New York, New York.
- [35] Cowles, D.S., 1987, "Design of A Spin-Stabilized Spacecraft Simulator with Liquid Fuel Stores," M.S. thesis, Iowa State University, Ames, Iowa.
- [36] Denavit, J., and Hartenberg, R.S., 1955, "A Kinematic Notation for Lower Pair Mechanisms Based on Matrices," *ASME Journal of Applied Mechanics*, Vol. 22, pp. 215-221.
- [37] De Smet, M., *et al*, 1989, "Dynamic Analysis of Flexible Structures Using Component Mode Synthesis," *ASME Journal of Applied Mechanics*, Vol. 56, pp. 874-880.
- [38] Flugrad, D. R., Anderson, M. D., and Cowles, D. S., 1990, "Experimental Study of a Test Rig to Simulate Liquid Sloshing in Spin-Stabilized Satellites," under review by *AIAA Journal of Guidance, Control, and Dynamics*.
- [39] Flugrad, D. R., and Obermaier, L.A. (in press), 1990, "Computer Simulation of a Test Rig to Model Liquid Sloshing in Spin-Stabilized Satellites," *ASME Journal of Dynamics Systems, Measurements, and Control*.
- [40] Gaultier, P.E., and Cleghorn, W.L., 1989, "Modeling of Flexible Manipulator Dynamics: A Literature Survey," *The 1st National Conference on Applied Mechanisms and Robotics*, Cincinnati, Ohio, 89AMR-2C-3, pp. 1-10.
- [41] Hill, D.E., 1985, "Dynamics and Control of Spin-Stabilized Spacecraft with Sloshing Fluid Stores," Ph.D. dissertation, Iowa State University, Ames, Iowa.
- [42] Hill, D.E., Baumgarten, J.R., and Miller, J.T., 1988, "Dynamic Simulation of Spin-Stabilized Spacecraft with Sloshing Fluid Stores," *AIAA Journal of Guidance, Control, and Dynamics*, Vol. 11, No. 6, pp. 597-599.
- [43] Hill, D.E., Baumgarten, J.R. (in press), 1992, "Control of Spin-Stabilized Spacecraft with Sloshing Fluid Stores", *ASME Journal of Dynamic System, Measurements, and Control*.

- [44] Huang, R. C., and Wiley, J. C., "Combining Rigid Body Dynamic And Finite Element Structural Models — A Comparison of Three Approaches," *ANSYS Conference Proceedings*, pp. 6.47 – 6.52, 1985.
- [45] Hughes, T.J.R., and Liu, W.K., 1978, "Implicit-Explicit Finite Elements in Transient Analysis: Stability Theory," *ASME Journal of Applied Mechanics*, Vol. 45, pp. 371-374.
- [46] Hughes, T.J.R., and Liu, W.K., 1978, "Implicit-Explicit Finite Elements in Transient Analysis: Implementation and Numerical Examples," *ASME Journal of Applied Mechanics*, Vol. 45, pp. 375-378.
- [47] Hughes, T.J.R., Pister, K.S., and Taylor, R.L., 1979, "Implicit-Explicit Finite Elements in Nonlinear Transient Analysis", *Computer Method in Applied Mechanics and Engineering*, Vol. 17/18, pp. 159-182.
- [48] Hurty, W.C., 1965, "Dynamic Analysis of Structural Systems Using Component Modes," *AIAA Journal*, Vol. 3, No. 4, pp. 678-685.
- [49] Hurty, W.C., 1960, "Vibrations of Structural Systems by Component Mode Synthesis," *ASCE Journal of Engineering Mechanics Division*, Vol. 86, pp. 51-69.
- [50] Kane, T.R., and Levinson, D.A., 1983, *Dynamics Theory and Applications*, McGraw-Hill, London.
- [51] Kassinos, A.C., and Prusa, J.M., 1989, "Numerical Solution of 2-D Viscous Sloshing in Rectangular Containers of Finite Aspect Ratio," under review by *Journal of Computational Physics*.
- [52] Kassinos, A.C., and Prusa, J.M., 1990, "Study of 3-D Viscous Sloshing in Spherical Containers," presented at *ASME Winter Annual Meeting*, November 25-30, Dallas, Texas.
- [53] Kelecý, F.J., Sethuramin, B., Xu, J., *et al*, 1992, "Three-Dimensional Liquid Sloshing in a rotating Spherical Container: Accounting for The Fluid-Structure Interaction", *Symposium on High-Performance Computing for flight Vehicles*, December 7-9, Washington, D.C.
- [54] Khulief, Y.A., and Shabana, A.A., 1986, "Dynamic Analysis of Constrained Systems of Rigid and Flexible Bodies with Intermittent Motion," *ASME Journal of Mechanisms, Transmissions, and Automation in Design*, Vol. 108, pp. 38-45.

- [55] Khulief, Y.A., 1988, "Physical and Modal Coordinates for Dynamic Analysis of Flexible Mechanisms," *ASME Trends and Developments in Mechanisms, Machines, and Robotics*, Vol. 2, pp. 417-422.
- [56] Logan, D.L., 1986, *A First Course in The Finite Element Analysis*, PWS Publishers, New York, New York.
- [57] Low, K.H., 1987, "A Systematic Formulation of Dynamic Equations for Robot Manipulators with Elastic Links," *Journal of Robotic Systems*, Vol. 4(3), pp. 435-456.
- [58] Low, K.H., 1989, "Solution Schemes for The System Equations of Flexible Robots," *Journal of Robotic System*, Vol. 6(4), pp. 383-405.
- [59] Low, K.H., and Vidyasagar, M., 1988, "A Lagrangian Formulation of the Dynamic Model for Flexible Manipulator Systems," *ASME Journal of Dynamic Systems, Measurement, and Control*, Vol. 110, pp. 175-181.
- [60] The Macneal-Schwendler Corp., 1989, *MSC/NASTRAN User's Manual*, New York, New York.
- [61] Mitchell, L.D., 1985, "Proposed Solution Methodology for the Dynamically Coupled Nonlinear Geared Rotor Mechanics Equations," *ASME Journal of Vibration, Acoustics, Stress, and Reliability in Design*, Vol. 107, pp. 112-116.
- [62] Naganathan, G., and Soni, A.H., 1986, "Non-Linear Flexibility Studies for Spatial Manipulators," *IEEE International Conference on Decision and Control*, Vol. 1, pp. 373-378.
- [63] Naganathan, G., and Soni, A.H., 1987, "Coupling Effects of Kinematics and Flexibility in Manipulators," *International Journal of Robotics Research*, M.I.T., Vol. 6, No. 1, pp. 75-84.
- [64] Naganathan, G., and Soni, A.H., 1987, "An Analytical and Experimental Investigation of Flexible Manipulator Performance," *IEEE International Conference on Robotics and Automation*, Vol. 1, pp. 767-773.
- [65] Naganathan, G., and Soni, A.H., 1988, "Nonlinear Modeling of Kinematic and Flexibility Effects in Manipulator Design," *ASME Journal of Mechanisms, Transmissions, and Automation Design*, Vol. 110, pp. 243-254.
- [66] Naganathan, G., and Soni, A.H., 1986 "Dynamic Response of a Manipulator," *ASME 9th Applied Mechanisms Conference Proceedings*, Kansas City, Missouri, Vol. 2, Session XII.A, pp. III.1-6.

- [67] Nagarajan, S., and Turcic, D.A., 1990, "Lagrangian Formulation of the Equations of Motion for Elastic Mechanisms With Mutual Dependence Between Rigid Body and Elastic Motions. Part I: Element Level Equations," *ASME Journal of Dynamic Systems, Measurement, and Control*, Vol. 112, pp. 203-214.
- [68] Nagarajan, S., and Turcic, D.A., 1990, "Lagrangian Formulation of the Equations of Motion for Elastic Mechanisms With Mutual Dependence Between Rigid Body and Elastic Motions. Part II: System Equations," *ASME Journal of Dynamic Systems, Measurement, and Control*, Vol. 112, pp. 215-224.
- [69] Nagarajan, S., Jablokow, A.G., and Turcic, D.A., 1988, "A Modal Analysis Solution Technique to the Equations of Motion for Elastic Mechanism Systems Including the Rigid-Body and Elastic Motion Coupling Terms," *ASME Trends and Developments in Mechanisms, Machines, and Robotics*, Vol. 2, pp. 407-416.
- [70] Newmark, N.M., 1952, "Computation of Dynamic Structural Response in the Range Approaching Failure," *Proceedings of the Symposium on Earthquake and Blast Effects on Structures*, Los Angeles, Published by Earthquake Engineering Research Institute.
- [71] Newmark, N.M., 1952, "Analysis and Designs of Structures Subjected to Dynamic Loading," *M.I.T. Conference on Building in the Atomic Age*.
- [72] Newmark, N.M., 1950, "Methods of Analysis of Structures Subjected to Dynamic Loading," Report issued by Directorate of Intelligence, U.S. Air Force, December 18.
- [73] Newmark, N.M., 1959, "A Method of Computation for Structural Dynamics," *ASCE Journal of the Engineering Mechanics Division*, Vol. 85, pp. 67-94.
- [74] Nikravesh, P.E., 1988, *Computer Aided Analysis of Mechanical Systems*, Prentice-Hall, Englewood Cliffs, New Jersey.
- [75] Obermaier, L.A., 1988, "Computer Simulation of A Spin-Stabilized Spacecraft Simulator with Liquid Stores," M.S. thesis, Iowa State University, Ames, Iowa.
- [76] Schick, T.E., 1990, "Motion Study of a Spin-Stabilized Satellite Test Rig," M.S. thesis, Iowa State University, Ames, Iowa.
- [77] Schick, T.E., and Flugrad, D.R., 1992, "Motion Study of a Spin-Stabilized Satellite Test Rig," under review by *AIAA Journal of Guidance, Control, and Dynamics*

- [78] Serna, M.A., and Bayo, E., 1988, "Penalty Formulations for the Dynamic Analysis of Elastic Mechanisms," *ASME Trends and Developments in Mechanisms, Machines, and Robotics*, Vol. 2, pp. 381-386.
- [79] Serna, M.A., and Bayo, E., 1988, "Numerical Implementation of Penalty Methods for the Analysis of Elastic Mechanisms," *ASME Trends and Developments in Mechanisms, Machines, and Robotics*, Vol. 2, pp. 449-456.
- [80] Serna, M.A., and Bayo, E., 1989, "A Simple and Efficient Computational Approach for the Forward Dynamics of Elastic Robots," *Journal of Robotic Systems*, Vol. 6, No. 4, pp. 363-382.
- [81] Shabana, A.A., and Wehage, R.A., 1983, "Variable Degree-of-Freedom Component Mode Analysis of Variant Flexible Mechanical Systems," *ASME Journal of Mechanisms, Transmissions, and Automation in Design*, Vol. 105, pp. 371-378.
- [82] Shabana, A.A., 1986, "Dynamics of Inertia Variant Flexible Systems Using Experimentally Identified Parameters," *ASME Journal of Mechanisms, Transmissions, and Automation in Design*, Vol. 108, pp. 358-366.
- [83] Simo, J.C., and Vu-Quoc, L., 1986, "On The Dynamics of Flexible Beams Under Large Overall Motions--The Plane Case: Part I," *ASME Journal of Applied Mechanics*, Vol. 53, pp. 849-854.
- [84] Simo, J.C., and Vu-Quoc, L., 1986, "On The Dynamics of Flexible Beams Under Large Overall Motions--The Plane Case: Part II," *ASME Journal of Applied Mechanics*, Vol. 53, pp. 855-863.
- [85] Song, J.O., and Haug, E.J., 1980, "Dynamic Analysis of Planar Flexible Mechanisms," *Computer Methods in Applied Mechanics and Engineering*, Vol. 24, pp. 359-381.
- [86] Sunada, W.H., and Dubowsky, S., 1981, "The Application of Finite Element Methods to the Dynamic Analysis of Flexible Spatial and Co-Planar Linkage Systems," *ASME Journal of Mechanical Design*, Vol. 103, pp. 643-651.
- [87] Sunada, W.H., and Dubowsky, S., 1983, "On the Dynamic Analysis and Behavior of Industrial Robotic Manipulators with Elastic Members," *ASME Journal of Mechanisms, Transmissions, and Automation in Design*, Vol. 105, pp. 42-51.
- [88] Timoshenko, S., and Gere, J., 1972, *Mechanics of Materials*, Van Nostrand Reinhold Company, New York, New York.

- [89] Turcic, D.A., and Midha, A., 1984, "Generalized Equations of Motion for the Dynamic Analysis of Elastic Mechanism Systems," *ASME Journal of Dynamic Systems, Measurement, and Control*, Vol. 106, pp. 243-248.
- [90] Turcic, D.A., and Midha, A., 1984, "Dynamic Analysis of Elastic Mechanism Systems. Part I: Applications," *ASME Journal of Dynamic Systems, Measurement, and Control*, Vol. 106, pp. 249-254.
- [91] Turcic, D.A., and Midha, A., 1984, "Dynamic Analysis of Elastic Mechanism Systems. Part II: Experimental Results," *ASME Journal of Dynamic Systems, Measurement, and Control*, Vol. 106, pp. 255-260.
- [92] Wood, W.L., 1990, *Practical Time-stepping Schemes*, Oxford Applied Mathematics and Computing Science Series, Clarendon Press, Oxford, England.
- [93] Xu, J., and Baumgarten, J.R., 1991, "A Finite Element/Lagrangian Formulation of Dynamic Motion Prediction for A Flexible Satellite Simulator with Both Rigid and Elastic Bodies," *Proceedings of the 2nd National Applied Mechanisms and Robotics Conference*, Cincinnati, Ohio, November 3-6, 91AMR-VIIB-5, pp. 1-8.
- [94] Xu, J., and Baumgarten, J.R., 1992, "Modeling of Flexible Multibody Articulated Structures with Mutually Coupled Motions. Part I: General Theory," *Proceedings of the 22nd ASME Mechanisms Conference*, Phoenix/Scottsdale, Arizona, September 13-16.
- [95] Xu, J., and Baumgarten, J. R., 1992, "Modeling of Flexible Multibody Articulated Structures with Mutually Coupled Motions. Part II: Application and Results," *Proceedings of the 22nd ASME Mechanisms Conference*, Phoenix/Scottsdale, Arizona, September 13-16.
- [96] Xu, J., and Baumgarten, J.R., 1992, "A Sequential Implicit-Explicit Integration Method in Solving Nonlinear Differential Equations from Flexible System Modeling," *Proceedings of the 22nd ASME Mechanisms Conference*, Phoenix/Scottsdale, Arizona, September 13-16.
- [97] Yang, Z., Sadler, J.P., and Rouch, K.E., 1988, "The Use of ANSYS for the Analysis of Flexible Four-Bar Linkages," *ASME Trends and Developments in Mechanisms, Machines, and Robotics*, ASMEDE-Vol. 15-2, pp. 441-447.
- [98] Yang, Z., and Sadler, J.P., 1990, "Large Displacement Finite Element Analysis of flexible Linkages," *ASME Journal of Mechanical Design*, Vol. 112, pp. 175-182.

- [99] Yang, Z., and Sadler, J.P., 1990, "Finite Element Modeling of Spatial Robot Manipulators," *ASME 21st Mechanisms, Machines, and Robots Conference Proceedings*, September, Chicago, Illinois.
- [100] Zienkiewicz, D.C., 1971, *The Finite Element Method in Engineering Science*, McGraw-Hill, London.

APPENDIX A. TRANSFORMATION MATRICES

If $(\hat{\mathbf{i}}_1, \hat{\mathbf{j}}_1, \hat{\mathbf{k}}_1)$ represents a Cartesian coordinate system, and $(\hat{\mathbf{i}}_2, \hat{\mathbf{j}}_2, \hat{\mathbf{k}}_2)$ is another Cartesian coordinate system after rotating an arbitrary angle, Θ , about one of the axes of the $(\hat{\mathbf{i}}_1, \hat{\mathbf{j}}_1, \hat{\mathbf{k}}_1)$ system, the following three rotational transformation matrices are defined

$$\mathbf{T}_{12}(\Theta_i) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\Theta_i & -\sin\Theta_i \\ 0 & \sin\Theta_i & \cos\Theta_i \end{bmatrix} \quad (\text{A-1})$$

$$\mathbf{T}_{12}(\Theta_j) = \begin{bmatrix} \cos\Theta_j & 0 & \sin\Theta_j \\ 0 & 1 & 0 \\ -\sin\Theta_j & 0 & \cos\Theta_j \end{bmatrix} \quad (\text{A-2})$$

$$\mathbf{T}_{12}(\Theta_k) = \begin{bmatrix} \cos\Theta_k & -\sin\Theta_k & 0 \\ \sin\Theta_k & \cos\Theta_k & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad (\text{A-3})$$

where Θ_i , Θ_j , Θ_k are the rotating angles about the $\hat{\mathbf{i}}_1$, $\hat{\mathbf{j}}_1$, and $\hat{\mathbf{k}}_1$ axes respectively.

The transformation matrices encountered in Chapters 3 and 4 are defined as

$$\mathbf{T}_{eo} = \begin{bmatrix} \cos\lambda_3 & -\sin\lambda_3 & 0 \\ \sin\lambda_3 & \cos\lambda_3 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos\lambda_1 & 0 & \sin\lambda_1 \\ 0 & 1 & 0 \\ -\sin\lambda_1 & 0 & \cos\lambda_1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\lambda_2 & -\sin\lambda_2 \\ 0 & \sin\lambda_2 & \cos\lambda_2 \end{bmatrix} \quad (\text{A-4})$$

$$\mathbf{T}_{o1} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad (\text{A-5})$$

$$\mathbf{T}_{o5} = \begin{bmatrix} -1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \quad (\text{A-6})$$

$$\mathbf{T}_{o34} = \begin{bmatrix} 0 & 0 & 1 \\ 0 & 1 & 0 \\ -1 & 0 & 0 \end{bmatrix} \quad (\text{A-7})$$

$$\mathbf{T}_{o78} = \begin{bmatrix} 0 & 0 & -1 \\ 0 & -1 & 0 \\ -1 & 0 & 0 \end{bmatrix} \quad (\text{A-8})$$

$$\mathbf{T}_{342} = \begin{bmatrix} 1 & 0 & \phi_{3my} \\ 0 & 1 & -\theta_2 \\ -\phi_{3my} & \theta_2 & 1 \end{bmatrix} \quad (\text{A-9})$$

$$\mathbf{T}_{786} = \begin{bmatrix} 1 & 0 & \phi_{7my} \\ 0 & 1 & -\theta_6 \\ -\phi_{7my} & \theta_6 & 1 \end{bmatrix} \quad (\text{A-10})$$

where λ_1 , λ_2 , and λ_3 are three rigid body rotation angles of the universal joint, ϕ_{3my} and ϕ_{7my} are the elastic rotation angles at the distal ends of beams 3 and 7, respectively, and the twist angles, θ_2 and θ_6 , are defined as

$$\theta_2 = \frac{1}{L_2} (d_{3mz} - d_{4mz}) \quad (\text{A-11})$$

and

$$\theta_6 = \frac{1}{L_6} (d_{7mz} - d_{8mz}) \quad (\text{A-12})$$

where L_2 and L_6 are the lengths of beams 2 and 6, respectively, and d_{3mz} , d_{4mz} , d_{7mz} , and d_{8mz} are the radial elastic deflections at the distal ends of beams 3, 4, 7, and 8, respectively.

APPENDIX B. SKEW-SYMMETRIC MATRICES

B.1 Skew-Symmetric Matrix

For a given vector, $\vec{\Omega}$, a corresponding skew-symmetric matrix, $\tilde{\Omega}$, associated with the vector, $\vec{\Omega}$, is defined as

$$\tilde{\Omega} = \begin{bmatrix} 0 & -\Omega_3 & \Omega_2 \\ \Omega_3 & 0 & -\Omega_1 \\ -\Omega_2 & \Omega_1 & 0 \end{bmatrix} \quad (\text{B-1})$$

where $\Omega_i (i = 1, 2, 3)$ are the components of the vector, $\vec{\Omega}$. Analogously, a skew-symmetric matrix $[\tilde{\mathbf{b}}]$ associated with a given matrix, \mathbf{b} , is defined as

$$[\tilde{\mathbf{b}}] = \begin{bmatrix} 0 & -\mathbf{b}_3 & \mathbf{b}_2 \\ \mathbf{b}_3 & 0 & -\mathbf{b}_1 \\ -\mathbf{b}_2 & \mathbf{b}_1 & 0 \end{bmatrix} \quad (\text{B-2})$$

with

$$[\mathbf{b}] = \begin{bmatrix} \mathbf{b}_1 \\ \mathbf{b}_2 \\ \mathbf{b}_3 \end{bmatrix} \quad (\text{B-3})$$

where $\mathbf{b}_1, \mathbf{b}_2, \mathbf{b}_3$ are the row vectors.

B.2 Special Matrix Operators

For simplicity, two special matrix product operators \otimes and Γ are introduced in the present section. These special operators have a higher priority over all other matrix operations, and they are defined as

$$\begin{aligned}\mathbf{A} \otimes \Gamma(\mathbf{B}) &= [\mathbf{A}_{ij}\mathbf{B}] \text{ for each submatrix} \\ \Gamma(\mathbf{B}) \otimes \mathbf{A} &= [\mathbf{B}\mathbf{A}_{ij}] \text{ for each submatrix} \\ \Gamma^T(\mathbf{B}) &= \Gamma(\mathbf{B}^T)\end{aligned}$$

where \mathbf{A} is a partitioned matrix with the following form

$$\mathbf{A} = \begin{bmatrix} \mathbf{A}_{11} & \cdots & \cdots & \cdots & \mathbf{A}_{1n} \\ \vdots & & & & \vdots \\ \vdots & & \mathbf{A}_{ij} & & \vdots \\ \vdots & & & \ddots & \vdots \\ \mathbf{A}_{m1} & \cdots & \cdots & \cdots & \mathbf{A}_{mn} \end{bmatrix} \quad (\text{B-4})$$

and the number of columns of each \mathbf{A}_{ij} ($i = 1, \dots, m; j = 1, \dots, n$) must be equal to the number of rows of matrix \mathbf{B} so that they are compatible for matrix operations.

B.3 Matrix Partial Derivatives

If \mathbf{q} is an n -dimensional column vector, and a scalar, Φ , and an m -dimensional vector, \mathbf{a} , are functions of \mathbf{q} , the following matrix partial derivatives may be defined

$$\frac{\partial \Phi}{\partial \mathbf{q}^T} = \left\{ \frac{\partial \Phi}{\partial q_i} \right\} \quad (\text{B-5})$$

$$\frac{\partial \mathbf{a}}{\partial \mathbf{q}^T} = \left[\frac{\partial a_j}{\partial q_i} \right] \quad (i = 1, \dots, n; j = 1, \dots, m) \quad (\text{B-6})$$

where $\{\partial\Phi/\partial q_i\}$ is an n -dimensional column vector, and $[\partial a_j/\partial q_i]$ is an $n \times m$ matrix in which i determines a row and j determines a column for the matrix. The following properties of matrix partial derivatives can then be derived from the above definitions [74].

$$\frac{\partial}{\partial \mathbf{q}^T} (\mathbf{a}^T \mathbf{b}) = \frac{\partial \mathbf{a}}{\partial \mathbf{q}^T} \mathbf{b} + \frac{\partial \mathbf{b}}{\partial \mathbf{q}^T} \mathbf{a} \quad (\text{B-7})$$

$$\frac{\partial}{\partial \mathbf{q}^T} (\mathbf{C} \mathbf{q}) = \mathbf{C}^T \quad (\text{B-8})$$

$$\frac{\partial}{\partial \mathbf{q}^T} (\mathbf{q}^T \mathbf{C} \mathbf{q}) = (\mathbf{C} + \mathbf{C}^T) \mathbf{q} \quad (\text{B-9})$$

$$\frac{\partial}{\partial \mathbf{q}^T} (\Phi_1 \Phi_2) = \frac{\partial \Phi_1}{\partial \mathbf{q}^T} \Phi_2 + \Phi_1 \frac{\partial \Phi_2}{\partial \mathbf{q}^T} \quad (\text{B-10})$$

$$\frac{\partial}{\partial \mathbf{q}^T} (\mathbf{D} \mathbf{a}) = \frac{\partial \mathbf{a}}{\partial \mathbf{q}^T} \mathbf{D}^T + \left[\frac{\partial \mathbf{d}_1}{\partial \mathbf{q}^T} \mathbf{a} \quad \cdots \quad \frac{\partial \mathbf{d}_n}{\partial \mathbf{q}^T} \mathbf{a} \right] \quad (\text{B-11})$$

where $\mathbf{D} = [\mathbf{d}_1 \cdots \mathbf{d}_n]^T$, \mathbf{a} , \mathbf{b} , Φ_1 , and Φ_2 are functions of \mathbf{q} , \mathbf{C} is a constant square matrix, and $\mathbf{d}_i (i = 1, \dots, n)$ are the subvectors in \mathbf{D} .

APPENDIX C. CONSTANT MATRICES DEFINED IN CHAPTER 3

The following matrices are the constant matrices defined in Chapter 3.

$$\mathbf{Y} = \begin{bmatrix} \mathbf{I} & \mathbf{D}_1 \\ \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{D}_2 \\ \mathbf{0} & \mathbf{0} \end{bmatrix} \quad (\text{C-1})$$

$$\mathbf{Z} = \begin{bmatrix} \mathbf{0} & \mathbf{0} \\ \mathbf{I}' & \mathbf{D}_3 \\ \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{D}_4 \end{bmatrix} \quad (\text{C-2})$$

$$\mathbf{k}_1 = \begin{bmatrix} 12 & 6l & -12 & 6l \\ & 4l^2 & -6l & 2l^2 \\ & & 12 & -6l \\ \text{Sym} & & & 4l^2 \end{bmatrix} \quad (\text{C-3})$$

$$\mathbf{k}_2 = \begin{bmatrix} 12 & -6l & -12 & -6l \\ & 4l^2 & 6l & 2l^2 \\ & & 12 & 6l \\ \text{Sym} & & & 4l^2 \end{bmatrix} \quad (\text{C-4})$$

$$[\beta_1] = \begin{bmatrix} \mathbf{I} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{I} & \mathbf{0} \end{bmatrix} \quad (\text{C-5})$$

$$[\beta_2] = \begin{bmatrix} \mathbf{0} & \mathbf{I} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{I} \end{bmatrix} \quad (\text{C-6})$$

$$\mathbf{D}_1 = \begin{bmatrix} -\frac{3}{l^2} & \frac{2}{l^3} \\ \frac{2}{l} & \frac{1}{l^2} \end{bmatrix} \quad (\text{C-7})$$

$$\mathbf{D}_2 = \begin{bmatrix} \frac{3}{l^2} & -\frac{2}{l^3} \\ -\frac{1}{l} & \frac{1}{l^2} \end{bmatrix} \quad (\text{C-8})$$

$$\mathbf{D}_3 = \begin{bmatrix} -\frac{3}{l^2} & \frac{2}{l^3} \\ \frac{2}{l} & -\frac{1}{l^2} \end{bmatrix} \quad (\text{C-9})$$

$$\mathbf{D}_4 = \begin{bmatrix} \frac{3}{l^2} & -\frac{2}{l^3} \\ \frac{1}{l} & -\frac{1}{l^2} \end{bmatrix} \quad (\text{C-10})$$

$$\mathbf{I} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \quad (\text{C-11})$$

$$\mathbf{I}' = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \quad (\text{C-12})$$

$$\mathbf{0} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix} \quad (\text{C-13})$$

where l is the length of an element, \mathbf{Y} and \mathbf{Z} are the coefficient matrices defined in the displacement functions, \mathbf{k}_1 and \mathbf{k}_2 are the stiffness matrices, $[\beta_1]$ and $[\beta_2]$ are the coefficient matrices defined in the structural stiffness matrix, \mathbf{D}_1 , \mathbf{D}_2 , \mathbf{D}_3 , \mathbf{D}_4 , \mathbf{I} , \mathbf{I}' , and $\mathbf{0}$ are the submatrices in matrices \mathbf{Y} and \mathbf{Z} .

APPENDIX D. COMPUTER SIMULATION PROGRAM

```

# This is an input data file. Every arithmetic value
# must be in D12.5 format. The length of each variable
# name must be less than or equal to 5. All data are in
# metric units, i.e., length-m, mass-kg, time-sec.
#
# equal sign "=" must
# be in this column
#
# |
# | value comments
# | |
# \// \// \//
# | | |
#
NSTEP = 1999 <-- NSTEP is an integer(<=4 digits)
NITER = 1 <-- NITER is an integer(<=2 digits)
G = 0.98000D+01
OMAGo = 0.30000D+02 <-- constant spin = 30 rpm
BETA = 0.30250D+00 <-- 0.25000D+00
GAMA = 0.60000D+00 <-- 0.50000D+00
DELT = 0.10000D-02 <-- NSTEP and DELT determine total time
To = 0.50000D+00 <-- constant spin velocity after 0.5 seconds
TDROP = 0.10000D+02 <-- collar is dropped at 10 seconds
Lus = 0.12700D+00 <-- Lus = 0.370 for the longer shaft
Es = 0.21000D+12 <-- for steel(0.9D+90 for rigid model)
RHO = 0.78330D+04 <-- for steel
A1 = 0.72000D-04 <-- 6 x 12 (mm x mm)
A5 = 0.72000D-04
A3 = 0.28300D-04 <-- D = 6.0 mm
A4 = 0.28300D-04
A7 = 0.28300D-04
A8 = 0.28300D-04
L1 = 0.16500D+00
L5 = 0.16500D+00
L2 = 0.10400D+00
L6 = 0.10400D+00
L3 = 0.29000D+00 <-- L3 = 0.265 originally
L4 = 0.29000D+00

```



```

L7    = 0.29000D+00
L8    = 0.29000D+00
Jy1   = 0.21600D-09    <-- Jy1,5 = 12*6**3/12
Jy5   = 0.21600D-09
Jy3   = 0.63600D-10    <-- Jy3,4,7,8 = PI*D**4/64
Jy4   = 0.63600D-10
Jy7   = 0.63600D-10
Jy8   = 0.63600D-10
Jz1   = 0.86400D-09    <-- Jz1,5 = 6*12**3/12
Jz5   = 0.86400D-09
Jz3   = 0.63600D-10    <-- Jz3,4,7,8 = PI*D**4/64
Jz4   = 0.63600D-10
Jz7   = 0.63600D-10
Jz8   = 0.63600D-10
La    = 0.29000D+00    <-- La = L3 or L7
Rt    = 0.90000D-01    <-- distance from arm to tank center
LMAS1 = 0.10770D+01    <-- liquid mass in tank 1
LMAS2 = 0.10770D+01    <-- liquid mass in tank 2
ENDFILE

```

```

#
# This is a MAKEFILE
# -- Jiechi Xu, Iowa State University, 9/9, 91
#
DIRO = /home/jiechi/newone/program
DIR1 = /home/jiechi/newone/lowercase/OBJFILES
#
FILES = Makefile DATA.DAT xumain.f input.f output.f\
        $(DIRO)/matrix.f    $(DIRO)/shaft.f    $(DIRO)/rbeam.f\
        $(DIRO)/fbeam.f    $(DIRO)/tank.f    $(DIRO)/stiff.f\
        $(DIRO)/assembly.f $(DIRO)/force.f    $(DIRO)/lib.f\
        $(DIRO)/mapping.f  $(DIRO)/partition.f $(DIRO)/imphase.f\
        $(DIRO)/exphase.f  $(DIRO)/corrector.f $(DIRO)/compat.f\
        $(DIRO)/postpro.f  $(DIRO)/gforout.f  $(DIRO)/xuwrite.f\
        $(DIRO)/xuoption.f $(DIRO)/xuinit.f
#
PRINTS = Makefile DATA.DAT
#
FFLAG1 = -c -u -C -fpe0 -g3 -O3 -check unde -static #final options
FFLAG2 = -c -u -C -fpe0 -g2 -O0 -check unde -static #for debugging
FFLAG3 = -c -u -C -fep0 -g3 -O4 -V -std -bestGnum -assume recursive\
        -assume noaccuracy_sensitive -check underflow -show code\
        -show include -show xref -static             #initial options
#
INCLUDE1 = $(DIRO)/parameter1.f
INCLUDE2 = $(DIRO)/parameter2.f
INCLUDE3 = $(DIRO)/parameter3.f
INCLUDE4 = $(DIRO)/parameter4.f
#
OBJECTS = $(DIR1)/xumain.o    $(DIR1)/matrix.o    $(DIR1)/shaft.o\
          $(DIR1)/rbeam.o    $(DIR1)/fbeam.o    $(DIR1)/tank.o\
          $(DIR1)/stiff.o    $(DIR1)/assembly.o $(DIR1)/force.o\
          $(DIR1)/lib.o      $(DIR1)/mapping.o   $(DIR1)/partition.o\
          $(DIR1)/imphase.o  $(DIR1)/exphase.o   $(DIR1)/corrector.o\
          $(DIR1)/input.o    $(DIR1)/output.o    $(DIR1)/compat.o\
          $(DIR1)/postpro.o  $(DIR1)/gforout.o   $(DIR1)/xuwrite.o\
          $(DIR1)/xuoption.o $(DIR1)/xuinit.o
#
# lib.f includes 4 subroutines: equation, gauss, solve, and couple.
#
# <== ***** ==>
dir:
    @mkdir OBJFILES

list: $(FILES) # to list recently changed files in subdir "program"
    @ls -l $?;ls -l

type: $(PRINTS) # to type recently changed "Makefile"

```

```

                                and/or "DATA.DAT" file(s) on the screen
@cat $?;ls -l

cleanup:
    @rm $(DIR1)/*.o;ls -l $(DIR1)/*.o;ls -l
# <== ***** ==>
xumain.out: $(OBJECTS)
    f77 $(OBJECTS) -o xumain.out
    @echo
    @echo    ===  Compilation is complete, dude!  ===
    @echo
    @echo    ===          What is next, Bart?          ===
    @echo
    @echo    ===          Aho, do not answer it!          ===
    @echo
    @echo    === The execute file is "xumain.out" ===
    @echo
    ls -l

$(DIR1)/xumain.o: xumain.f $(INCLUDE1) $(INCLUDE2)\
                  $(INCLUDE3) $(INCLUDE4)
    f77 $(FFLAG1) xumain.f -o $(DIR1)/xumain.o

$(DIR1)/xuwrite.o: $(DIR0)/xuwrite.f $(INCLUDE2)
    f77 $(FFLAG1) $(DIR0)/xuwrite.f -o $(DIR1)/xuwrite.o

$(DIR1)/xuoption.o: $(DIR0)/xuoption.f
    f77 $(FFLAG1) $(DIR0)/xuoption.f -o $(DIR1)/xuoption.o

$(DIR1)/xuinit.o: $(DIR0)/xuinit.f $(INCLUDE1) $(INCLUDE2)\
                  $(INCLUDE4)
    f77 $(FFLAG1) $(DIR0)/xuinit.f -o $(DIR1)/xuinit.o

$(DIR1)/input.o: input.f
    f77 $(FFLAG1) input.f -o $(DIR1)/input.o

$(DIR1)/output.o: output.f $(INCLUDE1) $(INCLUDE2)
    f77 $(FFLAG1) output.f -o $(DIR1)/output.o

$(DIR1)/compat.o: $(DIR0)/compat.f $(INCLUDE1) $(INCLUDE2)
    f77 $(FFLAG1) $(DIR0)/compat.f -o $(DIR1)/compat.o

$(DIR1)/matrix.o: $(DIR0)/matrix.f $(INCLUDE1) $(INCLUDE2)
    f77 $(FFLAG1) $(DIR0)/matrix.f -o $(DIR1)/matrix.o

$(DIR1)/shaft.o: $(DIR0)/shaft.f
    f77 $(FFLAG1) $(DIR0)/shaft.f -o $(DIR1)/shaft.o

```

```

$(DIR1)/rbeam.o: $(DIRO)/rbeam.f
    f77 $(FFLAG1) $(DIRO)/rbeam.f -o $(DIR1)/rbeam.o

$(DIR1)/fbeam.o: $(DIRO)/fbeam.f
    f77 $(FFLAG1) $(DIRO)/fbeam.f -o $(DIR1)/fbeam.o

$(DIR1)/tank.o: $(DIRO)/tank.f
    f77 $(FFLAG1) $(DIRO)/tank.f -o $(DIR1)/tank.o

$(DIR1)/stiff.o: $(DIRO)/stiff.f
    f77 $(FFLAG1) $(DIRO)/stiff.f -o $(DIR1)/stiff.o

$(DIR1)/assembly.o: $(DIRO)/assembly.f $(INCLUDE1) $(INCLUDE2)
    f77 $(FFLAG1) $(DIRO)/assembly.f -o $(DIR1)/assembly.o

$(DIR1)/force.o: $(DIRO)/force.f $(INCLUDE1) $(INCLUDE2)
    f77 $(FFLAG1) $(DIRO)/force.f -o $(DIR1)/force.o

$(DIR1)/mapping.o: $(DIRO)/mapping.f $(INCLUDE1) $(INCLUDE4)
    f77 $(FFLAG1) $(DIRO)/mapping.f -o $(DIR1)/mapping.o

$(DIR1)/partition.o: $(DIRO)/partition.f\
    $(INCLUDE1) $(INCLUDE3) $(INCLUDE4)
    f77 $(FFLAG1) $(DIRO)/partition.f -o $(DIR1)/partition.o

$(DIR1)/imphase.o: $(DIRO)/imphase.f\
    $(INCLUDE1) $(INCLUDE3) $(INCLUDE4)
    f77 $(FFLAG1) $(DIRO)/imphase.f -o $(DIR1)/imphase.o

$(DIR1)/exphase.o: $(DIRO)/exphase.f\
    $(INCLUDE1) $(INCLUDE3) $(INCLUDE4)
    f77 $(FFLAG1) $(DIRO)/exphase.f -o $(DIR1)/exphase.o

$(DIR1)/corrector.o: $(DIRO)/corrector.f\
    $(INCLUDE1) $(INCLUDE3) $(INCLUDE4)
    f77 $(FFLAG1) $(DIRO)/corrector.f -o $(DIR1)/corrector.o

$(DIR1)/lib.o: $(DIRO)/lib.f $(INCLUDE1) $(INCLUDE3)
    f77 $(FFLAG1) $(DIRO)/lib.f -o $(DIR1)/lib.o

$(DIR1)/postpro.o: $(DIRO)/postpro.f
    f77 $(FFLAG1) $(DIRO)/postpro.f -o $(DIR1)/postpro.o

$(DIR1)/gforout.o: $(DIRO)/gforout.f
    f77 $(FFLAG1) $(DIRO)/gforout.f -o $(DIR1)/gforout.o

```

```

      PROGRAM XUMAIN

C%
C*****
C      A FORTRAN PROGRAM TO SOLVE SECOND ORDER          *
C      NON-LINEAR IRREGULAR DIFFERENTIAL EQUATIONS      *
C      GOVERNING THE MOTION OF SATELLITE SIMULATOR      *
C      DESCRITIZATION ----- FINITE ELEMENT METHOD      *
C      NUMERICAL TECHNIQUE --- NEWMARK BETA METHOD        *
C      COORDINATE REDUCTION -- COMPONENT MODE SYTHESIS  *
C      NUMBER OF ELEMENT PER BEAM : ONE,TEN              *
C*****
C%
      INCLUDE '/home/jiechi/newone/program/parameter1.f'
      INCLUDE '/home/jiechi/newone/program/parameter2.f'
      INCLUDE '/home/jiechi/newone/program/parameter3.f'
      INCLUDE '/home/jiechi/newone/program/parameter4.f'

C%
      INTEGER NSTEP,NITER,NDROP,I,J,I1,NXU,NXUF,NXUI,NXUS

C%
      DOUBLE PRECISION L1,L2,L3,L4,L5,L6
      + ,T01(3,3),T03(3,3),T04(3,3),T05(3,3)
      + ,q9(11),q9d(11),q9dd(11),q10(11),q10d(11),q10dd(11)
      + ,POS9(3),VEL9(3),ACC9(3),POS10(3),VEL10(3),ACC10(3)
      + ,FOR9(3),TOR9(3),FOR10(3),TOR10(3),GFOR9(N),GFOR10(N)
      + ,PHI6(7,N),PHI7(NG7,N),PHI8(NG8,N),PHI9(11,N),PHI10(11,N)
      + ,q(N),qd(N),qdd(N),M(N,N),MD(N,N),MP(N,N),K(N,N),f(N),X(3)
      + ,T(0:2000),POS(N+6,0:2000),VEL(N+6,0:2000),ACC(N+6,0:2000)
      + ,G,PI,BETA,GAMA,DELT,Ang,W,W1,W2
      + ,DELTA(3),ICFD(6),TIME!,LMAS1,LMAS2

C%
      CHARACTER*7 INTER,APPR,SPIN,CHANGE

C%
      REAL Ti,Tf

C%
      COMMON/COM3/G      !<-- input(from input)
      COMMON/COMNM/BETA,GAMA,DELT !<-- input(from input)
      COMMON/COM6/NSTEP,NITER,NDROP !<-- input(from input)
      COMMON/COML/L1,L2,L3,L4,L5,L6 !<-- input(from input)
      COMMON/COMW1/W1 !<-- input(from xuinit)
      COMMON/COMQ/q,qd,qdd !<-- input(from xuinit)
      COMMON/COMMKF/M,MD,MP,K,f !<-- input(from xuinit)
      COMMON/COMPVA/POS,VEL,ACC !<-- input(from xuinit)
      COMMON/COMT/T01,T03,T04,T05 !<-- input(from xuinit)
      COMMON/COMOP/APPR,SPIN,INTER !<-- input(from xuoption)
      COMMON/COMPHI2/PHI6,PHI7,PHI8,PHI9,PHI10 !<-- input(from compat)
      COMMON/COM11/DELTA,ICFD !<-- output(to tank)
      COMMON/GFORCE/GFOR9,GFOR10 !<-- output(to force)

```

```

COMMON/COMNXU/NXU,NXUS                                !<-- output(to xuinit,xuwrite)
C%
DATA NXU,NXUF,NXUI,NXUS/96,97,98,99/
DATA Ang,T(0)/2*0.0D0/
C%
CLOSE(NXUF)
CLOSE(NXUI)
CLOSE(NXUS)
OPEN(UNIT=NXUF,FILE='FORCE.dat',STATUS='OLD')
OPEN(UNIT=NXUI,FILE='INERT.dat',STATUS='OLD')
OPEN(UNIT=NXUS,FILE='CABAL.dat',STATUS='OLD')
C%
PI=4.0D0*DATAN(1.0D0)
C%
C... initialization .....
CALL XUWRITE()      !<-- write out 1st information file  !
CALL XUINPUT()      !<-- read input data file            !
CALL XUOPTION()     !<-- choose options                  !
CALL XUINIT()       !<-- initial assignment and computing !
C.....!
C%
WRITE(*,*)'I am ready. Are you?'
WRITE(*,*)
WRITE(*,*)'Your choices are:'
WRITE(*,*)'1) Hit <ret> to run out of Ames.          ** OR **'
WRITE(*,*)'2) Type "quit" to crash the computer'
READ(*, '(A)')CHANGE
IF(CHANGE.EQ.'quit'.OR.CHANGE.EQ.'q')THEN
WRITE(*,*)
WRITE(*,*)'HELP!!! Your computer is dea.....d'
WRITE(*,*)
CLOSE(NXU)
OPEN(UNIT=NXU,FILE='z_read_xu_first',STATUS='OLD')
WRITE(NXU,*)'Your computer is crashed. No results'
STOP
END IF
WRITE(*,*)
WRITE(*,*)'That"s it *MAN*. (You *grin* evilly)'
WRITE(*,*)
C%
Ti=SECNDS(0.0)                                !<-- start tracking running time
C%
close(15)
C*** execute forward time step loop *****!
DO 120 J=1,NSTEP                                !<-- number of time steps  !*
T(J)=FLOAT(J)*DELT                                !*
C::: numerical integration :: <Newmark Method> :: !*
CALL PARTITION(M,MD,MP,K,f,q,qd)                !*

```

```

      CALL IMPHASE(M,MD,K,f,q,qd,qdd)      ! find d's      !*
      CALL EXPHASE(T(J),q,qd,qdd)          ! find Lamd's    !*
C:.....                                     !*
C--- renumber q,qd,qdd -----             !*
      DO 85 I1=1,N-ITH                      ! from ITH+1 to N !*
      q(N+1-I1)=q(N-I1)                     !                !*
      qd(N+1-I1)=qd(N-I1)                   !                !*
      qdd(N+1-I1)=qdd(N-I1)                 !                !*
85    CONTINUE                             !                !*
C-----                                     !*
C== input spin profile =====             !*
      IF(SPIN.EQ.'1')THEN                   !                !*
C+++ option (1) calculated +++++           !                !*
      CALL LAMD(T(J),X)                     !                !*
      q(ITH)=X(1)                           !                !*
      qd(ITH)=X(2)                          !                !*
      qdd(ITH)=X(3)                         ! <-- cheat?    !*
C+++++                                     !                !*
      ELSE                                  !                !*
C--- option (2) measured -----            !                !*
      READ(NXUS,*)T(J),W                    !                !*
      READ(NXUS,*)T(J),W2                  !                !*
      qd(ITH)=W*PI/30.0                    !                !*
      Ang=Ang+0.5*DELT                     !                !*
      + *(W1+qd(ITH))                      !                !*
      q(ITH)=Ang                           !                !*
      W2=W2*PI/30.0                        !                !*
      qdd(ITH)=0.5                         !                !*
      + *(W2-W1)/DELT                      ! <-- trouble  !*
      W1=qd(ITH)                           !                !*
      IF(MOD(J,2).EQ.1)                   !                !*
      + BACKSPACE(NXUS)                   !                !*
      BACKSPACE(NXUS)                     !                !*
C-----                                     !                !*
      END IF                               !                !*
C=====                                     !*
C??? determine up or down ??????          !*
      IF(J.LE.NDROP)THEN                   ! from 1 to ITH-1 !*
      DO 90 I1=1,ITH-1                     !                !*
      q(I1)=0.0                             !                !*
      qd(I1)=0.0                           !                !*
      qdd(I1)=0.0                          !                !*
90    CONTINUE                             !                !*
      END IF                               !                !*
C????????????????????????????????????    !*
C--- find kinematic properties -----      !*
      CALL COUPLE(q9,PHI9,q,11,N,1)        !                !*
      CALL COUPLE(q9d,PHI9,qd,11,N,1)      !                !*

```

```

      CALL COUPLE(q9dd,PHI9,qdd,11,N,1)      !      !*
      CALL COUPLE(q10,PHI10,q,11,N,1)      !      !*
      CALL COUPLE(q10d,PHI10,qd,11,N,1)    !      !*
      CALL COUPLE(q10dd,PHI10,qdd,11,N,1)  !      !*
      CALL POSTPRO(L1,L2,q9,q9d,q9dd,T01   !      !*
+      ,POS9,VEL9,ACC9)                    !      !*
      CALL POSTPRO(L5,L6,q10,q10d,q10dd,T05 !      !*
+      ,POS10,VEL10,ACC10)                 !      !*
C-----
C=== find local and global quantities =====
      DO 100 I1=1,N                        !      !*
      POS(I1,J)=q(I1)                     !      !*
      VEL(I1,J)=qd(I1)                    !      !*
      ACC(I1,J)=qdd(I1)                   !      !*
100    CONTINUE                           !      !*
      DO 105 I1=1,3                       !      !*
      POS(N+I1,J)=POS9(I1)                !      !*
      VEL(N+I1,J)=VEL9(I1)                 !      !*
      ACC(N+I1,J)=ACC9(I1)                 !      !*
      POS(N+3+I1,J)=POS10(I1)              !      !*
      VEL(N+3+I1,J)=VEL10(I1)              !      !*
      ACC(N+3+I1,J)=ACC10(I1)              !      !*
105    CONTINUE                           !      !*
C=====
C... select interaction option .....
      IF(INTER.EQ.'YES')THEN              !      !*
C... select approach(inertia force or matrix) .....
      IF(APPR.EQ.'2')THEN                  !      !*
C### read CFD inertia active forces #####
      READ(NXUF,*,END=199)TIME,FOR9(3)    !      !*
+      ,FOR9(2),FOR9(1)                    !      !*
      FOR9(1)=-FOR9(1)                     !      !*
      DO 110 I=1,3                         !      !*
      FOR10(I)=FOR9(I)                     !      !*
      TOR9(I)=0.0D0                        !      !*
      TOR10(I)=0.0D0                       !      !*
110    CONTINUE                           !      !*
C#####
C^^^ find generalized active forces ~~~~~
      CALL GFOROUT(L2,q9,POS9,T01,PHI9     !      !*
+      ,FOR9,TOR9,GFOR9)                   !      !*
      CALL GFOROUT(L6,q10,POS10,T05,PHI10  !      !*
+      ,FOR10,TOR10,GFOR10)               !      !*
C-----
      ELSE                                  !      !*
C... read CFD inertia matrix .....
      READ(NXUI,*,END=199)TIME,DELTA(3)    !      !*
+      ,DELTA(2),DELTA(1),ICFD(6),ICFD(5)  !      !*

```



```

      + ,ICFD(3),ICFD(4),ICFD(2),ICFD(1)      !      !      !      !*
      DELTA(1)=-DELTA(1)                      !      !      !      !*
      ICFD(2)=-ICFD(2)                      !      !      !      !*
      ICFD(3)=-ICFD(3)                      !      !      !      !*
C.....!      !      !      !*
      END IF                                !      !      !      !*
C.....!      !      !      !*
      END IF                                !      !      !      !*
C.....!      !      !      !*
      CALL MATRIX(M,MD,MP,K,f,q,qd)          !*
      CALL MAPPING(M,MD,MP,K,f,q,qd,qdd)     !*
      write(15,*)T(J),q(7)*1000.0,q(8)*1000.0 !<-- temporary !*
120    CONTINUE                             !*
C*****!
C%
199    Tf=SECNDS(Ti)                        !<-- end computing running time
C%
C... write out 2nd information file .....
      CLOSE(NXU)                             !
      OPEN(UNIT=NXU,FILE='z_read_xu_second',STATUS='UNKNOWN') !
      WRITE(NXU,*)'This is a second "read_xu" file.'      !
      WRITE(NXU,*)'Running time(seconds)=',Tf            !
      IF(INTER.EQ.'YES')THEN                             !
      WRITE(NXU,*)'This is an interaction run'           !
      ELSE                                                !
      WRITE(NXU,*)'This is a non-interaction run'        !
      END IF                                             !
C.....!
C... output results .....
      CALL OUTPUT(T,3,'dataP',POS,PI,NSTEP) ! <-- position
      CALL OUTPUT(T,3,'dataV',VEL,PI,NSTEP) ! <-- velocity
      CALL OUTPUT(T,3,'dataA',ACC,PI,NSTEP) ! <-- acceleration
C.....!
500    FORMAT(1X,F7.4,11(1X,D10.3))
600    FORMAT(1X,F7.4,8(1X,D10.3))
      STOP
      END
C%
C::::::::::::::::::::::::::::::::::::
C      subroutine lamd(spin profile) :
C::::::::::::::::::::::::::::::::::::
C%
      SUBROUTINE LAMD(T,X)
C      INPUT -- T
C      OUTPUT -- X
      INTEGER I
      DOUBLE PRECISION OMAGo,PI,To,T,X(3)
      COMMON/COMF/OMAGo,PI,To
      !<-- input(from input)

```

```

      DO 10 I=1,3
      X(I)=0.0
10    CONTINUE
C... specify sinusoidal function .....
      IF(T.LE.To)THEN
      X(1)=(0.5*OMAGo)*(T-DSIN((PI/To)*T)*(To/PI))
      X(2)=(0.5*OMAGo)*(1.0-DCOS((PI/To)*T))
      X(3)=(0.5*OMAGo)*(PI/To)*DSIN((PI/To)*T)
      ELSE
      X(1)=(0.5*OMAGo)*(2.0*T-To)
      X(2)=OMAGo
      X(3)=0.0
      END IF
C.....
      RETURN
      END

```

```

      SUBROUTINE XUWRITE()
C      INPUT  -- Ni1,Ni3,Ni4,Ni5,Ni7,Ni8,NXU
C      OUTPUT -- (none)
C*****
C      This subroutine is to write out information      *
C      to a file ,z_read_xu_first, about the final    *
C      output files which contain the computing      *
C      results.                                       *
C*****
C%
C%      INCLUDE '/home/jiechi/newone/program/parameter2.f'
C%
C%      INTEGER NXU,MOD34,MOD78,INT34,INT78
C%
C%      COMMON/COMNXU/NXU                !<-- input(from xumain)
C%
C%      MOD34=MOD((Ni1+Ni3+Ni4-1),2)
C%      MOD78=MOD((Ni5+Ni7+Ni8-1),2)
C%      INT34=INT((Ni1+Ni3+Ni4-1)/2)
C%      INT78=INT((Ni5+Ni7+Ni8-1)/2)
C%
C%      IF((INT34+INT78).GT.9)THEN
C%      WRITE(*,*)'The total number of output files for each'
C%      WRITE(*,*)'vector(position, velocity, and acceleration)'
C%      WRITE(*,*)'exceeds the allowed maximum number of "9".'
C%      WRITE(*,*)'You need to write an integer value to character'
C%      WRITE(*,*)'value converting subroutine to modify the'
C%      WRITE(*,*)'present one. You should also modify the'
C%      WRITE(*,*)'declaration of character "ADD" in subroutine'
C%      WRITE(*,*)'"OUTPUT".'
C%      WRITE(*,*)'Sorry, the program is stopped.'
C%      STOP
C%      END IF
C... write out information for output files .....
      CLOSE(NXU)
      OPEN(UNIT=NXU,FILE='z_read_xu_first',STATUS='NEW')
      WRITE(NXU,2)'1) There are ',INT34+INT78+1,' output files.'
      IF(MOD34.EQ.1.AND.MOD78.EQ.0)THEN
        WRITE(NXU,*)'2) Version "0" file contains Lamd vector',
+          ' and "d34m" vector only.'
        WRITE(NXU,2)'3) The rest files from 1 to ',INT34+INT78,
+          ' contain, in order,'
        WRITE(NXU,*)' the subvectors: d1, d3, d4, d5, d7,',
+          ' d8, and d78m.'
      END IF
      IF(MOD34.EQ.0.AND.MOD78.EQ.1)THEN
        WRITE(NXU,*)'2) Version "0" file contains Lamd vector',
+          ' and "d78m" vector only.'

```

```

        WRITE(NXU,2)'3) The rest files from 1 to ',INT34+INT78,
+         ' contain, in order,'
        WRITE(NXU,*)'   the subvectors: d1, d3, d4, d34m,',
+         ' d5, d7, and d8.'
    END IF
    IF(MOD34.EQ.1.AND.MOD78.EQ.1)THEN
        WRITE(NXU,*)'2) Version "0" file contains Lamd vector,',
+         ' d34m, and d78m vectors.'
        WRITE(NXU,2)'3) The rest files from 1 to ',INT34+INT78,
+         ' contain, in order,'
        WRITE(NXU,*)'   the subvectors: d1, d3, d4, d5, d7,',
+         ' and d8.'
    END IF
    IF(MOD34.EQ.0.AND.MOD78.EQ.0)THEN
        WRITE(NXU,*)'2) Version "0" file contains Lamd vector.'
        WRITE(NXU,2)'3) The rest files from 1 to ',INT34+INT78,
+         ' contain, in order,'
        WRITE(NXU,*)'   the subvectors: d1, d3, d4, d34m,',
+         ' d5, d7, d8, and d78m.'
    END IF
    WRITE(NXU,*)'4) Each file, except the "0" one,',
+         ' has exact 8 components.'
2    FORMAT(1X,A,I2,A)
C.....
    RETURN
    END

```

```

      SUBROUTINE XUOPTION()
C      INPUT  -- (key board input)
C      OUTPUT -- INTER,APPR,SPIN
C*****
C      This is a subroutine to merely choose  *
C      computing options which are:          *
C          1) interaction option              *
C          2) approach option                 *
C          3) spin profile option             *
C*****
C%
C      CHARACTER*7 APPR,SPIN,INTER
C%
C      COMMON/COMOP/APPR,SPIN,INTER          !<-- output(to xumain,xuinit)
C%
C... choose interaction & approach options .....
      INTER='NO'
      WRITE(*,*)'Do interaction?(<ret>=N)'
      READ(*,'(A)')INTER
      IF(INTER.EQ.'YES'.OR.INTER.EQ.'Y'.OR.
+      INTER.EQ.'yes'.OR.INTER.EQ.'y')THEN
      INTER='YES'
      WRITE(*,*)'Which approach?(1 or 2)'
      WRITE(*,*)'1) inertia matrix approach'
      WRITE(*,*)'2) inertia forces approach'
      READ(*,'(A)')APPR
      END IF
C.....!
C... choose spin profile option .....
      WRITE(*,*)'Which spin profile?(1 or 2)'
      WRITE(*,*)'1) calculated spin profile'
      WRITE(*,*)'2) measured spin profile'
      READ(*,'(A)')SPIN
C.....!
      RETURN
      END

```

```

      SUBROUTINE XUINIT()
C      INPUT  -- (COMMON and INCLUDE statements)
C      OUTPUT -- (COMMON statement)
C*****
C      The functions of this subroutine are to specify      *
C      and compute initial conditions, which include:      *
C          1) clear zeros                                  *
C          2) initial assignments                          *
C          3) initial calculations                         *
C*****
C%
      INCLUDE '/home/jiechi/newone/program/parameter1.f' !(define N)
      INCLUDE '/home/jiechi/newone/program/parameter2.f' !(define N)
      INCLUDE '/home/jiechi/newone/program/parameter4.f' !(define ITH)
C%
      INTEGER I,J,NXU,NXUS
C%
      DOUBLE PRECISION PI,G,Lus,L1,L2,L3,L4,L5,L6
      + ,LMAS1,LMAS2,DELTA(3),ICFD(6),TIME,W1
      + ,R1(3),R3(3),R4(3),R5(3),R7(3),R8(3)
      + ,T01(3,3),T03(3,3),T04(3,3),T034(3,3)
      + ,T05(3,3),T07(3,3),T08(3,3),T078(3,3)
      + ,POS(N+6,0:2000),VEL(N+6,0:2000),ACC(N+6,0:2000)
      + ,POS9(3),VEL9(3),ACC9(3),POS10(3),VEL10(3),ACC10(3)
      + ,q9(11),q9d(11),q9dd(11),q10(11),q10d(11),q10dd(11)
      + ,q(N),qd(N),qdd(N),M(N,N),MD(N,N),MP(N,N),K(N,N),f(N)
      + ,FOR9(3),TOR9(3),FOR10(3),TOR10(3),GFOR9(N),GFOR10(N)
      + ,PHI6(7,N),PHI7(NG7,N),PHI8(NG8,N),PHI9(11,N),PHI10(11,N)
C%
      + ,X(3),d(N-ITH)          !<-- watch here
C%
      CHARACTER*7 CHANGE,APPR,SPIN
C%
      COMMON/COM3/G              !<-- input(from input)
      COMMON/COM4/Lus            !<-- input(from input)
      COMMON/COMLMAS/LMAS1,LMAS2 !<-- input(input)output(matrix)
      COMMON/COML/L1,L2,L3,L4,L5,L6 !<-- input(from input)
      COMMON/COMNXU/NXU,NXUS     !<-- input(from xumain)
      COMMON/COMOP/APPR,SPIN     !<-- input(from xuoption)
      COMMON/COM11/DELTA,ICFD    !<-- output(to tank)
      COMMON/COMW1/W1            !<-- output(to xumain)
      COMMON/COMQ/q,qd,qdd       !<-- output(to xumain)
      COMMON/COMMKF/M,MD,MP,K,f  !<-- output(to xumain)
      COMMON/COMPVA/POS,VEL,ACC  !<-- output(to xumain)
      COMMON/GFORCE/GFOR9,GFOR10 !<-- output(to force)
      COMMON/COMR/R1,R3,R4,R5,R7,R8 !<-- output
      COMMON/COMT/T01,T03,T04,T05,T07,T08,T034,T078 !<-- output
C%

```

```

      PI=4.0D0*DATAN(1.0D0)
C%
C... clear zero start .....
      DO 10 I=1,3      !
      R1(I)=0.0        !
      R3(I)=0.0        !
      R4(I)=0.0        !
      R5(I)=0.0        !
      R7(I)=0.0        !
      R8(I)=0.0        !
      FOR9(I)=0.0      !
      TOR9(I)=0.0      !
      FOR10(I)=0.0     !
      TOR10(I)=0.0     !
      DO 10 J=1,3      !
      T01(I,J)=0.0     !
      T03(I,J)=0.0     !
      T04(I,J)=0.0     !
      T05(I,J)=0.0     !
      T07(I,J)=0.0     !
      T08(I,J)=0.0     !
      T034(I,J)=0.0    !
      T078(I,J)=0.0    !
10    CONTINUE        !
      DO 15 J=1,N      !
      q(J)=0.0         !
      qd(J)=0.0        !
      qdd(J)=0.0       !
      GFOR9(J)=0.0D0   !
      GFOR10(J)=0.0D0  !
15    CONTINUE        !
C... clear zero end .....!
C... assign initial values .....
      T01(1,1)=1.0     !
      T01(2,2)=1.0     !
      T01(3,3)=1.0     !
      T03(1,3)=1.0     !
      T03(2,2)=1.0     !
      T03(3,1)=-1.0    !
      T05(1,1)=-1.0    !
      T05(2,2)=-1.0    !
      T05(3,3)=1.0     !
      T07(1,3)=-1.0    !
      T07(2,2)=-1.0    !
      T07(3,1)=-1.0    !
      DO 20 J=1,3      !
      DO 20 I=1,3      !
      T04(I,J)=T03(I,J) !

```

```

      T08(I,J)=T07(I,J)      !
      T034(I,J)=T03(I,J)    !
      T078(I,J)=T07(I,J)    !
20    CONTINUE              !
      CALL COMPAT()          !<--- compatibility matrix subroutine
      R1(1)=0.0              !
      R1(2)=0.0              !
      R1(3)=Lus              !
      R3(1)=L1               !
      R3(2)=L2/2.0           !
      R3(3)=Lus              !
      R4(1)=L1               !
      R4(2)=-L2/2.0          !
      R4(3)=Lus              !
      R5(1)=0.0              !
      R5(2)=0.0              !
      R5(3)=Lus              !
      R7(1)=-L1              !
      R7(2)=L2/2.0           !
      R7(3)=Lus              !
      R8(1)=-L1              !
      R8(2)=-L2/2.0          !
      R8(3)=Lus              !
C.....!
C... assign initial q, qd .....
c      q(1)=PI/180.D0        ! 1 degree offset
c      q(6)=-6.696D-04       !
c      q(7)= 7.688D-03       !
c      q(10)=q(6)            ! It seems it's not a
c      q(11)=q(7)            ! good approach to give
c      q(13)=-1.090D-03      ! initial conditions for
c      q(14)= 7.264D-03      ! both q and qd.
c      q(15)=q(13)           !
c      q(17)=q(13)           !
c      q(18)=q(14)           !
c      q(19)=q(13)           !
C.....!
C... change initial values for q(1) and/or q(2) .....
      WRITE(*,*)'Default initial conditions:'      !
      WRITE(*,*)'q1 = 0, q2 = 0'                    !
      CHANGE='NO'                                    !
      WRITE(*,*)'Need to change?(<ret>=N)'          !
      READ(*, '(A)')CHANGE                          !
      IF(CHANGE.EQ. 'YES'.OR.CHANGE.EQ. 'Y'.OR.
+      CHANGE.EQ. 'yes'.OR.CHANGE.EQ. 'y')THEN      !
50    WRITE(*,*)'Which one?'                        !
      WRITE(*,*)'(1 = q1; 2 = q2; 3 = both)'        !
      READ(*, '(A)')CHANGE                          !

```



```

      IF(CHANGE.EQ.'3')THEN
      WRITE(*,*)'Input the values of q1 and q2(degrees)'
      READ(*,*)q(1),q(2)
      q(1)=q(1)*PI/180.DO
      q(2)=q(2)*PI/180.DO
      ELSE
      IF(CHANGE.EQ.'1')THEN
      WRITE(*,*)'Input the value of q1(degree)'
      READ(*,*)q(1)
      q(1)=q(1)*PI/180.DO
      ELSE
      WRITE(*,*)'Input the value of q2(degree)'
      READ(*,*)q(2)
      q(2)=q(2)*PI/180.DO
      END IF
      END IF
      CHANGE='NO'
      WRITE(*,*)'Now the values are:'
      WRITE(*,*)'q1 =',q(1),'q2 =',q(2)
      WRITE(*,*)'Need to change again?(<ret>=N)'
      READ(*,'(A)')CHANGE
      IF(CHANGE.EQ.'YES'.OR.CHANGE.EQ.'Y'.OR.
+      CHANGE.EQ.'yes'.OR.CHANGE.EQ.'y')GOTO 50
      END IF
C.....
C%
      WRITE(*,*)
      WRITE(*,*)'Waite a second. I am working hard.'
      WRITE(*,*)
C%
C... find initial kinematic properties at tank centers .....
      CALL COUPLE(q9,PHI9,q,11,N,1)
      CALL COUPLE(q9d,PHI9,qd,11,N,1)
      CALL COUPLE(q9dd,PHI9,qdd,11,N,1)
      CALL COUPLE(q10,PHI10,q,11,N,1)
      CALL COUPLE(q10d,PHI10,qd,11,N,1)
      CALL COUPLE(q10dd,PHI10,qdd,11,N,1)
      CALL POSTPRO(L1,L2,q9,q9d,q9dd,T01,POS9,VEL9,ACC9)
      CALL POSTPRO(L5,L6,q10,q10d,q10dd,T05,POS10,VEL10,ACC10)
C.....
C... select approach(inertia force or inertia matrix) .....
      IF(APPR.EQ.'2')THEN
C... compute initial active force .....
      DO 25 I=1,3
      FOR9(I)=-LMAS1*ACC9(I)
      FOR10(I)=-LMAS2*ACC10(I)
25      CONTINUE
      FOR9(3)=FOR9(3)-LMAS1*G

```

```

FOR10(3)=FOR10(3)-LMAS2*G          ! inertia !
C.....! force !
C... compute initial generalized active force ... ! !
      CALL GFOROUT(L2,q9,POS9,T01    ! ! !
      + ,PHI9,FOR9,TOR9,GFOR9)      ! ! !
      CALL GFOROUT(L6,q10,POS10,T05  ! ! !
      + ,PHI10,FOR10,TOR10,GFOR10)  ! ! !
C.....! ! !
      LMAS1=0.0D0      !<-- tricky here ! !
      LMAS2=0.0D0      !<-- tricky here ! !
      ELSE              !<----- ! !
C... read initial inertia matrix ..... ! !
      CLOSE(NXU)        ! ! !
      OPEN(NXU,FILE='INERT0.dat',STATUS='OLD') ! inertia !
      READ(NXU,*)TIME,(DELTA(I),I=1,3) ! matrix !
      + , (ICFD(I),I=1,6) ! ! !
C.....! ! !
      END IF            !<----- ! !
C.....! ! !
C... read measured spin profile(optional) ..... !
      IF(SPIN.EQ.'2')THEN !
      READ(NXUS,*)TIME,qd(ITH) !
      qd(ITH)=qd(ITH)*PI/30.0 !
      W1=qd(ITH) !
      END IF !
C.....! ! !
C... compute initial qdd ..... !
c      J=0 !
c      DO 30 I=1,N-ITH !
c      d(I)=q(I+ITH) !
c30    CONTINUE !
32     CALL MATRIX(M,MD,MP,K,f,q,qd) !
      CALL MAPPING(M,MD,MP,K,f,q,qd,qdd) !
c      CALL PARTITION(M,MD,MP,K,f,q,qd) !
c      CALL INITIAL(K,f,q) !
c      X(1)=0.0D0 !max !
c      X(2)=0.0D0 !temp !
c      X(3)=1.0D-05 !error !
c      DO 35 I=1,N-ITH !
c      IF(q(I).EQ.0.0D0)GOTO 34 !
c      X(2)=ABS((q(I)-d(I))/q(I)) !
c34    IF(X(1).LT.X(2))X(1)=X(2) !
c35    CONTINUE !
c      DO 40 I=1,N-ITH !
c      d(I)=q(I) !
c40    CONTINUE !
c      J=J+1 !
c      DO 45 I=1,N-ITH !

```

```

c      q(N+1-I)=q(N-ITH+1-I)      !
c45    CONTINUE                    !
c      DO 50 I=1,ITH               !
c      q(I)=0.0                    !
c50    CONTINUE                    !
c      WRITE(11,500)FLOAT(J),(q(I),I=1,11) !
c      WRITE(12,600)FLOAT(J),(q(I),I=12,N) !
c      IF(J.GE.NITER)STOP          !
c      IF(X(1).GT.X(3))GOTO 32     !
c      IF(J.LT.NITER)STOP          !
c      CALL EQUATION(qdd,M,MD,MP,K,f,q,qd) !
C.....!
C... keep initial kinematic properties .....
      DO 75 I=1,N                  !
      POS(I,0)=q(I)                !
      VEL(I,0)=qd(I)               !
      ACC(I,0)=qdd(I)              !
75    CONTINUE                    !
      DO 80 I=1,3                  !
      POS(N+I,0)=POS9(I)           !
      VEL(N+I,0)=VEL9(I)           !
      ACC(N+I,0)=ACC9(I)           !
      POS(N+3+I,0)=POS10(I)        !
      VEL(N+3+I,0)=VEL10(I)        !
      ACC(N+3+I,0)=ACC10(I)        !
80    CONTINUE                    !
C.....!
      RETURN
      END

C%
C:::::::::::::::::::::::::::::
C      subroutine initial      :
C:::::::::::::::::::::::::::::
C%
      SUBROUTINE INITIAL(K,f,q)
C      INPUT  -- K,f,INCLUDE(N,ITH)
C      OUTPUT -- q
C*****
C      This is a short subroutine which performs      *
C      calculation for initial qdd                    *
C*****
C%
      INCLUDE '/home/jiechi/newone/program/parameter1.f'
      INCLUDE '/home/jiechi/newone/program/parameter4.f'
C%
      INTEGER I,J,IX(N-ITH)
C%
      DOUBLE PRECISION K(N-ITH,N-ITH),f(N-ITH)

```

```

      + ,q(N-ITH),Kout(N-ITH,N-ITH),DUM(N-ITH)
C%
      DO 10 I=1,N-ITH
        q(I)=0.0
        DUM(I)=0.0
        DO 10 J=1,N-ITH
          Kout(I,J)=0.0
10      CONTINUE
        CALL GAUSS(K,N-ITH,IX,Kout,DUM)
        CALL SOLVE(Kout,N-ITH,IX,f,q)
        RETURN
      END

```

```

SUBROUTINE XUINPUT()
C%
C      INPUT  -- DATA.DAT file
C      OUTPUT -- COMMON statements
C%
      INTEGER NSTEP,NITER,NDROP,VALUE1
C%
      DOUBLE PRECISION L1,L2,L3,L4,L5,L6,L7,L8,A1,A3,A4,A5,A7,A8
+      ,Jy1,Jy3,Jy4,Jy5,Jy7,Jy8,Jz1,Jz3,Jz4,Jz5,Jz7,Jz8,La,Rt
+      ,MASS1,MASS3,MASS4,MASS5,MASS7,MASS8,LMA1,LMA2
+      ,PI,G,OMAGo,BETA,GAMA,DELT,To,TDROP,Lus,Es,RHO,VALUE2
C%
      CHARACTER*7,CH1,CH2,CH3,CH4,CH5,CH6,CH7,CH8,CH9,CH10
+      ,CH11,CH12,CH13,CH14,CH15,CH16,CH17,CH18,CH19,CH20
+      ,CH21,CH22,CH23,CH24,CH25,CH26,CH27,CH28,CH29,CH30
+      ,CH31,CH32,CH33,CH34,CH35,CH36,CH37,CH38,CH39,CH40
+      ,CH41,CH42
+      ,mark*72,CHANGE,X
C%
      COMMON/COM3/G                                !<-- output
      COMMON/COM4/Lus                               !<-- output
      COMMON/COM5/Es                                !<-- output
      COMMON/COM7/La,Rt                             !<-- output
      COMMON/COMF/OMAGo,PI,To                       !<-- output
      COMMON/COMLMAS/LMA1,LMA2                      !<-- output
      COMMON/COMNM/BETA,GAMA,DELT                   !<-- output
      COMMON/COM6/NSTEP,NITER,NDROP                 !<-- output
      COMMON/COML/L1,L2,L3,L4,L5,L6,L7,L8           !<-- output
      COMMON/COMJy/Jy1,Jy3,Jy4,Jy5,Jy7,Jy8         !<-- output
      COMMON/COMJz/Jz1,Jz3,Jz4,Jz5,Jz7,Jz8         !<-- output
      COMMON/COMMASS/MASS1,MASS3,MASS4,MASS5,MASS7,MASS8 !<-- output
C%
      CLOSE(40)
      OPEN(UNIT=40,FILE='DATA.dat',STATUS='OLD')
C%
10      READ(40,'(A)')mark
      IF(mark(1:1).EQ.'#'.OR.mark(1:1).EQ.'!'.OR.
+      mark(1:1).EQ.'%'.OR.mark(1:1).EQ.'$')GOTO 10
      BACKSPACE(40)
      READ(40,'(A,I5)')CH1 ,NSTEP
      READ(40,'(A,I2)')CH2 ,NITER
      READ(40,100)CH3 ,G
      READ(40,100)CH4 ,OMAGo
      READ(40,100)CH5 ,BETA
      READ(40,100)CH6 ,GAMA
      READ(40,100)CH7 ,DELT
      READ(40,100)CH8 ,To
      READ(40,100)CH9 ,TDROP

```

```

READ(40,100)CH10,Lus
READ(40,100)CH11,Es
READ(40,100)CH12,RHO
READ(40,100)CH13,A1
READ(40,100)CH14,A5
READ(40,100)CH15,A3
READ(40,100)CH16,A4
READ(40,100)CH17,A7
READ(40,100)CH18,A8
READ(40,100)CH19,L1
READ(40,100)CH20,L5
READ(40,100)CH21,L2
READ(40,100)CH22,L6
READ(40,100)CH23,L3
READ(40,100)CH24,L4
READ(40,100)CH25,L7
READ(40,100)CH26,L8
READ(40,100)CH27,Jy1
READ(40,100)CH28,Jy5
READ(40,100)CH29,Jy3
READ(40,100)CH30,Jy4
READ(40,100)CH31,Jy7
READ(40,100)CH32,Jy8
READ(40,100)CH33,Jz1
READ(40,100)CH34,Jz5
READ(40,100)CH35,Jz3
READ(40,100)CH36,Jz4
READ(40,100)CH37,Jz7
READ(40,100)CH38,Jz8
READ(40,100)CH39,La
READ(40,100)CH40,Rt
READ(40,100)CH41,LMS1
READ(40,100)CH42,LMS2
WRITE(*,*)
WRITE(*,*) '<----- Echo data file ----->'
WRITE(*,*)
REWIND(40)
15 READ(40,'(A)')mark
IF(mark(1:7).NE.'endfile'.AND.mark(1:7).NE.'ENDFILE')THEN
WRITE(*,'(A)')mark
GOTO 15
END IF
WRITE(*,'(A)')mark
WRITE(*,*)
WRITE(*,*) '<----- Echo is complete ----->'
WRITE(*,*)
CHANGE='NO'
WRITE(*,*) 'Any change?(<ret>=N)'

```

```

READ(*, '(A)')CHANGE
IF(CHANGE.EQ.'YES'.OR.CHANGE.EQ.'Y'.OR.
*   CHANGE.EQ.'yes'.OR.CHANGE.EQ.'y')THEN
20  WRITE(*,*)'Which one?(type variable name)'
    READ(*, '(A)')X
    WRITE(*,*)'Please input the value'
    IF(X.EQ.'NSTEP'.OR.X.EQ.'NITER'.OR.
*       X.EQ.'nstep'.OR.X.EQ.'niter')THEN
        READ(*,*)VALUE1
        IF(X.EQ.'NSTEP'.OR.X.EQ.'nstep')NSTEP=VALUE1
        IF(X.EQ.'NITER'.OR.X.EQ.'niter')NITER=VALUE1
    ELSE
        READ(*,*)VALUE2
        IF(X.EQ.'G' .OR.X.EQ.'g' )G =VALUE2
        IF(X.EQ.'OMAGo' .OR.X.EQ.'omago')OMAGo=VALUE2
        IF(X.EQ.'BETA' .OR.X.EQ.'beta' )BETA =VALUE2
        IF(X.EQ.'GAMA' .OR.X.EQ.'gama' )GAMA =VALUE2
        IF(X.EQ.'DELT' .OR.X.EQ.'delt' )DELT =VALUE2
        IF(X.EQ.'To' .OR.X.EQ.'to' )To =VALUE2
        IF(X.EQ.'TDROP' .OR.X.EQ.'tdrop')TDROP=VALUE2
        IF(X.EQ.'Lus' .OR.X.EQ.'lus' )Lus =VALUE2
        IF(X.EQ.'Es' .OR.X.EQ.'es' )Es =VALUE2
        IF(X.EQ.'RHO' .OR.X.EQ.'rho' )RHO =VALUE2
        IF(X.EQ.'A1' .OR.X.EQ.'a1' )A1 =VALUE2
        IF(X.EQ.'A5' .OR.X.EQ.'a5' )A5 =VALUE2
        IF(X.EQ.'A3' .OR.X.EQ.'a3' )A3 =VALUE2
        IF(X.EQ.'A4' .OR.X.EQ.'a4' )A4 =VALUE2
        IF(X.EQ.'A7' .OR.X.EQ.'a7' )A7 =VALUE2
        IF(X.EQ.'A8' .OR.X.EQ.'a8' )A8 =VALUE2
        IF(X.EQ.'L1' .OR.X.EQ.'l1' )L1 =VALUE2
        IF(X.EQ.'L5' .OR.X.EQ.'l5' )L5 =VALUE2
        IF(X.EQ.'L2' .OR.X.EQ.'l2' )L2 =VALUE2
        IF(X.EQ.'L6' .OR.X.EQ.'l6' )L6 =VALUE2
        IF(X.EQ.'L3' .OR.X.EQ.'l3' )L3 =VALUE2
        IF(X.EQ.'L4' .OR.X.EQ.'l4' )L4 =VALUE2
        IF(X.EQ.'L7' .OR.X.EQ.'l7' )L7 =VALUE2
        IF(X.EQ.'L8' .OR.X.EQ.'l8' )L8 =VALUE2
        IF(X.EQ.'Jy1' .OR.X.EQ.'jy1' )Jy1 =VALUE2
        IF(X.EQ.'Jy5' .OR.X.EQ.'jy5' )Jy5 =VALUE2
        IF(X.EQ.'Jy3' .OR.X.EQ.'jy3' )Jy3 =VALUE2
        IF(X.EQ.'Jy4' .OR.X.EQ.'jy4' )Jy4 =VALUE2
        IF(X.EQ.'Jy7' .OR.X.EQ.'jy7' )Jy7 =VALUE2
        IF(X.EQ.'Jy8' .OR.X.EQ.'jy8' )Jy8 =VALUE2
        IF(X.EQ.'Jz1' .OR.X.EQ.'jz1' )Jz1 =VALUE2
        IF(X.EQ.'Jz5' .OR.X.EQ.'jz5' )Jz5 =VALUE2
        IF(X.EQ.'Jz3' .OR.X.EQ.'jz3' )Jz3 =VALUE2
        IF(X.EQ.'Jz4' .OR.X.EQ.'jz4' )Jz4 =VALUE2
        IF(X.EQ.'Jz7' .OR.X.EQ.'jz7' )Jz7 =VALUE2

```

```

      IF(X.EQ.'Jz8' .OR.X.EQ.'jz8' )Jz8 =VALUE2
      IF(X.EQ.'La' .OR.X.EQ.'la' )La =VALUE2
      IF(X.EQ.'Rt' .OR.X.EQ.'rt' )Rt =VALUE2
      IF(X.EQ.'LMAS1'.OR.X.EQ.'lmas1')LMAS1=VALUE2
      IF(X.EQ.'LMAS2'.OR.X.EQ.'lmas2')LMAS2=VALUE2
    END IF
    CHANGE='NO'
    WRITE(*,*)'Another change?(<ret>=N)'
    READ(*,'(A)')CHANGE
    IF(CHANGE.EQ.'YES'.OR.CHANGE.EQ.'Y'.OR.
*    CHANGE.EQ.'yes'.OR.CHANGE.EQ.'y')GOTO 20
    WRITE(*,*)
    WRITE(*,*)'<== ***** Echo new data file ***** ==>'
    WRITE(*,*)
    WRITE(*,'(A,I5)')CH1, NSTEP
    WRITE(*,'(A,I2)')CH2, NITER
    WRITE(*,100)CH3, G
    WRITE(*,100)CH4, OMAGo
    WRITE(*,100)CH5, BETA
    WRITE(*,100)CH6, GAMA
    WRITE(*,100)CH7, DELT
    WRITE(*,100)CH8, To
    WRITE(*,100)CH9, TDROP
    WRITE(*,100)CH10,Lus
    WRITE(*,100)CH11,Es
    WRITE(*,100)CH12,RHO
    WRITE(*,100)CH13,A1
    WRITE(*,100)CH14,A5
    WRITE(*,100)CH15,A3
    WRITE(*,100)CH16,A4
    WRITE(*,100)CH17,A7
    WRITE(*,100)CH18,A8
    WRITE(*,100)CH19,L1
    WRITE(*,100)CH20,L5
    WRITE(*,100)CH21,L2
    WRITE(*,100)CH22,L6
    WRITE(*,100)CH23,L3
    WRITE(*,100)CH24,L4
    WRITE(*,100)CH25,L7
    WRITE(*,100)CH26,L8
    WRITE(*,100)CH27,Jy1
    WRITE(*,100)CH28,Jy5
    WRITE(*,100)CH29,Jy3
    WRITE(*,100)CH30,Jy4
    WRITE(*,100)CH31,Jy7
    WRITE(*,100)CH32,Jy8
    WRITE(*,100)CH33,Jz1
    WRITE(*,100)CH34,Jz5

```



```

WRITE(*,100)CH35,Jz3
WRITE(*,100)CH36,Jz4
WRITE(*,100)CH37,Jz7
WRITE(*,100)CH38,Jz8
WRITE(*,100)CH39,La
WRITE(*,100)CH40,Rt
WRITE(*,100)CH41,LMA51
WRITE(*,100)CH42,LMA52
WRITE(*,*)
WRITE(*,*) '<== ***** New echo is complete ***** ==>'
WRITE(*,*)
CHANGE='NO'
WRITE(*,*) 'Further change?(<ret>=N)'
READ(*, '(A)') CHANGE
IF (CHANGE.EQ. 'YES'.OR.CHANGE.EQ. 'Y'.OR.
*   CHANGE.EQ. 'yes'.OR.CHANGE.EQ. 'y') GOTO 20
END IF
WRITE(*,*)
WRITE(*,*) '-- reading data file finished --'
WRITE(*,*)
C%
PI=4.0*DATAN(1.0D0)
OMAGo=PI*(OMAGo/30.0)
NDROP=TDROP/DELT
MASS1=RHO*A1*L1
MASS3=RHO*A3*L3
MASS4=RHO*A4*L4
MASS5=RHO*A5*L5
MASS7=RHO*A7*L7
MASS8=RHO*A8*L8
C%
100 FORMAT(A,D12.5)
RETURN
END

```

[illegible]

[illegible]


```

OPEN(UNIT=NU,FILE=NAME,STATUS='NEW')
DO 75 K=0,NSTEP
  WRITE(NU,400)T(K),(X(I,K)*180.DO/PI,I=1,NR)
75  CONTINUE
  IF(INT34.GT.1)THEN
    DO 80 J=1,INT34-1
      NTEMP=NR+8*(J-1)
      WRITE(NAME,'(A,I1)')SN,J
      CLOSE(NU)
      OPEN(UNIT=NU,FILE=NAME,STATUS='NEW')
      DO 80 K=0,NSTEP
        WRITE(NU,200)T(K),(X(I,K)*1000.DO,X(I+1,K)*180.DO/PI
          *      ,I=NTEMP+1,NTEMP+7,2)
80      CONTINUE
      END IF
      NTEMP=NR+8*(INT34-1)
      WRITE(NAME,'(A,I1)')SN,INT34
      CLOSE(NU)
      OPEN(UNIT=NU,FILE=NAME,STATUS='NEW')
      DO 85 K=0,NSTEP
        WRITE(NU,200)T(K)
          *      ,(X(I,K)*1000.DO,X(I+1,K)*180.DO/PI,I=NTEMP+1,NTEMP+3,2)
          *      ,(X(I,K)*1000.DO,I=NTEMP+5,NTEMP+6)
          *      ,X(NTEMP+7,K)*180.DO/PI,X(NTEMP+8,K)*1000.DO
85      CONTINUE
      IF(INT78.GT.1)THEN
        DO 90 J=1,INT78-1
          NTEMP=NR+8*(INT34+J-1)
          WRITE(NAME,'(A,I1)')SN,INT34+J
          CLOSE(NU)
          OPEN(UNIT=NU,FILE=NAME,STATUS='NEW')
          DO 90 K=0,NSTEP
            WRITE(NU,200)T(K),(X(I,K)*1000.DO,X(I+1,K)*180.DO/PI
              *      ,I=NTEMP+1,NTEMP+7,2)
90          CONTINUE
          END IF
          NTEMP=NR+8*(INT34+INT78-1)
          WRITE(NAME,'(A,I1)')SN,INT34+INT78
          CLOSE(NU)
          OPEN(UNIT=NU,FILE=NAME,STATUS='NEW')
          DO 95 K=0,NSTEP
            WRITE(NU,200)T(K)
              *      ,(X(I,K)*1000.DO,X(I+1,K)*180.DO/PI,I=NTEMP+1,NTEMP+3,2)
              *      ,(X(I,K)*1000.DO,I=NTEMP+5,NTEMP+6)
              *      ,X(NTEMP+7,K)*180.DO/PI,X(NTEMP+8,K)*1000.DO
95          CONTINUE
          END IF
        END IF
      END IF

```

[illegible]

```

SUBROUTINE MATRIX(M,MD,MP,K,f,q,qd)

C%
C   INPUT  -- q,qd
C   OUTPUT -- M,MD,MP,K,f
C%

INCLUDE '/home/jiechi/newone/program/parameter1.f'
INCLUDE '/home/jiechi/newone/program/parameter2.f'

C%
INTEGER I,J

C%
DOUBLE PRECISION L1,L2,L3,L4,L5,L6,L7,L8
* ,f1(NG1),f3(NG3),f4(NG4),f2(7),f9(11),f0(3)
* ,f5(NG5),f7(NG7),f8(NG8),f6(7),f10(11),f(N)
* ,q1(NG1),q3(NG3),q4(NG4),q2(7),q9(11),q0(3)
* ,q5(NG5),q7(NG7),q8(NG8),q6(7),q10(11),q(N)
* ,q1d(NG1),q3d(NG3),q4d(NG4),q2d(7),q9d(11),q0d(3)
* ,q5d(NG5),q7d(NG7),q8d(NG8),q6d(7),q10d(11),qd(N)
* ,MD1(NG1,NG1),MD3(NG3,NG3),MD4(NG4,NG4),MD5(NG5,NG5)
* ,MP1(NG1,NG1),MP3(NG3,NG3),MP4(NG4,NG4),MP5(NG5,NG5)
* ,MD7(NG7,NG7),MD8(NG8,NG8),MP7(NG7,NG7),MP8(NG8,NG8)
* ,MD(N,N),MD0(3,3),MD2(7,7),MD6(7,7),MD9(11,11),MD10(11,11)
* ,MP(N,N),MP0(3,3),MP2(7,7),MP6(7,7),MP9(11,11),MP10(11,11)
* ,M1(NG1,NG1),M3(NG3,NG3),M4(NG4,NG4),M2(7,7),M9(11,11),M0(3,3)
* ,M5(NG5,NG5),M7(NG7,NG7),M8(NG8,NG8),M6(7,7),M10(11,11),M(N,N)
* ,K1(NG1,NG1),K3(NG3,NG3),K4(NG4,NG4),K2(7,7),K9(11,11),K0(3,3)
* ,K5(NG5,NG5),K7(NG7,NG7),K8(NG8,NG8),K6(7,7),K10(11,11),K(N,N)

C%
DOUBLE PRECISION R1(3),R3(3),R4(3),R5(3),R7(3),R8(3)
* ,J1(3,NG1-3),J5(3,NG5-3),T01(3,3),T05(3,3),T034(3,3)
* ,J3(3,NG3-3),J7(3,NG7-3),T03(3,3),T07(3,3),T078(3,3)
* ,J4(3,NG4-3),J8(3,NG8-3),T04(3,3),T08(3,3)
* ,Jy1,Jy3,Jy4,Jy5,Jy7,Jy8,Jz1,Jz3,Jz4,Jz5,Jz7,Jz8
* ,PHI0(3,N),PHI2(7,N),PHI6(7,N),PHI9(11,N),PHI10(11,N)
* ,PHI1(NG1,N),PHI3(NG3,N),PHI4(NG4,N)
* ,PHI5(NG5,N),PHI7(NG7,N),PHI8(NG8,N)
* ,MASS1,MASS3,MASS4,MASS5,MASS7,MASS8,LMAS1,LMAS2
* ,NT(3,3),ND(3,3),NP(3,9),TP(3,3)

C%
COMMON/COM8/NT,ND                                !<-- output
COMMON/COM9/NP,TP                                !<-- output
COMMON/COMR/R1,R3,R4,R5,R7,R8                    !<-- input(xuinit)
COMMON/COMT/T01,T03,T04,T05,T07,T08,T034,T078    !<-- input(xuinit)
COMMON/COMPHI1/PHI0,PHI1,PHI2,PHI3,PHI4,PHI5      !<-- input(compat)
COMMON/COMPHI2/PHI6,PHI7,PHI8,PHI9,PHI10          !<-- input(compat)
COMMON/COMLMAS/LMAS1,LMAS2                        !<-- input(input,xuinit)
COMMON/COML/L1,L2,L3,L4,L5,L6,L7,L8              !<-- input(input)
COMMON/COMJy/Jy1,Jy3,Jy4,Jy5,Jy7,Jy8             !<-- input(input)
COMMON/COMJz/Jz1,Jz3,Jz4,Jz5,Jz7,Jz8             !<-- input(input)

```

```

COMMON/COMMASS/MASS1,MASS3,MASS4,MASS5,MASS7,MASS8 !<-- input(input)
C%
C*** clear zero start *****C
      DO 10 J=1,7
        q2(J)=0.0D0
        q6(J)=0.0D0
        q2d(J)=0.0D0
        q6d(J)=0.0D0
        f2(J)=0.0D0
        f6(J)=0.0D0
      DO 10 I=1,7
        M2(I,J)=0.0D0
        M6(I,J)=0.0D0
        MD2(I,J)=0.0D0
        MD6(I,J)=0.0D0
        MP2(I,J)=0.0D0
        MP6(I,J)=0.0D0
        K2(I,J)=0.0D0
        K6(I,J)=0.0D0
10      CONTINUE
      DO 15 J=1,11
        q9(J)=0.0D0
        q10(J)=0.0D0
        q9d(J)=0.0D0
        q10d(J)=0.0D0
        f9(J)=0.0D0
        f10(J)=0.0D0
      DO 15 I=1,11
        M9(I,J)=0.0D0
        M10(I,J)=0.0D0
        MD9(I,J)=0.0D0
        MD10(I,J)=0.0D0
        MP9(I,J)=0.0D0
        MP10(I,J)=0.0D0
        K9(I,J)=0.0D0
        K10(I,J)=0.0D0
15      CONTINUE
      DO 20 J=1,NG1
        q1(J)=0.0D0
        q1d(J)=0.0D0
        f1(J)=0.0D0
      DO 20 I=1,NG1
        M1(I,J)=0.0D0
        MD1(I,J)=0.0D0
        MP1(I,J)=0.0D0
        K1(I,J)=0.0D0
20      CONTINUE
      DO 25 J=1,NG3

```



```

q3(J)=0.0D0
q3d(J)=0.0D0
f3(J)=0.0D0
DO 25 I=1,NG3
M3(I,J)=0.0D0
MD3(I,J)=0.0D0
MP3(I,J)=0.0D0
K3(I,J)=0.0D0
25 CONTINUE
DO 30 J=1,NG4
q4(J)=0.0D0
q4d(J)=0.0D0
f4(J)=0.0D0
DO 30 I=1,NG4
M4(I,J)=0.0D0
MD4(I,J)=0.0D0
MP4(I,J)=0.0D0
K4(I,J)=0.0D0
30 CONTINUE
DO 35 J=1,NG5
q5(J)=0.0D0
q5d(J)=0.0D0
f5(J)=0.0D0
DO 35 I=1,NG5
M5(I,J)=0.0D0
MD5(I,J)=0.0D0
MP5(I,J)=0.0D0
K5(I,J)=0.0D0
35 CONTINUE
DO 40 J=1,NG7
q7(J)=0.0D0
q7d(J)=0.0D0
f7(J)=0.0D0
DO 40 I=1,NG7
M7(I,J)=0.0D0
MD7(I,J)=0.0D0
MP7(I,J)=0.0D0
K7(I,J)=0.0D0
40 CONTINUE
DO 45 J=1,NG8
q8(J)=0.0D0
q8d(J)=0.0D0
f8(J)=0.0D0
DO 45 I=1,NG8
M8(I,J)=0.0D0
MD8(I,J)=0.0D0
MP8(I,J)=0.0D0
K8(I,J)=0.0D0

```

```

45      CONTINUE
      DO 50 J=1,N
      f(J)=0.0D0
      DO 50 I=1,N
      M(I,J)=0.0D0
      MD(I,J)=0.0D0
      MP(I,J)=0.0D0
      K(I,J)=0.0D0
50      CONTINUE
      DO 100 J=1,3
      q0(J)=0.0D0
      q0d(J)=0.0D0
      f0(J)=0.0D0
      DO 100 I=1,3
      ND(I,J)=0.0D0
      TP(I,J)=0.0D0
      NP(I,J)=0.0D0
      NP(I,J+3)=0.0D0
      NP(I,J+6)=0.0D0
      MO(I,J)=0.0D0
      MDO(I,J)=0.0D0
      MPO(I,J)=0.0D0
      KO(I,J)=0.0D0
100     CONTINUE
C*** clear zero end *****C
      NT(1,1)= 0.0
      NT(1,2)= 1.0
      NT(1,3)=-DSIN(q(1))
      NT(2,1)= DCOS(q(2))
      NT(2,2)= 0.0
      NT(2,3)= DCOS(q(1))*DSIN(q(2))
      NT(3,1)=-DSIN(q(2))
      NT(3,2)= 0.0
      NT(3,3)= DCOS(q(1))*DCOS(q(2))
      ND(1,3)=-qd(1)*DCOS(q(1))
      ND(2,1)=-qd(2)*DSIN(q(2))
      ND(2,3)=-qd(1)*DSIN(q(1))*DSIN(q(2))
      *      +qd(2)*DCOS(q(1))*DCOS(q(2))
      ND(3,1)=-qd(2)*DCOS(q(2))
      ND(3,3)=-qd(1)*DSIN(q(1))*DCOS(q(2))
      *      -qd(2)*DCOS(q(1))*DSIN(q(2))
      TP(1,1)=-DCOS(q(1))
      TP(1,2)=-DSIN(q(1))*DSIN(q(2))
      TP(1,3)=-DSIN(q(1))*DCOS(q(2))
      TP(2,2)= NT(3,3)
      TP(2,3)=-NT(2,3)
      NP(1,3)= TP(1,1)
      NP(1,6)= TP(1,2)

```

```

NP(1,9)= TP(1,3)
NP(2,4)= NT(3,1)
NP(2,6)= TP(2,2)
NP(2,7)=-NT(2,1)
NP(2,9)= TP(2,3)
CALL COUPLE(q1,PHI1,q,NG1,N,1)
CALL COUPLE(q3,PHI3,q,NG3,N,1)
CALL COUPLE(q4,PHI4,q,NG4,N,1)
CALL COUPLE(q5,PHI5,q,NG5,N,1)
CALL COUPLE(q7,PHI7,q,NG7,N,1)
CALL COUPLE(q8,PHI8,q,NG8,N,1)
CALL COUPLE(q0,PHI0,q,3,N,1)
CALL COUPLE(q2,PHI2,q,7,N,1)
CALL COUPLE(q6,PHI6,q,7,N,1)
CALL COUPLE(q9,PHI9,q,11,N,1)
CALL COUPLE(q10,PHI10,q,11,N,1)
CALL COUPLE(q1d,PHI1,qd,NG1,N,1)
CALL COUPLE(q3d,PHI3,qd,NG3,N,1)
CALL COUPLE(q4d,PHI4,qd,NG4,N,1)
CALL COUPLE(q5d,PHI5,qd,NG5,N,1)
CALL COUPLE(q7d,PHI7,qd,NG7,N,1)
CALL COUPLE(q8d,PHI8,qd,NG8,N,1)
CALL COUPLE(q0d,PHI0,qd,3,N,1)
CALL COUPLE(q2d,PHI2,qd,7,N,1)
CALL COUPLE(q6d,PHI6,qd,7,N,1)
CALL COUPLE(q9d,PHI9,qd,11,N,1)
CALL COUPLE(q10d,PHI10,qd,11,N,1)
CALL SHAFT(q0,q0d,M0,MD0,MP0,f0)
CALL RBEAM(L1, 1.OD0,q2,q2d,M2,MD2,MP2,K2,f2)
CALL RBEAM(L1,-1.OD0,q6,q6d,M6,MD6,MP6,K6,f6)
CALL TANK(L1,L2,T01,T034,q9, q9d, M9, MD9, MP9, K9, F9,LMAS1)
CALL TANK(L5,L6,T05,T078,q10,q10d,M10,MD10,MP10,K10,F10,LMAS2)
CALL FBEAM(Ni1,NG1,L1,MASS1,R1,T01,q1,q1d,M1,MD1,MP1,J1,F1)
CALL FBEAM(Ni3,NG3,L3,MASS3,R3,T03,q3,q3d,M3,MD3,MP3,J3,F3)
CALL FBEAM(Ni4,NG4,L4,MASS4,R4,T04,q4,q4d,M4,MD4,MP4,J4,F4)
CALL FBEAM(Ni5,NG5,L5,MASS5,R5,T05,q5,q5d,M5,MD5,MP5,J5,F5)
CALL FBEAM(Ni7,NG7,L7,MASS7,R7,T07,q7,q7d,M7,MD7,MP7,J7,F7)
CALL FBEAM(Ni8,NG8,L8,MASS8,R8,T08,q8,q8d,M8,MD8,MP8,J8,F8)
CALL STIFF(K1,J1,Ni1,NG1,L1,Jy1,Jz1)
CALL STIFF(K3,J3,Ni3,NG3,L3,Jy3,Jz3)
CALL STIFF(K4,J4,Ni4,NG4,L4,Jy4,Jz4)
CALL STIFF(K5,J5,Ni5,NG5,L5,Jy5,Jz5)
CALL STIFF(K7,J7,Ni7,NG7,L7,Jy7,Jz7)
CALL STIFF(K8,J8,Ni8,NG8,L8,Jy8,Jz8)
CALL ASSEMBLY(M,M0,M1,M2,M3,M4,M5,M6,M7,M8,M9,M10)
CALL ASSEMBLY(MD,MD0,MD1,MD2,MD3,MD4,MD5,MD6,MD7,MD8,MD9,MD10)
CALL ASSEMBLY(MP,MP0,MP1,MP2,MP3,MP4,MP5,MP6,MP7,MP8,MP9,MP10)
CALL ASSEMBLY(K,K0,K1,K2,K3,K4,K5,K6,K7,K8,K9,K10)

```

```
CALL FORCE(f,f0,f1,f2,f3,f4,f5,f6,f7,f8,f9,f10)
RETURN
END
```

```

SUBROUTINE ASSEMBLY(MX,MX0,MX1,MX2,MX3,MX4
* ,MX5,MX6,MX7,MX8,MX9,MX10)
C   OUTPUT -- MX
C   INPUT  -- MX0,MX1,MX2,MX3,MX4,MX5,MX6,MX7,MX8,MX9,MX10
C%
INCLUDE '/home/jiechi/newone/program/parameter1.f'
INCLUDE '/home/jiechi/newone/program/parameter2.f'
C%
INTEGER I,I0,I1,I2,I3,I4,I5,I6,I7,I8,I9,I10
*      ,J,J0,J1,J2,J3,J4,J5,J6,J7,J8,J9,J10
C%
DOUBLE PRECISION MX(N,N),MX0(3,3),MX2(7,7),MX6(7,7)
* ,MX9(11,11),MX10(11,11),MX1(NG1,NG1),MX3(NG3,NG3)
* ,MX4(NG4,NG4),MX5(NG5,NG5),MX7(NG7,NG7),MX8(NG8,NG8)
* ,PHI0(3,N),PHI2(7,N),PHI6(7,N),PHI9(11,N),PHI10(11,N)
* ,PHI1(NG1,N),PHI3(NG3,N),PHI4(NG4,N)
* ,PHI5(NG5,N),PHI7(NG7,N),PHI8(NG8,N)
* ,SUM0,SUM1,SUM2,SUM3,SUM4,SUM5
* ,SUM6,SUM7,SUM8,SUM9,SUM10,SUM
C%
COMMON/COMPHI1/PHI0,PHI1,PHI2,PHI3,PHI4,PHI5 !<--- input(compat)
COMMON/COMPHI2/PHI6,PHI7,PHI8,PHI9,PHI10 !<--- input(compat)
C%
DO 5 J=1,N
DO 5 I=1,N
MX(I,J)=0.0D0
5   CONTINUE
C%
DO 100 J=1,N
DO 100 I=1,N
SUM=0.0D0
DO 11 J0=1,3      !<-----
SUM0=0.0D0          !
DO 10 IO=1,3        !
SUM0=SUM0+PHI0(IO,I)*MX0(IO,J0)      !
10  CONTINUE        !
SUM=SUM+SUM0*PHI0(J0,J)              !
11  CONTINUE        !<-----
DO 16 J1=1,NG1     !<-----
SUM1=0.0D0          !
DO 15 I1=1,NG1      !
SUM1=SUM1+PHI1(I1,I)*MX1(I1,J1)      !
15  CONTINUE        !
SUM=SUM+SUM1*PHI1(J1,J)              !
16  CONTINUE        !<-----
DO 21 J2=1,7        !<-----
SUM2=0.0D0          !
DO 20 I2=1,7        !

```

```

SUM2=SUM2+PHI2(I2,I)*MX2(I2,J2)      !
20  CONTINUE                          !
SUM=SUM+SUM2*PHI2(J2,J)                !
21  CONTINUE      !<<-----
DO 26 J3=1,NG3  !<<-----
SUM3=0.0D0      !
DO 25 I3=1,NG3  !
SUM3=SUM3+PHI3(I3,I)*MX3(I3,J3)      !
25  CONTINUE                          !
SUM=SUM+SUM3*PHI3(J3,J)                !
26  CONTINUE      !<<-----
DO 31 J4=1,NG4  !<<-----
SUM4=0.0D0      !
DO 30 I4=1,NG4  !
SUM4=SUM4+PHI4(I4,I)*MX4(I4,J4)      !
30  CONTINUE                          !
SUM=SUM+SUM4*PHI4(J4,J)                !
31  CONTINUE      !<<-----
DO 36 J5=1,NG5  !<<-----
SUM5=0.0D0      !
DO 35 I5=1,NG5  !
SUM5=SUM5+PHI5(I5,I)*MX5(I5,J5)      !
35  CONTINUE                          !
SUM=SUM+SUM5*PHI5(J5,J)                !
36  CONTINUE      !<<-----
DO 41 J6=1,7    !<<-----
SUM6=0.0D0      !
DO 40 I6=1,7    !
SUM6=SUM6+PHI6(I6,I)*MX6(I6,J6)      !
40  CONTINUE                          !
SUM=SUM+SUM6*PHI6(J6,J)                !
41  CONTINUE      !<<-----
DO 46 J7=1,NG7  !<<-----
SUM7=0.0D0      !
DO 45 I7=1,NG7  !
SUM7=SUM7+PHI7(I7,I)*MX7(I7,J7)      !
45  CONTINUE                          !
SUM=SUM+SUM7*PHI7(J7,J)                !
46  CONTINUE      !<<-----
DO 51 J8=1,NG8  !<<-----
SUM8=0.0D0      !
DO 50 I8=1,NG8  !
SUM8=SUM8+PHI8(I8,I)*MX8(I8,J8)      !
50  CONTINUE                          !
SUM=SUM+SUM8*PHI8(J8,J)                !
51  CONTINUE      !<<-----
DO 56 J9=1,11   !<<-----
SUM9=0.0D0      !

```

```

DO 55 I9=1,11                                !
SUM9=SUM9+PHI9(I9,I)*MX9(I9,J9)             !
55  CONTINUE                                  !
SUM=SUM+SUM9*PHI9(J9,J)                      !
56  CONTINUE      !<<-----
DO 61 J10=1,11  !<<-----
SUM10=0.0D0                                           !
DO 60 I10=1,11                                       !
SUM10=SUM10+PHI10(I10,I)*MX10(I10,J10)             !
60  CONTINUE                                          !
SUM=SUM+SUM10*PHI10(J10,J)                          !
61  CONTINUE      !<<-----
MX(I,J)=SUM
100 CONTINUE
RETURN
END

```

```

SUBROUTINE FORCE(F,F0,F1,F2,F3,F4,F5,F6,F7,F8,F9,F10)
C      OUTPUT -- F
C      INPUT  -- F0,F1,F2,F3,F4,F5,F6,F7,F8,F9,F10
C%
      INCLUDE '/home/jiechi/newone/program/parameter1.f'
      INCLUDE '/home/jiechi/newone/program/parameter2.f'
C%
      INTEGER I,I0,I1,I2,I3,I4,I5,I6,I7,I8,I9,I10
C%
      DOUBLE PRECISION F0(3),F2(7),F6(7),F9(11),F10(11)
*      ,F1(NG1),F3(NG3),F4(NG4),F5(NG5),F7(NG7),F8(NG8),F(N)
*      ,PHI0(3,N),PHI2(7,N),PHI6(7,N),PHI9(11,N),PHI10(11,N)
*      ,PHI1(NG1,N),PHI3(NG3,N),PHI4(NG4,N),GFOR9(N)
*      ,PHI5(NG5,N),PHI7(NG7,N),PHI8(NG8,N),GFOR10(N)
C%
      COMMON/COMPHI1/PHI0,PHI1,PHI2,PHI3,PHI4,PHI5      !<--- input(compat)
      COMMON/COMPHI2/PHI6,PHI7,PHI8,PHI9,PHI10         !<--- input(compat)
      COMMON/GFORCE/GFOR9,GFOR10                       !<--- input(xumain)
C%
      DO 5 I=1,N
      F(I)=0.0D0
5      CONTINUE
      DO 100 I=1,N
      DO 10 I0=1,3
      F(I)=F(I)+PHI0(I0,I)*F0(I0)
10     CONTINUE
      DO 15 I1=1,NG1
      F(I)=F(I)+PHI1(I1,I)*F1(I1)
15     CONTINUE
      DO 20 I2=1,7
      F(I)=F(I)+PHI2(I2,I)*F2(I2)
20     CONTINUE
      DO 25 I3=1,NG3
      F(I)=F(I)+PHI3(I3,I)*F3(I3)
25     CONTINUE
      DO 30 I4=1,NG4
      F(I)=F(I)+PHI4(I4,I)*F4(I4)
30     CONTINUE
      DO 35 I5=1,NG5
      F(I)=F(I)+PHI5(I5,I)*F5(I5)
35     CONTINUE
      DO 40 I6=1,7
      F(I)=F(I)+PHI6(I6,I)*F6(I6)
40     CONTINUE
      DO 45 I7=1,NG7
      F(I)=F(I)+PHI7(I7,I)*F7(I7)
45     CONTINUE
      DO 50 I8=1,NG8

```



```
      F(I)=F(I)+PHI8(I8,I)*F8(I8)
50    CONTINUE
      DO 55 I9=1,11
      F(I)=F(I)+PHI9(I9,I)*F9(I9)
55    CONTINUE
      DO 60 I10=1,11
      F(I)=F(I)+PHI10(I10,I)*F10(I10)
60    CONTINUE
      F(I)=F(I)+GFOR9(I)+GFOR10(I)      !<--- generalized active force
100   CONTINUE
      RETURN
      END
```

```

SUBROUTINE FBEAM(Ni,NGi,L,MASS,Ri,Ti,qi,qid,Mi,MDi,MPi, Ji,Fi)
C%
C      INPUT  -- Ni,NGi,L,MASS,Ri,Ti,qi,qid
C      OUTPUT -- Mi,MDi,MPi,Ji,Fi
C%
!*****
!      NOTE: DIMENSIONS OF THE FOLLOWING DUMMY VARIABLES      *
!              ARE VARIANT.  THOSE DIMENSIONS OF DUMMY VARIABLES *
!              DEFINED IN ARRAY DECLARATION ARE FOR TEN FINITE  *
!              ELEMENTS OR LESS.                                *
!              TO USE MORE THAN TEN ELEMENTS, ONE MUST DEFINE NEW *
!              DIMENSIONS CALCULATED AS FOLLOWING:              *
!
!              NGi=4*(Ni+1)+3                                  *
!
!              WHERE "Ni" IS THE NUMBER OF FINITE ELEMENTS AND  *
!              "NGi" IS AN INTERMEDIATE NUMBER APPEARED IN     *
!              DIMENSIONS.                                      *
!
!      DUMMY VARIABLES -- B(3*NGi,3*NGi),E(3*NGi,NGi),D(3,3*NGi) *
!                      Gi2(3,NGi),TEMP3(3,NGi),TEMP6(3,NGi)      *
!                      TEMP8(NGi,3),X(NGi)                        *
!*****
C%
C      INTEGER Ni,NGi,I,I1,I2,J,J1,J2,K,K1,K2
C%
C      DOUBLE PRECISION L,Li,G,MASS,Ri(3),Ti(3,3),NT(3,3),ND(3,3)
*      ,qi(NGi),qid(NGi),Mi(NGi,NGi),MDi(NGi,NGi),MPi(NGi,NGi)
*      ,Ji(3,NGi-3),Fi(NGi),T3(3),Y(8,4),Z(8,4),SS(4,4),SSYZ(3,8)
*      ,AHAP(24,4),BHAP(24,24),DHAP(3,24),RS(3,4),NP(3,9),TP(3,3)
*      ,Gi1(3,3),Gi2(3,47),B(141,141),E(141,47),D(3,141),X(47)
*      ,TEMP1(3,3),TEMP2(3,3),TEMP3(3,47),TEMP4(3,3),TEMP5(3,3)
*      ,TEMP6(3,47),TEMP7(3,3),TEMP8(47,3) !%,Ci(NGi,NGi)
C%
COMMON/COM3/G          !<--- input(from input)
COMMON/COM8/NT,ND      !<--- input(from matrix)
COMMON/COM9/NP,TP      !<--- input(from matrix)
C%
!%***** CLEAR ZERO START *****%!
DO 20 I=1,3
  T3(I)=0.0
DO 5 J=1,4
  RS(I,J)=0.0
5  CONTINUE
DO 10 J=1,3
  Gi1(I,J)=0.0
  TEMP1(I,J)=0.0
  TEMP2(I,J)=0.0

```

```

TEMP4(I,J)=0.0
TEMP5(I,J)=0.0
TEMP7(I,J)=0.0
10  CONTINUE
    DO 15 J=1,8
        SSYZ(I,J)=0.0
15  CONTINUE
    DO 20 J=1,24
        DHAP(I,J)=0.0
20  CONTINUE
    DO 25 I=1,4
        DO 22 J=1,4
            SS(I,J)=0.0
22  CONTINUE
        DO 25 J=1,8
            Y(J,I)=0.0
            Z(J,I)=0.0
25  CONTINUE
        DO 30 I=1,24
            DO 28 J=1,4
                AHAP(I,J)=0.0
28  CONTINUE
            DO 30 J=1,24
                BHAP(I,J)=0.0
30  CONTINUE
        DO 45 I=1,3*NGi
            DO 40 J1=1,NGi
                E(I,J1)=0.0
40  CONTINUE
            DO 45 J=1,3*NGi
                B(I,J)=0.0
45  CONTINUE
            DO 60 I=1,3
                DO 50 J=1,NGi
                    Gi2(I,J)=0.0
                    TEMP3(I,J)=0.0
                    TEMP6(I,J)=0.0
                    TEMP8(J,I)=0.0
50  CONTINUE
                DO 55 J=1,NGi-3
                    Ji(I,J)=0.0
55  CONTINUE
                DO 60 J=1,3*NGi
                    D(I,J)=0.0
60  CONTINUE
                DO 65 I=1,NGi
                    Fi(I)=0.0
                    DO 65 J=1,NGi

```

```

      Mi(I,J)=0.0
      MDi(I,J)=0.0
      MPi(I,J)=0.0
65      CONTINUE
      !%***** CLEAR ZERO END *****%!
      Li=L/Ni
      T3(1)=-DSIN(qi(1))
      T3(2)= DCOS(qi(1))*DSIN(qi(2))
      T3(3)= DCOS(qi(1))*DCOS(qi(2))
      Y(1,1)= 1.0
      Y(1,3)=-3.0/(Li*Li)
      Y(1,4)= 2.0/(Li**3)
      Y(2,2)= 1.0
      Y(2,3)=-2.0/Li
      Y(2,4)= 1.0/(Li*Li)
      Y(5,3)=-Y(1,3)
      Y(5,4)=-Y(1,4)
      Y(6,3)=-1.0/Li
      Y(6,4)= Y(2,4)
      Z(3,1)= 1.0
      Z(3,3)= Y(1,3)
      Z(3,4)= Y(1,4)
      Z(4,2)=-1.0
      Z(4,3)=-Y(2,3)
      Z(4,4)=-Y(2,4)
      Z(7,3)=-Y(1,3)
      Z(7,4)=-Y(1,4)
      Z(8,3)=-Y(6,3)
      Z(8,4)=-Y(2,4)
      SS(1,1)=1.0
      SS(1,2)=Li/2.0
      SS(1,3)=Li*Li/3.0
      SS(1,4)=Li**3/4.0
      SS(2,1)=SS(1,2)
      SS(2,2)=SS(1,3)
      SS(2,3)=SS(1,4)
      SS(2,4)=Li**4/5.0
      SS(3,1)=SS(1,3)
      SS(3,2)=SS(2,3)
      SS(3,3)=SS(2,4)
      SS(3,4)=Li**5/6.0
      SS(4,1)=SS(1,4)
      SS(4,2)=SS(2,4)
      SS(4,3)=SS(3,4)
      SS(4,4)=Li**6/7.0
      DO 70 I=1,4
      DO 70 J=1,8
      SSYZ(2,J)=SSYZ(2,J)+SS(1,I)*Y(J,I)

```

```

SSYZ(3,J)=SSYZ(3,J)+SS(1,I)*Z(J,I)
70  CONTINUE
!%***** CONSTANT MATRICES START *****%!
Gi1(1,1)= MASS*(Ri(2)*Ri(2)+Ri(3)*Ri(3)+L*L*(Ti(2,1)*Ti(2,1)
*      +Ti(3,1)*Ti(3,1))/3.0+L*(Ri(2)*Ti(2,1)+Ri(3)*Ti(3,1)))
Gi1(2,2)= MASS*(Ri(1)*Ri(1)+Ri(3)*Ri(3)+L*L*(Ti(1,1)*Ti(1,1)
*      +Ti(3,1)*Ti(3,1))/3.0+L*(Ri(1)*Ti(1,1)+Ri(3)*Ti(3,1)))
Gi1(3,3)= MASS*(Ri(1)*Ri(1)+Ri(2)*Ri(2)+L*L*(Ti(1,1)*Ti(1,1)
*      +Ti(2,1)*Ti(2,1))/3.0+L*(Ri(1)*Ti(1,1)+Ri(2)*Ti(2,1)))
Gi1(1,2)=-MASS*(Ri(1)*Ri(2)+L*L*T(1,1)*Ti(2,1)/3.0
*      +L*(Ri(1)*Ti(2,1)+Ri(2)*Ti(1,1))/2.0)
Gi1(1,3)=-MASS*(Ri(1)*Ri(3)+L*L*T(1,1)*Ti(3,1)/3.0
*      +L*(Ri(1)*Ti(3,1)+Ri(3)*Ti(1,1))/2.0)
Gi1(2,3)=-MASS*(Ri(2)*Ri(3)+L*L*T(2,1)*Ti(3,1)/3.0
*      +L*(Ri(2)*Ti(3,1)+Ri(3)*Ti(2,1))/2.0)
Gi1(2,1)=Gi1(1,2)
Gi1(3,1)=Gi1(1,3)
Gi1(3,2)=Gi1(2,3)
DO 100 J=1,4
DO 100 I=1,8
AHAP(I,J)=Ti(1,2)*Y(I,J)+Ti(1,3)*Z(I,J)
AHAP(I+8,J)=Ti(2,2)*Y(I,J)+Ti(2,3)*Z(I,J)
AHAP(I+16,J)=Ti(3,2)*Y(I,J)+Ti(3,3)*Z(I,J)
100  CONTINUE
DO 105 I=1,24
DO 105 J=1,24
DO 105 I1=1,4
DO 105 J1=1,4
BHAP(I,J)=BHAP(I,J)+(MASS/Ni)*AHAP(I,I1)*SS(I1,J1)*AHAP(J,J1)
105  CONTINUE
!%***** SUMMATION DO LOOP START *****%!
DO 120 K=1,Ni      !%k summation loop(k is g in the notes)
K1=4*K
K2=K1+7
DO 110 J=1,4
RS(1,J)=(Ri(1)+Ti(1,1)*(K-1)*Li)*SS(1,J)+Ti(1,1)*SS(2,J)
RS(2,J)=(Ri(2)+Ti(2,1)*(K-1)*Li)*SS(1,J)+Ti(2,1)*SS(2,J)
RS(3,J)=(Ri(3)+Ti(3,1)*(K-1)*Li)*SS(1,J)+Ti(3,1)*SS(2,J)
110  CONTINUE
DO 115 I=1,3
DO 115 J=1,24
DO 115 J1=1,4
DHAP(I,J)=DHAP(I,J)+(MASS/Ni)*RS(I,J1)*AHAP(J,J1)
115  CONTINUE
DO 120 I=K1,K2
Gi2(1,I)=Gi2(1,I)+DHAP(2,I+17-K1)-DHAP(3,I+9-K1)
Gi2(2,I)=Gi2(2,I)+DHAP(3,I+1-K1)-DHAP(1,I+17-K1)
Gi2(3,I)=Gi2(3,I)+DHAP(1,I+9-K1)-DHAP(2,I+1-K1)

```

```

D(1,I)=D(1,I)+2.0*(DHAP(3,I+17-K1)+DHAP(2,I+9-K1))
D(2,I)=D(2,I)-(DHAP(1,I+9-K1)+DHAP(2,I+1-K1))
D(3,I)=D(3,I)-(DHAP(1,I+17-K1)+DHAP(3,I+1-K1))
D(1,I+NGi)=D(1,I+NGi)-(DHAP(1,I+9-K1)+DHAP(2,I+1-K1))
D(2,I+NGi)=D(2,I+NGi)+2.0*(DHAP(1,I+1-K1)+DHAP(3,I+17-K1))
D(3,I+NGi)=D(3,I+NGi)-(DHAP(2,I+17-K1)+DHAP(3,I+9-K1))
D(1,I+2*NGi)=D(1,I+2*NGi)-(DHAP(1,I+17-K1)+DHAP(3,I+1-K1))
D(2,I+2*NGi)=D(2,I+2*NGi)-(DHAP(2,I+17-K1)+DHAP(3,I+9-K1))
D(3,I+2*NGi)=D(3,I+2*NGi)+2.0*(DHAP(1,I+1-K1)+DHAP(2,I+9-K1))
DO 120 J=K1,K2
Mi(I,J)=Mi(I,J)+BHAP(I+1-K1,J+1-K1)+BHAP(I+9-K1,J+9-K1)
*      +BHAP(I+17-K1,J+17-K1)
B(I,J)=B(I,J)+BHAP(I+9-K1,J+9-K1)+BHAP(I+17-K1,J+17-K1)
B(I,J+NGi)=B(I,J+NGi)-BHAP(I+1-K1,J+9-K1)
B(I+NGi,J)=B(I+NGi,J)-BHAP(I+9-K1,J+1-K1)
B(I,J+2*NGi)=B(I,J+2*NGi)-BHAP(I+1-K1,J+17-K1)
B(I+2*NGi,J)=B(I+2*NGi,J)-BHAP(I+17-K1,J+1-K1)
B(I+NGi,J+NGi)=B(I+NGi,J+NGi)+BHAP(I+1-K1,J+1-K1)
*      +BHAP(I+17-K1,J+17-K1)
B(I+NGi,J+2*NGi)=B(I+NGi,J+2*NGi)-BHAP(I+9-K1,J+17-K1)
B(I+2*NGi,J+NGi)=B(I+2*NGi,J+NGi)-BHAP(I+17-K1,J+9-K1)
B(I+2*NGi,J+2*NGi)=B(I+2*NGi,J+2*NGi)+BHAP(I+1-K1,J+1-K1)
*      +BHAP(I+9-K1,J+9-K1)
E(I,J)=E(I,J)+BHAP(I+17-K1,J+9-K1)-BHAP(I+9-K1,J+17-K1)
E(I+NGi,J)=E(I+NGi,J)+BHAP(I+1-K1,J+17-K1)-BHAP(I+17-K1,J+1-K1)
E(I+2*NGi,J)=E(I+2*NGi,J)+BHAP(I+9-K1,J+1-K1)
*      -BHAP(I+1-K1,J+9-K1)
120  CONTINUE
!%***** SUMMATION DO LOOP & CONSTANT MATRICES END *****%!
DO 145 I=1,3
DO 145 J=1,3
DO 140 I1=1,NGi
TEMP2(I,J)=TEMP2(I,J)+D(I,I1+(J-1)*NGi)*qi(I1)
TEMP5(I,J)=TEMP5(I,J)+D(I,I1+(J-1)*NGi)*qid(I1)
DO 140 J1=1,NGi
TEMP1(I,J)=TEMP1(I,J)+qi(I1)*B(I1+(J-1)*NGi,J1+(I-1)*NGi)*qi(J1)
TEMP4(I,J)=TEMP4(I,J)+qi(I1)*B(I1+(J-1)*NGi,J1+(I-1)*NGi)*
*      qid(J1)+qid(I1)*B(I1+(J-1)*NGi,J1+(I-1)*NGi)*qi(J1)
140  CONTINUE
DO 145 I1=1,3
TEMP7(I,J)=TEMP7(I,J)+NP(I,I1+(J-1)*3)*qid(I1)
145  CONTINUE
DO 150 I=1,3
DO 150 J=1,NGi
DO 150 I1=1,NGi
TEMP3(I,J)=TEMP3(I,J)+qi(I1)*E(J+(I-1)*NGi,I1)
TEMP6(I,J)=TEMP6(I,J)+qid(I1)*E(J+(I-1)*NGi,I1)
TEMP8(J,I)=TEMP8(J,I)+E(I1+(I-1)*NGi,J)*qid(I1)

```

```

150      CONTINUE
!%***** MATRIX ASSEMBLY START *****%!
      DO 200 I=1,3
      DO 200 J=1,3
      DO 200 I1=1,3
      DO 200 J1=1,3
      Mi(I,J)=Mi(I,J)
      *   +NT(I1,I)*(Gi1(I1,J1)+TEMP1(I1,J1)+TEMP2(I1,J1))*NT(J1,J)
      MDi(I,J)=MDi(I,J)
      *   +ND(I1,I)*(Gi1(I1,J1)+TEMP1(I1,J1)+TEMP2(I1,J1))*NT(J1,J)
      *   +NT(I1,I)*(Gi1(I1,J1)+TEMP1(I1,J1)+TEMP2(I1,J1))*ND(J1,J)
      *   +NT(I1,I)*(      TEMP4(I1,J1)+TEMP5(I1,J1))*NT(J1,J)
      MPi(I,J)=MPi(I,J)
      *   -TEMP7(I,I1)*(Gi1(I1,J1)+TEMP1(I1,J1)+TEMP2(I1,J1))*NT(J1,J)
200      CONTINUE
      DO 220 J=1,3
      DO 220 I=1,NGi
      DO 220 I1=1,3
      Mi(I,J)=Mi(I,J) +NT(I1,J)*(Gi2(I1,I)+TEMP3(I1,I))
      Mi(J,I)=Mi(J,I) +NT(I1,J)*(Gi2(I1,I)+TEMP3(I1,I))
      MDi(I,J)=MDi(I,J)+ND(I1,J)*(Gi2(I1,I)+TEMP3(I1,I))
      *   +NT(I1,J)*TEMP6(I1,I)
      MDi(J,I)=MDi(J,I)+ND(I1,J)*(Gi2(I1,I)+TEMP3(I1,I))
      *   +NT(I1,J)*TEMP6(I1,I)
      MPi(I,J)=MPi(I,J)-TEMP8(I,I1)*NT(I1,J)
      MPi(J,I)=MPi(J,I)-TEMP7(J,I1)*(Gi2(I1,I)+TEMP3(I1,I))
      DO 220 J1=1,3
!.....
!      DO 220 J2=1,3      !
!      DO 220 I2=1,NGi    !
!      MPi(I,J)=MPi(I,J)-0.5*(D(J1,I+(I1-1)*NGi)+      !summation
!      *      (B(I+(I1-1)*NGi,I2+(J1-1)*NGi)+      !before
!      *      B(I+(J1-1)*NGi,I2+(I1-1)*NGi))*      !modification
!      *      qi(I2))*qid(J2)*NT(I1,J2)*NT(J1,J)      !
!.....
!.....
      X(I)=0.0
      DO 211 I2=1,NGi
      X(I)=X(I)+qi(I2)*(B(I+(I1-1)*NGi,I2+(J1-1)*NGi)+
      *      B(I+(J1-1)*NGi,I2+(I1-1)*NGi))      !summation
211      CONTINUE      !after
      DO 220 J2=1,3      !modification
      MPi(I,J)=MPi(I,J)-0.5*(X(I)+D(J1,I+(I1-1)*NGi))
      *      *qid(J2)*NT(I1,J2)*NT(J1,J)
!.....
220      CONTINUE
c      DO 230 I=1,NGi
c      DO 230 J=1,NGi

```

```

c      Ci(I,J)=MDi(I,J)+MPi(I,J)
c230   CONTINUE
      DO 240 K=1,Ni
      K1=4*K
      K2=K1+7
      DO 235 I=1,2      !%the 3rd row of TP is zero
      DO 235 J=K1,K2
      DO 235 I1=1,3
      DO 235 J1=1,3
      Ji(I,J-3)=Ji(I,J-3)+(MASS*G)/Ni*TP(I,I1)
      *      *Ti(I1,J1)*SSYZ(J1,J+1-K1)
235   CONTINUE
      DO 240 I=K1,K2
      DO 240 I1=1,3
      DO 240 J1=1,3
      Fi(I)=Fi(I)-(MASS*G)/Ni*SSYZ(I1,I+1-K1)*Ti(J1,I1)*T3(J1)
240   CONTINUE
      DO 250 I=1,2      !%the 3rd row of TP is zero
      DO 250 J=1,3
      Fi(I)=Fi(I)-(MASS*G)*TP(I,J)*(Ri(J)+0.5*L*Ti(J,1))
250   CONTINUE
!%***** MATRIX ASSEMBLY END *****%!
      RETURN
      END

```



```

SUBROUTINE RBEAM(Li,SYGN,qi,qid,Mi,MDi,MPi,Ki,Fi)
C%
C      INPUT  -- Li,SYGN,qi,qid
C      OUTPUT -- Mi,MDi,MPi,Ki,Fi
C%
      INTEGER I,I1,I2,J,J1,J2
C%
      DOUBLE PRECISION MASS,G,Lus,Li,SYGN,qi(7),qid(7),Mi(7,7)
*      ,MDi(7,7),MPi(7,7),Ki(7,7),Fi(7),NT(3,3),ND(3,3),Iij(3,3)
*      ,G1(3,3),G2(4,3),H1(3,3),H2(3,3),H3(4,3),H4(3,4),H1D(3,3)
*      ,H2D(3,3),H3D(4,3),H4D(3,4),NP(3,9),TP(3,3),B(12,12)
*      ,D(3,12),TEMP1(4),TEMP2(3,3),TEMP3(4)!%,Ci(7,7)
C%
      COMMON/COM3/G          !<--- input(from input)
      COMMON/COM4/Lus        !<--- input(from input)
      COMMON/COM8/NT,ND      !<--- input(from matrix)
      COMMON/COM9/NP,TP      !<--- input(from matrix)
C%
      DATA MASS/0.14/
C%
C*** clear zero start *****C
      DO 5 I=1,4
        TEMP1(I)=0.0
        TEMP3(I)=0.0
5      CONTINUE
      DO 15 I=1,3
        DO 10 J=1,3
          Iij(I,J)=0.0
          G1(I,J)=0.0
          H1(I,J)=0.0
          H2(I,J)=0.0
          H1D(I,J)=0.0
          H2D(I,J)=0.0
          TEMP2(I,J)=0.0
10     CONTINUE
        DO 15 J=1,4
          G2(J,I)=0.0
          H3(J,I)=0.0
          H4(I,J)=0.0
          H3D(J,I)=0.0
          H4D(I,J)=0.0
15     CONTINUE
        DO 25 I=1,12
          DO 20 J=1,12
            B(I,J)=0.0
20     CONTINUE
        DO 25 J=1,3
          D(J,I)=0.0

```

```

25      CONTINUE
        DO 30 I=1,7
          Fi(I)=0.0
          DO 30 J=1,7
            Mi(I,J)=0.0
            MDi(I,J)=0.0
            MPi(I,J)=0.0
            Ki(I,J)=0.0
30      CONTINUE
C*** clear zero end *****C
        Iij(1,1)=183.6D-06
        Iij(3,3)=Iij(1,1)
        Iij(2,2)=8.4D-06
        G1(1,1)= (Lus*Lus)
        G1(1,3)=-SYGN*Li*Lus
        G1(2,2)= (Li*Li)+(Lus*Lus)
        G1(3,1)= G1(1,3)
        G1(3,3)= (Li*Li)
        G2(1,1)=-SYGN*Lus
        G2(1,3)= Li
        G2(3,2)=-SYGN*Li
        H1(1,1)= (qi(4)*qi(4))+(qi(6)*qi(6))
        H1(2,2)= (qi(6)*qi(6))
        H1(2,3)=-SYGN*qi(4)*qi(6)
        H1(3,2)= H1(2,3)
        H1(3,3)= (qi(4)*qi(4))
        H2(1,1)= 2.0*Lus*qi(6)
        H2(1,2)=-Li*qi(4)
        H2(1,3)=-Li*qi(6)*SYGN
        H2(2,1)= H2(1,2)
        H2(2,2)= H2(1,1)
        H2(2,3)=-Lus*qi(4)*SYGN
        H2(3,1)= H2(1,3)
        H2(3,2)= H2(2,3)
        H2(3,3)= 0.0
        H3(1,1)=-qi(6)*SYGN
        H3(3,1)= qi(4)*SYGN
        H4(1,2)= qi(7)*Iij(1,1)*SYGN
        H4(2,4)= Iij(2,2)*SYGN
        H4(3,2)= Iij(3,3)
        H1D(1,1)= 2.0*((qi(4)*qid(4))+(qi(6)*qid(6)))
        H1D(2,2)= 2.0*(qi(6)*qid(6))
        H1D(2,3)=-qi(4)*qid(6)+qi(6)*qid(4))*SYGN
        H1D(3,2)= H1D(2,3)
        H1D(3,3)= 2.0*(qi(4)*qid(4))
        H2D(1,1)= 2.0*Lus*qid(6)
        H2D(1,2)=-Li*qid(4)
        H2D(1,3)=-Li*qid(6)*SYGN

```

```

H2D(2,1)= H2D(1,2)
H2D(2,2)= H2D(1,1)
H2D(2,3)=-Lus*qid(4)*SYGN
H2D(3,1)= H2D(1,3)
H2D(3,2)= H2D(2,3)
H2D(3,3)= 0.0
H3D(1,1)=-qid(6)*SYGN
H3D(3,1)= qid(4)*SYGN
H4D(1,2)= qid(7)*Iij(1,1)*SYGN
B(1,1)=1.0
B(3,3)=1.0
B(7,7)=1.0
B(5,11)=-SYGN
B(11,5)=-SYGN  !%B(9,7)= -SYGN
B(9,9)=1.0
D(1,3)=2.0*Lus
D(1,5)=-Li
D(1,11)=-Li*SYGN
D(2,1)=D(1,5)
D(2,7)=D(1,3)
D(2,9)=-Lus*SYGN
D(3,3)=D(1,11)
D(3,5)=D(2,9)
TEMP1(1)= MASS*qid(6)*SYGN
TEMP1(2)= 0.0
TEMP1(3)=-MASS*qid(4)*SYGN
TEMP1(4)= Iij(1,1)*qid(5)*SYGN
DO 40 I=1,3
DO 40 J=1,3
TEMP2(I,1)=TEMP2(I,1)+NP(I,J)*qid(J)
TEMP2(I,2)=TEMP2(I,2)+NP(I,J+3)*qid(J)
TEMP2(I,3)=TEMP2(I,3)+NP(I,J+6)*qid(J)
40 CONTINUE
DO 50 I=1,3
DO 50 J=1,3
DO 50 I1=1,3
DO 50 J1=1,3
Mi(I,J)=Mi(I,J)+MASS*NT(I1,I)*NT(J1,J)*
*      (G1(I1,J1)+H1(I1,J1)+H2(I1,J1)+Iij(I1,J1)/MASS)
MDi(I,J)=MDi(I,J)+MASS*(ND(I1,I)*NT(J1,J)*
*      (G1(I1,J1)+H1(I1,J1)+H2(I1,J1)+Iij(I1,J1)/MASS)+
*      NT(I1,I)*ND(J1,J)*
*      (G1(I1,J1)+H1(I1,J1)+H2(I1,J1)+Iij(I1,J1)/MASS)+
*      NT(I1,I)*NT(J1,J)*(H1D(I1,J1)+H2D(I1,J1)))
MPi(I,J)=MPi(I,J)-MASS*TEMP2(I,I1)*NT(J1,J)*
*      (G1(I1,J1)+H1(I1,J1)+H2(I1,J1)+Iij(I1,J1)/MASS)
50 CONTINUE
DO 55 I=1,3

```

```

DO 55 J=1,4
MPi(J+3,I)=MPi(J+3,I)-TEMP1(J)*NT(1,I)
DO 55 I1=1,3
Mi(I,J+3)=Mi(I,J+3)+MASS*NT(I1,I)*
*      (G2(J,I1)+H3(J,I1)+H4(I1,J)/MASS)
Mi(J+3,I)=Mi(J+3,I)+MASS*NT(I1,I)*
*      (G2(J,I1)+H3(J,I1)+H4(I1,J)/MASS)
MDi(I,J+3)=MDi(I,J+3)+MASS*(ND(I1,I)*
*      (G2(J,I1)+H3(J,I1)+H4(I1,J)/MASS)+
*      NT(I1,I)*(H3D(J,I1)+H4D(I1,J)/MASS))
MDi(J+3,I)=MDi(J+3,I)+MASS*(ND(I1,I)*
*      (G2(J,I1)+H3(J,I1)+H4(I1,J)/MASS)+
*      NT(I1,I)*(H3D(J,I1)+H4D(I1,J)/MASS))
MPi(I,J+3)=MPi(I,J+3)-MASS*TEMP2(I,I1)*
*      (G2(J,I1)+H3(J,I1)+H4(I1,J)/MASS)
DO 55 J1=1,3
C%*****
c      TEMP3(1)=0.0
c      TEMP3(2)=0.0
c      TEMP3(3)=0.0
c      TEMP3(4)=0.0
C%*****
      TEMP3(J)=0.0
      DO 52 I2=1,4
      TEMP3(J)=TEMP3(J)+qi(I2+3)*
*      (B(J+4*(J1-1),I2+4*(I1-1))+B(J+4*(I1-1),I2+4*(J1-1)))
52      CONTINUE
      DO 55 J2=1,3
      MPi(J+3,I)=MPi(J+3,I)-0.5*MASS*qid(J2)*NT(I1,J2)*NT(J1,I)
*      *(TEMP3(J)+D(J1,J+4*(I1-1)))
55      CONTINUE
      Mi(4,4)=MASS
      Mi(6,6)=MASS
      Mi(5,5)=Iij(1,1)*qi(7)*qi(7)+Iij(3,3)
      Mi(7,7)=Iij(2,2)
      MDi(5,5)=2.0*Iij(1,1)*qi(7)*qid(7)
      MPi(7,5)=-Iij(1,1)*qi(7)*qid(5)
C%*****
c      DO 60 I=1,7
c      DO 60 J=1,7
c      Ci(I,J)=MDi(I,J)+MPi(I,J)
c60      CONTINUE
C%*****
      Ki(1,4)=(MASS*G)*TP(1,2)*SYGN
      Ki(1,6)=(MASS*G)*TP(1,3)
      Ki(2,4)=(MASS*G)*TP(2,2)*SYGN
      Ki(2,6)=(MASS*G)*TP(2,3)
      Fi(1)=- (MASS*G)*(Li*TP(1,1)*SYGN+Lus*TP(1,3))

```

```
Fi(2)=- (MASS*G)*Lus*TP(2,3)
Fi(3)= 0.0
Fi(4)=- (MASS*G)*DCOS(qi(1))*DSIN(qi(2))*SYGN
Fi(5)= 0.0
Fi(6)=- (MASS*G)*DCOS(qi(1))*DCOS(qi(2))
Fi(7)= 0.0
RETURN
END
```

```

SUBROUTINE SHAFT(q,qd,M,MD,MP,f)

C%
C   INPUT  -- q,qd
C   OUTPUT -- M,MD,MP,f
C%

INTEGER I,J,I1,J1

C%
DOUBLE PRECISION q(3),qd(3),M(3,3),MD(3,3),MP(3,3)
*   ,f(3),NT(3,3),ND(3,3),NP(3,9),TP(3,3),E(3,3),X1(3,3)
*   ,Ils,Mus,Mcb,G,Lus!%,C(3,3)

C%
COMMON/COM3/G          !<--- input(from input)
COMMON/COM4/Lus        !<--- input(from input)
COMMON/COM8/NT,ND      !<--- input(from matrix)
COMMON/COM9/NP,TP      !<--- input(from matrix)

C%
Ils=1220.0D-06
Mus=0.504
Mcb=1.54
E(1,1)=218308.46D-06
E(2,2)=27548.66D-06
E(3,3)=190759.8D-06

C%
DO 10 I=1,3
DO 10 J=1,3
M(I,J)=0.0
MD(I,J)=0.0
MP(I,J)=0.0
E(I,J)=0.0
X1(I,J)=0.0
10 CONTINUE

C%
DO 20 I=1,3
DO 20 J=1,3
DO 20 I1=1,3
X1(I,J)=X1(I,J)+NP(I,I1+(J-1)*3)*qd(I1)
DO 20 J1=1,3
M(I,J)=M(I,J) +NT(I1,I)*E(I1,J1)*NT(J1,J)
MD(I,J)=MD(I,J)+ND(I1,I)*E(I1,J1)*NT(J1,J)
*   +NT(I1,I)*E(I1,J1)*ND(J1,J)
20 CONTINUE
M(3,3)=M(3,3)+Ils
DO 30 I=1,3
DO 30 J=1,3
DO 30 I1=1,3
DO 30 J1=1,3
MP(I,J)=MP(I,J)-X1(I,I1)*E(I1,J1)*NT(J1,J)
30 CONTINUE

```

```

C%*****
c      DO 40 I=1,3
c      DO 40 J=1,3
c      C(I,J)=MD(I,J)+MP(I,J)
c40    CONTINUE
C%*****
      f(1)=((0.5*Mus+Mcb)*G*Lus)*DSIN(q(1))*DCOS(q(2))
      f(2)=((0.5*Mus+Mcb)*G*Lus)*DCOS(q(1))*DSIN(q(2))
      f(3)=0.0
      RETURN
      END

```

```

SUBROUTINE TANK(Li,Lj,Ti,Tab,qi,qid,Mi,MDi,MPi,Ki,Fi,MASS2)
C%
C      INPUT  -- Li,Lj,Ti,Tab,qi,qid
C      OUTPUT -- Mi,MDi,MPi,Ki,Fi
C%
      INTEGER I,J,I1,J1,I2,J2
C%
      DOUBLE PRECISION MASS,G,Lus,La,Rt,Li,Lj,Ti(3,3),Tab(3,3)
*      ,qi(11),qid(11),Mi(11,11),MDi(11,11),MPi(11,11),Ki(11,11)
*      ,Fi(11),Rr(3),Re(3,8),B(24,24),D(3,24),E(8,24),G1(3,3)
*      ,G2(8,3),G3(8,8),H1(3,3),H2(3,3),H3(8,3),H4(8,3),H1D(3,3)
*      ,H2D(3,3),H3D(8,3),H4D(8,3),Iij(3,3),NT(3,3),ND(3,3),NP(3,9)
*      ,TP(3,3),Te3(3),TEMP1(3,3),TEMP2(8,3),TEMP3(8,3)!%<-,Ci(11,11)
*      ,DELT(3),ICFD(6),Ic1(3,3),Itcl(3,3),Ij(3,3),Tabj(3,3)
*      ,TEMP4(8,3),G3D(8,8),FH(8,24),G3P(8,8),MASS1,MASS2
C%
      COMMON/COM3/G          !<-- input(from input)
      COMMON/COM4/Lus        !<-- input(from input)
      COMMON/COM7/La,Rt      !<-- input(from input)
      COMMON/COM8/NT,ND      !<-- input(from matrix)
      COMMON/COM9/NP,TP      !<-- input(from matrix)
      COMMON/COM11/DELT,ICFD !<-- input(from xumain,xuinit)
C%
      MASS1=0.58D0          ! MASS1 = solid mass; MASS2 = liquid mass
      MASS=MASS1+MASS2      ! MASS = total mass
C%
C*** clear zero starts *****C
      DO 20 I=1,3
        Fi(I)=0.0
        Rr(I)=0.0
        Te3(I)=0.0
      DO 10 J=1,3
        G1(I,J)=0.0
        H1(I,J)=0.0
        H2(I,J)=0.0
        H1D(I,J)=0.0
        H2D(I,J)=0.0
        Ic1(I,J)=0.0
        Itcl(I,J)=0.0
        Ij(I,J)=0.0
        Iij(I,J)=0.0
        Tabj(I,J)=0.0
        TEMP1(I,J)=0.0
        Mi(I,J)=0.0
        MDi(I,J)=0.0
        MPi(I,J)=0.0
        Ki(I,J)=0.0
10      CONTINUE

```



```

DO 15 J=1,24
  D(I,J)=0.0
15  CONTINUE
DO 20 J=1,8
  Re(I,J)=0.0
  G2(J,I)=0.0
  H3(J,I)=0.0
  H4(J,I)=0.0
  H3D(J,I)=0.0
  H4D(J,I)=0.0
  TEMP2(J,I)=0.0
  TEMP4(J,I)=0.0
  Mi(I,3+J)=0.0
  MDi(I,3+J)=0.0
  MPi(I,3+J)=0.0
  Ki(I,3+J)=0.0
  Mi(3+J,I)=0.0
  MDi(3+J,I)=0.0
  MPi(3+J,I)=0.0
  Ki(3+J,I)=0.0
20  CONTINUE
DO 30 I=1,8
  Fi(I)=0.0
  TEMP3(I)=0.0
DO 25 J=1,8
  G3(I,J)=0.0
  G3D(I,J)=0.0
  G3P(I,J)=0.0
  Mi(3+I,3+J)=0.0
  MDi(3+I,3+J)=0.0
  MPi(3+I,3+J)=0.0
  Ki(3+I,3+J)=0.0
25  CONTINUE
DO 30 J=1,24
  E(I,J)=0.0
  FH(I,J)=0.0
30  CONTINUE
DO 35 I=1,24
DO 35 J=1,24
  B(I,J)=0.0
35  CONTINUE
C*** clear zero ends *****C
  Rr(1)=Ti(1,1)*Li+(Tab(1,3)*Rt)
  Rr(2)=0.0
  Rr(3)=Lus+Tab(3,1)*(La+Rt)
C... update .....
  Rr(1)=Rr(1)+(MASS2/MASS)*Tab(1,3)*DELT(3)      !
  Rr(2)=Rr(2)+(MASS2/MASS)*Tab(2,2)*DELT(2)      !

```

```

      Rr(3)=Rr(3)+(MASS2/MASS)*Tab(3,1)*DELT(1)      !
C.....!
      Re(1,6)= 0.5*Tab(1,3)
      Re(1,7)=-Tab(1,3)*Rt
      Re(1,8)= Re(1,6)
      Re(2,1)= Ti(2,2)
      Re(2,5)= Tab(2,2)
      Re(2,6)=-Tab(2,2)*Rt/Lj
      Re(2,8)=-Re(2,6)
      Re(3,3)= 1.0
      Re(3,7)= Tab(3,1)*Rt
C... update .....!
      Re(1,6)=Re(1,6)+(MASS2/MASS)*Tab(1,3)*DELT(2)/Lj      !
      Re(1,7)=Re(1,7)-(MASS2/MASS)*Tab(1,3)*DELT(1)      !
      Re(1,8)=Re(1,8)-(MASS2/MASS)*Tab(1,3)*DELT(2)/Lj      !
      Re(2,6)=Re(2,6)-(MASS2/MASS)*Tab(2,2)*DELT(3)/Lj      !
      Re(2,8)=Re(2,8)+(MASS2/MASS)*Tab(2,2)*DELT(3)/Lj      !
      Re(3,7)=Re(3,7)+(MASS2/MASS)*Tab(3,1)*DELT(3)      !
C.....!
      Ic1(1,1)=4.51D-03      !<-- solid inertia matrix
      Ic1(2,2)=4.96D-03
      Ic1(3,3)=Ic1(2,2)
      Itcl(1,1)=ICFD(1)      !<-- liquid inertia matrix
      Itcl(1,2)=ICFD(2)
      Itcl(1,3)=ICFD(3)
      Itcl(2,2)=ICFD(4)
      Itcl(2,3)=ICFD(5)
      Itcl(3,3)=ICFD(6)
      Itcl(2,1)=Itcl(1,2)
      Itcl(3,1)=Itcl(1,3)
      Itcl(3,2)=Itcl(2,3)
      Ij(1,1)=- (MASS2*MASS2/MASS)*(DELT(2)**2+DELT(3)**2)
      Ij(1,2)= (MASS2*MASS2/MASS)*DELT(1)*DELT(2)
      Ij(1,3)= (MASS2*MASS2/MASS)*DELT(1)*DELT(3)
      Ij(2,2)=- (MASS2*MASS2/MASS)*(DELT(1)**2+DELT(3)**2)
      Ij(2,3)= (MASS2*MASS2/MASS)*DELT(2)*DELT(3)
      Ij(3,3)=- (MASS2*MASS2/MASS)*(DELT(1)**2+DELT(2)**2)
      Ij(2,1)= Ij(1,2)
      Ij(3,1)= Ij(1,3)
      Ij(3,2)= Ij(2,3)
      DO 40 I=1,3
      DO 40 J=1,3
      Ij(I,J)=Ij(I,J)+Ic1(I,J)+Itcl(I,J)
40    CONTINUE
      Tabj(1,1)= 1.0
      Tabj(1,2)= 0.0
      Tabj(1,3)= qi(10)
      Tabj(2,1)= 0.0

```

```

Tabj(2,2)= 1.0
Tabj(2,3)=-(qi(9)-qi(11))/Lj
Tabj(3,1)=-Tabj(1,3)
Tabj(3,2)=-Tabj(2,3)
Tabj(3,3)= 1.0
DO 45 I=1,3
DO 45 J=1,3
DO 45 I1=1,3
DO 45 J1=1,3
DO 45 I2=1,3
DO 45 J2=1,3
    Iij(I,J)=Iij(I,J)+Tab(I,I1)*Tabj(I1,J1)*Ij(J1,I2)*
*          Tabj(J2,I2)*Tab(J,J2)
45 CONTINUE
Te3(1)=-DSIN(qi(1))
Te3(2)=DCOS(qi(1))*DSIN(qi(2))
Te3(3)=DCOS(qi(1))*DCOS(qi(2))
C%
DO 50 I=1,8
    D(1,I)=2.0*(Rr(2)*Re(2,I)+Rr(3)*Re(3,I))
    D(1,8+I)=-(Rr(1)*Re(2,I)+Rr(2)*Re(1,I))
    D(1,16+I)=-(Rr(1)*Re(3,I)+Rr(3)*Re(1,I))
    D(2,8+I)=2.0*(Rr(1)*Re(1,I)+Rr(3)*Re(3,I))
    D(2,16+I)=-(Rr(2)*Re(3,I)+Rr(3)*Re(2,I))
    D(3,16+I)=2.0*(Rr(1)*Re(1,I)+Rr(2)*Re(2,I))
    D(2,I)=D(1,8+I)
    D(3,I)=D(1,16+I)
    D(3,8+I)=D(2,16+I)
    G2(I,1)=Rr(2)*Re(3,I)-Rr(3)*Re(2,I)
    G2(I,2)=Rr(3)*Re(1,I)-Rr(1)*Re(3,I)
    G2(I,3)=Rr(1)*Re(2,I)-Rr(2)*Re(1,I)
DO 50 J=1,8
    B(I,J)=Re(2,I)*Re(2,J)+Re(3,I)*Re(3,J)
    B(I,8+J)=-Re(1,I)*Re(2,J)
    B(I,16+J)=-Re(1,I)*Re(3,J)
    B(8+I,J)=-Re(2,I)*Re(1,J)
    B(8+I,8+J)=Re(1,I)*Re(1,J)+Re(3,I)*Re(3,J)
    B(8+I,16+J)=-Re(2,I)*Re(3,J)
    B(16+I,J)=-Re(3,I)*Re(1,J)
    B(16+I,8+J)=-Re(3,I)*Re(2,J)
    B(16+I,16+J)=Re(1,I)*Re(1,J)+Re(2,I)*Re(2,J)
    E(I,J)=-B(16+I,8+J)+B(8+I,16+J)
    E(I,8+J)=-B(I,16+J)+B(16+I,J)
    E(I,16+J)=-B(8+I,J)+B(I,8+J)
50 CONTINUE
C%
TEMP4(2,1)=Ti(1,1)*qi(7)
TEMP4(2,3)=Ti(3,3)

```

```

TEMP4(4,2)=Ti(2,2)
TEMP4(7,2)=Tab(2,2)
G1(1,1)= Rr(2)**2+Rr(3)**2
G1(1,2)=-Rr(1)*Rr(2)
G1(1,3)=-Rr(1)*Rr(3)
G1(2,2)= Rr(1)**2+Rr(3)**2
G1(2,3)=-Rr(2)*Rr(3)
G1(3,3)= Rr(1)**2+Rr(2)**2
G1(2,1)= G1(1,2)
G1(3,1)= G1(1,3)
G1(3,2)= G1(2,3)
DO 52 I=1,8
DO 52 J=1,3
DO 52 I1=1,3
    H4(I,J)=H4(I,J)+TEMP4(I,I1)*Iij(I1,J)
52 CONTINUE
H4D(2,1)=Ti(1,1)*Iij(1,1)*qid(7)
H4D(2,2)=Ti(1,1)*Iij(1,2)*qid(7)
H4D(2,3)=Ti(1,1)*Iij(1,3)*qid(7)
DO 54 I=1,8
DO 54 J=1,8
DO 54 I1=1,3
    G3(I,J)=G3(I,J)+H4(I,I1)*TEMP4(J,I1)
    G3D(I,J)=G3D(I,J)+H4D(I,I1)*TEMP4(J,I1)
54 CONTINUE
G3D(1,2)=G3D(1,2)+Ti(1,1)*H4(1,1)*qid(7)
G3D(2,2)=G3D(2,2)+Ti(1,1)*H4(2,1)*qid(7)
G3D(3,2)=G3D(3,2)+Ti(1,1)*H4(3,1)*qid(7)
G3D(4,2)=G3D(4,2)+Ti(1,1)*H4(4,1)*qid(7)
G3D(5,2)=G3D(5,2)+Ti(1,1)*H4(5,1)*qid(7)
G3D(6,2)=G3D(6,2)+Ti(1,1)*H4(6,1)*qid(7)
G3D(7,2)=G3D(7,2)+Ti(1,1)*H4(7,1)*qid(7)
G3D(8,2)=G3D(8,2)+Ti(1,1)*H4(8,1)*qid(7)
FH(2,4)= Ti(1,1)*Iij(1,1)
FH(2,12)=Ti(1,1)*Iij(1,2)
FH(2,20)=Ti(1,1)*Iij(1,3)
G3P(4,2)=Ti(1,1)*(Ti(1,1)*Iij(1,1)*qi(7)+
*      Ti(3,3)*Iij(1,3))*qid(5)
G3P(4,4)=Ti(1,1)*Ti(2,2)*Iij(1,2)*qid(5)
G3P(4,7)=Ti(1,1)*Tab(2,2)*Iij(1,2)*qid(5)
C%
DO 80 I=1,3
DO 60 J=1,3
    TEMP1(I,1)=TEMP1(I,1)+NP(I,J)*qid(J)
    TEMP1(I,2)=TEMP1(I,2)+NP(I,3+J)*qid(J)
    TEMP1(I,3)=TEMP1(I,3)+NP(I,6+J)*qid(J)
DO 60 I1=1,8
    H2(I,J)=H2(I,J)+D(I,I1+8*(J-1))*qi(3+I1)

```

```

      H2D(I,J)=H2D(I,J)+D(I,I1+8*(J-1))*qid(3+I1)
DO 60 J1=1,8
      H1(I,J)=H1(I,J)+qi(3+I1)*B(I1+8*(I-1),J1+8*(J-1))*qi(3+J1)
      H1D(I,J)=H1D(I,J)+qid(3+I1)*B(I1+8*(I-1),J1+8*(J-1))*qi(3+J1)
      +qi(3+I1)*B(I1+8*(I-1),J1+8*(J-1))*qid(3+J1)
60    *
      CONTINUE
      DO 80 J=1,8
      DO 80 I1=1,8
      H3(J,I)=H3(J,I)+E(J,I1+8*(I-1))*qi(3+I1)
      H3D(J,I)=H3D(J,I)+E(J,I1+8*(I-1))*qid(3+I1)
      TEMP2(J,I)=TEMP2(J,I)+(E(I1,J+8*(I-1))+
      *      FH(I1,J+8*(I-1))/MASS)*qid(3+I1)
80    CONTINUE
C%
      DO 200 I=1,3
      DO 100 J=1,3
      Fi(I)=Fi(I)-MASS*G*TP(I,J)*Rr(J)
      DO 100 I1=1,3
      DO 100 J1=1,3
      Mi(I,J)=Mi(I,J)+MASS*NT(I1,I)*(G1(I1,J1)+H1(I1,J1)
      *      +H2(I1,J1)+Iij(I1,J1)/MASS)*NT(J1,J)
      MDi(I,J)=MDi(I,J)+MASS*(ND(I1,I)*(G1(I1,J1)+H1(I1,J1)+
      *      H2(I1,J1)+Iij(I1,J1)/MASS)*NT(J1,J)+NT(I1,I)*
      *      (G1(I1,J1)+H1(I1,J1)+H2(I1,J1)+Iij(I1,J1)/MASS)*
      *      ND(J1,J)+NT(I1,I)*(H1D(I1,J1)+H2D(I1,J1))*NT(J1,J))
      MPi(I,J)=MPi(I,J)-MASS*TEMP1(I,I1)*(G1(I1,J1)+H1(I1,J1)
      *      +H2(I1,J1)+Iij(I1,J1)/MASS)*NT(J1,J)
100   CONTINUE
      DO 200 J=1,8
      DO 200 I1=1,3
      Mi(I,3+J)=Mi(I,3+J)+(MASS*NT(I1,I)*(G2(J,I1)
      *      +H3(J,I1)+H4(J,I1)/MASS))
      Mi(3+J,I)=Mi(3+J,I)+(MASS*NT(I1,I)*(G2(J,I1)
      *      +H3(J,I1)+H4(J,I1)/MASS))
      MDi(I,3+J)=MDi(I,3+J)+(MASS*ND(I1,I)*(G2(J,I1)+H3(J,I1)
      *      +H4(J,I1)/MASS)+MASS*NT(I1,I)
      *      *(H3D(J,I1)+H4D(J,I1)/MASS))
      MDi(3+J,I)=MDi(3+J,I)+(MASS*ND(I1,I)*(G2(J,I1)+H3(J,I1)
      *      +H4(J,I1)/MASS)+MASS*NT(I1,I)
      *      *(H3D(J,I1)+H4D(J,I1)/MASS))
      MPi(I,3+J)=MPi(I,3+J)-MASS*TEMP1(I,I1)*(G2(J,I1)
      *      +H3(J,I1)+H4(J,I1)/MASS)
      MPi(3+J,I)=MPi(3+J,I)-MASS*TEMP2(J,I1)*NT(I1,I)
      Ki(I,3+J)=Ki(I,3+J)+MASS*G*TP(I,I1)*Re(I1,J)
      DO 200 J1=1,3
      TEMP3(J)=0.0
      DO 114 I2=1,8
      TEMP3(J)=TEMP3(J)+qi(3+I2)*(B(J+8*(J1-1),I2+8*(I1-1))

```

```

*                                     +B(J+8*(I1-1),I2+8*(J1-1)))
114  CONTINUE
      DO 200 J2=1,3
      MPi(3+J,I)=MPi(3+J,I)-0.5*MASS*qid(J2)*NT(I1,J2)*NT(J1,I)
*                                     *(TEMP3(J)+D(J1,J+8*(I1-1)))
200  CONTINUE
C%
      DO 300 I=1,8
      DO 250 J=1,3
      Fi(3+I)=Fi(3+I)-MASS*G*Re(J,I)*Te3(J)
250  CONTINUE
      DO 300 J=1,8
      Mi(3+I,3+J)=0.5*MASS*(B(I,J)+B(8+I,8+J)+B(16+I,16+J))+G3(I,J)
      MDi(3+I,3+J)=G3D(I,J)
      MPi(3+I,3+J)=G3P(I,J)
300  CONTINUE
C%*****
c      DO 400 I=1,11
c      DO 400 J=1,11
c      Ci(I,J)=MDi(I,J)+MPi(I,J)
c400  CONTINUE
C%*****
      RETURN
      END

```

```

      SUBROUTINE STIFF(Ki,Ji,Ni,NGi,L,Jyi,Jzi)
C%
C      OUTPUT -- Ki
C      INPUT  -- Ji,Ni,NGi,L,Jyi,Jzi
C%
      INTEGER I,J,I1,Ni,NGi
C%
      DOUBLE PRECISION Ki(NGi,NGi),Ji(3,NGi-3),Ks(8,8)
      * ,L,Li,Ei,Jyi,Jzi,COEFy,COEFz
C%
      COMMON/COM5/Ei          !<--- input(from input)

C*** clear zero start *****C
      DO 10 J=1,8
      DO 10 I=1,8
      Ks(I,J)=0.0
10    CONTINUE
      DO 20 J=1,NGi
      DO 20 I=1,NGi
      Ki(I,J)=0.0
20    CONTINUE
C*** clear zero end *****C
      Li=L/Ni
      COEFy=Ei*Jyi/(Li**3)
      COEFz=Ei*Jzi/(Li**3)
      Ks(1,1)=12.0*COEFz
      Ks(1,2)=6.0*Li*COEFz
      Ks(1,5)=-Ks(1,1)
      Ks(1,6)= Ks(1,2)
      Ks(2,1)= Ks(1,2)
      Ks(2,6)=2.0*Li*Li*COEFz
      Ks(2,2)= Ks(2,6)*2.0
      Ks(2,5)=-Ks(1,2)
      Ks(5,1)= Ks(1,5)
      Ks(5,2)= Ks(2,5)
      Ks(5,5)= Ks(1,1)
      Ks(5,6)= Ks(2,5)
      Ks(6,1)= Ks(1,6)
      Ks(6,2)= Ks(2,6)
      Ks(6,5)= Ks(5,6)
      Ks(6,6)= Ks(2,2)
      Ks(3,3)=12.0*COEFy
      Ks(3,4)=-6.0*Li*COEFy
      Ks(3,7)=-Ks(3,3)
      Ks(3,8)= Ks(3,4)
      Ks(4,3)= Ks(3,4)
      Ks(4,8)=2.0*Li*Li*COEFy
      Ks(4,4)= Ks(4,8)*2.0

```

```

Ks(4,7)=-Ks(3,4)
Ks(7,3)= Ks(3,7)
Ks(7,4)= Ks(4,7)
Ks(7,7)= Ks(3,3)
Ks(7,8)= Ks(4,7)
Ks(8,3)= Ks(3,8)
Ks(8,4)= Ks(4,8)
Ks(8,7)= Ks(7,8)
Ks(8,8)= Ks(4,4)
DO 30 I=1,3
DO 30 J=4,NGi
Ki(I,J)=Ji(I,J-3)
30  CONTINUE
DO 40 I1=1,Ni      !% I1 denotes g in the formula
DO 40 J=(4*I1),(4*I1)+7
DO 40 I=(4*I1),(4*I1)+7
Ki(I,J)=Ki(I,J)+Ks(I-(4*I1)+1,J-(4*I1)+1)
40  CONTINUE
RETURN
END

```



```

SUBROUTINE EQUATION(qdd,M,MD,MP,K,f,q,qd)
C%
C   OUTPUT -- qdd
C   INPUT  -- M,MD,MP,K,f,q,qd
C%
C   INCLUDE '/home/jiechi/newone/program/parameter1.f'
C   INCLUDE '/home/jiechi/newone/program/parameter3.f'
C%
C   INTEGER INDEX(L),I,J
C%
C   DOUBLE PRECISION qdd(L),M(L,L),MD(L,L),MP(L,L),K(L,L)
*   ,f(L),q(L),qd(L),Mout(L,L),DUM(L)
C%
C   DO 10 I=1,L
C     qdd(I)=0.0
C     DUM(I)=0.0
C     DO 10 J=1,L
C       Mout(I,J)=0.0
10  CONTINUE
C     DO 20 I=1,L
C       DO 20 J=1,L
C         f(I)=f(I)-K(I,J)*q(J)-(MD(I,J)+MP(I,J))*qd(J)
20  CONTINUE
C     CALL GAUSS(M,L,INDEX,Mout,DUM)      !%to solve --> M * qdd = f
C     CALL SOLVE(Mout,L,INDEX,f,qdd)      !%qdd is a variable vector
C     RETURN
C   END
C*****
SUBROUTINE GAUSS(AA,N,L,A,S)
C%
C   INPUT  -- AA,N
C                                     ! [AA]{X}={B}
C   OUTPUT -- L,A
C   DUMMY  -- S
C%
C   INTEGER I,J,K,LK,N,L(N)
C%
C   DOUBLE PRECISION AA(N,N),A(N,N),S(N),R,RMAX,XMULT
C%
C   CHARACTER*26 ERROR
C%
C   DO 10 I=1,N
C     L(I)=0
C     S(I)=0.0
C     DO 10 J=1,N
C       A(I,J)=0.0
10  CONTINUE
C   ERROR='ALL ZERO ELEMENTS IN ROW #'
C   DO 30 I=1,N

```

```

DO 30 J=1,N
A(I,J)=AA(I,J)
30 CONTINUE
DO 40 I=1,N
L(I)=I
S(I)=0.0
DO 35 J=1,N
S(I)=DMAX1(S(I),ABS(A(I,J)))
35 CONTINUE
IF(S(I).EQ.0.0)GOTO 99
40 CONTINUE
DO 60 K=1,N-1
RMAX=0.0
DO 50 I=K,N
R=ABS(A(L(I),K))/S(L(I))
IF(R.LE.RMAX)GOTO 50
J=I
RMAX=R
50 CONTINUE
LK=L(J)
L(J)=L(K)
L(K)=LK
DO 60 I=K+1,N
XMULT=A(L(I),K)/A(LK,K)
A(L(I),K)=XMULT
IF(A(L(I),K).EQ.0.0)GOTO 60
DO 55 J=K+1,N
A(L(I),J)=A(L(I),J)-XMULT*A(LK,J)
55 CONTINUE
60 CONTINUE
GOTO 100
99 WRITE(*,'(1X,A30,1X,I3)')ERROR,I
STOP
100 RETURN
END

```

C*****! .

SUBROUTINE SOLVE(A,N,L,B,X)

C%

C INPUT -- A,N,L,B

! [A]{X}={B}

C OUTPUT -- X

C%

INTEGER I,J,N,L(N)

C%

DOUBLE PRECISION A(N,N),B(N),X(N),SUM

C%

SUM=0.0

DO 5 I=1,N

X(I)=0.0

```

5      CONTINUE
      DO 10 J=1,N-1
      DO 10 I=J+1,N
      B(L(I))=B(L(I))-A(L(I),J)*B(L(J))
10     CONTINUE
      X(N)=B(L(N))/A(L(N),N)
      DO 30 I=1,N-1
      SUM=B(L(N-I))
      DO 20 J=N-I+1,N
      SUM=SUM-A(L(N-I),J)*X(J)
20     CONTINUE
      X(N-I)=SUM/A(L(N-I),N-I)
30     CONTINUE
      RETURN
      END
C*****!
      SUBROUTINE COUPLE(C,A,B,N1,N2,N3)
C%
C      OUTPUT -- C
C      INPUT  -- A,B,N1,N2,N3
C%
      INTEGER I,J,K,N1,N2,N3                      !C=A*B
C%
      DOUBLE PRECISION C(N1,N3),A(N1,N2),B(N2,N3)
C%
      DO 100 J=1,N3
      DO 100 I=1,N1
      C(I,J)=0.0
100    CONTINUE
      DO 200 J=1,N3
      DO 200 I=1,N1
      DO 200 K=1,N2
      C(I,J)=C(I,J)+A(I,K)*B(K,J)
200    CONTINUE
      RETURN
      END
C*****!
      SUBROUTINE SKEW(AS,A)
C%
C      INPUT  -- A
C      OUTPUT -- AS
C%
      DOUBLE PRECISION AS(3,3),A(3)
C%
      AS(1,1)= 0.0
      AS(1,2)=-A(3)
      AS(1,3)= A(2)
      AS(2,1)= A(3)

```

```
AS(2,2)= 0.0  
AS(2,3)=-A(1)  
AS(3,1)=-A(2)  
AS(3,2)= A(1)  
AS(3,3)= 0.0  
RETURN  
END
```

```

      SUBROUTINE GFOROUT(Lj,qj,Pj,Ti,PHIj,Rforce,Torque,GFORj)
C%
C      INPUT  -- Lj,qj,Pj,Ti,PHIj,Rforce,Torque
C      OUTPUT -- GFORj                      !<--- generalized active force
C%
C*****
C      Rforce and Torque are expressed in inertial      !
C      coordinates.                                     !
C*****
C%
C      INCLUDE'/home/jiechi/newone/program/parameter1.f'
C%
C      INTEGER I,J,I1,J1
C%
C      DOUBLE PRECISION Lj,qj(N),Ti(3,3),Te(3,3),PHIj(11,N)
C      * ,GFORj(N),Pj(3),P3(3),P4(3),TM1(3,3),TM2(3,3),TM3(3,3)
C      * ,TM4(3,3),TM5(3,3),Tab(3,3),P5(3),P6(3),P7(3),W3(3,8)
C      * ,W4(3,8),Wxy(3,8),Wyz(3,8),P8(3,8),P9(3,8),Vsub(3,11)
C      * ,Wsub(3,11),Vqj(3,N),Wqj(3,N),Ps5(3,3),Ps6(3,3),Ps7(3,3)
C      * ,Psj(3,3),TEMP2(3,3),NT(3,3),ND(3,3),Rforce(3),Torque(3)
C%
C      COMMON/COM8/NT,ND                                !<--- input(matrix)
C      COMMON/COM10/P3,P4,TM1,TM2,TM3,TM4,TM5,Tab       !<--- input(postpro)
C      COMMON/COM12/Te                                   !<--- input(postpro)
C%
C      DO 25 J=1,3
C      P5(J)=0.0D0
C      P6(J)=0.0D0
C      P7(J)=0.0D0
C      DO 10 I=1,3
C      TEMP2(I,J)=0.0D0
10    CONTINUE
C      DO 15 I=1,8
C      W3(J,I)=0.0D0
C      W4(J,I)=0.0D0
C      Wxy(J,I)=0.0D0
C      Wyz(J,I)=0.0D0
C      P8(J,I)=0.0D0
C      P9(J,I)=0.0D0
15    CONTINUE
C      DO 20 I=1,11
C      Vsub(J,I)=0.0D0
C      Wsub(J,I)=0.0D0
20    CONTINUE
C      DO 25 I=1,N
C      Vqj(J,I)=0.0D0
C      Wqj(J,I)=0.0D0
C      GFORj(I)=0.0D0

```

```

25      CONTINUE
C%
      W3(1,2)=qj(7)
      W3(2,4)=1.0D0
      W3(3,2)=1.0D0
      W4(1,6)=1.0D0/Lj
      W4(1,8)=-W4(1,6)
      W4(2,7)= 1.0D0
      W4(3,6)=-qj(10)*W4(1,6)
      W4(3,8)=-W4(3,6)
      Wyz(2,4)= 1.0D0
      Wyz(3,2)= 1.0D0
      Wxy(1,6)= W4(1,6)
      Wxy(1,8)=-Wxy(1,6)
      Wxy(2,7)= 1.0D0
      P8(2,1)=1.0D0
      P8(3,3)=1.0D0
      P9(2,5)=1.0D0
      P9(3,6)=0.5D0
      P9(3,8)=0.5D0
      DO 40 J=1,3
      DO 40 I=1,3
      P7(J)=P7(J)+TM5(J,I)*P4(I)
      DO 40 I1=1,3
      P5(J)=P5(J)+TM2(J,I)*TM3(I,I1)*P3(I1)
40      CONTINUE
      DO 45 J=1,3
      DO 45 I=1,3
      DO 45 J1=1,3
      DO 45 I1=1,3
      P6(J)=P6(J)+TM2(J,I)*TM3(I,J1)*TM4(J1,I1)*P7(I1)
45      CONTINUE
C%
      CALL SKEW(Ps5,P5)
      CALL SKEW(Ps6,p6)
      CALL SKEW(Ps7,P7)
      CALL SKEW(Psj,Pj)
      DO 60 J=1,3
      DO 60 I=1,3
      TEMP2(I,J)=0.0D0
60      CONTINUE
      DO 65 J=1,3
      DO 65 I=1,3
      Wsub(I,J)=NT(I,J)
      DO 65 J1=1,3
      Vsub(I,J)=Vsub(I,J)-Psj(I,J1)*NT(J1,J)
      DO 65 I1=1,3
      TEMP2(I,J)=TEMP2(I,J)+Ti(I,J1)*TM1(J1,I1)*

```

```

*                               (Ps5(I1,J)+Ps6(I1,J))
65  CONTINUE
    DO 70 I=1,3
    DO 70 J=1,8
    DO 70 I1=1,3
    Vsub(I,3+J)=Vsub(I,3+J)+Ti(I,I1)*P8(I1,J)
*                               -TEMP2(I,I1)*Wyz(I1,J)
70  CONTINUE
    DO 75 J=1,3
    DO 75 I=1,3
    TEMP2(I,J)=0.000
75  CONTINUE
    DO 80 J=1,3
    DO 80 I=1,3
    DO 80 J1=1,3
    DO 80 I1=1,3
    TEMP2(I,J)=TEMP2(I,J)+Tab(I,J1)*TM4(J1,I1)*Ps7(I1,J)
80  CONTINUE
    DO 85 I=1,3
    DO 85 J=1,8
    DO 85 I1=1,3
    Vsub(I,3+J)=Vsub(I,3+J)+Tab(I,I1)*P9(I1,J)
*                               -TEMP2(I,I1)*Wxy(I1,J)
    Wsub(I,3+J)=Wsub(I,3+J)+Ti(I,I1)*W3(I1,J)
*                               +Tab(I,I1)*W4(I1,J)
85  CONTINUE
    DO 90 I=1,3
    DO 90 J=1,N
    DO 90 I1=1,11
    Vqj(I,J)=Vqj(I,J)+Vsub(I,I1)*PHIj(I1,J)
    Wqj(I,J)=Wqj(I,J)+Wsub(I,I1)*PHIj(I1,J)
90  CONTINUE
    DO 100 J=1,N
    DO 100 I=1,3
    DO 100 I1=1,3
    GFORj(J)=GFORj(J)+Vqj(I,J)*Te(I1,I)*Rforce(I1)
*                               +Wqj(I,J)*Te(I1,I)*Torque(I1)
100 CONTINUE
    RETURN
    END

```

```

      SUBROUTINE MAPPING(M,MD,MP,K,f,q,qd,qdd)
C%
C      INPUT  -- M,MD,MP,K,f,q,qd,qdd (befor mapping)
C      OUTPUT -- M,MD,MP,K,f,q,qd,qdd (after mapping)
C%
C*****
C      To eliminate the ITHth equation to a sub- *
C      set of equation system with N-1 variables *
C      *
C      This mapping subroutine is valid for      *
C      1 < ITH < N                               *
C      where ITH is the index of the row and the *
C      column of the matrix to be eliminated.   *
C*****
C%
      INCLUDE '/home/jiechi/newone/program/parameter1.f' !def N
      INCLUDE '/home/jiechi/newone/program/parameter4.f' !def ITH
C%
      INTEGER I,J,NN
C%
      PARAMETER(NN=N*N)
C%
      DOUBLE PRECISION M(NN),MD(NN),MP(NN),K(NN),f(N)
      * ,q(N),qd(N),qdd(N)
C%
      IF(ITH.EQ.1.OR.ITH.EQ.N)THEN
      c      IF(ITH.EQ.1)THEN
      c      WRITE(*,*)'The MAPPING subroutine cann"t be used'
      c      WRITE(*,*)'to eliminate the 1st row and the 1st column of the'
      c      WRITE(*,*)'matrix which you are expecting to be mapped to a'
      c      WRITE(*,*)'submatrix.'
      c      END IF
      c      IF(ITH.EQ.N)THEN
      c      WRITE(*,*)'The MAPPING subroutine cann"t be used'
      c      WRITE(*,*)'to eliminate the last row and the last column of the'
      c      WRITE(*,*)'matrix which you are expecting to be mapped to a'
      c      WRITE(*,*)'submatrix.'
      c      END IF
      c      STOP
      c      END IF
      DO 10 I=1,ITH-1
      f(I)=f(I)-(M(N*(ITH-1)+I)*qdd(ITH)+(MD(N*(ITH-1)+I)
      *      +MP(N*(ITH-1)+I))*qd(ITH)+K(N*(ITH-1)+I)*q(ITH))
10      CONTINUE
      DO 20 I=ITH+1,N
      f(I-1)=f(I)-(M(N*(ITH-1)+I)*qdd(ITH)+(MD(N*(ITH-1)+I)
      *      +MP(N*(ITH-1)+I))*qd(ITH)+K(N*(ITH-1)+I)*q(ITH))
20      CONTINUE

```



```

DO 30 I=ITH+1,N
q(I-1)=q(I)
qd(I-1)=qd(I)
qdd(I-1)=qdd(I)
30  CONTINUE
DO 50 J=1, ITH-1
DO 40 I=1, ITH-1
M(((J-1)*(N-1))+I)=M(((J-1)*N)+I)
MD(((J-1)*(N-1))+I)=MD(((J-1)*N)+I)
MP(((J-1)*(N-1))+I)=MP(((J-1)*N)+I)
K(((J-1)*(N-1))+I)=K(((J-1)*N)+I)
40  CONTINUE
DO 50 I=ITH+1,N
M(((J-1)*(N-1))+I-1)=M(((J-1)*N)+I)
MD(((J-1)*(N-1))+I-1)=MD(((J-1)*N)+I)
MP(((J-1)*(N-1))+I-1)=MP(((J-1)*N)+I)
K(((J-1)*(N-1))+I-1)=K(((J-1)*N)+I)
50  CONTINUE
DO 70 J=ITH+1,N
DO 60 I=1, ITH-1
M(((J-2)*(N-1))+I)=M(((J-1)*N)+I)
MD(((J-2)*(N-1))+I)=MD(((J-1)*N)+I)
MP(((J-2)*(N-1))+I)=MP(((J-1)*N)+I)
K(((J-2)*(N-1))+I)=K(((J-1)*N)+I)
60  CONTINUE
DO 70 I=ITH+1,N
M(((J-2)*(N-1))+I-1)=M(((J-1)*N)+I)
MD(((J-2)*(N-1))+I-1)=MD(((J-1)*N)+I)
MP(((J-2)*(N-1))+I-1)=MP(((J-1)*N)+I)
K(((J-2)*(N-1))+I-1)=K(((J-1)*N)+I)
70  CONTINUE
RETURN
END

```

```

SUBROUTINE PARTITION(M,MD,MP,K,f,q,qd)
C%
C      INPUT  -- M,MD,MP,K,f,q,qd
C      OUTPUT -- M,MD,K,f(indices are renumbered)
C%
C*****
C      This subroutine is used to partition a mixed set of *
C      dynamic motion equations into two sets of equivalent *
C      motion equations in which the rigid and elastic body *
C      governing equations are separated as shown below. *
C      .. .. *
C      { Mrr * Lamd + Mre * d = Fr *
C      { .. .. *
C      { Mes * d + Mee * d + Kee * d = Fes *
C      *
C      The coefficient matrix Mes and the equivalent *
C      force vector Fes are assembled after executing *
C      this subroutine. The rest matrices Kee and *
C      *
C      Mee remain unchanged and correspond to their *
C      subparts in the original matrices K and MD. The *
C      *
C      indices of Mes, Mee, Kee, and Fes are renumbered. *
C      X, an intermediate matrix variable, is defined as *
C      -1 *
C      X = Mer * Mrr *
C*****
C%
C      INCLUDE '/home/jiechi/newone/program/parameter1.f'
C      INCLUDE '/home/jiechi/newone/program/parameter3.f'
C      INCLUDE '/home/jiechi/newone/program/parameter4.f'
C%
C      INTEGER I,J,I1,IX(ITH-1),Imax,Jmax
C      * ,INDEX,INDEXO,INDEXI,INDEXJ,INDEXK
C%
C      DOUBLE PRECISION M(L,L),MD(L,L),MP(L,L),K(L,L),f(L)
C      * ,q(N),qd(N),y(ITH-1),b(ITH-1),A(ITH-1,ITH-1)
C      * ,AA(ITH-1,ITH-1),DUM(ITH-1)
C%
C*****
C      Note: Dimensions of q and qd are really L *
C      instead of N. But they are defined *
C      as N for the reason of consequential *
C      use of the subroutines IMPHASE, *
C      EXPHASE, and CORRECTOR. *
C*****
C%
C*** clear zero start *****C

```

```

      DO 10 J=1,ITH-1
      y(J)=0.0
      b(J)=0.0
      DUM(J)=0.0
      DO 10 I=1,ITH-1
      A(I,J)=0.0
      AA(I,J)=0.0
10      CONTINUE
C*** clear zero end *****C
C==== to perform X =====
C**** to perform submatrix A ****
      DO 15 J=1,ITH-1
      DO 15 I=1,ITH-1
      A(I,J)=M(I,J)
15      CONTINUE
C*****
      DO 30 J=1,N-ITH
C**** to perform subvector b ****
      DO 20 I=1,ITH-1
      b(I)=M(I,J+ITH-1)
20      CONTINUE
C*****
      CALL GAUSS(A,ITH-1,IX,AA,DUM) !solve --> [A]*{y}={b}
      CALL SOLVE(AA,ITH-1,IX,b,y) !y is a varying vector
      DO 30 I=1,ITH-1
      M(J+ITH-1,I)=y(I)
30      CONTINUE
C=====
C**** to perform Fr *****
      DO 50 I=1,ITH-1
      DO 40 J=1,ITH-1
      f(I)=f(I)-(MD(I,J)+MP(I,J))*qd(J)
40      CONTINUE
      DO 50 J=ITH,L
      f(I)=f(I)-(MD(I,J)+MP(I,J))*qd(J)-K(I,J)*q(J)
50      CONTINUE
C*****
C**** to perform Fes *****
      DO 70 I=ITH,L
      DO 60 J=1,ITH-1
      f(I)=f(I)-(MD(I,J)+MP(I,J))*qd(J)-M(I,J)*f(J)
60      CONTINUE
      DO 70 J=ITH,L
      f(I)=f(I)-MP(I,J)*qd(J)
70      CONTINUE
C*****
C**** to perform Mes *****
      DO 80 J=ITH,L

```

```

DO 20 I=ITH,L
DO 80 I1=I,ITH-1
M(I,J)=M(I,J)-M(I,I1)*M(I1,J)
80 CONTINUE !Mes is stored in Mee
C*****
C**** index renumbering for M,MD,K,f *****
      Jmax=AIN(TFLOAT((N-ITH)**2/L))
      DO 90 J=1,Jmax+1
      Imax=L-AIN(TFLOAT(J/(Jmax+1)))
      *      *(L-MOD((N-ITH)**2,L))
      DO 90 I=1,Imax
      INDEX=L*(J-1)+I
      INDEXK=AIN(TFLOAT((INDEX-1)/(N-ITH)))
      INDEXO=(N+INDEXK)*(ITH-1)
      INDEX=INDEX+INDEXO
      INDEXJ=ITH+INDEXK
      INDEXI=INDEX-L*(INDEXJ-1)
      M(I,J)=M(INDEXI,INDEXJ)
      MD(I,J)=MD(INDEXI,INDEXJ)
      K(I,J)=K(INDEXI,INDEXJ)
90 CONTINUE
      DO 100 I=1,N-ITH
      f(I)=f(I+ITH-1)
100 CONTINUE
C*****
      RETURN
      END

```

```

      SUBROUTINE IMPHASE(M,C,K,f,q,qd,qdd)
C%
C      INPUT  -- M,C,K,f,q,qd,qdd
C      OUTPUT -- q,qd,qdd (d's are updated)
C              (Lamd's are unchanged)
C%
C*****
C      To solve the following 2nd order differential *
C      equations using Newmark method *
C      .. *
C      
$$M * \ddot{q} + C * \dot{q} + K * q = f$$
 *
C      *
C      Initial conditions are provided. *
C*****
C%
C      INCLUDE '/home/jiechi/newone/program/parameter1.f'
C      INCLUDE '/home/jiechi/newone/program/parameter3.f'
C      INCLUDE '/home/jiechi/newone/program/parameter4.f'
C%
C      INTEGER I,J,IX(N-ITH),LS
C%
C      PARAMETER(LS=N-ITH)
C%
C      DOUBLE PRECISION M(LS,LS),C(LS,LS),K(LS,LS),f(LS)
C      * ,q(N),qd(N),qdd(N),Mout(LS,LS),DUM(LS),BETA,GAMA
C      * ,DELT,X1,X2,X3
C%
C      COMMON/COMNM/BETA,GAMA,DELT      !<--- input(from input)
C%
C*****
C      Note: The dimensions of q, qd, and qdd are *
C      really L instead of N. But they are *
C      defined as N for the reason of *
C      consequential use of the subroutines *
C      EXPHASE AND CORRECTOR. *
C*****
C%
C      X1=0.0D0
C      X2=0.0D0
C      X3=0.0D0
C      DO 10 J=1,LS
C      DUM(J)=0.0D0
C      DO 10 I=1,LS
C      Mout(I,J)=0.0D0
10    CONTINUE
C*** adding damping *****
C      DO 15 J=1,LS      !
C      DO 15 I=1,LS      !

```

```

      C(I,J)=C(I,J)+1.0D0      !
15      CONTINUE              !
C*****
C*** renumbering q,qd,qdd *****
      DO 25 I=1,ITH-1          !befor: {q} = { Lamd | d }
      X1=q(1)                  !after: {q} = { d | Lamd }
      X2=qd(1)                 !
      X3=qdd(1)                !It's also true for qd and qdd.
      DO 20 J=1,L-1            !
      q(J)=q(J+1)              !
      qd(J)=qd(J+1)            !
      qdd(J)=qdd(J+1)          !
20      CONTINUE              !
      q(L)=X1                  !
      qd(L)=X2                 !
      qdd(L)=X3                !
25      CONTINUE              !
C*****
      DO 30 J=1,LS
      q(J)=q(J)+DELT*qd(J)+DELT**2*(0.5-BETA)*qdd(J)
      qd(J)=qd(J)+DELT*(1.0-GAMA)*qdd(J)
      DO 30 I=1,LS
      M(I,J)=M(I,J)+DELT*GAMA*C(I,J)+DELT**2*BETA*K(I,J)
30      CONTINUE
      DO 40 J=1,LS
      DO 40 I=1,LS
      f(I)=f(I)-C(I,J)*qd(J)-K(I,J)*q(J)
40      CONTINUE
      CALL GAUSS(M,LS,IX,Mout,DUM)
      CALL SOLVE(Mout,LS,IX,f,qdd)
      DO 50 I=1,LS
      q(I)=q(I)+DELT**2*BETA*qdd(I)
      qd(I)=qd(I)+DELT*GAMA*qdd(I)
50      CONTINUE
C*** rearranging q,qd,qdd *****
      X1=0.0D0                 !
      X2=0.0D0                 !
      X3=0.0D0                 !
      DO 70 I=1,ITH-1          !
      X1=q(L)                  !befor: {q} = { d | Lamd }
      X2=qd(L)                 !after: {q} = { Lamd | d }
      X3=qdd(L)                !
      DO 60 J=1,L-1            !It's also true for qd qnd qdd.
      q(N-J)=q(L-J)            !
      qd(N-J)=qd(L-J)          !
      qdd(N-J)=qdd(L-J)        !
60      CONTINUE              !
      q(1)=X1                  !

```

```
      qd(1)=X2      !  
      qdd(1)=X3     !  
70    CONTINUE      !  
C*****  
      RETURN  
      END
```

```

SUBROUTINE EXPHASE(T,q,qd,qdd)
C%
C      INPUT  -- T,q,qd,qdd
C      OUTPUT -- q,qd,qdd (Lamd's are updated)
C                  (d's are unchanged)
C%
C*****
C      This subroutine is used to execute explicit phase in      *
C      solving rigid body dynamic motion as shown below.      *
C      ..      ..      *
C      [Mrr] {Lamd} = {Fr} - [Mre] { d }      (a)      *
C      ..      ..      *
C      All the values in the coefficient matrices and the      *
C      vectors are performed at the future time step level.      *
C      Terminology: new -- future time step      *
C                  old -- current time step      *
C      ..      ..      *
C      1) Subvector "Lamd" will be replaced by the predictor      *
C      phase values as an approximation of the new values      *
C      based on Newmark algorithm. Subvector "d" is already      *
C      a new vector found in the implicit phase in solving      *
C      the elastic motion equations.      *
C      2) Calling subroutine MATRIX with the new vectors will      *
C      yield the new mass submatirces [Mrr] and [Mre] and      *
C      the new force subvector {Fr}.      ..      *
C      3) Solving Eq.(a) will give the new values of {Lamd}.      *
C*****
C%
C      INCLUDE '/home/jiechi/newone/program/parameter1.f'
C      INCLUDE '/home/jiechi/newone/program/parameter3.f'
C      INCLUDE '/home/jiechi/newone/program/parameter4.f'
C%
C      INTEGER I,J
C%
C      DOUBLE PRECISION M(N,N),MD(N,N),MP(N,N),K(N,N),f(N)
C      * ,q(N),qd(N),qdd(N),DUM(3),T,BETA,GAMA,DELT
C%
C      COMMON/COMNM/BETA,GAMA,DELT      !<--- input(from input)
C%
C      DUM(1)=0.0D0
C      DUM(2)=0.0D0
C      DUM(3)=0.0D0
C      DO 10 J=1,N
C      f(J)=0.0D0
C      DO 10 I=1,N
C      M(I,J)=0.0D0
C      MD(I,J)=0.0D0
C      MP(I,J)=0.0D0

```



```

      K(I,J)=0.0D0
10    CONTINUE
C**** Newmark predictor phase *****
      DO 20 I=1,ITH-1
        q(I)=q(I)+DELT*qd(I)+DELT**2*(0.5-BETA)*qdd(I)
        qd(I)=qd(I)+DELT*(1.0-GAMA)*qdd(I)
20    CONTINUE
C*****
      CALL LAMD(T,DUM)
C**** to renumber q,qd,qdd ****
      DO 30 I=1,N-ITH
        q(N+1-I)=q(N-I)
        qd(N+1-I)=qd(N-I)
        qdd(N+1-I)=qdd(N-I)
30    CONTINUE
      q(ITH)=DUM(1)
      qd(ITH)=DUM(2)
      qdd(ITH)=DUM(3)
C*****
      CALL MATRIX(M,MD,MP,K,f,q,qd)
      CALL MAPPING(M,MD,MP,K,f,q,qd,qdd)
      CALL CORRECTOR(M,MD,MP,K,f,q,qd,qdd)
100   FORMAT(11(1X,D10.3))
200   FORMAT(8(1X,D10.3))
      RETURN
      END

```

!%Newmark corrector
!%phase

```

      SUBROUTINE COMPAT()
C%
C      INPUT  -- INCLUDE statements
C      OUTPUT -- COMMON statements
C%
C*****
C      Boundary condition assumptions:                *
C      1) The first nodes for beams 1, 5, 3, 4, 7, and 8 *
C          are all clamped.                            *
C      2) The last nodes of beams 3, 4, 7, and 8 are also *
C          clamped. Besides,                            *
C          d3my=d4my; phi3my=phi4my; phi3mz=phi4mz=0    *
C          d7my=d8my; phi7my=phi8my; phi7mz=phi8mz=0    *
C*****
C%
C      INCLUDE '/home/jiechi/newone/program/parameter1.f'
C      INCLUDE '/home/jiechi/newone/program/parameter2.f'
C%
C      INTEGER I,J
C%
C      DOUBLE PRECISION PHIO(3,N),PHI1(NG1,N),PHI3(NG3,N)
C      * ,PHI4(NG4,N),PHI2(7,N),PHI9(11,N),PHI5(NG5,N)
C      * ,PHI7(NG7,N),PHI8(NG8,N),PHI6(7,N),PHI10(11,N)
C      * ,DP(4,4),DPP(4,4)
C%
C      COMMON/COMPHI1/PHIO,PHI1,PHI2,PHI3,PHI4,PHI5      !<--- output
C      COMMON/COMPHI2/PHI6,PHI7,PHI8,PHI9,PHI10         !<--- output
C%
C      DO 50 J=1,N
C      DO 10 I=1,3
C      PHIO(I,J)=0
10    CONTINUE
C      DO 15 I=1,7
C      PHI2(I,J)=0
C      PHI6(I,J)=0
15    CONTINUE
C      DO 20 I=1,11
C      PHI9(I,J)=0
C      PHI10(I,J)=0
20    CONTINUE
C      DO 25 I=1,NG1
C      PHI1(I,J)=0
25    CONTINUE
C      DO 30 I=1,NG3
C      PHI3(I,J)=0
30    CONTINUE
C      DO 35 I=1,NG4
C      PHI4(I,J)=0

```

```

35     CONTINUE
      DO 40 I=1,NG5
        PHI5(I,J)=0
40     CONTINUE
      DO 45 I=1,NG7
        PHI7(I,J)=0
45     CONTINUE
      DO 50 I=1,NG8
        PHI8(I,J)=0
50     CONTINUE
C*** defining BC's for the last nodes of beams 3,4,7,8 ***
      DO 55 J=1,4
        DO 55 I=1,4
          DP(I,J)=0.0D0
          DPP(I,J)=0.0D0
55     CONTINUE
      DP(1,1)=1.0D0
      DP(3,2)=1.0D0
      DP(4,3)=1.0D0
      DPP(1,1)=1.0D0
      DPP(3,4)=1.0D0
      DPP(4,3)=1.0D0
C*****
C=== for Lamd vector of each beam ===
      DO 60 I=1,3
        PHI0(I,I)=1.0D0
        PHI1(I,I)=1.0D0
        PHI2(I,I)=1.0D0
        PHI3(I,I)=1.0D0
        PHI4(I,I)=1.0D0
        PHI5(I,I)=1.0D0
        PHI6(I,I)=1.0D0
        PHI7(I,I)=1.0D0
        PHI8(I,I)=1.0D0
        PHI9(I,I)=1.0D0
        PHI10(I,I)=1.0D0
60     CONTINUE
C=====
C+++ for rigid bodies,two blocks and two tanks +++++
      DO 65 I=1,4
        PHI2(I+3,I+3+4*(Ni1-1))=1.0D0
        PHI9(I+3,I+3+4*(Ni1-1))=1.0D0
        PHI6(I+3,I+3+4*(Ni1+Ni3+Ni4-1+Ni5-1))=1.0D0
        PHI10(I+3,I+3+4*(Ni1+Ni3+Ni4-1+Ni5-1))=1.0D0
        PHI9(I+3+4,I+3+4*(Ni1+Ni3+Ni4-2))=1.0D0
        PHI10(I+3+4,I+3+4*(Ni1+Ni3+Ni4-1+Ni5+Ni7+Ni8-2))=1.0D0
65     CONTINUE
C++++

```

[illegible]

```

SUBROUTINE CORRECTOR(M,MD,MP,K,Fr,q,qd,qdd)
C%
C   INPUT  -- M,MD,MP,K,Fr,q,qd,qdd
C   OUTPUT -- q,qd,qdd (only Lamd's are updated)
C%
      INCLUDE '/home/jiechi/newone/program/parameter1.f'
      INCLUDE '/home/jiechi/newone/program/parameter3.f'
      INCLUDE '/home/jiechi/newone/program/parameter4.f'
C%
      INTEGER I,J,IX(ITH-1)
C%
      DOUBLE PRECISION M(L,L),MD(L,L),MP(L,L),K(L,L),Fr(L)
      * ,q(L),qd(L),qdd(L),Ms(ITH-1,ITH-1),Mout(ITH-1,ITH-1)
      * ,DUM(ITH-1),BETA,GAMA,DELT
C%
      COMMON/COMNM/BETA,GAMA,DELT      !<--- input(from input)
C%
      DO 5 J=1,ITH-1
      DUM(J)=0.0D0
      DO 5 I=1,ITH-1
      Ms(I,J)=0.0D0
      Mout(I,J)=0.0D0
5      CONTINUE
C**** to get new Fr and Mrr *****
      DO 20 I=1,ITH-1
      DO 10 J=1,ITH-1
      Fr(I)=Fr(I)-(MD(I,J)+MP(I,J))*qd(J)
10      CONTINUE
      DO 15 J=ITH,L
      Fr(I)=Fr(I)-(MD(I,J)+MP(I,J))*qd(J)
      *      -K(I,J)*q(J)-M(I,J)*qdd(J)
15      CONTINUE
20      CONTINUE
      DO 30 J=1,ITH-1
      DO 30 I=1,ITH-1
      Ms(I,J)=M(I,J)
30      CONTINUE
      *      define: Ms=Mrr
C*****
      CALL GAUSS(Ms,ITH-1,IX,Mout,DUM)
      CALL SOLVE(Mout,ITH-1,IX,Fr,qdd)
C**** Newmark corrector phase *****
      DO 40 I=1,ITH-1
      q(I)=q(I)+DELT**2*BETA*qdd(I)
      qd(I)=qd(I)+DELT*GAMA*qdd(I)
40      CONTINUE
C*****
      RETURN
      END

```

```

      SUBROUTINE POSTPRO(Li,Lj,qj,qdj,qddj,Ti,Pj,Vj,Aj)

C%
C      INPUT  -- Li,Lj,qj,qdj,qddj,Ti
C      OUTPUT -- Pj,Vj,Aj
C%
C*****
C      To calculate tank kinematic values of Pj, Vj,  *
C      and Aj relative to the inertial frame.          *
C                                                         *
C      Pj=[Te]*Pjo                                     *
C      Vj=[Te]*Vjo                                     *
C      Aj=[Te]*Ajo                                     *
C                                                         *
C      where Pjo, Vjo, and Ajo are the corresponding  *
C      values relative to the moving frame, and [Te]  *
C      is a transformation matrix between the inertial *
C      frame and the moving frame.                     *
C*****
C%
      INTEGER I,J,I1,I2,J1

C%
      DOUBLE PRECISION Te(3,3),Te1(3,3),Te2(3,3),Te3(3,3)
      * ,TM1(3,3),TM2(3,3),TM3(3,3),TM4(3,3),TM5(3,3)
      * ,Tmd1(3,3),Tmd2(3,3),Tmd4(3,3),Tmd5(3,3)
      * ,TMdd1(3,3),TMdd2(3,3),TMdd4(3,3),TMdd5(3,3)
      * ,Ti(3,3),Tab(3,3),Tj(3,3),Tdab(3,3),Tdj(3,3)
      * ,Tddab(3,3),Tddj(3,3),qj(11),qdj(11),qddj(11)
      * ,W(3),Wd(3),Ws(3,3),Wds(3,3),P1(3),P2(3),P3(3),P4(3)
      * ,Pd2(3),Pd3(3),Pdd2(3),Pdd3(3),TEMP1(3),TEMP2(3,3)
      * ,THETA,THETAd,THETAdd,Li,Lj,Lus,La,Rt,NT(3,3),ND(3,3)
      * ,Pj(3),Vj(3),Aj(3)

C%
      COMMON/COM4/Lus                                !<--- input(input)
      COMMON/COM7/La,Rt                              !<--- input(input)
      COMMON/COM8/NT,ND                             !<--- input(matrix)
      COMMON/COM10/P3,P4,TM1,TM2,TM3,TM4,TM5,TaB    !<--- output(gforout)
      COMMON/COM12/Te                                !<--- output(gforout)

C%
      DO 10 J=1,3
      W(J)=0.0D0
      Wd(J)=0.0D0
      TEMP1(J)=0.0D0
      Pj(J)=0.0D0
      Vj(J)=0.0D0
      Aj(J)=0.0D0
      DO 10 I=1,3
      Te(I,J)=0.0D0
      Te1(I,J)=0.0D0

```

```

Te2(I,J)=0.0D0
Te3(I,J)=0.0D0
TM1(I,J)=0.0D0
TM2(I,J)=0.0D0
TM3(I,J)=0.0D0
TM4(I,J)=0.0D0
TM5(I,J)=0.0D0
TMd1(I,J)=0.0D0
TMd2(I,J)=0.0D0
TMd4(I,J)=0.0D0
TMd5(I,J)=0.0D0
TMdd1(I,J)=0.0D0
TMdd2(I,J)=0.0D0
TMdd4(I,J)=0.0D0
TMdd5(I,J)=0.0D0
Tab(I,J)=0.0D0
Tj(I,J)=0.0D0
Tdab(I,J)=0.0D0
Tdj(I,J)=0.0D0
Tddab(I,J)=0.0D0
Tddj(I,J)=0.0D0
TEMP2(I,J)=0.0D0
10 CONTINUE
C%
THETA=(qj(9)-qj(11))/Lj
THETAd=(qdj(9)-qdj(11))/Lj
THETAdd=(qddj(9)-qddj(11))/Lj
Te1(1,1)= DCOS(qj(3))
Te1(1,2)=-DSIN(qj(3))
Te1(2,1)=-Te1(1,2)
Te1(2,2)= Te1(1,1)
Te1(3,3)= 1.0D0
Te2(1,1)= DCOS(qj(1))
Te2(1,3)= DSIN(qj(1))
Te2(2,2)= 1.0D0
Te2(3,1)=-Te2(1,3)
Te2(3,3)= Te2(1,1)
Te3(1,1)= 1.0D0
Te3(2,2)= DCOS(qj(2))
Te3(2,3)=-DSIN(qj(2))
Te3(3,2)=-Te3(2,3)
Te3(3,3)= Te3(2,2)
TM1(1,1)= DCOS(qj(7))
TM1(1,3)= DSIN(qj(7))
TM1(2,2)= 1.0D0
TM1(3,1)=-TM1(1,3)
TM1(3,3)= TM1(1,1)
TMd1(1,1)=-DSIN(qj(7))*qdj(7)

```

```

TMd1(1,3)= DCOS(qj(7))*qdj(7)
TMd1(3,1)=-TMd1(1,3)
TMd1(3,3)= TMd1(1,1)
TMdd1(1,1)=-DCOS(qj(7))*qdj(7)**2-DSIN(qj(7))*qddj(7)
TMdd1(1,3)=-DSIN(qj(7))*qdj(7)**2+DCOS(qj(7))*qddj(7)
TMdd1(3,1)=-TMdd1(1,3)
TMdd1(3,3)= TMdd1(1,1)
TM2(1,1)= DCOS(qj(5))
TM2(1,2)=-DSIN(qj(5))
TM2(2,1)=-TM2(1,2)
TM2(2,2)= TM2(1,1)
TM2(3,3)= 1.0D0
TMd2(1,1)=-DSIN(qj(5))*qdj(5)
TMd2(1,2)=-DCOS(qj(5))*qdj(5)
TMd2(2,1)=-TMd2(1,2)
TMd2(2,2)= TMd2(1,1)
TMdd2(1,1)=-DCOS(qj(5))*qdj(5)**2-DSIN(qj(5))*qddj(5)
TMdd2(1,2)= DSIN(qj(5))*qdj(5)**2-DCOS(qj(5))*qddj(5)
TMdd2(2,1)=-TMdd2(1,2)
TMdd2(2,2)= TMdd2(1,1)
TM3(1,3)= 1.0D0
TM3(2,2)= 1.0D0
TM3(3,1)=-1.0D0
TM4(1,1)= DCOS(qj(10))
TM4(1,3)= DSIN(qj(10))
TM4(2,2)= 1.0D0
TM4(3,1)=-TM4(1,3)
TM4(3,3)= TM4(1,1)
TMd4(1,1)=-DSIN(qj(10))*qdj(10)
TMd4(1,3)= DCOS(qj(10))*qdj(10)
TMd4(3,1)=-TMd4(1,3)
TMd4(3,3)= TMd4(1,1)
TMdd4(1,1)=-DCOS(qj(10))*qdj(10)**2-DSIN(qj(10))*qddj(10)
TMdd4(1,3)=-DSIN(qj(10))*qdj(10)**2+DCOS(qj(10))*qddj(10)
TMdd4(3,1)=-TMdd4(1,3)
TMdd4(3,3)= TMdd4(1,1)
TM5(1,1)= 1.0D0
TM5(2,2)= DCOS(THETA)
TM5(2,3)=-DSIN(THETA)
TM5(3,2)=-TM5(2,3)
TM5(3,3)= TM5(2,2)
TMd5(2,2)=-DSIN(THETA)*THETAAd
TMd5(2,3)=-DCOS(THETA)*THETAAd
TMd5(3,2)=-TMd5(2,3)
TMd5(3,3)= TMd5(2,2)
TMdd5(2,2)=-DCOS(THETA)*THETAAd**2-DSIN(THETA)*THETAAdd
TMdd5(2,3)= DSIN(THETA)*THETAAd**2-DCOS(THETA)*THETAAdd
TMdd5(3,2)=-TMdd5(2,3)

```



```

TMdd5(3,3)= TMdd5(2,2)
P1(1)=0.0D0
P1(2)=0.0D0
P1(3)=Lus
P4(1)=Rt
P4(2)=0.0D0
P4(3)=Rt
P2(1)=Li
P2(2)=qj(4)
P2(3)=qj(6)
Pd2(1)=0.0D0
Pd2(2)=qdj(4)
Pd2(3)=qdj(6)
Pdd2(1)=0.0D0
Pdd2(2)=qddj(4)
Pdd2(3)=qddj(6)
P3(1)=La
P3(2)=qj(8)
P3(3)=0.5D0*(qj(9)+qj(11))
Pd3(1)=0.0D0
Pd3(2)=qdj(8)
Pd3(3)=0.5D0*(qdj(9)+qdj(11))
Pdd3(1)=0.0D0
Pdd3(2)=qddj(8)
Pdd3(3)=0.5D0*(qddj(9)+qddj(11))

C%
DO 30 J=1,3
DO 30 I=1,3
DO 30 J1=1,3
DO 30 I1=1,3
Te(I,J)=Te(I,J)+Te1(I,J1)*Te2(J1,I1)*Te3(I1,J)
DO 30 I2=1,3
Tab(I,J)=Tab(I,J)+Ti(I,J1)
*      *TM1(J1,I1)*TM2(I1,I2)*TM3(I2,J)
Tdab(I,J)=Tdab(I,J)+Ti(I,J1)*
*      (TMd1(J1,I1)*TM2(I1,I2)
*      +TM1(J1,I1)*TMd2(I1,I2))*TM3(I2,J)
Tddab(I,J)=Tddab(I,J)+Ti(I,J1)*
*      (TMdd1(J1,I1)*TM2(I1,I2)
*      +2.0D0*TMd1(J1,I1)*TMd2(I1,I2)
*      +TM1(J1,I1)*TMdd2(I1,I2))*TM3(I2,J)
30 CONTINUE
DO 35 J=1,3
DO 35 I=1,3
DO 35 J1=1,3
DO 35 I1=1,3
Tj(I,J)=Tj(I,J)+Tab(I,J1)      *TM4(J1,I1)*TM5(I1,J)
Tdj(I,J)=Tdj(I,J)+Tdab(I,J1)  *TM4(J1,I1)*TM5(I1,J)

```

```

*           +Tab(I,J1)*(Tmd4(J1,I1)*TM5(I1,J)
*           +TM4(J1,I1)*Tmd5(I1,J))
Tddj(I,J)=Tddj(I,J)+Tddab(I,J1)*TM4(J1,I1)*TM5(I1,J)
*           +2.0D0*Tdab(I,J1)*(Tmd4(J1,I1)*TM5(I1,J)
*           +TM4(J1,I1)*Tmd5(I1,J))
*           +Tab(I,J1)*(2.0D0*Tmd4(J1,I1)*Tmd5(I1,J)
*           +TMdd4(J1,I1)*TM5(I1,J)
*           +TM4(J1,I1)*TMdd5(I1,J))
35  CONTINUE
DO 40 J=1,3
DO 40 I=1,3
W(J)=W(J)+NT(J,I)*qdj(I)
Wd(J)=Wd(J)+ND(J,I)*qdj(I)+NT(J,I)*qddj(I)
TEMP1(J)=TEMP1(J)+Ti(J,I)*P2(I)+Tab(J,I)*P3(I)
*           +Tj(J,I)*P4(I)
40  CONTINUE
DO 45 I=1,3
DO 45 J=1,3
Pj(I)=Pj(I)+Te(I,J)*(P1(J)+TEMP1(J))
45  CONTINUE
CALL SKEW(Ws,W)
DO 50 I=1,3
DO 50 I1=1,3
DO 50 J1=1,3
Vj(I)=Vj(I)+Te(I,I1)*(Ws(I1,J1)*Pj(J1)+Ti(I1,J1)*Pd2(J1)
*           +Tab(I1,J1)*Pd3(J1)+Tdab(I1,J1)*P3(J1)
*           +Tdj(I1,J1)*P4(J1))
TEMP2(I,I1)=TEMP2(I,I1)+Ws(I,J1)*Ws(J1,I1)
50  CONTINUE
CALL SKEW(Wds,Wd)
DO 55 I=1,3
DO 55 I1=1,3
DO 55 J1=1,3
Aj(I)=Aj(I)+Te(I,I1)*
*           (Ti(I1,J1)*Pdd2(J1)+Tab(I1,J1)*Pdd3(J1)
*           +2.0D0*Tdab(I1,J1)*Pd3(J1)+Tddab(I1,J1)*P3(J1)
*           +Tddj(I1,J1)*P4(J1)+Ws(I1,J1)*Vj(J1)*2.0D0
*           +(Wds(I1,J1)-TEMP2(I1,J1))*Pj(J1))
55  CONTINUE
RETURN
END

```

```

      INTEGER N
      PARAMETER(N=19)
C      N=4*(Ni1+Ni3+Ni4+Ni5+Ni7+Ni8-2)+3

C*****
C
C      Ni's --- total number of elements for each
C                  individual flexible beam. The number
C                  of nodes are, therefore, (Ni's+1).
C      NG's --- total number of generalized coordinates
C                  for each individual flexible beam.
C
C      1) There are four degrees of freedom for each
C          node, two deflections and two rotations.
C      2) There are three rigid body rotating angles
C          in which one is spin velocity, which is
C          specified as input, and the rest two are
C          unknowns which are to be predicted.
C
C      Vector of generalized
C      coordinates for this spicific case:
C
C          T      T      T      T      T      T
C      {q }={Lamd | d11 | d31 | d41 | d34m |
C
C
C          T      T      T      T
C          | d51 | d71 | d81 | d78m }
C
C      Notes: 1) "T" denotes a transpose of a vector.
C              2) In "dij", "i", the first index, denotes
C                  the number of a beam; "j", the second
C                  index, denotes the number of a node.
C              3) The last nodes of beams 3,4,7, and 8
C                  are treated specially. Geometrical
C                  boundary conditions are applied to
C                  these nodes. "d34m" includes the
C                  combined DOF of the last nodes for
C                  beams 3 and 4. "d78m" is for beams
C                  7 and 8.
C              4) Each "dij" vector consists of four
C                  components.
C              5) The total number of generalized
C                  coordinates are 2*(4*(1+2*2))+3.
C
C*****
      INTEGER Ni1,Ni3,Ni4,Ni5,Ni7,Ni8
      INTEGER NG1,NG3,NG4,NG5,NG7,NG8
      PARAMETER(Ni1=1,Ni3=1,Ni4=1,Ni5=1,Ni7=1,Ni8=1)

```

```
PARAMETER(NG1=4*(Ni1+1)+3,NG3=4*(Ni3+1)+3,NG4=4*(Ni4+1)+3)  
PARAMETER(NG5=4*(Ni5+1)+3,NG7=4*(Ni7+1)+3,NG8=4*(Ni8+1)+3)
```

```
INTEGER L  
PARAMETER(L=N-1)
```

```
INTEGER ITH  
PARAMETER(ITH=3)
```