The Pennsylvania State University

The Graduate School

Department of Mechanical Engineering

A MODULAR APPROACH TO THE DYNAMICS OF COMPLEX ROBOT SYSTEMS

A Thesis in

Mechanical Engineering

by

Clifford S. Bonaventura

Submitted in Partial Fulfillment
of the Requirements
for the Degree of

Doctor of Philosophy

August 2002

We approve the thesis of Clifford S. Bonaventura

Date of Signature

_____ _____

Kathryn W. Jablokow
Associate Professor of Mechanical Engineering
Thesis Adviser
Chair of Committee

_____ _____

H. Joseph Sommer III
Professor of Mechanical Engineering

_____ _____

Kon-Well Wang
William E. Diefenderfer Chaired Professor of Mechanical Engineering

_____ _____

David J. Cannon
Associate Professor of Industrial Engineering

_____ _____

Keith W. Buffinton
Professor of Mechanical Engineering, Bucknell University
Special Member

_____ _____

Richard C. Benson
Professor of Mechanical Engineering
Head of the Department of Mechanical Engineering

ABSTRACT

The need for a more general means of modelling and simulating the dynamics of arbitrary robot systems has developed from the increased complexity of modern robot tasks. Today, cooperating robots are used in unstructured environments such as hazardous waste remediation and space-based construction. Multiple robots are now often connected in series and/or in parallel to accomplish their goal. Long-reach flexible manipulators may require bracing to reduce oscillations and increase accuracy. Dynamic simulation of such systems is complicated by the numerous contacts made by the robots as they interact with one another and their surroundings. Most current simulation algorithms are not well suited to the treatment of multiple contacts and require substantial reprogramming when new system tasks or configurations occur. Even algorithms which are inherently suitable for systems with time-varying topologies are limited in the classes of contacts which they allow.

This dissertation presents the development and validation of a Modular Robot Dynamic Simulation (MRDS) algorithm. In response to these increasing complexities exhibited by modern, multi-robot systems, this algorithm has been developed to be as general as possible and, as a result, is capable of handling serial, parallel, and hybrid series/parallel topologies for both open-chain and closed-chain systems. The modularity of the approach is achieved by decomposing complex systems into individual physical components such as robots, end effector tools, payloads, obstacles, or other similar objects and devices. This modular nature allows the open chain dynamics of all system models to be performed independently and simultaneously. This is true also for each module's Forward Kinematics problem and for the computation of several position dependent inertia matrices used to incorporate con-

tact constraints. Consequently, the use of parallel processing is ideal for this algorithm, as one (or more) modules may be treated per processor. This results in increased computational efficiency and real-time capability.

Contacts in the system are represented using a general joint/contact model which describes the structure of the contact force and relative motions between a pair of contacting bodies. These contact models are used to connect modules to one another, to fixed boundaries, and to themselves in the case of a module forming or having in its design a single closed loop. With this modelling technique, the MRDS algorithm can be used for contacts that are either holonomic or non-holonomic. In addition, contact modes may be constant or vary with time.

The operational space inertia matrix is used to project the dynamics of each module to the operational frame associated with each externally contacted body. A multi-point operational space inertia matrix is also used here for the treatment of modules subject to multiple concurrent contacts. Special “relative forms” of the Jacobian matrix and various inertia matrices are also developed and used for efficiency in treating modules which have a locally closed loop.

Special attention is paid to the connection of two (or more) modules in series, as this ability to serially connect objects dynamically without the need to rederive and re-code new equations or coupling terms is new in the literature on robot dynamic simulation. The algorithm effectively formats serial connections so that they can be treated no differently than parallel connections. However, additional options available for serial connections are discussed, including special methods for Forward Kinematics and the complete elimination of

constraint violations. For the general case, constraint violation suppression is discussed and used successfully for both serial and parallel connections.

The MRDS algorithm is also shown to be applicable to constrained, structurally flexible manipulators. Issues related to second order strain and kinematic effects are discussed, including appropriate linearization schemes. These linearization and second-order modelling issues have been explored in other works for open chain systems. Here, these topics are examined for both open loop and closed loop constraints and for their place in the operational space dynamic formulation and the MRDS algorithm.

Validation of the modular algorithm is achieved through the simulation of various multi-module, structurally flexible systems. These include parallel cooperations, serial cooperations, base-excitation, and macro/mini formulations. Simulation predictions compare extremely well with those of Working Model® 2D. Results from the example systems also demonstrate the success of the constraint violation suppression system and the effect of the chosen linearization schemes on open and closed loop operations.

With the features and capabilities described, the MRDS algorithm is especially well suited for the constrained motion dynamic simulation of complex, multi-robot, reconfigurable systems like those used in space applications. The modular nature allows alternate configurations of the same system components to be simulated quickly and easily without the need to derive new dynamic equations or coupling terms. Nor is there any need to make substantial changes to existing simulation coding. The MRDS algorithm is also computationally efficient, open-ended, and general, making it particularly valuable for the treatment of the increasingly more complex robot systems in use today.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGEMENTS

This dissertation has been a long time in development and there are many people I would like to thank. First and foremost, I must thank my advisor, Dr. Kathryn W. Jablokow, for her guidance, patience, and support. Over the years, she has been both mentor and friend. I would also like to thank Professor Keith W. Buffinton for his invaluable insight and enthusiastic support. I am also grateful to Dr. H. Joseph Sommer III, Dr. David J. Cannon, and Dr. Kon-Well Wang for their time and effort in support of this work.

I also wish to thank my wife, family, friends, and co-workers for their continuous encouragement and belief in me. I would never have been able to complete this work without their unwavering support. A second note of thanks goes to my wife for her love, patience, and understanding.

I would like to thank the people at Sandia National Laboratories for providing me the opportunity to work there as a graduate student intern in 1995 and again in 1996. Lastly, I would like to acknowledge the National Science Foundation for their financial support through the Graduate Research Traineeship.

For my wife and parents

Chapter 1 INTRODUCTION

In recent years, there has been a significant rise in the demand for robots to perform increasingly more complex tasks. It is now quite common for multiple robots to work cooperatively to accomplish their task. In doing so, there can be many different contact conditions within the total robot system. Specifically, contacts can, at any time, be made or broken between the participating robots, as well as between the robots and the surrounding environment.

Dynamic simulation of such systems is an important, if not essential tool used for design, testing, optimization, and operator training. For example, with space-based robot systems, the need to examine design variations and alternate system configurations mandates the use of dynamic simulation. To carry out such investigations on real systems subject to earth's gravity would be prohibitively costly, if not impossible. Dynamic simulation is often the only practical means of developing and enhancing these complex multi-robot systems.

Current algorithms for dynamic simulation, however, do not readily support the possibility of significant changes in *system topology*[1]. In other words, when contact conditions between components of a complex system are altered, new equations of motion and new simulation programming are usually required to account for the change in dynamic behavior. Most simulation programs developed for a multi-robot system of a specific topology are of little value when the same robots are rearranged to form a new system topology.

1. *System topology* refers to the configuration in which multiple robots interact with one another and their environment. These formations can include tree-structures, closed chains, hybrids of series and parallel, etc.

In order to support the dynamics of reconfigurable systems and those with changing, time-varying contact conditions, a new modular approach to dynamic simulation has been developed and given the name MRDS (Modular Robot Dynamic Simulation). The modularity of this method stems from the decomposition of complex systems into individual subsystems. The subsystems of interest in this work are robots, but may also consist of tools, payloads, obstacles, or other similar objects, devices, and machines. Essentially these subsystems, or *modules*, are any dynamically significant structures with their own equations of motion and their own unique functions in the overall system. A single module could be as complex as the robot arm on the space shuttle or as simple as a standard test mass[2].

The modular nature of this algorithm allows an analyst to simulate alternate configurations of system components without the rederivation of the new system's equations of motion and without the need to rewrite substantial sections of simulation coding. Most efficient robot dynamic simulations are recursive and limited to tree-structured systems. An alteration in the connectivity of the "branches" will typically require an entirely new simulation code with new equations of motion.

With the ever increasing speed of modern computers, it is the efficiency of the analyst that now deserves more attention. While a computationally efficient algorithm will always be advantageous in dynamic simulations, it is also desirable to minimize the burden on the developer. The algorithm presented here is intended to meet both of these needs: computational efficiency and a reduced workload for the analyst when simulating alternate configurations of the same fundamental system components.

---

2. Although it is possible, an individual link in a serial chain would not typically be represented as an individual module.

While these two qualities will certainly be beneficial in the dynamic simulation of multi-robot systems, the algorithm must be able to accommodate the wide variety of complexities encountered in such systems to be of any value. Possible complexities in multi-robot systems have been broken down into the following three areas: 1) the complexity of an individual structure, 2) the complexity of the contacts between the structures, and 3) the complexity of the overall system topology.

The algorithm developed needs to be as general as possible in each of these three areas. After all, a computationally efficient algorithm that can be rapidly reconfigured would be of narrow interest if it were limited to rigid-link robots connected together by revolute joints forming a tree structure and subject to a single holonomic constraint. Therefore, the MRDS algorithm was developed with as much generality as possible in each of the areas described above. Details of this generality are highlighted in the following sections.

1.1 Generality of Individual Structures

In order to maximize the range of robot dynamics which can be simulated, the MRDS algorithm was developed to be applicable to robots with the following properties and operating conditions:

1) Serial and/or parallel chains of links

2) Rigid and/or flexible links

3) Open or closed chain motion

4) Fixed or freely moving bases

5) Multiple concurrent external contacts

Although this is not a comprehensive list of allowable complexities, the ability to treat individual structures with any of these features will make the resultant algorithm useful for a great many space-based and earth-bound robot systems.

1.2 Generality of Module Connections

An algorithm allowing a very wide variety of individual structures but allowing only simple constraints between them would undermine its own usefulness. Instead, contacts between adjacent modules and between modules and environmental boundaries should be as general as possible. Consequently, the MRDS algorithm makes use of a *general joint model* [58] to specify the form and structure of both the contact force and kinematic constraints. This model permits, but is not limited to, the following contact conditions:

1) Connections through single or multiple-degree-of-freedom joints

2) Connections through rigid grasps

3) Hard point or surface contacts (such as planar sliding) with or without friction

4) Six-degree-of-freedom “contacts” (for inertial connection to free-floating bases)

5) Holonomic and non-holonomic kinematic constraints

6) Fixed or time-varying contact modes

1.3 Applicable System Topologies

As pointed out above, most efficient Forward Dynamics algorithms are recursive, and many are limited to tree-structured robots. However, most modern multi-robot systems are not quite so simple. In space applications, like those described below, separate system com-

ponents are combined in a variety of ways. Limitations on system topology would greatly reduce the usefulness of the algorithm. Consequently, MRDS has been devised to permit each of the following topological variations:

1) Modules connected in series (tree-structures)

2) Modules connected in parallel (closed loops)

3) Hybrid topologies of series and parallel connections (such as bracing)

4) Connections between different bodies (links) of the same module

1.4 Examples of Complexities in Multi-Robot Systems

In order to make clear the modular nature of MRDS and its inherent versatility, several examples of multi-robot systems are described below. These examples include simple robot systems with series and/or parallel connections. Also discussed are several real world systems containing a wide variety of the complexities outlined above. The importance and purpose of the complexities in these systems are also described.

1.4.1 Series Connection

A series connection of robots is formed when one robot holds the base of a second robot in its gripper, as shown in Figure 1. Robot systems consisting solely of serial connections are said to have a tree-structured topology. This type of cooperative behavior may be used to increase the reach of the robot system or to improve dexterity. It also has the advantage of allowing manipulation that accomplishes the desired task while avoiding degenerate configurations. These same advantages can also be identified for a single redundant robot.

However, Yin and Zheng [73] point out that a series connection of two six-degree-of-freedom robots, although kinematically equivalent to a twelve-degree-of-freedom redundant robot, has added versatility in task performance. This is a result of the modular nature of connecting two robots. In a serial connection, the second robot (the one being held) can be chosen to meet the requirements of the task. If the task is changed, the second robot can, if necessary, be replaced with a more suitable robot. This feature is not feasible with the single twelve-degree-of-freedom redundant robot.

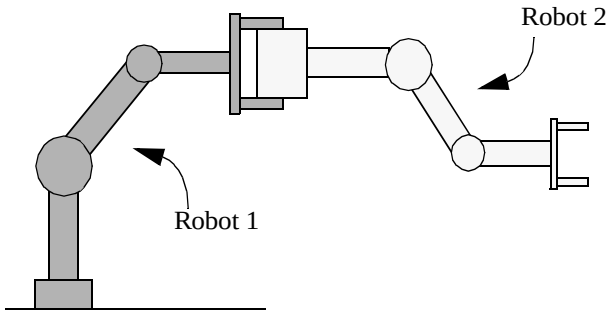

FIGURE 1. Robots Connected in Series

1.4.2 Parallel Connection

Figure 2 shows a parallel connection which is formed when two robots cooperatively hold or manipulate a common object. Parallel robot connections result in the formation of closed kinematic loops. Connections of this type are often used when manipulation of large or heavy payloads is required. In addition, parallel connections add to the rigidity of a robot system. This in turn will increase the positioning accuracy of the robots. Also contributing to accurate positioning is the noncumulative nature of actuator errors. This is not the case for serially connected mechanisms.

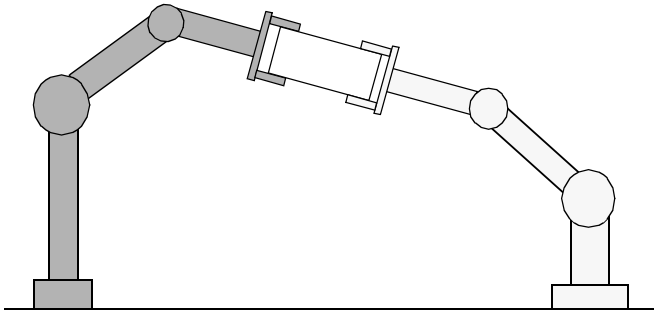

FIGURE 2. Robots Connected in Parallel

1.4.3 Hybrid Configurations

Hybrid configurations of series and parallel connections are also possible. For instance, Figure 3 shows a multi-legged robot system. In this example, all of the robot legs work in parallel with one another to carry the robot body. The main manipulator of the system is connected in series with the rest of the structure. Also note that the topology of this system changes continuously as the legs come into and leave contact with the ground.

Another example of a hybrid connection is common in systems where bracing is necessary. In some applications, use of a structurally compliant robot may be required. In an effort to reduce oscillations and increase end effector accuracy, bracing strategies may be employed. In such cases, the long, flexible robot is braced through contact with some fixed surface or by the grasp from a second, less flexible robot, as illustrated in Figure 4. This configuration results in a hybrid configuration of series and parallel link connections involving structurally compliant members [75].

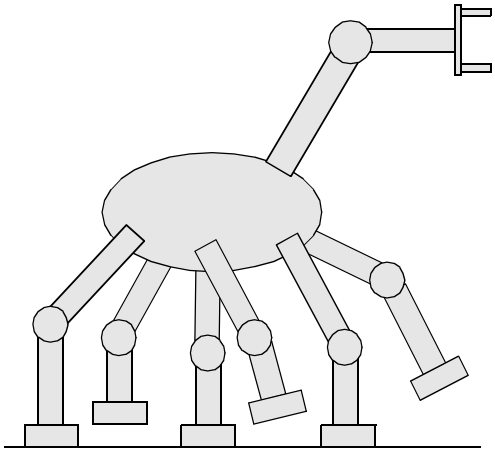

FIGURE 3. Hybrid Topology - Multi-legged Robot

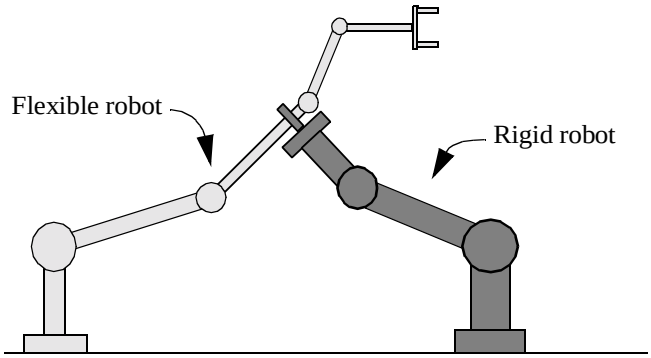

FIGURE 4. Hybrid Topology - Bracing

1.4.4 Real World Examples

Real world examples of complex robot systems can be found in the fields of hazardous waste remediation and space robots. At Sandia National Laboratories in Albuquerque, New Mexico, investigations are being made into the use of robots for clean-up of hazardous waste. A schematic of such a system is shown in Figure 5. In order to reach the bottom of the underground waste storage tanks, the robots are typically very long and slender, and,

therefore, flexible. To improve stability of the arm, bracing of the robot is often necessary. As previously mentioned, this results in hybrid configurations of series and parallel robot chains.

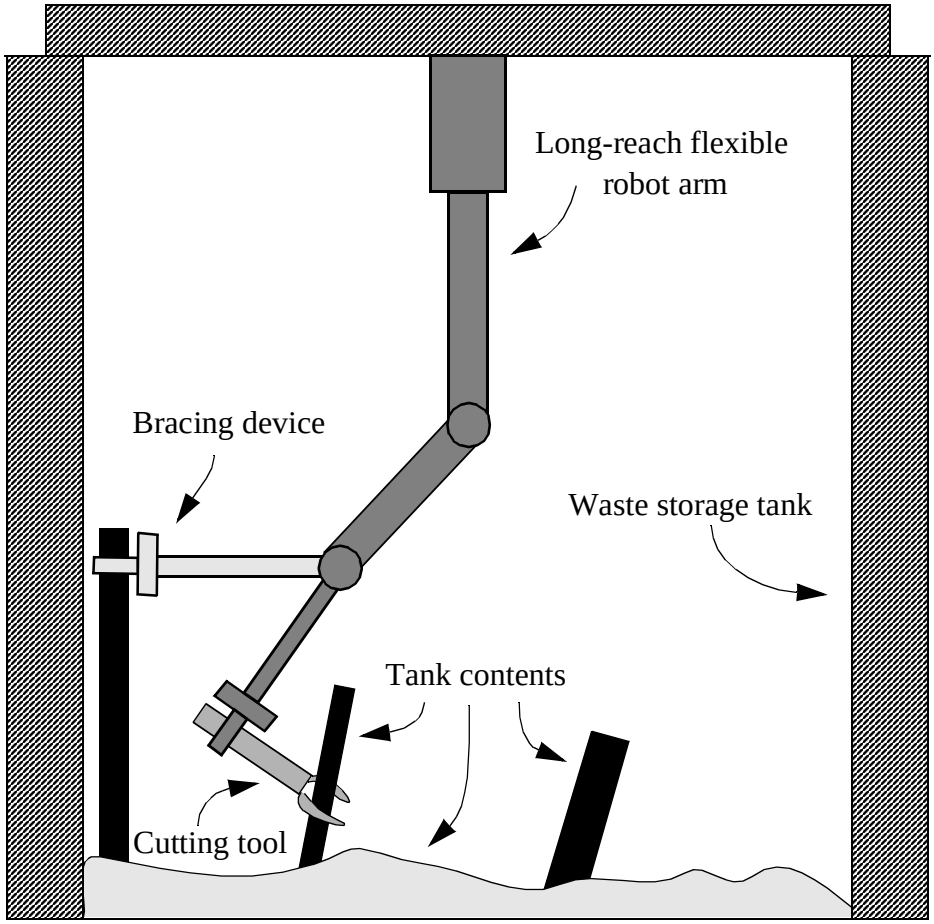

FIGURE 5. Hazardous Waste Storage Tank

Hazardous waste remediation is also an application area in which external contact conditions are subject to frequent variations. The long-reach robots will be required to use several different end effector tools such as cutting shears, sensors, excavators, vacuum tools, and drills, each of which contacts the environment in a different way. It is also possible that

a second, more dexterous manipulator may be grasped at its base to perform a delicate task. These examples demonstrate the complexity added to the system's dynamic model as a result of the differing contact conditions inherent to each tool.

Another real world example of a complex robot system is found in space station assembly. As part of the International Space Station Freedom Program, the Canadian Space Agency has developed the Space Station Remote Manipulator System (SSRMS) [45] (see Figure 6). When performing construction or payload transfer tasks, this robot is often used in conjunction with the Special Purpose Dexterous Manipulator (SPDM), wherein the two robots are joined by a series connection; that is, the SSRMS grabs the SPDM by its base so that the two robots can work together cooperatively as shown in Figure 6. The SPDM, shown more closely in Figure 7, is composed of two arms mounted on a mobile base. When these two arms are used in parallel during the cooperative manipulation of a payload, a hybrid configuration of series and parallel chains is formed. This system demonstrates the complexity which can arise from a variable system topology. Specifically, the system can, at any time, contain tree-structures and/or closed chains.

The combined SSRMS/SPDM system is also an example of a reconfigurable system in which system topology may vary from one operation to the next. This is demonstrated by the two configurations shown in Figure 8. These two systems contain the same basic components, however, each has a different arrangement of these components. This results in two distinctly different system topologies. Also adding to the complexity of these space-based systems are the presence of a free-floating base and structurally compliant members.

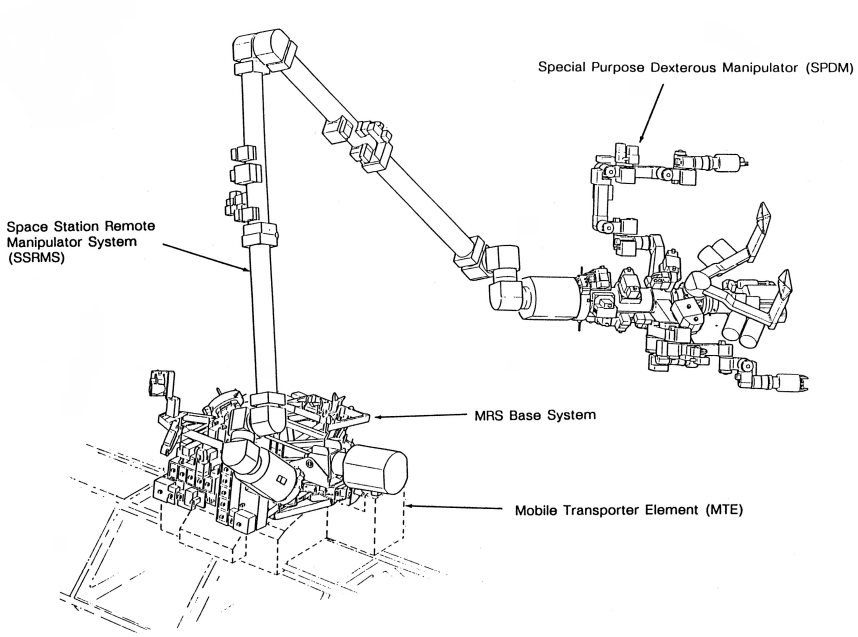

FIGURE 6. SSRMS / SPDM System [45]

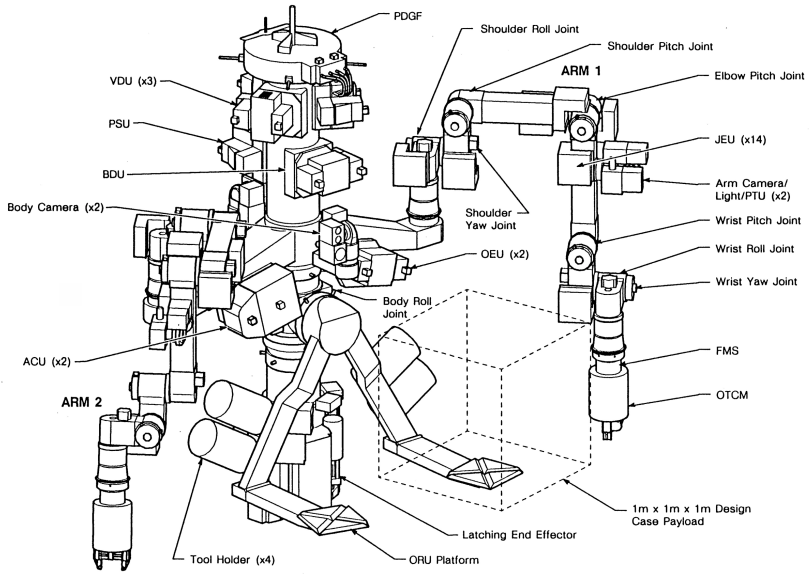

FIGURE 7. SPDM Mechanical Architecture [45]

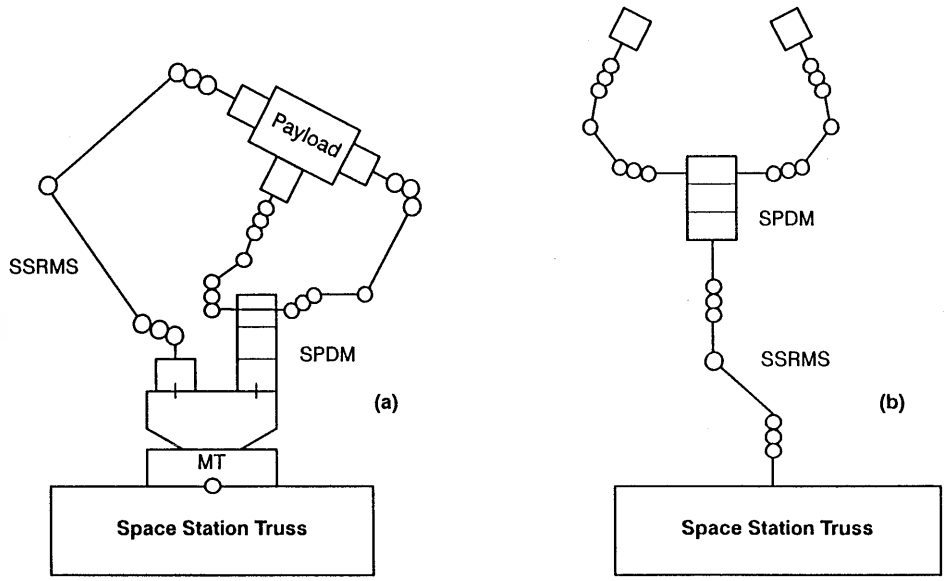

FIGURE 8. Two Alternate System Topologies for the SSRMS / SPDM System [45]

1.5 Overview of the Modular Algorithm

The modular algorithm developed in this work is based on the constrained motion algorithm of Lilly [39][40]. It has been enhanced here in a number of ways so that all of the generalities and complexities outlined in the previous sections can be treated with the resultant modular simulation algorithm.

With the new MRDS algorithm, each robot, payload, and end effector tool in the previous example systems can be treated as an individual module. In the dynamic model of these modularized system components, contact with the environment or with other modules is represented by an unknown external contact force. The structure of this contact force, as well as that of the kinematic constraint imposed by the contact, is defined through the use

of the general joint model of Roberson and Schwertassek [58]. The operational space dynamic formulation [30] is used in conjunction with the general joint model to incorporate these contact forces into the dynamic model of the system components and the entire complex system.

Operational space dynamics is a means of describing the relationship between the external forces and the Cartesian motion of a manipulator's tip (or any particular contact point). Simply put, it is the projection of the manipulator dynamics to the point of external contact. Details of the operational space formulation and the general joint model are discussed in Chapter 3.

In the remainder of this section, we will use the hazardous waste remediation system shown in Figure 5 as an example to demonstrate the structure of the new modular algorithm. The braced arm in this system is connected in series with a small dexterous manipulator, which in turn contacts the material at the bottom of the tank. Employing the MRDS approach, the system is decomposed into independent modules connected to one another using the general joint model, as shown in Figure 9. This figure depicts the connectivity (or topology) of the system components that comprise the global system. Each component is represented graphically by a square, indicating that its dynamics will be analyzed as an isolated module. Circles are used to depict the contact models which describe the type and structure of the connection existing between individual modules and between modules and the system environment.

The steps of the new simulation algorithm are divided into two primary levels for treating the dynamics of complex robot systems, as shown in Figure 10. These levels and the

flow of information between them are described briefly here. Full derivation and computational details of the algorithm are discussed in Chapter 5.

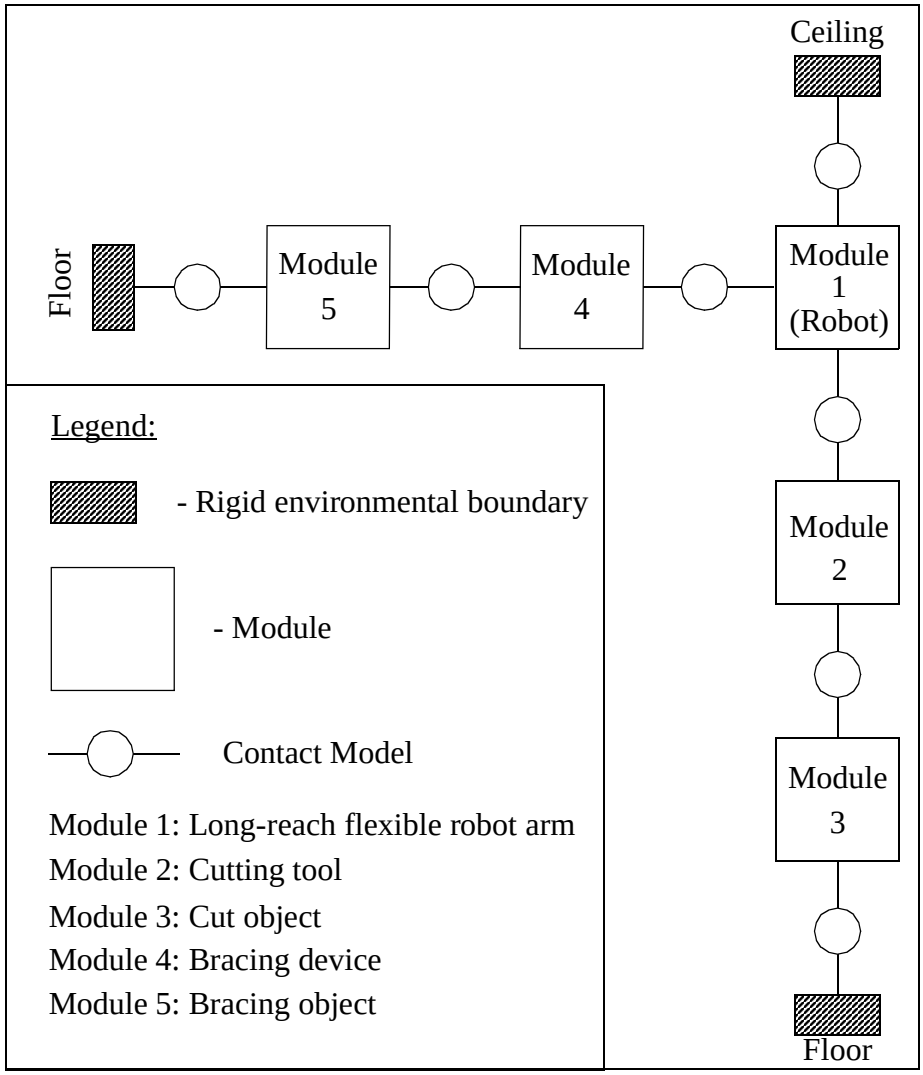

FIGURE 9. Modular Decomposition for the System Shown in Figure 5

At the modular level of the algorithm, the “modular dynamics” of each individual structure in the system are computed. Here, the structure is treated as if it is disconnected from and acting independently of all other components in the system. This allows the preliminary

dynamics of the module to be determined using open chain methodologies. Additional information related to the presence of the external contact is also calculated at this level, including open chain kinematics and the effective operational (Cartesian) inertia at each contact point.

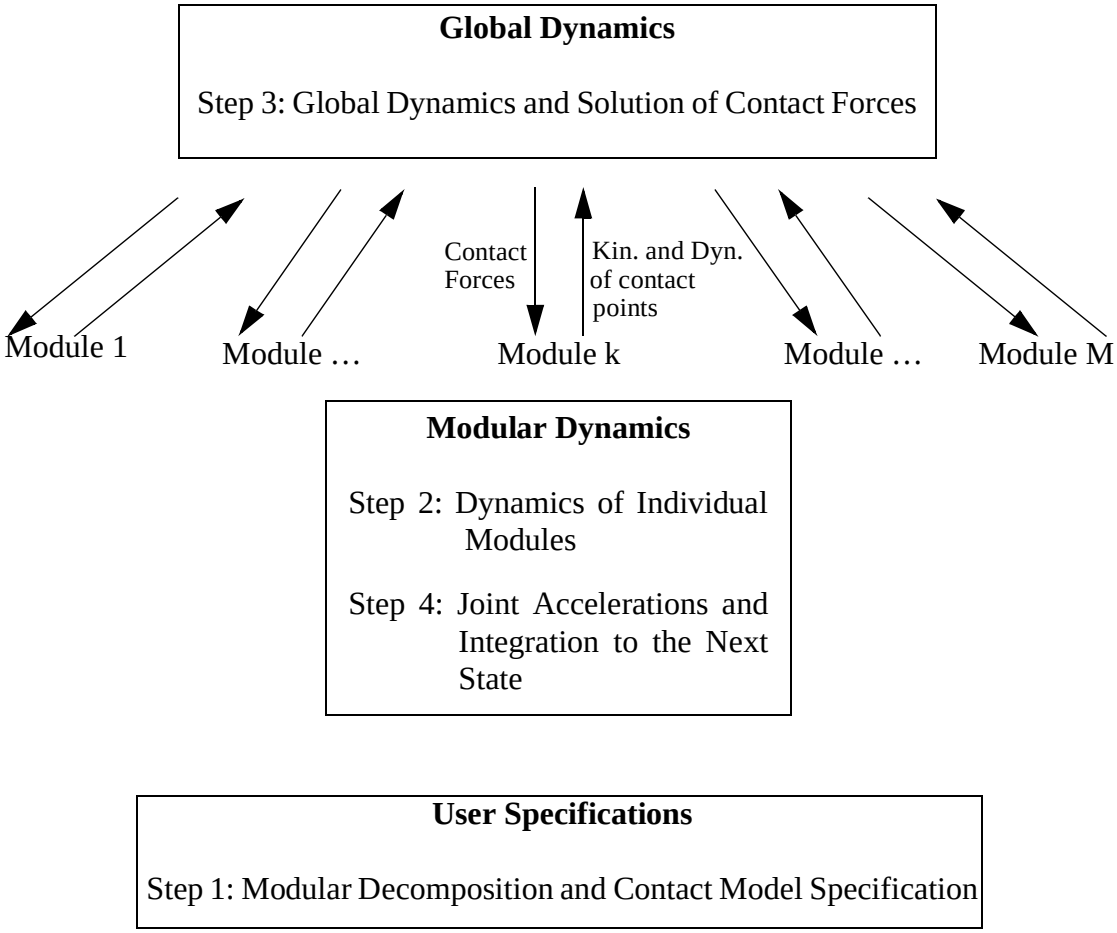

FIGURE 10. Structure of the MRDS Algorithm

With this approach, the generalized coordinates of any one module will not appear in the modular level calculations of any other module in the system. This stems from the fact

that all couplings between components at the modular level are expressed simply as an unknown contact force vector and corresponding kinematic constraint. As a result, the modular dynamics can be computed in parallel with one (or more) modules per processor. This feature can greatly enhance the computational efficiency and real-time capability of the algorithm.

When the calculations required at the modular level have been completed, the resultant information is passed to the global level of the algorithm. At this level, information computed at the modular level is used together with the contact model descriptions to assemble a system of linear equations for the entire system. These equations are then solved for the unknown components of the spatial contact forces. These force terms are passed back to the modular level where they are used to correct the open chain accelerations. The resulting closed chain accelerations are in turn integrated to produce the next state positions and rates for the module. Forward Kinematics can then be used to update all position and velocity terms throughout the system, and the simulation proceeds to the next cycle.

With this modular structuring, changes in system topology can be quickly initialized in the dynamic simulation simply by rearranging and reconnecting the system modules. Similarly, changes in contact conditions can be dealt with by implementing new or modified contact models. In either case, it is not necessary to rederive new dynamic equations or to reprogram large sections of simulation coding. This can be a tremendous asset to the analyst who needs to explore numerous simulations of reconfigurable and other similarly complex systems.

1.6 Summary and Organization

The modular algorithm presented in this work is designed to emphasize computational efficiency, while at the same time minimizing the demands on the analyst. This method is meant to be as open and general as possible, allowing multiple robots, multiple concurrent contacts with fixed or time-varying contact modes, structural flexibility, both holonomic and non-holonomic constraints, parallel and/or tree-structured topologies, and enabling rapid reconfiguration of system components to permit dynamic simulation of reconfigurable systems without significant new work by the analyst.

Prior to the presentation and derivation of this new method, a review of literature pertaining to the dynamic simulation of complex robot systems is given in Chapter 2. This includes material related to both open and closed-chain motion of mobile robots, flexible robots, and systems of complex topology.

From this study of existing methods in constrained motion dynamic simulation, it is found that the algorithm of Lilly [39][40] is the most suitable starting point to achieve the modular algorithm described herein. In Chapter 3, Lilly's algorithm is reviewed, along with the operational space dynamic formulation and the general joint model, both of which are key components of Lilly's original algorithm and this new modular methodology.

In Chapter 4, Lilly's algorithm is extended to the treatment of single contact systems with a more comprehensive range of contact conditions and system topologies than in the original algorithm or previous applications [6]. Full extension to treatment of multiple robots and multiple concurrent constraints is presented in Chapter 5, where new operational space inertia matrices are presented, and the true modular nature of the MRDS algorithm is

derived.

Chapter 6 covers a few special topics, such as singularities, constraint violations, and the subtleties of connecting two modules in series. Chapter 7 describes the application of the algorithm to robots with structural flexibility. Validation of the MRDS algorithm is the focus of Chapter 8. This was achieved using Working Model® 2D and the simulation of several planar, flexible, multi-robot, multi-contact systems.

Finally, a summary and description of future work concludes this presentation.

Chapter 2 LITERATURE REVIEW

In the development of this modular approach, many different aspects of robot dynamics are considered. These aspects include the treatment of mobile as well as fixed base robots, robots modelled with rigid and/or flexible members, and robots operating in open or closed chain configurations. In the following sections, a review of existing literature dealing with these and other relevant issues of robot dynamics is presented.

2.1 Open Chain Dynamics

Many publications exist on the subject of the dynamics of a single, fixed-based robot operating in an open chain formation. In [43], Luh et al. present an O(N) recursive method[3] for solving the Inverse Dynamics problem using a Newton-Euler approach. Using 3×3 rotation matrices and position vectors instead of 4×4 homogenous transformation matrices, Hollerbach [24] improves upon the standard $O(N^4)$ Lagrangian approach of [26][65], achieving an O(N) Lagrangian method for inverse dynamics. This method, although not as efficient as the Newton-Euler approach of Luh et al. [43], is shown by Silver [61] to be equivalent to the Newton-Euler method when rotational kinematics are represented by angular velocities.

As a result of the increased efficiency of these Inverse Dynamics algorithms, Walker and Orin [68] develop a solution method for the Forward Dynamics problem based on existing inverse dynamic techniques. The result is a recursive Newton-Euler algorithm that is

3. O(N) indicates an “order N” algorithm, in which the number of required floating point multiplications and additions varies linearly with the number of degrees of freedom of the system, N.

$O(N^3)$, where the N^3 dependency arises from the effective inversion of the system mass/ inertia matrix. Featherstone [19] develops a method that does not require inversion of the mass/inertia matrix, using the calculation of "articulated-body inertias". This results in an O(N) recursive method for the forward dynamics problem, however, it is less efficient than the method of Walker and Orin for $N < 12$. In [8], Brandl et al. develop an O(N) recursive method for forward dynamics which eliminates needless calculation of non-working constraint forces. In addition, this method allows any general connection or joint to be included in the model, unlike that of Featherstone [19], which permits only revolute and prismatic joints.

Kane and Levinson [27] develop very efficient methods for both forward and inverse dynamics. As with the method of Brandl et al. [8], calculation of the non-working constraint forces is eliminated. This approach achieves its efficiency by formulating the kinematic and dynamic equations using carefully chosen generalized velocity coordinates.

In [70], Wang and Ravani present efficient recursive algorithms for the computation of the end-effector position and orientation, as well as the computation of the Jacobian matrix. In addition, a modified version of the inverse dynamics algorithm of Luh et al. [43] is developed that incorporates simultaneous calculation of the Jacobian matrix. Wang and Ravani also present a more efficient version of the Walker and Orin method for forward dynamics. As with the inverse dynamics approach, the Jacobian matrix is calculated at the same time with little additional computational cost. Additional O(N) methods to compute various forms of the Jacobian matrix are presented by Orin and Schrader [54].

2.1.1 Mobile-base Robots

When the base member of a robot is not inertially fixed, the process of modelling and simulating the robot becomes somewhat more difficult. Disturbances in position and orientation, experienced by the mobile base due to reaction forces generated by manipulator arm motions, must be taken into account. Mobile base robots are commonly found in space-based applications. Existing literature on the dynamics of space robots has focused primarily on open chain configurations, and several publications are reviewed here.

To prevent changes in base attitude resulting from joint motions, Longman et al. [42] propose the use of reaction wheels to keep the space manipulator base orientation fixed. The reaction force and moment generated by motion of the arm are explicitly computed, however, no attention is given to translational base control. At the other extreme, Alexander and Cannon [1] discuss the use of jet thrusters for translational base control such that the manipulator tip is always kept within reach of its desired location.

A novel concept for simplifying the kinematic and dynamic equations of a space robot is proposed by Vafa and Dubowsky [67]. This method, known as the Virtual Manipulator approach, involves modelling the space manipulator system as a fixed base manipulator with its base located at the center of mass of the actual space robot system. The formulation of this model stems from the conservation of linear and angular momentum, which Papadopoulos and Dubowsky [56] use to show the existence of “dynamic singularities” in a space robot work space.

Conservation of momentum is also used by Umetani and Yoshida [66] to develop the *generalized Jacobian matrix*. The generalized Jacobian relates robot tip velocity to joint

rates while taking into account the motion of the base. Hence, this Jacobian contains "dynamic" terms based on the relative magnitude between base mass and inertia and the masses and inertias of the manipulator links. It is demonstrated that as the manipulator base becomes more massive, the generalized Jacobian reduces to the purely "kinematic" ground-based Jacobian matrix [66].

The generalized Jacobian is shown by Nakamura and Mukherjee [50] to be useful for purposes of path planning of a space robot trajectory. These same authors also use the generalized Jacobian in [48] for the formulation and computation of the inverse dynamics of a space manipulator. In addition, it is shown that the conditions of zero initial momentum and zero external contact force are not necessary in formulating and using the generalized Jacobian matrix[4].

Apart from space robots, a second class of mobile base manipulators includes multi-limbed or walking robots. However, robots of this nature are almost always treated as a system of parallel chains (the legs) cooperatively moving the robot base body. Therefore, literature on this class of mobile robot is reviewed in Section 2.2.2 with the reviews of other multi-robot literature.

2.1.2 Flexible Robots

To meet the demand for high-speed manipulation without the use of large, expensive, high-powered actuators, today's robots are constructed to be more lightweight than their predecessors [33]. This results in increased flexibility of the manipulator links. In order to

4. In this instance, additional terms must be added to the dynamic formulation.

achieve reliable performance of the robot controller, as well as accurate results from a dynamic simulation, this inherent structural compliance must be accounted for in the dynamic model of the robot.

Methods for modelling link flexibility usually fall into one of the following three categories [37]: 1) partial differential equations [16], 2) finite element analysis [5], or 3) the assumed modes methods [7]. Partial differential equations, although affording more accuracy in the model, are quite difficult to solve in real-time operations. Advantages of finite element approaches include the ease in modelling complex-shaped links and the simplistic application of boundary conditions. Computer methods for assembly and order-reduction, however, are quite complicated. Clearly, in the literature, the assumed-modes method is the most common for modelling of flexible robot dynamics.

The assumed modes method is applicable primarily to links with regular cross-sections. However, since most flexible robots are designed with long and slender links of regular cross-section, this is not a detrimental factor. In addition, the assumed modes method is more easily programmed than finite element methods [33]. Difficulty comes in the selection of the mode shapes, a subject investigated by various researchers [23][51][63].

Another important issue is the effects of geometric and centrifugal stiffening, also referred to as *foreshortening* [3][15][25][51][55][62][63]. These stiffening effects can be inappropriately neglected when the dynamic equations for the flexible members are linearized too soon in the derivation. This can lead to large errors or unstable behavior in a simulation of the robot. Inclusion of these effects and several associated methods for properly linearizing the open chain dynamic equations can be found in [15][55][62]. This issue

is discussed in greater detail in Chapter 6.

Dynamic equations for robots with flexible links are most commonly derived using Lagrangian methods. Book [7] develops a non-linear recursive Lagrangian dynamics formulation using 4×4 transformation matrices for both the joint kinematics and the link deformation. As Hollerbach [24] notes, the Lagrangian formulation for rigid body manipulators in its traditional form [26][65] results in differential equations which are computationally quite intensive and, therefore, less efficient than Newton-Euler approaches. It is, however, a difficult task to apply Newton-Euler techniques to a structurally flexible link.

Following the example set by Silver [61], King et al. [33] reformulate the Lagrangian approach used by Book [7], expressing rotational kinematics by angular velocities and using 3×3 transformation matrices and position vectors as opposed to 4×4 transformation matrices. The result is a computationally efficient recursive algorithm for calculating both Inverse and Forward Dynamics of a flexible manipulator. In fact, if the flexible modelling is eliminated in this method, the Inverse Dynamics algorithm becomes equivalent to that of Luh et al. [43] and the Forward Dynamics algorithm matches that of Walker and Orin [68].

2.1.3 Multiple Robots Connected in Series

Research into multi-robot systems has focused primarily on parallel connections, where two or more robots cooperatively manipulate a common load. Far less attention has been given to the series connection of robots. Several published works on the subject are presented here.

In [73], Yin and Zheng present a method for coordinating two serially connected robots using a method based on consumption of the least amount of energy. Their goal is to devise

a method for generating large Cartesian velocities of the tip while avoiding degenerate configurations. Lee and Kim [35] present the inverse kinematics of two robots connected in series. In their study, the individual robots each have three degrees of freedom and are either serial or parallel manipulators. The connected systems include the following combinations: series-series, series-parallel, parallel-series, and parallel-parallel.

In treating the dynamics of a system involving the series connection of robots, it is common practice to model the system only in joint space. This has the effect of treating the system as one (usually redundant) robot composed of the links and joints of the originally separate manipulators. In the method employed by Lew and Book [37], joint space dynamic equations for two robots connected in series are constructed using a symbolic operator software (Mathematica). Special care is taken so that many of the terms developed for each robot operating individually can be used in the model of the combined system. The symbolic development is used to generate the coupling terms needed to complete the joint space dynamic equations. These coupling terms, however, are applicable only to the series connection of the two robots involved. New coupling terms must be derived for any change in the topology of the components.

A few researchers [13][74][75] have applied the operational space dynamic formulation to systems containing two serially connected robots. These works, however, focus only on studying certain properties of the serially connected systems. For instance, Chen and Kumar [13] study the mobility and inertial characteristics of two serially connected robots, and these measures are compared to those for a single robot. Manipulability, force transmission, and resistance to disturbance are examined by Zheng [74], where these characteristics are

compared for two robots connected in series and two robots connected in parallel. Finally, in the work of Zheng and Luh [75], the operational space inertia matrix (see Section 3.1.2) is discussed and compared for robots connected in series or in parallel.

It appears that no literature is available on the use of operational space formulations for the dynamic simulation of serially connected robots. The work presented here makes this application.

2.2 Constrained Motion / Closed Chain Dynamics

Constrained motion of a robot occurs when a closed kinematic loop is formed by the robot's own topology and/or by the robot and its environment, as shown in Figure 11. Dynamic simulation of such a system is complicated by two factors: (1) the forces exerted on/by the robot end effector must be found concurrently with the generalized (i.e. joint) coordinate accelerations; and (2) the generalized coordinates originally defined for the unconstrained system no longer constitute an independent set due to the kinematic constraints imposed by the external contact.

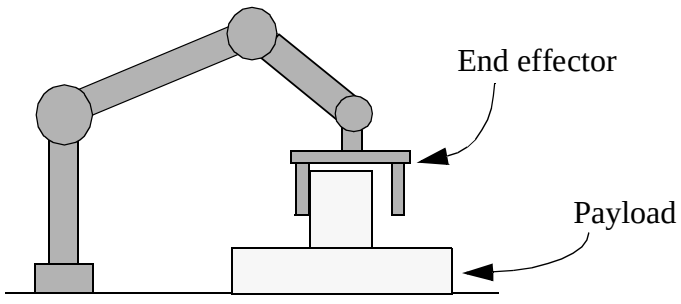

FIGURE 11. A Robot Performing a Constrained Motion Task

Research in constrained motion simulation of single closed chain robots is extensive for fixed-base systems. In general, constrained motion simulation (i.e. Forward Dynamics) involves the determination of n_c contact forces acting at the robot tip and the N joint accelerations, where n_c is the number of degrees of freedom constrained by the contact, and N, as before, is the number of single-degree-of-freedom manipulator joints. Solution methods proposed by Featherstone [20] and Orin and McGhee [53] use kinematic contact constraints to augment the robot dynamic equations of motion. The solution of such augmented equations requires the equivalent inversion of an $(N + n_c) \times (N + n_c)$ matrix. The robot dynamic equations are also augmented by McClamroch [46] using constraint space expressions for the contact force, resulting in a singular system of differential equations. A reduction approach is used for the solution of these singular systems, but this results in a very complicated functional form.

Brandl et al. extend their open chain simulation algorithm [8] to multi-body systems with kinematic loops in [9]. As before, this linear recursive method allows for many types of connections, but is not completely general. Rodriguez et al. [59] also develop a linear recursive method for the simulation of constrained robots. In their approach, a spatial operator algebra derived from filtering and smoothing theory is used to develop a compact form of the linear recursive equations. While technically sound, a fundamental physical interpretation of this approach is difficult.

Several algorithms for constrained motion dynamic simulation [2][29][39][40][59] take advantage of the operational space dynamic formulation derived by Khatib [30]. In these algorithms, the operational space inertia matrix is used to find a solution for the unknown

contact forces acting (in most cases) at the robot tip. This requires, at most, an effective 6×6 matrix inversion. This matrix inversion can be reduced further by projecting the operational space equations onto a "constraint space" [2][29][39][40], which corresponds to the Cartesian directions constrained by the closed chain contact. By using the mathematical tools and notation of Roberson and Schwertassek [58], the reduced matrix inversion is $n_c \times n_c$, where $n_c \leq 6$. Chang, et al. [12] extended the operational space formulation by developing dynamic equations which describe not only the behavior of the end effector (operational space), but also the self-motion (null space) of the robot, and the interactions between the two.

Unfortunately, developments of this kind usually result in the same complex formulation encountered by McClamroch [46]. Careful separation of the robot dynamic equations into constrained and unconstrained components, however, greatly reduces the complexity of the resultant equations, as shown in Lilly [39][40]. In fact, the O(N) algorithm developed by Lilly is the most computationally efficient closed-chain algorithm known at this time.

A dynamic model very similar to the one presented here in the context of Forward Dynamics has been used for efficient Inverse Dynamics, as well as for dynamically-decoupled force/motion control [11][21]. The models are similar in their efficient use of the operational space framework and the projection of the system's dynamic equations into the constrained space using a dual-basis contact model.

Chen [14] developed dynamic equations for a manipulator subjected to external contact forces that may include friction. A Lagrange multiplier method relates these friction forces to the contact normal force. This method then uses second order kinematic constraints im-

posed by the contact to determine the unknown contact normal force. In the work presented here, friction forces are incorporated in a similar fashion, however, the second order kinematic constraints are included in the form of a dual-basis contact model.

Second order kinematic constraints are also used in [64], where the explicit dynamic equations of motion for nonholonomically constrained mechanical systems are obtained using an extended form of D'Alembert's Principle. Like the Forward Dynamics algorithm presented in this paper, this development by Udwadia et al. relies on the determination of unconstrained dynamics with corrections resulting from the contact force.

2.2.1 Flexible Robots

Methods for incorporation and treatment of unknown external contact forces in the closed chain modelling and simulation of flexible manipulators is, in general, no different than those methods used for rigid body manipulators. However, as pointed out by Tadikonda and Singh [63], the problems encountered in open chain formulations, such as choosing appropriate mode shapes and accounting for the stiffening effects, can be much more difficult for closed chain operations. For example, boundary conditions are altered when a flexible robot comes in contact with an external body and forms a closed loop. A change in boundary conditions may diminish the accuracy of a flexible model to the point where additional or alternate mode shapes must be chosen. Similarly, errors in the link lengths that arise from the problems associated with foreshortening will result in small constraint violations for a closed chain flexible robot. These violations, if uncorrected, will grow unbounded as a simulation progresses. Similar difficulties with constrained flexible systems are explored by Haering [23] using a flexible beam cantilevered to a rotating table at its

base and attached to the same table at its tip by a spring tether.

Ider and Amirouche [25] present a simulation approach based on Kane's method [27] which uses relative coordinates for rigid body degrees of freedom and assumed mode shapes for the flexible modelling. Another approach based on Kane's method is derived by Anderson [3]. The rigid body constrained motion algorithm he develops in [2] is, here, extended to account for link flexibility. In this method, the contact forces required to enforce the kinematic constraints are determined through the use of a constraint stabilization technique.

Lew and Book [38] develop dynamic equations for a flexible manipulator that is constrained at more than one location. These equations are then transformed into two subspaces using Singular Value Decomposition of the constraint equations. These subspaces are described by the constrained and constraint-free directions of the contact. The transformed equations are then used for the purposes of controlling a braced flexible manipulator. No methods are proposed, however, for simulation of such a system.

Kim and Haug [32] extend to closed loop configurations the recursive algorithm for open chain, tree-structured, flexible manipulators presented in [31]. In this method, a joint in the chain is cut. Constraint equations and Lagrange multipliers representing cut joint forces and torques are then introduced. Dynamics of the chain are recursively projected from tip to base in a manner very similar to the approach of Featherstone [19] using articulated-body inertias. This recursive projection of the manipulator dynamics from tip to base is opposite the convention of operational space methods wherein the dynamics are projected to the robot tip.

2.2.2 Multiple Robots Connected in Parallel

Research into the dynamic simulation of complex, multi-robot systems has been mostly limited to parallel connections. The two most common types of parallel robot systems in research publications are: 1) the cooperative manipulation of a single object by two or more robots [34][44][47][49][52][71] and 2) the walking of a multi-legged robot in which each leg is treated as an individual robot acting in parallel with the others to maneuver the base body [22][52].

In the first of these two cases, Luh and Zheng [44] formulate the dynamics of two co-operating manipulators individually and relate them through the dynamics of the held object. They use a leader/follower model in an effort to eliminate motion errors between the two robots. Alternatively, Koivo and Unseren [34] obtain a joint space model of two manipulators holding a common object by combining dynamic and kinematic constraints with the equations of motion of the manipulators. Variable transformations are then used to reduce the model so that only independent coordinates are kept.

Oh and Orin [52] present a method that is applicable to both cooperative manipulation and walking robots. Their method is based on an extension of the constrained motion algorithm presented by Orin and McGhee [53]. Here, the constraint forces between the manipulator tips and the payload or between the legs and the ground are combined with the joint space dynamics of the parallel robots and the motion of the payload or base body. Freeman and Orin [22] develop a simulation algorithm for a four-legged walking robot. In this method, the closed-chain system is decoupled into a tree-structured open-chain system by using a spring-damper model of the ground interactions at the tip of each leg. The resulting open

chain structure is simulated using the algorithm of Brandl et al. [8].

Several researchers [47][49][71] make use of the operational space dynamic formulation [30] when dealing with the parallel cooperation of two or more robots manipulating a common object. Typically, in these methods, the inverse operational space inertia matrices of each individual arm are computed and used in the model of the connected system. Wen and Kreutz-Delgado [71] use this method for motion and force control, while McMillan et al. [47] develop a dynamic simulation of the system, including the possibility that one or more of the manipulators may reach a singular configuration. Murphy et al. [49] employ a similar model in which they first develop the open chain joint space dynamic equations for two manipulators operating on a six-degree-of-freedom platform. When the two arms cooperatively manipulate a payload, the dynamics of each arm are represented in the operational space formulation and combined with the dynamics of the payload.

2.3 Complex Combinations of Dynamic Properties

Only a handful of published works have been identified which deal with the dynamics of a manipulator system that is flexible, contains a mobile base, and is operated in closed-chain or constrained motion tasks. In [45], Ma et al. develop the joint space dynamic equations for the SSRMS / SPDM system described in Section 1.4.4. This is a space-based multiple manipulator system with a long-reach flexible manipulator, the SSRMS, connected in series with the SPDM. Flexibility is incorporated using finite element methods and model order reduction techniques. Closed chain operations formed by the cooperative grasp of a common object by the two arms of the SPDM are simulated by reducing the generalized

coordinates to an independent set. This is accomplished using the natural orthogonal complement of the matrix defining the kinematic constraints.

A constrained motion algorithm for the simulation of the robot arm on the space shuttle, known as the Shuttle Remote Manipulator System (SRMS), was developed by Lockheed Engineering and Sciences Company (LESC) in [41]. The dynamic model of the SRMS includes link and joint flexibility, as well as motion of the base body (the shuttle itself). For simulation of closed chain operations, a method similar to that of Freeman and Orin [22] is used in which a spring-damper model of the unknown external contact allows the decoupling of the system into an open chain configuration. Specification of the spring and damper parameters, however, is necessary for each different contact configuration investigated.

In an effort to eliminate the need for re-tuning the spring-damper values, Bonaventura and Lilly [6] develop an alternative closed chain simulation algorithm based on the methods of Lilly in [39][40]. Although rigid body simulations are accurately produced, constraint violations grow unbounded when flexible modelling is included. This was deemed the result of the simplistic Euler integration scheme used by LESC in their open-chain simulation of the SRMS. Achieving a successful application of the algorithm of Lilly to constrained flexible systems was one of the primary motivations for the work presented here.

2.4 Summary

Although a great deal of attention has been given to the dynamic simulation of robots, no single algorithm is known to be applicable to multi-robot systems containing both serial and parallel connections and subject to multiple concurrent, time-variable, non-holonomic

constraints. The constrained motion algorithm of Lilly [39][40], however, is suitable for extension to meet these needs. The evolution of Lilly's algorithm into the new Modular Robot Dynamic Simulation (MRDS) approach is the subject of the remainder of this work.

Chapter 3 BACKGROUND THEORY AND NEW NOTATION SCHEME

Before presenting the details of the MRDS algorithm for simulating complex robot dynamics, it will be helpful to put in place a notational scheme and to review key areas of background theory. This will include a brief description of the fundamentals of both joint space and operational space dynamics, as well as the general joint model developed by Roberson and Schwertassek [58]. Also in this chapter, we will review the constrained motion simulation algorithm developed by Lilly [39][40], which was the starting point for the modular algorithm developed here.

3.1 The Constrained Motion Algorithm of Lilly

As noted in Chapter 2, Lilly's constrained motion algorithm [39][40] is the most computationally efficient closed chain algorithm known at this time. In addition, it is also the most suitable for the modular architecture developed here, as it is designed to utilize the existing open chain manipulator model without the need for revision. This method incorporates the effects of the unknown constraint force into the dynamic equation of the manipulator, yet keeps terms corresponding to the unconstrained robot system grouped separately. The algorithm was first developed for simple closed chain mechanisms[5] composed solely of rigid bodies.

5. Simple closed chain mechanisms are those for which all contacts are broken by the removal of a single body. This includes systems where several fixed-base robots cooperatively grasp a central body or where a walking robot's main body is transported by several parallel robot legs.

3.1.1 Joint Space Dynamics

The development of the mathematical model begins with the joint space dynamics of a single, rigid-body robot with N degrees of freedom. The joint space formulation can be expressed by the following equation [68][6]:

$$\tau = H \cdot \ddot{q} + C + G + J^T \cdot F \tag{1}$$

where τ is an $N \times 1$ vector of joint actuator torques/forces; H is the $N \times N$ robot *joint space inertia matrix*, which is positive definite and typically symmetric[7]; q, $\dot{q}$, and $\ddot{q}$ are $N \times 1$ vectors of joint positions, velocities, and accelerations[8]; C is an $N \times 1$ vector of Coriolis and centripetal force terms; G is an $N \times 1$ vector of gravitational forces; J is the $6 \times N$ Jacobian matrix for the end effector; and F is a 6×1 [9] spatial vector of external forces/moments exerted by the robot end effector. (Note that spatial quantities, which include both linear and angular terms, will be discussed in Section 3.2.1).

As with any dynamic simulation algorithm, the joint space coordinate accelerations, $\ddot{q}$, must be determined given the joint torques, τ, and the initial joint coordinate positions and velocities, q and $\dot{q}$. Thus, Equation 1 is solved for $\ddot{q}$ as follows:

---

6. Equation 1 represents a simplified manipulator that does not consider such effects as friction, motor inertia, or link flexibility. Inclusion of these additional effects will naturally increase the complexity of the model. The added complexities of link flexibility will be addressed at length in Chapter 7.

7. The joint space inertia matrix, H, is commonly symmetric, however, it may be non-symmetric when appearing in linearized dynamic equations for a robot with structural flexibility.

8. In this work, reference to “joint” coordinates is often used in place of “generalized” coordinates, which would be more appropriate, as the algorithms presented apply to robot dynamic models based on any collection of independent variables. In addition, the relationship between the position and velocity coordinates is described by the derivative for simplicity. A more general relationship in the case of non-holonomy is accepted by the MRDS algorithm.

9. In this work, the operational space (task space) is always assumed to be six-dimensional, though the algorithms presented are suited to smaller dimensions as well.

$$\ddot{q} = H^{-1} \cdot (\tau - C - G) - (H^{-1} \cdot J^{T}) \cdot F \quad (2)$$

For open chain simulations, F, the external force vector, is assumed to be zero or known. Dynamic simulation of an open-chain robot then requires the computation of all terms in Equation 2, solution of the joint accelerations, $\ddot{q}$, and numerical integration to produce the joint positions and velocities of the next state, q and $\dot{q}$, respectively.

For closed chain simulation, however, the force vector, F, is unknown and must be found concurrently with the generalized coordinate accelerations. In Lilly's constrained motion algorithm, this process begins with an alternate form of Equation 2, as follows:

$$\ddot{q} = \ddot{q}^{open} - \Psi \cdot F \quad (3)$$

Note that the terms are grouped so that the unknown contact force vector, F, is isolated.

The first term on the right side of Equation 3, $\ddot{q}^{open}$, corresponds to the joint acceleration vector of the manipulator in an unconstrained, open-chain configuration with all external contact forces removed. Computation of $\ddot{q}^{open}$ can therefore make use of any suitable open-chain simulation algorithm (e.g. [8][68][70]).

The second term represents a correction term which accounts for the closed chain contact. The $N \times 6$ matrix, Ψ, is called the *cross space inertia matrix,* since it is used to map forces in operational space to accelerations in joint space. It is a function only of the generalized coordinates, q, and is defined as:

$$\Psi = H^{-1} \cdot J^{T} \quad (4)$$

Given the present state of the manipulator and the input joint actuator torques/forces,

the open chain terms, $\ddot{q}^{open}$ and Ψ, are completely defined. The efficient computation of Ψ is discussed in detail in [39].

To continue from Equation 3, it is necessary to account for the unknown contact force vector. As shown in various sources [2][29][39][40][59], this is best and most efficiently accomplished by expressing the robot dynamics using the joint space formulation in combination with the operational space formulation [30].

3.1.2 Operational Space Dynamics

Operational space equations for a robot are formed by expressing the dynamics of the robot relative to some point in operational space (hereafter referred to as an *operational point*). For this work, an operational point will be specified at any (and every) location at which an external contact is present. In the development of operational space dynamic equations throughout this section, however, we will choose the operational point to be the end effector, as is usually the case in operational space dynamic formulations.

Presented by Khatib for use in manipulator control using Cartesian end effector variables, the fundamental operational space dynamic equation is given by [30]:

$$F_{eff} = \Lambda \cdot \ddot{x} + \mu + \rho \tag{5}$$

where F_{eff} is the 6×1 vector of forces/moments that the robot tip exerts on the environment (including the projected contribution of τ, the applied joint torques); Λ is the 6×6 *operational space inertia matrix*; x, $\dot{x}$, and $\ddot{x}$ are the 6×1 spatial vectors of position[10],

10. As with the joint coordinates, this relationship between spatial position and velocity is permitted to be more complex than the simple time derivative. This notation is for simplicity in the equations.

velocity, and acceleration of the robot end effector; μ is the 6×1 vector of Coriolis and centripetal forces; and ρ is the 6×1 vector of gravitational forces.

As shown in Equation 5 and depicted in Figure 12, the operational space inertia matrix, Λ, relates spatial forces exerted by the end effector to the spatial end effector acceleration. This is in contrast to the more familiar joint space inertia matrix, H, through which joint torques are related to joint accelerations.

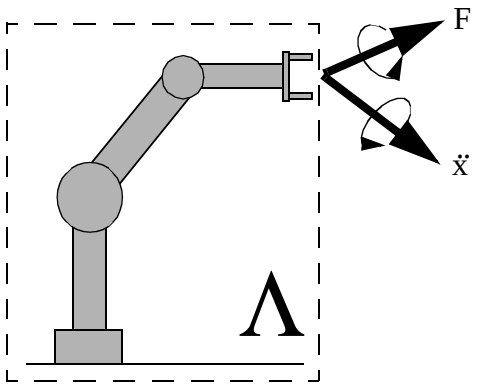

FIGURE 12. Visualization of the Operational Space Inertia Matrix

It is shown in [30] that the 6×6 operational space inertia matrix can be expressed as follows:

$$\Lambda = (J \cdot H^{-1} \cdot J^{T})^{-1} \tag{6}$$

As before, J is the $6 \times N$ Jacobian matrix, and H is the $N \times N$ joint space inertia matrix.

Given that the joint space inertia matrix is positive definite and, therefore, always invertible, the existence of the operational space inertia matrix is determined solely by the dimension and rank of the Jacobian. Specifically, for any N-degree-of-freedom manipulator, Λ is always a 6×6 matrix[11], however, it is only explicitly defined when $N \geq 6$. If

$N < 6$ or if the manipulator is in a singular configuration, then the rank of the Jacobian is less than six, and Λ cannot be explicitly determined[12].

The *inverse operational space inertia matrix*, however, is defined for all N regardless of singularity and is given by:

$$\Lambda^{-1} = J \cdot H^{-1} \cdot J^{T} \tag{7}$$

This matrix is typically symmetric[13] and is positive semi-definite. Furthermore, Λ^{-1} is positive definite, and therefore invertible, only when the Jacobian matrix, J, is of full rank. It is useful to note that it is Λ^{-1}, not Λ, which is used in Lilly's constrained motion algorithm. Its appearance in the algorithm comes from an alternate form of the operational space dynamic equations as shown in the steps below.

The more useful form of the operational space dynamic equation is generated by introducing joint space dynamics into a kinematic expression for the absolute acceleration of the contact point. This acceleration can be produced using the Jacobian matrix as follows:

$$\dot{x} = J \cdot \dot{q} \tag{8}$$

$$\ddot{x} = J \cdot \ddot{q} + \dot{J} \cdot \dot{q} \tag{9}$$

Equation 2 can be substituted into this last equation to produce the following:

$$\ddot{x} = J \cdot H^{-1} \cdot (\tau - C - G) + \dot{J} \cdot \dot{q} - (J \cdot H^{-1} \cdot J^{T}) \cdot F \tag{10}$$

---

11. Task space is assumed to be six-dimensional here for convenience. It can be less than or equal to six.
12. Unless the dimension of the task space is reduced accordingly.
13. It is symmetric whenever H is symmetric (see footnote 7 on page 36).

Equation 10 may be rewritten to produce a difference expression similar to that of Equation 3. This gives the following:

$$\ddot{x} = \ddot{x}^{open} - \Lambda^{-1} \cdot F \tag{11}$$

Again note that the unknown contact force has been isolated.

The first term on the right side of Equation 11, $\ddot{x}^{open}$, gives an expression for the acceleration of the robot end effector as if there were no spatial contact force acting at the robot tip. The second term is a correction factor to account for the presence of the unknown tip force. Described above, the 6×6 contact force coefficient matrix, Λ^{-1}, is the inverse operational space inertia matrix and is a function only of the generalized coordinates, q. As with $\ddot{q}^{open}$ and Ψ, if the present state and driving actuator torques/forces are known, $\ddot{x}^{open}$ and Λ^{-1} may be computed. Details regarding the computation of the open chain terms, $\ddot{q}^{open}$ and $\ddot{x}^{open}$, as well as the two force coefficient matrices, Ψ and Λ^{-1}, are given in [39][40].

3.1.3 The Simplified Contact Model and Solution of the Contact Force

In conjunction with the two dynamic formulations represented by Equations 3 and 11, a simplified form of the contact model of Roberson and Schwertassek [58] is used. This model specifies the structure of both the end effector acceleration and contact force vectors using vector spaces which describe the free (unconstrained) and constrained directions of the contact as follows:

$$\ddot{x} = \phi^{u} \cdot \ddot{x}^{u} + \phi^{c} \cdot \ddot{x}^{c} \tag{12}$$

$$F = \phi^{u} \cdot F^{u} + \phi^{c} \cdot F^{c} \tag{13}$$

The original, more general form of the Roberson and Schwertassek contact model will be reviewed at the end of this chapter.

In the continuing effort to solve for the contact force vector and effectively reduce the problem to that of an open chain system, Equations 12 and 13 are introduced into Equation 11. Using the orthogonal properties of the contact model vector spaces (see Section 3.3.1), the resulting equations can be placed in the following compact form:

$$B^{c} \cdot F^{c} = y^{c} \tag{14}$$

where:

$$B^{c} = (\phi^{c})^{T} \cdot \Lambda^{-1} \cdot \phi^{c} \tag{15}$$

$$y^{c} = (\phi^{c})^{T} \cdot (\ddot{x}^{open} - \Lambda^{-1} \cdot \phi^{u} \cdot F^{u}) - \ddot{x}^{c} \tag{16}$$

Equation 14 is a linear system of equations in the unknown components of the contact force vector, F^{c}. Using a linear system solver, such as LU or Singular Value Decomposition, these forces can be determined. With Equation 13, the full contact force vector, F, can then be constructed and used in Equation 3 to determine the closed chain/constrained joint space coordinate accelerations of the manipulator, $\ddot{q}$. Numerical integration of these accelerations will give the joint rates and positions which completely describe the next state of the constrained manipulator, and the dynamic simulation proceeds to the next time step.

The formulation of this constrained motion algorithm has several inherent limitations

which have, until now, not been addressed in previous publications [6][39][40]. In particular, the algorithm only allows a limited class of contacts which do not vary with time. In the following sections and in Chapter 4, we will reformulate and enhance the algorithm in a way which enables the treatment of a more comprehensive range of contact conditions. These changes will also serve as important steps in extending this algorithm to the treatment of general multi-robot, multi-contact systems, which will be demonstrated in Chapter 5. We begin with the establishment of a notational framework that will aid in clarifying all required kinematic, dynamic, and contact model terms.

3.2 Notation Scheme

The objective of this dissertation is to develop an efficient constrained motion simulation algorithm for complex robot systems. In such systems, there can be multiple robots and multiple contacts. A suitable notation scheme will be necessary to deal with the many kinematic, dynamic, and contact equations that describe the system. The following section presents this new notation scheme, including a discussion of coordinate frames and spatial notation. In addition, we will discuss the many forms of the contact force vector that will play a vital role in the development of the MRDS algorithm.

3.2.1 Spatial Notation

The spatial notation of Featherstone [20] will be used extensively throughout this work, since it allows both kinematic and dynamic equations to be expressed in a particularly concise way. In general, spatial vectors (6×1) and matrices (6×6) are defined such that the first three elements (or rows, for a matrix) are translational quantities, and the remaining

three are angular quantities [8][20]. For example, the spatial force, F, has the form $\left[f^T \; n^T\right]^T$, where f is a 3×1 vector of forces, and n is a 3×1 vector of moments.

3.2.2 Coordinate Frames

To clearly describe the kinematics and dynamics of a given contact, it will be helpful to assign several specific coordinate frames as shown in Figure 13. This model may be used for any arbitrary contact, but here it represents the contact between a robot end effector (body E) and some payload (body P). Each body has a Cartesian coordinate frame (also denoted by E and P) attached at a convenient point, and the absolute and relative position vectors of these frames are shown. Point C represents a contact point between the two bodies, and frame I is a fixed inertial reference frame.

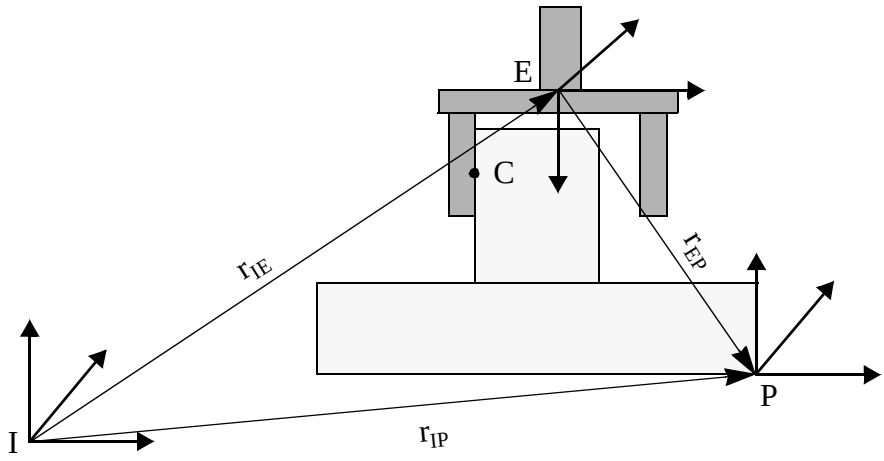

FIGURE 13. Coordinate Frames for Contacting Bodies

With these coordinate frames defined, the kinematic and dynamic variables used in the modular constrained motion algorithm will be labeled (when appropriate) with the leading superscript I, E, or P to indicate the coordinate frame of expression. One exception is the

contact force vector which has two leading superscripts. This exception is discussed in greater detail in the next section.

3.2.3 Spatial Contact Force Vectors

Throughout this work, spatial contact force vectors are used in equations for joint space dynamics, operational space dynamics, and contact modelling. Often, these vectors take different forms. It is therefore necessary to establish a consistent means for distinguishing between these alternate forms and to specify appropriate transformations for converting from one form to another. For accuracy, each form of a spatial force vector should contain the following information: (1) the direction of the force vector, (2) the coordinate frame of expression, and (3) the *torque reference point*.

The subscripts of the force vector specify its direction. For example, F_{EP} represents the spatial contact force exerted *by* body E *on* body P. Reversing the subscript order is equivalent to a sign change, as shown below:

$$^{EP}F_{EP} = -^{EP}F_{PE} \tag{17}$$

The coordinate frame of expression is given by the leading superscript and can be changed by pre-multiplying the spatial contact force by the appropriate 6×6 matrix of direction cosines, $\Re$, as shown in the following example:

$$^{EP}F_{EP} = {}^{E}\Re_{P} \cdot {}^{PP}F_{EP} \tag{18}$$

Note, however, that for the contact force vector, there is more than one leading superscript. The second leading superscript specifies the *torque reference point* (TRP). This

point, together with the contact point (e.g. point C in Figure 13), defines the moment arm through which torques are generated by the linear contact forces. These torques are included in the angular components of the spatial contact force vector. An example operation for changing the TRP is:

$$^{EE}F_{EP} = {}^{E}S_{EP}^{T} \cdot {}^{EP}F_{EP} \tag{19}$$

where the spatial rigid-body transformation matrix, ${}^{E}S_{EP}^{T}$, is the linear operator used to transform a spatial force acting at point P to a spatial force acting at point E. In general, S has the following form [9][59]:

$$^{i}S_{jk} = \begin{bmatrix} 1 & -\tilde{r}_{jk}^{i} \\ 0 & 1 \end{bmatrix} \tag{20}$$

where 1 is a 3×3 identity matrix, 0 is a 3×3 matrix of zeros, and

$$\tilde{r}_{jk}^{i} = \begin{bmatrix} 0 & -z & y \\ z & 0 & -x \\ -y & x & 0 \end{bmatrix} \tag{21}$$

is a 3×3 skew-symmetric matrix derived from the linear position vector, $^{i}r_{jk} = \begin{bmatrix} x & y & z \end{bmatrix}^{T}$.

The transformations shown in Equations 17, 18, and 19 will be used to transform the dynamic equations of the constrained motion algorithm to a standard form which is consistent with the general contact model. This standard form will be discussed in the following section and again in Chapter 4.1.

3.3 General Joint Model

When connecting two or more modules together in the new modular approach, or when making a connection between a module and the environment, it will be necessary to establish an appropriate contact model. In order to maintain as much generality as possible, a contact model is needed that can represent any body-to-body contact in which relative motion can be specified, including such examples as a revolute or prismatic joint, a multiple-degree-of-freedom joint (such as universal or spherical joints), a rigid grasp, planar sliding, a hard-point contact, or even the rolling or sliding of two surfaces. In [58], Roberson and Schwertassek develop such a contact model and refer to it as the general joint model.

The original constrained motion algorithm of Lilly [39][40], reviewed in the previous section, utilized a simplified version of the original general joint model. However, in the interest of making the MRDS algorithm as general and flexible as possible, the first extension of the algorithm will be to incorporate the original form of the Roberson and Schwertassek contact model, which allows a much wider variety of contact conditions. Details of this more general model are reviewed here for an arbitrary contact between rigid bodies E and P. This review also allows all key terms to be specified using the new notational scheme, eliminating any ambiguity in previous applications of the model [6][39][40].

3.3.1 Contact Kinematics

Initially, a set of generalized minimal velocity coordinates is specified which describes the velocity of body P relative to body E. Velocity coordinates are used (rather than position coordinates) to allow both non-holonomic and holonomic constraints [58]. These velocity

coordinates are collected in the $(6 - n_c) \times 1$ vector, $\dot{x}^u$, where $n_c \leq 6$ is the number of spatial directions in which the relative velocity is subject to constraints.

Given these coordinates, the form of the contact model vector spaces is determined. Using these vector spaces, spatial expressions for the velocity and acceleration of body P relative to body E, ${}^E\dot{x}_{EP}$ and ${}^E\ddot{x}_{EP}$, may be written, respectively, as follows [58]:

$$ {}^E\dot{x}_{EP} = \phi^u \cdot \dot{x}^u + \phi^c \cdot \dot{x}^c \tag{22} $$

$$ {}^E\ddot{x}_{EP} = \phi^u \cdot \ddot{x}^u + \phi^c \cdot \ddot{x}^c + \dot{\phi}^u \cdot \dot{x}^u + \dot{\phi}^c \cdot \dot{x}^c \tag{23} $$

The $6 \times (6 - n_c)$ matrix, ϕ^u, resolved in frame E, has full column rank, $6 - n_c$. It represents the *free modes* of the contact, and its columns make up a basis for this free vector space (also known as the *motion space* of the contact). The structure of ϕ^u is determined by the selection of the generalized minimal velocity coordinates, $\dot{x}^u$. The $6 \times n_c$ matrix, ϕ^c, is chosen so that the columns of the 6×6 matrix, $\phi = \left[\phi^u \;\; \phi^c\right]$, form a basis for all of $\mathbf{R}^6$.

The $n_c \times 1$ vector, $\dot{x}^c$, contains the coordinates describing the relative velocity in the constrained directions of the contact. There are two types of *constrained modes* which may be represented by the basis vectors in ϕ^c: *locked modes* and *kinematically excited modes* [58]. In the contact directions corresponding to a locked mode, there is no relative motion between the two contacting bodies. Therefore, all elements of $\dot{x}^c$ corresponding to locked modes are zero. For kinematically excited modes, however, the elements of $\dot{x}^c$ are known functions of time. This type of contact typically involves the prescribed motion of an actuated intermediate body (see [58] for examples).

A dual basis for $\mathbf{R}^6$, $\psi = \left[\psi^u \;\; \psi^c\right]$, is constructed from the first as follows:

$$\psi^T = \phi^{-1} \tag{24}$$

This definition leads to the following relationships:

$$\psi^T\phi = \begin{bmatrix} (\psi^u)^T \\ (\psi^c)^T \end{bmatrix} \begin{bmatrix} \phi^u & \phi^c \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \tag{25}$$

where 1 and 0 are identity and zero matrices of the appropriate dimensions, respectively. The constrained modes of the contact are represented by the $6 \times n_c$ matrix ψ^c, the columns of which form a basis for this vector space. This is referred to as the *constraint space* of the contact. In the case of orthogonal modes, the vector bases ϕ and ψ are equivalent. However, the free modes may form a nonorthogonal basis, as in the case of a planar translation combined with kinematic excitation, in which case $\phi \neq \psi$ [58]. Other contact conditions, such as a screw connection, also result in nonidentical vector bases ϕ and ψ, as will be demonstrated below in Section 3.3.3.

Variable mode vectors are also supported by this model. They are typical for higher pairs and for constraints which result from intermediate mechanisms if more than one rotational mode is free, as in the case of a universal joint. Mode vectors for all lower pairs are constant, however, which may explain their general popularity in multibody algorithms.

3.3.2 Contact Forces

The dynamic interaction between the two contacting bodies may be described by a resultant spatial force. However, as described in the previous section, this spatial contact force vector may be expressed in various forms depending on its direction, the coordinate

frame of expression, and the torque reference point (TRP). From the contact model of Roberson and Schwertassek [58], the appropriate form of the contact force vector is ${}^{EP}F_{EP}$, which denotes the force exerted by body E on body P, expressed in the coordinates of frame E, and using point P as the torque reference point. This force vector is resolved in the dual basis defined above as follows:

$$ {}^{EP}F_{EP} = \psi^u \cdot F^u + \psi^c \cdot F^c \tag{26} $$

where F^u is the $(6 - n_c) \times 1$ vector of applied or friction forces, and F^c is the $n_c \times 1$ vector of constraint forces exerted by body E on body P along the constrained directions of the contact.

The vector spaces ϕ^u and ϕ^c, as well as ψ^u and ψ^c, are used to define the type of contact being investigated. Two classes of contacts between a pair of rigid bodies were previously defined in [39]. For Class I contacts, the components of F^u are known and are, in most cases, zero. However, when the contact involves an actuated joint, the components of F^u are nonzero and must be supplied, as is true of any Forward Dynamics algorithm. Examples of Class I contacts include revolute and prismatic joints, rigid connections, hard point contacts, and soft finger contacts with no slipping.

Class II contacts occur when the components of F^u are dependent on the components of F^c and include such interactions as a sliding contact on a surface with a characteristic coefficient of friction. The algorithm presented here has been developed for both Class I and Class II contacts. However, since the calculations for the treatment of Class II contacts are considerably more involved, developments are given in this work for Class I contacts

only. The additional calculations required for Class II contacts are discussed in [39]. Similar work with Class II contacts is discussed by Chen in [14].

3.3.3 Contact Model Examples

To clarify the construction of the contact model vector spaces, we provide an example contact between bodies E and P that is equivalent to a revolute joint. For this type of contact, the generalized minimal velocity vector, $\dot{x}^u$, is given by the scalar joint velocity. Using Denavit-Hartenberg [17] parameters for establishing coordinate frames, the joint angle is denoted by θ_E and is the angular position about the $\hat{z}_E$ unit vector. Therefore, the motion space is:

$$\phi^u = \psi^u = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 & 0 \end{bmatrix}^T \tag{27}$$

while the constraint space of the joint may be written:

$$\phi^c = \psi^c = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix} \tag{28}$$

Note that in this case, ϕ^u and ψ^u are the same, as are their counterpart vector spaces, ϕ^c and ψ^c.

An example of a contact in which this is not true is a screw joint, which permits interdependent sliding and turning motion between the two bodies. The joint angle, θ_E, and the

translation along the axis, d_E, are coupled by ρ, the scalar pitch of the screw as follows:

$$d_E = \rho \cdot \theta_E \tag{29}$$

Because the two motions are coupled, there is only one degree of freedom associated with this joint. Either $\dot{\theta}_E$ or $\dot{d}_E$ may be chosen as the generalized minimal coordinate, $\dot{x}^u$. Once again, we will choose $\dot{x}^u = \dot{\theta}_E$.

The corresponding motion space, ϕ^u, and remaining basis vectors for $\mathbf{R}^6$, ϕ^c, are [58]:

$$\phi^u = \begin{bmatrix} 0 \\ 0 \\ \rho \\ 0 \\ 0 \\ 1 \end{bmatrix} \qquad \phi^c = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix} \tag{30}$$

The constraint space and remaining dual basis vectors for $\mathbf{R}^6$, ψ^c and ψ^u, respectively, are:

$$\psi^c = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & -\rho & 0 & 0 \end{bmatrix} \qquad \psi^u = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} \tag{31}$$

For additional examples of the more complex contacts, including those with powered joints, kinematic excitation, orthogonal modes, nonorthogonal modes, constant modes, and variable modes, see Roberson and Schwertassek [58] and Lilly [39]. For a less general version of this contact model used in dynamically-decoupled force/motion control, see Featherstone, et al. [21].

3.4 Summary

In this chapter, we have reviewed the original constrained motion algorithm of Lilly [39][40], as well as the fundamentals of both the joint space and operational space formalisms. A notation scheme has been put in place that will allow precise specification of all kinematic, dynamic, and contact model terms. The general joint model of Roberson and Schwertassek [58] has been presented. This model allows a wider range of contacts to be considered than the simplified form used in previous works [6][39][40].

In the next chapter, Lilly's constrained motion algorithm will be enhanced by the use of the new notation scheme and the more general contact model. This extension will produce an efficient modular algorithm for the dynamic simulation of not only one robot in constrained motion, but two robot systems cooperating in parallel or in series. Full extension to multiple contacts between multiple robots will be addressed in Chapter 5, while the extension to robots with structural flexibility will be dealt with in Chapter 7.

Chapter 4 SYSTEMS WITH A SINGLE CONTACT MODEL

This chapter presents the first extension to Lilly's original constrained motion algorithm. Our goal is to develop a standard form for introducing the general contact model from Section 3.3 into the updated constrained motion algorithm, thereby expanding the range of allowable contact conditions. All equations will be presented using the notation scheme defined in Section 3.2. The result will be an accurate and useful presentation of the constrained motion algorithm which will be efficient and effective across a broad range of robot applications. As such, this chapter serves as the first step in achieving a constrained motion algorithm capable of simulating the dynamics of the far more sophisticated, time-varying, multi-robot systems which appear today.

This first extension will provide the ability to simulate the dynamics of systems which contain a single external contact such as those shown in Figure 14. Note that these systems can contain series connections of independent robots, parallel connections, or even hybrid cases like the bracing example. Extension to systems with multiple contacts between multiple robots will be dealt with in Chapter 5.

Throughout this work, we will denote the two interacting bodies associated with a contact model as body E and body P. Although this algorithm is applicable to general (i.e. non-robot) structures, we assume throughout that body E belongs to a robot (also referred to as Structure E). Body P, on the other hand, may belong to one of two possible structures: 1) an object or device external to the robot (referred to as Structure P), or 2) one of the robot's own member bodies. These two contact configurations are referred to as Type A and Type B, respectively. Figures 14a through 14d show systems with a Type A contact, while

Figure 14e gives an example of a Type B contact.

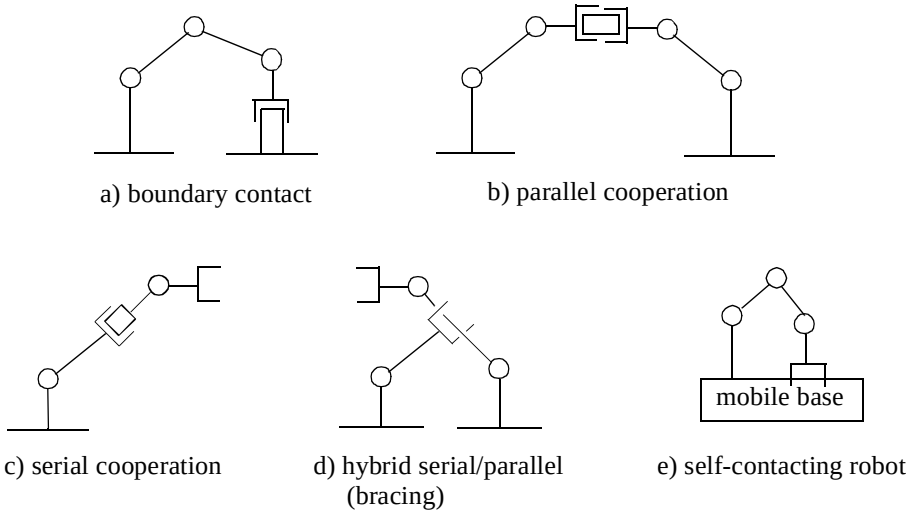

FIGURE 14. Applicable Single Contact Systems

The enhanced constrained motion algorithm is composed of four basic steps, which may be described as follows:

Step 1: Contact Model Specification

Step 2: Dynamics of the Individual System Structures

Step 3: Dynamics of the Combined System and Solution of Contact Forces

Step 4: Joint Accelerations and Integration to the Next State

In the sections that follow, the equations used by the enhanced algorithm will be derived for both Type A and Type B contact configurations. At the same time, computational steps used to implement the algorithm will be outlined in Tables 1 through 6.

4.1 Incorporation of the General Contact Model

The enhanced algorithm begins with the incorporation of the contact model discussed in Section 3.3. In the original algorithm, presented in [39] and [40] and applied in [6], the simplified contact model was introduced through Equations 12 and 13. These equations, however, apply only to a restricted class of contacts when compared with the more general contact model defined by Equations 23 and 26.

Specifically, in Equations 12 and 13, it was assumed that the two dual bases, $\left[\phi^u \ \phi^c\right]$ and $\left[\psi^u \ \psi^c\right]$, are equivalent (orthogonal modes). By replacing Equation 13 with Equation 26, we allow contacts in which the dual vector spaces are not equivalent, such as the screw connection example of Section 3.3.3. Furthermore, it was also assumed that the contact vector spaces, ϕ^u and ϕ^c, do not vary with time. Using Equation 23 in place of Equation 12, we remove this restriction. Contacts with variable modes include the universal joint and the rolling or sliding of two nonplanar bodies [58].

In addition to being less restrictive, Equations 23 and 26 are also more explicit than Equations 12 and 13. For instance, in Equation 13, it is unclear which form of the contact force vector, F, should be represented using the contact model vector spaces. A similar argument can be made about the ambiguity of $\ddot{x}$ in Equation 12. This brings an important issue to light.

The given contact modelling technique may be used to represent pairs of force and acceleration vectors other than our choice of ${}^{EP}F_{EP}$ and ${}^{E}\ddot{x}_{EP}$ (for example ${}^{PE}F_{PE}$ and ${}^{P}\ddot{x}_{PE}$). At issue is the selection of a *consistent pair*. An inconsistent choice (e.g. ${}^{EP}F_{EP}$ and ${}^{P}\ddot{x}_{PE}$) will lead to errors in the overall formulation, such as a violation of the dual base relation-

ship given by Equations 25.

With these issues in mind, and to allow for the most comprehensive range of contacts possible, we choose Equations 23 and 26 as the consistent pair of contact model expressions to replace Equations 12 and 13 in the constrained motion algorithm. With this "standard form" chosen, Table 1 lists components of the contact model which the user must provide, as well as a few notes on specification procedures. Note that the labels E and P are omitted, as their interpretation is now always implied by our standard pair: ${}^{E}\ddot{x}_{EP}$ and ${}^{EP}F_{EP}$. This initialization process represents Step 1 in the enhanced constrained motion algorithm.

TABLE 1. Step 1: Contact Model Specification

Model Components	Notes
ϕ^u, ϕ^c, ψ^u, ψ^c	See Section 3.3.1 and [58] for specification methods
$\dot{\phi}^u$, $\dot{\phi}^c$	Zero for constant modes (note: $\dot{\phi}^c$ only needed if $\dot{x}^c \neq 0$)
$\dot{x}^u$	Supply initial values (note: only needed if $\dot{\phi}^u \neq 0$) (see Chapter 6)
$\dot{x}^c$, $\ddot{x}^c$	Zero for locked modes; known function of time for kinematically excited modes (note: $\dot{x}^c$ only needed if $\dot{\phi}^c \neq 0$)
F^u	Class I contacts: zero or known function of time; Class II contacts: see [39]

4.2 Dynamics of the Individual System Structures

With the standard form of the contact model defined, we now turn our attention to the dynamics of the individual structures present in the system of interest. We begin with a Type A contact.

4.2.1 Type A Contact: Contact with an External Body or Boundary

In this configuration, Structure E (the robot) and Structure P are two separate entities, each having its own separate dynamic equations. The structures are joined only by the single contact between them. We begin with the joint space and operational space dynamic equations of the robot (Structure E) using the notation scheme of Section 3.2. The new explicit form of Equation 1 may be written as follows[14]:

$$\tau = H \cdot \ddot{q} + C + G + {}^{I}J_{IE}^{T} \cdot {}^{IE}F_{EP} \tag{32}$$

where, aside from ${}^{I}J_{IE}$, all terms were defined and discussed in Chapter 3. The matrix ${}^{I}J_{IE}$ is the Jacobian matrix which relates the robot joint velocities to the spatial velocity of the end effector (frame E) as follows:

$${}^{I}\dot{x}_{IE} = {}^{I}J_{IE} \cdot \dot{q} \tag{33}$$

As in Equation 2, the joint space dynamic equation is solved for the generalized acceleration coordinates:

$$\ddot{q} = H^{-1} \cdot (\tau - C - G) - (H^{-1} \cdot {}^{I}J_{IE}^{T}) \cdot {}^{IE}F_{EP} \tag{34}$$

where H is positive definite and, therefore, invertible for real systems [39]. This expression can be simplified as follows:

$$\ddot{q} = \ddot{q}^{open} - {}^{I}\Psi_{IE} \cdot {}^{IE}F_{EP} \tag{35}$$

---

14. Computational efficiency may be gained by representing the contact force and Jacobian matrix in frame E. The inertial frame is used to simplify the derivation of equations.

The $N \times 6$ matrix, ${}^{I}\Psi_{IE}$, is an example of the *cross space inertia matrix* which maps forces in operational space to accelerations in joint space. It is a function only of the generalized coordinates, q, and is defined as follows:

$$ {}^{I}\Psi_{IE} = H^{-1} \cdot {}^{I}J_{IE}^{T} \tag{36} $$

To produce the operational space dynamic equations, a kinematic expression for the acceleration of the contact point is needed. This can be produced by differentiating Equation 33, the result of which is:

$$ {}^{I}\ddot{x}_{IE} = {}^{I}J_{IE} \cdot \ddot{q} + {}^{I}\dot{J}_{IE} \cdot \dot{q} \tag{37} $$

Introducing the joint space dynamics of Equation 34 into Equation 37, the operational space dynamic equation of the robot is:

$$ {}^{I}\ddot{x}_{IE} = {}^{I}J_{IE} \cdot H^{-1} \cdot (\tau - C - G) + {}^{I}\dot{J}_{IE} \cdot \dot{q} - ({}^{I}J_{IE} \cdot H^{-1} \cdot {}^{I}J_{IE}^{T}) \cdot {}^{IE}F_{EP} \tag{38} $$

This expression can be grouped and rewritten to match the form of Equation 11, as follows:

$$ {}^{I}\ddot{x}_{IE} = {}^{I}\ddot{x}_{IE}^{open} - {}^{I}\Lambda_{EE}^{-1} \cdot {}^{IE}F_{EP} \tag{39} $$

The 6×6 contact force coefficient matrix, ${}^{I}\Lambda_{EE}^{-1}$, is the *inverse operational space inertia matrix* of the robot [30][39] and is also a function only of the generalized coordinates, q. It is defined for all N as follows:

$$ {}^{I}\Lambda_{EE}^{-1} = {}^{I}J_{IE} \cdot H^{-1} \cdot {}^{I}J_{IE}^{T} \tag{40} $$

This matrix is positive semi-definite. Furthermore, ${}^{I}\Lambda_{EE}^{-1}$ is invertible, only when the Jaco-

bian matrix, ${}^{I}J_{IE}$, is of full rank (non-singular).

For Type A contacts, Structure P may be any external object or device, such as a payload, a fixed boundary, or even a second robot. Its dynamic equations are developed in exactly the same fashion as those outlined above for the robot in Equations 32 through 40. The key resultant equations needed in the constrained motion algorithm (verified simply by switching labels E and P in Equations 35 and 39 above) are:

$$\ddot{q} = \ddot{q}^{open} - {}^{I}\Psi_{IP} \cdot {}^{IP}F_{PE} \tag{41}$$

$${}^{I}\ddot{x}_{IP} = {}^{I}\ddot{x}_{IP}^{open} - {}^{I}\Lambda_{PP}^{-1} \cdot {}^{IP}F_{PE} \tag{42}$$

A special condition worth noting occurs when body P is an inertially fixed boundary. Under this condition, the equations above become trivial, since $\ddot{q} = \ddot{q}^{open} = 0$ and ${}^{I}\ddot{x}_{IP} = {}^{I}\ddot{x}_{IP}^{open} = 0$. A similar but less restrictive condition is when Structure P is considered to be too massive for its motion to be altered by the contact[15]. Such a condition might arise if the robot were to make contact with some powerful piece of reciprocating machinery. In this event, the matrices ${}^{I}\Psi_{IP}$ and ${}^{I}\Lambda_{PP}^{-1}$ are assumed to be negligible, and the motion of Structure P can be computed using simple open chain methods [8][68][70] with the equations: $\ddot{q} = \ddot{q}^{open}$ and ${}^{I}\ddot{x}_{IP} = {}^{I}\ddot{x}_{IP}^{open}$.

Returning again to the general case, we note that a number of terms may be computed in parallel for the two structures of this system, which leads to increased overall efficiency of the algorithm. Calculation of $\ddot{q}^{open}$ and $\ddot{x}^{open}$ may be completed simultaneously for both structures using any suitable open chain dynamic simulation algorithm, such as those found

15. when compared with the maximum possible magnitude of the contact force generated by the robot.

in [8], [68], or [70]. The appropriate Ψ and Λ^{-1} matrices may be calculated using Equations 36 and 40, respectively, or the more efficient O(N) methods discussed in [39]. Table 2 summarizes these terms for easy reference. These computations, whether accomplished in parallel or serially, represent Step 2 of the enhanced constrained motion algorithm for a Type A contact configuration and demonstrate the first evidence of the modularity of the simulation algorithm.

As noted in Table 2, the position, orientation, and spatial velocity of each operational point (E and P) are also computed in this step. These terms are needed in Step 3 of the algorithm, where the dynamics of the combined system are developed. First, however, we will derive Step 2 for Type B contacts.

TABLE 2. Step 2: Dynamics of the Individual System Structures (Type A Contact)

Terms	Structure	Notes
$\ddot{q}^{open}$ and ${}^{I}\ddot{x}_{IE}^{open}$	E	Suitable open chain algorithm (e.g. [8][68][70])
${}^{I}\Psi_{IE}$ and ${}^{I}\Lambda_{EE}^{-1}$	E	Equations 36 and 40, or see [39]
${}^{I}r_{IE}$, ${}^{I}R_{E}$, and ${}^{I}\dot{x}_{IE}$	E	Suitable Forward Kinematics algorithm
$\ddot{q}^{open}$ and ${}^{I}\ddot{x}_{IP}^{open}$	P	Suitable open chain algorithm (e.g. [8][68][70])
${}^{I}\Psi_{IP}$ and ${}^{I}\Lambda_{PP}^{-1}$	P	Equations 36 and 40 (for Structure P), or see [39]
${}^{I}r_{IP}$, ${}^{I}R_{P}$, and ${}^{I}\dot{x}_{IP}$	P	Suitable Forward Kinematics algorithm

Note: each structure's terms can be computed in parallel for increased efficiency

4.2.2 Type B Contact: Contact with a Member Body of the Robot

The second configuration type occurs when the robot makes contact with itself (or if the robot contains a closed loop in its structural design). Such an example was discussed in [6],

where a dynamic simulation was developed for operation of the Shuttle Remote Manipulator System (SRMS) contacting objects in the shuttle cargo bay. For this and all configurations of Type B, the joint space dynamic equation given by Equation 32 must be modified to account for the presence of two external contact forces: one exerted by body E and another exerted by body P. This modified form can be written as follows:

$$\tau = H \cdot \ddot{q} + C + G + {}^{I}J_{IP}^{T} \cdot {}^{IP}F_{PE} + {}^{I}J_{IE}^{T} \cdot {}^{IE}F_{EP} \tag{43}$$

The two force terms, however, can be combined using the following force transformations from Section 3.2.3:

$${}^{IE}F_{EP} = {}^{I}\Re_{E} \cdot {}^{E}S_{EP}^{T} \cdot {}^{EP}F_{EP} \tag{44}$$

$${}^{IP}F_{PE} = -{}^{I}\Re_{E} \cdot {}^{EP}F_{EP} \tag{45}$$

With the force terms combined, the generalized joint acceleration equation is:

$$\ddot{q} = \ddot{q}^{open} - {}^{E}\Psi_{EP} \cdot {}^{EP}F_{EP} \tag{46}$$

where

$${}^{E}\Psi_{EP} = H^{-1} \cdot {}^{E}J_{EP}^{T} \tag{47}$$

and

$${}^{E}J_{EP} = {}^{E}\Re_{I} \cdot ({}^{I}J_{IP} - {}^{I}S_{EP} \cdot {}^{I}J_{IE}) \tag{48}$$

The *relative* Jacobian matrix, ${}^{E}J_{EP}$, describes a relationship between the robot joint rates and a *relative* spatial velocity (P relative to E, in this case), as opposed to an *absolute* spatial

velocity defined by a *traditional* Jacobian matrix. This relationship is given by the following equation:

$$ {}^{E}\dot{x}_{EP} = {}^{E}J_{EP} \cdot \dot{q} \tag{49} $$

Finally, the operational space dynamic equations corresponding to Equations 39 and 42 are derived by the same method to obtain:

$$ {}^{I}\ddot{x}_{IE} = {}^{I}\ddot{x}_{IE}^{open} - ({}^{I}J_{IE} \cdot H^{-1} \cdot {}^{E}J_{EP}^{T}) \cdot {}^{EP}F_{EP} \tag{50} $$

$$ {}^{I}\ddot{x}_{IP} = {}^{I}\ddot{x}_{IP}^{open} - ({}^{I}J_{IP} \cdot H^{-1} \cdot {}^{E}J_{EP}^{T}) \cdot {}^{EP}F_{EP} \tag{51} $$

Table 3 summarizes the terms which must be calculated for a Type B contact configuration in Step 2 of the enhanced constrained motion algorithm.

TABLE 3. Step 2: Dynamics of the Individual System Structures (Type B Contact)

Terms	Notes
$\ddot{q}^{open}$, ${}^{I}\ddot{x}_{IE}^{open}$, ${}^{I}\ddot{x}_{IP}^{open}$	Suitable open chain algorithm (e.g. [8][68][70])
${}^{E}\Psi_{EP}$ and ${}^{E}L_{EP}^{-1}$	Equations 47 and 61

Note: the term ${}^{E}L_{EP}^{-1}$ is derived in the next section but is computed in this step for Type B

4.3 Dynamics of the Combined System

In the original constrained motion algorithm [39][40], the contact force vector is determined by combining the contact model with the operational space dynamic equation of the robot end effector (body E). As we apply a more general form of the contact model, however, this form of the operational space dynamic equation is insufficient. Instead, we need

an operational space dynamic equation that involves both contacting bodies (E and P).

In Section 4.2, operational space dynamic equations were developed for bodies E and P in their isolated states (Equations 39 and 42, respectively, for Type A and Equations 50 and 51, respectively, for Type B). Each of these equations was derived by inserting the joint space dynamics of the structure into a kinematic expression for the *absolute* acceleration of the contact body. In order to produce the operational space dynamic equation for the combined system, a kinematic expression for the *relative* acceleration is needed.

To derive this acceleration, we begin with the relationships describing the absolute and relative linear positions and angular velocities of the contacting bodies. Respectively, these relationships are as follows:

$$ {}^{I}r_{IP} = {}^{I}r_{IE} + {}^{I}R_{E} \cdot {}^{E}r_{EP} \tag{52} $$

$$ {}^{I}\omega_{IP} = {}^{I}\omega_{IE} + {}^{I}R_{E} \cdot {}^{E}\omega_{EP} \tag{53} $$

where, in general terms, ${}^{i}r_{jk}$ and ${}^{i}\omega_{jk}$ are the linear position and angular velocity of frame k relative to frame j, expressed in frame i, respectively, and R is the appropriate 3×3 direction cosine matrix. These equations are differentiated to produce ${}^{I}\dot{r}_{IP}$, ${}^{I}\ddot{r}_{IP}$, and ${}^{I}\dot{\omega}_{IP}$. The resulting equations can then be cast in spatial notation to give the spatial velocities and accelerations of the contacting bodies as follows:

$$ {}^{I}\dot{x}_{IP} = {}^{I}S_{EP} \cdot {}^{I}\dot{x}_{IE} + {}^{I}\Re_{E} \cdot {}^{E}\dot{x}_{EP} \tag{54} $$

$$ {}^{I}\ddot{x}_{IP} = {}^{I}S_{EP} \cdot {}^{I}\ddot{x}_{IE} + {}^{I}\zeta_{EEP} + {}^{I}\Re_{E} \cdot {}^{E}\ddot{x}_{EP} \tag{55} $$

where

$$ {}^{I}\zeta_{EEP} = \begin{bmatrix} {}^{I}\omega_{IE} \times [{}^{I}\omega_{IE} \times {}^{I}R_{E} \cdot {}^{E}r_{EP}] + 2 \cdot {}^{I}\omega_{IE} \times {}^{I}R_{E} \cdot {}^{E}\dot{r}_{EP} \\ {}^{I}\omega_{IE} \times {}^{I}R_{E} \cdot {}^{E}\omega_{EP} \end{bmatrix} \tag{56} $$

As before, $\Re$ is a spatial direction cosine matrix, and S is a spatial rigid-body transformation. The terms, ${}^{E}r_{EP}$, ${}^{E}\dot{r}_{EP}$, and ${}^{E}\omega_{EP}$ in ${}^{I}\zeta_{EEP}$ are found from Equations 52 and 54. This process is discussed in Chapter 6, where several options for serial connections are outlined.

In the final step of this kinematic derivation, Equation 55 is rearranged to obtain ${}^{E}\ddot{x}_{EP}$ as follows:

$$ {}^{E}\ddot{x}_{EP} = {}^{E}\Re_{I} \cdot ({}^{I}\ddot{x}_{IP} - {}^{I}S_{EP} \cdot {}^{I}\ddot{x}_{IE} - {}^{I}\zeta_{EEP}) \tag{57} $$

It is this kinematic expression for the relative acceleration of the contacting bodies that can be converted into an operational space dynamic equation for the combined system. This is accomplished by replacing the terms ${}^{I}\ddot{x}_{IE}$ and ${}^{I}\ddot{x}_{IP}$ in Equation 57 with the operational space dynamic equations for each isolated structure developed in Section 4.2. For Type A, these terms are replaced by Equations 39 and 42, together with the force transformations in Equations 44 and 45. For Type B, ${}^{I}\ddot{x}_{IE}$ and ${}^{I}\ddot{x}_{IP}$ are replaced with Equations 50 and 51. In both cases, this substitution results in the following common form:

$$ {}^{E}\ddot{x}_{EP} = {}^{E}\ddot{x}_{EP}^{open} + {}^{E}L_{EP}^{-1} \cdot {}^{EP}F_{EP} \tag{58} $$

where ${}^{E}\ddot{x}_{EP}^{open}$ for both Type A and Type B is given by:

$$ {}^{E}\ddot{x}_{EP}^{open} = {}^{E}\Re_{I} \cdot ({}^{I}\ddot{x}_{IP}^{open} - {}^{I}S_{EP} \cdot {}^{I}\ddot{x}_{IE}^{open} - {}^{I}\zeta_{EEP}) \tag{59} $$

The expression for ${}^{E}L_{EP}^{-1}$, however, is dependent on configuration type. This matrix is

effectively the inverse operational space inertia matrix for the combined system. For a Type A contact configuration, it is defined as follows[16]:

$$ {}^{E}L_{EP}^{-1} = {}^{E}\Re_{I} \cdot ({}^{I}\Lambda_{PP}^{-1} + {}^{I}S_{EP} \cdot {}^{I}\Lambda_{EE}^{-1} \cdot {}^{I}S_{EP}^{T}) \cdot {}^{I}\Re_{E} \tag{60} $$

For a Type B contact configuration, after appropriate factoring of the contact force term, ${}^{E}L_{EP}^{-1}$ is given by:

$$ {}^{E}L_{EP}^{-1} = {}^{E}J_{EP} \cdot H^{-1} \cdot {}^{E}J_{EP}^{T} \tag{61} $$

which has the same structural form as any inverse operational space inertia matrix, but uses the relative Jacobian in place of the traditional Jacobian. For Type B, this term would, in fact, be calculated at the modular level in Step 2, as noted below Table 3.

Table 4 summarizes the terms which must be computed to complete the dynamics of the combined system. These computations represent the first part of Step 3 of the enhanced constrained motion algorithm.

TABLE 4. Step 3a: Dynamics of the Combined System

Terms	Notes
${}^{I}\zeta_{EEP}$ and ${}^{I}S_{EP}$	Equation 56 and Equation 20
${}^{E}\ddot{x}_{EP}^{open}$	Equation 59
${}^{E}L_{EP}^{-1}$	Equation 60 for Type A, Equation 61 for Type B (see Table 3)

Equation 58 is the operational space dynamic equation for the combined system. Com-

16. Set ${}^{I}\Lambda_{PP}^{-1}$ to zero for the special examples of Type A contact with a fixed boundary or comparatively massive structure.

paring Equations 39 and 58, we see that these two expressions are very similar. Their functions, however, are quite different. In Equation 39, the inverse operational space inertia matrix, ${}^{I}\Lambda_{EE}^{-1}$, is used to relate the spatial contact force to the *absolute* spatial acceleration of body E. Only the dynamics of the robot (Structure E) are involved in this term. Alternatively, in Equation 58, the matrix, ${}^{E}L_{EP}^{-1}$, includes the dynamics of the structures on both sides of the contact (for Type A) and relates the spatial contact force to the *relative* spatial acceleration of bodies E and P. With Equation 58 describing the dynamics of the combined system, we are now ready to introduce the new, standard form of the contact model and complete the constrained motion algorithm.

4.4 Calculation of the Contact Forces and Joint Accelerations

Having produced the operational space dynamic equation for the combined system for both Type A and Type B contacts, we can now apply the same solution procedure used in Section 3.1 for Equations 11, 12, and 13 to the enhanced expressions given by Equations 58, 23, and 26, respectively. We begin by replacing the terms ${}^{E}\ddot{x}_{EP}$ and ${}^{EP}F_{EP}$ in Equation 58 by their contact model representations in Equations 23 and 26. The result is:

$$\phi^{u} \cdot \ddot{x}^{u} + \phi^{c} \cdot \ddot{x}^{c} + \dot{\phi}^{u} \cdot \dot{x}^{u} + \dot{\phi}^{c} \cdot \dot{x}^{c} = {}^{E}\ddot{x}_{EP}^{open} + {}^{E}L_{EP}^{-1} \cdot (\psi^{u} \cdot F^{u} + \psi^{c} \cdot F^{c}) \qquad (62)$$

From the properties of the dual bases given in Equation 25, we find that $(\psi^{c})^{T} \cdot \phi^{u} = 0$ and $(\psi^{c})^{T} \cdot \phi^{c} = 1$. Using these properties, the unknown acceleration components in the unconstrained directions of the contact, $\ddot{x}^{u}$, can be eliminated by pre-multiplying Equation 62 by $(\psi^{c})^{T}$. This leaves F^{c}, the components of the contact force vector in the constrained directions, as the only remaining unknown term. The result is a linear system

of equations which can be assembled in the following form:

$$B^c \cdot F^c = y^c \tag{63}$$

where the $n_c \times n_c$ matrix, B^c, and the $n_c \times 1$ vector, y^c, are defined as follows:

$$B^c = (\psi^c)^T \cdot {}^E L_{EP}^{-1} \cdot \psi^c \tag{64}$$

$$y^c = \ddot{x}^c + (\psi^c)^T \cdot (\dot{\phi}^u \cdot \dot{x}^u + \dot{\phi}^c \cdot \dot{x}^c - {}^E\dot{x}_{EP}^{open} - {}^E L_{EP}^{-1} \cdot \psi^u \cdot F^u) \tag{65}$$

This system of equations is solved for F^c using any appropriate linear system solver. With the constrained contact force components known, the complete force vector, F, is assembled using Equation 26, and the solution for the joint space coordinate accelerations, $\ddot{q}$, is found using Equations 35, 41, and 46, as appropriate (see Table 6). Numerical integration of $\ddot{q}$ gives the next state joint positions and velocities, and the dynamic simulation proceeds to the next time step. Tables 5 and 6 summarize the computations for Steps 3b and 4, which complete the enhanced constrained motion algorithm.

Several terms used in the calculation of B^c and y^c require further comment. The variables ${}^E r_{EP}$, ${}^E\dot{r}_{EP}$, and ${}^E\omega_{EP}$ are obtained from appropriate kinematic relationships between operational points E and P. For serial connections, it is sometimes possible to avoid the occurrence of constraint violations through the proper calculation of these terms. The generalized velocity variables of the contact model, $\dot{x}^u$, are generally found from the kinematic constraint given by Equation 22. Further details regarding these terms may be found in Chapter 6. Other issues regarding singularities and constraint violation suppression are also addressed in Chapter 6.

TABLE 5. Step 3b: Solution of Contact Forces

Terms	Notes
B^c, y^c	Equations 64 and 65, respectively
F^c	Equation 63 using suitable linear system solver
${}^{EP}F_{EP}$	Equation 26
${}^{IE}F_{EP}$, ${}^{IP}F_{PE}$	Equations 44 and 45 (Type A only)

TABLE 6. Step 4: Joint Accelerations and Integration to the Next State

Terms	Structure	Notes
$\ddot{q}$	E (robot)	Equation 35 (Type A) Equation 46 (Type B)
	P (Type A only)	Equation 41
q, $\dot{q}$	E and P	Integration to next state

Note: These computations can be performed in parallel for each structure

4.5 Computational Complexity

The computational complexity of Lilly's original constrained motion algorithm is O(N), as described in [39]. The algorithm developed here is also of O(N). This is clearly the case for systems composed only of Type A contacts, since the O(N) methods of [39] can be used to calculate all forms of the cross space and inverse operational space inertia matrices (${}^{I}\Psi_{IE}$, ${}^{I}\Lambda_{EE}^{-1}$, ${}^{I}\Psi_{IP}$, and ${}^{I}\Lambda_{PP}^{-1}$, as appropriate).

Type B contacts lead to terms such as ${}^{E}\Psi_{EP} = H^{-1} \cdot {}^{E}J_{EP}^{T}$ and ${}^{E}L_{EP}^{-1} = {}^{E}J_{EP} \cdot H^{-1} \cdot {}^{E}J_{EP}^{T}$, which, if computed by these direct multiplication formulas, would result in $O(N^3)$ computational complexity (due to the explicit use of H^{-1}). However, by inserting the expression

for the relative Jacobian matrix given by Equation 48, and recalling the definitions of ${}^{I}\Psi_{IE}$, ${}^{I}\Lambda_{EE}^{-1}$, ${}^{I}\Psi_{IP}$, and ${}^{I}\Lambda_{PP}^{-1}$, we can arrive at the following O(N) expressions:

$$ {}^{E}\Psi_{EP} = ({}^{I}\Psi_{IP} - {}^{I}\Psi_{IE} \cdot {}^{I}S_{EP}^{T}) \cdot {}^{I}\Re_{E} \tag{66} $$

and

$$ {}^{E}L_{EP}^{-1} = {}^{E}\Re_{I}\left[{}^{I}\Lambda_{PP}^{-1} - {}^{I}\Lambda_{PE}^{-1} \cdot {}^{I}S_{EP}^{T} - {}^{I}S_{EP} \cdot \Lambda_{EP}^{-1} + {}^{I}S_{EP} \cdot \Lambda_{EE}^{-1} \cdot {}^{I}S_{EP}^{T}\right]{}^{I}\Re_{E} \tag{67} $$

With these expressions, the entire algorithm developed in this chapter can be performed with O(N) computational complexity. However, Equations 66 and 67 may be less efficient than their direct-multiplication counterparts when N is sufficiently small[17].

In light of these observations, if the direct multiplication methods are going to be used, it is beneficial to choose an open chain algorithm that will produce not only $\ddot{q}^{open}$, ${}^{I}\ddot{x}_{IE}^{open}$, and ${}^{I}\ddot{x}_{IP}^{open}$, but also H^{-1} and the Jacobian matrices, ${}^{I}J_{IE}$ and ${}^{I}J_{IP}$. Although no single open chain algorithm is known to supply all of these terms explicitly, the efficient methods of Wang and Ravani [70] can easily be extended for this purpose. Existing O(N) methods for calculating the Jacobian may also be found in [11] and [54].

4.6 Summary

In this chapter, the constrained motion algorithm of [39][40] has been enhanced to allow a more comprehensive range of contact conditions. Contact vector spaces may now be time-varying (e.g. a universal joint) and/or non-orthogonal (e.g. the screw connection).

17. The exact value of N would depend on the methods chosen to compute J, H, H^{-1}, $\ddot{q}^{open}$, etc.

Correct incorporation of the contact model has been established by choosing standard forms of the contact force and relative acceleration vectors, removing ambiguities found in the original version of the algorithm, while preserving its inherent efficiency. In addition, the algorithm has now been expanded to the treatment of two robots cooperating in series.

Additional work in expanding this algorithm from its treatment of one or two robots composed of rigid bodies to the treatment of multiple robots with flexible members is the subject of Chapters 5 and 7.

Chapter 5 THE MRDS FOR MULTIPLE STRUCTURES AND MULTIPLE CONTACTS

Complex systems such as the hazardous waste remediation system shown in Figure 5 or the robotic space station construction system described in [45] are complicated by many factors. The primary complexity is simply the number and intricacy of the structures involved. Additional considerations are the environment of the operation and the tendency of such systems to include multiple and varied interactions among the system members.

In this chapter, we will extend the constrained motion algorithm from Chapter 4 to systems that contain numerous structures subject to multiple concurrent contacts. Our goal is to devise an algorithm that can be applied to the most varied types of mechanical systems possible. We wish to include the widest possible set of topologies, including series connections, parallel connections, and mixed systems of series and parallel interactions, like those presented in Chapter 1. As in Chapter 4, we also wish to include the widest range of contact conditions possible, including time-varying and non-holonomic constraints.

The Modular Robot Dynamic Simulation (MRDS) algorithm developed in this chapter comes from the continued extension of the constrained motion algorithm of Lilly [39][40]. It contains the same general sequence of steps as given by the first extension in Chapter 4. In this case, however, applicable systems can contain more than just two robots and more than just a single external contact. In addition, the major steps of the algorithm are split into two distinct levels: a modular level and a global level. This structure is shown in Figure 15, together with the algorithm's four steps and the general flow of information between levels.

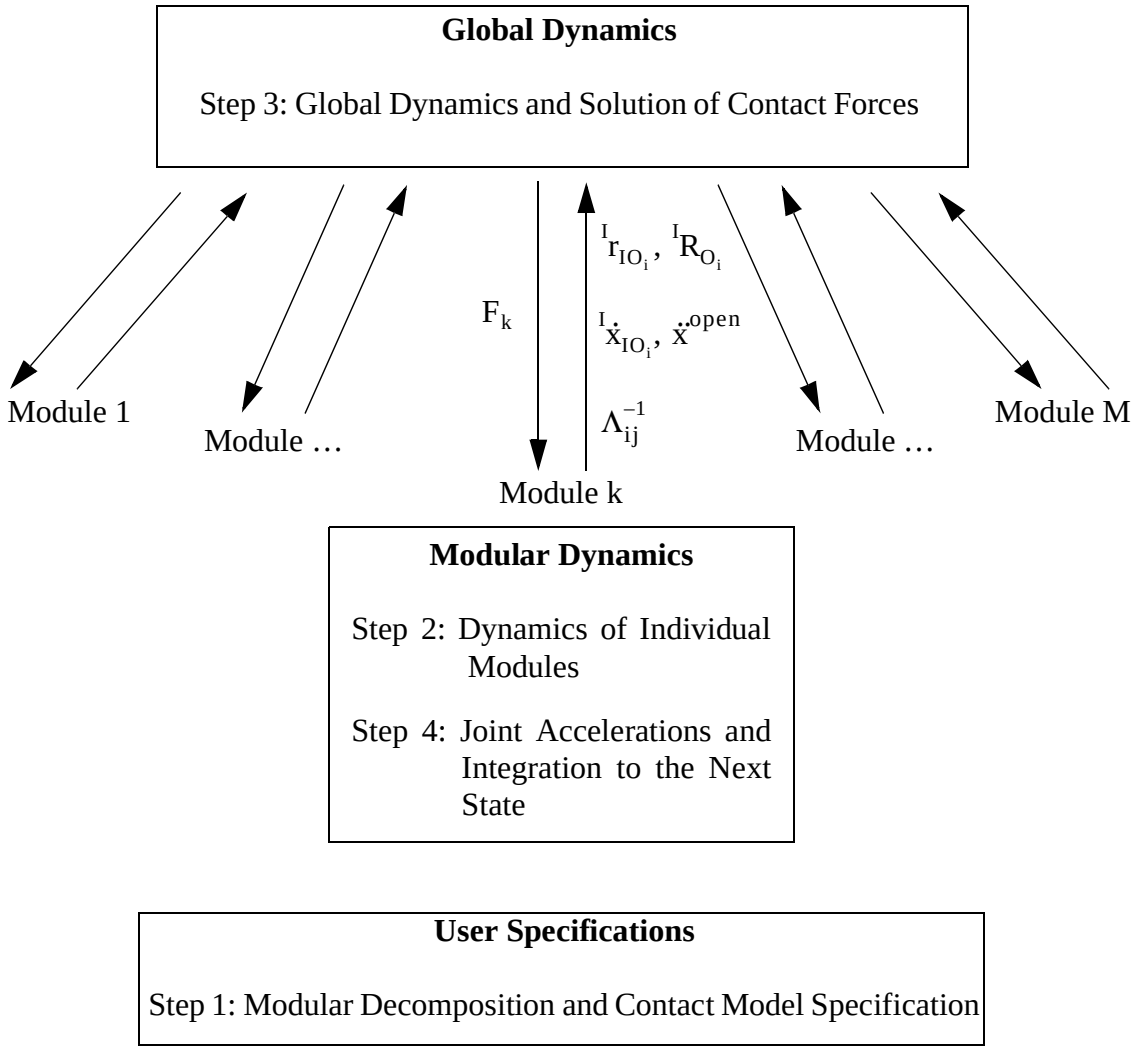

FIGURE 15. Detailed Structure of the MRDS Algorithm

At the modular level, the dynamics of each module are computed as if that particular structure were isolated and free of all external contacts. This allows the modular calculations to be performed on parallel processors, resulting in substantial increases in computational efficiency and real-time capability.

At the global level of the simulation, calculations from each module of the modular level are combined to form a linear system of equations in the unknown components of each

contact force vector. It will be shown that the appropriate combination of the modular computations is based solely on the topology of the system (i.e. the connectivity between modules).

Contacts are once again represented using the modelling technique of Roberson and Schwertassek [58], which describes the relative motion and dynamic interaction of the contacting bodies. The contact models allow the projection of the operational space dynamic equations into the constraint space, thereby eliminating the unknown accelerations in the unconstrained directions. This leaves the components of the contact forces in the constrained directions as the only unknown terms. They can be found using a suitable linear system solver.

In the sections that follow, the equations used in this new algorithm are developed based on the work from Chapter 4 and will again allow for both Type A and Type B contacts. The modular nature of the algorithm will be demonstrated, and the computational considerations for applying the algorithm will also be discussed. The result will be a constrained motion algorithm capable of simulating the dynamics of the far more sophisticated, time-varying systems being developed today for use in such complex applications as hazardous waste remediation and space-based assembly.

Throughout the derivation of the equations, the example system shown below in Figure 16 will be used to demonstrate the modularity of the algorithm. This example will also help to clarify the methods used to identify the connectivity relationships between the modules and to show which terms must be calculated as a result of the connectivity.

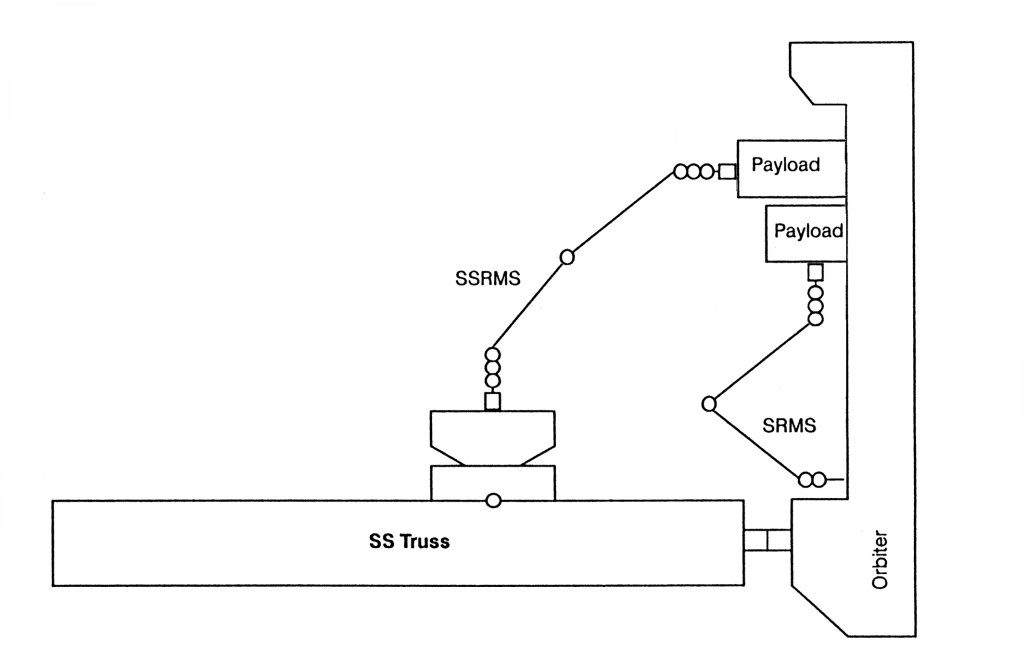

FIGURE 16. Example of a Complex Multi-Robot System [45]

The system in Figure 16 will be represented using three modules and four contacts. One module will represent the combination of the space station (SS) and the SSRMS. Another will represent a payload being transferred from the space shuttle to the space station. The final module will represent the combination of the space shuttle (orbiter), the SRMS, and the payload it is pushing along the shuttle's cargo bay. In actual implementation of the algorithm, this last module would perhaps be better represented by three modules. However, a single module will be used in order to demonstrate the handling of a Type B contact in the MRDS algorithm. The combination of these three structures into one dynamic module is consistent with the dynamic modelling and simulation coding developed in [6].

To begin the derivation of the algorithm, we will need to increase the complexity of the notation scheme described in Chapter 3 so that multiple contacts and multiple modules can

be properly distinguished. In addition, new forms of the operational space inertia matrix that arise in multi-contact operational space dynamic equations will be presented and discussed.

5.1 Modular Representation, Numbering, and Notation

Extending the constrained motion algorithm of Chapter 4 to the treatment of complex systems composed of multiple structures and multiple interactions makes necessary the introduction of additional notation. Also required is a numbering scheme that will allow us to keep track of all objects and all contact points in the system. These issues will be addressed throughout this section.

When applying this algorithm to a complex system, it is first necessary to create a modular representation of the system. An example of this is given in Figures 16 and 17. Using this method, all suitable structures (e.g. robots) are represented by a square block. Each block is considered a *module*. The connection between a module and itself, another module, or an environmental boundary is represented using a circle and indicates the use of a contact model in the simulation.

To keep track of all of these components, contact models in the system (each circle) are numbered from 1 to C, and the index "i" (or "j", when necessary) is used to refer to the "i^{th}" (or "j^{th}") contact. Similarly, all modules (each square) are numbered from 1 to M, and the index "k" is used to denote the "k^{th}" module of the system. Table 7 summarizes these conventions.

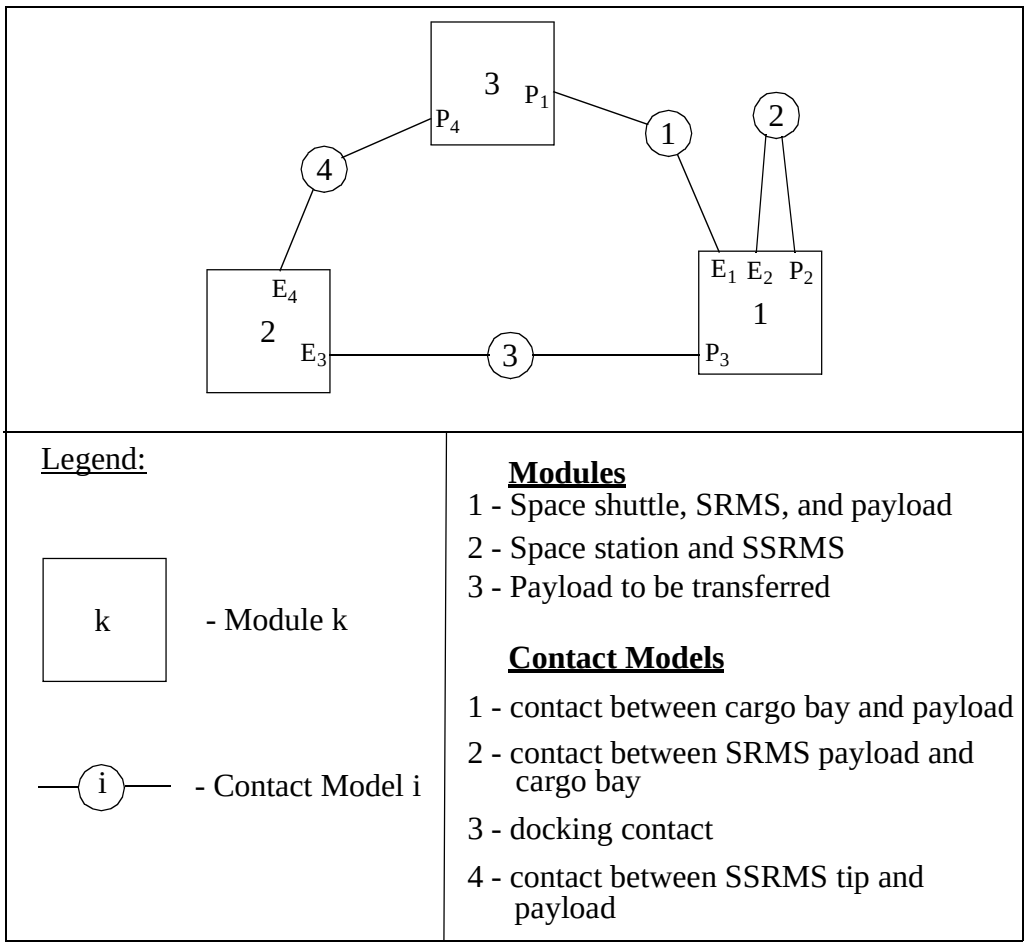

FIGURE 17. Modular Decomposition for the System Shown in Figure 16

TABLE 7. Numbering the System Modules and Contacts

Symbol	Description
C	the total number of contact models used in the system
M	the total number of modules used in the system
i, j	contact model indices in the range from 1 to C
k	module number index in the range from 1 to M

Each contact model requires the specification of two reference frames, one on each side of the contact. These frames, as well as the rigid bodies to which they are attached, are again

labeled using the letters E and P, as described in Chapter 3 (see Figure 13). Since there are now multiple contacts, however, these labels are subscripted with the number of the corresponding contact model. In this manner, the "i^{th}" contact model connects rigid bodies E_i and P_i, each of which has a fixed reference frame, denoted as E_i and P_i, respectively. The absolute reference frame for the system is again denoted by I, since it is considered to be inertially fixed.

The points at the origins of frames E_i and P_i are again referred to as *operational points*. Each of these points is used as the reference point for an operational space dynamic equation [30]. In order to present the equations of this paper as compactly as possible, the symbol O_i is used in equations that apply equally to all operational points, regardless of distinction between E_i and P_i. One final note regarding operational points is the use of the function $m(O_i)$, which returns the number of the module that contains body (and reference frame) O_i. These notational conventions are summarized in Table 8.

From our example system, contact model i = 4 connects point E_4 on the SSRMS and space station module to point P_4 on the payload. In another example of this notation, the function $m(E_4)$ has a value of 2 since operational point E_4 belongs to a body in module 2.

TABLE 8. Labeling the Operational Points

Symbol	Description
E_i	rigid body and coordinate frame label in the range from E_1 to E_C
P_i	rigid body and coordinate frame label in the range from P_1 to P_C
O_i	generic label for an operational point (can represent E_i or P_i)
$m(O_i)$	function which returns the number of the module containing frame and body O_i

5.2 New Forms of the Operational Space Inertia Matrix

The operational space dynamic formulation, and consequently, the operational space inertia matrix, both play a prominent role in the algorithm of Lilly [39][40] and in the method of Chapter 4. The same is true in the modular algorithm presented here. Due to the presence of multiple contacts, however, the basic operational space inertia matrix is not, by itself, sufficient within the operational space dynamic equations for complex systems.

From its original development by Khatib [30], the operational space inertia matrix relates spatial forces at an operational point to spatial accelerations of the same point. This function is represented pictorially in Figure 18 (Figure 12 repeated here for convenience). For systems with multiple contacts, the operational space dynamic formulation must be able to relate the spatial forces acting at one operational point to the spatial accelerations of another point, as depicted in Figure 19.

In Chapter 4 (see Equation 60), the solution for the contact force vector acting between two separate structures involved two inverse operational space inertia matrices: ${}^{I}\Lambda_{EE}^{-1}$ and ${}^{I}\Lambda_{PP}^{-1}$. Each of these matrices is an example of the original operational space inertia matrix (albeit inverted). Both of these matrices can be expressed using a common form:

$$\Lambda^{-1} = J \cdot H^{-1} \cdot J^{T} \tag{68}$$

where for ${}^{I}\Lambda_{EE}^{-1}$, the joint space inertia matrix, H, is for Structure E and the Jacobian matrix pertains to the kinematics of operational point E.

This same general form can again be used to describe a new type of operational space inertia matrix. In this case, however, two different Jacobians are involved in the mathemat-

ical expression: one for the first operational point and one for the second. For any pair of operational points, O_i and O_j, belonging to a common module, this new form can be given as:

$$\Lambda^{-1}_{O_i O_j} = {}^{I}J_{IO_i} \cdot H^{-1} \cdot {}^{I}J^{T}_{IO_j} \tag{69}$$

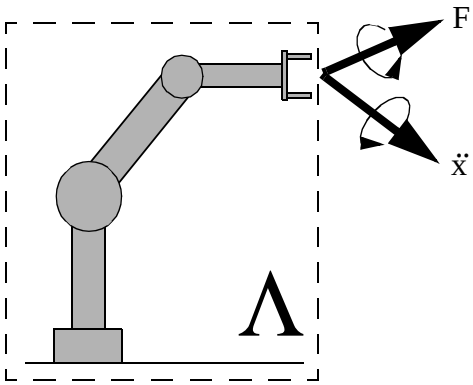

FIGURE 18. Visualization of the Operational Space Inertia Matrix

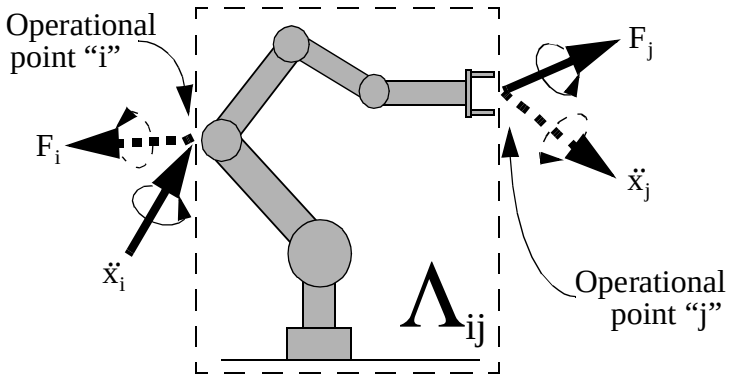

FIGURE 19. Visualization of the Multi-point Operational Space Inertia Matrix

This expression is similar in form to Equation 68. In fact, when i and j are equal, this formula is equivalent to that of the original operational space inertia matrix, and can therefore be used to represent $\Lambda^{-1}_{E_iE_i}$ and $\Lambda^{-1}_{P_iP_i}$. When i and j are not equal, we have the new form, which we will call the *multi-point operational space inertia matrix*. In this instance, Equation 69 can be used to produce $\Lambda^{-1}_{E_iP_j}$ and $\Lambda^{-1}_{P_iE_j}$.

The multi-point operational space inertia matrix was first developed by this author in 1995 and was presented in a proposal for this work in 1996. At the same time, Khatib, the creator of the original operational space framework, extended his work, independently arriving at the same result [60]. As with his earlier works, Khatib's operational space dynamics and these new inertia matrices were applied to the problem of robot control. In [11], he went on to develop an O(N) algorithm for $\Lambda^{-1}_{O_iO_j}$ $(i \neq j)$ for tree-structured (branching) manipulators. The use of $\Lambda^{-1}_{O_iO_j}$ here is the first known application to dynamic simulation.

Although Equation 69 gives the proper form of the inverse operational space inertia matrix, it does not sufficiently distinguish one structure from another, nor does it assure existence of the matrix (notationally). A more complete formulation, which specifies the precise joint space inertia matrix involved and is valid for all operational points in the system, is given by the following:

$$\Lambda^{-1}_{O_iO_j} = \begin{cases} {}^{I}J_{IO_i} \cdot H^{-1}_{m(O_i)} \cdot {}^{I}J^{T}_{IO_j} & \text{if } m(O_i) = m(O_j) \\ 0 & \text{if } m(O_i) \neq m(O_j) \end{cases} \tag{70}$$

Equation 70 specifies that the joint space inertia matrix indicated is the one for the structure (module) that contains both operational points O_i and O_j. Also, it shows that if O_i and

O_j are not contained within the same structure, then the inverse operational space inertia matrix relating the forces and accelerations associated with these two points, O_i and O_j, is a zero matrix.

Yet another wrinkle in these inertia matrices is the possible occurrence of a Type B contact. As shown in Chapter 4, it is possible with Type B contacts to consolidate terms, leading to a potential increase in computational efficiency. This was achieved using the "relative Jacobian matrix" (see Equation 48 in Section 4.2.2). The updated form of the relative Jacobian in the notation for a multi-contact system is as follows:

$$ {}^{E_i}J_{E_iP_i} = {}^{E_i}\Re_I \cdot ({}^{I}J_{IP_i} - {}^{I}S_{E_iP_i} \cdot {}^{I}J_{IE_i}) \tag{71} $$

Allowing for this possibility, three additional forms of the inverse operational space inertia matrix can be added to the one from Equation 70. All four cases are summarized in Table 9. Note that the subscript R on Λ^{-1} denotes the use of a relative Jacobian. Further discussion of the terms in Table 9 is given in Section 5.5.2.

TABLE 9. New Forms of the Operational Space Inertia Matrix

Contact		**if Contact Models i and j**	
i is Type	**j is Type**	**are both connected to module k**	**are not both connected to module k**
A	A	$\Lambda^{-1}_{O_iO_j} = {}^{I}J_{IO_i} \cdot H_k^{-1} \cdot {}^{I}J^{T}_{IO_j}$	$\Lambda^{-1}_{O_iO_j} = 0$
A	B	$\Lambda^{-1}_{O_iR_j} = {}^{I}J_{IO_i} \cdot H_k^{-1} \cdot {}^{E_j}J^{T}_{E_jP_j}$	$\Lambda^{-1}_{O_iR_j} = 0$
B	A	$\Lambda^{-1}_{R_iO_j} = {}^{E_i}J_{E_iP_i} \cdot H_k^{-1} \cdot {}^{I}J^{T}_{IO_j}$	$\Lambda^{-1}_{R_iO_j} = 0$
B	B	$\Lambda^{-1}_{R_iR_j} = {}^{E_i}J_{E_iP_i} \cdot H_k^{-1} \cdot {}^{E_j}J^{T}_{E_jP_j}$	$\Lambda^{-1}_{R_iR_j} = 0$

Returning again to our example system, Table 10 shows all forms of the operational space inertia matrix that would be computed for each module in the system. Note that relative forms can be used for all terms corresponding to contact model 2 attached at both ends to module 1.

The inverse operational space inertia matrices given by Table 9 are computed at the modular level and passed to the global level as shown in the flowchart of Figure 15 (represented generically by Λ_{ij}^{-1}). These matrices are, in fact, the basic building blocks used to construct the linear system of equations in the unknown contact forces. The exact construction of that system of equations, and hence the proper use of these matrices, is based solely on the structural topology of the system. System topology and the variables influenced by it are discussed in the next section.

TABLE 10. Operational Space Inertia Matrices for the Example System

Module	Required Operational Space Inertia Matrices[a]
1	$\Lambda_{E_1E_1}^{-1}$, $\Lambda_{E_1R_2}^{-1}$, $\Lambda_{E_1P_3}^{-1}$, $\Lambda_{R_2R_2}^{-1}$, $\Lambda_{R_2P_3}^{-1}$, $\Lambda_{P_3P_3}^{-1}$
2	$\Lambda_{E_3E_3}^{-1}$, $\Lambda_{E_3E_4}^{-1}$, $\Lambda_{E_4E_4}^{-1}$
3	$\Lambda_{P_1P_1}^{-1}$, $\Lambda_{P_1P_4}^{-1}$, $\Lambda_{P_4P_4}^{-1}$

a. Additional required matrices are denoted by switching the subscript order in the matrices listed above. If the joint space inertia matrix is symmetric, these additional matrices are simply the transpose of those listed.

5.3 System Connectivity: Functions and New Variable Structures

In Chapter 4, the constrained motion algorithm involved the projection of the operational space dynamics of the system into the constraint space, which resulted in a linear system of equations in the unknown contact forces. The same procedure is again used here. Now, however, there will be multiple operational space dynamic equations with one corresponding to each contact model in the system.

5.3.1 Global Variables

In order to combine dynamics of the entire system and produce a single system of linear equations, the algorithm makes use of a number of global level variables. These variables will be vectors and matrices composed of “block” elements. Each block element of a global vector will be associated with one of the contact models in the system, while each block element in a global matrix will correspond to a pair of the system’s contact models. In this section, these global variables are presented in terms of their block structure and dimensions. To simplify the overall dimensions, we define the following total:

$$n_C = \sum_{i=1}^{C} n_{c_i} \tag{72}$$

where n_{c_i} is the number of unknown scalar contact forces that must be determined for the “i^{th}” contact model. The number n_C gives the total number of constrained (velocity) directions of the global system, and it is therefore the dimension of the linear system of equations formed and solved at the global level of the algorithm.

The global variables used in the algorithm are shown in Table 11. An example of each

variety of "block structure" is given in Figure 20. Note that global variables appear in bold. Each of the variables shown in Table 11 will be defined specifically when it appears in the derivation of the algorithm equations in Section 5.4.

TABLE 11. Global Variables

Global Variables	Structure	Dimension	Dimension of Block Elements
${}^{EP}\mathbf{F}_{EP}$, ${}^{E}\ddot{\mathbf{x}}_{EP}$, ${}^{E}\ddot{\mathbf{x}}_{EP}^{open}$, $\mathbf{y}$	Block Vector	$6C \times 1$	6×1
$\mathbf{F}^c$, $\ddot{\mathbf{x}}^c$, $\mathbf{y}^c$		$n_C \times 1$	$n_{c_i} \times 1$
$\mathbf{F}^u$		$(6C - n_C) \times 1$	$(6 - n_{c_i}) \times 1$
ψ^c	Block Diagonal Matrix	$6C \times n_C$	$6 \times n_{c_i}$
ψ^u		$6C \times (6C - n_C)$	$6 \times (6 - n_{c_i})$
$\mathbf{B}$	Block Matrix	$6C \times 6C$	6×6
$\mathbf{B}^c$		$n_C \times n_C$	$n_{c_i} \times n_{c_i}$

Block Vector Block Diagonal Matrix Block Matrix

$$ {}^{EP}\mathbf{F}_{EP} = \begin{bmatrix} {}^{E_1P_1}F_{E_1P_1} \\ \cdots \\ {}^{E_jP_j}F_{E_jP_j} \\ \cdots \\ {}^{E_CP_C}F_{E_CP_C} \end{bmatrix} \qquad \psi^c = \begin{bmatrix} \psi_1^c & 0 & \cdots & \cdots & 0 \\ 0 & & & & \\ & & \psi_i^c & & \\ & & & & 0 \\ 0 & \cdots & \cdots & 0 & \psi_C^c \end{bmatrix} \qquad \mathbf{B} = \begin{bmatrix} B_{11} & B_{12} & \cdots & \cdots & B_{1C} \\ B_{21} & & & & \\ & & B_{ij} & & \\ & & & & \\ B_{C1} & \cdots & \cdots & & B_{CC} \end{bmatrix} $$

FIGURE 20. Examples of "Block Structure" for Global Variables

5.3.2 Cross Level Variables (Modular and Global)

The variables shown below in Table 12 are the cross level variables that are used to map

information between the global and modular levels. Each of these variables is associated with a specific module (Module k), while the block elements of each variable correspond to each and every contact model in the global system. The exact definition of each block element is dependent upon the connectivity of the module, as will be shown in Section 5.4.1 and Table 15. As with the global variables, these cross level variables also appear in bold.

TABLE 12. Cross Level Variables (Modular and Global)

Mixed Variables	Structure	Dimension	Dimension of Block Elements
$\mathbf{F}_k$	Block Vector	$6C \times 1$	6×1
$\mathbf{J}_k$	Block Matrix	$6C \times N_k$	$6 \times N_k$
Ψ_k		$N_k \times 6C$	$N_k \times 6$
$\Lambda_{O_i}^{-1}$		$6 \times 6C$	6×6

Note: N_k is the number of single degree of freedom joints associated with Module k

5.3.3 Connectivity Functions

Two types of contact configurations were presented in Chapter 4. Type A contacts were those made between the module and some external object, such as a second module or an environmental boundary. Type B contacts were those made between two operational points belonging to the same module, such as the case of the space shuttle robot arm contacting a point in the cargo bay.

Given the possibility of Type A and B contacts, and the labeling of operational points as E_i or P_i, there are numerous ways in which the complex systems we wish to analyze can be connected and labeled. However, at both the modular and global levels, these combina-

tions can be found to fit into several categories, each of which leads to a precise form for the block elements of the global and cross level variables presented in the previous sections. These topological distinctions are discussed next

5.3.3.1 Modular Connectivity Function

In describing the connectivity of complex systems, four possible conditions will be recurrent in our presentation of the dynamic equations. To keep track of these possibilities, we define a *modular connectivity function*, MC(k, j). The two inputs to the function are a module number, k, and a contact model number, j. For any pair of inputs from the system, there will be four possible outcomes, as defined in Figure 21 and Table 13.

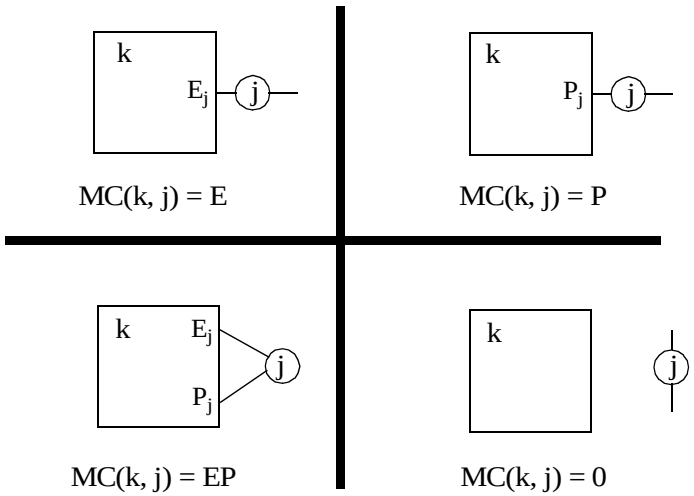

FIGURE 21. Visualization of the Modular Connectivity Function

TABLE 13. Modular Connectivity Function

MC(k, j) Equals	when Contact Model j is Type	joined to module k at	Mathematical Equivalent	Example System
E	A	E_j only	$m(E_j) = k$ and $m(P_j) \neq k$	k=1, j=1
P	A	P_j only	$m(E_j) \neq k$ and $m(P_j) = k$	k=1, j=3
EP	B	E_j and P_j	$m(E_j) = k$ and $m(P_j) = k$	k=1, j=2
0	not connected to module k		$m(E_j) \neq k$ and $m(P_j) \neq k$	k=1, j=4

Note: k = 1 to M for each module, j = 1 to C for each contact model

Table 13 shows two definitions for each outcome of the modular connectivity function: one in a more visual sense of the system topology, the other in more mathematical terms using the function $m(O_i)$. The final column of Table 13 shows a module number and contact model number from the example system in Figure 17 that demonstrate the connectivity defined by each row.

Essentially, the modular connectivity function indicates whether or not contact model j connects to module k, and if it does, whether it connects to point E_j, P_j, or to both E_j and P_j. The four conditions are mutually exclusive so that only one value can be returned by the function for any input pair, k and j, in the system.

These four conditions establish mathematically the connectivity relationship between module k and every contact model in the system. The modular connectivity function will be useful for identifying the specific terms to be calculated at the modular level, as well as which force transformations are necessary after the global level solution for the standard contact force vectors. This will be shown in detail in Section 5.5.3.

5.3.3.2 Global Connectivity Function

Where the modular connectivity function is used to define the relationship between a module and every contact in the system, the *global connectivity function*, presented below in Figure 22 and Table 14, defines the relationship between every possible pair of contact models in the system. This function will be used to specifically determine the exact structure of the force coefficient matrix of the linear system of equations in the unknown contact forces that is generated and solved in the global level of the algorithm.

Again there are four possible outcomes for this function based on the presence of Type A and Type B contacts. Ultimately, it is the new forms of the operational space inertia matrices given in Table 9 of Section 5.2 that are placed in the global force coefficient matrix according to the global connectivity function. This process stems from the derivation of the full system dynamic equations in both joint and operational space. Section 5.4 shows the development of these equations, which form the foundation of the MRDS algorithm.

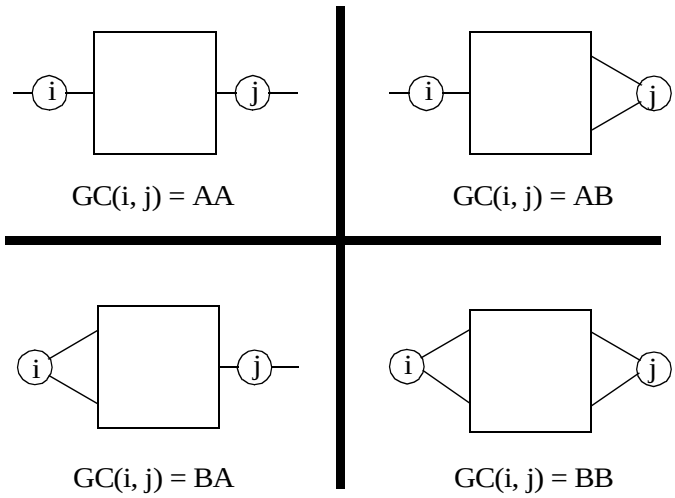

FIGURE 22. Visualization of the Global Connectivity Function

TABLE 14. Global Connectivity Function

GC(i, j) Equals	if Contact "i" is Type	and Contact "j" is Type	Mathematical Equivalent
AA	A	A	$m(E_i) \neq m(P_i)$ and $m(E_j) \neq m(P_j)$
BA	B	A	$m(E_i) = m(P_i)$ and $m(E_j) \neq m(P_j)$
AB	A	B	$m(E_i) \neq m(P_i)$ and $m(E_j) = m(P_j)$
BB	B	B	$m(E_i) = m(P_i)$ and $m(E_j) = m(P_j)$

Note: i = 1 to C, j = 1 to C where C is the number of contact models in the system

5.4 Development of the Algorithm Equations

In this section, the equations used in the constrained motion algorithm are developed following the same derivation used in Chapter 4. Now, however, the equations will accommodate multiple modules (structures) subject to multiple concurrent contacts. As before, this new algorithm uses the joint space and operational space dynamic equations of each structure together with the contact modelling technique to project the system dynamics into the constraint space. This allows the formation and solution of a linear system of equations in the unknown components of all contact forces. With the contact forces determined, the problem is effectively reduced to that of an open chain simulation. The computational steps of the algorithm will be summarized in Section 5.5.

5.4.1 Dynamics of Each Module

The algorithm derivation requires the dynamic equations for each module in the system. This includes both the joint space and operational space dynamic formulations. For each module, there will be *one* joint space dynamic equation (of dimension N_k), as well as *one* operational space dynamic equation (of dimension 6) for *each* operational point that the

module contains.

5.4.1.1 Joint Space Dynamics

The development again begins with the joint space dynamic formulation. However, with multiple structures (modules) in the system, each subject to multiple concurrent contacts, the joint space dynamic equations take on a slightly modified form. For some module "k", the new equations are as follows [68]:

$$\tau_k = H_k \cdot \ddot{q}_k + C_k + G_k + \mathbf{J}_k^T \cdot \mathbf{F}_k \tag{73}$$

Equation 73 can be solved for the joint accelerations of module k as follows:

$$\ddot{q}_k = \ddot{q}_k^{open} - \Psi_k \cdot \mathbf{F}_k \tag{74}$$

The term, $\ddot{q}_k^{open}$, is the generalized coordinate accelerations that would result if all external contact forces were set to zero for the current configuration (the current timestep of the simulation). Not present in the single contact algorithm are the terms $\mathbf{F}_k$, $\mathbf{J}_k$, and Ψ_k. These cross level variables from Section 5.3.2 arise from the fact that multiple contacts, and therefore multiple contact forces, may be acting on module k.

In the single contact algorithm, only one contact model was present in the system, and therefore, only one spatial contact force vector. In the above equations, <u>all</u> of the system's contact force vectors are included in the term $\mathbf{F}_k$, including those not in direct contact with the module. Similar to the use of spatial notation, this approach allows for a compact presentation of the dynamic equations.

The exact elements of the new terms, $\mathbf{F}_k$, $\mathbf{J}_k$, and Ψ_k, are dependent on the connectiv-

ity of module k within the system, which is in turn expressed by the modular connectivity function outlined in Table 13 of Section 5.3.3.1. As discussed, there are four possible connectivities for a module based on the notation and numbering choices. Table 15 shows the exact definitions of the block elements used within $\mathbf{F}_k$, $\mathbf{J}_k$, and Ψ_k that correspond to these choices. Once again, the system of Figure 17 is used as an example for each relationship.

TABLE 15. Terms for Modular Dynamics Dependent on Modular Connectivity

MC(k, j) Equals	Contact Model j is		j^{th} block element of			Example System
	Type	joined to k by	$\mathbf{J}_k$	$\mathbf{F}_k$	Ψ_k	
E	A	E_j	${}^{I}J_{IE_j}$	${}^{IE_j}F_{E_jP_j}$	${}^{I}\Psi_{IE_j}$	k=1, j=1
P	A	P_j	${}^{I}J_{IP_j}$	${}^{IP_j}F_{P_jE_j}$	${}^{I}\Psi_{IP_j}$	k=1, j=3
EP	B	E_j and P_j	${}^{E_j}J_{E_jP_j}$	${}^{E_jP_j}F_{E_jP_j}$	${}^{E_j}\Psi_{E_jP_j}$	k=1, j=2
0	not connected to k		0	0	0	k=1, j=4

As shown in Table 15, the possible elements of the modular Jacobian, $\mathbf{J}_k$, include the traditional Jacobian matrices, ${}^{I}J_{IE_j}$ and ${}^{I}J_{IP_j}$, as well as the relative Jacobian matrix, ${}^{E_j}J_{E_jP_j}$. Elements of the modular force vector, $\mathbf{F}_k$, include ${}^{IE_j}F_{E_jP_j}$, ${}^{IP_j}F_{P_jE_j}$, and ${}^{E_jP_j}F_{E_jP_j}$, each a different form of the spatial contact force vector discussed in Section 3.2.3. Lastly, the elements of the term, Ψ_k, can include ${}^{I}\Psi_{IE_j}$, ${}^{I}\Psi_{IP_j}$, and ${}^{E_j}\Psi_{E_jP_j}$.

The matrices ${}^{I}\Psi_{IE_j}$ and ${}^{I}\Psi_{IP_j}$ can each be defined using the following generic expression:

$$ {}^{I}\Psi_{IO_j} = H^{-1}_{m(O_j)} \cdot {}^{I}J^{T}_{IO_j} \tag{75} $$

while ${}^{E_j}\Psi_{E_jP_j}$, which makes use of the relative Jacobian matrix, is given by:

$$ {}^{E_j}\Psi_{E_jP_j} = H^{-1}_{m(E_j)} \cdot {}^{E_j}J^{T}_{E_jP_j} \tag{76} $$

The terms, ${}^{I}\Psi_{IE_j}$, ${}^{I}\Psi_{IP_j}$, and ${}^{E_j}\Psi_{E_jP_j}$, are all examples of the cross space inertia matrix described in Section 4.2. Exact calculation of these terms and the associated computational complexities are discussed in Section 5.5.2.

5.4.1.2 Operational Space Dynamics

The next task is to develop the operational space dynamic equations for all of the operational points contained by the module. This can be accomplished by inserting the joint space dynamics of the module into a kinematic expression for the absolute acceleration of the operational point. The necessary kinematic expression for some operational point, O_i, can be obtained as follows:

$$ {}^{I}\dot{x}_{IO_i} = {}^{I}J_{IO_i} \cdot \dot{q}_{m(O_i)} \tag{77} $$

$$ {}^{I}\ddot{x}_{IO_i} = {}^{I}J_{IO_i} \cdot \ddot{q}_{m(O_i)} + {}^{I}\dot{J}_{IO_i} \cdot \dot{q}_{m(O_i)} \tag{78} $$

Introducing the joint space dynamics given by Equation 74 into this last expression, where we choose the joint space equation of the module that contains point O_i (module $k = m(O_i)$), the desired operational space dynamic equation is expressed as follows:

$$ {}^{I}\ddot{x}_{IO_i} = {}^{I}\dot{x}^{open}_{IO_i} - \Lambda^{-1}_{O_i} \cdot \mathbf{F}_{m(O_i)} \tag{79} $$

This equation applies for all operational points in the system (i = 1 to C). The open chain

term, $\ddot{x}^{I,open}_{IO_i}$, is the spatial acceleration of frame O_i that would result if all external contact forces were removed from the module. Also note that the term, $\mathbf{F}_{m(O_i)}$, is the same as $\mathbf{F}_k$. For consistency, it has merely been identified in terms of the operational point and not the module number.

Like the other cross level variables, $\mathbf{F}_k$, $\mathbf{J}_k$, and Ψ_k of Table 15, the block elements of $\Lambda^{-1}_{O_i}$ can also be defined according to the modular connectivity function. However, in order for Equation 79 to apply to any operational point on any module, we express the module number "k" that is input to the modular connectivity function in terms of the operational point using function $m(O_i)$. Consequently, the "j^{th}" block element of $\Lambda^{-1}_{O_i}$ can be determined as shown in Table 16.

TABLE 16. Operational Space Inertia Matrices Dependent on Modular Connectivity

MC($m(O_i)$, j) Equals	**Contact Model j is**		**j^{th} block element of $\Lambda^{-1}_{O_i}$**	**Example System**
	Type	**joined to $m(O_i)$ by**		
E	A	E_j	$\Lambda^{-1}_{O_iE_j}$	$O_i = P_3$, j=1
P	A	P_j	$\Lambda^{-1}_{O_iP_j}$	$O_i = P_3$, j=3
EP	B	E_j and P_j	$\Lambda^{-1}_{O_iR_j}$	$O_i = P_3$, j=2
0	not connected to module $m(O_i)$		0	$O_i = P_3$, j=4

The possible block elements of $\Lambda^{-1}_{O_i}$ include $\Lambda^{-1}_{O_iE_j}$, $\Lambda^{-1}_{O_iP_j}$, and $\Lambda^{-1}_{O_iR_j}$. The first two matrices, $\Lambda^{-1}_{O_iE_j}$ and $\Lambda^{-1}_{O_iP_j}$, are each defined by the generic expression of Equation 70, while $\Lambda^{-1}_{O_iR_j}$, which makes use of the relative Jacobian matrix, is given in Table 9. All of these three terms are examples of the new forms of the inverse operational space inertia matrix

described in Section 5.2.

Functionally, each block element of $\Lambda_{O_i}^{-1}$ (one per contact model) is the inertia matrix that relates the spatial force associated with one contact model to the absolute acceleration of frame O_i. When a contact model does not have a direct connection to the module containing frame O_i, the associated inertia (block element of $\Lambda_{O_i}^{-1}$) is zero, as shown in Table 16 for the fourth possible value of the modular connectivity function. The computation of these inertia matrices is discussed in Section 5.5.

5.4.2 Dynamics of the Global System

Equation 79 represents the operational space dynamic equation for any operational point in the system. It provides an expression that relates the contact forces from the entire system to the absolute acceleration of one operational point belonging to one module. As demonstrated in Chapter 4, however, the proper use of a contact model requires an operational space dynamic equation which includes the dynamics of the structures on *both* sides of the contact and that gives the expression for the *relative* acceleration across the contact, not an absolute acceleration.

The proper dynamic equation can be generated starting with the kinematic expression for the relative acceleration across the contact. This was derived in Chapter 4 and is repeated here for some contact model "i" (i = 1 to C) using the updated notation as:

$$ {}^{E_i}\ddot{x}_{E_iP_i} = {}^{E_i}\Re_I \cdot ({}^{I}\ddot{x}_{IP_i} - {}^{I}S_{E_iP_i} \cdot {}^{I}\ddot{x}_{IE_i} - {}^{I}\zeta_{E_iE_iP_i}) \qquad (80) $$

where

$$ {}^{I}\zeta_{E_iE_iP_i} = \begin{bmatrix} {}^{I}\omega_{IE_i} \times [{}^{I}\omega_{IE_i} \times {}^{I}R_{E_i} \cdot {}^{E_i}r_{E_iP_i}] + 2 \cdot {}^{I}\omega_{IE_i} \times {}^{I}R_{E_i} \cdot {}^{E_i}\dot{r}_{E_iP_i} \\ {}^{I}\omega_{IE_i} \times {}^{I}R_{E_i} \cdot {}^{E_i}\omega_{E_iP_i} \end{bmatrix} \tag{81} $$

is again used simply to keep the equation above and those that follow as compact as possible. Described in further detail in Chapter 6, the terms ${}^{E_i}r_{E_iP_i}$, ${}^{E_i}\dot{r}_{E_iP_i}$, and ${}^{E_i}\omega_{E_iP_i}$ in ${}^{I}\zeta_{E_iE_iP_i}$ are found from the kinematic relationships between operational points E_i and P_i and can be specially computed to avoid constraint violations in serial connections.

To produce the operational space dynamic equation that allows proper use of the contact model, the terms ${}^{I}\ddot{x}_{IE_i}$ and ${}^{I}\ddot{x}_{IP_i}$ in Equation 80 are each replaced using Equation 79: once for operational point $O_i = E_i$ and again for point $O_i = P_i$. Stacking all "C" of the resulting equations into one global system of equations, we can produce the following compact form:

$$ {}^{E}\ddot{\mathbf{x}}_{EP} = {}^{E}\ddot{\mathbf{x}}_{EP}^{open} + \mathbf{B} \cdot {}^{EP}\mathbf{F}_{EP} \tag{82} $$

Each block element of ${}^{E}\ddot{\mathbf{x}}_{EP}^{open}$ gives the open chain relative acceleration across contact model i (i = 1 to C). This term is the same as that derived in Chapter 4, and it is repeated here with updated notation:

$$ {}^{E_i}\ddot{x}_{E_iP_i}^{open} = {}^{E_i}\Re_I \cdot ({}^{I}\ddot{x}_{IP_i}^{open} - {}^{I}S_{E_iP_i} \cdot {}^{I}\ddot{x}_{IE_i}^{open} - {}^{I}\zeta_{E_iE_iP_i}) \tag{83} $$

The construction of the block elements of $\mathbf{B}$ is given in Table 17. A more in depth examination of this matrix is deferred until the completion of the algorithmic development.

Finally, it should be noted that all block elements of the global contact force vector,

${}^{EP}\mathbf{F}_{EP}$, are already in the "standard form", ready for use with the contact model.

TABLE 17. Structure of B

if GC(i, j) Equals	**then B_{ij}, the i, j Block Element of $\mathbf{B}$, is Equal to**
AA	${}^{E_i}\Re_I \left[[\Lambda^{-1}_{P_iP_j}] - [\Lambda^{-1}_{P_iE_j} \cdot {}^{I}S^{T}_{E_jP_j}] - [{}^{I}S_{E_iP_i} \cdot \Lambda^{-1}_{E_iP_j}] + [{}^{I}S_{E_iP_i} \cdot \Lambda^{-1}_{E_iE_j} \cdot {}^{I}S^{T}_{E_jP_j}] \right] {}^{I}\Re_{E_j}$
BA	$\left[[\Lambda^{-1}_{R_iP_j}] - [\Lambda^{-1}_{R_iE_j} \cdot {}^{I}S^{T}_{E_jP_j}] \right] {}^{I}\Re_{E_j}$
AB	${}^{E_i}\Re_I \left[[\Lambda^{-1}_{P_iR_j}] - [{}^{I}S_{E_iP_i} \cdot \Lambda^{-1}_{E_iR_j}] \right]$
BB	$\Lambda^{-1}_{R_iR_j}$

5.4.3 Constraint Space Dynamics and Determination of the Contact Forces

Equation 82 represents the operational space dynamics of the entire system of modules and contacts. Additionally, it is in the form that allows proper application of the contact models, as described by the "standard form" developed in Chapter 4.1. The equations for the standard form of the contact model, expressed in the notation of a multi-contact system, are as follows:

$${}^{E_i}\dot{x}_{E_iP_i} = \phi^u_i \cdot \dot{x}^u_i + \phi^c_i \cdot \dot{x}^c_i \tag{84}$$

$${}^{E_i}\ddot{x}_{E_iP_i} = \phi^u_i \cdot \ddot{x}^u_i + \phi^c_i \cdot \ddot{x}^c_i + \dot{\phi}^u_i \cdot \dot{x}^u_i + \dot{\phi}^c_i \cdot \dot{x}^c_i \tag{85}$$

$${}^{E_iP_i}F_{E_iP_i} = \psi^u_i \cdot F^u_i + \psi^c_i \cdot F^c_i \tag{86}$$

Introduction of the contact models is accomplished by replacing each block element, ${}^{E_i}\ddot{x}_{E_iP_i}$, of ${}^{E}\ddot{\mathbf{x}}_{EP}$ with the contact model expression given by Equation 85. Similarly, we replace each block element, ${}^{E_iP_i}F_{E_iP_i}$, of ${}^{EP}\mathbf{F}_{EP}$ with the expression given by Equation 86.

The resulting system of equations can then be projected from the operational space into the constraint space by taking advantage of the properties of the contact model vector spaces given by Equation 25 of Chapter 3. This is achieved by pre-multiplying each block "i" of the global system of equations by the matrix, $(\psi_i^c)^T$, or equivalently by pre-multiplying the entire global system of equations by the block diagonal matrix $(\psi^c)^T$ (see Table 11 and Figure 20). This process eliminates each of the unknown relative accelerations given in Equation 85 by the term $\ddot{x}_i^u$.

This process results in a linear system of equations for the global system where the only unknown terms are the components of the contact forces in the constrained directions, collectively denoted by the global block vector $\mathbf{F}^c$. This system of equations has precisely the same form as that derived in Chapter 4, only it is of larger dimension due to the larger number of contact models. This expression is as follows:

$$\mathbf{B}^c \cdot \mathbf{F}^c = \mathbf{y}^c \tag{87}$$

The global coefficient matrix, $\mathbf{B}^c$ is given by:

$$\mathbf{B}^c = (\psi^c)^T \cdot \mathbf{B} \cdot \psi^c \tag{88}$$

where the construction of the block elements of $\mathbf{B}$ was given in Table 17. The remaining term, $\mathbf{y}^c$, from Equation 87 may be expressed as follows:

$$\mathbf{y}^c = \ddot{\mathbf{x}}^c + (\psi^c)^T \cdot [\mathbf{y} - \mathbf{B} \cdot \psi^u \cdot \mathbf{F}^u] \tag{89}$$

where the "i^{th}" block element of $\mathbf{y}$ is defined as:

$$y_i = \dot{\phi}_i^u \cdot \dot{x}_i^u + \dot{\phi}_i^c \cdot \dot{x}_i^c - {}^{E_i}\dot{x}_{E_iP_i}^{open} \tag{90}$$

Examining more closely the conditions given in Table 17 for the construction of the matrix, $\mathbf{B}$, we find that the terms required may be placed solely on the basis of the global connectivity function described in Section 5.3.3.2. Therefore, simply by examining the topology of the system, all of the required terms, such as the inverse operational space inertia matrices, as well as the spatial rigid body rotation and translation matrices, can be specifically identified and their placement in the system of equations pin-pointed.

It should also be noted that many terms in the construction of **B** may be zero, since many of the specified operational space inertia matrices will involve pairs of operational points that do not belong to the same module. In fact, when a pair of contact models, i and j, have no associated operational points belonging to the same module, the entire term, B_{ij}, will be a zero matrix.

Equations 87 through 90 complete the development of the equations required by the MRDS algorithm for simulation of a complex system of multiple modules and multiple contacts. The sequence of the computational steps used to apply the algorithm and issues of computational complexity are discussed in the next section.

5.5 Steps of the Modular Algorithm

The computational process for this algorithm follows the same pattern as that used for

single contact systems. This process is summarized in Figure 15, which shows both the global and modular levels of the algorithm, as well as the flow of data between these two levels. A more specific discussion of the computational process follows.

5.5.1 Step 1: Modular Decomposition and Contact Model Specification

The first step in applying this algorithm is for the analyst to supply information for initialization. This requires the analyst to decompose the system into modules (numbered from 1 to M) and contacts (numbered from 1 to C) in a fashion similar to that shown in Figure 17. After the specification and numbering of the modules and contact models, the next step is to label the operational points in the system. For each contact model, a choice must be made as to which side will be considered the “E” side and which side will be the “P” side. For the purposes of notation, the body on the “E” side of contact “i”, as well as the coordinate frame rigidly attached to that body, will be denoted as E_i. Similarly, the label P_i is used for the body and frame on the “P” side of contact “i”. Note that if the contact mode vectors will be constant in one of these frames, then that side should be given the “E” label. This will avoid the need to supply expressions for the temporal derivative of the contact mode vectors.

The procedures above address the notational issues used in identifying the topology of the system. The next task required is to specify the exact structure of the contact models in the system. The details of this process were discussed at length in Chapter 4. In this case, the specifications addressed by Table 1 of Section 4.1 must be repeated for all “C” of the contact models in the system.

5.5.2 Step 2: Dynamics of the Individual Modules

The second step of the algorithm belongs to the modular level and is therefore required for each module in the system. This feature makes the use of parallel computation ideal for increased efficiency and real-time capability. In this step, the open chain dynamics of the module are computed. The open chain generalized coordinate accelerations, $\ddot{q}_k^{open}$, may be found using any convenient method, such as the methods in [8], [68], or [70]. The remaining computations at the modular level then depend on the various connectivity cases.

The open chain spatial acceleration, $\ddot{x}^{open}$, shown in Figure 15, is a generic term used to account for both Type A and Type B contacts. For each operational point, O_i, in the module associated with a Type A contact, the open chain inertial acceleration, ${}^{I}\ddot{x}_{IO_i}^{open}$, must be computed, again using any convenient method (e.g. [8]). For Type B contacts, it is the relative acceleration vector, ${}^{E_i}\ddot{x}_{E_iP_i}^{open}$, that is needed. This may be found by an efficient open chain means[18] or following the formula of Equation 83, which would in turn require computation of the absolute accelerations of both contact points (${}^{I}\ddot{x}_{IE_i}^{open}$ and ${}^{I}\ddot{x}_{IP_i}^{open}$).

The next requirement at the modular level is the computation of the force coefficient matrices. Similar to the possibilities for the spatial accelerations just discussed, we require the matrix, ${}^{I}\Psi_{IO_i}$, for each operational point associated with a Type A contact, while ${}^{E_j}\Psi_{E_jP_j}$ is needed for each Type B contacts with the module. Cross space inertia matrices in the form of ${}^{I}\Psi_{IO_i}$ may be determined using the direct multiplication method of Equation 75 or by implementing the efficient $O(N_k)$ algorithms presented in [39].

For the Type B form, ${}^{E_j}\Psi_{E_jP_j}$ can also be computed by direct multiplication, as given by

18. such as an adapted version of the method in [8]

Equation 76. An $O(N_k)$ alternative can be found by inserting the definition of the relative Jacobian, expanding terms, and using the definition of the Type A forms of the multi-point inertia matrix. This results in the following expression:

$$ {}^{E_j}\Psi_{E_jP_j} = ({}^{I}\Psi_{IP_i} - {}^{I}\Psi_{IE_i} \cdot {}^{I}S^{T}_{E_jP_j}) \cdot {}^{I}\Re_{E_j} \tag{91} $$

where the $O(N_k)$ methods of [39] are used to find ${}^{I}\Psi_{IP_i}$ and ${}^{I}\Psi_{IE_i}$.

In Figure 15, Λ^{-1}_{ij} is used to represent all forms of the inverse operational space inertia matrix that are computed at the modular level and passed to the global level. There are four possible forms of the inertia matrix, corresponding to the four values of the global connectivity function given in Section 5.3.3.2.

The operational space inertia matrices of the form $\Lambda^{-1}_{O_iO_j}$ are required when contacts i and j both make Type A contact with the module in question. If contact i is Type A and j is Type B, then the matrix $\Lambda^{-1}_{O_iR_j}$ is needed. Conversely, if i is Type B and j is Type A, we require $\Lambda^{-1}_{R_iO_j}$. Finally, if both contacts i and j are Type B contacts attached to the module in question, then $\Lambda^{-1}_{R_iR_j}$ should be computed. Computation of any of these forms of the inverse operational space inertia matrices can be achieved by the direct multiplication methods given by the equations in Table 9.

A more efficient order (N_k) method for the traditional inverse operational space inertia matrices, $\Lambda^{-1}_{E_iE_i}$ or $\Lambda^{-1}_{P_iP_i}$, can be found in [39]. A similar method has been developed for the multi-point forms of the matrix, $\Lambda^{-1}_{E_iP_j}$ and $\Lambda^{-1}_{P_iE_j}$, provided the individual robot (module) is a branching structure. This method can be found in [11].

$O(N_k)$ methods for the three "relative" forms can again be obtained by inserting the for-

mula for the relative Jacobian into the equations of Table 9 and grouping terms appropriately. This results in the following alternate formulations:

$$\Lambda^{-1}_{O_iR_j} = \left[[\Lambda^{-1}_{O_iP_j}] - [\Lambda^{-1}_{O_iE_j} \cdot {}^{I}S^{T}_{E_jP_j}] \right] {}^{I}\Re_{E_j} \tag{92}$$

$$\Lambda^{-1}_{R_iO_j} = {}^{E_i}\Re_{I} \left[[\Lambda^{-1}_{P_iO_j}] - [{}^{I}S_{E_iP_i} \cdot \Lambda^{-1}_{E_iO_j}] \right] \tag{93}$$

$$\Lambda^{-1}_{R_iR_j} = {}^{E_i}\Re_{I} \left[[\Lambda^{-1}_{P_iP_j}] - [\Lambda^{-1}_{P_iE_j} \cdot {}^{I}S^{T}_{E_jP_j}] - [{}^{I}S_{E_iP_i} \cdot \Lambda^{-1}_{E_iP_j}] + [{}^{I}S_{E_iP_i} \cdot \Lambda^{-1}_{E_iE_j} \cdot {}^{I}S^{T}_{E_jP_j}] \right] {}^{I}\Re_{E_j} \tag{94}$$

As in Chapter 4, the entire algorithm developed in this chapter can be performed with O(N) computational complexity. When N_k is sufficiently small, however, Equations 91 through 94 may, be less efficient than the direct-multiplication methods shown in Table 9 and Equation 76. Consequently, if the direct multiplication methods will be used, it is prudent to select an open chain algorithm that computes $\dot{q}^{open}_k$, the various forms of $\ddot{x}^{open}$, H^{-1}_k, and the Jacobian matrices, ${}^{I}J_{IE_i}$ and ${}^{I}J_{IP_i}$. The method of Wang and Ravani [70] can be efficiently extended to achieve these goals.

Table 18 shows in detail the calculations performed for Step 2 at the modular level for each module in the system. All terms indicated in the table can be computed using a parallel processor for each module without regard for the contact force, the kinematic constraints, or the dynamics of the other modules in the system. The kinematic terms, ${}^{I}r_{IO_i}$, ${}^{I}R_{O_i}$, and ${}^{I}\dot{x}_{IO_i}$, are used at the global level of the algorithm for the calculation of the open chain relative spatial accelerations for each contact.

TABLE 18. Step 2: Dynamics of Each Individual Module Computed in Parallel

		Terms	Notes
		$\ddot{q}_k^{open}$	suitable open chain algorithm (e.g. [8][68][70])
		${}^{I}r_{IO_i}$, ${}^{I}R_{O_i}$, and ${}^{I}\dot{x}_{IO_i}$	suitable Forward Kinematic calculations
if i is Type A		${}^{I}\dot{x}_{IO_i}^{open}$; ${}^{I}\Psi_{IO_i}$	suitable open chain algorithm (e.g. [8][68][70]); Equation 75 or [39]
	and j is Type A	${}^{I}\Lambda_{O_iO_j}^{-1}$	Table 9 or [39] or [11]
	and j is Type B	${}^{I}\Lambda_{O_iR_j}^{-1}$	Table 9 or Equation 92
if i is Type B		${}^{E_i}\dot{x}_{E_iP_i}^{open}$, ${}^{E_i}\Psi_{E_iP_i}$	Equation 83; Equation 76 or 91
	and j is Type A	${}^{I}\Lambda_{R_iO_j}^{-1}$	Table 9 or Equation 93
	and j is Type B	${}^{I}\Lambda_{R_iR_j}^{-1}$	Table 9 or Equation 94

5.5.3 Step 3: Global Dynamics and Solution of Contact Forces

Once the required terms from each module are computed, they are passed to the global level of the algorithm. At this level, the terms ${}^{I}S_{E_iP_i}$ and ${}^{I}\zeta_{E_iE_iP_i}$ are computed for each contact. These terms depend only on the current position and velocity variables from the modules on either side of a contact and can therefore be derived using modular level kinematic computations. A further discussion of these terms is given in Chapter 6.

Using the modular level information, the global terms $\mathbf{B}^c$ and $\mathbf{y}^c$ are computed according to Equations 88, 89, 90, and the equations found in Table 17. It should be noted that the matrix, $\mathbf{B}$, and therefore $\mathbf{B}^c$, is symmetric, provided H_k^{-1} is symmetric for each module in the system. This is almost always the case for structures composed of rigid bodies.

With $\mathbf{B}^c$ and $\mathbf{y}^c$ computed, the linear system of equations for the global system, given by Equation 87, can be solved for the unknown components of each contact force vector, collectively denoted as $\mathbf{F}^c$. Solution of this equation may be accomplished using any efficient method such as LU or Singular Value Decomposition. Note that for most systems, the global coefficient matrix is likely to be sparse, or in the case of serially connected systems, banded. In such cases, more efficient solution methods can be employed [57].

With the components of the contact forces in the constrained directions now known, each of the 6×1 spatial contact force vectors, ${}^{E_iP_i}F_{E_iP_i}$, can be reconstructed according to Equation 86. These forces are then transformed as needed to produce the term $\mathbf{F}_k$ for each module (k = 1 to M), according to the modular connectivity summarized in Table 15. This is accomplished using the following transformations (as developed in Chapter 4):

$$ {}^{IE_i}F_{E_iP_i} = {}^{I}S^{T}_{E_iP_i} \cdot {}^{I}\Re_{E_i} \cdot {}^{E_iP_i}F_{E_iP_i} \tag{95} $$

$$ {}^{IP_i}F_{P_iE_i} = -{}^{I}\Re_{E_i} \cdot {}^{E_iP_i}F_{E_iP_i} \tag{96} $$

The terms needed for Step 3 are summarized in Table 19.

5.5.4 Step 4: Joint Accelerations and Integration to the Next State

As shown in Figure 15, the external contact forces for each module, $\mathbf{F}_k$, are passed back to the modular level, at which time the generalized coordinate accelerations of each module may be determined according to Equation 74. The simulation cycle then concludes with integration to produce the next state variables for each module. With the next state information available, all position and velocity terms are updated at both the modular and global

levels.

TABLE 19. Step 3: Dynamics of the Global System

Terms	Notes
${}^{I}\zeta_{E_iE_iP_i}$ and ${}^{I}S_{E_iP_i}$	Equation 81 and Equation 20
${}^{E_i}\ddot{x}^{open}_{E_iP_i}$	Equation 83
$\mathbf{B}$	Table 17
$\mathbf{B}^c$, $\mathbf{y}^c$	Equation 88 and 89, respectively
$\mathbf{F}^c$	Equation 87 using suitable linear system solver
${}^{E_iP_i}F_{E_iP_i}$	Equation 86
${}^{IE_i}F_{E_iP_i}$, ${}^{IP_i}F_{P_iE_i}$	Equation 95 and 96 (Type A only)

The calculations required by Step 4 are given in Table 20.

TABLE 20. Step 4: Joint Accelerations and Integration to the Next State

Terms	Notes
$\ddot{q}_k$	Equation 74
q_k, $\dot{q}_k$	Integration to next state

Note: These computations can be performed in parallel for each structure

5.6 Summary

In Chapter 4, the constrained motion algorithm of [39][40] was expanded to allow a more comprehensive range of contact conditions than ever before. Precise application of the contact models was established by choosing standard forms of the contact force vector

and relative acceleration that can be represented using the contact model vector spaces. The resulting algorithm allows the efficient simulation of one or two robots subject to one external constraint.

Here, in Chapter 5, the algorithm is again expanded to allow the dynamic simulation of more complicated systems involving multiple robots and multiple external contacts. The resulting algorithm is modular and allows the parallel calculation of the dynamics of each structure (module) in the system. This modularity provides the capacity to reconfigure the system by rearranging the modules and their connections without the need for significant remodelling or reprogramming. Furthermore, the algorithm is computationally efficient, as the entire set of steps can all be achieved with O(N) complexity.

A further extension of the MRDS algorithm in Chapter 7 allows the simulation of complex systems composed of flexible members. Before this extension, however, a few special issues are discussed in Chapter 6, including singularities, constraint violations, and the subtleties involved in the serial connection of two modules.

Chapter 6 SERIES CONNECTIONS, CONSTRAINT VIOLATIONS, AND OTHER SPECIAL TOPICS

This chapter presents a number of subtleties present in the application of the MRDS algorithm. These are related mostly to serial connections, Forward Kinematics, contact kinematics, and the suppression or possible avoidance of constraint violations. We begin with a general discussion of the MRDS treatment of a module connected in series.

6.1 Connection of Two Modules in Series Using Base Motion Variables

Application of the MRDS algorithm requires a full set of base motion variables for any module that, in its open chain state with all contacts removed, has no internally modelled connection to a fixed boundary. For modules like this, base motion variables are necessary for computation of the open chain terms, $\ddot{q}^{open}$ and ${}^{I}\dot{x}_{IO}^{open}$. Without the inclusion of full six-dimensional[19] base motion variables in the generalized coordinate, rate, and acceleration vectors (q, $\dot{q}$, and $\ddot{q}$), accurate computation of the module's instantaneously free-flying open chain behavior is not possible (without a loss of true modularity). Proper computation of the open chain spatial accelerations is necessary for the solution of the corrective contact force terms that determine the behavior of the constrained system.

For this reason, the MRDS algorithm is most efficient when applied to modules with more than just a few degrees of freedom. For example, allowing a module to represent a single revolute link of a serial chain would be an inefficient use of the MRDS approach. In

19. A size smaller than six may be appropriate for a task space of smaller size. For example, three dimensional base motion variables would be sufficient for planar systems.

this instance, six degrees of freedom are necessary in the module describing the link when, in total, only one additional coordinate is needed for the overall system.

Instead, the MRDS algorithm has been specifically developed for systems with multiple, cooperating, multi-degree-of-freedom, complex robots such as the space-based robots from Chapter 1. These structures are mobile and reconfigurable by nature. For complex robot systems such as these, base motion variables are almost certainly included in the existing dynamic equations, making the requirement for these variables a triviality.

More importantly, inclusion of these terms is what gives the MRDS algorithm its modularity for serial connections. These terms make it possible to reconfigure system modules quickly and easily. In essence, they provide a "handle" for the module so that it can be moved or attached in new ways by the modules that contact it. A module in the MRDS environment becomes a reusable, multi-faceted dynamic model.

A "less modular" alternative to the inclusion of full base-body motion in a serially-connected module is through augmentation of the joint space dynamics. This approach was presented by Lew and Book [37]. In this method, the joint space dynamics of two robots connected in series are constructed from the existing joint space dynamic modelling of each robot operating separately. Coupling terms in joint space are added to account for the interactions between the two, as opposed to the use of a general contact model. Although the overall dimension of the resultant equations is minimized, and the constraint forces are not needlessly obtained, this method does not have the freedom of the MRDS system.

Using the approach of Lew and Book, new coupling terms would be needed every time a new configuration is used. For example, the swapping of the distal robot to something

new, or the connection of the distal robot to an alternate base robot would always require new coupling terms to be derived and introduced into the dynamic simulation code. Attachment of a complex robot via numerous possible mounting points on an equally complex spacecraft would necessitate a unique set of coupling terms for each new case. This puts a heavy burden on the analyst and programmer. In addition, no method is discussed by Lew and Book for the closed-chain motion of two serially connected robots or for systems with more than two cooperating manipulators.

With the MRDS algorithm, a one-time addition of base motion variables[20] to the model and corresponding code of a module to be used in series will allow, in subsequent simulations, the resultant module to be manipulated by any MRDS module or kinematic input. This will give the analyst the freedom to connect the mobile robot module to nearly any object or device, or to provide any input kinematic base excitation, without the need to derive or encode new dynamic coupling terms. Moreover, the additional computational expense of adding dependent base motion variables is offset by the ability to compute terms for each module simultaneously on parallel processors. Ultimately, with base dynamics incorporated in the dynamic model, the module is free to be used and reused in innumerable ways and configurations.

By including full base motion variables in any module not directly connected to the inertial boundary in its internal modelling, all so-called “series” connections can effectively be treated no differently than parallel connections when using the MRDS algorithm. In both cases, the algorithmic requirements are the same. The open chain joint accelerations must

20. Assuming they aren’t already present in the available model.

be found at the modular level, along with the open chain spatial accelerations of all contact points. It is immaterial whether the contact point is part of a tip-to-tip contact or a tip-to-base connection. Serial connections do, however, provide additional modelling options related to the Forward Kinematics problem and the treatment of numerical constraint violations. These issues are covered in the next few sections.

6.2 Computation of Contact Model Kinematics

The MRDS algorithm requires the open chain relative acceleration across each contact in the system. Given by Equation 83, this term, ${}^{E_i}\ddot{x}^{open}_{E_iP_i}$, is calculated from the absolute inertial accelerations of the two operational points, ${}^{I}\ddot{x}^{open}_{IE_i}$ and ${}^{I}\ddot{x}^{open}_{IP_i}$. Also required in the calculation of ${}^{E_i}\ddot{x}^{open}_{E_iP_i}$ is the term containing the Coriolis and centrifugal terms for the contact, ${}^{I}\zeta_{E_iE_iP_i}$. From the definition of this term in Equation 81, we see that the current state relative position and velocity terms are necessary. Specifically, we must know ${}^{E_i}r_{E_iP_i}$, ${}^{E_i}\dot{r}_{E_iP_i}$, and ${}^{E_i}\omega_{E_iP_i}$. In addition, although not explicitly referenced by any equation, ${}^{E_i}R_{P_i}$ may also prove helpful.

In a simulation, position and velocity terms are considered to be known for the current state while trying to solve for the current accelerations. The issue here is how to compute these relative positions and velocities when in a modular framework and adhering to the general contact model. The MRDS procedure for this task is described here.

At the start of a simulation cycle, the generalized coordinate and velocity vectors for each module, q and $\dot{q}$, have just been determined from the integration of the last cycle's accelerations, $\ddot{q}$. The first step for each module is to perform the Forward Kinematic calculations to determine the current spatial position and velocity for each contact point in the

module. For modules connected in series, the Forward Kinematics problem does not need to "wait" for the kinematics of the tip body connected to its base. Since the base motion variables are included in a serially connected MRDS module, and since the current state of the base is available from the integration of $\ddot{q}$, Forward Kinematics can proceed immediately and independently of all other modules in the system. This is in keeping with the modularity of the algorithm and suitability for parallel processing.

Once completed, this process will give the terms ${}^{I}r_{IO_i}$, ${}^{I}\dot{r}_{IO_i}$, ${}^{I}R_{O_i}$, and ${}^{I}\omega_{IO_i}$ for each contact point of each module, where, as before, "O_i" is used to represent either an "E_i" or "P_i" operational point. The next task for each module is the computation of the open chain generalized coordinate accelerations, $\ddot{q}^{open}$, which are found using any suitable open chain algorithm. From these accelerations, the open chain spatial accelerations of each contact point, ${}^{I}\ddot{x}_{IO_i}^{open}$, can be found (again from Forward Kinematics). The final step at the modular level is the computation of ${}^{I}\Psi_{IO_i}$ for each contact point, and ${}^{I}\Lambda_{O_iO_j}^{-1}$ for each pair of points[21].

When the modular calculations are complete, all necessary terms are passed to the global level of the algorithm. This includes all of the operational space inertia matrices and all of the contact point kinematic terms: ${}^{I}r_{IO_i}$, ${}^{I}R_{O_i}$, ${}^{I}\dot{x}_{IO_i}$, and ${}^{I}\ddot{x}_{IO_i}^{open}$. For each contact, the relative position and velocity terms can then be found from the simple kinematic relations given by Equations 52 through 54. Specifically, these solutions are as follows:

$$ {}^{E_i}r_{E_iP_i} = {}^{E_i}R_I \cdot ({}^{I}r_{IP_i} - {}^{I}r_{IE_i}) \tag{97} $$

21. Note that relative forms of these terms are found for Type B contacts.

$$ {}^{E_i}\dot{x}_{E_iP_i} = \begin{bmatrix} {}^{E_i}\dot{r}_{E_iP_i} \\ {}^{E_i}\omega_{E_iP_i} \end{bmatrix} = {}^{E_i}\Re_I \cdot ({}^{I}\dot{x}_{IP_i} - {}^{I}S_{E_iP_i} \cdot {}^{I}\dot{x}_{IE_i}) \tag{98} $$

If desired, the coordinate frame transformation can be found from

$$ {}^{E_i}R_{P_i} = {}^{E_i}R_I \cdot {}^{I}R_{P_i} = ({}^{I}R_{E_i})^T \cdot {}^{I}R_{P_i} \tag{99} $$

With these terms calculated, ${}^{I}\zeta_{E_iE_iP_i}$ can be computed (see Equation 81), and together with the open chain absolute accelerations, the open chain relative acceleration can also be obtained (see Equation 83).

The above derivation should make it clear that parallel and series connections are essentially no different in the MRDS algorithm. The inclusion of the base motion variables for all serially connected robots makes the Forward Kinematics problem self-contained (i.e. modular). With appropriate constraint violation suppression[22], there is no need for the Forward Kinematics problem to proceed in an outward recursion across a serial connection between modules. With reference to Figure 1, the kinematics of the first robot's tip body and the second robot's base body are each determined independently in their own respective, modular kinematic calculations at the beginning of each simulation cycle.

Note also that the contact model vector spaces are not needed in this process. The constraints expressed by the contact model have already been enforced in the solution of the contact forces, and the independently determined kinematics of each module will not violate them, assuming suitable constraint violation suppression is used when necessary. In the

22. See Section 6.4 for a definition and discussion of constraint violations and suppression techniques.

next section, an option for eliminating constraint violations from a serial connection is discussed.

6.3 Elimination of Constraint Violations for Series Connections

In the option discussed here to eliminate constraint violations from serial connections, the Forward Kinematics problem does extend across serial contacts, wherein the second robot's base body kinematics will be, in part, directly dependent on the kinematics of the supporting external body. In relation to the MRDS algorithm, the use of this option results in an additional step: "Step 5: Global Kinematics". In this new step, the position and velocity of a "tip body" are used in conjunction with the contact model equations to set the state of the serially-connected base body in the constrained directions of the contact. Details of this process are given here.

At the end of Step 4 of the MRDS algorithm, the joint accelerations of each module have been found and integrated to the next state. As in the previous section, the Forward Kinematics problem proceeds for all modules to produce the spatial position and velocity of each operational point in the system. This is achieved through purely modular calculations as described above.

The next step is to determine the contact model velocity variables, $\dot{x}_i^u$, associated with the motion space. They can be obtained from the following equation[23]:

$$\phi_i^u \cdot \dot{x}_i^u = {}^{E_i}\dot{x}_{E_iP_i} - \phi_i^c \cdot \dot{x}_i^c \tag{100}$$

23. Solution for $\dot{x}_i^u$ would also be necessary for any contact involving a time-variant motion space.

Solution for $\dot{x}_i^u$ is possible as ϕ_i^u is always of full column rank, $6 - n_{c_i}$, ϕ_i^c and $\dot{x}_i^c$ are both known, and ${}^{E_i}\dot{x}_{E_iP_i}$ is available from the preceding Forward Kinematics calculations together with Equation 98.

One method of solving Equation 100 is to pre-multiply that equation by $(\psi_i^u)^T$. From the dual basis relationship given in Equation 25, this will isolate $\dot{x}_i^u$ and will also eliminate velocity components in the constrained directions, $\dot{x}_i^c$, as follows:

$$\dot{x}_i^u = (\psi_i^u)^T \cdot {}^{E_i}\dot{x}_{E_iP_i} \tag{101}$$

As described in Section 3.3, the term $\dot{x}_i^c$ is either zero for locked modes or a known function of time for kinematically excited modes. Therefore, using the known quantities of $\dot{x}_i^c$ and the values of $\dot{x}_i^u$ from Equation 100 or 101, the relative spatial velocity can be reconstructed without the possibility of constraint violations, using the standard contact model expression as follows:

$${}^{E_i}\dot{x}_{E_iP_i} = \phi_i^u \cdot \dot{x}_i^u + \phi_i^c \cdot \dot{x}_i^c \tag{102}$$

The relative spatial velocity computed here is different than that in Equation 98. In this case, velocity components in the constrained directions come directly from known quantities that define the kinematic constraint. In Equation 98, the components in the constrained directions may include roundoff and integration inaccuracies, leading to a violation of the constraint.

Using the new relative spatial velocity from Equation 102 that eliminates the possibility of constraint violations, the absolute spatial velocity of the base body of the serially con-

nected robot can then be found from the following[24]:

$$ {}^{I}\dot{x}_{IP_i} = {}^{I}S_{E_iP_i} \cdot {}^{I}\dot{x}_{IE_i} + {}^{I}\Re_{E_i} \cdot {}^{E_i}\dot{x}_{E_iP_i} \tag{103} $$

This procedure for setting the base body's spatial velocity would also be followed to establish the base body position and orientation. To achieve this, expressions are needed for the relative position and orientation constraints between the two bodies that are compatible with the velocity constraints. Formal expressions are not given here, as this process will be unique to the variables and mapping chosen in the representation of the relative orientation of the bodies for a potentially non-holonomic constraint. Further discussion of position constraints is given in Section 6.5.

In essence, the process described in this section effectively keeps the relative position and velocity in the motion space of the constraint, but "overwrites" the base body position and velocity variables associated with the constrained directions of the contact in a way that eliminates any possibility of constraint violations. Noted above, the implementation of this method effectively adds a new step to the global level of the algorithm, since the process of mapping kinematic terms across the contact must be performed "above" the modular level.

As a reminder, it is important to recall that this method is by no means necessary. Following the procedure from the previous section in this chapter, the state of the base body can be found purely from the modular integration of the generalized acceleration vector, and the possible growth of constraint violations can be suppressed through a variety of

24. This equation assumes that point P belongs to the serially connected base body. If this is not the case, the given equation can easily be recast to solve for the inertial acceleration of frame E.

available techniques. Such techniques are discussed in the next section, along with treatment of singularities and numerical ill-conditioning.

6.4 Constraint Violations, Singularities, and Numerical Ill-conditioning

In a perfect computing system, the structure of the contact model will maintain the constrained relationship between the two contacting bodies throughout the simulation. However, on real computing systems, small accumulating errors will grow, leading to a potentially unstable or grossly inaccurate simulation. Along with constraint violations, another numerical issue in the Forward Dynamics problem relates to singularities and numerical ill-conditioning. In this section, methods for dealing with these numerical issues are discussed.

At the modular level of the MRDS algorithm, the solution of the open chain joint accelerations effectively requires the inversion of the joint space inertia matrix. To account for the possibility of this matrix being singular or ill-conditioned, a sophisticated inversion technique can be used, such as Singular Value Decomposition. This technique is quite computationally expensive as compared with an LU Decomposition, and so should be used only when necessary. In general, singular configurations are avoided during motion planning and may correspond to physically undesirable robot configurations.

At the global level, the solution of the linear system of equations for the unknown contact force components effectively requires that the matrix $\mathbf{B^c}$ is always invertible. It is possible, however, that when one or more of the constraint equations becomes linearly dependent on the others (or nearly so), matrix $\mathbf{B^c}$ will become singular or poorly conditioned. In this case, the solution for the components of the contact forces may be highly in-

accurate, which in turn leads to inaccuracy in the joint variables. In addition, constraint violations arising from the drift associated with roundoff error inherent with solution by a digital computer will also contribute to inaccuracies. Although these errors are small and the drift is gradual, they can become unstable after long periods of simulation time.

Anderson discusses these two issues in great detail in [2], where he reviews three well-known methods for controlling constraint violations. These methods include constraint stabilization strategies, penalty formulations, and stabilized penalty procedures. After addressing the strengths and weaknesses of each method, Anderson presents a new approach which successfully limits constraint violations, while at the same time avoids the need to invert a potentially singular contact force coefficient matrix. The use of this technique appears well suited for use with the MRDS algorithm, and its incorporation will be considered in future applications.

Currently, however, a simple proportional and derivative (PD) control approach is used to suppress constraint violations in the example systems presented in Chapter 8. With this technique, the spatial contact force vectors found in the global level of the algorithm are modified as follows:

$$ {}^{E_iP_i}F^{*}_{E_iP_i} = {}^{E_iP_i}F_{E_iP_i} + K_v \cdot \Phi_v + K_p \cdot \Phi_p \tag{104} $$

where K_p and K_v are the proportional and derivative gains, respectively, and Φ_v and Φ_p are expression that give the amount of error in the enforcement of the velocity and position constraints associated with the contact. With this method in place, ${}^{E_iP_i}F^{*}_{E_iP_i}$ replaces ${}^{E_iP_i}F_{E_iP_i}$ in all subsequent calculations leading the modular level computation of the constrained

joint accelerations.

The error or constraint violation in the velocity terms can be given by the difference between the known value of $\dot{x}_i^c$ and the associated value determined from the relative spatial velocities computed in Equation 98. Specifically, this error can be expressed as:

$$\Phi_v = [(\psi_i^c)^T \cdot {}^{E_i}\Re_I \cdot ({}^I\dot{x}_{IP_i} - {}^I S_{E_iP_i} \cdot {}^I\dot{x}_{IE_i})] - \dot{x}_i^c \tag{105}$$

As before, an expression for any position constraints will be specific to the modelling method used to describe the relative kinematics between two bodies (potentially) subject to both holonomic and non-holonomic velocity constraints. The position error, however, would be found in a manner similar to that for the velocity. A brief discussion of position constraints is given in Section 6.5.

The use of this technique has the effect of adding a small amount of stiffness and damping to the contact. The purpose of these terms is to correct for deviations from the contact constraints that arise from numerical round-off error. Since only very small errors are encountered in a single timestep, the corrective force from the stiffness and damping terms is also very small.

For the example systems presented in Chapter 8, this technique was only required in a handful of cases. It was particularly beneficial in the simulation of closed loop systems undergoing relatively large deflections. Adequate values for the proportional and derivative gains were found to be K_p = 0.1 N/m and K_v = 0.2 N-s/m, respectively. Inclusion of the suppression technique in systems that did not exhibit a growth in constraint violations showed no numerically significant differences in the predicted results. This shows that the

corrective terms from the constraint violation suppression did not adversely affect simulation outcomes in the cases studied. Distortion of the simulation outputs are possible when the gains required to control violations are too large. This was not the case for any of the example systems simulated.

6.5 Position Constraints Compatible with the Contact Model

Position constraints between two contacting structures (in this case modules) are typically expressed as follows [38][46]:

$$\Phi_p(q_{m(E_i)}, q_{m(P_i)}) = 0 \tag{106}$$

For the general contacts of the MRDS algorithm which allow kinematically excited contact modes, this expression is more accurately given by [2][63]:

$$\Phi_p(q_{m(E_i)}, q_{m(P_i)}, t) = 0 \tag{107}$$

As noted in the previous sections, expression of the position constraints must be compatible with the contact model's holonomic velocity constraints (if any). Taking the derivative of Equation 107 with respect to time, we get the following:

$$\frac{\partial \Phi_p}{\partial q_{m(P_i)}} \cdot \dot{q}_{m(P_i)} + \frac{\partial \Phi_p}{\partial q_{m(E_i)}} \cdot \dot{q}_{m(E_i)} + \frac{d\Phi_p}{dt} = 0 \tag{108}$$

As a comparison, Equation 105 is rewritten with the spatial accelerations expressed using the appropriate Jacobian matrices as follows:

$$\Phi_v = [(\psi_i^c)^T \cdot {}^{E_i}\Re_I \cdot ({}^I J_{IP_i} \cdot \dot{q}_{m(P_i)} - {}^I S_{E_i P_i} \cdot {}^I J_{IE_i} \cdot \dot{q}_{m(E_i)})] - \dot{x}_i^c \tag{109}$$

Recalling that $\dot{x}_i^c$ is a known function of time, the terms in Equations 108 and 109 can be easily correlated as follows:

$$\frac{\partial \Phi_p}{\partial q_{m(P_i)}} = (\psi_i^c)^T \cdot {}^{E_i}\Re_I \cdot {}^{I}J_{IP_i} \tag{110}$$

$$\frac{\partial \Phi_p}{\partial q_{m(E_i)}} = -(\psi_i^c)^T \cdot {}^{E_i}\Re_I \cdot {}^{I}S_{E_iP_i} \cdot {}^{I}J_{IE_i} \tag{111}$$

$$\frac{d\Phi_p}{dt} = -\dot{x}_i^c \tag{112}$$

This exercise may shed some light on the nature of the position constraints that are compatible with any holonomic velocity constraints established by the general contact model. It should be pointed out, however, that when non-holonomic velocity constraints are also present, there will be additional velocity constraints aside from those produced by differentiating the position constraints. Consequently, the constraint space, ψ_i^c, used in the equations of this section is different from the constraint space defined and used throughout the rest of this dissertation. Here, these vector space describes only the directions constrained by holonomic constraints.

A classic example of a contact with both holonomic and non-holonomic constraints is the rolling of a sphere on a flat surface with no slipping. In Cartesian coordinates, five variables are needed to describe the position of the sphere relative to a frame fixed in the flat surface. These include three parameters to define the rotational position of the sphere and two translational coordinates. The third translational coordinate is not needed due to the single position constraint, which dictates that the center of the sphere will always be a fixed

distance from the surface (equal to the sphere's radius).

Differentiating this single position constraint produces a valid holonomic velocity constraint. It dictates that the relative motion in the direction normal to the surface will always be zero. For rolling with no slipping, however, there are two additional velocity constraints that dictate zero relative velocity at the contact point in the two in-plane directions of the flat surface. With these two non-holonomic constraints added, there is a total of three velocity constraints. Despite the overall non-holonomic nature of this example contact, a position constraint compatible with the holonomic velocity constraint can still be found and used in the constraint violation suppression technique from the previous section.

6.6 Overlap of Contact Motion Space and Base Motion Variables: An Example

The serial connection of a module offers one additional subtlety not yet described in full. This issue deals with the potential overlap between the motion space of a serial contact and the motion described by the inclusion of full base motion variables. This overlap is described by way of an example system.

Although it was stated above that the MRDS algorithm is best applied when the modules are complex, multi-degree-of-freedom structures, the example here is for the serial connection of two modules, where each module represents a single rigid link. Despite the lack of efficiency in applying the MRDS algorithm to such a trivial system, this example's simplicity helps to demonstrate the concept at hand.

For this system, consider that the first module is a single, rotating link with a fixed base. The second module is a single link connected in series to the first by a revolute contact model. As this second module is serially connected, the generalized coordinates must contain

full base motion. For this planar system, the generalized coordinate vector is as follows:

$$q = \begin{bmatrix} x \\ y \\ \theta \end{bmatrix} \tag{113}$$

Already, the overlap between the contact model and the base motion variables is evident. The contact model contains one degree of freedom: the rotation around the joint axis. This same degree of freedom exists in the base motion variables of the second module[25].

Applying the MRDS algorithm, the first step is the modular decomposition and contact modelling. We have already defined the bodies belonging to each module and chosen a revolute contact model as the connection between them. For the labeling of the operational points, let us assign the “E” operational point (and frame) to the tip of the single link in the first module and the “P” operational point (and frame) to the “base reference end” of the link in the second module.

The modular calculations performed for Step 2 will produce the open chain accelerations for each module. This includes the joint accelerations and the spatial accelerations of the contact points. For the first module, these accelerations stem from the joint torque applied at the base of this first link. Recall that no tip force is present in these open chain calculations.

For the second module, things are handled a bit differently. Here, the torque acting on the link of the second module can and should be included in the contact model description

25. As this module consists of only one rigid body, the entire generalized coordinate vector consists only of the base motion variables. In a more general case, q would have other additional coordinates.

of the module-to-module contact force. In the equations for the contact model, this applied torque is included as the single value in the term F^u, and it is a known function of time.

With this torque included in the contact model, the open chain calculations for this example system are left with no input forces (aside from gravity, if applicable). In the open chain state, the contact forces are not included, and here the joint torque is considered a contact force since the contact model represents the revolute joint.

Also calculated in Step 2 at the modular level are the contact force coefficient matrices for each module, Ψ and Λ^{-1}. For the second module, the "P" coordinate frame and the base body reference frame used to define the generalized coordinates are one and the same. Consequently, the Jacobian matrix for the P operational point is simply a three by three identity matrix. The current state kinematics of the P contact point are trivially calculated as they are equal to the generalized coordinate and rate vectors, q and $\dot{q}$. The kinematics of the contact point and the operational space inertia matrix are passed to the global level of the algorithm, as are the corresponding quantities from the first module.

In the construction of the linear system of equations for the components of the unknown contact force, F^c, the joint torque associated with the contact model is included in the term F^u, as described above. Solution of the linear system of equations formed at the global level provides the two constraint force components acting in the x and y directions of the contact.

These forces are passed back to each module for the final step of the algorithm. At this time, the open chain joint accelerations are corrected using the contact force and the cross space inertia matrix. The newly-found closed-chain joint accelerations are then integrated to find the next state for each module. For the second module, this integration includes the

base motion variables (which, as noted earlier, are the only variables in this particular module).

Unlike the earlier calculations in Step 2, where the joint torque was accounted for by the contact model, here, the motion permitted in the motion space of the contact is actually specified from the module's generalized coordinates. The motion space variables of the contact model, $\dot{x}^u$, and their corresponding position variables are not explicitly determined or involved. Again, this demonstrates the overlap.

Effectively, the contact model in this example is used to solve for the unknown contact forces and to implicitly establish the allowable relative motions. The actual movement in this motion space, however, comes from the module's internal calculations and not explicitly from the contact model velocity equations.

This lack of explicit use of the motion space variables, $\dot{x}^u$, is also true for parallel contacts. For parallel contacts, this may be expected. For serial contacts, it is perhaps less intuitive. Traditionally, with a parallel connection of two modules, one would expect the degrees of freedom in the contact motion space to be dependent coordinates in the combined system, while a subset of the generalized coordinates in each of these modules will be the independent coordinates. For serial connections, however, one would likely expect the degrees of freedom of the contact motion space to be independent configuration variables along with the modular generalized coordinates. In the MRDS algorithm, this is not the case.

Instead, the inclusion of full base motion variables for a serially connected module allows the independent configuration variables to be kept within the module rather than in

the contact model. The dependent variables are those in the motion space of the contact model and those in the base motion variables associated with the constrained directions of the contact. This explains why the motion space variables, $\dot{x}^u$, are not explicitly obtained in order to establish the state of the system[26].

In the simple example discussed above, the contact frame (frame P) and the base motion frame were one and the same. For particularly complex serial connections, this may not be possible or practical. In this situation, it may be beneficial to treat the base motion variables, $\dot{x}^u$, as independent configuration variables for the system. This, however, would make it necessary to determine the next state values of $\dot{x}^u$ through the integration of the current state acceleration variables, $\ddot{x}^u$. These acceleration components could be found at the global level from the following equation (formed from Equations 82, 85, and 86):

$$\phi_i^u \cdot \ddot{x}_i^u + \phi_i^c \cdot \ddot{x}_i^c + \dot{\phi}_i^u \cdot \dot{x}_i^u + \dot{\phi}_i^c \cdot \dot{x}_i^c = {}^{E_i}\dot{x}_{E_iP_i}^{open} + \sum_{j=1}^{nc} [B_{ij} \cdot (\psi_j^u \cdot F_j^u + \psi_j^c \cdot F_j^c)] \qquad (114)$$

Once the contact forces have been determined from the linear system of equations, Equation 114 can be solved for $\ddot{x}^u$, as it is the only remaining unknown term, and ϕ_i^u always has full column rank. Using the dual basis relationships from Equation 25, $\ddot{x}^u$ could also be found by pre-multiplying Equation 114 by $(\psi^u)^T$. Integration of $\ddot{x}^u$ would lead to the next state values for relative angular velocity and position (non-holonomic constraints would require proper mapping to produce a full set of position coordinates).

26. These variables are needed explicitly, however, in the case of time-variant modes (see Equation 90).

6.7 Summary

The sections of this chapter are provided to offer more insight into the application of the MRDS algorithm, particularly for the treatment of serially connected modules. As shown, these serial connections can be treated in exactly the same manner as parallel connections. The primary difference is the additional modelling options afforded to serial connections with regards to constraint violations and Forward Kinematics.

Having presented the full details of the MRDS application to rigid structures, attention is now turned to the treatment of flexible systems. Applicability of the algorithm to these types of systems is shown in the next chapter, along with a discussion of related linearization issues. In Chapter 8, many of the issues discussed in this chapter are incorporated in the simulation of a number of example systems, each of which is composed of serial or parallel connections of structurally flexible robot modules.

Chapter 7 SYSTEMS WITH STRUCTURAL FLEXIBILITY

Sophisticated robot systems like those used in space applications or a hazardous waste storage tank can seldom be adequately modelled with rigid-body members alone. Space robots, like the Shuttle Remote Manipulator System and Space Station Remote Manipulator System of Chapter 1 are designed to be long reach, slender, and lightweight, and consequently, are quite compliant. The MRDS algorithm of Chapter 5 has, to this point, not been shown to include structurally flexible robots. In this chapter, this added complexity is examined and incorporated into the algorithm.

Fundamentally, there is no difference at all in the application of the algorithm when dealing with flexible robots. The use of joint space dynamics, operational space dynamics, and contact modelling all still apply. However, as is the case with any method for dynamic simulation, the treatment of flexible structures involves numerous additional difficulties not present in the simulation of only rigid bodies. A sample of such difficulties includes the following:

- the choice in modelling method (e.g. finite element, assumed modes, or lumped parameter methods)
- selection and enforcement of boundary conditions
- possible inclusion of second order strain or kinematic relationships that lead to first order dynamic effects (e.g. “spin stiffening” effects)
- the heightened difficulties of stability and accuracy, often requiring more sophisticated numerical procedures (integration scheme, matrix inversion, etc.)

Furthermore, the difficulties listed above are made more so for constrained flexible structures due to the change in boundary conditions and the enforcement of closed-loop kinematic constraints.

While all the issues listed above are important, they are not unique to the use of the MRDS algorithm. Any chosen method of dynamic simulation would require careful attention to these points. As such, this work will not address each and every possible difficulty that may be encountered. Instead, the purpose of this chapter is to demonstrate the general applicability of the MRDS algorithm to the treatment of structurally flexible systems.

7.1 Applicability of the MRDS Algorithm

The ability to apply the MRDS algorithm to structurally flexible systems is shown in two ways. In this chapter, the governing dynamic equations of such a system are shown to be of the same basic form as those encountered in the derivations of Chapters 4 and 5. Certain limitations and special considerations are also discussed. In Chapter 8, applicability is demonstrated through the results from numerous simulations of various structurally flexible systems.

7.1.1 Joint Space Dynamics

As always, the MRDS equations begin with the expression for the joint space dynamics of the module, which in this case, is a structurally flexible robot or device. Using finite element, assumed modes, or lumped parameter methods, these various derivation methods typically lead to the following form of the joint space dynamic equations of motion [36][38][41][45][51][63][27]:

$$\tau = H \cdot \ddot{q} + C + G + K \cdot q + J^T \cdot F \tag{115}$$

where the term, K, which was not present in the rigid-body formulations of Chapters 4 and 5, is a stiffness matrix. It should be noted that some methods of modelling constrained flexible manipulators may not explicitly produce an equation of exactly this form. This is typical of recursive methods [3][32][33]. However, any method employed will result in a formulation that is equivalent in kinematic and dynamic content and can, therefore, be adapted or augmented to provide all terms necessary in the MRDS algorithm.

In general, the generalized coordinate vector, q, can be separated into an $N_\theta \times 1$ vector, q_θ, for the rigid coordinates, and an $N_f \times 1$ vector, q_f, of the flexible coordinates, where $N_\theta + N_f = N$. This same grouping can be applied throughout Equation 115 as follows:

$$\begin{bmatrix} \tau_\theta \\ \tau_f \end{bmatrix} = \begin{bmatrix} H_{\theta\theta} & H_{\theta f} \\ H_{f\theta} & H_{ff} \end{bmatrix} \cdot \begin{bmatrix} \ddot{q}_\theta \\ \ddot{q}_f \end{bmatrix} + \begin{bmatrix} C_\theta \\ C_f \end{bmatrix} + \begin{bmatrix} G_\theta \\ G_f \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ 0 & K_{ff} \end{bmatrix} \cdot \begin{bmatrix} q_\theta \\ q_f \end{bmatrix} + \begin{bmatrix} J_\theta^T \\ J_f^T \end{bmatrix} \cdot F \tag{116}$$

where K_{ff} is the $N_f \times N_f$ component of the stiffness matrix for the flexible coordinates. All other terms have the same fundamental function and definitions as in previous forms of the joint space dynamic equation (see Equation 1), but here, each term has been subscripted to show their relationship to the rigid and flexible coordinates of the system. It should be noted that all terms, with the exception of the external force vector, F, now have some component corresponding to the flexible degrees of freedom of the system.

Again, the joint space dynamic equation is recast in the following form:

---

27. Generic notation and a single contact force vector are used for simplicity. These equations could easily be extended to multiple contacts following the developments in Chapter 5.

$$\ddot{q} = H^{-1} \cdot (\tau - C - G - K \cdot q) - (H^{-1} \cdot J^{T}) \cdot F \tag{117}$$

$$= \ddot{q}^{open} - \Psi \cdot F \tag{118}$$

where:

$$\Psi = H^{-1} \cdot J^{T} = \begin{bmatrix} H_{\theta\theta} & H_{\theta f} \\ H_{f\theta} & H_{ff} \end{bmatrix}^{-1} \cdot \begin{bmatrix} J_{\theta}^{T} \\ J_{f}^{T} \end{bmatrix} \tag{119}$$

The expression of Equation 118 demonstrates that for the MRDS algorithm, there is no real difference in the usage of the joint space dynamic equations when applied to flexible robots. The open chain joint accelerations are calculated according to any suitable algorithm. These values are then corrected by the contribution from the external force, F. The only difference, at least in terms of the MRDS governing equations, is the presence of the additional modelling to account for the compliant nature of the module.

7.1.2 Operational Space Dynamics

The operational space dynamic formulation can also be expressed in the same general form as seen in the previous chapters. This dynamic equation can be derived in the same manner, starting with the kinematic expression for the spatial acceleration of the contact point. The sequence of equations is as follows:

$$\dot{x} = J \cdot \dot{q} = \begin{bmatrix} J_{\theta} & J_{f} \end{bmatrix} \cdot \begin{bmatrix} \dot{q}_{\theta} \\ \dot{q}_{f} \end{bmatrix} \tag{120}$$

$$\ddot{x} = J \cdot \ddot{q} + \dot{J} \cdot \dot{q} \tag{121}$$

Inserting the solution for the joint accelerations, $\ddot{q}$, from Equation 118, we obtain the now familiar form of the operational space dynamics as follows[28]:

$$\ddot{x} = \ddot{x}^{open} - \Lambda^{-1} \cdot F \tag{122}$$

where:

$$\Lambda^{-1} = J \cdot H^{-1} \cdot J^T = \begin{bmatrix} J_\theta & J_f \end{bmatrix} \cdot \begin{bmatrix} H_{\theta\theta} & H_{\theta f} \\ H_{\theta f}^T & H_{ff} \end{bmatrix}^{-1} \cdot \begin{bmatrix} J_\theta^T \\ J_f^T \end{bmatrix} \tag{123}$$

As above, the additional modelling necessary for structural flexibility does not prevent expression of the operational space dynamics in terms of an open chain acceleration and correction for the contribution of the external contact force. Again, the notable difference is the inclusion of flexible components in all key terms.

7.1.3 Kinematics of the Contact Bodies

The final aspect of the MRDS algorithm is the contact model, which is incorporated in conjunction with the operational space dynamics of each module, and the expression of the relative spatial acceleration between the two contacting bodies. Deferring to the derivation in Chapter 4, ultimately the kinematic expression required is for the open chain relative acceleration between the two contacting bodies, which is again given by:

$${}^{E}\dot{x}_{EP}^{open} = {}^{E}\Re_{I} \cdot ({}^{I}\dot{x}_{IP}^{open} - {}^{I}S_{EP} \cdot {}^{I}\dot{x}_{IE}^{open} - {}^{I}\zeta_{EEP}) \tag{124}$$

---

28. Again the derivation here is given for single contact systems, but can be extended to multiple contact systems by the same means given in Chapter 5.

Absolute open chain accelerations, ${}^{I}\ddot{x}_{IE}^{open}$, and ${}^{I}\ddot{x}_{IP}^{open}$, as before, are computed at the modular level. Calculation of these terms will naturally require consideration of the flexible motions of the structure. Modelled deflections and deflection rates must be used in the calculation of these terms, however, this typically poses little greater computational challenge than the equivalent rigid-body kinematic terms.

7.1.4 Limitations and Future Enhancements

As demonstrated in the preceding discussion, the steps of the MRDS algorithm are unchanged when applied to flexible structures, provided the additional modelling is included in all of the traditional terms. Two aspects of this application require special consideration and additional future research. These issues relate to the nature of the contact model and the computational complexity of the algorithm.

7.1.4.1 The Contact Model

As noted in its first presentation in Chapter 3, the contact model is used to describe the kinematic and dynamic relationships for the contact between two rigid bodies. Although the modular algorithm can be applied to structures containing flexible members, module-to-module and module-to-boundary contacts must currently be between two rigid-body components of the structures involved. Until revised contact modelling relationships are developed that allow general contact to a flexible member, this limitation should not be overlooked.

One very useful exception to this limitation is described here. If a contact model is to be used to represent contact to a flexible body, the coordinate frame for that body (“E” or “P”, as the case may be) must be defined at the contact point or surface. By way of expla-

nation, consider that the contact model is used to express the free and constrained directions of the relative motion and contact force. The motion of a frame interior to the body is different than a frame at the contact surface. For a deformable body, the relationship between them cannot, of course, be described using the rigid-body translation matrix, S. The directions of the motion space of the contact cannot be related purely by a kinematic expression when expressed in a frame internal to the flexible body. The deformation of the body caused by the contact force must be considered. These problems are avoided if the "E" and "P" coordinate frames are attached at the contacting surfaces when the bodies in contact are not rigid. This exception is used in the example simulations in the next chapter to allow application of the general joint model between flexible links.

7.1.4.2 Computational Complexity

The second aspect of the MRDS algorithm in need of further enhancement is the calculation of the contact force coefficient matrices, Ψ and Λ^{-1}. O(N) methods for the calculation of these matrices are given in [39] and [11] for Type A contacts. O(N) expressions for Type B were identified in Equations 91 through 94. These methods, however, have only been specified for rigid-body manipulators. Consequently, until these O(N) procedures can be extended to the treatment of structurally flexible manipulators, the direct multiplication methods of Equations 119 and 123 must be used.

7.2 Non-linear Effects and Linearization Methods

As outlined at the beginning of this chapter, the presence of structural flexibility will increase the complexity in achieving an accurate and stable dynamic simulation, especially

under constrained motion. One of those listed difficulties that deserves special attention deals with non-linear strain or kinematic effects. This subject has been well covered in the literature for unconstrained motion [15][28][51][55][62], but has received little attention for constrained systems [10][63]. As the MRDS is specifically intended for systems with multiple concurrent constraints, this issue is discussed in detail here. Moreover, the linearization issue may be, in fact, of greater impact in the MRDS algorithm, due to the combination of the joint space and operational space dynamics formulations.

7.2.1 Review of Existing Linearization Schemes

In the typical derivation of the dynamic equations of motion for a structurally flexible manipulator, terms that are second order and higher in the flexible coordinates (q_f, $\dot{q}_f$, and $\ddot{q}_f$) are neglected as part of the linearization process for systems modelled using the assumption of small deflections. However, as shown by numerous researchers [3][25][36][51][63], terms that are of first order importance in the dynamic equations may stem from kinematic or strain relations that are of second order. Premature linearization of these underlying relationships will result in equations of motion that are inaccurate for certain operations.

The linearization issue was most notably brought to light by Kane et al. [28], where the dynamics of a cantilevered beam attached to a moving base are discussed at length. This work demonstrated that traditional methods, including many commercially available modelling packages, implement these prematurely linearized equations, resulting in inaccurate dynamic behavior during high speed rigid-body motions of the beam's mobile base. Such methods are said to lack the "spin stiffening" effects.

Padilla and von Flotow [55], explore the suitability of various linearization schemes[29] for both one and two-link planar flexible robots. In their work, non-linear strain relations are used in the derivation of equations of motion to ensure the inclusion of these stiffening effects. The authors then identify three different linearization schemes with respect to the order of terms involving the flexible generalized coordinates and rates, q_f and $\dot{q}_f$. These schemes are named the "consistent", "inconsistent", and "ruthless" models based on their approach to linearization. This terminology has been used by other researchers [15][36][62] and will be adopted here as well.

To facilitate the discussion of these linearization schemes, we will use the joint space dynamic equations of a single rotating unconstrained Bernoulli-Euler[30] beam. For such a system, the term $K_{ff} \cdot q_f$ from Equation 116 can be expressed in the following form:

$$K_{ff} \cdot q_f = [K_E + \dot{\theta}^2 \cdot (K_G - H_{ff})] \cdot q_f \tag{125}$$

where $\dot{q}_\theta = \dot{\theta}$ is the scalar angular velocity describing the rigid-body rotation rate at the base of the flexible link; q_f is the vector of flexible coordinates; K_E is the component of the stiffness matrix related to the beam's bending inertia; K_G is often referred to as the geometric stiffness; and H_{ff} is a component of the joint space inertia matrix (see Equation 116).

Derivation of the dynamic equations using the consistent model, as defined by Padilla

---

29. The term "linearization" is widely used in existing literature and is used here, as well. It should be noted, however, that some of these "linearization schemes" are achieved simply by neglecting troublesome or computationally expensive terms. A strict mathematical linearization process is not always implied by the term.

30. Links are assumed to have a small cross section compared to their length, and the effects of shear and rotary inertia are not considered.

and von Flotow [55], requires the use of second order strain relationships at the outset to assure that all first order terms in the flexible coordinates and their rates are present in the final equations. The inconsistently linearized model is akin to the aforementioned “traditional” methods, in which the entire derivation is formed using only linear strain relationships. With the inconsistent model, the geometric stiffness matrix, K_G, is lost, as this term derives from the higher order strain relationships used in the consistent method. This prematurely linearized model produces an illogical and incorrect formulation that predicts a softening of the beam stiffness for higher rotational speeds. Consequently, the results in [55] show the failure of the inconsistent model under high-speed rotation, supporting the observation of Kane et al. [28]. The results of the consistent model, however, correctly predict a stiffening of the beam for high-speed operations.

The ruthless model is one in which all terms that are non-linear in the flexible coordinates are eliminated. This includes terms in the dynamic equations that are first order in the flexible coordinates but contain the product of a flexible and rigid body coordinate. In this case, both K_G and H_{ff} are eliminated from the model, as they are part of a term containing the non-linear product, $\dot{\theta}^2 \cdot q_f$. This model predicts neither stiffening nor softening as the beam rotates at a greater speed.

It is the assertion of Padilla and von Flotow that for motions where the rigid-body rates are lower than ten percent of the lowest modal frequency of the system, the ruthless model is of acceptable accuracy and lower computational cost than the consistent model[31]. The inconsistent model is never recommended by Padilla and von Flotow under any circum-

31. Similar limits are said to exist for the magnitude of translational accelerations when using the ruthless model for a beam on a fully mobile base, though no exact threshold is given.

stances.

For a two-link planar robot, first order terms that derive from the non-linear strain relations are present in the joint space inertia matrix and Coriolis terms, as well as the stiffness matrix. Consequently, with the consistent model, terms such as $H_{\theta\theta}$ and $H_{\theta f}$ in Equation 116 can contain terms of first order in the flexible coordinates. The terms $H_{f\theta}$ and H_{ff}, however, cannot, as they form a product with the flexible acceleration coordinates, $\ddot{q}_f$. In the ruthless model, first order terms in the flexible coordinates are removed from $H_{\theta\theta}$ and $H_{\theta f}$ due to the non-linear product with $\ddot{q}_\theta$. Consequently, with the ruthless model, the entire joint space inertia matrix of the two-link robot is solely a function of the rigid-body coordinates. Again, the "widespread use" of the ruthless model is recommended by Padilla and von Flotow for cases where the rigid body rotational rates and translational accelerations are acceptably small.

Support of the ruthless model is contradicted by Damaren and Sharf [15], where again different levels of linearization are compared. In their work, a three-dimensional manipulator similar to the Shuttle Remote Manipulator System (SRMS) is examined. Four different linearization schemes are used, including the consistent, inconsistent, and ruthless approaches. The fourth model deals with the linearization used when expressing rotational kinematics of a body in three dimensions. These results show that the ruthless model is, in fact, insufficient in the three-dimensional case even at rigid-body rates below the "ten percent" guideline. In [62], the analysis from [15] is extended to include material damping effects. In this instance, their findings suggest that the non-linear stiffening effects and the problems associated with them are much less significant with the presence of material

damping in the model.

7.2.2 Inclusion of Second Order Kinematic Effects via Foreshortening

The stiffening effects accounted for by the use of non-linear strain relations in [55] can also be incorporated through the use of second order kinematic relationships [10]. This is the approach taken in the application of the MRDS algorithm to the example systems discussed in the next chapter. For this reason, these second order kinematic expressions are discussed here.

Using first order kinematics to describe a beam in bending, the tip of a deformed link is modelled with only transverse deflection. The tip deflects by a small amount in a direction perpendicular to the undeformed axis of the beam. This simplistic relationship dictates that the distance from one end of the link to the other along the undeformed longitudinal axis always remains the same. Second order kinematic expressions account for the curling of the beam, which brings the tip slightly inward toward the base when deformed. In this case, the arc length of the deformed beam remains constant at all times, but its projection against the undeformed axis is slightly less than the beam's undeformed length, as shown in Figure 23. This second order kinematic phenomenon is widely referred to as *foreshortening*.

For finite element or assumed modes modelling of Bernoulli-Euler beams, the transverse deflection of a point on link "i", denoted by w_i, can be expressed as follows:

$$w_i(x, t) = \sum_{j=1}^{n_i} \sigma_{ij}(x) \cdot q_{f_{ij}}(t) \tag{126}$$

where the “x” coordinate is measured along the undeformed axis[32]; “t” is the time coordinate; σ_{ij} is the j^{th} shape function (finite element methods) or mode shape (for assumed modes methods) of the i^{th} link; $q_{f_{ij}}$ is the associated time-varying coordinate; and n_{f_i} is the number of shape functions or mode shapes used to model the flexibility of the i^{th} link. This displacement is shown in Figure 23, where L_i is the length of the undeformed link.

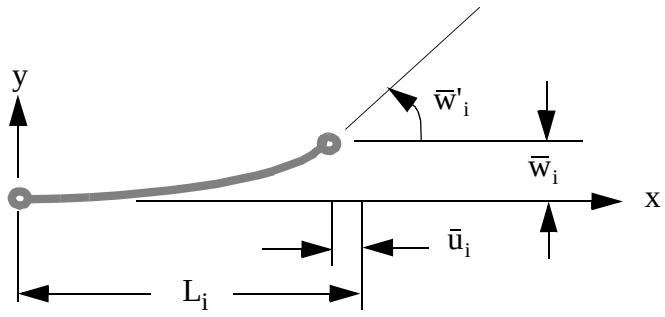

FIGURE 23. Foreshortening of a Link Associated with Transverse Deflection

The slope of the deformed link at any point along the undeformed axis is given by the following equation:

$$w'_i(x, t) = \sum_{j=1}^{n_i} \sigma'_{ij}(x) \cdot q_{f_{ij}}(t) \tag{127}$$

where the ' indicates a spatial derivative with respect to the x coordinate. Quantities evaluated at the tip of a link are indicated by the use of the “overbar”. Specifically, the tip deflection and slope are given by

32. An alternative method [10] that also incorporates the second order kinematic effects uses the “arc-length” measurement for the spatial coordinate, rather than the longitudinal coordinate.

$$\bar{w}_i = w_i(L_i, t) \text{ and } \bar{w}'_i = w'_i(L_i, t), \tag{128}$$

respectively.

Finally, the second order kinematic expression of foreshortening associated with a transverse deflection is denoted as u_i, which can be expressed as:

$$u_i(x, t) = -\frac{1}{2} \cdot \int_0^x (w'_i(x, t))^2 \cdot dx \tag{129}$$

Note that this term is always negative, as any point in the beam will always deflect closer to the base regardless of the direction of the transverse bending. Note also that this axial deflection is entirely a function of the transverse bending. It does not in any way relate to axial deformations from tension or compression of the beam.

Incorporation of the second order kinematic term given by Equation 129 is the method used in this work to account for the spin stiffening effects in the example systems used to validate the MRDS algorithm. However, unlike the works referenced above which dealt with open chain formulations, the MRDS algorithm is designed for constrained motion and makes use of both the joint space and operational space dynamic formulations. Choosing an appropriate linearization method becomes somewhat more unclear when combining these formulations. This is the topic of the next section.

7.3 Linearization in the MRDS Algorithm

As shown in Section 7.1, the joint space dynamic equation of a structurally flexible manipulator can be used in the MRDS algorithm without revision of the algorithm steps. In light of the discussion of linearization, however, we examine here the effect of these issues

in relation to constrained motion and to the use of the operational space dynamic formulation.

As described, implementation of the consistently linearized model will permit terms in the joint space dynamic equations that are first order in the flexible coordinates but can also be non-linear due to the possible combination with rigid-body coordinates or rates. As in the example above, $H_{\theta\theta}$ and $H_{\theta f}$ can include dependence on the flexible coordinates despite the product with the rigid-body accelerations, $\ddot{q}_\theta$. The point to be made here is that the linearization applies not to the matrix H by itself, but to the product $H \cdot \ddot{q}$.

The question that arises for constrained motion, then, is how to consistently linearize the term $J^T \cdot F$. Just as H is not to be linearized without regard for its product with $\ddot{q}$, so too should J^T not be linearized without regard for its product with F. However, the structure of F is not given as functions of the rigid and flexible coordinates and their rates. What level of linearization is then appropriate for J^T? Should first order terms be allowed throughout, in J_θ only (as they are in $H_{\theta\theta}$ and $H_{\theta f}$), or removed entirely?

Consider also that the Jacobian matrix serves a dual purpose, appearing in the joint space dynamics and implicitly in the operational space dynamics (see Equations 120 through 123). Moreover, if the direct multiplication method is used to determine Ψ and Λ^{-1}, then the Jacobian is required explicitly for both formalisms. Is the level of linearization appropriate for the Jacobian in the joint space dynamic equations necessarily appropriate for the operational space dynamics?

Inclusion of higher order terms when not needed for accuracy is not only a waste of computation time, but it can also make integration of the acceleration variables more diffi-

cult and more costly. As shown in the works discussed earlier for open chain systems [15][55][62], the best linearization scheme is usually dependent on the system (i.e. planar vs. three-dimensional, damped vs. undamped) and the particular operation (higher vs. lower speeds and accelerations). The same case-dependent nature is likely true for constrained flexible systems, regardless of the simulation algorithm used.

For the MRDS algorithm, the terms required from the modular level calculations are: the open chain joint accelerations for the module, $\ddot{q}^{open}$; the open chain spatial acceleration and cross point inertia matrix for each contact point, ${}^{I}\ddot{x}_{IO}^{open}$ and ${}^{I}\Psi_{IO}$, respectively; the inverse multi-point operational space inertia matrix, ${}^{I}\Lambda^{-1}_{O_iO_j}$, for every pair of operational points in the module, and, as discussed in Chapter 6, the kinematic terms for every contact point, ${}^{I}r_{IO_i}$, ${}^{I}R_{O_i}$, and ${}^{I}\dot{x}_{IO_i}$. For the example systems used in Chapter 8, solution for $\ddot{q}^{open}$ was always found using the consistently linearized approach. Using the direct multiplication methods, the linearization for the terms ${}^{I}\Psi_{IO}$ and ${}^{I}\Lambda^{-1}_{O_iO_j}$ relates directly to that chosen for the Jacobian matrix. For flexible systems, the kinematic terms for each contact point can also be of higher order or linearized to first order.

Given these observations, three choices were examined in relation to the linearization of the Jacobian (and hence, the force coefficient matrices) and the contact point kinematic terms. In one set of simulations, these terms were linearized to remove all second order terms, as well as the first order terms in J_θ stemming from second order kinematics. Simulations were then repeated with linearization to first order, but this time allowing the first order terms derived from second order kinematics to be retained in J_θ. In the last set of simulations, full second order terms were used in all kinematic expressions and throughout the

construction of the Jacobian.

The results, to be described in full detail in Chapter 8, indicate the need for retaining the second order terms in systems containing closed loops and subject to relatively large deflections and internal forces. Open loop systems and those closed loop systems with smaller deflections and internal forces did not require the second order kinematics.

While this study of the linearization schemes applicable to the structure of the MRDS algorithm is informative, a more comprehensive examination of the effect of linearization schemes on constrained flexible systems is needed. Following the lead set in [15], [55], and [62] for unconstrained systems, an excellent approach for studying this problem further would be to apply a set of differently linearized models to the constrained motion of both planar and three-dimensional structurally flexible robots both with and without material damping in the model. Simulation experiments of high and low speed maneuvers and higher and lower internal forces would potentially lead to guidelines about the required linearization scheme in conjunction with constrained motion. As this is beyond the scope of this work, it is left as a future exercise.

7.4 Summary

Regardless of the linearization issues, the discussion of Section 7.1 shows that the MRDS algorithm can be applied to structurally flexible systems without revision of the fundamental steps of the algorithm. Certainly, the analyst using the MRDS algorithm needs to be careful in many aspects of the simulation. This includes the choice of modelling method (finite element or assumed modes, for example), the establishment and enforcement of boundary conditions, the inclusion of higher order effects and the choice of a linearization

scheme, etc., just as he or she would need to do for any constrained motion simulation of flexible bodies.

Further evidence of the suitability of the MRDS algorithm for structurally flexible systems is given in the next chapter where simulation results for a number of multi-module, multi-contact systems with structural flexibility are presented.

Chapter 8 ALGORITHM CODING AND VALIDATION

To test and validate the MRDS algorithm, simulation code was developed for the dynamic simulation of several structurally flexible robot systems. These systems were created from modular building blocks and featured a variety of system topologies and operational conditions. This chapter presents the results of these system simulations and shows the effectiveness and applicability of the MRDS algorithm.

8.1 Modelling and Dynamic Equations of Example Systems

Three robot modules were used in the creation of the flexible robot systems considered in the testing of the MRDS algorithm. A general description of each of these three modules is as follows:

- A planar two-link robot with two revolute joints and a fixed base
- A planar one-link robot with one revolute joint and a fixed base
- A planar one-link robot with a freely mobile base

The dynamic equations of motion were derived using Extended Hamilton's Principle and are included in Appendix A. Modelling of link deflections was accomplished using the assumed modes method with cantilevered mode shapes used for all flexible links. The links used in the modules were long and slender and, hence, Bernoulli-Euler beam theory was applied. Consequently, the effects of shear and rotary inertia were ignored, and the magnitude of elastic deflections and rates were assumed to be small.

Transverse deflections of the links were given by Equation 126, and second order kinematics describing the shortening effects were used throughout the derivation of all kinemat-

ic and dynamic equations (see Equation 129). The consistent linearization scheme, an approach described in Section 7.2, was used for the open chain joint space dynamic equations.

Three different levels of linearization were explored for the computation of the contact point kinematics and the Jacobian matrix (and therefore, for the contact force coefficient matrices, as well). In the first case, second order kinematics and all first order terms derived from the second order terms were eliminated in the linearization process (similar to the "inconsistent" method in [55]). In the next method, first order terms derived from second order kinematics were allowed to remain in the Jacobian matrix, while all other kinematic terms were linearized to first order. This case is essentially an extension of the consistent model applied to the operational space dynamics. In the last case, all second order kinematic terms were allowed to remain in both the contact point kinematics and in the Jacobian, including subsequent first order terms. This case goes beyond all previously identified methods from [55] and Section 7.2, as those cases never contained any second order terms in the final equations.

The generalized coordinates for the two-link planar revolute robot, shown in Figure 24, were chosen as follows:

$$q = \begin{bmatrix} q_1 \\ q_2 \\ q_{(2+i)} \\ q_{(2+nf_1+i)} \end{bmatrix} = \begin{bmatrix} \theta_1 \\ \theta_2^r + \bar{w}'_1 \\ q_{f1} \\ q_{f2} \end{bmatrix} \tag{130}$$

where $\bar{w}'_1$ is the elastic slope at the tip of link 1 (see Equations 127 and 128), q_{fi} is the

$nf_i \times 1$ vector of modal coordinates for the bending of link i, nf_i is the number of mode shapes used for link i, and θ_2^r is the relative angle between the tip of the first link and the base of the second. Note that the absolute rotation angle for the second link, θ_2^a, which is used in the simulation output plots, is given by:

$$\theta_2^a = \theta_1 + \overline{w}'_1 + \theta_2^r = q_1 + q_2 \tag{131}$$

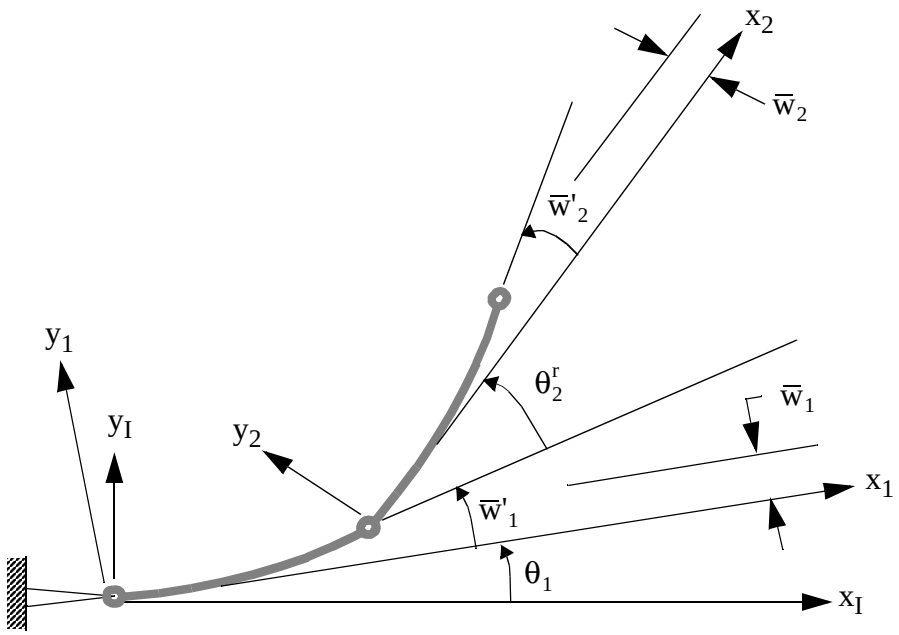

FIGURE 24. Coordinate Frames and Parameters for the Two-link Flexible Robot Module

The dynamic equations of the fixed-base one-link robot are the same as for the two-link robot with all terms pertaining to the second link removed. In this case, the generalized co-ordinates were chosen as follows:

$$q = \begin{bmatrix} q_1 \\ q_{(1+i)} \end{bmatrix} = \begin{bmatrix} \theta_1 \\ q_{f1} \end{bmatrix} \tag{132}$$

For the one-link robot with the mobile base shown in Figure 25, the generalized coordinates included the position of the proximal end of the link expressed in the inertial coordinate frame. The elements of q are as follows:

$$q = \begin{bmatrix} q_1 \\ q_2 \\ q_3 \\ q_{(3+i)} \end{bmatrix} = \begin{bmatrix} x_I \\ y_I \\ \theta_1 \\ q_{f1} \end{bmatrix} \tag{133}$$

Although for the sake of efficiency, the mobile, one-link robot would not typically be designated as its own module, as pointed out in Section 6.1, this approach was used here so that the ability to achieve series connections could be demonstrated fairly simply. As shown in the preceding equation, base motion variables have been included in the generalized coordinates to allow proper computation of the open chain dynamic behavior.

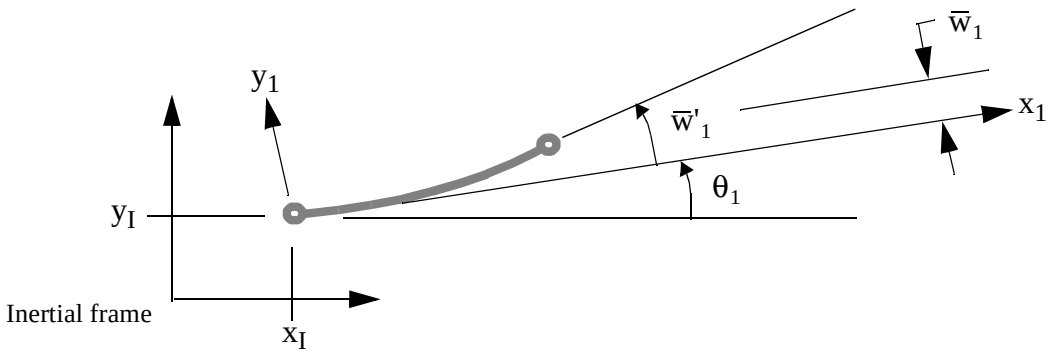

FIGURE 25. Coordinate Frames and Parameters for the Free-flying One-link Flexible Robot Module

8.2 Simulation Coding

The user interface of the simulation code is shown below in Figure 26. With this interface, all relevant inputs for construction and operation of the various robot systems considered can be set, including the following:

- the number of modules and number of contacts in the system
- length, mass per unit length, and bending properties (EI) of each link
- the number of mode shapes (0, 1, 2, or 3) used per flexible link
- the initial positions of each revolute joint and the base of any mobile base robots
- the contact point labels (assignment of “E” and “P”)
- the location of the contact on each module (the fixed base robots allow tip contact and the mobile base robots allow either base or tip contact)
- the torque coefficients for a sinusoidal torque command at all joints and contact points (for contact models with actuation)
- the simulation step size and total duration

With this code, connections between the robots or between the robot and fixed environmental boundary can be chosen from a list of six available contact models. These predefined contact models included rigid grasp, revolute joint, translation in the x-direction with no tip rotation, translation in the x-direction with free rotation, translation in the y-direction with no tip rotation, and translation in the y-direction with free rotation.

Also possible from the interface screen is the option to choose a predefined system con-

figuration, for which the modules and contacts have already been configured. This prevents the need to constantly assemble and connect the modules with every simulation, while still allowing new configurations to be created from scratch or by modifying one of these existing formations.

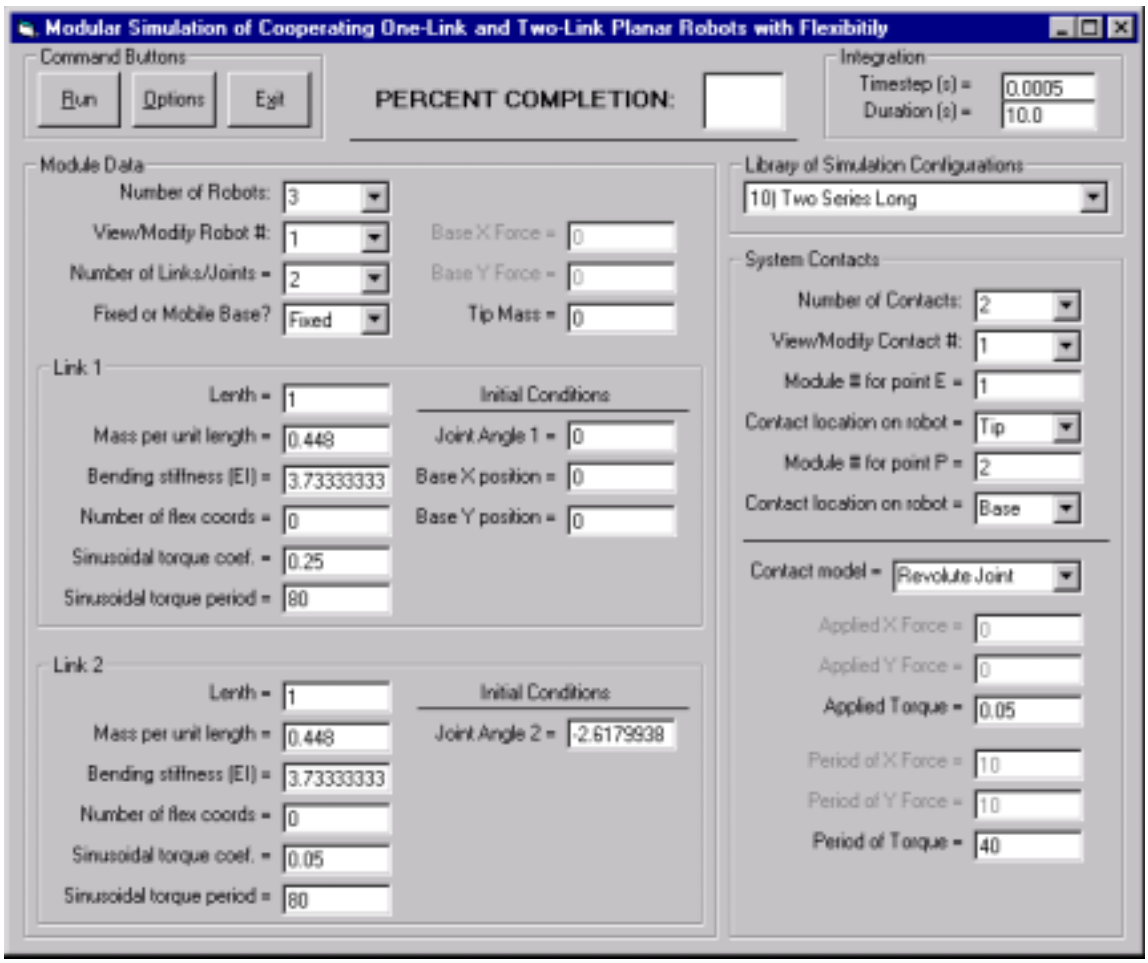

FIGURE 26. MRDS Interface for Systems of One and Two-Link Flexible Robots

Predefined systems included the simulation of each of the three module types listed at the beginning of Section 8.1 acting in a purely open chain manner. This allowed testing and validation of the fundamental dynamics of each module, prior to using them in cooperative

tasks. Other more simple cases included the use of a one-link fixed robot module joined in series by a revolute contact module to the mobile-link robot module. By comparing this with the open chain simulation of the two-link module, the MRDS's ability to simulate two modules in series was tested very easily.

A similar example came from two more predefined systems. In the first, the two-link robot module was constrained to slide in the x-direction at a fixed tip location in the y-direction with the tip free to rotate. Then, the same fundamental system was built using the fixed one-link robot in series with the mobile one-link robot and the same sliding contact model attached to the tip of the mobile robot. This comparison allowed the first testing of a multiple concurrent contact, since in this latter form, the second module had two contact models attached with one at the base and one at the tip.

Additional cases included the parallel cooperation of two, two-link robots joined at the tip, and a four-link robot which was formed by the series connection of the two-link module, the mobile one-link model, and another mobile one-link module.

Simulation coding employs a fourth order Runge-Kutta integration scheme with a fixed stepsize. Inversion of the joint space inertia matrices was achieved using the LU Decomposition algorithm from [57]. The same routine was also used for solving the linear system of equations in the unknown components of the contact force vectors. Singular Value Decomposition [57] was also used for matrix inversion and linear system solution as a means to study troublesome simulations.

8.3 Validation of the MRDS

Simulations generated from the coding of the MRDS algorithm were validated by com-

parison with simulation results from a commercially available product called Working Model® 2D (version 5.2) [72]. As shown in Figure 27, the approach used in Working Model to model link flexibility is to break the link into a user-specified number of rigid segments of equal length. These segments are connected by rotational springs, where the spring constant is established based on the bending stiffness (EI) entered by the user and the aforementioned number of segments. Resultant spring constants can then be further modified by the user if desired. A description of some of the difficulties encountered with Working Model is included in Appendix B.

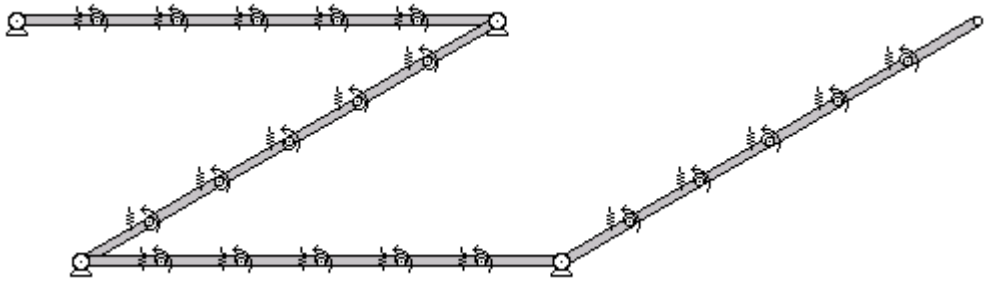

FIGURE 27. Model of a Four-link Flexible Robot Created with Working Model

Using the coded version of the MRDS algorithm described above, numerous dynamic simulations were performed and compared with similar simulations from Working Model. This section shows a collection of these simulated systems, the results, and describes the successes and limitations observed. In total, six basic systems are discussed, each showcasing a specific system topology. Several systems were examined under different operating conditions. For each system, some (usually all) of the links are modelled with flexibility. These systems are as follows:

1) One rotating link

2) Two-link robot with revolute joints

3) Kinematic base-excitation of a two-link robot

4) A Macro/Mini four-link robot

5) Four-link robot with all links of equal magnitude

6) Parallel cooperation of two, two-link robots

In all cases, compared outputs typically include the following: joint angles, inertial tip position in the x and y directions, transverse tip deflections of all flexible links, joint velocities, and constraint violations (for cases involving one or more contact models). As most readers can visualize angle measurements in degrees better than radians, plots of joint angles are given in degrees. By contrast, joint velocities are given in radians per second to facilitate comparison to system natural frequencies. This comparison is suggested by Padilla and von Flotow [55], as their results for one and two-link flexible robot simulations indicate certain difficulties in generating results for cases where the joint velocity is greater than ten percent of the system's lowest natural frequency. System natural frequencies are shown in Table 21 for Systems 1, 2a, and 2b, which are described in detail in the sections below.

TABLE 21. Natural Frequencies of Basic Example Systems

Example System	Link 1 (rad/s)		Link 2 (rad/s)	
System 1	$\omega_{11} = 10.2$	$\omega_{12} = 63.6$	N/A	N/A
System 2a with link 2 at 90 degrees	$\omega_{11} = 5.83$	$\omega_{12} = 49.6$	$\omega_{21} = 40.6$	$\omega_{22} = 254$
System 2b with link 2 at 90 degrees	$\omega_{11} = 3.68$	$\omega_{12} = 47.0$	$\omega_{21} = 12.8$	$\omega_{22} = 187$

The material properties of the links in all six systems were the same. Specifically, the links were modelled with rectangular cross sections and made of aluminum. Table 22 shows the cross-sectional dimensions, mass per unit length, and bending properties for all links. For all systems, link lengths were specified as one meter, one-half meter, or one-tenth meter, depending on the specific case as described below.

Torque commands for all joints in all systems were given the following sinusoidal format:

$$\tau_i = A_i \cdot \sin(2 \cdot \pi \cdot t / T_i) \tag{134}$$

where i is the joint number, t is the time variable, A_i is a user-specified amplitude, and T_i is a user-specified period. This format was selected for its simplicity in the development of simulations in Working Model. For the example systems of this chapter, specific values for amplitude and period were chosen for each operation so that all links experienced significant motion and generated adequate excitation of lower flexible modes.

TABLE 22. Material Properties Used for All Links in All Example Systems

Property	Value
Width	0.08 meters
Height	0.002 meters
Mass per unit length	0.448 kg/m
Young's Modulus (E)	70e9 N/m^2
Inertia (I)	5.3333e-11 m^4
Bending Stiffness (EI)	3.7333 N/m^2

In the following sections, key simulation results are featured. A full presentation of all

results for each system is given in Appendix C.

8.3.1 System 1: One Rotating Flexible Link

This simple case is included to demonstrate the ability and accuracy of the Working Model approach to modelling structural flexibility in a rotating link. Figure C-1 (see Appendix C) shows the comparison of MRDS and Working Model outputs for this maneuver. In this case, the link is one meter in length, and the amplitude and period for the torque command given by Equation 134 are shown in Table 23. These values drive the equivalent rigid link from rest to rest through an angle of 180 degrees. As shown in Figure C-1, the results for the joint angle are indistinguishable (the maximum error is less than 9.61e-2 degrees).

TABLE 23. Torque Parameters for System 1

	i = 1
A_i (Nm)	0.02948
T_i (s)	10

Figure 28 shows a comparison of the predicted tip deflection as determined with the MRDS code and with four separate cases of Working Model, each using a different number of segments per link to model the flexibility. The progression from four segments per link to ten segments per link shows improving accuracy[33] between Working Model and MRDS with regard to the magnitude and frequency of the higher frequency response. Although the

33. For all examples in this chapter, increasing agreement between Working Model and MRDS simulation results was obtained by increasing the number of segments per link in Working Model. Note, however, that convergence of the two methods as the number of segments and mode shapes approach infinity is not guaranteed. This is due to modelling differences, where Bernoulli-Euler beam theory was chosen for the MRDS coding, while Working Model's rigid segments allow large deformations and the inclusion of shear and rotary inertia effects.

case with eight segments per link shows a slightly lower overall deflection, the higher frequency is clearly better predicted than in the cases of four or six segments per link. Better agreement in the overall deflection could be obtained by retuning the spring constant. However, the spring constant used here is the value found to be “best” for a variety of loadings and operations (see Appendix B).

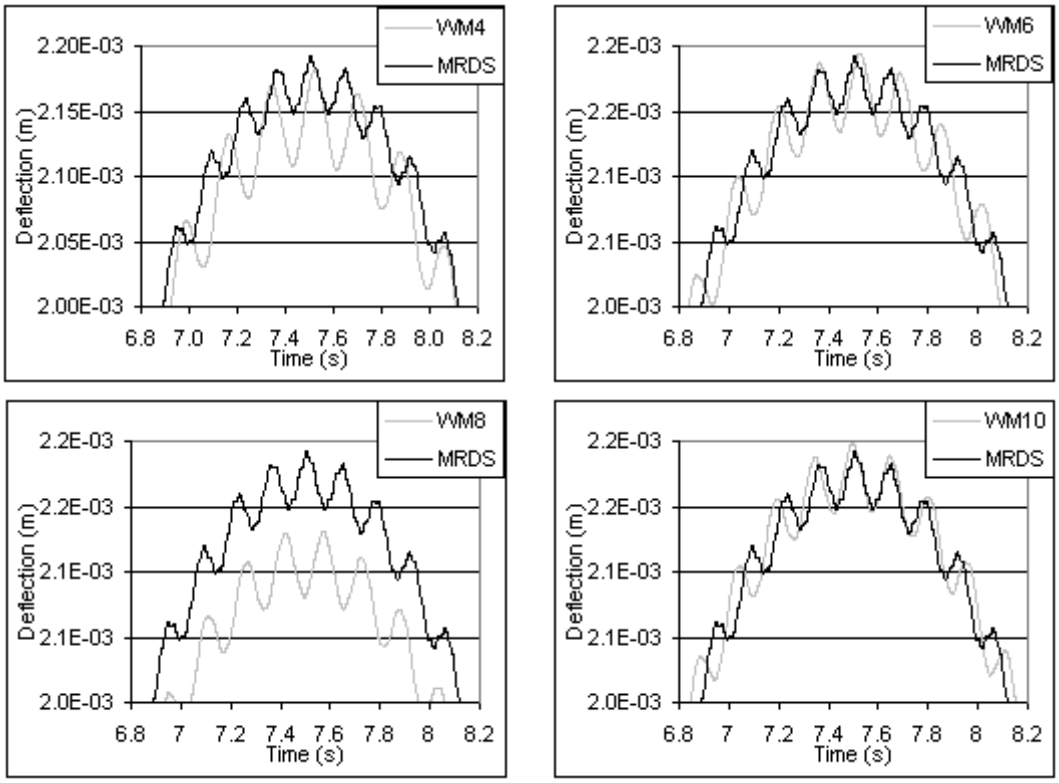

FIGURE 28. Improving Agreement with Increasing Segments Per Link in Working Model

8.3.2 System 2: Two-Link Flexible Robot

This system was created using the fixed-base, two-link module with the first link having a length of one meter and the second link being one half meter long. Three different cases were analyzed. The first case involved the simple open chain operation of the robot. The

second case was also open chain, but included a large, concentrated mass at the tip of the second robot. The final case modelled a closed chain operation, where the tip was constrained to slide in the x-direction only. Results from all cases are compared and contrasted below.

8.3.2.1 System 2a: Simple Open Chain

In this operation, the torque command parameters for the format given by Equation 134 are shown in Table 24. Figure C-2 shows the results for all of the standard outputs. As before, joint angles are in near perfect agreement, as are predictions for tip position (maximum errors of 0.0874 degrees and 4.678e-4 meters, respectively). Slight variations can be found between the predictions of tip deflection, as is expected given the differing methodologies for modelling link flexibility.

TABLE 24. Torque Parameters for System 2a

	i = 1	**i = 2**
A_i (Nm)	0.105	0.011
T_i (s)	10	10

8.3.2.2 System 2b: Open Chain with Concentrated Tip Mass

This case is similar to System 2a, but contains a 0.5 kg point mass at the tip of the second link. Figure C-3 shows the comparison of the standard outputs. As shown, peak joint velocities in this case are nearly the same as in System 2a, but this constitutes a higher percentage of the system's lowest natural frequency due to the presence of the tip mass. Not surprisingly, tip deflections of the links are considerably higher than for System 2a. Torque

signals for this system are based on the parameters in Table 25.

TABLE 25. Torque Parameters for System 2b

	i = 1	**i = 2**
A_i (Nm)	0.315	0.111
T_i (s)	10	10

In addition to larger deflections, this system also shows a more complex behavior of the flexible links than in the previous systems. Note that the sudden changes in tip deflections are well matched, and as before, the joint angle and tip position outputs are virtually indistinguishable. Improved agreement is again achieved as more segments are used in the lumped parameter model of Working Model. Figure 29 shows the predictions of the tip deflection of the first link for values of six and eight segments per link in Working Model. Clearly, the results from the eight segment-per-link case are superior to that of the six segment case.

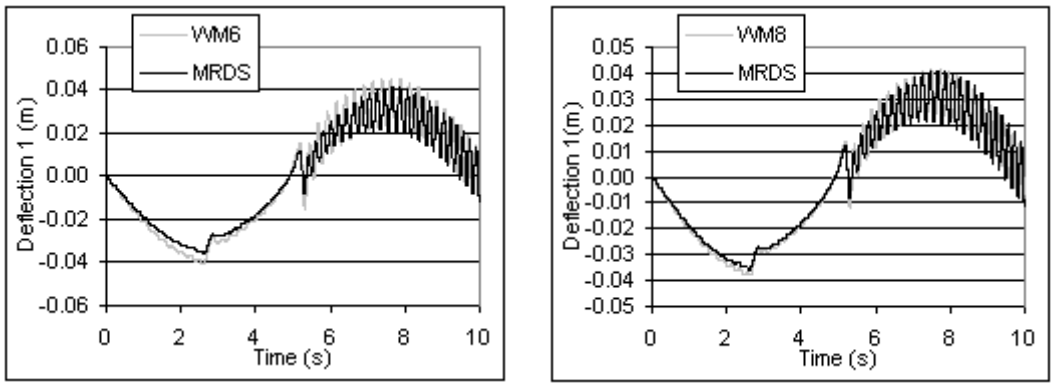

FIGURE 29. Another Example of Better Agreement with More Segments in Working Model

8.3.2.3 System 2c: Tip Constrained to Slide

In this case, the two-link robot has its tip constrained to slide in the x-direction at a fixed value of 0.6 meters in the y-direction. The sliding surface is modelled as frictionless, and tip rotation is unconstrained during the operation. The torque commands for this operation use the values shown in Table 26. Figures C-4 and C-5 show the output variables for this system, including in this case, the magnitude of the contact force at the tip constraint.

TABLE 26. Torque Parameters for System 2c

	i = 1	**i = 2**
A_i (Nm)	0.0531	0.0
T_i (s)	10	10

As this system is the first to employ a contact model, these results are also the first to show a plot of constraint violations. In a perfect simulation, the y-coordinate of the robot tip should hold exactly at 0.6 meters throughout the entirety of the simulation. As shown in Figure C-4, the Working Model simulation does deviate slightly, however, this amount is directly related to the accuracy settings set by the user. The MRDS results holds much more closely to the constrained y value, however, as shown in Figure C-5, the amount of deviation does begin to grow linearly towards the end of the simulation. Note that this occurs at the end of the simulation, where joint velocity reaches its highest values. Also note that these results were generated without any means of constraint violation suppression. Results using the constraint violation suppression technique from Section 6.4 are discussed in greater detail for later systems.

8.3.3 System 3: Kinematic Base Excitation of a Two-Link Robot

In this system, the two-link robot from System 2 is mounted on a mobile base that is constrained to move back and forth in the x-direction with the following translational velocity:

$$\dot{x} = 0.1 \cdot \sin(2 \cdot \pi \cdot t/5) \tag{135}$$

At the same time, the two joints are driven with sinusoidal torque commands with the values of amplitude and period given in Table 27.

TABLE 27. Torque Parameters for System 3

	i = 1	**i = 2**
A_i (Nm)	0.105	0.011
T_i (s)	10	10

The MRDS system model for this case uses two modules, each of which corresponds to a free-floating flexible link. Two contact models are used: 1) a revolute contact between the tip of the proximal module and the base of the distal module, and 2) a revolute contact between the base of the proximal module and the kinematically excited base of the system. Incorporation of the base excitation in the MRDS model was exceedingly simple, as the inclusion of the base kinematics can be achieved simply by specifying the open chain spatial acceleration of the moving base. This is simply the temporal derivative of Equation 135, which specified the translational base velocity. Consequently, no model of the physical parameters of the base are needed. It is simply a matter of setting $^{I}\ddot{x}_{IE}^{open}$ equal to the temporal derivative of Equation 135 in the calculation of term $^{E}\ddot{x}_{EP}^{open}$.

Figures C-6 and C-7 show the results for this case, where again, joint and tip agreement is excellent. Figure C-7 also shows the constraint violation associated with the revolute contact between the two links (modules). The violation is exceedingly small and oscillatory due, in part, to the flexibility in the system. Also, as shown, it remains bounded throughout the simulation. Again, no means of suppressing the constraint violations were used in the generations of these results.

Constraint violation of the contact between the moving base and the first link (module) is given by the difference between the x position of the proximal end of the first link and the kinematically established position corresponding to the input base velocity of Equation 135. Oddly enough, this violation is found to grow throughout the simulation by a peculiar amount. At exactly five seconds (equivalent to one full cycle of the base excitation), the violation was found to be exactly equal to $\pi/10000$ (i.e. 3.14159e-4). Similarly, at the end of the simulation, when exactly two full cycles have been completed, the error is twice the previous amount. It is believed that this is a numerical inaccuracy related to the integration of the sinusoidal base excitation function in Visual Basic. As discussed in Section 6.3, this constraint violation could be removed completely by passing the kinematic excitation directly to the module representing the first link. By setting the x position of the base directly from the kinematic excitation, rather than integrating the calculated x direction acceleration of the base of the first link, no violation would occur.

8.3.4 System 4: Macro/Mini Four-Link Robot

A four-link open chain robot was modelled, where the first two links each had a length of one meter. The two end links, having the same mass density and cross-section as the first

two links, were each given lengths of 0.1 meters. Structural flexibility was included for the two longer links, while the shorter links were modelled as rigid members.

Each of the four torque commands used the same sinusoidal input format given by Equation 134. The coefficients for each of joint were set as shown in Table 28.

TABLE 28. Torque Parameters for System 4

	i = 1	**i = 2**	**i = 3**	**i = 4**
A_i (Nm)	0.01	0.001	-0.0002	-0.00001
T_i (s)	80	80	10	10

Figures C-8 and C-9 show the comparison between MRDS and Working Model predictions. The MRDS model was achieved using three modules. The first represented the fixed-base two-link robot. This module was connected in series by a revolute contact model to the second module, which represented a single free-floating flexible link. The final module also represented a free-floating link, and was connected by its base through a revolute contact to the tip of the previous link (module). Constraint violations for each of these contact models are also given in Figure C-9, where again, the values are seen to be extremely small and remain bounded throughout the simulation, despite the lack of any constraint violation suppression.

8.3.5 System 5: A Four-Link Planar Flexible Robot

As in the previous case of the Macro/Mini four-link robot, three modules were used to create the examples in this four-link system configuration. Here, however, all four links of the robot were set to be one meter in length, and all were modelled with structural flexibil-

ity. Three operations are presented here. The first and second are considered slow and fast operations, respectively, while the third case exhibited large inertial loading.

8.3.5.1 System 5a: Slow Speed Operation

Coefficients for the four torque commands having the same form given by Equation 134 are shown in Table 29. Figures C-10 and C-11 show the results for this case. Again, constraint violations are small and bounded.

TABLE 29. Torque Parameters for System 5a

	i = 1	**i = 2**	**i = 3**	**i = 4**
A_i (Nm)	0.01	0.002	-0.002	-0.0015
T_i (s)	20	15	20	15

8.3.5.2 System 5b: High Speed Operation

The robot from System 5a was operated with higher joint torques, as shown in Table 30. Figures C-12 and C-13 show the predicted behavior. Note that the peak velocities of the first three joints are approximately ten times larger than in System 5a, and that of the fourth is nearly 20 times greater. Naturally, peak tip deflections are also much larger in this case.

TABLE 30. Torque Parameters for System 5b

	i = 1	**i = 2**	**i = 3**	**i = 4**
A_i (Nm)	-0.01	0.04	-0.04	-0.02
T_i (s)	40	15	15	30

For this system, again with no means of suppression in place, constraint violations are

no longer bounded and begin to grow linearly from five seconds into the simulation until the end of the twenty-second simulation.

8.3.5.3 System 5c: Large Inertial Loading

In this system, inertial loading is maximized by commanding the robot to rotate with all links fully extended. Torque coefficients for this operation are given in Table 31, while results are shown in Figures C-14 and C-15.

TABLE 31. Torque Parameters for System 5c

	i = 1	**i = 2**	**i = 3**	**i = 4**
A_i (Nm)	0.25	0.15	0.05	0.01
T_i (s)	80	80	40	40

This operation was the most problematic of all the example systems discussed in this chapter. Numerous variations of the model were explored in the attempt to produce a stable and accurate simulation. The fundamental model, developed using two cantilevered mode shapes for each link, became unstable and ultimately crashed approximately seven seconds into the 13-second operation. Using only one cantilevered mode shape for each link, the simulation again became unstable. In this case, the crash occurred approximately 12 seconds into the 13-second maneuver.

Other variations of the model included combinations of the following: 1) the use of second order kinematic terms, 2) implementation of the constraint violation suppression technique, 3) the use of substantially smaller timesteps, and 4) the use of Singular Value Decomposition routines in place of the LU Decompositions used for matrix inversions and

linear system solutions. None of these variations succeeded in producing a stable simulation.

Ultimately, an accurate and stable simulation was produced using the MRDS algorithm. This was achieved by including three cantilevered mode shapes for the first link of the four-link robot, and two cantilevered mode shapes for each of the remaining links. Although a full analysis of this result is still needed, it is theorized that the frequencies associated with the second bending modes of one or more of the distal links are actually higher than the third modal frequency of the first link. Consequently, the inclusion of an equal number of mode shapes for all four links results in the "skipping" of a lower system modal frequency. To capture this mode, which is apparently necessary to ensure stability, the first link must be modelled with one more mode shape than is used to model the three other links. This theory was further supported by the stable and accurate simulation achieved using two cantilevered mode shapes for the first link and a single cantilevered mode shape for the other three.

The difficulty in achieving a stable simulation of this operation is similar to a case discussed by Padilla and von Flotow [55]. As in their case, the difficulty here is attributed, at least in part, to the inappropriateness of the cantilevered mode shapes for this operation. The large inertia loading and larger motor torques exhibited here add significant tip forces and moments on each of the four flexible links (especially on the first link).

One possible improvement would be the use of the "CLTI" mode shapes employed in [51] and [36]. These mode shapes are based on a cantilevered beam with a tip inertia and would better incorporate the effect of the large inertial loading present on the tip of the first

link. Another examination of improved mode shapes is given by Haering in [23], where the accuracy of beam stress is examined using three different modelling approaches.

8.3.6 System 6: Parallel Cooperation of Two, Two-Link Flexible Robots

For this system, an un-powered revolute contact was used to connect two, two-link robots in parallel. In the MRDS model, two modules were used, each representing a two link flexible robot. Two examples are presented for this configuration. The first operation uses very small input torques. The second uses much higher torques and consequently generates significantly larger internal forces and deflections as the links pull against one another.

8.3.6.1 System 6a: Small Torque Input

Table 32 shows the sinusoidal torque coefficients used for this operation. For this system, all links were one meter long. Results are given in Figures C-16 and C-17. Also shown is the constraint violation of the single contact, which grows steadily from 2.58 seconds into the 30-second simulation. Using the constraint violation suppression technique described in Section 6.4, this growing violation is successfully kept in check for the duration of the simulation. This is shown in the last graph of Figure C-17.

TABLE 32. Torque Parameters for System 6a

	i = 1 Left	**i = 2 Left**	**i = 1 Right**	**i = 2 Right**
A_i (Nm)	0.02	0	0.02	0
T_i (s)	10	10	10	10

8.3.6.2 System 6b: Higher Torque Input with Higher Tensile Forces

System 6b was operated with larger input torque commands as given by Table 33 and with the second link of each robot reduced to 0.5 meters. As with System 5c, discussed above, the MRDS code was (initially) unable to complete the simulation of this system.

TABLE 33. Torque Parameters for System 6b

	i = 1 Left	**i = 2 Left**	**i = 1 Right**	**i = 2 Right**
A_i (Nm)	0.195	-0.125	0.45	0.001
T_i (s)	10	10	10	10

In this case, the difficulty is not attributable to high inertial loading, but rather to the sizeable error in the neglect of the foreshortening effects. In this system, a closed loop is formed through the cooperation of the two robots. Position constraints must be very precisely satisfied to complete loop closure. With the high torque commands applied to this system, transverse deflections are very large for this case. In fact, for the first link of each robot, transverse deflections are nearly 10% of the length of the links. This is the upper limit of validity for the simple Bernoulli-Euler beam assumptions.

With transverse deflections of this size, the magnitude of foreshortening will also be larger than in previous simulation results. Omission of the second order kinematics in this case contributes to a significant violation of the loop closure.

For this system, second order kinematics are necessary to achieve accurate and stable simulation results. With the inclusion of these higher order kinematic terms, excellent agreement with Working Model results as exhibited in the responses shown in Figures C-

18 and C-19.

The importance of second order kinematics in certain closed-loop systems is also reported in [63]. It is reported that the exactness in loop closure necessitates the inclusion of the foreshortening effects. Failure to include them produces a position error that is at least as significant as a constraint violation. This problem is not as prevalent with open chain systems, where position accuracy in loop closure is not an issue. The same holds true for the MRDS algorithm. Despite the use of contact modelling and kinematic constraints in serial connections between modules, the need for second order kinematics seems limited to only those systems with a kinematic closed loop and higher internal forces leading to large deflections.

8.4 Summary

The results described in this chapter show the success of the MRDS algorithm in solving the Forward Dynamics problem for a system of interacting dynamic modules. The results for these example systems show the ability to treat systems with serial and parallel cooperation, macro/mini systems, and systems with kinematic base excitation. The successful implementation of a simple constraint violation suppression technique is also demonstrated.

Structural flexibility was included in all systems, and response predictions were well matched by results from Working Model. The need to carefully select mode shapes in the modelling is demonstrated by the difficulty encountered with the four-link robot simulation when subjected to large inertial forces using simple cantilevered mode shapes. Similarly, the need to use full second order kinematics is shown for cases of closed-loop operations

subject to large internal forces. These areas of added caution are, of course, not unique to the MRDS approach, but are problematic for nearly any method of constrained motion flexible body modelling. Overall, the success of this modular approach to dynamic simulation is clearly demonstrated.

Chapter 9 SUMMARY AND FUTURE DEVELOPMENT

This dissertation presents a Modular Robot Dynamic Simulation (MRDS) algorithm that is applicable to multiple cooperating, structurally flexible robots with reconfigurable topology. The algorithm is designed to be applicable to numerous complexities found in modern robot systems. It is also intended to be both computationally efficient and efficient with regard to the analyst's time and effort in generating new simulations from existing dynamic building blocks.

The MRDS algorithm has its roots in the constrained motion algorithm of Lilly [39][40], which was originally developed for the dynamic simulation of simple closed chain mechanisms composed of rigid bodies. The work presented here extends this algorithm in a number of ways, ultimately evolving into a modular algorithm capable of simulating the dynamics of modules connected in series, in parallel, or hybrid configurations.

The general joint model of Roberson and Schwertassek [58] is incorporated into the MRDS algorithm, allowing joints and general contacts between pairs of rigid bodies. The model uses a dual basis to describe the structure of the spatial contact force and relative velocity between the contacting bodies. Precise notation is implemented, and a "standard pair" of the force and velocity vectors is established to ensure consistent application of the model. Use of this model allows the treatment of holonomic and non-holonomic constraints, as well as constant and time-varying contact modes.

The first full algorithm presented in the dissertation makes use of the new notation and upgraded contact model to achieve dynamic simulation of "single contact systems". These systems include the cooperation of two robots in parallel or in series and the closed-loop

motion of a single robot formed by contact with an environmental boundary or with one of its own internal member bodies. This latter example, denoted as a Type B contact, is shown to make use of a relative Jacobian matrix associated with the relative kinematics between the two contacting points within the same robot module. The resulting algorithm is shown to be O(N) for both Type A (external) and Type B (internal closed-loop) contacts.

The full modular algorithm is then developed for the dynamic simulation of multiple robots (or other dynamically significant objects/devices) subject to multiple concurrent contacts. New forms of the operational space inertia matrix are presented, which are used to relate spatial contact forces from one contact point (operational point) to spatial accelerations at a second contact point. These matrices are used in the generation of a linear system of equations in the system-wide unknown contact force components. The exact structure of the coefficient matrix of this equation is shown to depend solely on the topology of the system.

The algorithm's modularity allows all calculations pertaining to a single module to be performed without any knowledge or interaction from other modules in the system. This feature makes possible the use of parallel processing on multiple processors. Specifically, one processor can be assigned to calculate the open chain dynamics of one module in the multi-module system. This architecture provides the potential to greatly enhance the computational efficiency and real-time capability associated with the simulation of complex systems.

A great deal of attention is given to the subtleties involved in a serial connection of modules. Specifically, two methods are given for the calculation of the Forward Kinemat-

ics in the system. In the first method, these calculations are purely modular, and with the use of constraint violation suppression, base-body kinematics in a serially connected module need not “wait” for the kinematics of the supporting contact body. In the second method, constraint violations in serial connections are completely eliminated by setting the base-body kinematics associated with the constrained directions of the contact directly from the combination of the tip body kinematics and the contact constraints.

The MRDS algorithm is also shown to be applicable to systems composed of structurally flexible bodies. Included in this demonstration is a discussion of the second order strain and kinematic effects often neglected in dynamic modelling due to the premature linearization of the dynamic equations. Several linearization schemes from available literature on open chain manipulators are reviewed, and their incorporation into the MRDS algorithm is discussed. Special attention is given to the linearization process with regards to constrained motion dynamics and the use of the operational space dynamic formulation.

Validation of the MRDS algorithm is achieved through the simulation of numerous example systems and the subsequent comparison to simulation results obtained from Working Model® 2D. The simulations of several one and two-link, structurally flexible robots show the increasing correlation between Working Model and the MRDS algorithm as a greater number of segments per link are used in the lumped parameter models of Working Model.

More complex systems are built from three basic modular building blocks, which include a two-link, fixed base robot; a one-link, fixed base robot; and a one-link robot with a free-flying base. All modules include structural flexibility using the Bernoulli-Euler beam assumptions and the assumed modes method with zero, one, two, or three cantilevered

mode shapes per link. Second order kinematic relationships were used in the derivation of all kinematic and dynamic equations for each module, and several linearization schemes were examined.

Example systems included the base excitation of the two-link module, the serial cooperation of three modules to form a four-link robot with all links of equal magnitude, and the latter example repeated, but with a Macro/Mini construction. Excellent correlation between Working Model and the MRDS results is shown in a multitude of predicted responses. Simulation results showed that cases involving closed-loops and large internal forces require the inclusion of all second order kinematic terms, including the associated first order dynamic effects. This was not the case for open-loop maneuvers and closed-loop operations with lower interbody forces.

A constraint violation suppression system is shown to be successful at limiting the growth of violations in the few cases exhibiting any such growth. This method involves a proportional and derivative control applied to the solution for the contact forces, where the gains are applied to the position and velocity constraint violations, respectively. No adverse effect on the response predictions is observed from the inclusion of the constraint violation suppression due to the small gains used and the very small errors observed at each timestep.

9.1 Advantages, Achievements, and Impact

Dynamic simulation of multi-robot systems with complex and time-varying topologies has always been a difficult task. With the achievements and capabilities of the MRDS algorithm, this task is made significantly easier. Specific impact of this work in the field of robot dynamic simulation is outlined here.

MRDS is the only known Forward Dynamics algorithm applicable to multiple robots with multiple constraints, constrained flexible systems, topologies composed of series and/or parallel connections, holonomic and non-holonomic contacts, and both constant and time-varying contact modes. The algorithm also allows efficient handling of modules that form a single internal closed loop.

The MRDS algorithm makes use of the operational space dynamic formulation, which has been shown by many researchers to be efficient in the simulation of multiple robots co-operating in parallel and single robots performing constrained motion tasks. Application of the operational space dynamic formulation to the simulation of a series connection of multiple robots is new to the field and is an integral part in achieving the modularity of the algorithm. Another new contribution is the incorporation of the recently developed multi-point operational space dynamic formulation into the simulation of robots subject to multiple concurrent contacts.

Application of the operational space formulation to constrained flexible systems is also a recent achievement in existing simulation algorithms. This work addresses the issue of second order strain and kinematic effects and explores the requirements and ramifications associated with different linearization schemes. Prior to this work, the investigation of suitable linearization schemes and their impact on non-linear dynamic effects has been focused almost entirely on open-chain, single robot systems.

A prominent feature of this algorithm is that its modular construction allows increased computational potential by allowing modular level terms to be computed simultaneously on parallel processors. In addition, efficient O(N) methods may be employed for all mod-

ular calculations, with the exception of the force coefficient matrices for flexible systems. It is expected, however, that these matrices can also be found by O(N) means, and this will be the subject of further research.

The MRDS algorithm is focused not only on computational efficiency, but also on the efficiency with respect to the analyst's time and effort. The modular nature of the algorithm makes it ideal for the simulation of complex reconfigurable systems, such as those used in space applications. New simulations of alternate topologies of the same basic modules can be generated simply by redefining the modular connectivity. The analyst is not burdened with the derivation of new equations or coupling terms and is spared from making significant revisions to existing simulation coding. This is a new achievement for serial connections of robots.

9.2 Current Limitations and Future Work

Throughout the development of this algorithm, a small number of limitations have been noted. These limitations are aspects of the algorithm in need of further research in the effort to provide the most efficient, general, and useful constrained dynamic simulation algorithm possible.

One such limitation is the current lack of an O(N) method for the computation of the various inertia matrices ($\Lambda^{-1}_{O_i O_j}$ and ${}^{I}\Psi_{IO_i}$) required by the algorithm for structurally flexible modules. At this time, O(N) methods are only available for rigid body manipulators. In the case of flexible bodies, the direct multiplication methods must be used. It is expected, however, that O(N) methods can be found with further research. In that event, the entire algorithm will be achievable in O(N) computational complexity.

Another limitation is the requirement that the general joint model be applied to contacts between two rigid bodies. Although the algorithm is applicable to structurally flexible systems, modular interactions that make use of the contact model cannot currently be applied to deformable bodies. A very useful exception to this is when the operational frames (E and P) can be assigned at the contact point or contact surface, rather than an internal location within the deformable body. This exception is exploited in the example systems presented in the dissertation, where external contacts between flexible links were described in frames associated with the tip or base of the interacting links.

Another area of future work deals with the efficiency of the global algorithm. The algorithm currently requires the inversion of a potentially large matrix in the global solution for the unknown contact force components. It is expected, however, that this process could be broken down into smaller inversions based on the topology of the system. This process could lead to a series of six-by-six or smaller inversions, as opposed to inversion of the entire global coefficient matrix. Until this recursive solution is explored further, the current method can make use of more computationally efficient inversion processes whenever this coefficient matrix is found to be sparse or banded. This will be dictated by the topology of the total system.

As suggested in Chapter 7, further investigation is needed in relation to the effect of linearization on constrained flexible systems. As has been the case with other researchers for open chain flexible models, the effect of various levels of linearization needs to be explored. The so-called “ruthless”, “consistent”, “inconsistent”, and fully non-linear models should be developed and compared for two-dimensional and three-dimensional constrained

flexible robots. Insightful but conflicting conclusions have been reached by researchers for open chain systems. The preliminary investigation put forth here suggests that systems containing a closed kinematic loop are more likely to require full second order kinematics. Linearized kinematics seem to be acceptable for modules cooperating in an open-loop formation.

Future developments of the MRDS system will also deal with more “user friendly” aspects, such as the inclusion of a graphical interface for building the modular topology and the development of module and contact model libraries. The graphical interface would allow the user to establish the modular topology of the system using drag and drop routines to layout squares and circles (modules and contact models). Each of these elements could then be correlated to a selection from the library of existing modules and contacts. A utility to import new models from existing or newly established code would be available. Other user-friendly enhancements beyond the realm of the Forward Dynamics problem would be the development of modular approaches to Trajectory Planning, Inverse Kinematics, and Inverse Dynamics.

To minimize the need for contact point dependent programming or repeated reprogramming, module coding would be constructed in a standard form. Simulation coding for the open chain generalized coordinate accelerations ($\ddot{q}^{open}$) would be developed for the module regardless of contact locations. Coding for the kinematic and inertial quantities that are specific to the location of the contact point, however, would need to be open-ended. For this purpose, the code should be able to compute these quantities for any point on any link on demand.

With an open-ended ability to compute terms specific to the contact location within a module, the MRDS system would begin to approach “plug-and-play” dynamic simulation. Obviously, this is a simplification of the effort it takes to achieve an efficient, accurate, stable dynamic simulation. However, the architecture of the MRDS algorithm has the potential for this kind of use in the hands of a careful analyst. In this proposed environment, the analyst would still be responsible for “tuning” the system to achieve proper results. This includes the handling of the integration scheme (method used, step size, accuracy controller, etc.), the inclusion or exclusion of appropriate terms, adjusting of mode shapes for changing boundary conditions, suppression of constraint violations, etc.

In conclusion, it is clear that there are an endless number of avenues that could be explored in extending the MRDS system. This includes investigations into the handling of flexible joints, actuator dynamics, impact dynamics, friction, material damping, and many other similar features. With this in mind, the MRDS algorithm has been purposefully constructed to have a very wide range of applicability and to be both open-ended and inclusive.

In this dissertation, the fundamental architecture of the MRDS algorithm has been developed and presented. The system, in its current form, has been shown to already be applicable to many systems with a variety of complexities. In time, and with the further research and enhancements suggested here, the MRDS algorithm should continue to support more and more complex features found in the multi-robot systems of today and in those to come.

Appendix A DYNAMIC EQUATIONS OF EXAMPLE SYSTEMS

As described in Chapter 8, three modules were developed for validation of the MRDS algorithm. The first of these modules represents a two-link planar flexible robot with two revolute joints and a fixed base. The second represents a single rotating flexible link, and the dynamic equations for this module can be formed from those of the two-link robot by removing all terms and variables pertaining to that robot's second link. The final module represents a free-flying, planar, flexible link.

In all cases, the dynamic equations were derived using Extended Hamilton's Principle, the Bernoulli-Euler assumptions for link flexibility, the assumed modes method with cantilevered mode shapes, and a second order kinematic description of link bending (i.e. the inclusion of foreshortening effects). The sections that follow give the full kinematic and dynamic equations derived and used in the simulation coding.

A.1 Motion Equations for the Two-Link Module

See Figure 24 and Section 8.1 for a description of the geometry and coordinates associated with the two-link module.

$$\tau = H \cdot \ddot{q} + C + K \cdot q + J^T \cdot F$$

$$\tau = \begin{bmatrix} \tau_1 \\ \tau_2 \\ \tau_{f1} \\ \tau_{f2} \end{bmatrix} = \text{the generalized joint torque/force vector}$$

$$H = \begin{bmatrix} H_{1,1} & H_{1,2} & H_{1,f1} & H_{1,f2} \\ H_{2,1} & H_{2,2} & H_{2,f1} & H_{2,f2} \\ H_{f1,1} & H_{f1,2} & H_{f1,f1} & H_{f1,f2} \\ H_{f2,1} & H_{f2,2} & H_{f2,f1} & H_{f2,f2} \end{bmatrix} = \text{the joint space inertia matrix}$$

$$C = \begin{bmatrix} C_1 \\ C_2 \\ C_{f1} \\ C_{f2} \end{bmatrix} = \text{the vector of generalized Coriolis and centrifugal force terms}$$

$$K = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & K_{f1,f1} & K_{f1,f2} \\ 0 & 0 & K_{f2,f1} & K_{f2,f2} \end{bmatrix} = \text{the stiffness matrix (including geometric and centrifugal)}$$

$$J = \begin{bmatrix} J_{x,1} & J_{x,2} & J_{x,f1} & J_{x,f2} \\ J_{y,1} & J_{y,2} & J_{y,f1} & J_{y,f2} \\ J_{\theta,1} & J_{\theta,2} & J_{\theta,f1} & J_{\theta,f2} \end{bmatrix} = \text{the Jacobian matrix for the tip of the second link}$$

$$q = \begin{bmatrix} q_1 \\ q_2 \\ q_{f1} \\ q_{f2} \end{bmatrix} = \begin{bmatrix} \theta_1 \\ \theta_2^r + \bar{w}'_1 \\ q_{f1} \\ q_{f2} \end{bmatrix} = \text{the generalized coordinate vector}$$

Generalized coordinates, q_1 and q_2 are scalar quantities, while the term q_{f1} is $NF1 \times 1$ and q_{f2} is $NF2 \times 1$. NF1 and NF2 are the number of cantilevered mode shapes used to model the structural flexibility of links 1 and 2, respectively. The scalar joint torque components τ_1 and τ_2 are the supplied joint torques for joints 1 and 2, respectively, while τ_{f1} is $NF1 \times 1$, and τ_{f2} is $NF2 \times 1$. These latter terms are expressed as follows:

$$\tau_{f1}(i) = -\bar{\sigma}'_1(i) \cdot \tau_2$$

$$\tau_{f2}(i) = 0$$

The dimension of all other terms in this section can be determined by compatibility with the dimensions of the four terms in the generalized coordinate vector, q, and those in the generalized joint torque vector, τ.

Transverse tip deflection, deflection rate, and slope, are given, respectively, by the following three equations for link m (m = 1 or 2).

$$\bar{w}_m = \sum_{j=1}^{NFm} \bar{\sigma}_m(j) \cdot q_{fm}(j)$$

$$\dot{\bar{w}}_m = \sum_{j=1}^{NFm} \bar{\sigma}_m(j) \cdot \dot{q}_{fm}(j)$$

$$\bar{w}'_m = \sum_{j=1}^{NFm} \bar{\sigma}'_m(j) \cdot q_{fm}(j)$$

where the ' indicates a spatial derivative with respect to the x coordinate (defining the location along the undeformed longitudinal axis of the link) and quantities evaluated at the tip of a link are indicated by the use of the “overbar”.

Link foreshortening, foreshortening rate, and foreshortening acceleration are given, respectively, by the following three equations for link 1.

$$\bar{u}_1 = -\frac{1}{2} \cdot \sum_{i=1}^{NF1} \sum_{j=1}^{NF1} a_{13}(i,j) \cdot q_{f1}(j) \cdot q_{f1}(i)$$

$$\dot{\bar{u}}_1 = -\sum_{i=1}^{NF1} \sum_{j=1}^{NF1} a_{13}(i,j) \cdot \dot{q}_{f1}(j) \cdot q_{f1}(i)$$

$$\ddot{\bar{u}}_1 = -\sum_{i=1}^{NF1} \sum_{j=1}^{NF1} (a_{13}(i,j) \cdot (\ddot{q}_{f1}(j) \cdot q_{f1}(i) + \dot{q}_{f1}(j) \cdot \dot{q}_{f1}(i)))$$

The same quantities for link 2 are given, respectively, by the following three equations.

$$\bar{u}_2 = -\frac{1}{2} \cdot \sum_{i=1}^{NF2} \sum_{j=1}^{NF2} b_{23}(i,j) \cdot q_{f2}(j) \cdot q_{f2}(i)$$

$$\dot{\bar{u}}_2 = -\sum_{i=1}^{NF2} \sum_{j=1}^{NF2} b_{23}(i,j) \cdot \dot{q}_{f2}(j) \cdot q_{f2}(i)$$

$$\ddot{\bar{u}}_2 = -\sum_{i=1}^{NF2} \sum_{j=1}^{NF2} (b_{23}(i,j) \cdot (\ddot{q}_{f2}(j) \cdot q_{f2}(i) + \dot{q}_{f2}(j) \cdot \dot{q}_{f2}(i)))$$

The first three cantilevered mode shapes and their spatial derivatives have the following boundary values at the tip of link m (m = 1 or 2):

$$\bar{\sigma}_m(1) = 2$$

$$\bar{\sigma}_m(2) = -2$$

$$\bar{\sigma}_m(3) = 2$$

$$\bar{\sigma}'_m(1) = 2.75301096934507 / L_m$$

$$\bar{\sigma}'_m(2) = -9.56155682042328 / L_m$$

$$\bar{\sigma}'_m(3) = 15.6973320929786 / L_m$$

where L_m is the length of link m. Other material properties appearing in the equations of this Appendix are as follows (again for link m, where m = 1 or 2):

ρ_m = mass per unit length of link m

EI_m = bending inertia of link m

M = mass concentrated at tip of link 2

Terms from the dynamic equation, shown above are as follows:

$$H_{1,1} = P_1 + P_2 + P_4 + 2 \cdot A_1$$

$$H_{1,2} = P_2 + A_1$$

$$H_{1,f1}(j) = a_{12}(j) + (P_3 + A_6) \cdot \bar{\sigma}_1(j)$$

$$H_{1,f2}(j) = A_{27}(j)$$

$$H_{2,1} = H_{1,2}$$

$$H_{2,2} = P_2$$

$$H_{2,f1}(j) = A_6 \cdot \bar{\sigma}_1(j)$$

$$H_{2,f2}(j) = z_5(j)$$

$$H_{f1,1}(i) = H_{1,f1}(i) + A_{19}(i) \cdot s_2$$

$$H_{f1,2}(i) = H_{2,f1}(i) + A_{19}(i) \cdot s_2$$

$$H_{f1,f1}(i,j) = a_{14}(i,j) + P_5(i,j)$$

$$H_{f1,f2}(i,j) = z_4(j) \cdot c_2 \cdot \bar{\sigma}_1(i)$$

$$H_{f2,1}(i) = H_{1,f2}(i) + A_{210}(i) \cdot s_2$$

$$H_{f2,2}(i) = H_{2,f2}(i)$$

$$H_{f2,f1}(i,j) = H_{f1,f2}(j,i)$$

$$H_{f2,f2}(i,j) = z_6(i,j)$$

$$C_1 = A_2 \cdot \dot{q}_1 + A_5 \cdot L_1 \cdot T_2 - A_4 \cdot T_1$$

$$C_2 = \dot{q}_1 \cdot (A_2 + A_4 \cdot \dot{q}_1)$$

$$C_{f1}(i) = T_3 \cdot A_{110}(i) + A_5 \cdot \bar{\sigma}_1(i) \cdot T_2 - \dot{q}_1^2 \cdot A_{18}(i)$$

$$C_{f2}(i) = \dot{q}_1 \cdot (A_{29}(i) + \dot{q}_1 \cdot A_{28}(i))$$

$$K_{f1,f1}(i,j) = a_{16}(i,j) + T_3 \cdot A_6 \cdot a_{13}(i,j) + \dot{q}_1^2 \cdot (a_{15}(i,j) - a_{14}(i,j) + P_3 \cdot a_{13}(i,j))$$

$$K_{f1,f2}(i,j) = -T_3 \cdot \bar{\sigma}_1(i) \cdot c_2 \cdot z_4(j)$$

$$K_{f2,f1}(i,j) = -\dot{q}_1^2 \cdot c_2 \cdot z_4(i) \cdot \bar{\sigma}_1(j)$$

$$K_{f2,f2}(i,j) = a_{26}(i,j) + T_3 \cdot y_3(i,j) + \dot{q}_1^2 \cdot c_2 \cdot L_1 \cdot y_4(i,j)$$

$$J_{x,1} = -(L_1 + \bar{u}_1) \cdot s_1 - \bar{w}_1 \cdot c_1 - (L_2 + \bar{u}_2) \cdot s_{12} - \bar{w}_2 \cdot c_{12}$$

$$J_{x,2} = -(L_2 + \bar{u}_2) \cdot s_{12} - \bar{w}_2 \cdot c_{12}$$

$$J_{x,f1}(j) = -s_1 \cdot \bar{\sigma}_1(j) - d_{13}(j) \cdot c_1$$

$$J_{x,f2}(j) = -s_{12} \cdot \bar{\sigma}_2(j) - e_{23}(j) \cdot c_{12}$$

$$J_{y,1} = (L_1 + \bar{u}_1) \cdot c_1 - \bar{w}_1 \cdot s_1 + (L_2 + \bar{u}_2) \cdot c_{12} - \bar{w}_2 \cdot s_{12}$$

$$J_{y,2} = (L_2 + \bar{u}_2) \cdot c_{12} - \bar{w}_2 \cdot s_{12}$$

$$J_{y,f1}(j) = c_1 \cdot \bar{\sigma}_1(j) - d_{13}(j) \cdot s_1$$

$$J_{y,f2}(j) = c_{12} \cdot \bar{\sigma}_2(j) - e_{23}(j) \cdot s_{12}$$

$$J_{\theta,1} = 1$$

$$J_{\theta,2} = 1$$

$$J_{\theta,f1}(j) = 0$$

$$J_{\theta,f2}(j) = \bar{\sigma}'_2(j)$$

Trigonometric terms:

$$c_1 = \cos(q_1)$$

$$s_1 = \sin(q_1)$$

$$c_2 = \cos(q_2)$$

$$s_2 = \sin(q_2)$$

$$c_{12} = \cos(q_1 + q_2)$$

$$s_{12} = \sin(q_1 + q_2)$$

Constant terms:

$$z_1 = \rho_2 \cdot L_2 + M$$

$$z_2 = \rho_2 \cdot L_2 / 2 + M$$

$$z_3 = \rho_2 \cdot L_2/3 + M$$

$$z_4(i) = \rho_2 \cdot a_{21}(i) + M \cdot \bar{\sigma}_2(i)$$

$$z_5(i) = \rho_2 \cdot a_{22}(i) + M \cdot L_2 \cdot \bar{\sigma}_2(i)$$

$$z_6(i, j) = \rho_2 \cdot a_{24}(i, j) + M \cdot \bar{\sigma}_2(i) \cdot \bar{\sigma}_2(j)$$

$$P_1 = \rho_1 \cdot L_1^3/3$$

$$P_2 = z_3 \cdot L_2^2$$

$$P_3 = z_1 \cdot L_1$$

$$P_4 = z_1 \cdot L_1^2$$

$$P_5(i, j) = z_1 \cdot \bar{\sigma}_1(i) \cdot \bar{\sigma}_1(j)$$

Terms that vary with time:

$$T_1 = (2 \cdot \dot{q}_1 + \dot{q}_2) \cdot \dot{q}_2$$

$$T_2 = \dot{q}_1 + \dot{q}_2$$

$$T_3 = (\dot{q}_1 + \dot{q}_2)^2$$

$$A_1 = z_2 \cdot (L_1 \cdot L_2 \cdot c_2 + \bar{w}_1 \cdot L_2 \cdot s_2) - y_1 \cdot L_1 \cdot s_2$$

$$A_2 = 2 \cdot z_2 \cdot L_2 \cdot s_2 \cdot \dot{\bar{w}}_1$$

$$A_4 = z_2 \cdot (L_1 \cdot L_2 \cdot s_2 - \bar{w}_1 \cdot L_2 \cdot c_2) + y_1 \cdot L_1 \cdot c_2$$

$$A_5 = -2 \cdot y_2 \cdot s_2$$

$$A_6 = z_2 \cdot L_2 \cdot c_2$$

$$A_{18}(i) = z_1 \cdot \bar{w}_1 \cdot \bar{\sigma}_1(i)$$

$$A_{19}(i) = z_2 \cdot d_{13}(i) \cdot L_2 - y_1 \cdot \bar{\sigma}_1(i)$$

$$A_{110}(i) = -z_2 \cdot L_2 \cdot s_2 \cdot \bar{\sigma}_1(i)$$

$$A_{27}(i) = z_4(i) \cdot L_1 \cdot c_2 + z_5(i)$$

$$A_{28}(i) = z_4(i) \cdot L_1 \cdot s_2$$

$$A_{29}(i) = 2 \cdot z_4(i) \cdot \dot{\bar{w}}_1 \cdot s_2$$

$$A_{210}(i) = z_4(i) \cdot \bar{w}_1 - y_5(i) \cdot L_1$$

$$y_1 = \rho_2 \cdot d_{21} + M \cdot \bar{w}_2$$

$$y_2 = \sum_{j=1}^{NF2} [(\rho_2 \cdot a_{21}(j) + M \cdot \bar{\sigma}_2(j)) \cdot \dot{q}_{f2}(j)]$$

$$y_3(i, j) = 0.5 \cdot \rho_2 \cdot a_{25}(i, j) + M \cdot L_2 \cdot b_{23}(i, j) - (\rho_2 \cdot a_{24}(i, j) + M \cdot \bar{\sigma}_2(i) \cdot \bar{\sigma}_2(j))$$

$$y_4(i, j) = \rho_2 \cdot a_{23}(i, j) + M \cdot b_{23}(i, j)$$

$$y_5(i) = \sum_{j=1}^{NF2} y_4(i, j) \cdot q_{f2}(j)$$

$$d_{13}(i) = \sum_{j=1}^{NF1} a_{13}(i, j) \cdot q_{f1}(j)$$

$$d_{21} = \sum_{j=1}^{NF2} a_{21}(j) \cdot q_{f2}(j)$$

$$e_{23}(i) = \sum_{j=1}^{NF2} b_{23}(i, j) \cdot q_{f2}(j)$$

Constant functions of the mode shapes:

$$a_{12}(i) = \rho_1 \cdot \int_0^{L1} x_1 \cdot \sigma_1(i) \cdot dx_1$$

$$a_{12}(1) = \rho_1 \cdot L_1^2 \cdot 0.568825743709911$$

$$a_{12}(2) = \rho_1 \cdot L_1^2 \cdot 0.090766786886387$$

$$a_{12}(3) = \rho_1 \cdot L_1^2 \cdot 3.24163743697444E\text{-}02$$

$$a_{13}(i, j) = \int_0^{L1} \sigma'_1(i) \cdot \sigma'_1(j) \cdot dx_1$$

$$a_{13}(1, 1) = 4.64777831867864 / L_1$$

$$a_{13}(1, 2) = a_{13}(2, 1) = -7.37987526966198 / L_1$$

$$a_{13}(1, 3) = a_{13}(3, 1) = 3.9415154871266 / L_1$$

$$a_{13}(2, 2) = 32.417399027969 / L_1$$

$$a_{13}(2, 3) = a_{13}(3, 2) = -22.3524444398869 / L_1$$

$$a_{13}(3, 3) = 77.2988908022961 / L_1$$

$$a_{14}(i, j) = \rho_1 \cdot \int_0^{L1} \sigma_1(i) \cdot \sigma_1(j) \cdot dx_1$$

$$a_{14}(1, 1) = a_{14}(2, 2) = a_{14}(3, 3) = \rho_1 \cdot L_1$$

$$a_{14}(1, 2) = a_{14}(1, 3) = a_{14}(2, 1) = 0$$

$$a_{14}(2, 3) = a_{14}(3, 1) = a_{14}(3, 2) = 0$$

$$a_{15}(i, j) = \frac{1}{2} \cdot \rho_1 \cdot \int_0^{L1} (L_1^2 - x_1^2) \cdot \sigma'_1(i) \cdot \sigma'_1(j) \cdot dx_1$$

$$a_{15}(1, 1) = \frac{1}{2} \cdot \rho_1 \cdot L_1 \cdot 2.38667274821651$$

$$a_{15}(1, 2) = a_{15}(2, 1) = \frac{1}{2} \cdot \rho_1 \cdot L_1 \cdot -1.37171056689486$$

$$a_{15}(1, 3) = a_{15}(3, 1) = \frac{1}{2} \cdot \rho_1 \cdot L_1 \cdot -1.58475844061916$$

$$a_{15}(2, 2) = \frac{1}{2} \cdot \rho_1 \cdot L_1 \cdot 12.9564497281511$$

$$a_{15}(2,3) = a_{15}(3,2) = \frac{1}{2} \cdot \rho_1 \cdot L_1 \cdot 0.338815769523045$$

$$a_{15}(3,3) = \frac{1}{2} \cdot \rho_1 \cdot L_1 \cdot 35.7190397578695$$

$$a_{16}(i,j) = EI_1 \cdot \int_0^{L1} \sigma''_1(i) \cdot \sigma''_1(j) \cdot dx_1$$

$$a_{16}(1,1) = EI_1 \cdot 12.3623633683262 / L_1^3$$

$$a_{16}(2,2) = EI_1 \cdot 485.518818513371 / L_1^3$$

$$a_{16}(3,3) = EI_1 \cdot 3806.54626639145 / L_1^3$$

$$a_{16}(1,2) = a_{16}(1,3) = a_{16}(2,1) = 0$$

$$a_{16}(2,3) = a_{16}(3,1) = a_{16}(3,2) = 0$$

$$a_{21}(i) = \int_0^{L2} \sigma_2(i) \cdot dx_2$$

$$a_{21}(1) = L_2 \cdot 0.782991756039626$$

$$a_{21}(2) = L_2 \cdot 0.433935895110719$$

$$a_{21}(3) = L_2 \cdot 0.254425296866106$$

$$a_{22}(i) = \int_0^{L2} x_2 \cdot \sigma_2(i) \cdot dx_2$$

$$a_{22}(1) = L_2^2 \cdot 0.568825743709911$$

$$a_{22}(2) = L_2^2 \cdot 0.090766786886387$$

$$a_{22}(3) = L_2^2 \cdot 3.24163743697444E\text{-}02$$

$$a_{23}(i, j) = \int_0^{L2} (L_2 - x_2) \cdot \sigma'_2(i) \cdot \sigma'_2(j) \cdot dx_2$$

$a_{23}(1, 1) = 1.57087818999425$

$a_{23}(1, 2) = a_{23}(2, 1) = -0.422320389109118$

$a_{23}(1, 3) = a_{23}(3, 1) = -1.07208483410593$

$a_{23}(2, 2) = 8.64714269356121$

$a_{23}(2, 3) = a_{23}(3, 2) = 1.89005470803216$

$a_{23}(3, 3) = 24.9521133081691$

$$a_{24}(i, j) = \int_0^{L2} \sigma_2(i) \cdot \sigma_2(j) \cdot dx_2$$

$a_{24}(1, 1) = a_{24}(2, 2) = a_{24}(3, 3) = L_2$

$a_{24}(1, 2) = a_{24}(1, 3) = a_{24}(2, 1) = 0$

$a_{24}(2, 3) = a_{24}(3, 1) = a_{24}(3, 2) = 0$

$$a_{25}(i, j) = \int_0^{L2} (L_2^2 - x_2^2) \cdot \sigma'_2(i) \cdot \sigma'_2(j) \cdot dx_2$$

$a_{25}(1, 1) = L_2 \cdot 2.38667274821651$

$a_{25}(1, 2) = a_{25}(2, 1) = L_2 \cdot -1.37171056689486$

$a_{25}(1, 3) = a_{25}(3, 1) = L_2 \cdot -1.58475844061916$

$a_{25}(2, 2) = L_2 \cdot 12.9564497281511$

$a_{25}(2, 3) = a_{25}(3, 2) = L_2 \cdot 0.338815769523045$

$$a_{25}(3, 3) = L_2 \cdot 35.7190397578695$$

$$a_{26}(i, j) = EI_2 \cdot \int_0^{L1} \sigma''_2(i) \cdot \sigma''_2(j) \cdot dx_2$$

$$a_{26}(1, 1) = EI_2 \cdot 12.3623633683262 / L_2^3$$

$$a_{26}(2, 2) = EI_2 \cdot 485.518818513371 / L_2^3$$

$$a_{26}(3, 3) = EI_2 \cdot 3806.54626639145 / L_2^3$$

$$a_{26}(1, 2) = a_{26}(1, 3) = a_{26}(2, 1) = 0$$

$$a_{26}(2, 3) = a_{26}(3, 1) = a_{26}(3, 2) = 0$$

$$b_{23}(i, j) = \int_0^{L2} \sigma'_2(i) \cdot \sigma'_2(j) \cdot dx_2$$

$$b_{23}(1, 1) = 4.64777831867864 / L_2$$

$$b_{23}(1, 2) = b_{23}(2, 1) = -7.37987526966198 / L_2$$

$$b_{23}(1, 3) = b_{23}(3, 1) = 3.9415154871266 / L_2$$

$$b_{23}(2, 2) = 32.417399027969 / L_2$$

$$b_{23}(2, 3) = b_{23}(3, 2) = -22.3524444398869 / L_2$$

$$b_{23}(3, 3) = 77.2988908022961 / L_2$$

A.2 Motion Equations for the Mobile-Link Module

See Figure 25 and Section 8.1 for a description of the geometry and coordinates associated with the one-link module.

$$\tau = H \cdot \ddot{q} + C + K \cdot q + J_{TIP}^{T} \cdot F_{TIP} + J_{BASE}^{T} \cdot F_{BASE}$$

$$\tau = \begin{bmatrix} \tau_x \\ \tau_y \\ \tau_\theta \\ \tau_{f1} \end{bmatrix} = \text{the generalized joint torque/force vector}$$

$$H = \begin{bmatrix} H_{x,x} & H_{x,y} & H_{x,\theta} & H_{x,f1} \\ H_{y,x} & H_{y,y} & H_{y,\theta} & H_{y,f1} \\ H_{\theta,x} & H_{\theta,y} & H_{\theta,\theta} & H_{\theta,f1} \\ H_{f1,x} & H_{f1,y} & H_{f1,\theta} & H_{f1,f1} \end{bmatrix} = \text{the joint space inertia matrix}$$

$$C = \begin{bmatrix} C_x \\ C_y \\ C_\theta \\ C_{f1} \end{bmatrix} = \text{the generalized vector of Coriolis and centrifugal forces}$$

$$K = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & K_{f1,f1} \end{bmatrix} = \text{the stiffness matrix (including geometric and centrifugal)}$$

$$J_{TIP} = \begin{bmatrix} J_{x,x} & J_{x,y} & J_{x,\theta} & J_{x,f1} \\ J_{y,x} & J_{y,y} & J_{y,\theta} & J_{y,f1} \\ J_{\theta,x} & J_{\theta,y} & J_{\theta,\theta} & J_{\theta,f1} \end{bmatrix} = \text{the Jacobian matrix for the tip of the link}$$

$$J_{BASE} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} = \text{the Jacobian matrix for the base of the link}$$

$$q = \begin{bmatrix} q_x \\ q_y \\ q_\theta \\ q_{f1} \end{bmatrix} = \begin{bmatrix} x_I \\ y_I \\ \theta_1 \\ q_{f1} \end{bmatrix} = \text{the generalized coordinate vector}$$

Generalized coordinates, q_x, q_y, and q_z are scalar quantities, while the term q_{f1} is $NF1 \times 1$. NF1 is the number of cantilevered mode shapes used to model the structural flexibility of link 1. The terms τ_x and τ_y are the known forces acting on the base of the link in the x_I and y_I directions. They are both scalar values. τ_θ, also a scalar value, is the known torque applied to the base of the link. τ_{f1} is $NF1 \times 1$ and can be expressed as follows:

$$\tau_{f1}(i) = 0$$

The dimension of all other terms in this section can be determined by compatibility with the dimensions of the four terms in the generalized coordinate vector, q, and those in the generalized joint torque vector, τ.

New terms for this module are shown below. The definitions for terms a_{12}, a_{13}, a_{14}, a_{15}, and a_{16} are the same as in the previous section, as are all incarnations of “w” and “u” terms for link 1.

Terms from the joint space dynamic equation shown above:

$$H_{x,x} = \rho_1 \cdot L_1$$

$$H_{x,y} = 0$$

$$H_{x,\theta} = -(s_3 \cdot \rho_1 \cdot L_1^2/2 + h_{21} \cdot c_3)$$

$$H_{x,f1}(j) = -n_{21}(j) \cdot s_3$$

$$H_{y,x} = 0$$

$$H_{y,y} = \rho_1 \cdot L_1$$

$$H_{y,\theta} = c_3 \cdot \rho_1 \cdot L_1^2/2 - h_{21} \cdot s_3$$

$$H_{y,f1}(j) = n_{21}(j) \cdot c_3$$

$$H_{\theta,x} = H_{x,\theta}$$

$$H_{\theta,y} = H_{y,\theta}$$

$$H_{\theta,\theta} = \rho_1 \cdot L_1^3/3$$

$$H_{\theta,f1}(j) = a_{12}(j)$$

$$H_{f1,x}(i) = H_{x,f1}(i) - h_{23}(i) \cdot c_3$$

$$H_{f1,y}(i) = H_{y,f1}(i) + h_{23}(i) \cdot s_3$$

$$H_{f1,\theta}(i) = H_{\theta,f1}(i)$$

$$H_{f1,f1}(i,j) = a_{14}(i,j)$$

$$C_x = -\rho_1 \cdot L_1^2/2 \cdot \dot{q}_3^2 \cdot c_3 - 2 \cdot \dot{q}_3 \cdot g_{21} \cdot c_3 + h_{21} \cdot \dot{q}_3^2 \cdot s_3$$

$$C_y = -\rho_1 \cdot L_1^2/2 \cdot \dot{q}_3^2 \cdot s_3 - 2 \cdot \dot{q}_3 \cdot g_{21} \cdot s_3 - h_{21} \cdot \dot{q}_3^2 \cdot c_3$$

$$C_\theta = 0$$

$$C_{f1}(j) = 0$$

$$K_{f1,f1}(i,j) = a_{16}(i,j) + \dot{q}_3^2 \cdot (a_{15}(i,j) - a_{14}(i,j))$$

$$J_{x,x} = 1$$

$$J_{x,y} = 0$$

$$J_{x,\theta} = -(L_1 + \bar{u}_1) \cdot s_3 - \bar{w}_1 \cdot c_3$$

$$J_{x,f1}(j) = -s_3 \cdot \bar{\sigma}_1(j) - h_{13}(j) \cdot c_3$$

$$J_{y,x} = 0$$

$$J_{y,y} = 1$$

$$J_{y,\theta} = (L_1 + \bar{u}_1) \cdot c_3 - \bar{w}_1 \cdot s_3$$

$$J_{y,f1}(j) = c_3 \cdot \bar{\sigma}_1(j) - h_{13}(j) \cdot s_3$$

$$J_{\theta,x} = 0$$

$$J_{\theta,y} = 0$$

$$J_{\theta,\theta} = 1$$

$$J_{\theta,f1}(j) = \bar{\sigma}'_1(j)$$

Trigonometric terms:

$$c_3 = \cos(q_3) = \cos(\theta)$$

$$s_3 = \sin(q_3) = \sin(\theta)$$

Terms that vary with time:

$$g_{21} = \sum_{j=1}^{NF1} n_{21}(j) \cdot \dot{q}_{f1}(j)$$

$$h_{13}(i) = \sum_{j=1}^{NF1} a_{13}(i,j) \cdot q_{f1}(j)$$

$$h_{21} = \sum_{j=1}^{NF1} n_{21}(j) \cdot q_{f1}(j)$$

$$h_{23}(i) = \sum_{j=1}^{NF1} n_{23}(i,j) \cdot q_{f1}(j)$$

Constant functions of the mode shapes:

$$n_{21}(i) = \rho_1 \cdot \int_0^{L1} \sigma_1(i) \cdot dx_1$$

$$n_{21}(1) = \rho_1 \cdot L_1 \cdot 0.782991756039626$$

$$n_{21}(2) = \rho_1 \cdot L_1 \cdot 0.433935895110719$$

$$n_{21}(3) = \rho_1 \cdot L_1 \cdot 0.254425296866106$$

$$n_{23}(i, j) = \rho_1 \cdot \int_0^{L1} (L_1 - x_1) \cdot \sigma'_1(i) \cdot \sigma'_1(j) \cdot dx_1$$

$$n_{23}(1, 1) = \rho_1 \cdot 1.57087818999425$$

$$n_{23}(1, 2) = n_{23}(2, 1) = \rho_1 \cdot -0.422320389109118$$

$$n_{23}(1, 3) = n_{23}(3, 1) = \rho_1 \cdot -1.07208483410593$$

$$n_{23}(2, 2) = \rho_1 \cdot 8.64714269356121$$

$$n_{23}(2, 3) = n_{23}(3, 2) = \rho_1 \cdot 1.89005470803216$$

$$n_{23}(3, 3) = \rho_1 \cdot 24.9521133081691$$

Appendix B WORKING MODEL® 2D

The simulations achieved using the MRDS algorithm for the systems described in Chapter 8 were validated using the commercially available product Working Model® 2D (version 5.2) [72]. As shown in Figure 27, Working Model's approach to modelling structural flexibility is a lumped parameter method, where each flexible link is broken down into a user-specified number of rigid segments of equal length. The segments are connected to one another by rotational springs, where the spring constant is (initially) based on the bending stiffness (EI) entered by the user and the aforementioned number of segments. Resultant spring constants can be further modified by the user, if desired.

Working Model allows the integration scheme to be selected by the user. The choices include Euler integration or Kutta-Merson with either a fixed or variable timestep. For the validations of the MRDS results, the fixed timestep Kutta-Merson method was used.

A number of limitations were encountered when using Working Model to validate results from the MRDS coding. Acceptable accuracy was only obtained with six or more segments per link, with Working Model predictions steadily approaching MRDS results as more segments were used. However, for a system with a total of four links, like the one in Figure 27, a limitation in the number of objects (segments, springs, points, constraints, motors, etc.) allowed by Working Model made it impossible to use more than six segments per link. It is therefore likely, based on the results from simpler cases where eight or ten segments per link was attainable, that results from the more complex systems would show even greater agreement if more than six segments per link could have been used[34].

34. See footnote 33 on page 156

Another issue was the need to scale the automatically-generated spring constants. The formula used by Working Model to determine the rotational spring constant can be expressed as follows:

$$k_{wm} = \frac{EI \cdot n}{L} \tag{136}$$

where EI is the link's bending stiffness, n is the chosen number of segments per link, and L is the link length. This equation, however, is only accurate when the number of segments approaches infinity. For attainable numbers[35], however, significant errors exist when compared with theoretical predictions of standard beam bending problems. To explore this issue, several beam bending problems were simulated in Working Model, including the following: tip-loading of a cantilevered beam, deflection of a cantilevered beam under its own weight, and the rigid-body rotation of a single flexible beam.

As cantilevered mode shapes were selected for use in the MRDS coding, the spring constants established by Working Model's formula were scaled to produce better accuracy with theoretical formulas for cantilevered beam bending. Following basic theory in the deflection of a tip-loaded cantilevered beam, the "exact" formula for rotational springs used to model the flexibility of a segmented beam is as follows:

$$k_{exact} = \frac{3 \cdot EI}{n^2 \cdot L} \cdot \sum_{i=1}^{n-1} i^2 \tag{137}$$

Comparing the two expressions of rotational spring constant, the following relationship can

35. Due to Working Model's limitation on the number of objects allowed, only six to ten segments per link were attainable in the example systems simulated for this work.

be found:

$$k_{exact} = \left(\frac{3}{n^3} \cdot \sum_{i=1}^{n-1} i^2 \right) \cdot k_{wm} \tag{138}$$

where the expression in the parenthesis is the "scale factor" that should be used to adjust the Working Model constant to that for which tip deflections match tip-loaded cantilevered beam theory. Note that as the number of segments, n, approaches infinity, the scale factor approaches a value of one.

Although no single scale factor (for a specific number of segments) is appropriate to all operations (e.g. a single tip-loaded or gravitationally-loaded cantilevered beam, a single rotating beam, or multiple rotating beams), a representative value was chosen that produced that best accuracy across all types of systems. Table 34 shows the "best" scale factors used in simulations discussed in Chapter 8.

TABLE 34. Spring Constant Scale Factors Used in Working Model

Number of Segments Per Link	Scale Factor
4	0.60
6	0.72
8	0.81
10	0.83

One final limitation with Working Model was the trade-off between stability and total simulation time. For complex systems, such as the four flexible-link model, stability problems could only be avoided with a timestep of five milliseconds or smaller. However, any simulation in Working Model having greater than 20,000 timesteps was impossible to com-

plete due to limitations in available RAM. Therefore, the total duration of several simulations was limited by the timestep necessary for stability (and accuracy) and the need to end all simulations at or prior to 20,000 cycles.

Appendix C SIMULATION RESULTS FOR ALL EXAMPLE SYSTEMS

This appendix contains graphical comparisons of the results generated with the MRDS algorithm and Working Model® 2D for each of the example systems discussed in Chapter 8.

All graphical results show predictions of a specific variable over time. Consequently, in all graphs, the x-axis represents time and is measured in seconds. The y-axis gives the measure of a specific output. The possible outputs are shown below, where "i" represents the number of the joint, joint rate, or deflection. Numbers are assigned starting at the base of the system and proceeding toward the tip. For the systems with parallel robots, "L" and "R" are used to distinguish the left and right robots.

- "Angle i" = angle measured from the inertial reference frame for joint i
- "Angular Velocity i" = rotational speed of joint i around the z-axis of the inertial frame
- "(X/Y) Tip Position" = the tip position of the last link in a serial chain
- "Deflection i" = the transverse tip deflection, $\overline{w}_i = w_i(L_i, t)$ of link i
- "CV i" = the i^{th} constraint violation. For serial connections, this is the magnitude of the position difference between the contacting tip and base. For a parallel connection, it is the magnitude of the position difference between the two contacting tips. Constraint violations are only shown for MRDS results. None are available from Working Model.

The label “WM8” in the legend indicates that the results shown were generated using eight segments per link for each flexible link in Working Model. A label of “WM6” indicates six segments per link. Results labeled as “MRDS” were obtained using two cantilevered mode shapes per link with the exception of System 5c which used three cantilevered mode shapes for the first link and two for the remaining links. The term “CVS” stands for “Constraint Violation Suppression” and is used to identify results generated with or without the use of the suppression technique.

C.1 System 1: One Rotating Flexible Link

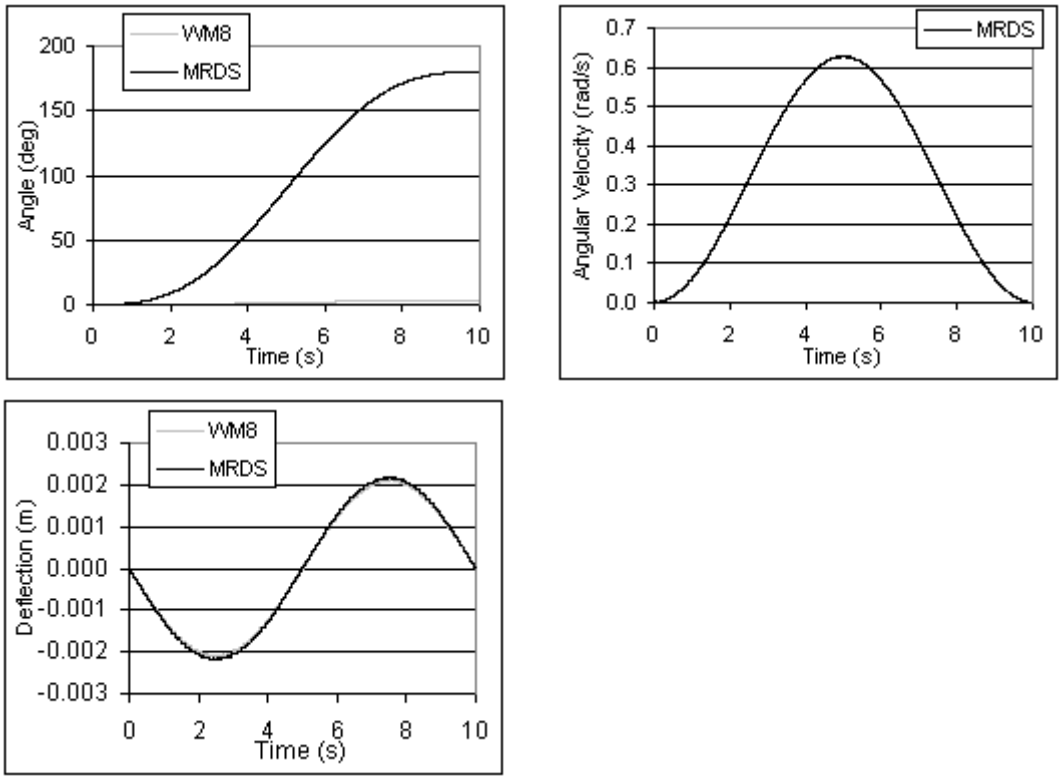

FIGURE C-1. System 1: One Rotating Flexible Link

C.2 System 2a:Two Flexible Links - Simple Open Chain Rotation

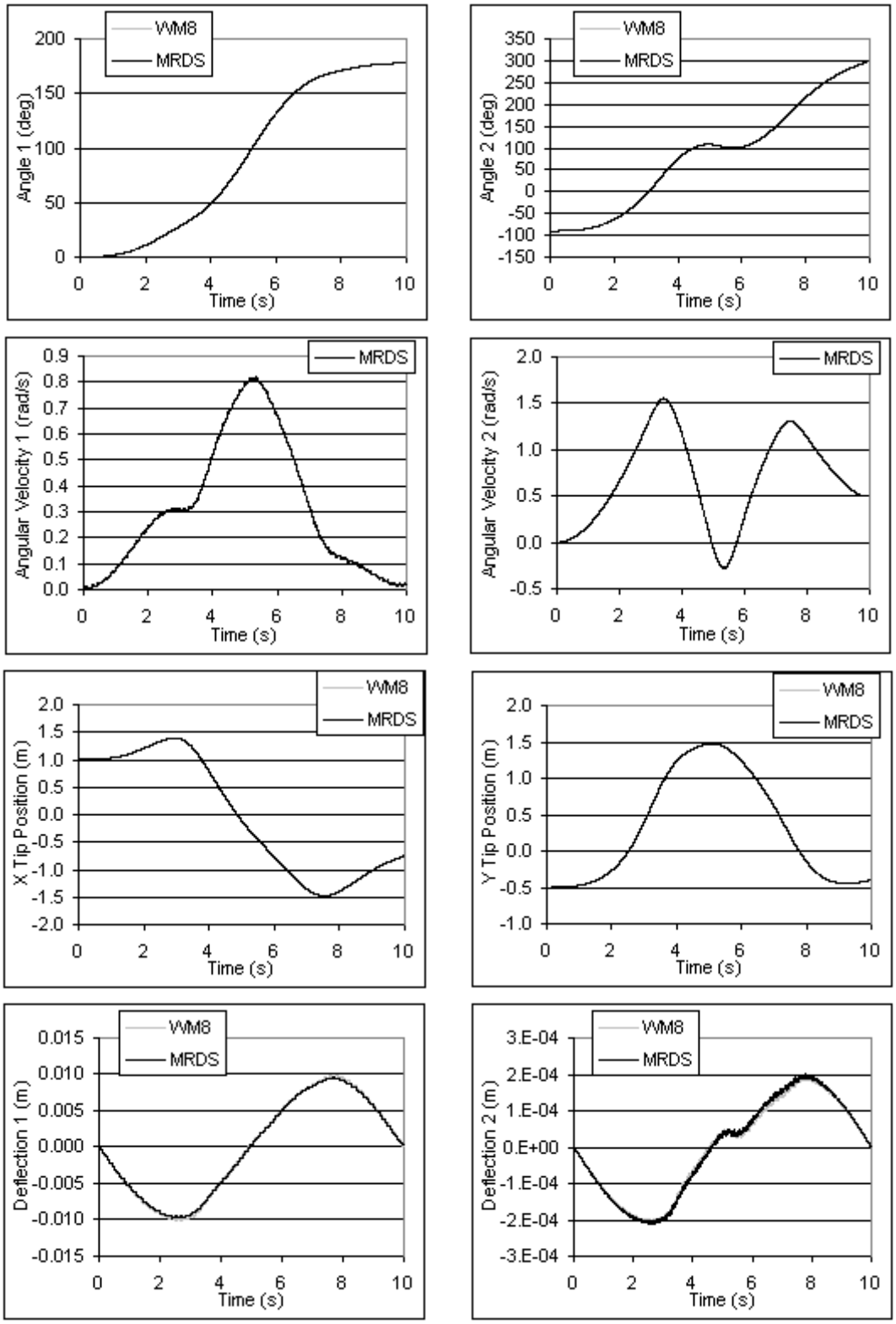

FIGURE C-2. System 2a: Two Rotating Flexible Links

C.3 System 2b: Two-Link Flexible Robot with Concentrated Tip Mass

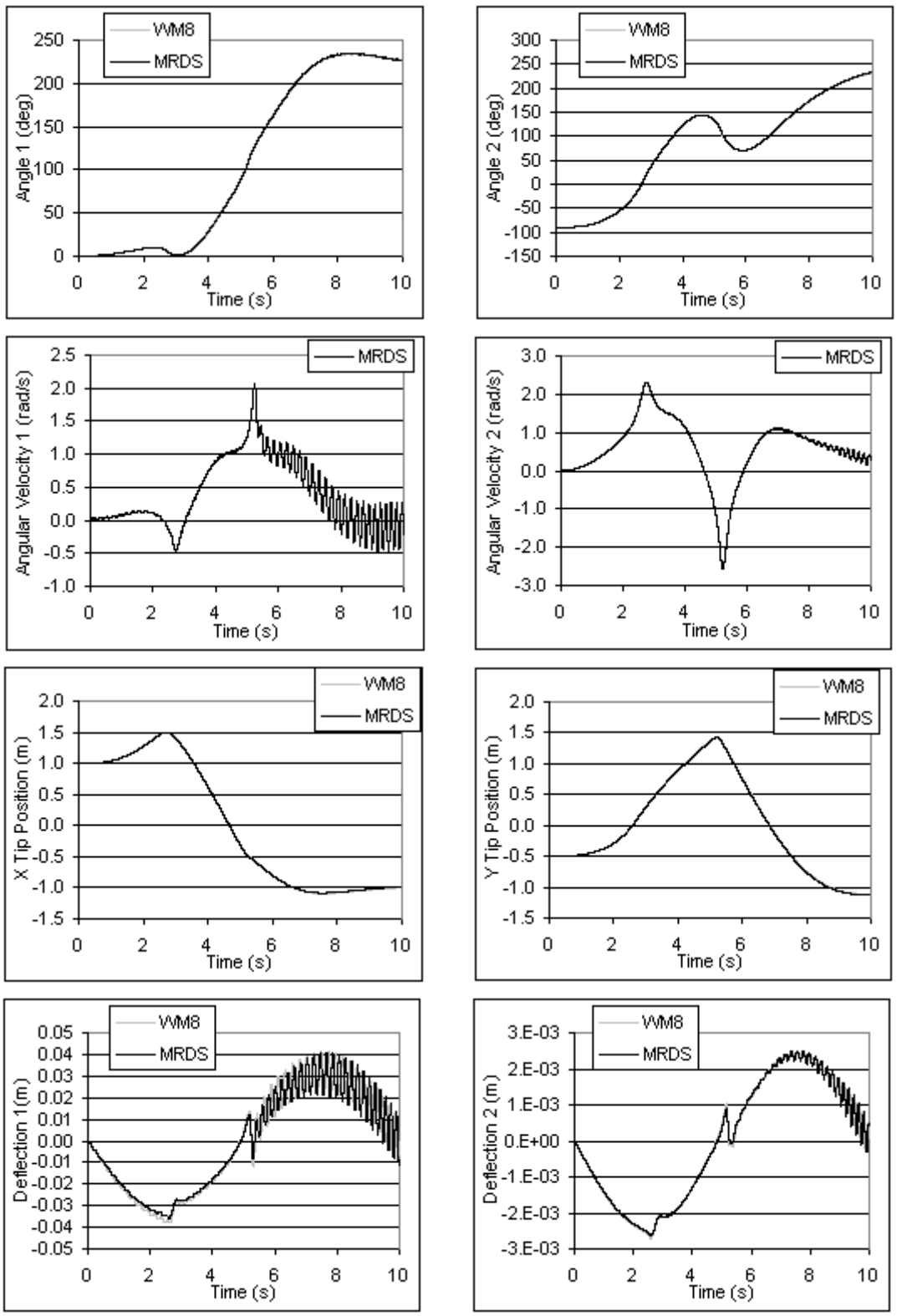

FIGURE C-3. System 2b: Two Rotating Flexible Links with Tip Mass

C.4 System 2c: Two-Link Flexible Robot - Tip Constrained to Slide

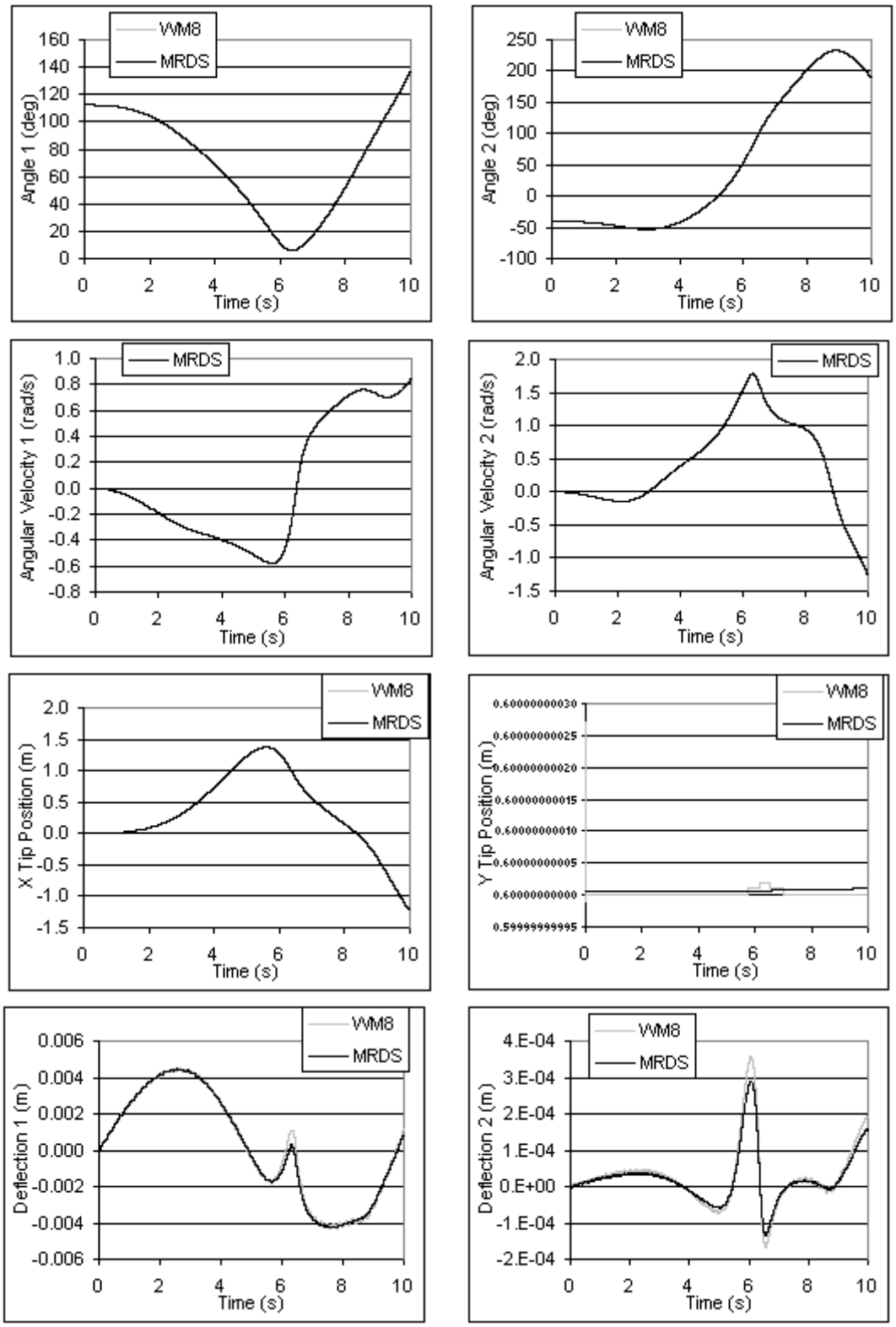

FIGURE C-4. System 2c: Two Rotating Flexible Links Constrained to Slide

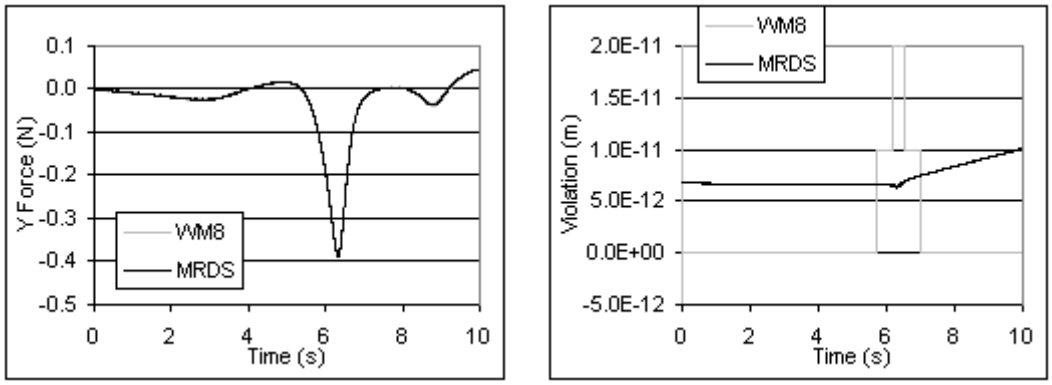

FIGURE C-5. System 2c: Two Rotating Flexible Links Constrained to Slide (cont.)

C.5 System 3: Two-Link Robot with Base Excitation

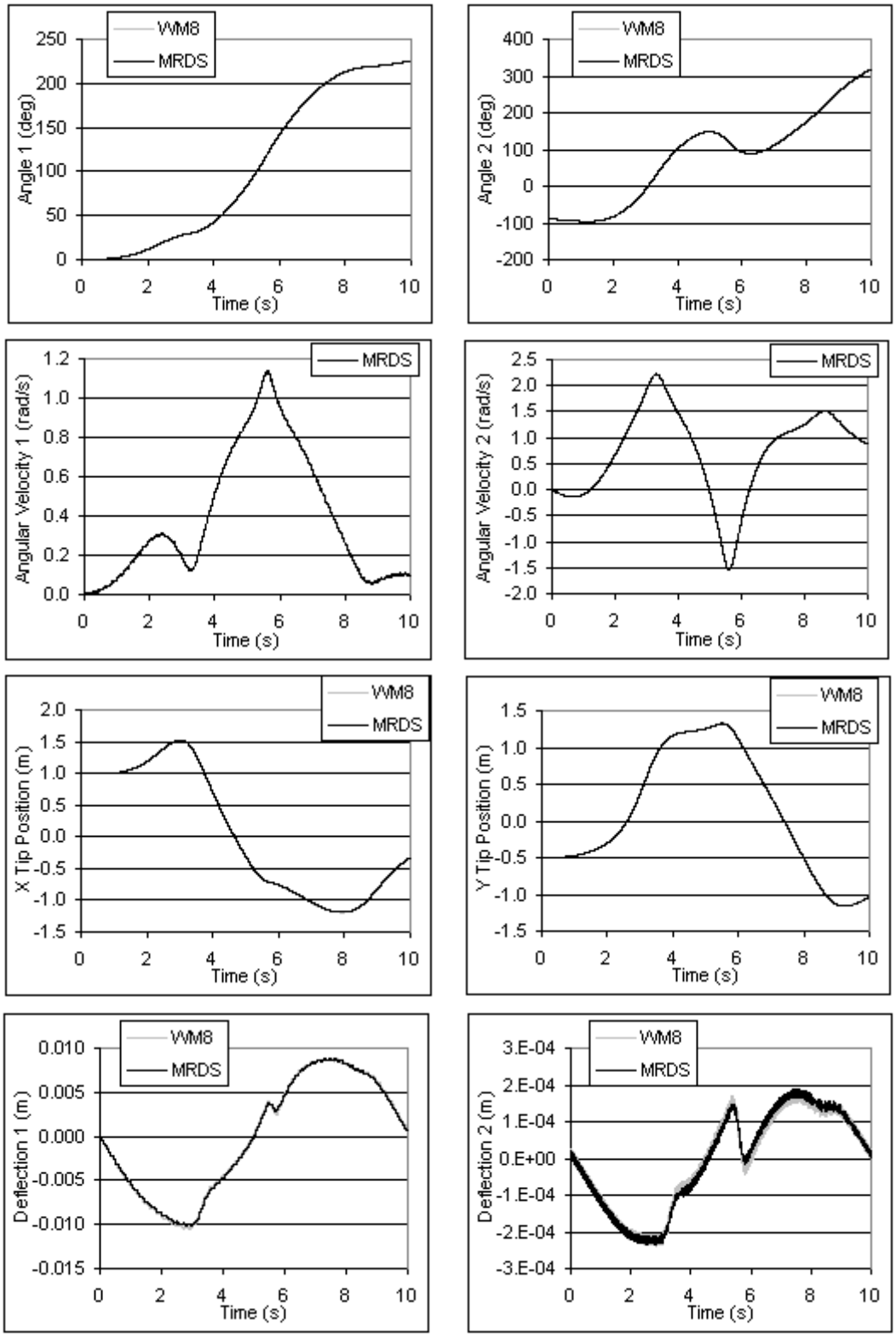

FIGURE C-6. System 3: Two Rotating Flexible Links with Base Excitation

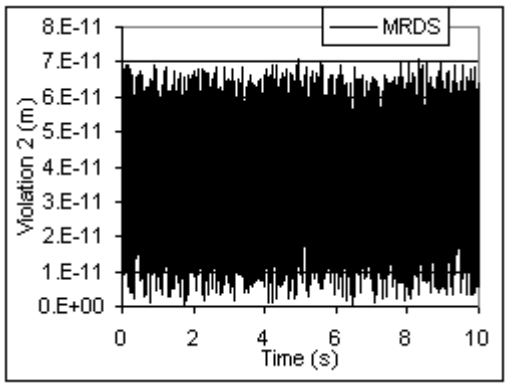

FIGURE C-7. System 3: Two Rotating Flexible Links with Base Excitation (cont.)

C.6 System 4: Four-Link Macro/Mini Robot

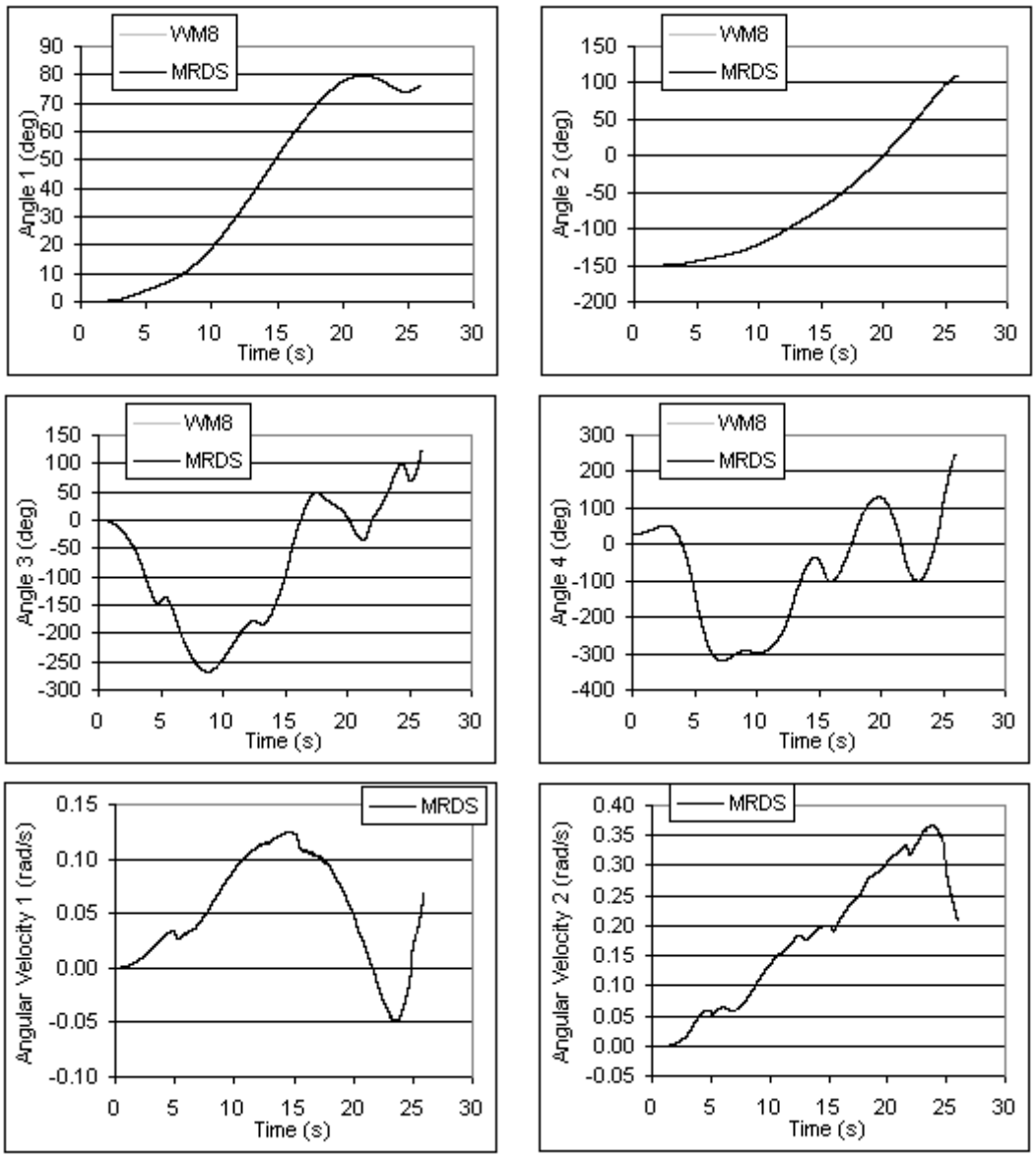

FIGURE C-8. System 4: Macro/Mini Four-link Robot

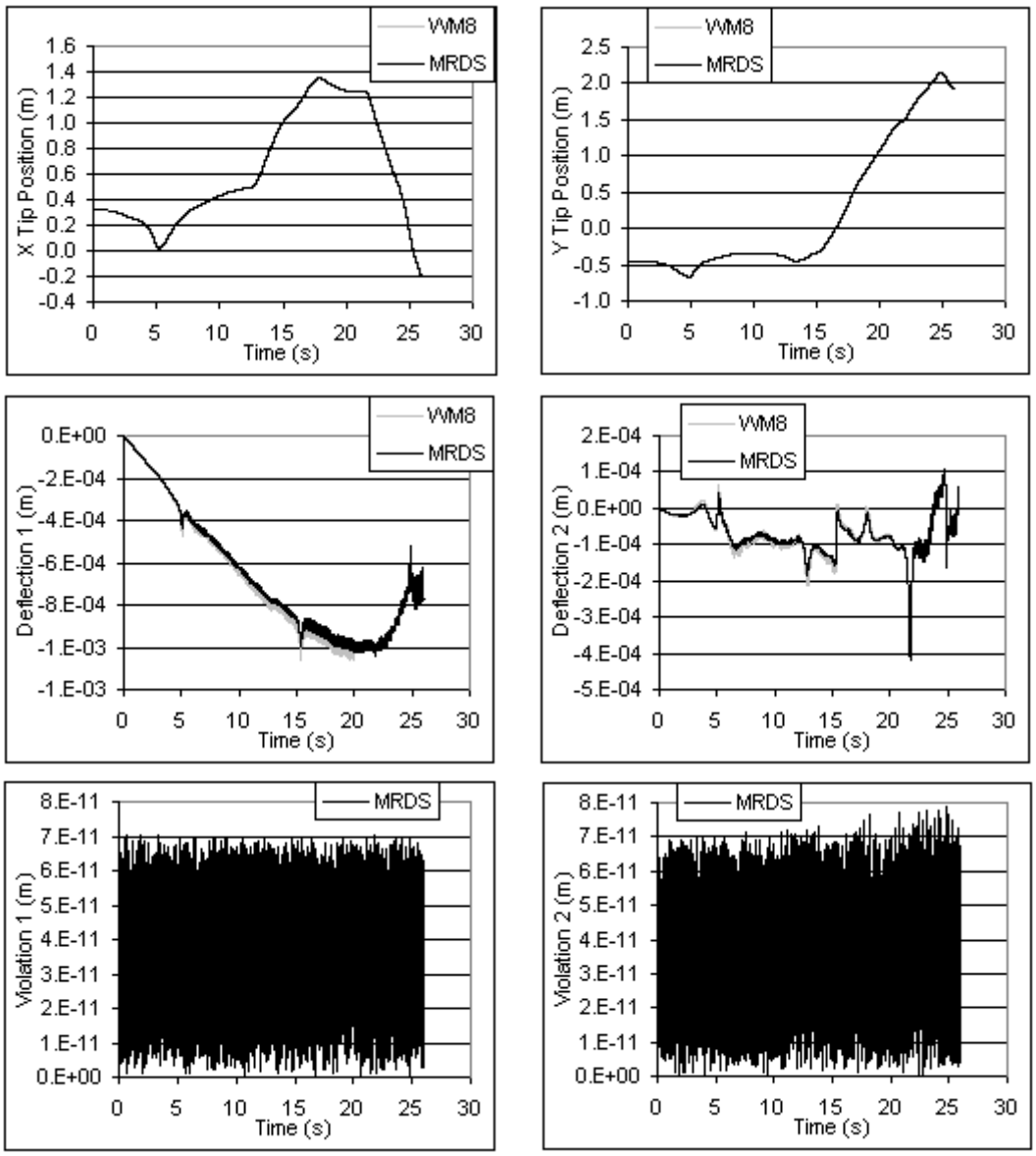

FIGURE C-9. System 4: Macro/Mini Four-link Robot (cont.)

C.7 System 5a: Four-Link Flexible Robot - Slow

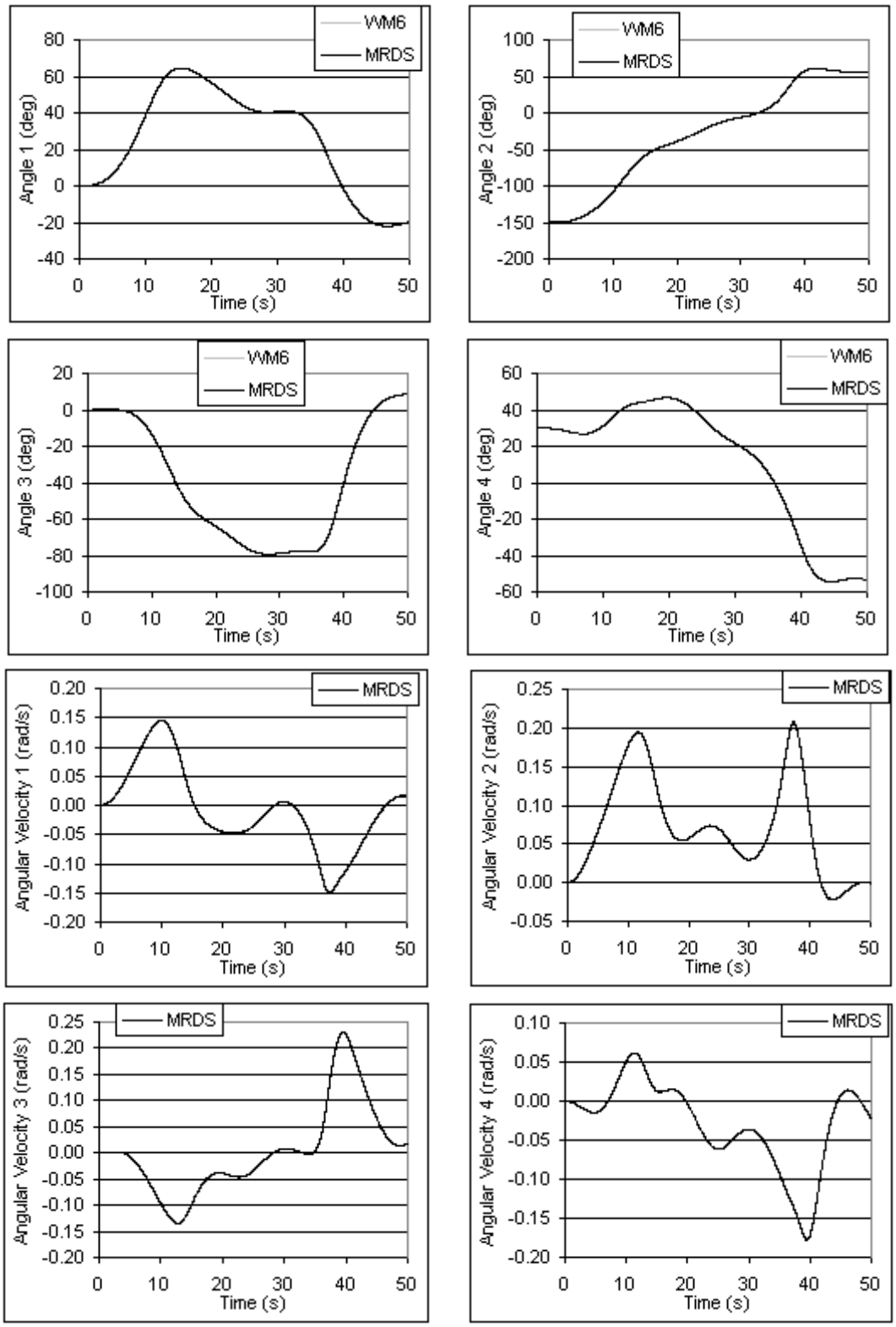

FIGURE C-10. System 5a: Four-link Flexible Robot - Slow

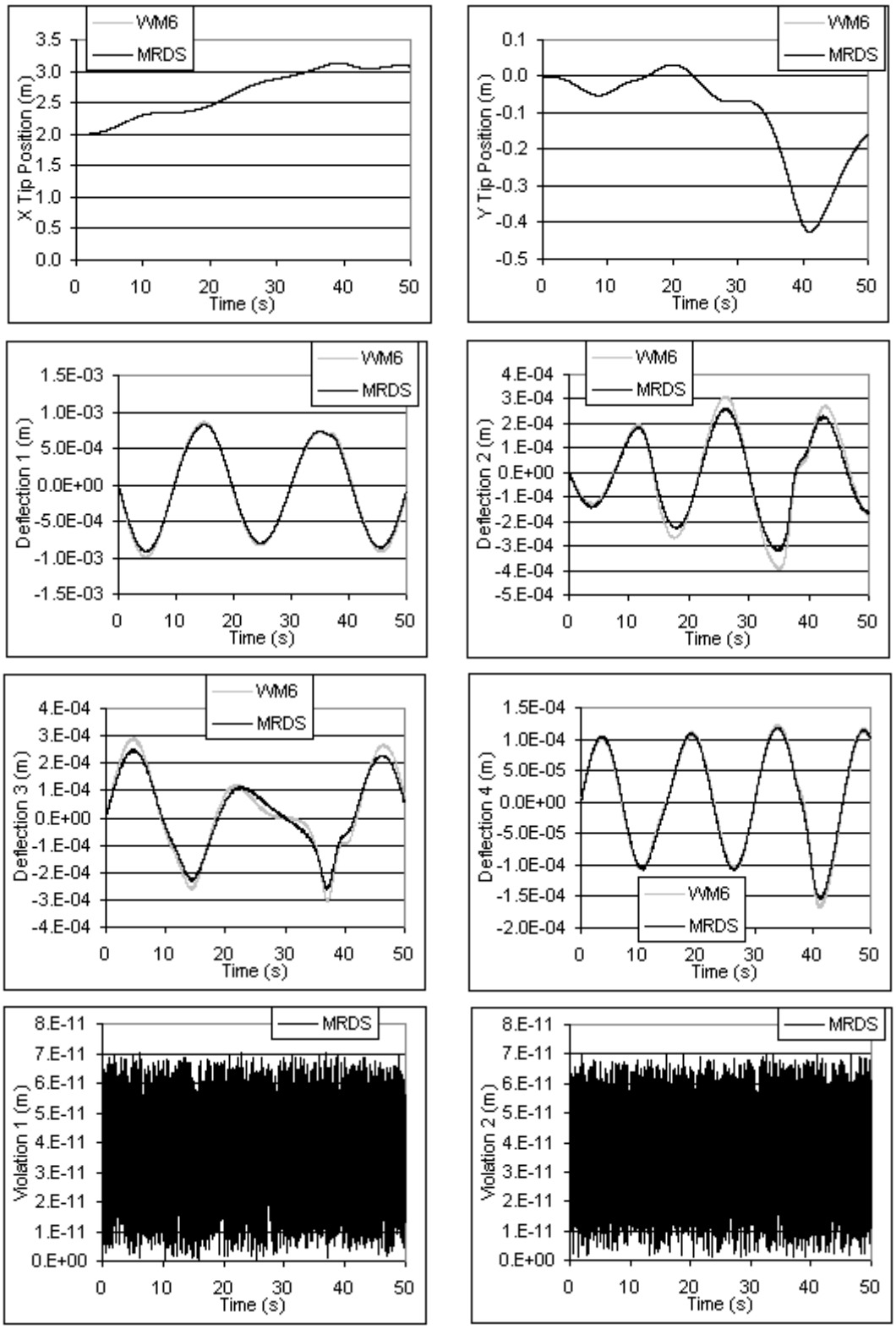

FIGURE C-11. System 5a: Four-link Flexible Robot - Slow (cont.)

C.8 System 5b: Four-Link Flexible Robot - Fast

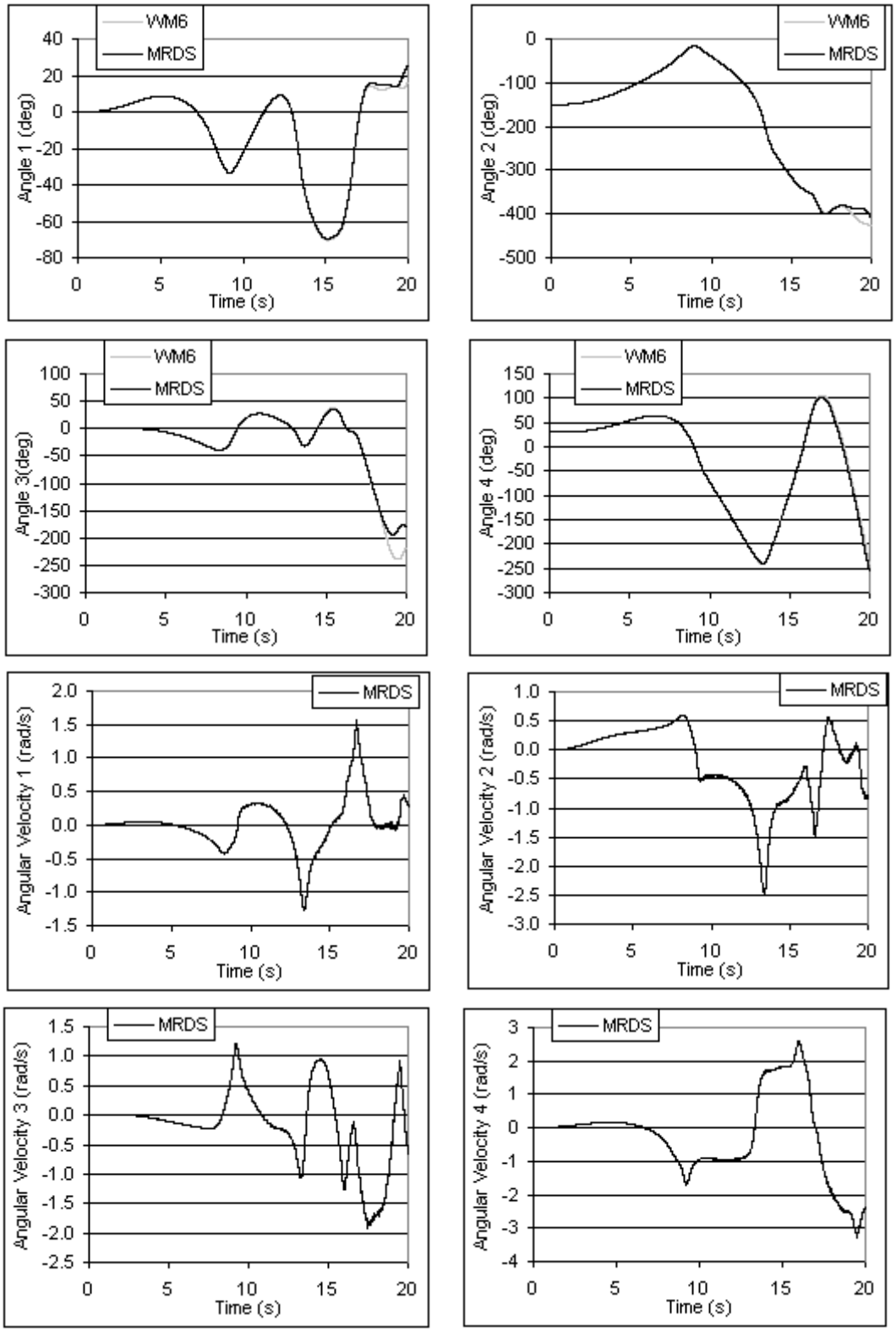

FIGURE C-12. System 5b: Four-Link Flexible Robot - Fast

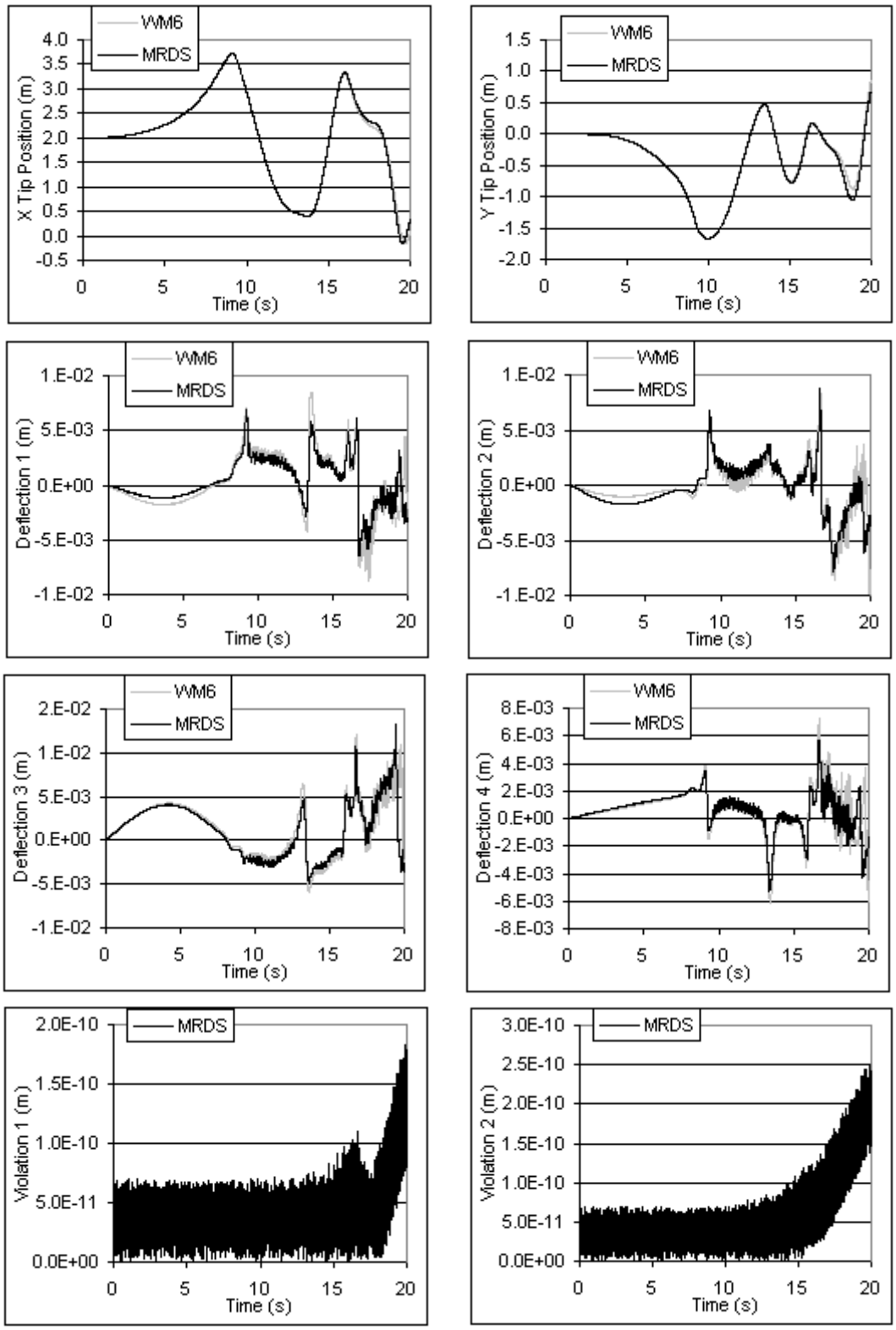

FIGURE C-13. System 5b: Four-link Flexible Robot - Fast (cont.)

C.9 System 5c: Four-Link Flexible Robot - Large Inertial Load

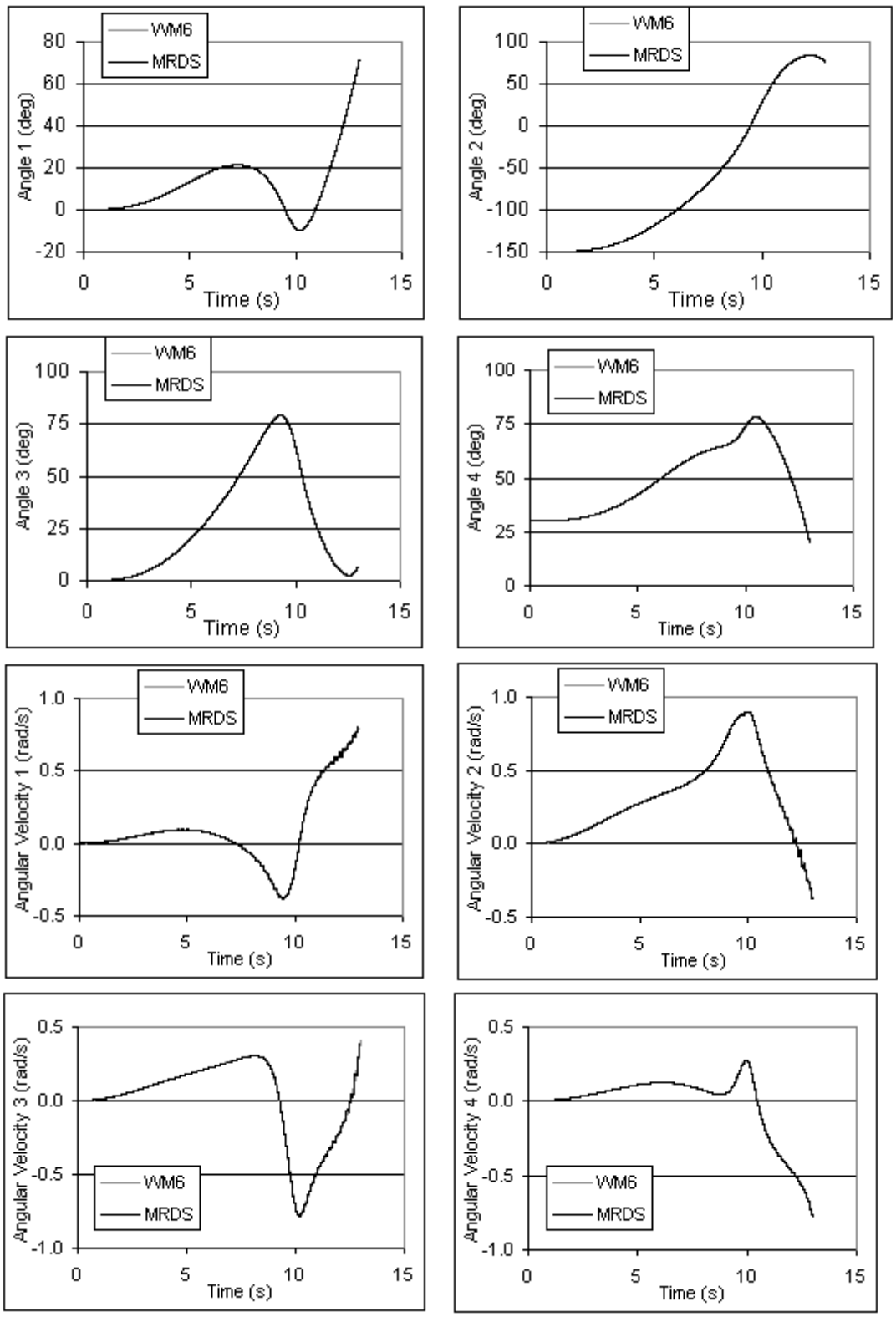

FIGURE C-14. System 5c: Four-link Flexible Robots - Large Inertial Load

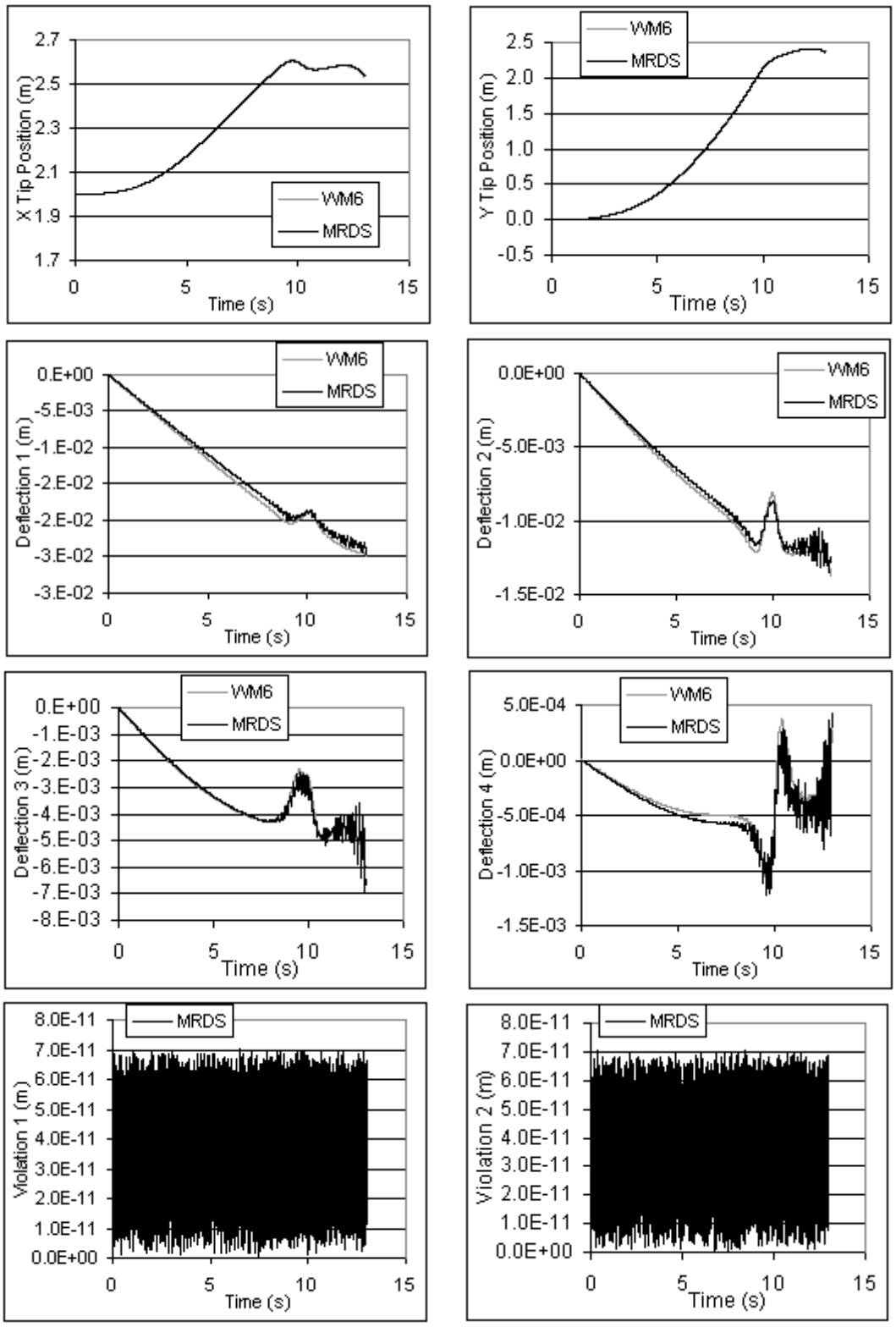

FIGURE C-15. System 5c: Four-link Flexible Robots - Large Inertial Load (cont.)

C.10 System 6a: Two Parallel Two-Link Flexible Robots - Small Deflections

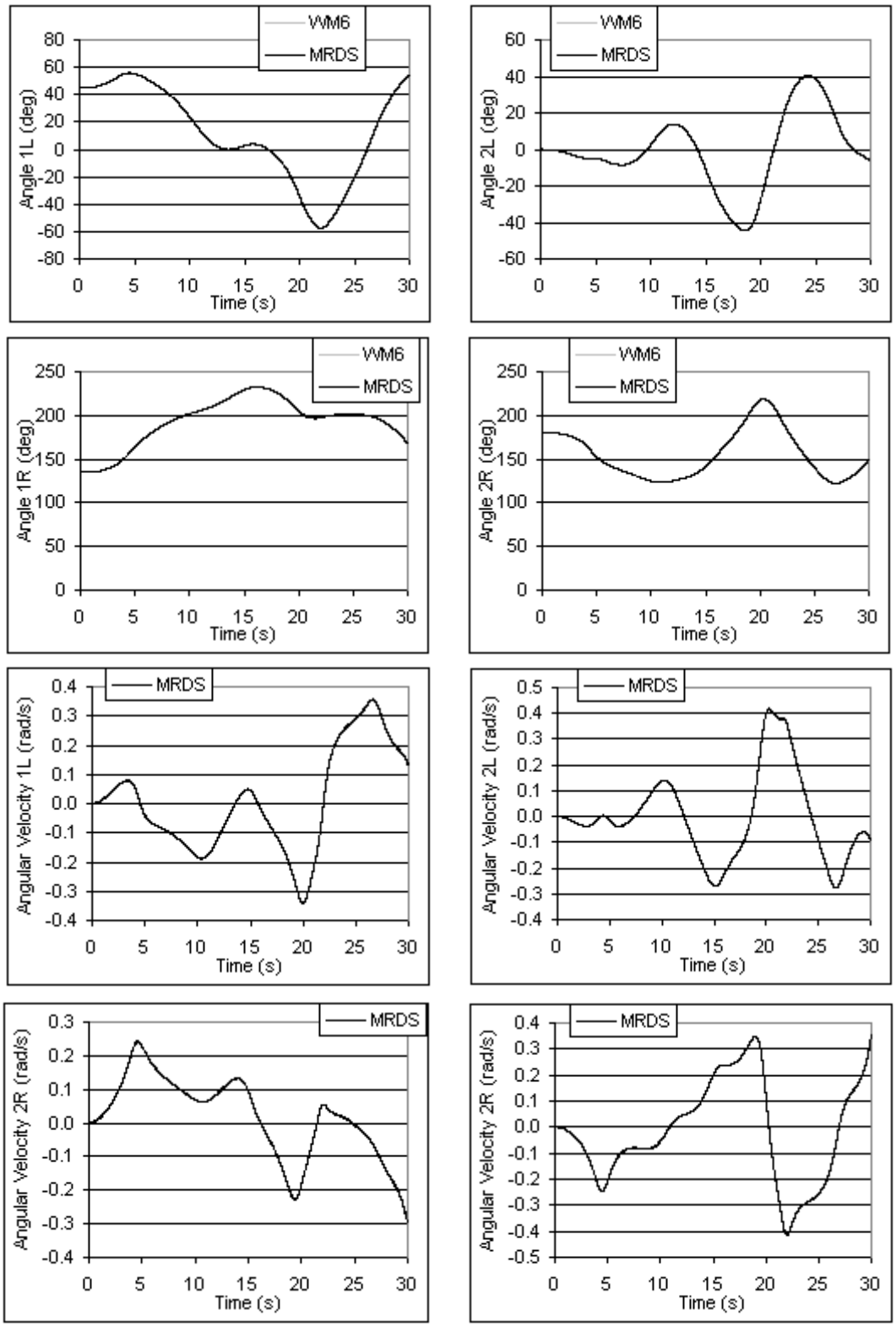

FIGURE C-16. System 6a: Two Parallel Two-link Flexible Robots - Low Forces

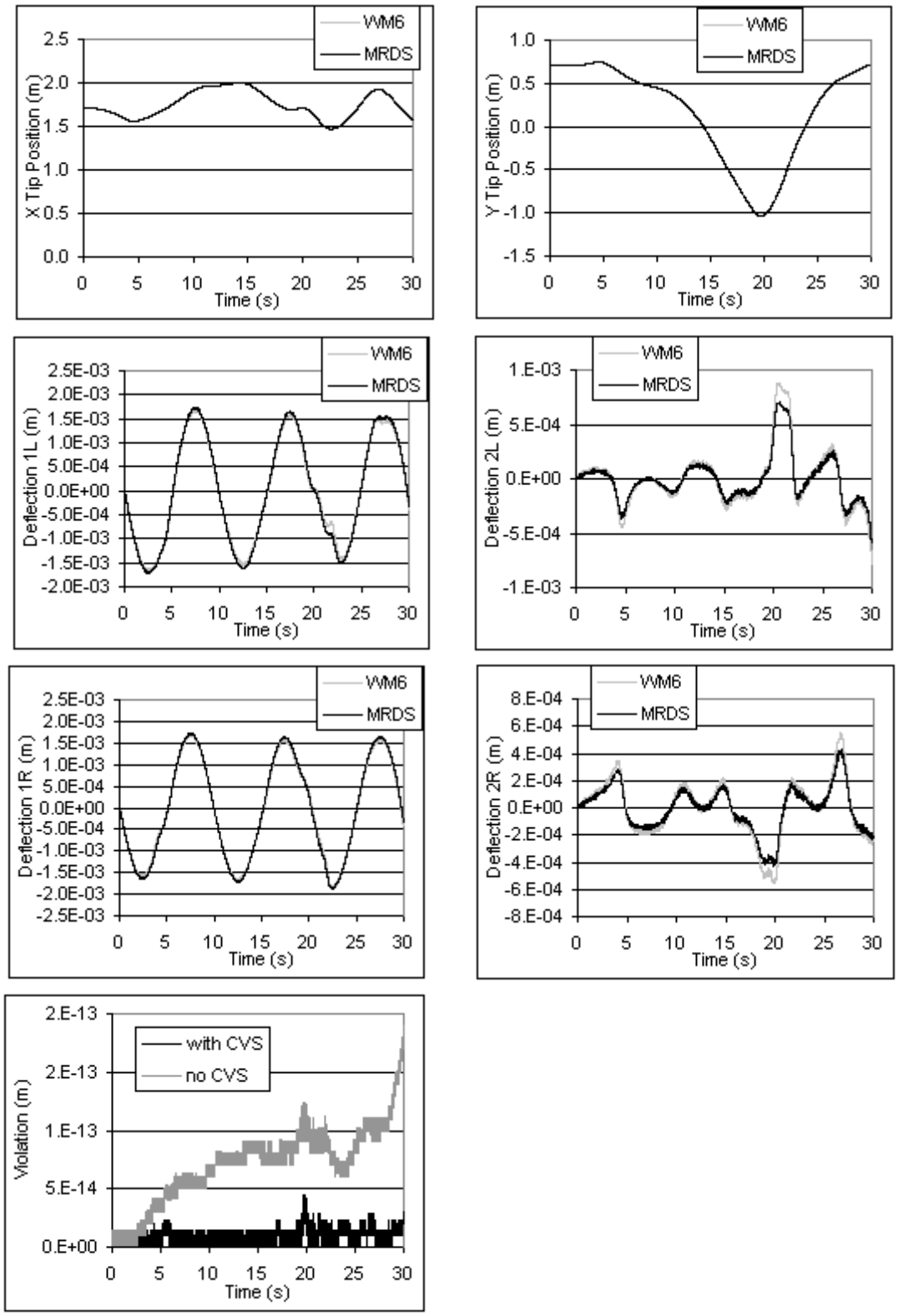

FIGURE C-17. System 6a: Two Parallel Two-link Flexible Robots - Low Forces (cont.)

C.11 System 6b: Two Parallel Two-Link Flexible Robots - Large Deflections

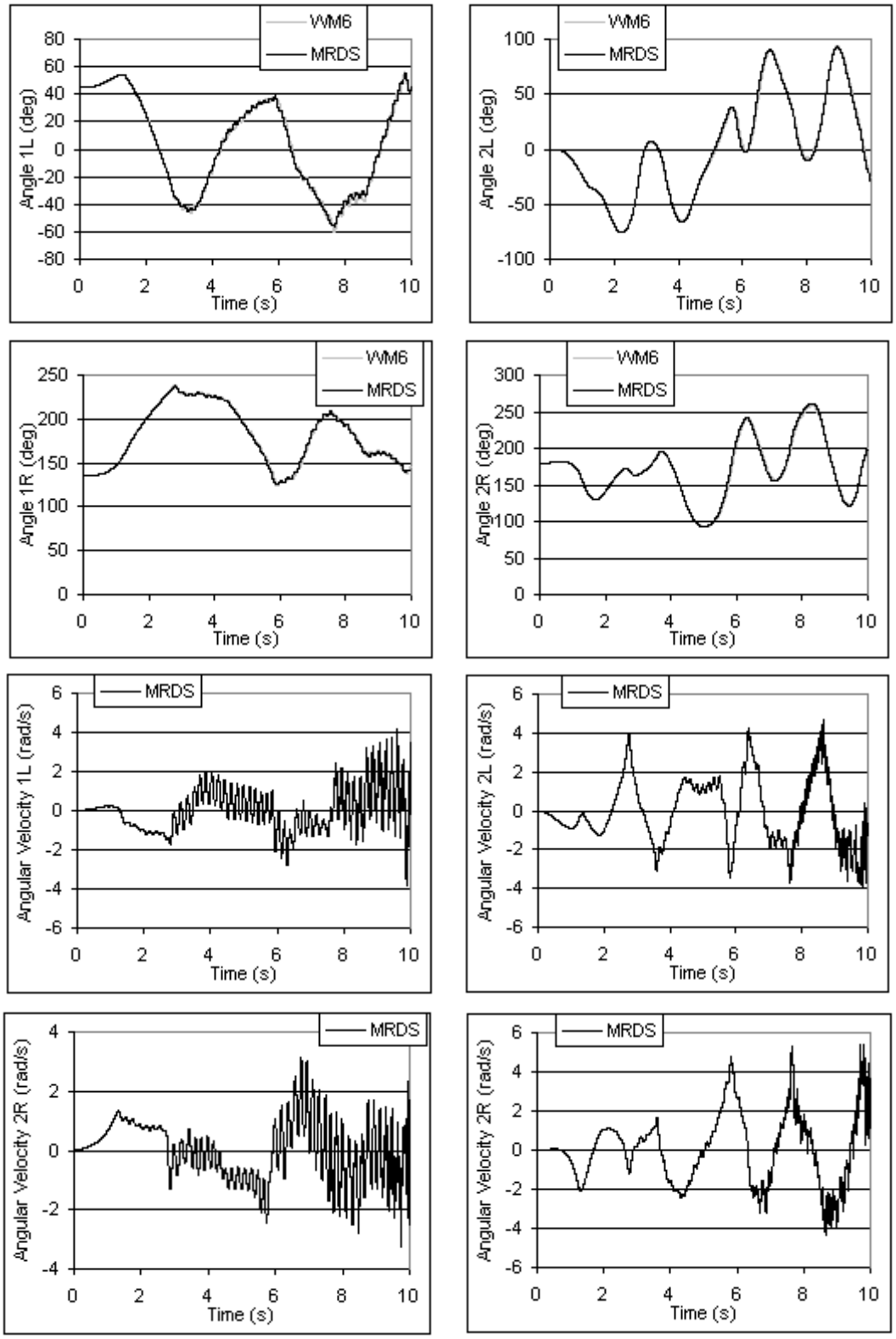

FIGURE C-18. System 6b: Two Parallel Two-link Flexible Robots - High Forces

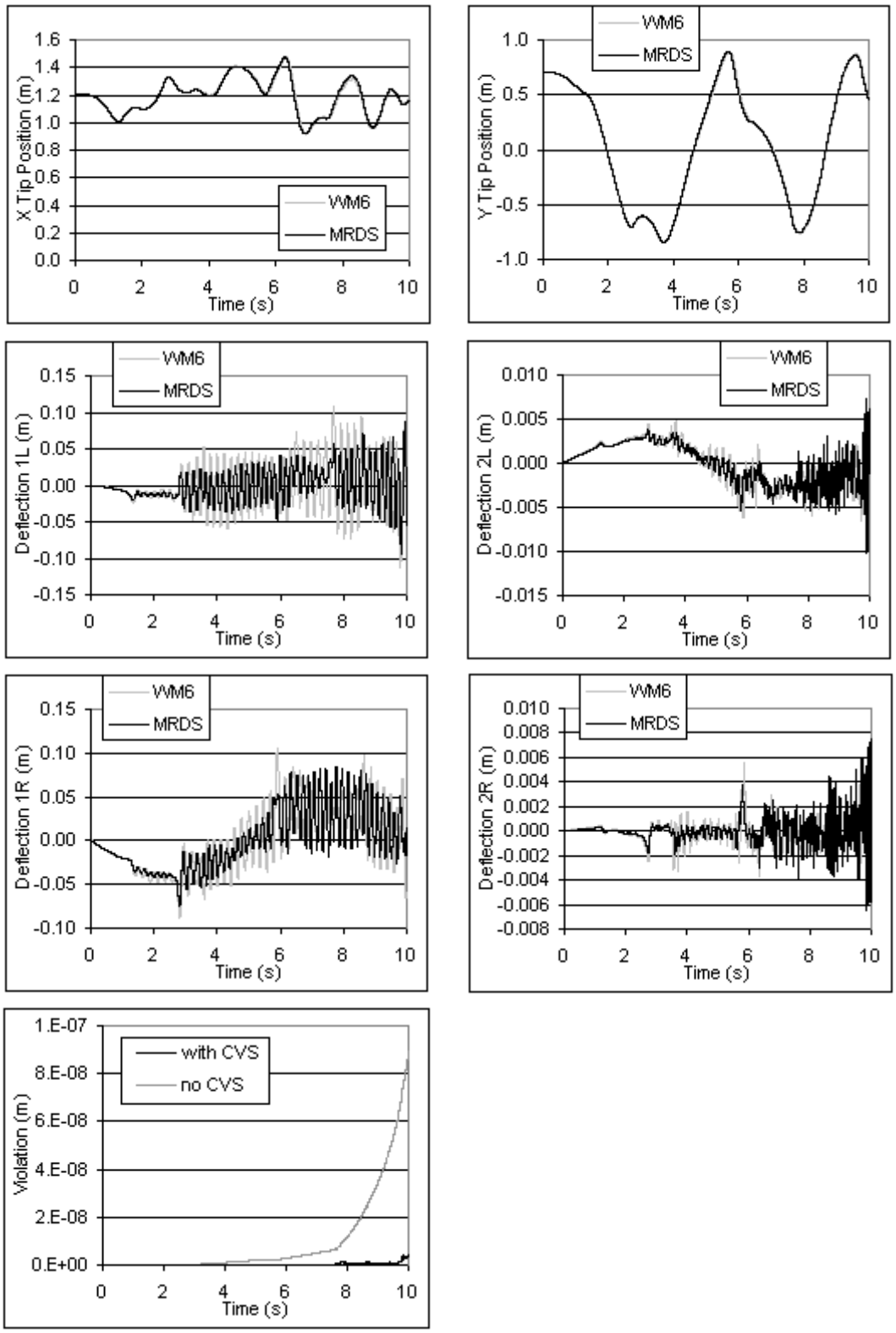

FIGURE C-19. System 6b: Two Parallel Two-link Flexible Robots - High Forces (cont.)

REFERENCES

[1] Alexander, H. L., and R. H. Cannon (1987). "Experiments on the Control of a Satellite Manipulator," *Proceedings of the 1987 American Control Conference*, Seattle, WA.

[2] Anderson, K. S. (1992). "An Order n Formulation for the Motion Simulation of General Multi-Rigid-Body Constrained Systems," *Computers and Structures*, vol. 43, no. 3, pp. 565-579.

[3] Anderson, K. S. (1993). "An Efficient Formulation for the Modelling of General Multi-Flexible-Body Constrained Systems," *International Journal of Solids and Structures*, vol. 30, no. 7, pp. 921-945.

[4] Anderson, R. J. (1993). "SMART: A Modular Architecture for Robotics and Teleoperation," *Proceedings of the 1993 IEEE International Conference on Robotics and Automation*, Atlanta, GA, vol. 2, pp. 416-421.

[5] Bayo, E. (1987). "A Finite Element Approach to Control the End-Point Motion of a Single Link Flexible Robot," *Journal of Robotic Systems*, vol. 4, pp. 63-75.

[6] Bonaventura, C. S., and K. W. Lilly (1995). "A Constrained Motion Algorithm for the Shuttle Remote Manipulator System," *IEEE Control Systems*, vol. 15, no. 5, pp. 6-16.

[7] Book, W. J. (1984). "Recursive Lagrangian Dynamics of Flexible Manipulator Arms," *International Journal of Robotics Research*, vol. 3, no. 3, pp. 87-101.

[8] Brandl, H., R. Johanni, and M. Otter (1986). "A Very Efficient Algorithm for the Simulation of Robots and Similar Multibody Systems Without Inversion of the Mass

Matrix," *Proceedings of IFAC/IFIP/IMACS International Symposium on the Theory of Robots*, Vienna, Austria.

[9] Brandl, H., R. Johanni, and M. Otter (1987). "An Algorithm for the Simulation of Multibody Systems with Kinematic Loops," *Proceedings of the IFToMM Seventh World Congress on the Theory of Machines and Mechanisms*, Sevilla, Spain.

[10] Buffinton, K. W. (1992). "Dynamics of Elastic Manipulators with Prismatic Joints" *Journal of Dynamic Systems, Measurement, and Control*, vol. 114, pp. 41-49.

[11] Chang, K., and O. Khatib (2000). "Operational Space Dynamics: Efficient Algorithms for Modeling and Control of Branching Mechanisms", *Proceedings of the 2000 IEEE International Conference on Robotics and Automation*, San Francisco, CA, pp. 850-856.

[12] Chang, P., K. C. Park, and S. Lee (2000). "An Extension to Operational Space for Kinematically Redundant Manipulators: Kinematics and Dynamics", *IEEE Transactions on Robotics and Automation*, vol. 16, no. 5, pp. 592-596.

[13] Chen, W., and V. R. Kumar (1989). "Dynamic Characteristics of Redundantly Actuated Systems in Robotics," *Proceedings of the ASME Winter Annual Meeting,* San Francisco, CA, DSC-vol. 14, pp. 131-142.

[14] Chen, Y. H. (1999). "Equations of Motion of Constrained Mechanical Systems: Given Force Depends on Constraint Force", *Mechatronics*, pp. 411-428.

[15] Damaren, C., and I. Sharf (1995). "Simulation of Flexible-Link Manipulators with Inertial and Geometric Nonlinearities", *Journal of Dynamic Systems, Measurement, and Control*, vol. 117, pp. 74-87.

[16] Ding, X., T. J. Tarn, and A. K. Bejczy (1988). "Novel Approach to the Modelling and Control of Flexible Robot Arms." *Proceedings of the IEEE Conference on Decision and Control*, Austin, TX, December, pp. 52-57.

[17] Denavit, J., and R. S. Hartenberg (1955). "A Kinematic Notation for Lower-Pair Mechanisms Based on Matrices," *Journal of Applied Mechanics*, June, pp. 215-221.

[18] Doeblin, E. (1970). *System Modeling and Response: Theoretical and Experimental Approaches*. John Wiley and Sons. New York, NY.

[19] Featherstone, R. (1983). "The Calculation of Robot Dynamics Using Articulated-Body Inertias," *The International Journal of Robotics Research*, vol. 2, no. 1, pp. 13-30.

[20] Feathersone, R. (1986). "The Dynamics of Rigid Body Systems with Multiple Concurrent Contacts," *Robotics Research: The Third International Symposium*, MIT Press, pp. 189-196.

[21] Featherstone, R., S. S. Thiebaut, and O. Khatib (1999). "A General Contact Model for Dynamically-Decoupled Force/Motion Control", *Proceedings of the 1999 IEEE International Conference on Robotics and Automation*, Detroit, MI, pp. 3281-3286.

[22] Freeman, P.S., and D. E. Orin (1991). "Efficient Dynamic Simulation of a Quadruped Using a Decoupled Tree-Structure Approach," *The International Journal of Robotics Research*, vol. 10, no. 6, pp. 619-627.

[23] Haering, W. (2002). "Continuous Shape Functions and Time Varying Boundary Conditions for Beam Dynamics," *Proceedings of the 43rd AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference*, Denver, CO, April 22-25 2002, AIAA Paper 2002-1506.

[24] Hollerbach, J. M. (1980). “A Recursive Lagrangian Formulation of Manipulator Dynamics and a Comparative Study of Dynamics Formulation Complexity,” *IEEE Transactions on Systems, Man, and Cybernetics*, vol. SMC-10, no. 11, pp. 730-736.

[25] Ider, S. K., and F. M. L. Amirouche (1989). “Nonlinear Modelling of Flexible Multibody Systems Dynamics Subject to Variable Constraints,” *ASME Journal of Applied Mechanics*, vol. 56, pp. 444-450.

[26] Kahn, M. E. (1969). *The Near-Minimum-Time Control of Open-Loop Articulated Kinematic Chains*. Stanford Artificial Intelligence Project Memo. AIM-106, December.

[27] Kane, T. R., and D. A. Levinson (1983). “The Use of Kane’s Dynamical Equations in Robotics,” *The International Journal of Robotics Research*, vol. 2, no. 3, pp. 3-21.

[28] Kane, T. R., R. R. Ryan, and A. K. Banerjee (1987). “Dynamics of a Cantilever Beam Attached to a Moving Base,” *Journal of Guidance, Control, and Dynamics*, vol. 10, no. 2, pp. 139-151.

[29] Kankaanranta, R. K., and H. N. Koivo (1988). “Dynamics and Simulation of Compliant Motion of a Manipulator,” *IEEE Journal of Robotics and Automation*, vol. 4, pp. 163-173.

[30] Khatib, O. (1987). “A Unified Approach for Motion and Force Control of Robot Manipulators: The Operational Space Formulation,” *IEEE Journal of Robotics and Automation*, vol. RA-3, no. 1, pp. 43-53.

[31] Kim, S. S., and E. J Haug (1988). “A Recursive Formulation for Flexible Multibody Dynamics, Part I: Open-Loop Systems,” *Computer Methods in Applied Mechanics and Engineering*, vol. 71, pp. 293-314.

[32] Kim, S. S., and E. J Haug (1989). "A Recursive Formulation for Flexible Multibody Dynamics, Part II: Closed-Loop Systems," *Computer Methods in Applied Mechanics and Engineering*, vol. 74, pp. 251-269.

[33] King, J. O., V. G. Gourishankar, and R. E. Rink (1987). "Lagrangian Dynamics of Flexible Manipulators Using Angular Velocities Instead of Transformation Matrices," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. SMC-17, no. 11, pp. 1059-1068.

[34] Koivo, A. J., and M. A. Unseren (1990). "Modelling Closed Chain Motion of Two Manipulators Holding a Rigid Object," *Journal of Mechanism and Machine Theory*, vol. 25, no. 4, pp. 427-438.

[35] Lee, S., and S. Kim (1993). "Efficient Inverse Kinematics for Serial Connections of Serial and Parallel Manipulators," *Proceedings of the 1993 IEEE/RJS International Conference on Intelligent Robots and Systems*, Yokohama, Japan, pp. 1635-1641.

[36] Lertpiriyasuwat, V., M. C. Berg, and K. W. Buffinton (2000). "Extended Kalman Filtering Applied to a Two-Axis Robotic Arm with Flexible Links," *The International Journal of Robotics Research*, vol. 19, no. 3, pp. 254-270.

[37] Lew, J. Y., and W. J. Book (1992). "Dynamics of Two Serially Connected Manipulators," *Proceedings of the ASME Winter Annual Meeting*, Anaheim, CA, DSC-vol. 39, pp. 17-22.

[38] Lew, J. Y., and W. J. Book (1993). "Hybrid Control of Flexible Manipulators with Multiple Contact," *Proceedings of the 1993 International Conference on Robotics and Automation*, vol. 2, pp. 242-247.

[39] Lilly, K. W. (1993). *Efficient Dynamic Simulation of Robotic Mechanisms*. Kluwer Academic Publishers, Norwell, MA.

[40] Lilly, K. W., and D. E. Orin (1991). “Efficient Dynamic Simulation of a Single Closed-Chain Manipulator,” *Proceedings of the 1991 IEEE International Conference on Robotics and Automation*, Sacramento, CA, pp. 210-215.

[41] Lockheed Engineering and Sciences Company (1992). “Shuttle Remote Manipulator System Constrained Motion Algorithm,” Technical Correspondence, (Don Hayden, LESC).

[42] Longman, R. W., R. E. Lindberg, M. F. Zedd (1987). “Satellite-Mounted Robot Manipulators - New Kinematics and Reaction Moment Compensation,” *The International Journal of Robotics Research*, vol. 6, no. 3, pp. 87-103.

[43] Luh, J. Y. S., M. W. Walker, and R. P. Paul (1980). “On-Line Computational Scheme for Mechanical Manipulators,” *ASME Journal of Dynamic Systems, Measurement, and Control*, vol. 102, pp. 69-76.

[44] Luh, J. Y. S., and Y. F. Zheng (1987). “Constrained Relations between Two Coordinated Industrial Robots for Motion Control,” *The International Journal of Robotics Research*, vol. 6, no. 3, pp. 60-70.

[45] Ma, O., K. Buhariwala, N. Roger, J. Maclean, and R. Carr (1997). “MDSF - A Generic Development and Simulation Facility for Flexible, Complex Robotic Systems,” Robotica, vol. 15, pp. 49-62.

[46] McClamroch, N. H. (1986). “Singular Systems of Differential Equations as Dynamic Models for Constrained Robot Systems,” *Proceedings of the 1986 IEEE International Conference on Robotics and Automation*, San Francisco, CA, pp. 21-28.

[47] McMillan, S., P. Sadayappan, and D. E. Orin (1994). "Efficient Dynamic Simulation of Multiple-Manipulator Systems with Singular Configurations," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 24, no. 2, pp. 306-313.

[48] Mukherjee, R., and Y. Nakamura (1992). "Formulation and Efficient Computation of the Inverse Dynamics of Space Robots," *IEEE Transactions on Robotics and Automation*, vol. 8, no. 3, pp. 400-406.

[49] Murphy, S. H., J. T. Wen, and G. N. Saridis (1991). "Simulation of Cooperating Robot Manipulators on a Mobile Platform," *IEEE Transactions on Robotics and Automation*, vol. 7, no. 4, pp. 468-478.

[50] Nakamura, Y., and R. Mukherjee (1989). "Nonholonomic Path Planning of Space Robots," *Proceedings of the 1989 IEEE International Conference on Robotics and Automation*, Scottsdale, AZ, pp. 1050-1055.

[51] Oakley, C. M., and R. H. Cannon (1989). "Equations of Motion for an Experimental Planar Two-Link Flexible Manipulator," *Proceedings of the ASME Winter Annual Meeting*, San Francisco, CA, DSC-vol. 15.

[52] Oh, S. Y., and D. E. Orin (1986). "Dynamic Computer Simulation of Multiple Closed-Chain Robotic Mechanisms," *Proceedings of the 1986 IEEE International Conference on Robotics and Automation*, San Francisco, CA, pp. 15-20.

[53] Orin, D. E., and R. B. McGhee (1981). "Dynamic Computer Simulation of Robotic Mechanisms," *Theory and Practice of Robots and Manipulators*, Warsaw, Poland, pp. 286-296.

[54] Orin, D. E., and W. W. Schrader (1984). "Efficient Computation of the Jacobian for Robot Manipulators," *The International Journal of Robotics Research*, vol. 3, no. 4, pp. 66-75.

[55] Padilla, C. E., and A. H. von Flotow (1992). "Nonlinear Strain-Displacement Relations and Flexible Multibody Dynamics," *Journal of Guidance, Control, and Dynamics*, vol. 15, no. 1, pp. 128-136.

[56] Papadopoulos, E., and S. Dubowsky (1990). "On the Dynamic Singularities in the Control of Free-Floating Space Manipulators," *Proceedings of the ASME Winter Annual Meeting*, San Francisco, CA, pp. 45-52.

[57] Press, W. H., S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery (1992). *Numerical Recipes in C: The Art of Scientific Computing*, 2nd ed., Cambridge University Press, Cambridge, London.

[58] Roberson, R. E., and R. F. Schwertassek (1987). *Dynamics of Multibody Systems.* Springer-Verlag, Berlin, Germany.

[59] Rodriguez, G., K. Kreutz-Delgado, and A. Jain (1991). "A Spatial Operator Algebra for Manipulator Modelling and Control," *The International Journal of Robotics Research*, vol. 10, no. 4, pp. 371-381.

[60] Russakow, J., O. Khatib, and S. M. Rock (1995). "Extended Operational Space Formulations for Serial-to-Parallel Chain (Branching) Manipulators," *Proceedings of 1995 IEEE International Conference on Robotics and Automation*, Nagoya, Japan, pp. 1056-1062.

[61] Silver, W. M. (1982). "On the Equivalence of Lagrangian and Newton-Euler Dynamics for Manipulators," *International Journal of Robotics Research*, vol. 1, no. 2, pp. 60-70.

[62] Stanway, J., I. Sharf, and C. Damaren (1996). “Validation of a Dynamics Simulation for a Structurally Flexible Manipulator,” *Proceedings of 1996 IEEE International Conference on Robotics and Automation*, Minneapolis, Minnesota, pp. 1959-1965.

[63] Tadikonda, S. S. K., and R. P. Singh (1991). “Dynamics of Closed-loop Systems Containing Flexible Bodies,” *AIAA Guidance, Navigation, and Control Conference*, vol. 1, pp. 1932-1938.

[64] Udwadia, F. E., R. E. Kalaba, and E. Hee-Chang (1997). “Equations of Motion for Constrained Mechanical Systems and the Extended D’Alembert’s Principle”, *Quarterly of Applied Mathematics*, vol. LV, no. 2, pp. 321-331.

[65] Uicker, J. J. (1965). *On the Dynamic Analysis of Spatial Linkages Using* 4×4 *Matrices*. Ph.D. dissertation, Northwestern University.

[66] Umetani, Y., and K. Yoshida (1989). “Resolved Motion Rate Control of Space Manipulators with Generalized Jacobian Matrix,” *IEEE Transactions on Robotics and Automation*, vol. 5, no. 3, pp. 303-314.

[67] Vafa, Z., and S. Dubowsky (1990). “The Kinematics and Dynamics of Space Manipulators: The Virtual Manipulator Approach,” *The International Journal of Robotics Research*, vol. 9, no. 4, pp. 3-21.

[68] Walker, M. W., and D. E. Orin (1982). “Efficient Dynamic Computer Simulation of Robotic Mechanisms,” *ASME Journal of Dynamic Systems, Measurement, and Control*, vol. 104, pp. 205-211.

[69] Wampler, C., K. Buffinton, and J. Shu-hui (1985). “Formulation of Equations of Motion for Systems Subject to Constraints,” *Journal of Applied Mechanics*, vol. 52, pp. 465-470.

[70] Wang, L. T., and B. Ravani (1985). "Recursive Computations of Kinematic and Dynamic Equations for Mechanical Manipulators," *IEEE Journal of Robotics and Automation*, vol. RA-1, no. 3, pp. 124-131.

[71] Wen, J. T., and Kreutz-Delgado (1992). "Motion and Force Control of Multiple Robotic Manipulators," *Automatica*, vol. 28, no. 4, pp. 729-743.

[72] Working Model® 2D - version 5.0. (2000) - User's Manual. MSC.Software Corporation.

[73] Yin, Q., and Y. F. Zheng (1993). "Coordinating Two Serially Connected Robots for Generating Large Cartesian Velocities," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 23, no. 2, pp. 554-563.

[74] Zheng, Y. F. (1989). "Kinematic and Dynamic Behavior of the Parallel and Serial Connections of Two Industrial Robots," *Proceedings of the ASME Winter Annual Meeting*, San Francisco, CA, DSC-vol. 14, pp. 143-149.

[75] Zheng, Y. F., and J. Y. S. Luh (1993). "On the Inertia Duality of Parallel-Series Connections of Two Robots in Operational Space," *IEEE Transactions on Robotics and Automation*, vol. 9, no. 6, pp. 846-854.

VITA

Clifford S. Bonaventura

January 1992	Bachelor of Science in Mechanical Engineering Minor in Engineering Mechanics The Pennsylvania State University
August 1994	Master of Science in Mechanical Engineering The Pennsylvania State University
Summer 1995,1996	Graduate Student Internship Program Sandia National Laboratories, Albuquerque, NM
1995 - 1999	Graduate Research Traineeship National Science Foundation
1999 - Present	Project Manager ZETA-TECH Associates, Inc., Cherry Hill, NJ

"Development of a Constrained Motion Algorithm for the Shuttle Remote Manipulator System (SRMS)," Masters Paper, The Pennsylvania State University, University Park, PA, 1994.

"Development of a Constrained Motion Algorithm for the Shuttle Remote Manipulator System (SRMS)," *Proceedings of the 1994 IEEE International Conference on Systems, Man, and Cybernetics*, pp. 2254-2259, 1994 (with K. W. Lilly).

"A Generalized Formulation for Simulation of Space Robot Constrained Motion," *Proceedings of the 1995 IEEE International Conference on Robotics and Automation*, pp. 2835-2840, 1995 (with K. W. Lilly).

"A Constrained Motion Algorithm for the Shuttle Remote Manipulator System," *IEEE Control Systems*, vol. 15, no. 5, pp. 6-16, 1995 (with K. W. Lilly).

"Intelligent System for Real-time Prediction of Railway Vehicle Response to Interaction with Track Geometry," *Proceedings of the 2000 ASME/IEEE Joint Railroad Conference*, ASME RTD vol. 18, pp. 31-45, 2000 (with A. M. Zarembski and J. W. Palese).

"An O(N) Modular Algorithm for the Dynamic Simulation of Robots Constrained by a Single Contact," submitted in May 2002 to *IEEE Journal of Systems, Man, and Cybernetics*, (with K. W. Jablokow).