CHAPTER I

Introduction

1.1 Motivation

For decades the single chain robotic manipulator has been the primary tool of manufacturers and researchers alike. Its capabilities are limited both by the simplicity of its mechanical design, and in many cases, its conservative control algorithms. Currently, robotics is being applied to ever more complex problem domains which require more complex systems and control algorithms that push the limits of the systems' mechanical capabilities. These new systems can have multiple chains, mobile bases, and experience environments that are vastly different from the designer's workbench.

Multiple manipulator systems for factory settings are being developed that must cooperate to lift larger loads or perform more complex manipulation tasks for assembly. These systems are characterized by fixed bases, and closed kinematic loops. Telerobotic systems for applications such as hazardous waste cleanup are also under development. These systems have mobile bases with one or more manipulators. Development of multilegged vehicles is also being pursued for traversal over rugged terrain.

Use of these complex robotic systems is becoming increasingly desirable in the underwater environment for a range of applications such as biological and geophysical research, construction, salvage, rescue and recovery, and maintenance and repair of man-made underwater structures. Such applications may require manipulators mounted on mobile platforms with sophisticated control systems. Multilegged vehicles such as the one shown in Figure 1 are also under development for surveying and construction applications [1]. The importance of these *unmanned* underwater robotic vehicle (URV) systems for marine research and subsea development continues to grow because their manned counterparts are much more expensive to develop and maintain [2]. This increase in use has brought about a concomitant need for accurate simulations of these systems [3].

With the desire for increased capabilities and performance of these robotic systems comes an increase in the cost of their design, development, and operation. Dy-

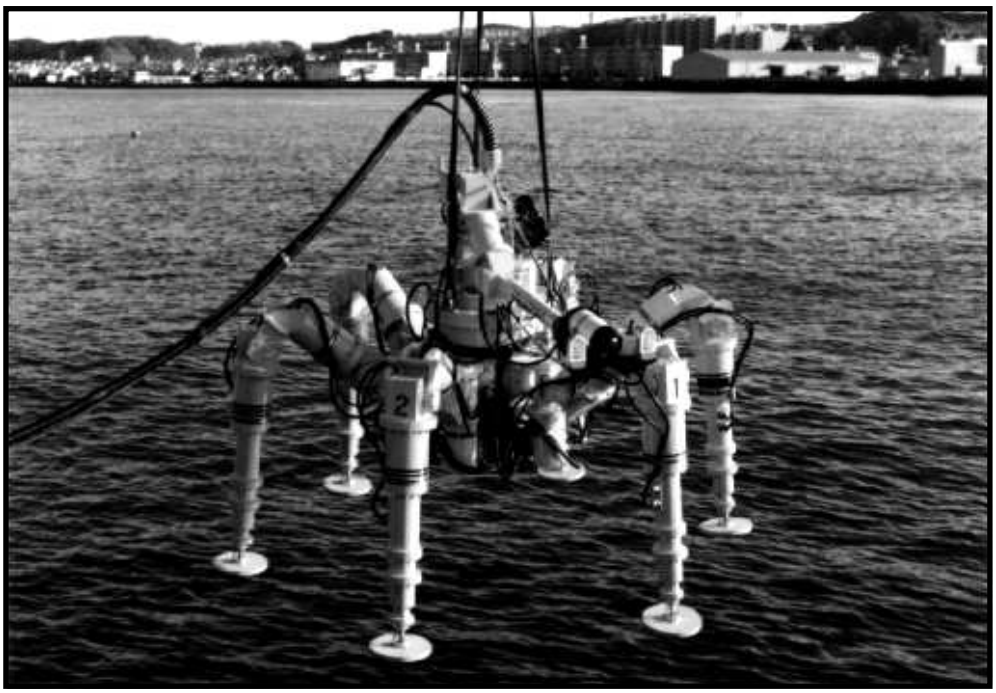

Figure 1: The *Aquarobot* hexapod [1].

namic simulation has proven to be a cost effective tool for single chain manipulators where ten years of research has led to the development of a number of very efficient algorithms and commercial software packages. Researchers continue to search for more efficient dynamics algorithms for complex land-based systems [4, 5, 6, 7], but the application of these ideas has not been applied to URV systems. The work presented here contributes to this research effort.

In both of these domains, the use of dynamic simulation can save significant time and money during the mechanical and control design, test and evaluation phases for these systems. Early in the design process, the use of simulation to perform a thorough dynamic analysis can reduce the need for costly prototypes by eliminating many candidate designs. This not only saves on cost but also the time needed to construct successive prototype generations. In addition, simulators can aid in the design of control algorithms. By using the simulated robotic system to test such algorithms, potentially damaging instabilities due to algorithm errors are eliminated, and risks encountered when the control system is finally implemented in hardware are reduced.

Until only very recently, dynamic simulation of robotic systems has been consid-

ered an off-line problem, and this is adequate for the design applications mentioned above. With an increase in computing capabilities and the development of more efficient algorithms, real-time dynamic simulation of these systems is also within reach. As a result, interest in on-line simulation and its applications is also increasing, such as hardware- and human-in-the-loop operation. With hardware-in-the-loop simulation, control system hardware can also be tested off-line by interfacing it to a computer performing a real-time dynamic simulation of the system [8]. This further reduces the risk to the robotic systems, and is especially important in URV development because the underwater environment is unstructured and unpredictable.

Human-in-the-loop applications can also be implemented when a real-time simulation is coupled with a realistic 3-D graphical display of the system. One such application is a simulator to train operators in the use of the robotic systems before allowing hands-on training. This is analogous to the use of flight simulators to train aviators. Another application can be found in the teleoperation of space-based and untethered underwater vehicles. Because significant delays (speed-of-light, acoustic propagation) occur in communications with such vehicles, human control is significantly degraded. By providing the teleoperator with a simulated display of the system, on-line with no delay, enhanced performance of human-machine interaction can be realized [9, 10].

Beyond real-time computational rates, the application of *super*-real-time simulation has also been considered. In this scheme, seconds of motion trajectory need to be simulated in milliseconds [11]. It is desired in some applications such as in advanced control schemes where trajectory planning is used. This has been proposed for multilegged vehicles [12] where prediction of the action of the present control is used to ensure safety and stability along a desired trajectory.

1.2 Objectives

For dynamic simulation, the primary problem to be solved is called forward or direct dynamics. This problem requires the determination of motion of the system (accelerations of all of the bodies) given the set of forces acting on the system (external forces and internal joint actuator torques/forces). Once the accelerations are known, numerical integration is used to determine the updated state (positions and velocities) of the system. For multibody robotic systems, the dynamic equations of motion are complex and their computation dominates in the simulation. Therefore, they are the primary obstacle to achieving the required computational rates for the on-line applications. This problem is compounded by increases in the structural and task complexity found in the systems considered in this dissertation, which have multiple chains and redundant numbers of degrees of freedom, operating at higher speeds,

with closed kinematic loops, and may operate in the underwater environment.

This dissertation focuses on two classes of multiple chain robotic systems. They are comprised of one reference member which interacts with a number (m) of supporting serial chains. These are referred to as *simple closed chain mechanisms* where the removal of the reference member breaks all closed loops in the system [13]. For Type 1 systems, the ends of each chain are attached to the reference member through an actuated joint. The chains are then closed through contact of the opposite end of the chain with the ground. The typical example of a Type 1 system is the legged vehicle. In Type 2 systems, the chains are attached to a common base (e.g., the ground) through an actuated joint, and they are closed through an unpowered contact or grasp of the reference member (e.g., a load). Examples of this are multiple cooperating manipulators or dextrous hands.

In an effort to achieve real-time rates, the development of computationally efficient dynamic simulation algorithms for these systems is the primary objective of this research. A number of efficient algorithms have been previously developed to simulate the dynamics of robotic mechanisms. The most efficient approaches to this problem employ recursive equations. Most of these methods are based on either the Composite Rigid Body (CRB) method developed by Walker and Orin (Method 3, [14]) or the Articulated Body (AB) method developed by Featherstone [15]. Since their inception, improvements have been made to the computational efficiency of these algorithms. Balafoutis and Patel [16] have developed a most efficient implementation of the CRB algorithm, and Brandl, Johanni, and Otter [17] have developed an efficient implementation of the AB algorithm.

The AB approach has an advantage because its computational requirements grow linearly with the number of degrees of freedom, whereas the CRB method has cubic complexity. For typical serial chain manipulators, the AB algorithms are accepted as the most efficient. However, if the number of links is small enough or multiple, closed chain systems are being considered, algorithms based on the CRB method can be more efficient. Therefore, the main objective of this dissertation, is to develop new efficient algorithms based on these approaches and determine which is the most efficient for the systems of interest. In the process, both approaches will be expressed using uniform notation and equivalent methods of modeling these mechanisms so that comparisons are fair and the correct conclusions can be reached.

Parallelization of these algorithms is another technique which can be used to increase the computational rates of the simulation. Fine-grain parallel algorithms for single serial chains have been developed in most previous work [18, 19, 20, 21, 22, 23]. These approaches, while designed to perform well on special-purpose parallel architectures, are much less efficient when implemented on the general-purpose parallel systems widely available today. These systems typically have a limited number of loosely coupled processors that are not designed to handle the large amounts of inter-

processor communication and synchronization required by the fine-grain tasks. One form of parallelism in the simulation explored in this work is related to the structural parallelism in the multi-chain mechanisms and is called *spatial parallelism*. With this, the forward dynamics computation is parallelized. The second, is *temporal* parallelism which is implemented with parallel numerical integration. They can also be combined to further increase the computational rates.

The extension of these algorithms to space-based robots with rigid links is straightforward – requiring only the removal of gravity. The use of robotic systems in an underwater environment is relatively new, and efficient multibody dynamics algorithms have not been developed for these systems. More importantly, the application of existing algorithms for land-based systems to URVs is not straightforward because of complex hydrodynamic effects. Therefore, an additional objective for development of a simulator for these systems is the study of the hydrodynamic forces that are exerted on submerged rigid bodies. Special effort must be made to express these effects using a notation consistent with existing robotics algorithms, and a detailed explanation of how to incorporate these effects is given.

This work culminates in the development of a software package in C++, called **DynaMechs***, that is capable of performing the dynamic simulation for a large class of tree-structured robotic systems having star topologies. These systems range from a single chain with a fixed base to multiple chains with a mobile base. Object oriented design (OOD) techniques are explored to aid in the implementation of an efficient algorithm that is capable of simulating fixed and mobile bases, single and multiple chains with any number of links, and various link types. Techniques such as object hierarchies, encapsulation, inheritance, and polymorphism are employed, and issues such as code reuse and maintenance are discussed. The result is a flexible, efficient, and maintainable package.

1.3 Spatial Notation

Since the dynamics algorithms discussed in this dissertation are complex, the purpose of this section is to introduce some of the notation employed. To begin, the convention for labeling the rigid bodies in simple closed chain mechanisms is presented. First, the reference member is indicated by an r. The links in a serial chain are numbered in order from 1, nearest the base, to N, which is farthest from the base. Note that for Type 1 systems, link 1 is attached through an actuated joint to the reference member, while link N is in contact with the reference member for Type 2 systems. With multiple chains these links are labeled with ordered pairs where the first is the chain number, from 1 to m, and the second is the link number.

*Pronounced dynamics, and stands for dynamics of mechanisms.

These labels are used with spatial notation to specify the dynamic and kinematic quantities associated with each rigid body in the system.

Traditionally, robot dynamics and kinematics algorithms have used three-dimensional angular and linear components of velocity, acceleration and force quantities separately, which leads to overly complex expressions for these algorithms. The purpose for the development of spatial notation was to alleviate some of this complexity and make the equations easier to comprehend. The notation used in this dissertation is based on that used by Lilly and Orin [24]. With this approach, the corresponding angular and linear components of velocity, acceleration, and force are combined into six-element vectors [15, 25]. For example, the spatial velocity of link i is given as follows:

$$\mathbf{v}_i = \begin{bmatrix} \boldsymbol{\omega}_i \\ \boldsymbol{v}_i \end{bmatrix}, \tag{1.1}$$

where $\boldsymbol{\omega}_i$ is the angular velocity of link i expressed in the ith coordinate system, and $\boldsymbol{v}_i$ is its linear velocity. Note that the convention in this dissertation is to use an italic bold variable, such as $\boldsymbol{v}$, to refer to a Cartesian (three-dimensional) vector representing either a rotational or translational quantity, and a block bold variable, such as $\mathbf{v}$, to refer to a spatial (six-dimensional) vector. Similarly, its spatial acceleration is given as follows:

$$\mathbf{a}_i = \begin{bmatrix} \dot{\boldsymbol{\omega}}_i \\ \boldsymbol{a}_i \end{bmatrix}. \tag{1.2}$$

where $\dot{\boldsymbol{\omega}}_i$ is its angular acceleration and $\boldsymbol{a}_i$ is its linear acceleration. A spatial force exerted onto link i is given as:

$$\mathbf{f}_i = \begin{bmatrix} \boldsymbol{n}_i \\ \boldsymbol{f}_i \end{bmatrix}, \tag{1.3}$$

where $\boldsymbol{n}_i$ is a moment and $\boldsymbol{f}_i$ is the three-dimensional force vector.

The 6×6 spatial transformation matrix, ${}^{i}\mathbf{X}_{i-1}$, is used to transform spatial vectors between coordinate systems that are fixed to each rigid body. For example, a spatial velocity is transformed to a link's outboard neighbor as follows:

$${}^{i}\mathbf{v}_{i-1} = {}^{i}\mathbf{X}_{i-1}\, {}^{i-1}\mathbf{v}_{i-1}, \tag{1.4}$$

where ${}^{i-1}\mathbf{v}_{i-1}$ is the velocity of link $i-1$ expressed in its own coordinate system, and ${}^{i}\mathbf{v}_{i-1}$ is the same vector expressed with respect to link i's coordinate system. For brevity, the leading superscript is often omitted from ${}^{i-1}\mathbf{v}_{i-1}$ when it is the same as the subscript. The transpose of the spatial transformation matrix is used to transform spatial forces in the other direction:

$${}^{i-1}\mathbf{f}_i = {}^{i}\mathbf{X}_{i-1}^{T}\, \mathbf{f}_i. \tag{1.5}$$

The spatial transformation between links $i-1$ and i is expressed as:

$$ {}^{i}\mathbf{X}_{i-1} \;=\; \begin{bmatrix} {}^{i}\boldsymbol{R}_{i-1} & \mathbf{0} \\ {}^{i}\boldsymbol{R}_{i-1}\,{}^{i-1}\tilde{\boldsymbol{p}}_{i}^{\,T} & {}^{i}\boldsymbol{R}_{i-1} \end{bmatrix}. \tag{1.6} $$

where ${}^{i}\boldsymbol{R}_{i-1}$ is the rotation matrix describing the change in orientation between link $(i-1)$'s and i's coordinate systems, and ${}^{i-1}\boldsymbol{p}_{i}$ is the position of link i's coordinate system with respect to link $(i-1)$'s. The tilde above the position vector signifies that its components should be combined in a skew symmetric matrix such that $\tilde{\boldsymbol{b}}\boldsymbol{a} = \boldsymbol{b} \times \boldsymbol{a}$. For single degree-of-freedom (DOF) revolute and prismatic joints, modified Denavit-Hartenberg (MDH) parameters as defined in [26] can be used to specify these quantities as shown in Appendix A.

Mass and inertia (first and second moments) properties of a rigid body may also be combined in a 6×6 matrix. For link i, this spatial inertia matrix is defined as follows:

$$ \mathbf{I}_{i} \;=\; \begin{bmatrix} \bar{\boldsymbol{I}}_{i} & \tilde{\boldsymbol{h}}_{i} \\ \tilde{\boldsymbol{h}}_{i}^{T} & m_{i}\mathbf{1}_{3} \end{bmatrix}, \tag{1.7} $$

where $\bar{\boldsymbol{I}}_{i}$ is the 3×3 moment of inertia tensor for link i defined with respect to its own coordinate system. The 3×1 vector, $\boldsymbol{h}_{i}$, is its first mass moment and is equal to $(m_{i}\,\boldsymbol{s}_{i})$ where m_{i} is its mass and $\mathbf{s}_{i}$ is the vector from the origin of the link's coordinate system to its center of mass. Finally, $\mathbf{1}_{3}$ is the 3×3 identity matrix. Note that $\mathbf{I}_{i}$ is constant because it is defined with respect to the coordinate system fixed to the link.

The spatial transformation matrix is also used to transform the spatial inertia matrix to an adjacent coordinate system. This is accomplished with the following congruence transformation [27]:

$$ {}^{i-1}\mathbf{I}_{i} \;=\; {}^{i}\mathbf{X}_{i-1}^{T}\,{}^{i}\mathbf{I}_{i}\,{}^{i}\mathbf{X}_{i-1}, \tag{1.8} $$

where ${}^{i}\mathbf{I}_{i}$ is the spatial inertia of link i expressed with respect to its own coordinate system, and ${}^{i-1}\mathbf{I}_{i}$ is the inertia of the link expressed with respect to coordinate system $i-1$.

Finally, a spatial quantity is related to the joints that connect the links of a serial chain. Called the joint motion subspace or just the joint motion axis, $\boldsymbol{\phi}_{i}$ is a $6 \times n_{i}$ matrix where n_{i} is the number of degrees of freedom in the joint. It is defined in a coordinate system fixed to the link and is therefore constant. For single DOF joints used extensively in this dissertation in which the z-axis of the link's coordinate system lies along the joint axis, $\boldsymbol{\phi}_{i}$ is given by $[0\ 0\ 1\ 0\ 0\ 0]^{T}$ for revolute joints and $[0\ 0\ 0\ 0\ 0\ 1]^{T}$ for prismatic joints. As a result, the joint positions, q_{i}, velocities, $\dot{q}_{i}$, accelerations, $\ddot{q}_{i}$, and its inputs, τ_{i}, are given by scalars associated with these vectors.

1.4 Organization

In Part I of this dissertation, the simulation of land-based systems with multiple chains is studied. Chapters II–IV present the development of simulation algorithms for multiple manipulator systems, and Chapter V deals with multilegged vehicles.

In Chapter II, a dynamics algorithm for multiple manipulator systems lifting a common object is developed. Lilly and Orin's [24] operational space formulation of the dynamics is presented where arbitrarily constrained (rigid) grasps are assumed and the resulting system contains closed, kinematic loops. An efficient implementation of these equations, based on the CRB method and having $O(mN^3)$ complexity, is developed and the resulting computational requirements are determined.

In Chapter III, parallelization of this algorithm is investigated. The manipulator system's parallel topology leads to spatial parallelism where the computation of the dynamics for each serial chain is performed simultaneously. Temporal parallelism can be achieved with parallel numerical integration algorithms where the dynamics of the entire system for different points in time is computed simultaneously. Combining both forms leads to a two-dimensional parallelism and still greater increases in the computation rates. The resulting algorithm is implemented on a Cray Y-MP8 and speedup results are reported for a four PUMA system.

Chapter IV addresses the problems introduced by manipulator singularities on the dynamics algorithm. Mathematical characterization of the dynamic equations of motion when one or more manipulators becomes singular is made. Based on this study, more robust algorithms are developed, and the effect on the computational requirements is determined. This also leads to the development of a new, more efficient dynamics algorithm for dual arm systems with and without singularities.

In Chapter V, a different approach for the simulation of multilegged vehicles is taken. With soft constraints at the feet, the decoupled tree-structure (DTS) approach [7] can be taken where compliance is modeled at the ground contacts and the closed kinematic loops are broken. A new CRB-based algorithm with $O(mN^3)$ complexity is developed for these multiple open-chain systems having star topologies. Efficient kinematic modeling for these systems is also presented, and the computational requirements are determined. Because N is small for these systems, the resulting algorithm is more efficient than the competing AB methods which have a computational complexity of $O(mN)$.

In Part II, the development of a real-time dynamic and hydrodynamic simulation system for underwater robotic vehicle (URV) systems is presented. Chapter VI examines the hydrodynamic forces that are exerted on submerged rigid bodies such as added mass, drag, fluid acceleration and buoyancy. A number of assumptions must be made to simplify the computation of these forces, and the computational requirements for each is presented.

In Chapter VII, the efficient AB algorithm, for the simulation of land-based robotic systems, forms the basis of a new algorithm developed for URV systems. To accomplish this, a very efficient method for incorporating the hydrodynamic forces is developed and the equations for the dynamics of multiple chains and a mobile base are examined. The computational requirements of the resulting $O(mN)$ algorithm are listed.

In Chapter VIII, OOD techniques are presented for the implementation of a simulation system, called **DynaMechs**, that is based on this algorithm. It is capable of simulating a large class of tree-structured robotic systems, having star topologies, which include fixed and mobile base systems with any number of serial chains containing a wide variety of joint types such as revolute, prismatic, and ball-in-socket joints. To achieve this functionality, techniques such as encapsulation, inheritance, and polymorphism are applied, and a very efficient simulator results. Performance results and examples of graphical interfaces on a Silicon Graphics workstation are also presented.

Finally, in Chapter IX, the contributions of this work are summarized and future areas of research are suggested. Two appendices are also included that examine the computational requirements of many common operations used in the simulation of these robotic systems. In Appendix A, kinematic modeling of serial chains using a modified Denavit-Hartenberg (MDH) notation is presented. Using the MDH parameters, new methods for achieving more efficient transformation operations of vectors and matrices are presented. In Appendix B, single chain robotic algorithms for inverse dynamics, computation of joint-space inertia matrices, and the AB method for forward dynamics are presented.