1994

Efficient Dynamic Simulation of an Unmanned Underwater Vehicle with a Manipulator

McMillian, Scott; Orin, David E.; McGhee, Robert B.

Efficient Dynamic Simulation of an Unmanned Underwater Vehicle with a Manipulator

Scott McMillan David E. Orin Robert B. McGhee

Department of Electrical Engineering
The Ohio State University
Columbus, Ohio 43210

Department of Computer Science
Naval Postgraduate School
Monterey, CA 93943

Abstract

In this paper, an efficient dynamic simulation algorithm is developed for an unmanned underwater vehicle (UUV) with a robotic manipulator. It is based on an efficient O(N) algorithm where N is the number of links in the manipulator, and has been extended to include the full effects of a mobile base and various hydrodynamic forces that are exerted on these systems in underwater environments. The effects modeled in this paper are added mass, viscous drag, fluid acceleration, and buoyancy forces. With efficient implementation of the resulting algorithm, the amount of computation including hydrodynamics almost doubles over the original algorithm for a six degree-of-freedom land-based manipulator with a mobile base. Nevertheless, the amount of computation still only grows linearly with the number of links in the manipulator.

1 Introduction

The importance of unmanned underwater vehicles (UUVs) for marine research and subsea development is growing rapidly because their manned counterparts are much more expensive to develop and maintain [1]. This increase in use has brought about a concomitant need for accurate simulations of these systems [2], and with the addition of robotic manipulators to these vehicles, such simulations must become more sophisticated. As with land- and space-based robotics, accurate dynamic simulation can be a cost effective tool in the development of UUVs. Specifically, dynamic simulation can reduce the need for costly prototypes by eliminating many candidate designs early in the development process. In addition, simulators can aid in the design of control algorithms for these systems. By using the simulated UUV to initially test such algorithms, the possibility of potentially damaging instabilities due to algorithm error is eliminated, and risks encountered when the algorithms are finally implemented in hardware are reduced.

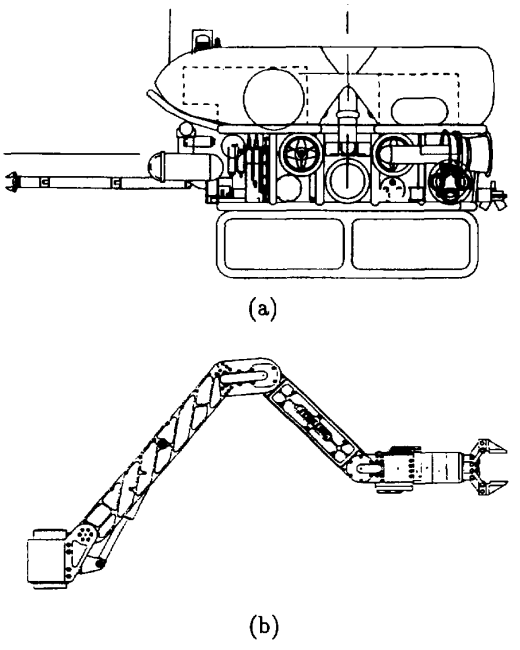
With real-time simulation rates, other uses for dynamic simulators are also possible. One is hardware-in-the-loop simulation where control hardware and software systems are tested by connecting them to a real-time simulation of the UUV. Human-in-the-loop applications can also be implemented when such a simulation is coupled with a realistic 3-D graphical display of the system. With this setup, pilots and mission specialists can be trained in much the same way that aircraft simulators are used to train aviators. Another application can be found in the teleoperation of untethered UUVs. Because significant delays occur in acoustic communication 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 [3].

In order to achieve real-time rates, efficient dynamic simulation algorithms for these systems must be developed. A number of efficient algorithms have been developed to simulate the dynamics of more common land-based robotic systems. The two most notable approaches to this problem are the Composite Rigid Body (CRB) method [4] and the Articulated-Body (AB) method [5]. The latter has an advantage because its computation grows linearly with the number of degrees of freedom, whereas the CRB method has cubic complexity.

The purpose of this paper is to develop an efficient $O(N)$ algorithm for dynamic simulation of a UUV equipped with a manipulator. Figure 1 shows the specific system being used in this research which is the new 4000 meter remotely-operated vehicle (ROV), *Tiburon*, currently under development at the Monterey Bay Aquarium Research Institute (MBARI) [6]. This vehicle will have a six degree-of-freedom Schilling *Titan II* manipulator [7] mounted on the front. Since an efficient implementation of the AB algorithm has been shown to require less computation for systems with more than five degrees of freedom [8], it will be used as the basis for the algorithm developed here.

To accomplish this task, two areas must be addressed. The first is efficiently simulating the effects of a mobile base (the UUV). The resulting algorithm

1050-4729/94 $03.00



(a)

(b)

Figure 1: The MBARI UUV system: (a) *Tiburon*, and (b) the Schilling *Titan II* manipulator.

would also find application in the simulation of space-based robotic systems. The second is determining the most efficient method for including hydrodynamic effects. To this end, Yuh [9] has identified the most important forces which include added mass, viscous drag, fluid acceleration and buoyancy. This paper will develop a method for computing these terms in an articulated linkage system, and equally important, will efficiently incorporate these into the AB algorithm.

In the next section, our dynamics notation is presented along with the principal equations. The hydrodynamic terms needed in the simulation of a single rigid body are presented in Section 3. Then, the method for extending the AB algorithm is presented in Section 4 to include these hydrodynamic terms and the dynamics of a mobile base. Finally, the computational requirements for the resulting algorithm are presented in Section 5, along with a comparison of algorithms for the fixed- and mobile-base systems on land or in space.

2 Background and Notation

In this section, the spatial notation [10] used in this paper to develop the dynamics equations is presented. For this discussion, the model of a UUV/manipulator

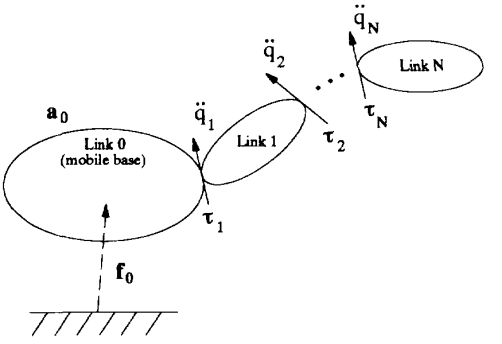


Figure 2: Serial-chain manipulator model with mobile base.

system shown in Figure 2 is used which is a serial chain of $N+1$ rigid bodies. The UUV is represented by the mobile base and designated as link 0. The links of the manipulator are then numbered from 1, closest to the base, to N, the link containing the end-effector. The joints between each of these links can have an arbitrary number of degrees of freedom (up to six). For simplicity, we will assume single degree-of-freedom revolute or prismatic joints in this discussion. This is not limiting, since multiple degrees of freedom can be simulated by concatenation of single degree-of-freedom joints. Nevertheless, this assumption accounts for the vast majority of robotic systems while simplifying the analysis of the computational requirements. Also, the modeling of a mobile base will be discussed in Section 4.

The joint axes between the links, which are indicated by the solid arrows in the figure, are specified by six-element unit vectors, ϕ_i. These vectors are part of the spatial notation used in our work that combines three-dimensional angular and translational quantities into a single vector [5, 10]. Since they are also defined in coordinate systems fixed to the respective links in which the z-axis lies along the joint axis, ϕ_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. The state of the manipulator and its inputs can be specified with scalars that are defined with respect to these vectors. From Figure 2, the state of the manipulator is given by N scalar joint positions, q_i, and velocities, $\dot{q}_i$, and the input joint torques or forces are given by τ_i.

Given these quantities, the goal of dynamic simulation for a fixed-base manipulator is to compute the N joint accelerations, $\ddot{q}_i$, and then numerically integrate these to obtain new values for the joint positions and velocities. An important equation in the development of this simulation algorithm is the force balance on

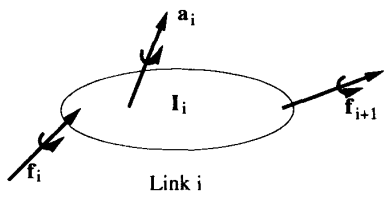


Figure 3: Spatial Notation for Rigid-Body Dynamics.

each link. As illustrated in Figure 3, the force balance equation for link i is given as follows:

$$\mathbf{f}_i = \mathbf{I}_i\mathbf{a}_i - \boldsymbol{\beta}_i + {}^{i+1}\mathbf{X}_i^T\mathbf{f}_{i+1}, \quad (1)$$

where $\mathbf{f}_i$ is the spatial force exerted onto link i by its inboard link and contains the effect of τ_i, and $\mathbf{f}_{i+1}$ is the force exerted by link i onto the next link outboard and includes the effect of τ_{i+1}. The spatial force, $\mathbf{f}_i$, is a six element vector combining the three-dimensional moment, $\boldsymbol{n}$, and translational force, $\boldsymbol{f}$, vectors and is written as $[\boldsymbol{n}_i^T \ \boldsymbol{f}_i^T]^T$. Likewise, the vector, $\mathbf{a}_i$, is the spatial acceleration of link i which is equal to $[\dot{\boldsymbol{\omega}}_i^T \ \boldsymbol{a}_i^T]^T$ consisting of the three-vectors for angular, and translational acceleration. Note that the convention in this paper is to use an italic bold variable, such as $\boldsymbol{a}$, to refer to a Cartesian (three-dimensional) vector representing either a rotational or translational quantity, and a block bold variable, such as $\mathbf{a}$, to refer to a spatial (six-dimensional) vector. Also, $\boldsymbol{\beta}_i$, in Eq. 1, is the vector of bias forces which include the effects of forces due to coriolis and centripetal accelerations and are a function of link velocities [11].

The spatial transformation matrix, ${}^{i+1}\mathbf{X}_i$, is used to transform spatial vectors to different coordinate systems and is defined as follows:

$${}^{i+1}\mathbf{X}_i = \begin{bmatrix} {}^{i+1}\mathbf{R}_i & 0 \\ {}^{i+1}\mathbf{R}_i\,{}^i\tilde{\boldsymbol{p}}_{i+1}^T & {}^{i+1}\mathbf{R}_i \end{bmatrix}, \quad (2)$$

where ${}^{i+1}\mathbf{R}_i$ is the 3×3 rotation matrix from the coordinate system attached to link i to the one attached to link $i+1$, and ${}^i\boldsymbol{p}_{i+1}$ is the Cartesian vector specifying the position of the origin of link $i+1$'s coordinate system with respect to link i's. The tilde above the vector signifies that its components should be combined in a skew symmetric matrix such that $\tilde{\boldsymbol{b}}\boldsymbol{c} = \boldsymbol{b}\times\boldsymbol{c}$.

In the second term of Eq. 1, $\mathbf{I}_i$ is link i's spatial inertia which relates the spatial acceleration of the link to the resultant spatial force. This 6×6 inertia matrix combines the link's mass and inertia properties (zeroth, first and second mass moments) as follows:

$$\mathbf{I}_i = \begin{bmatrix} \bar{\mathbf{I}}_i & m_i\tilde{\mathbf{s}}_i \\ m_i\tilde{\mathbf{s}}_i^T & m_i\mathbf{1}_3 \end{bmatrix}, \quad (3)$$

where $\bar{\mathbf{I}}_i$ is the 3×3 moment of inertia tensor for the link with respect to its own coordinate system, m_i is its mass, $\mathbf{s}_i$ is the vector from the link's coordinate system to its center of mass, and $\mathbf{1}_3$ is the 3×3 identity matrix.

3 Hydrodynamics for Rigid Bodies

Equation 1 gives one of the basic equations for simulation algorithms developed for serial-chain mechanisms. When the motion of its rigid bodies is to be simulated in an underwater environment, a number of additional forces must be added to this equation to model the various hydrodynamic effects. While these forces result from incompressible fluid flow determined by the Navier-Stokes (distributed fluid-flow) equations, "lumped" approximations to these forces are used in this work. To this end, Yuh [9] has identified four separate effects that need to be included in a dynamic simulation of submerged rigid bodies. Under the simplifying assumption that the net hydrodynamic force on a body can be represented as a sum of separately identified components modeling the effects of added mass, drag, fluid acceleration, and buoyancy as illustrated in Figure 4, this section develops a notation consistent with the previous section, and derives the equations needed to compute these hydrodynamic forces exerted on a single rigid body.

For the development of the equations in this section, it is further assumed that the fluid is irrotational and unbounded. The former is acceptable since rotation due to any vortices in the fluid would be small compared to rotation of the body, or it would be on such a small scale compared to the extent of the body as to be negligible. The exception is wave action in shallow depths, which is not an environment that will be encountered by most UUV systems. The unbounded assumption poses more of a problem because it is violated with proximity to the bottom or surface of the fluid, or even the other links. For our first order approximations, however, it should generally be acceptable.

3.1 Added Mass

To those acquainted with the dynamics of manipulators in space or air, probably the most surprising hydrodynamic effect is the added mass force. When a body is accelerated through a fluid, some of the surrounding fluid is also accelerated with the body. A force is exerted on the surrounding fluid to achieve this acceleration, and the reaction force, which is equal in magnitude and opposite in direction, is exerted on the

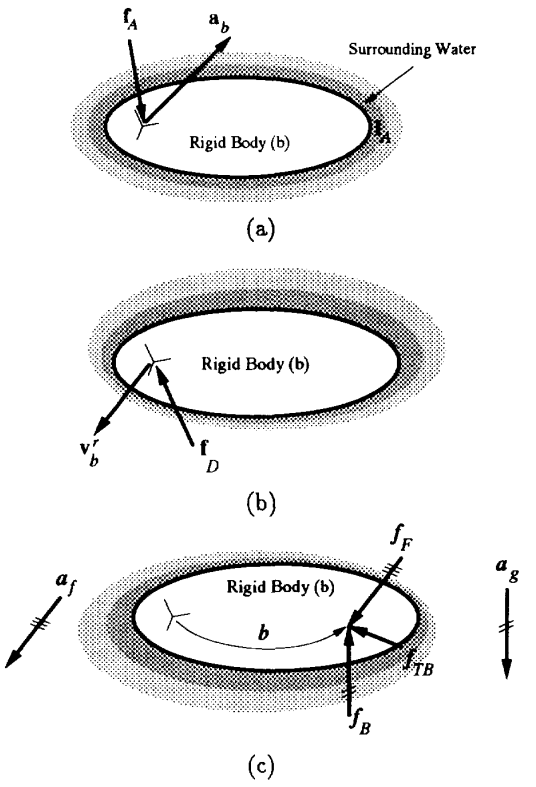


Figure 4: Hydrodynamic forces: (a) added mass, (b) drag, and (c) buoyancy and fluid acceleration.

body. The latter is referred to as the added mass force. Its computation within our framework is presented in this section.

With our assumptions about lumped approximations, the fluid that is accelerated has inertia properties, called added mass, that can be specified with a 6×6 matrix, $\mathbf{I}_A$, as shown in Figure 4(a). As with the spatial inertia for a rigid body, this added mass matrix is symmetric and positive-definite. Since this inertia is a function of the body's surface geometry, however, there is no concept of principal axes as in rigid body analysis along which, torque and angular momentum are collinear. In fact, with added mass, unlike the rigid body's mass, an applied translational force can result in a non-collinear acceleration of the center of gravity as well. Consequently, the added mass matrix does not have the same structure as shown in Eq. 3 for the spatial inertia of a rigid body in space or air. For a general body shape, the matrix will be full which leads to notably different dynamic behavior as compared to the rigid body counterpart.

Newman [12] derived a set of equations to compute the added mass force that is exerted on a rigid body accelerating through an unbounded, inviscid fluid that is itself not accelerating (that is, has steady, irrotational flow). This was found by taking the derivative of the total momentum of the fluid. As shown in [11], translating Newman's equations into spatial notation results in the following equation:

$$\mathbf{f}_A = -\mathbf{I}_A \begin{bmatrix} \dot{\omega}_b \\ \dot{v}_b \end{bmatrix} - \begin{bmatrix} \tilde{\omega}_b & \tilde{v}_b \\ 0 & \tilde{\omega}_b \end{bmatrix} \mathbf{I}_A \begin{bmatrix} \omega_b \\ v_b \end{bmatrix}, \quad (4)$$

where negative signs are needed to compute the reaction force that is exerted *onto* the rigid body, and ω_b and v_b are the angular and translational velocities of the body, respectively. The translational acceleration term, $\dot{v}_b$, is not the true acceleration of the rigid body, but is rather the time derivative of v_b with respect to the body's rotating reference frame.

Since the AB simulation algorithm uses the true acceleration of the body, a_b, Eq. 4 must be modified before it can be efficiently incorporated into the algorithm. This is accomplished by using the following relationship:

$$a_b = \dot{v}_b + \omega_b \times v_b, \quad (5)$$

where $\dot{v}_b$ is frequently referred to as the *rate of growth* of a vector, and $\omega_b \times v_b$ the *rate of transport*. Substituting this into Eq 4 leads to the following equation for the added mass force:

$$\mathbf{f}_A = -\mathbf{I}_A \begin{bmatrix} \dot{\omega}_b \\ a_b \end{bmatrix} + \mathbf{I}_A \begin{bmatrix} 0 \\ \omega_b \times v_b \end{bmatrix} - \begin{bmatrix} \tilde{\omega}_b & \tilde{v}_b \\ 0 & \tilde{\omega}_b \end{bmatrix} \mathbf{I}_A \begin{bmatrix} \omega_b \\ v_b \end{bmatrix}. \quad (6)$$

In the final step of this derivation, the effects of fluid translational acceleration on the force resulting from added mass can be incorporated by replacing the translational acceleration of the rigid body with its translational acceleration *relative* to the surrounding fluid, a_b^r [12]. This is defined as follows:

$$a_b^r = a_b - {}^b a_f, \quad (7)$$

where ${}^b a_f$ is the translational acceleration of the fluid expressed in the body-fixed coordinate system. Likewise, the translational velocity term is replaced in the added mass equation with the relative translational velocity, v_b^r.

It is desirable to write Eq. 6 in a form similar to Eq. 1 so that it may be more easily incorporated into the AB dynamics algorithm. To accomplish this, the terms multiplying the unknown spatial acceleration,

$\mathbf{a}_b$, must be separated from the rest which leads to the final form of the added mass force equation:

$$\mathbf{f}_A = -\mathbf{I}_A \begin{bmatrix} \dot{\omega}_b \\ a_b \end{bmatrix} + \beta_A, \quad (8)$$

where

$$\beta_A = \mathbf{I}_A \begin{bmatrix} 0 \\ {}^b a_f + \omega_b \times v_b^r \end{bmatrix} - \begin{bmatrix} \tilde{\omega}_b & \tilde{v}_b^r \\ 0 & \tilde{\omega}_b \end{bmatrix} \mathbf{I}_A \begin{bmatrix} \omega_b \\ v_b^r \end{bmatrix}, \quad (9)$$

which is a function of known state variables.

3.2 Drag and Lift

When an object moves through a viscous fluid, drag and lift forces are exerted on it. For most land-based robots, the motions are slow enough (and the air is thin enough) that these air resistance forces can be neglected. For underwater robotics, the fluid is significantly denser and even reasonably slow motions can result in large forces. Ignoring hydrodynamic lift and the related forces due to vortex shedding which are believed to be small for the application at hand, drag forces arise from pressure or profile drag as well as shear or skin drag. For underwater manipulation, the shear drag will also be small, so that the emphasis here is on the modeling of the pressure drag resulting from non-zero normal components of relative velocity between the body's surface and the fluid.

In order to solve the distributed flow problem for a general body, a surface integral over the entire body is required to compute the drag force and moment, $\mathbf{f}_D$, exerted on the body. This surface integral is a function of the normal components of velocity over the entire body and also drag coefficients that are generally poorly known and variable which renders them computationally expensive. Therefore, simplifying assumptions are made to significantly reduce the amount of computation while still computing a *reasonable* drag force. The most significant assumption in this paper is that the links can be approximated by cylinders. The resulting procedure to compute the drag force is based on one by Sarpkaya and Isaacson [13] to compute the total force on a stationary cylinder arbitrarily oriented with respect to the fluid velocity, and has been extended in our work to include the effects of arbitrary translational and angular velocity of the cylinder as well.

The benefit of using cylinders is that strip theory can be used to replace the surface integral with a line integral along the length of the cylinder. With this simplification, the velocity relative to the fluid and normal to the edge of each circular disk element, v^n, must be determined. Given the angular velocity of the link and its translational velocity relative to the fluid, the translational velocity relative to the fluid of a disk centered at a point, p, along the axis of the cylinder is approximated. Assuming the radius of the cylinder is small compared to the length, it is given as follows:

$$v^r(p) = v_b^r + \omega_b \times p. \quad (10)$$

where ω_b is the angular velocity of the cylinder which is also the angular velocity relative to the fluid since it is assumed to be irrotational, and v_b^r is the translational velocity of the cylinder relative to the fluid at the body-fixed coordinate system. The normal velocity, v^n, is the component of v_b^r normal to the axis of the cylinder. With this, a procedure is developed in [11] that can be used to compute the resultant spatial drag force which is a function of the drag coefficient, C_D and modified square law of relative velocity defined as follows:

$$\mathbf{f}_D = f(C_D, |v^n|\ v^n). \quad (11)$$

In the model developed in [11], Gauss-quadrature approximates the line integral so that computation of the normal velocity occurs only at four points along the cylinder.

Another assumption that has been made is that the added mass matrix and drag coefficients are known and constant. In actuality, these quantities are coupled and extremely difficult to compute with a high degree of accuracy, and vary non-linearly with respect to velocity and other parameters [13]. However, we believe that over the range of operating conditions typically encountered by a UUV, a constant coefficient assumption is the only reasonable approach and is adequate for the purposes of the desired application.

3.3 Buoyancy and Fluid Acceleration

Because of the similarity between buoyancy and fluid acceleration forces, they are presented together in this section. Both are translational forces as illustrated in Figure 4(c). They are exerted at the center of buoyancy of the body, which is the center of volume of the body or equivalently the center of mass of the fluid that was displaced by the body. Finally, they are proportional to the mass of the fluid that is displaced by the body, m_f.

The buoyancy force, f_B, is exerted on the body vertically through, and acting in the direction opposite of gravity. This force is a result of Archimedes principle which states that a body immersed in a fluid is buoyed up with a force equal to the weight of the fluid displaced by the body. In terms of the gravitational

acceleration, $\boldsymbol{a}_g$, this force is computed as follows:

$$\boldsymbol{f}_B = -m_f\, \boldsymbol{a}_g. \tag{12}$$

The gravitational acceleration, $\boldsymbol{a}_g$ points down along the positive z-axis as is traditional in marine mechanics, and has magnitude g, the gravitational constant. A similar equation was given by Yuh [9] for the fluid acceleration force which is given as follows:

$$\boldsymbol{f}_F = m_f\, \boldsymbol{a}_f, \tag{13}$$

where $\boldsymbol{a}_f$ is the acceleration of the fluid.

For increased computational efficiency, both forces are combined as follows:

$$\boldsymbol{f}_{TB} = m_f\, (\boldsymbol{a}_f - \boldsymbol{a}_g), \tag{14}$$

which we call the *total buoyancy* force. To use this in existing robot dynamics algorithms, the equivalent spatial force exerted at the origin of the body-fixed coordinate system must be found. The resulting force is computed by the following equation:

$$\mathbf{f}_{TB} = \begin{bmatrix} \boldsymbol{b} \times \boldsymbol{f}_{TB} \\ \boldsymbol{f}_{TB} \end{bmatrix}, \tag{15}$$

where $\boldsymbol{b}$ is the vector from the body-fixed coordinate system to its center of buoyancy.

4 UUV Simulation Algorithm

In this section, the development of a dynamic simulation algorithm for an articulated chain of rigid bodies in an underwater environment is discussed. To begin, the dynamic equation of motion for a single rigid body (one link) in a fluid is examined. To introduce the hydrodynamic forces into the equation, the spatial force balance on a given link i is written which includes the forces derived in the previous section and the link forces from Eq. 1. This equation is written as follows:

$$\begin{aligned} \mathbf{f}_i - {}^{i+1}\mathbf{X}_i^T \mathbf{f}_{i+1} &+ \mathbf{f}_{A_i} + \mathbf{f}_{D_i} + \mathbf{f}_{TB_i} \\ &= \mathbf{I}_i \mathbf{a}_i - \boldsymbol{\beta}_i, \end{aligned} \tag{16}$$

where $\mathbf{f}_{A_i}$, the added mass force from Eq. 8, is also a function of the unknown link acceleration, $\mathbf{a}_i$. Grouping these acceleration terms together and rewriting the equation to resemble the form of Eq. 1, and adding the gravitational force which to this point was not accounted for, leads to the following hydrodynamic equation of motion for the link:

$$\mathbf{f}_i = \mathbf{I}_i^H \mathbf{a}_i - \boldsymbol{\beta}_i^H + {}^{i+1}\mathbf{X}_i^T \mathbf{f}_{i+1}, \tag{17}$$

where

$$\mathbf{I}_i^H = \mathbf{I}_i + \mathbf{I}_{A_i}, \text{ and} \tag{18}$$

$$\boldsymbol{\beta}_i^H = \boldsymbol{\beta}_i + \boldsymbol{\beta}_{A_i} + \mathbf{f}_{D_i} + \mathbf{f}_{TB_i} + \mathbf{f}_{g_i}. \tag{19}$$

This equation states that the spatial acceleration of the body and the exerted force are linearly related through the sum of the rigid body's spatial inertia and the added mass of the fluid surrounding the body.

A comparison of Eq. 17 with Eq. 1 indicates a few changes that must be made to the AB algorithm to incorporate these hydrodynamic terms. First, the use of the link's spatial inertia, $\mathbf{I}_i$ is replaced with its *hydrodynamic inertia* from Eq. 18. While this affects the calculation of the AB inertias [5, 8] within the AB algorithm, the matrix addition to compute $\mathbf{I}_i^H$ does not add to the amount of computation because both quantities are constant and the sum can be computed off-line.

Because the acceleration is multiplied by a sum of spatial inertia and added mass matrices for the body, however, gravitation force must be accounted for in a different manner than previously. Without added mass, the most efficient method for including gravitational effects was to bias the serial-chain's base acceleration with a fictitious gravitational acceleration in the direction opposite to gravity. Gravitational force was then indirectly computed for each link in the chain during a forward recursion which computes the acceleration for each link based on the previous link's acceleration. Since gravitational forces are not exerted on the added mass of the bodies, an explicit computation for the gravitational force exerted on the body, $\mathbf{f}_{g_i}$, is necessary and has been included.

The computation of $\boldsymbol{\beta}_i$ is replaced with the computation of $\boldsymbol{\beta}_i^H$ from Eq. 19. While this still includes the computation of $\boldsymbol{\beta}_i$, computation of the hydrodynamic forces adds a significant amount of computation to the AB algorithm and requires the computation of $\boldsymbol{v}_i^r$, $\boldsymbol{a}_f$ and $\boldsymbol{a}_g$ with respect to each link's local coordinate system.

Finally, the dynamics of a mobile base is included to enable the simulation of a manipulator mounted on a UUV. This is accomplished by modeling the vehicle as another link (link 0) in the serial chain which has a six degree-of-freedom joint (indicated by the dashed arrow in Figure 2) between it and a fictitious base fixed in the inertial frame. The computation involving this "link" can be modified to achieve greater efficiency if it is treated as a single six degree-of-freedom joint, rather than six successive single degree-of-freedom joints, whose spatial representation, $\boldsymbol{\phi}_0$, is given as a 6×6 identity matrix.

The most notable change from a fixed base is the computation of this joint acceleration which is equal to the spatial acceleration of this link expressed in the body-fixed coordinate system, $\mathbf{a}_0$. Using notation taken from the AB algorithm in [11], this can be com-

Table 1: Computational requirements of AB simulation algorithm (Numbers in parentheses for $N = 6$).

	×	+
Fixed Base	$243N - 268$ (1190)	$224N - 262$ (1082)
Mobile Base	$243N + 69$ (1527)	$224N + 48$ (1392)
Mobile Base with Hydrodynamics	$413N + 238$ (2716)	$368N + 186$ (2394)

puted as follows:

$$\mathbf{a}_0 = (\mathbf{I}_0^*)^{-1} \boldsymbol{\beta}_0^*, \tag{20}$$

where $\mathbf{I}_0^*$ is the 6×6 AB inertia of the entire UUV system that is "felt" at the vehicle body's coordinate system when all of the manipulator joints are free to move, and $\boldsymbol{\beta}_0^*$ is the net force acting on this inertia including UUV thruster forces, all reaction forces due to joint torques in the manipulator, and $\boldsymbol{\beta}_0^H$. Also note that the matrix to be inverted is of constant size and does not add to the complexity of the resulting algorithm.

5 Computational Requirements

With the modifications described in the previous section, an efficient implementation of the AB hydrodynamic simulation algorithm results. The computational cost of the equations in the algorithm have been examined in detail for manipulators with revolute joints and the results are listed in the last line of Table 1. In this table, the number of floating point multiplies and divides are combined under the label of multiplies (×), and the number of additions and subtractions are combined under the label of additions (+). Also note that the minimum number of trigonometric functions is used (one sine/cosine pair for each revolute joint and three for the base) and is not included in the results. The total number of floating point operations for a UUV system with an N-link manipulator with revolute joints is $413N + 238$ multiplies and $368N + 186$ additions.

The details for achieving this result can be found in [11]. The most significant efficiencies in the computation are gained with proper implementation of transformations of vectors and matrices between coordinate systems attached to each rigid body in the system. In our implementation of the algorithm, the orientation of the UUV vehicle is specified by three Euler angles. To transform needed Cartesian quantities expressed in the inertial reference frame to the body-fixed frame, it is most efficient to use three separate planar rotations instead of building a 3×3 rotation matrix and performing a complete matrix vector multiply. Using the same concept, rotations between adjacent links in the serial chain are even more efficient. With the use of modified Denavit-Hartenberg parameters [10], they can be accomplished with two planar rotations instead of three. Featherstone [5] has extended this to transformations of spatial vectors which is accomplished using two planar screw transformations. The single most important efficiency, however, is achieved in the congruence transformation of a 6×6 symmetric mass matrix. A careful analysis in [14] shows that this is accomplished most efficiently with [89M, 90A]*.

For comparison, Table 1 also lists the computational costs for the simulation without hydrodynamics for the same manipulator with revolute joints for both fixed and mobile bases. The latter can be used as an efficient algorithm for space-based robotic systems. The primary difference in computation between this and the UUV algorithm occurs in the kinematics steps where additional velocities, accelerations, and hydrodynamic terms in $\boldsymbol{\beta}_i^H$ are needed. When the system is simplified even further with the additional assumption of a fixed base, even less computation is required as shown in the first line of Table 1. The fact that base velocity and acceleration are zero, can be used to significantly reduce the amount of computation required for Link 1 of the manipulator. The resulting number of operations for the Fixed Base algorithm is comparable to and slightly less than the results of Brandl, Johanni, and Otter [8].

The numbers in parentheses in Table 1 correspond to the amount of computation that will be required to simulate a system with a six degree-of-freedom manipulator with revolute joints, and the last line represents the amount required to simulate the MBARI ROV and the Schilling manipulator. The cost for the hydrodynamic simulation represents close to a 75% increase in computation over the mobile base simulation (without hydrodynamics) and approximately a 125% increase in computation over the simulation of this manipulator on a fixed base.

6 Summary and Conclusions

In this paper, an efficient algorithm has been developed for dynamic simulation of an underwater vehicle equipped with a manipulator. Since many efficient algorithms for the simulation of land-based manipu-

*[89M,90A] = 89 multiplies and 90 additions.

lators have already been developed, one was chosen as the basis for our work. The system for which this algorithm has been developed is the Monterey Bay Aquarium Research Institute's ROV, *Tiburon*, with a Schilling manipulator. Based on the number of degrees of freedom in this manipulator, the Articulated-Body (AB) simulation algorithm is the most efficient and was used in this paper.

A significant goal of this research was to efficiently incorporate hydrodynamic effects into this algorithm. A number of hydrodynamic effects on a single rigid body were identified from previous work by Yuh [9] which includes added mass, drag, fluid acceleration and buoyancy forces. These were derived in a form consistent with the robot dynamics algorithm to facilitate their incorporation into the AB algorithm. Then, this result was extended to systems with serial chains of rigid bodies. Finally, the algorithm was modified to include the effects of a mobile base.

A detailed analysis of the amount of computation required by the algorithm has also been accomplished. The resulting hydrodynamic algorithm was found to require $[(413N+238)\mathrm{M}, (368N+186)\mathrm{A}]$ computations for a UUV with a manipulator that has N revolute joints. For the MBARI ROV with the Schilling Arm ($N = 6$), the computation of the dynamics requires [2716M, 2394A]. Although this is nearly twice as many operations as compared to the operations required for systems in space or air, it is still low enough to achieve real-time simulation rates on reasonably priced workstations.

Currently, this algorithm is being implemented using an object-oriented approach in C++. The resulting simulation package will be capable of efficiently simulating a variety of robotic systems underwater, as well as on land and in space. After completion, the next step is to evaluate the accuracy of this new approach to computing the hydrodynamic forces. This involves a comparison of simulation results, of a number of objects under various fluid motion conditions, to actual studies and test results that are reported in marine engineering publications.

7 Acknowledgments

This work was supported in part by an AT&T Ph.D. Scholarship and NFS Grant No. BCS-9311269 to The Ohio State University, and by NFS Grant No. BCS-9109989 to the Naval Postgraduate School. The authors would also like to thank Dr. Anthony Healey of the Naval Postgraduate School for informative discussions on hydrodynamics, and James B. Newman of MBARI for access to and involvement in the *Tiburon* project.

8 References

[1] *Proceedings of the 20th Annual Technical Symposium and Exhibition of the Association for Unmanned Vehicle Systems.* (Washington, D.C.): Association for Unmanned Vehicle Systems, June 1993.

[2] M. J. Zyda, R. B. McGhee, S. Kwak, D. B. Nordman, R. C. Rogers, and D. Marco, "Three-Dimensional Visualization of Mission Planning and Control for the NPS Autonomous Underwater Vehicle," *IEEE Journal of Oceanic Engineering*, vol. 15, no. 3, pp. 217–221, 1990.

[3] J. Funda and R. P. Paul, "Teleprogramming: Overcoming Communication Delays in Remote Manipulation," in *Proc. of 1st IARP Workshop on Mobile Robots for Subsea Environments*, (Monterey, CA), pp. 155–162, October 1990.

[4] M. W. Walker and D. E. Orin, "Efficient Dynamic Computer Simulation of Robotic Mechanisms," *Journal of Dynamic Systems, Measurement, and Control*, vol. 104, pp. 205–211, September 1982.

[5] R. Featherstone, "The Calculation of Robot Dynamics Using Articulated-Body Inertias," *The Int. Journal of Robotics Research*, The MIT Press, vol. 2, pp. 13–30, Spring 1983.

[6] J. B. Newman and B. H. Robison, "Development of a Dedicated ROV for Ocean Science," *Marine Technology Society Journal*, vol. 26, pp. 46–53, Winter 1992.

[7] Anon., "Titan II High Resolution Bilateral Force Feedback Remote Manipulator System," Technical Report, Schilling Development, Inc., Davis, CA, April 1991.

[8] H. Brandl, R. Johanni, and M. Otter, "A Very Efficient Algorithm for the Simulation of Robots and Similar Multibody Systems Without Inversion of the Mass Matrix," in *Proc. of IFAC/IFIP/IMACS Int. Symp. on Theory of Robots*, (Vienna, Austria), December 1986.

[9] J. Yuh, "Modeling and Control of Underwater Robotic Vehicles," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 20, pp. 1475–1483, November/December 1990.

[10] K. W. Lilly and D. E. Orin, "Alternate Formulations for the Manipulator Inertia Matrix," *International Journal of Robotics Research*, The MIT Press, vol. 10, pp. 64–74, February 1991.

[11] S. McMillan, D. E. Orin, and R. B. McGhee, "Simulating Hydrodynamic Effects for Underwater Manipulation," Technical Report NPSCS–93–014, Naval Postgraduate School, Monterey, CA, September 1993.

[12] J. N. Newman, *Marine Hydrodynamics*. Cambridge, MA: The MIT Press, 1977.

[13] T. Sarpkaya and M. Isaacson, *Mechanics of Wave Forces on Offshore Structures*. New York, NY: Van Nostrand Reinhold Co., 1981.

[14] S. McMillan, "Efficient Computation of Articulated Body Inertias," Technical Report NPSCS–93–010, Naval Postgraduate School, Monterey, CA, September 1993.