Proceedings of the 2005 IEEE
International Conference on Robotics and Automation
Barcelona, Spain, April 2005

A New Perspective on O(n) Mass-Matrix Inversion for Serial Revolute Manipulators

Kiju Lee
Department of Mechanical Engineering
Johns Hopkins University
Baltimore, Maryland, USA
kiju@jhu.edu

Gregory S. Chirikjian
Department of Mechanical Engineering
Johns Hopkins University
Baltimore, Maryland, USA
gregc@jhu.edu

***Abstract* – This paper describes a new algorithm for the efficient mass-matrix inversion of serial manipulators. Whereas several well-known *O(n)* algorithms already exist, our presentation is an alternative and completely different formulation that builds on Fixman's theorem from the polymer physics literature. The main contributions here are therefore adding a new perspective to the manipulator dynamics literature and providing an alternative to existing algorithms. The essence of this theory is to consider explicitly the band-diagonal structure of the inverted mass matrix of a manipulator with no constraints on link length, offsets or twist angles, and then build in constraints by appropriate partitioning of the inverse of the unconstrained mass matrix. We present the theory of the partitioned mass matrix and inverse of the mass matrix for serial revolute manipulators. The planar N-link manipulator with revolute joints is used to illustrate the procedure. Numerical results verify the *O(n)* complexity of the algorithm. Exposure of the robotics community to this approach may lead to new ways of thinking about manipulator dynamics and control.**

Index Terms* – *manipulator dynamics, O(n), revolute manipulators, Fixman's theorem

I. Introduction

This paper is concerned with an *O(n)* forward dynamics computation for serial manipulators. The topic of *O(n)* algorithms for inverting the mass matrix for n-degree-of-freedom manipulators as an alternative to the more traditional $O(n^3)$ computations needed for general matrix inversion is not a new idea in the robotics community. Since the first *O(n)* algorithm for dynamics calculation developed in the multi-body systems literature by Vereshchagin in 1975 [11], researchers have tried to improve the efficiency and extend it using many different approaches. In the robotics literature, the Luh-Walker-Paul recursive Newton-Euler approach [15] has been a cornerstone of manipulator inverse dynamics for many years. Hollerbach showed that *O(n)* inverse dynamics could also be achieved within a Lagrangian dynamics setting [14]. In [3] and [4], Rodriguez described *O(n)* solutions for both the forward and inverse dynamics problems for serial manipulators by using recursive techniques from linear filtering and smoothing theory: Kalman filtering and Bryson-Frazier fixed time-interval smoothing ([12], [13]). He also showed two recursive factorization methods of the mass matrix for fixed-base and mobile-base manipulators [4]: ① Newton-Euler factorization and ② Innovations factorization. As another approach, a new decomposition method using analytical Gaussian Elimination (GE) of the inertia matrix was introduced by Saha in [9]. Extending his previous work, in [8], he presented a recursive forward dynamics algorithm for open-loop, serial-chain robots. This work builds on the work of Angeles and Ma who developed the Natural Orthogonal Complement for the manipulator mass matrix [16]. Saha's algorithm has *O(n)* computational complexity and is also based on reverse GE applied to the analytical expressions of the elements of the inertia matrix.

Recently, the paper by Anderson and Critchley ([10]) presented a numerical analysis and simulation of multi-rigid-body dynamic systems which are heavily constrained. They described the computational efficiency gains related with the new *O(n+m)* operations for systems having *n* generalized coordinates and m constraints. In addition, Featherstone ([6], [7]) showed a new efficient factorization of the joint space inertia matrix (JSIM) for branched kinematic trees. He described two factorization methods for JSIM: ① LTL factorization and ② LTDT factorization.

In general, the forward dynamics problem is to solve for the accelerations from the forces. Let $G(\vec{q})$ denotes the inertia matrix, which is a function of joint angles. The dynamical equations to be solved are of the form

$$G(\vec{q})\ddot{\vec{q}} = \vec{\tau} - \vec{C}(\vec{q},\dot{\vec{q}}) \tag{1}$$

where $\vec{\tau}$ is the vector of joint torques, $\vec{C}(\vec{q},\dot{\vec{q}})$ is the vector of Coriolis and centrifugal terms, $\ddot{\vec{q}}$ is the vector of joint accelerations, and we assume that there is no gravitational terms in this paper. In the classical $O(n^3)$ algorithms, $G(\vec{q})$ is inverted explicitly and then applied to the right side of the equation to yield $\ddot{\vec{q}}$. In contrast, *O(n)* algorithms solve the problem without explicitly computing the inverse of $G(\vec{q})$.

In this paper, we introduce an alternative and completely different approach to solve the mass-matrix inversion problem in *O(n)* computational complexities. Our method is based on Fixman's idea for partitioned internal coordinates in polymer chains. In [1], Fixman presented an effective way to calculate the mass metric tensor determinant (MMTD) of the polymer chains (i.e., determinant of the mass matrix) containing holonomic constraints. He was concerned with a freely jointed chain which has *N+1* point masses at each joint. Internal coordinates (i.e., joint angles) were partitioned into *f* degrees of freedom $(a_1,\cdots,a_f)$, called 'soft variables', and r holonomic constraints $(b_1,\cdots,b_r)$, called 'hard variables' (i.e., constraints on link lengths), such that *f+r=3(N+1).* If G^{aa} is defined as the mass matrix of the constrained system, then (1) can be written as

$$G^{aa}\ddot{\vec{q}}_a = \vec{\tau} - \vec{C}(\vec{q}_a,\dot{\vec{q}}_a) \qquad (2)$$

where $\vec{q}_a = [a_1,\cdots,a_f]^T$. The forward dynamics problem of serial manipulators containing constraints is to get $\ddot{\vec{q}}_a$ from (2). This paper is concerned with an *O(n)* algorithm to solve this problem. Fixman also presented simulation for polymer dynamics in [2].

In section II, we provide an analytical expression for the mass matrix and the inverse of mass matrix, we explain Fixman's theorem and our extension of his idea. In section III, we apply our extension of Fixman's theorem to an N-link planar serial manipulator with revolute joints.

II. O(N) ALGORITHM FOR FORWARD DYNAMICS

In the context of a manipulator with fixed base, the number of point masses is reduced by one when compared with a freely floating polymer chain. Therefore, given a set of N point masses $\{m_1,\cdots,m_N\}$ with a corresponding set of absolute position vectors $\{\vec{x}_1,\cdots,\vec{x}_N\}$, we define a *3N*-dimentional position vector and vector of n generalized coordinates as:

$$\vec{x} = \begin{pmatrix}\vec{x}_1\\ \vdots \\ \vec{x}_N\end{pmatrix} \text{ and } \vec{q} = \begin{pmatrix}q_1\\ \vdots \\ q_n\end{pmatrix}.$$

The mass matrix can be calculated as:

$$G = \left[\frac{\partial \vec{x}}{\partial \vec{q}}\right]^T [\mathrm{diag}(m_i)] \left[\frac{\partial \vec{x}}{\partial \vec{q}}\right].$$

In the case of an unconstrained system, *n=3N,* all of the matrices in above are square, and we can use the formula $(\mathrm{AB})^{-1} = \mathrm{B}^{-1}\mathrm{A}^{-1}$ together with the fact that

$$\left[\frac{\partial \vec{x}}{\partial \vec{q}}\right]^{-1} = \left[\frac{\partial \vec{q}}{\partial \vec{x}}\right]$$

to get

$$G^{-1} = \left[\frac{\partial \vec{q}}{\partial \vec{x}}\right][\mathrm{diag}(1/m_i)]\left[\frac{\partial \vec{q}}{\partial \vec{x}}\right]^T. \qquad (3)$$

While this is not the case of interest in manipulator dynamics, it is a useful construction that leads to efficient manipulator dynamics.

Fixman's theorem [1]

A chain of *N+1* point masses is subjected to r holonomic constraints, $(b_1,\cdots,b_r)$. The system is described with internal coordinates $(q_1,\cdots,q_{3(N+1)})$ partitioned into f soft variables and r hard variables as $(a_1,\cdots,a_f;b_1,\cdots,b_r)$ such that *f+r=3(N+1).* Then, the elements of the total mass matrix and the inverse of total mass matrix can be computed by

$$G_{ij} = \sum_{l=0}^{N} m_l(\partial\vec{x}_l/\partial q_i)\cdot(\partial\vec{x}_l/\partial q_j) \qquad (4)$$

and

$$H_{ij} = (G_{ij})^{-1} = \sum_{l=0}^{N}\frac{1}{m_l}(\partial q_i/\partial\vec{x}_l)(\partial q_j/\partial\vec{x}_l)^T \qquad (5)$$

where $\vec{x}_l$ denotes the absolute position of the l^{th} point mass. According to the partitioned internal coordinates, the mass matrix and the inverted mass matrix can be divided in 4 block matrices as:

$$G = \begin{pmatrix}G^{aa} & G^{ab}\\ G^{ba} & G^{bb}\end{pmatrix}, \text{ and } H = \begin{pmatrix}H^{aa} & H^{ab}\\ H^{ba} & H^{bb}\end{pmatrix} \qquad (6)$$

where we define $H = G^{-1}$. All blocks in H can be obtained by the following equations:

$$\begin{aligned}
H_{ij}^{aa} &= \sum_{l=0}^{N}\frac{1}{m_l}(\partial a_i/\partial\vec{x}_l)(\partial a_j/\partial\vec{x}_l)^T\\
H_{ij}^{ab} &= \sum_{l=0}^{N}\frac{1}{m_l}(\partial a_i/\partial\vec{x}_l)(\partial b_j/\partial\vec{x}_l)^T\\
H_{ij}^{ba} &= \sum_{l=0}^{N}\frac{1}{m_l}(\partial b_i/\partial\vec{x}_l)(\partial a_j/\partial\vec{x}_l)^T\\
H_{ij}^{bb} &= \sum_{l=0}^{N}\frac{1}{m_l}(\partial b_i/\partial\vec{x}_l)(\partial b_j/\partial\vec{x}_l)^T.
\end{aligned} \qquad (7)$$

Using the fact that $G\cdot H = I$, where I denotes the identity matrix,

$$G\begin{pmatrix}I & H^{ab}\\ 0 & H^{bb}\end{pmatrix} = \begin{pmatrix}G^{aa} & 0\\ G^{ba} & I\end{pmatrix},$$

and it follows that

$$(\det G)(\det H^{bb}) = \det G^{aa}. \qquad (8)$$

Equation (8) is Fixman's theorem, which states that the determinant of the constrained mass matrix (i.e., the mass matrix of interest in robotics) can be obtained from the determinants of two matrices that are easier to compute than the original. In robotics we are concerned not only with obtaining the determinant of this matrix, but rather inverting it. This can be achieved in an efficient way by extending Fixman's theorem as described below.

Extending Fixman's method for fast computation of the inversed mass matrix

Fixman described an efficient method for computing the MMTD of the system as reviewed above. As an extension of his idea, we use the same partitioned internal coordinates to calculate the inversed mass matrix. The equation for the forward dynamics problem of the constrained serial manipulator can be expressed as

$$\ddot{\vec{q}}_a = (G^{aa})^{-1}\vec{b} \tag{9}$$

where $\vec{b} = \vec{\tau} - \vec{C}(\vec{q},\dot{\vec{q}})$.

In general, the mass matrix of the system has the form of a full (non-sparse) matrix. Therefore, the conventional non-recursive computation of inversion would be performed in $O(n^3)$. On the other hand, all the blocks of H have zeros for most of entries except around the diagonal elements, because the formulations include derivatives of parameters with respect to the absolute position vectors. (This will be showed in section III for the case of a particular serial manipulator, and it will be obvious why this is true in general.) This means that all blocks are in fact sparse and band-limited for a large matrix dimension, so that, in general, the computation can be performed very efficiently. Therefore, we investigate a new method for fast inversion of the manipulator mass matrix using the blocks of H to calculate $(G^{aa})^{-1}$, instead of direct inversion.

Our approach is to solve the larger system of equations

$$\begin{pmatrix} G^{aa} & 0 \\ G^{ba} & I \end{pmatrix}\begin{pmatrix} \vec{X} \\ \vec{Y} \end{pmatrix} = \begin{pmatrix} \vec{b} \\ 0 \end{pmatrix} \tag{10}$$

Obviously, $\vec{X} = \ddot{\vec{q}}_a$, and the solution of (10) is equal to the solution of (9). Performing block-by-block multiplication, we see that

$$\begin{pmatrix} H^{aa} & H^{ab} \\ H^{ba} & H^{bb} \end{pmatrix}\begin{pmatrix} G^{aa} & 0 \\ G^{ba} & I \end{pmatrix} = \begin{pmatrix} I & H^{ab} \\ 0 & H^{bb} \end{pmatrix}.$$

This can be written in a different way as

$$\begin{pmatrix} G^{aa} & 0 \\ G^{ba} & I \end{pmatrix}^{-1} = \begin{pmatrix} I & H^{ab} \\ 0 & H^{bb} \end{pmatrix}^{-1}\begin{pmatrix} H^{aa} & H^{ab} \\ H^{ba} & H^{bb} \end{pmatrix}$$

which means that we can solve (10) if we can efficiently calculate the above matrices. Using the fact that (See appendix for proof.)

$$\begin{pmatrix} I & H^{ab} \\ 0 & H^{bb} \end{pmatrix}^{-1} = \begin{pmatrix} I & -H^{ab}(H^{bb})^{-1} \\ 0 & (H^{bb})^{-1} \end{pmatrix}$$

we get

$$\begin{pmatrix} \vec{X} \\ \vec{Y} \end{pmatrix} = \begin{pmatrix} I & -H^{ab}(H^{bb})^{-1} \\ 0 & (H^{bb})^{-1} \end{pmatrix}\begin{pmatrix} H^{aa} & H^{ab} \\ H^{ba} & H^{bb} \end{pmatrix}\begin{pmatrix} \vec{b} \\ 0 \end{pmatrix} \tag{11}$$

If we define $\vec{X} = \ddot{\vec{q}}_a$, we can find by block multiplications that

$$\ddot{\vec{q}}_a = [H^{aa} - H^{ab}\cdot(H^{bb})^{-1}\cdot(H^{ab})^T]\vec{b} \tag{12}$$

For fast calculation, we need to use the property that all blocks of H are sparse and band-limited. If we define

$$\left.\begin{array}{l} H^{aa}\vec{b} = \vec{c} \\ (H^{ab})^T\vec{b} = \vec{d} \\ (H^{bb})^{-1}\vec{d} = \vec{e} \\ H^{ab}\vec{e} = \vec{f} \end{array}\right\} \text{, then } \ddot{\vec{q}}_a = \vec{c} - \vec{f} \tag{13}$$

We can compute $\vec{c}$ and $\vec{d}$ efficiently because most of the entries in H^{aa} and $(H^{ab})^T$ are zeros for serial manipulators with many links. To calculate $\vec{e}$, we use the LU decomposition for H^{bb} as

$$(H^{bb})^{-1}\vec{d} = \vec{e} \longrightarrow \vec{d} = H^{bb}\vec{e} \xrightarrow{H^{bb}=L_hU_h} \vec{d} = L_hU_h\vec{e}$$

where L_h is the lower triangular matrix and U_h is the upper triangular matrix with zero entries all most parts of the matrix except the diagonal and sub(super) diagonal.
If we define

$$U_h\vec{e} = \vec{z}\text{, so that } \vec{d} = L_h\vec{z}\,,$$

we can calculate $\vec{z}$ and $\vec{e}$ in *O(n)* computations. Finally, $\vec{f}$ can also be computed in *O(n)* as can $\ddot{\vec{q}}_a = \vec{c} - \vec{f}$.

III. The Forward Dynamics for an N-link Planar Serial Manipulator

We consider a planar revolute N-link serial manipulator which has concentrated point masses at the joints as an example on which to demonstrate this algorithm. We assume that the link lengths are constants, i.e. are hard variables defining constraints, and the joint angles are control inputs, i.e. soft variables of the system. Fig. 1 shows the serial manipulator in the planar case. Note that $\vec{x}_i$ denotes the absolute position vector to the i^{th} joint from the base frame, {B}, L_i is the link length, and θ_i is the joint angle of the i^{th} link.

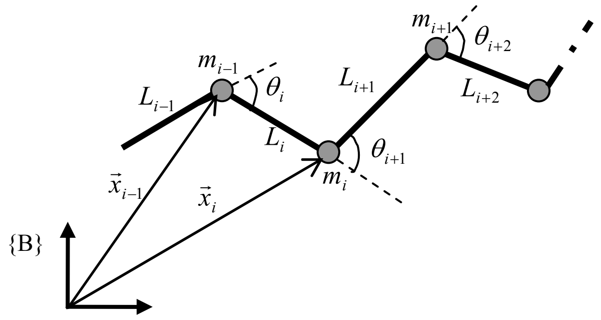

Fig.1. N-link serial manipulator

The absolute position vectors to the joint relative to the base frame are

$$\vec{x}_1 = [L_1 \cos\theta_1,\, L_1 \sin\theta_1]^T$$

and therefore

$$\vec{x}_i = \vec{x}_{i-1} + [L_i \cos(\sum_{j=1}^{i} \theta_j),\, L_i \sin(\sum_{j=1}^{i} \theta_j)]^T \quad (14)$$

for $i = 2, \cdots, N$. Also, we define $\vec{q}$ to parameterize $\vec{x}$ as:

$$\vec{q} = [\underbrace{\theta_1, \cdots, \theta_n}_{\text{soft variables}}, \underbrace{L_1, \cdots, L_n}_{\text{hard variables}}]^T$$

In this problem, link lengths, which are the constraints on the system, are expressed as:

$$L_i = \|\vec{x}_i - \vec{x}_{i-1}\| \quad \text{or} \quad (15)$$

$$L_i^2 = (\vec{x}_i - \vec{x}_{i-1}) \cdot (\vec{x}_i - \vec{x}_{i-1}) \quad (16)$$

Taking the partial derivatives of (16) with respect to $\vec{x}_i$ and $\vec{x}_{i-1}$, we find

$$L_i \frac{\partial L_i}{\partial \vec{x}_i} = (\vec{d}_i)^T \quad \text{and} \quad L_i \frac{\partial L_i}{\partial \vec{x}_{i-1}} = -(\vec{d}_i)^T \quad (17)$$

where $\vec{d}_i = \vec{x}_i - \vec{x}_{i-1}$. It is obvious that L_i does not depend on any other of the position vectors $\vec{x}_k$ for $k \neq i$ or $k \neq i-1$. Therefore, the above can be written as:

$$\frac{\partial L_i}{\partial \vec{x}_k} = \frac{1}{L_i} (\vec{d}_i)^T (\delta_{k,i} - \delta_{k,i-1}) . \quad (18)$$

Similarly, we can calculate the derivatives of joint angles with respect to absolute position vectors. If we define $\vec{x}_0 = [0\ 0]^T$ and $\vec{x}_{-1} = -L_{-1} e_1$, then θ_i is only related to the positions $\vec{x}_i$, $\vec{x}_{i-1}$, and $\vec{x}_{i-2}$ by the following equation

$$\cos\theta_i = \frac{1}{L_i L_{i-1}} (\vec{x}_i - \vec{x}_{i-1}) \cdot (\vec{x}_{i-1} - \vec{x}_{i-2}) . \quad (19)$$

Taking the partial derivatives with respect to $\vec{x}_i$, $\vec{x}_{i-1}$, and $\vec{x}_{i-2}$, we have

$$L_{i-1} L_i \frac{\partial \cos\theta_i}{\partial \vec{x}_i} = (\vec{x}_{i-1} - \vec{x}_{i-2})^T . \quad (20\text{-}1)$$

$$L_{i-1} L_i \frac{\partial \cos\theta_i}{\partial \vec{x}_{i-1}} = (\vec{x}_i + \vec{x}_{i-2} - 2\vec{x}_{i-1})^T . \quad (20\text{-}2)$$

$$L_{i-1} L_i \frac{\partial \cos\theta_i}{\partial \vec{x}_{i-2}} = (\vec{x}_{i-1} - \vec{x}_i)^T . \quad (20\text{-}3)$$

Since the above are equal to zero for all values except the cases of *j=i, i-1*, or *i-2*, it follows that we can write

$$\frac{\partial \theta_i}{\partial \vec{x}_j} = \frac{-1}{L_{i-1} L_i \sin\theta_i} [\delta_{i,j} (\vec{d}_{i-1})^T + \delta_{i-1,j} (\vec{d}_i - \vec{d}_{i-1})^T - \delta_{i-2,j} (\vec{d}_{i-1})^T] \quad (21)$$

where

$$\frac{\partial \cos\theta_i}{\partial \vec{x}_j} = -\sin\theta_i \frac{\partial \theta_i}{\partial \vec{x}_j} .$$

From the above equations, we can construct the H matrix as:

$$H = \begin{pmatrix} H^{\theta\theta} & H^{\theta L} \\ H^{L\theta} & H^{LL} \end{pmatrix} . \quad (22)$$

We can calculate elements of each block matrix from:

$$H_{ij}^{\theta\theta} = \sum_{l=0}^{N} \frac{1}{m_l} (\partial\theta_i / \partial\vec{x}_l)(\partial\theta_j / \partial\vec{x}_l)^T$$
$$H_{ij}^{\theta L} = \sum_{l=0}^{N} \frac{1}{m_l} (\partial\theta_i / \partial\vec{x}_l)(\partial L_j / \partial\vec{x}_l)^T$$
$$H_{ij}^{L\theta} = \sum_{l=0}^{N} \frac{1}{m_l} (\partial L_i / \partial\vec{x}_l)(\partial\theta_j / \partial\vec{x}_l)^T$$
$$H_{ij}^{LL} = \sum_{l=0}^{N} \frac{1}{m_l} (\partial L_i / \partial\vec{x}_l)(\partial L_j / \partial\vec{x}_l)^T \quad (23)$$

By calculating above equations, we have 4 blocks of H which are very sparse, and band-limited as shown below,

$$H^{\theta\theta} = \begin{bmatrix} * & * & * & 0 & \cdots & 0 \\ * & * & * & * & \ddots & \vdots \\ * & * & \ddots & \ddots & \ddots & 0 \\ 0 & * & \ddots & \ddots & * & * \\ \vdots & \ddots & \ddots & * & * & * \\ 0 & \cdots & 0 & * & * & * \end{bmatrix} \quad H^{\theta L} = \begin{bmatrix} * & * & 0 & \cdots & \cdots & 0 \\ * & * & * & \ddots & & \vdots \\ * & * & \ddots & \ddots & \ddots & \vdots \\ 0 & * & \ddots & \ddots & * & 0 \\ \vdots & \ddots & \ddots & \ddots & * & * \\ 0 & \cdots & 0 & * & * & * \end{bmatrix}$$
$$H^{L\theta} = \begin{bmatrix} * & * & * & 0 & \cdots & 0 \\ * & * & * & * & \ddots & \vdots \\ 0 & * & \ddots & \ddots & \ddots & 0 \\ \vdots & \ddots & \ddots & \ddots & * & * \\ \vdots & & \ddots & * & * & * \\ 0 & \cdots & \cdots & 0 & * & * \end{bmatrix} \quad H^{LL} = \begin{bmatrix} * & * & 0 & \cdots & \cdots & 0 \\ * & * & * & \ddots & & \vdots \\ 0 & * & \ddots & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & * & 0 \\ \vdots & & \ddots & * & * & * \\ 0 & \cdots & \cdots & 0 & * & * \end{bmatrix} \quad (24)$$

Finally, using the procedure of (13), we can solve the forward dynamic equation as:

$$\ddot{\theta} = (H^{\theta\theta} - H^{L\theta} (H^{LL})^{-1} H^{\theta L}) \vec{b} . \quad (25)$$

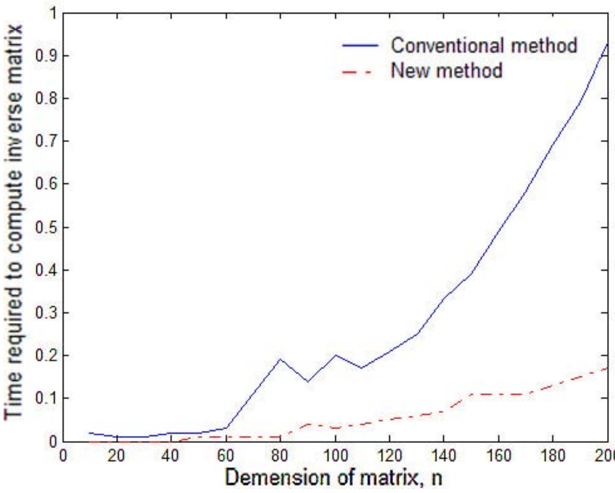

Fig. 2. Numerical results to compare the time requirements

Fig. 2 shows the numerical results to verify the efficiency of our *O(n)* algorithm. It compares the computational time required to compute the inverse of the mass matrices for different values of N (number of links) using our new method with the conventional $O(n^3)$

method. We note that our method is not effective for very small values of *N*, because the direct inversion for small dimensional matrices can be even faster than using our algorithm. However, the graph shows that the time difference between the conventional $O(n^3)$ algorithm and the new *O(n)* algorithm increases significantly as *N* become large.

IV. Discussion and Future work

A robotic manipulator is usually considered as a collection of rigid bodies connected with joints. Our algorithm for the fast inversion of the mass matrix shows that the mass matrix for such a system can be inverted efficiently by temporarily relaxing the constraints and using the band-limited and sparse nature of the inverses of the unconstrained matrices. Our implementation of this idea at the current time only addresses the specific model in which point masses are concentrated at the joints of a planar revolute manipulator. As a short term goal, we are going to extend the algorithm to the more general case of planar revolute serial manipulators composed of rigid bodies. We will then extend the method to other joint types as well as the three-dimensional case.

V. Conclusion

This paper is concerned with a new *O(n)* algorithm for the forward dynamics computation for N-link serial manipulators. The theory of the partitioned mass matrix and inverse of mass matrix was described. We considered the band-diagonal structure of the inversed mass matrix of a holonomic manipulator (i.e. no constraints), and then constraints were built in by appropriate partitioning of the inverse of the unconstrained mass matrix. The planar N-link serial manipulator with revolute joints was used to demonstrate the procedure, and the numerical results were presented to verify the *O(n)* complexity.

Appendix

We can use GE (Gaussian Elimination) for matrix inversion for block matrices if certain blocks are invertible as:

$$\left(\begin{array}{cc|cc} I & A & I & 0 \\ 0 & B & 0 & I \end{array}\right) \rightarrow \left(\begin{array}{cc|cc} I & A & I & 0 \\ 0 & I & 0 & B^{-1} \end{array}\right) \rightarrow \left(\begin{array}{cc|cc} I & 0 & I & -AB^{-1} \\ 0 & I & 0 & B^{-1} \end{array}\right)$$

Therefore,

$$\begin{pmatrix} I & A \\ 0 & B \end{pmatrix}^{-1} = \begin{pmatrix} I & -AB^{-1} \\ 0 & B^{-1} \end{pmatrix}$$

where *A* and *B* are block matrices and 0 denotes the zero matrix.

If we take derivatives of $\vec{x}$, $3n \times 1$ vector, with respect to $\vec{u}$, $n \times 1$ vector, then we get $3n \times n$ matrix as:

$$\frac{\partial \vec{x}}{\partial \vec{u}} = \begin{bmatrix} \frac{\partial \vec{x}_1}{\partial u_1} & \cdots & \frac{\partial \vec{x}_1}{\partial u_k} & \cdots & \frac{\partial \vec{x}_1}{\partial u_n} \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ \frac{\partial \vec{x}_j}{\partial u_1} & \cdots & \frac{\partial \vec{x}_j}{\partial u_k} & \cdots & \frac{\partial \vec{x}_j}{\partial u_n} \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ \frac{\partial \vec{x}_N}{\partial u_1} & \cdots & \frac{\partial \vec{x}_N}{\partial u_k} & \cdots & \frac{\partial \vec{x}_N}{\partial u_s} \end{bmatrix}$$

The following notation was used.

$$\mathrm{diag}(c_i) = \begin{bmatrix} c_1 I & 0 & \cdots & \cdots & 0 \\ 0 & \ddots & \ddots & & \vdots \\ \vdots & \ddots & c_k I & \ddots & \vdots \\ \vdots & & \ddots & \ddots & 0 \\ 0 & \cdots & \cdots & 0 & c_N I \end{bmatrix}$$

where *I* stands for 3×3 identity matrix and $i = 1, \cdots, N$.

References

[1] M. Fixman, “Classical statistical mechanics of constraints: A theorem and application to polmers,” *Proc. Nat. Acad. Sci.,* vol. 71, no. 8, August 1974.

[2] M. Fixman, “Simulation of polymer dynamics. 1. General theory,” *J. Chem. Phys*., vol. 69, no. 4, August 1978.

[3] G. Rodriguez, “Kalman Filtering, smoothing, and recursive robot arm forward and inverse dynamics’,” *IEEE J. Robotics and Automation*, vol. RA-3, no. 6, December 1987.

[4] G. Rodriguez, K. Keutz-Delgado, “Spatial operator factorization and inversion of the manipulator mass matrix,” *IEEE Trans. Robotics and Automation*, vol. 8, no. 1, February 1992.

[5] A. Jain and G. Rodriguez, “Recursive flexible multibody system dynamics using spatial operators,” *J. Guidance, Control and Dynamics*, vol. 15, pp. 1453-1466, November 1992.

[6] R. Featherstone, “The calculation of robot dynamics using articulate-ed-body inertia,” *Int. J. Robotics Res*., vol 2, no. 1, Spring 1983

[7] R. Featherstone, “Efficient factorization of the joint space inertia ma-trix for branched kinematic trees,” submitted to *Intl. Journal of Robotics Research*, 2004.

[8] S.K. Saha, “Analytical expression for the inverted inertia matrix of serial robots,” *Int. J. Robotics Research*, vol. 18, no. 1, January 1999.

[9] S.K. Saha, “A decomposition of the manipulator inertia matrix,” *IE EE Trans. Robot. Automat.* 13(2):301-304.

[10] K.S. Anderson and J. H. Critchley, “Improved ‘Order-N’ performan-ce algorithm for the simulation of constrained multi-rigid-body dynamic systems,” *Multibody system dynamics* 9: 185-212, 2003.

[11] Vereshchagin, A.F., “Gauss principle of least constraint for modeling the dynamics of automatic manipulators using a digital computer,” *Soviet Physics-Doklady* 20(1), 1975.

[12] R.E. Kalman, “A new approach to linear filtering and prediction pro-blems,” *ASME Trans. J. Basic Eng.,* vol. D, pp. 35-45, Mar. 1960.

[13]A.E. Bryson, Jr., and M. Frazier, “Smoothing for linear and nonlinear dynamic systems,” *Proc. Optimum Sys. Synthesis Conf.*, U.S. Air Force Tech. Rep. ASD-TDR-63-119, Feb. 1963.

[14]J.M. Hollerbach, “A recursive Lagrangian formulation of manipulator dynamics and a comparative study of dynamics formulation complexity,” *Tutorial on Robotics*, C.S.G. Lee, R.C. Gonzalez, K.S. Fu, eds., IEEE Computer Society Press, Silver Spring, Maryland, 1983, pp. 111-117.

[15] J.Y.S. Luh, M.W. Walker, and R.P. Paul, “On-line computational scheme for mechanical manipulators,” *Transactions of the ASME Journal of Dynamic Systems, Measurement, and Control*, 1980.

[16] J. Angeles and O. Ma, “Dynamic simulation of n-axis serial robotic manipulators using a natural orthogonal complement,” *Int. J. Robot. Res*. 7(5), 1998, pp.32-47.