Comprehensive Theory of Differential Kinematics and Dynamics for Motion Optimization

Ko Ayusawa and Eiichi Yoshida
CNRS-AIST JRL, UMI3218/RL, National Institute of Advanced Industrial Science and Technology (AIST), Japan.
EMail: {k.ayusawa, e.yoshida}@aist.go.jp

Abstract—This paper presents a novel unified theoretical framework for differential kinematics and dynamics for complex robot motion optimization. By introducing $18 \times 18$ comprehensive motion transformation matrix (CMTM), forward differential kinematics and dynamics including velocity and acceleration can be written in a simple chain product like ordinary rotational matrix. This formulation enables analytical computation of derivative of various physical quantities including joint force or torques with respect to joint coordinate variables and their derivatives for a robot trajectory in an efficient manner ($O(N_J)$, where $N_J$ is the number of the robot's DOF), which is useful for motion optimization.

I. INTRODUCTION

In recent robotics research, optimization has been increasing its importance in many aspects. Classic inverse kinematics and inverse dynamics problem [19] can be handled as an optimization problem of joint coordinates or torques under some constraints which are derived from physical consistency or desired tasks. Some practical optimization frameworks are proposed and applied to not only motion planning and control of a humanoid robot [25, 8], but also to motion reconstruction [26], contact estimation, and musculoskeletal analysis [7, 20] of a digital human model. Model identification problem [13] is also an optimization problem about model parameters with inverse dynamics computation. In these basic problems, only one type of physical quantities is optimized; inverse kinematics computes joint coordinates, inverse dynamics computes joint torques, identification computes inertial parameters, etc.

However, practical problems are usually their combinations and different physical quantities need to be simultaneously optimized through several time instances. Trajectory optimization under physical consistent conditions is a typical example. In locomotion planning, as violation of conditions at a certain time instance leads to future risk of falls, several sets of joint coordinates during certain period often need to be optimized simultaneously in order to predict future risks. Since this optimization usually requires huge computation cost, the balancing problem is often simplified, for example, by utilizing low-dimensional model [12]. Another example is kinematic calibration of human body segments [15] from motion capture measurement since the joint angles cannot be measured directly by encoders unlike standard robot calibration [13]. It requires simultaneous optimization of the geometric parameters and the generalized coordinates [2].

More recently, application of optimization has been intensively investigated in the field of anthropomorphic systems: humanoid robotics and human motion analysis. Humanoid robots are expected to execute more complicated and practical tasks such as disaster response [16], evaluation of human oriented products as physical human simulator [18], etc. In such applications, anthropomorphic motion optimization faces far more complicated problems combining modeling, kinematics, dynamics, planning, and control. For instance, inertial parameters identification of a humanoid robot is important to realize precise and dynamic control [3]. However, the optimal motion generation to maximize the total identification performance is a complex problem of trajectory optimization with balancing problem [4]. In the humanoid application as an active dummy for assistive device evaluation, imitation of human-like motion is necessary. This technique, called motion retargeting [10, 22], usually involves inverse kinematics problem of both human and humanoid, identification of the morphing function, and motion control of a robot with considering physical consistency. Yet another example is human simulators: recent detailed simulators are often connected to the other simulation systems like deformation computation such as FEM [1]. Those simulators are often used to optimize motion of digital human or parameters of a product to be designed with the simulator. In such problems, we usually solve the simultaneous problem of modeling, kinematics, and dynamics problems [4]. However, the above issues are currently difficult to be solved, and some of them are still open problems.

In order to establish a comprehensive optimization framework that can handle critical issues required from the computational and practical point of view, the partial derivative of any physical quantities with respect to the joint coordinates is indispensable when evaluating various types of conditions represented by the coordinates, derivatives, and forces of both Cartesian and joint space. Suleiman, et al [25] developed the fundamental framework of humanoid motion optimization. It utilizes the works of Park, et al [21] and Sohl, et al [23] and formulates the analytical partial derivative of Cartesian coordinates, their derivative, and joint torques with respect to the joint coordinates and their derivatives. In spite of its possibility of extension to handle many types of problems mentioned above, there are still two issues. First, as the formulation mainly focuses on the manifold of Cartesian spaces, it is difficult to handle a free-floating base and spherical joints that are often used in human and humanoid kinematics modeling. The second issue is the computational complexity; if we compute the partial derivative of joint torque of floating-
base side, the computational complexity is proportional to the square of the degree of freedom (DOF), i.e. $O(N_J^2)$, where $N_J$ is the number of DOF. It leads to huge computational cost of optimization when dealing with a large-DOF system.

In this paper, we reformulate the differential kinematics and dynamics in order to perform fast computation of the analytical partial derivative of Cartesian variables and generalized forces with respect to the joint coordinates and their derivatives. We introduce a 18-dimensional Comprehensive Motion Transformation Matrix (CMTM) in order to formulate the standard forward differential kinematics problem. This formulation makes it possible to reduce the computation of differential forward kinematics of kinematic chain to a simple chain product of the matrices in the similar manner as standard rotational matrix, or the 6 dimensional matrix used in adjoint map on SE(3) \[21\]. The CMTM also allows formulating an analytical form of several partial derivatives with respect to the joint coordinates and their derivatives including different types of joints. This partial derivative of link variables is extended form of basic Jacobian \[14\], and can be derived by the same formulation to introduce basic Jacobians. The Jacobian of joint torque is also extended form of linear/angular momentum Jacobian \[11, 24\], which is also formulated in the same manner thanks to the CMTM. In addition, each computational cost of new Jacobians is $O(N_J)$.

II. MOTION OPTIMIZATION FRAMEWORK

This section presents the overview of motion optimization problem and the flow of the computation. Let the generalized coordinates of a robot be $\mathbf{q}$, with their trajectories parameterized by $\mathbf{a}$ and time instance $t$: $\mathbf{q}(\mathbf{a}, t)$. The trajectories are represented by, for example, polynomial interpolation, Fourier series, or B-splines, etc. Their derivatives $\dot{\mathbf{q}}$ and $\ddot{\mathbf{q}}$ are computed with $\mathbf{a}$ and $t$ according to the implemented trajectory parameterization.

Let us concatenate $\mathbf{q}$, $\dot{\mathbf{q}}$, and $\ddot{\mathbf{q}}$ into $\mathbf{x}$, and consider physical quantities $\mathbf{y}$ which are represented by $\mathbf{x}$. The candidates of $\mathbf{y}$ are, for example, the position, orientation, linear and angular velocity, linear and angular acceleration of each link coordinate, the joint torques and the constraint forces acting on the joint coordinates, etc. Let $\mathbf{y}_{ij}$ be $i$-th quantity at $j$-th time instance $t_j$, $\mathbf{x}_j$ be the coordinates and their derivatives at $t_j$, and $\mathbf{Y}$ are the whole set of the quantities to be evaluated. In this paper, the set of time instance $t_j$ is given and constant, and the following optimization problem to be solved.

$$ \min_{\mathbf{a}} \ c(\mathbf{Y}) $$
$$ \text{subject to } \forall \ k \quad g_k(\mathbf{Y}) \leq 0 $$

where, $c$ is the cost function to be evaluated, and $g_k$ is $k$-th inequality constraint. Since equality constraints can be represented by two inequality constraints, they are summarized and represented by inequality form.

The above optimization problem is usually computationally expensive. In order to accelerate the computational speed, the efficient optimization techniques usually require analytical gradient computation of the cost function and each constraint. The gradient can be decomposed as follows:

$$ \frac{\partial h}{\partial \mathbf{a}} = \sum_i \sum_j \frac{\partial h}{\partial \mathbf{y}_{ij}} \frac{\partial \mathbf{y}_{ij}}{\partial \mathbf{x}_j} \frac{\partial \mathbf{x}_j}{\partial \mathbf{a}} (h = c \ or \ g_k) $$

The gradient $\partial c/\partial \mathbf{y}_{ij}$ is determined by the form of the evaluation function. The partial derivative $\partial \mathbf{x}_j/\partial \mathbf{a}$ can be computed from the implemented trajectory parameterization. The term $\partial \mathbf{y}_{ij}/\partial \mathbf{x}_j$ means the partial derivative of several types of quantities of multi-body system with respect to the joint coordinates and their derivatives. The typical example is the partial derivative of the position and orientation of each link with respect to the joint coordinates which are known as the basic Jacobian \[14\]. In this case, the derivatives of the velocities, the accelerations, the joint torques with respect to $\mathbf{q}$, $\dot{\mathbf{q}}$, $\ddot{\mathbf{q}}$ are required. By utilizing the analytical formulations for manipulators \[21, 23\], the motion optimization framework of Eq. \[1\] was applied for a humanoid robot by Suleiman, et al \[25\]. Though it can have the possibility to handle many types of motion optimization, there still remain theoretical and practical issues in order to solve the motion optimization for humanoid systems.

First, the formulation should be extended for spherical joints, free-floating base or other types of joints in order to be applied to anthropomorphic systems, because the formulations are originally for manipulators. The second issue is the computational complexity of the computation. The formulations in \[25\] basically utilize the classical recursive formula of forward kinematics and inverse dynamics \[17\]. The derivatives with respect to the coordinates of one joint are computed according to the recursive formula, and it is applied for every joint coordinates. Therefore, when computing the partial derivative of the variables of one link, the computational complexity is almost $O (N_J^2)$, where $N_J$ is the number of DOF.

As mentioned above, one typical example of $\partial \mathbf{y}_{ij}/\partial \mathbf{x}_j$ is the basic Jacobian, and its computational cost of standard basic Jacobian is $O (N_J)$. This paper introduces $18 \times 18$ matrix which can represent the forward kinematics computation including velocities and accelerations by simple chain products. The matrix has same features of $6 \times 6$ transformation matrix which represents position and orientation as mentioned later. By utilizing this matrix together with the formulation of deriving basic Jacobians, the computation method of arbitrary $\partial \mathbf{y}_{ij}/\partial \mathbf{x}_j$ is introduced in this paper. When computing the derivatives of joint torques, the derivation form of COM Jacobians \[24\] are utilized. In the formulation, several types of joints also can be handled, and its computational complexity is $O (N_J)$.

III. MATHEMATICAL NOTATIONS AND COMPREHENSIVE MOTION TRANSFORMATION MATRIX

This section presents the preliminary notation of variables in the paper, and introduces a useful matrix in order to represent the forward kinematics computation including velocity and acceleration of multi-body system.
A. Definitions of basic geometric and mechanical variables

- **O**, and **E** are \( n \times n \) zero and identity matrix respectively.
- Skew operator is represented as follows:
  \[
  [x \times] \triangleq \begin{bmatrix}
  0 & -x(3) & x(2) \\
  x(3) & 0 & -x(1) \\
  -x(2) & x(1) & 0
  \end{bmatrix}
  \]  
  (3)
- The equations of motion of a rigid body are:
  \[
  \begin{align*}
  f_c &= O_3 \left[ f \times \right] \\
  J_0 &\dot{\omega} + J_0 \omega \times \omega = \left[ f \times \right] \\
  \end{align*}
  \]  
  (4)
- Linear and angular velocities are concatenated and defined as the following vector:
  \[
  \nu \triangleq [\nu^T \omega^T]^T 
  \]  
  (7)
- The position and orientation matrix of the coordinate system of a rigid body are \( p \) and \( R \) respectively.
- Let \( \omega \) and \( \nu \) be the linear and angular velocity represented by the local coordinate respectively, and the following relationship holds.
  \[
  \nu = \dot{R}^T \dot{p} 
  \]  
  (5)
- The 6\( \times \)6 transformation matrix of position \( p \) and orientation \( R \) is defined as follows:
  \[
  A(p, R) \triangleq \begin{bmatrix}
  R & [p \times]R \\
  O_3 & R
  \end{bmatrix}
  \]  
  (6)
- Linear and angular velocities are concatenated and defined as the following vector:
  \[
  [\nu_\ast] \triangleq \begin{bmatrix}
  [\omega \times] \\
  O_3 \\
  [\omega \times]
  \end{bmatrix}, \quad [\nu_\ast \nu_\ast] \triangleq \begin{bmatrix}
  [\nu] \\\n  [\nu] \\\n  [\nu]
  \end{bmatrix}
  \]  
  (8)
- The above operator satisfies the binary operation axioms in Appendix A. The followings also hold:
  \[
  A = A(\nu_\ast) \\
  A(\nu_\ast)A^{-1} = [(A\nu_\ast)_\ast]
  \]  
  (9, 10)
- Inertial properties of a rigid body consist of mass \( m \), center of mass \( c \), and inertia tensor \( I_c \). They can be summarized as the following 6\( \times \)6 matrix:
  \[
  M \triangleq \begin{bmatrix}
  mE_3 & m[c \times] \\
  m[c \times] & I_c + m[c \times][c \times]^T
  \end{bmatrix}
  \]  
  (11)
- Let the inertial forces of a rigid body be \( \tilde{f} \) and the moment around its coordinate be \( n \). They are represented by global frame. Then, let us define 6-axis force \( f \) represented by the local coordinate as follows:
  \[
  f \triangleq \begin{bmatrix}
  R \tilde{f}^T \\
  Rn^T
  \end{bmatrix}^T
  \]  
  (12)
- The equations of motion of a rigid body are:
  \[
  f = M\ddot{\nu} - [\nu]^T M\nu
  \]  
  (13)
- The variation of the equation of motion of a rigid body is written as follows by using matrix \( D \).
  \[
  \delta f = M\delta \nu - D\delta \nu
  \]  
  (14)
  \[
  D \triangleq -[(M\nu)_\ast] - [\nu]^T M
  \]  
  (15)
- Operation \( [\cdot] \) is defined as follows:
  \[
  [f, \nu, \omega] = \begin{bmatrix}
  O_3 & \left[ f \times \right] \nu \\
  J_0 & \left[ f \times \right] \omega
  \end{bmatrix}, \quad [f, f_1] = f_2 \Rightarrow [\dot{f}_2; \dot{f}_1]
  \]  
  (16)

B. Comprehensive motion transformation matrix (CMTM)

Let us define the following new 18\( \times \)18 matrix \( X \) and call it comprehensive motion transformation matrix (CMTM).

\[
X(A, \nu, \omega) \triangleq \begin{bmatrix}
X_1 & O_6 & O_6 \\
X_2 & X_1 & O_6 \\
X_3 & X_2 & X_1
\end{bmatrix}
\]  
(17)

The following variation of 18 dimensional vector is defined as follows:

\[
\delta x \triangleq [\delta \alpha^T \delta \nu^T \delta \omega^T]^T
\]  
(18)

where variation \( \delta \alpha \) has the following relationship.

\[
[(\delta \alpha)_\ast] = A^{-1}(\delta A)
\]  
(19)

Vector \( \delta x \) is the concatenated vector of the variation of standard 6 dimensional coordinates, velocities, and accelerations. In order to handle the differential operation of matrix \( X \), the following variation of 18 dimensional vector is newly defined as follows:

\[
\delta \xi \triangleq \begin{bmatrix}
\delta \alpha^T & \delta \nu^T & \delta \omega^T
\end{bmatrix}^T
\]  
(20)

where,

\[
\delta \xi \triangleq \delta \nu + [(\delta \alpha)_\ast] \nu
\]  
(21)

\[
\delta \eta \triangleq \frac{1}{2} [\delta \nu + [(\delta \alpha)_\ast] \nu + [(\delta \xi)_\ast] \nu]
\]  
(22)

For the convenience of explanation, let us summarize the above equations as follows:

\[
\delta \xi = S\delta x
\]  
(23)

Matrix \( S \) transforms variation \( \delta x \) into that of that of new vector \( \delta \xi \), and is written as follows:

\[
S(\nu, \omega) \triangleq \begin{bmatrix}
E_6 & O_6 & O_6 \\
\dfrac{1}{2} (\nu_\ast) & E_6 & O_6 \\
\dfrac{1}{2} (\nu_\ast) & \dfrac{1}{2} (\nu_\ast) & 2E_6
\end{bmatrix}
\]  
(24)

The inverse matrix of \( S \) always exists, and is computed as follows:

\[
S^{-1} = \begin{bmatrix}
E_6 & O_6 & O_6 \\
\nu_\ast & E_6 & O_6 \\
\nu_\ast & \nu_\ast & 2E_6
\end{bmatrix}
\]  
(25)

Although the variation \( \delta x \) is what we are familiar with in robotic analysis, its usage makes the forthcoming analysis of Jacobian matrices intractable. By using the newly defined variation \( \delta \xi \), the analysis becomes easier and clearer as shown later on, and once the Jacobian is derived it can be always transformed back to \( \delta x \) by the matrix \( S \).
Appendix A, which can be easily checked. In addition, the mathematical features as matrix chain, and important formulas about kinematics and dynamics.

C. Definition and formulas of kinematics chain

The kinematic chain is tree-structured, and the indices are chosen from the base link to the end of branches. Let us now define the following matrix and operator:

\[
\delta \xi^* \triangleq \begin{bmatrix} \delta \alpha_1 & 0_6 & 0_6 \\ \delta \zeta & \delta \alpha_2 & 0_6 \\ \delta \eta & \delta \alpha_3 & \delta \alpha_4 \end{bmatrix}
\] (26)

\[
\delta \xi_1 \cdot \delta \xi_2 \triangleq [\delta \xi^*] \delta \xi_2
\] (27)

By utilizing the above operator, the following relationship holds:

\[
\delta X = X[(\delta \xi)^*]
\] (28)

It can be check by computing each block matrix of \(\delta X_i\):

\[
\delta X_1 = \delta A = A[\delta \alpha_1] = X_1[\delta \alpha_1]
\] (29)

\[
\delta X_2 = \delta [v] + A[\delta v] = X_1[\delta \zeta] + X_2[\delta \alpha_2]
\] (30)

\[
\delta X_3 = \frac{1}{2} A[\{\delta v_1 + [\delta v_2, v] + [v_1, \delta v_2] \}
\]

\[
= X_1[\delta \eta] + X_2[\delta \alpha_3] + X_3[\delta \alpha_4]
\] (31)

Eq. (28) has the same form as Eq. (9). Actually, operator \([\delta \xi_1 \cdot \delta \xi_2]\) can satisfy from the binary operation axioms in Appendix A, which can be easily checked. In addition, the following equation also holds.

\[
X[\delta \xi^*]X^{-1} = (X \delta \xi)^*
\] (32)

The set of matrix \(X\) and operator \((\cdot)\) has the similar mathematical features as matrix \(A\) and \((\cdot)\) (i.e. the set of Adjoint map and Lie bracket operator). It means that many formulas of kinematics operations about position and orientation can be replaced with those including velocities and accelerations. Therefore, matrix \(X\) can comprehensively handle the kinematics transformation about motion.

C. Definition and formulas of kinematics chain

This sub-section shows the notation of open kinematic chain, and important formulas about kinematics and dynamics.

- The kinematic chain is tree-structured, and the indices are chosen from the base link to the end of branches.
- \(p(i)\) is the index of a root-side link connected to link \(i\).
- \(C(i)\) is the set of indices of leaves-side links connected to link \(i\).
- \(\mathcal{C}(i)\) is the set of all leaves-side links recursively connected to link \(i\).
- \(\mathcal{P}(i)\) is the set of all root-side links recursively connected to link \(i\).
- Let us define the following sets \(\mathcal{P}(i) \triangleq \{i, \mathcal{P}(i)\}, \mathcal{C}(i) \triangleq \{i, \mathcal{C}(i)\}\)
- \(s_{j(i)}\) is the following selection function:

\[
s_{j(i)} = \begin{cases} 1 & (i \in \mathcal{P}(j)) \\ 0 & (\text{others}) \end{cases}
\] (33)

- Let us represent quantity \(y\) of link \(i\) such as \(y_i\).
- Let us denote \(y'_j\) as the relative variable of \(y\) from link \(i\) to \(j\).

- \(\mathbf{q}_i\) is \(n_J\) set of joint variables (angles), where \(n_J\) is the number of DOF of joint \(i\), and the followings relationship holds between the joint variables and the relative velocities between link \(i\) and \(p(i)\):

\[
A^{i(i)}_p = e((K_i \mathbf{q}_i) \cdot)
\] (34)

- Matrix \(K_i\) is \(6 \times n_J\) constant matrix defined according to the type of joint \(i\). For example, if joint \(i_1\) has \(z\)-axis rotational joint and joint \(i_2\) has a spherical joint, the corresponding matrices are followings:

\[
K_{i_1} = [0 \ 0 \ 0 \ 0 \ 0 \ 1]^T, \ K_{i_2} = [O_3 \ E_3]^T,
\] (35)

- \(\delta \theta_i\) is a variance defined in the tangent vector space of \(A^{i(i)}_p\), which has the following differential relationship.

\[
\delta A^{i(i)}_p = A^{i(i)}_p [\delta \alpha^{i(i)}_p \cdot] = A^{i(i)}_p ([K_i \delta \theta_i] \cdot)
\] (36)

As a note, in the case of spherical joints or free floating joints, \(\delta \theta_i\) is not equal to \(\delta q_i\) because of Eq. (36).

- \(\psi_i\) means the joint velocity variables, and the following equations hold between \(\psi_i\) and the relative coordinates:

\[
\dot{\mathbf{q}}_i = K_i \psi_i
\] (37)

- \(f_j^{(p)}\) means the constraint force of joint \(j\), which has the following relationship with inertial force \(f_j\) of link \(j\) and its links connected in leaves-side.

\[
f_j^{(p)}(j) = f_j + \sum_{k \in C(j)} A^{k(j)-1} f_k
\] (38)

where \(j = p(k)\) holds due to \(k \in C(j)\).

- \(j_j\) represents \(n_J\) dimensional vector of joint torque, and can be extracted from 6 dimensional vector of joint constraint forces \(f_j^{(p)}\) as follows:

\[
j_j = K_j^T f^{(p)}_j
\] (39)

D. CMTMs in kinematic chain

Let us consider the following chain product among CMTMs:

\[
X_j = X_i X'_j
\] (40)

The component of \(X_i\) is written down into the followings:

\[
X_{1j} = X_{1i} \ X'_1
\] (41)

\[
X_{2j} = X_{1i} \ X'_2 + X_{2i} \ X'_1
\] (42)

\[
X_{3j} = X_{1i} \ X'2 + X_{2i} \ X'_3 + X_{3i} \ X'_1
\] (43)

Now let us see Eq. (41), Eq. (42), and Eq. (43) respectively. From Eq. (17), Eq. (41) can be transformed into:

\[
A_i = A_i A'_j
\] (44)

Therefore, Eq. (41) means the standard forward kinematics operation.

Next, Eq. (42) can be converted into:

\[
A_j [\mathbf{v}_j, \cdot ] = A_j [A'_j [\mathbf{v}_j, \cdot ] + A_i [\mathbf{v}_i, \cdot ] A_j'
\]

\[
= A_j ([\mathbf{v}_j + A_j' A_j^{-1} \mathbf{v}_i, \cdot ])
\] (45)
From the above equation, the following holds identically.

\[ \mathbf{v}_j = \mathbf{v}'_j + A^j_{i-1} \mathbf{v}_i \]  

(46)

where, Eq. (46) indicates that Eq. (42) means the differential forward kinematics operation about linear and angular velocities.

Finally, Eq. (43) can be written down as follows:

\[
\begin{align*}
1/2 A_j ([\mathbf{v}_j, + |\mathbf{v}_j,|^2] &= 1/2 A_i ([\mathbf{v}_i, + |\mathbf{v}_i,|^2]) + A_j [\mathbf{v}_i, A_j [\mathbf{v}_j,] \\
&= 1/2 A_j ([\mathbf{v}^j + A^j_{i-1} \mathbf{v}_i, + (A^j_{i-1} \mathbf{v}_i),] + |\mathbf{v}_j,|^2) \\
&= 1/2 A_j ([\mathbf{v}^j + A^j_{i-1} \mathbf{v}_i, + (A^j_{i-1} \mathbf{v}_i),] + |\mathbf{v}_j,|^2) \\
&= \sum_{k \in \mathcal{P}(j)} \delta X_{j}(\delta \chi_{k})^{*} = \sum_{k \in \mathcal{P}(j)} X_{p(k)}^{j} X_{p(k)}^{j} (54)
\end{align*}
\]

Therefore, the following holds identically.

\[ \mathbf{v}_j = \mathbf{v}'_j + A^j_{i-1} \mathbf{v}_i + (A^j_{i-1} \mathbf{v}_i), \mathbf{v}'_j \]  

(47)

where, Eq. (48) implies that Eq. (43) is equivalent with the differential forward kinematics operation about linear and angular accelerations.

As can be seen from all the above, the chain products of CMTMs in Eq. (40) represent the standard and differential kinematics computation of a kinematic chain. The formulation using CMTM can therefore handle the kinematics operations in a comprehensive manner. This feature becomes a strong advantage when introducing arbitrary Jacobian matrices in the next section.

Now let us define the following variation of 3nli dimensional vector consisting of the variations of joint variable, velocity, and its derivative:

\[ \delta \chi \triangleq [\delta \theta^T \delta \psi^T \delta \psi^iT]^T \]  

(49)

According to Eq. (20), Eq. (50), and Eq. (51), the relationship between \( \delta \chi^{(i)} \) and \( \delta \chi_i \) is summarized as follows:

\[ \delta \chi^{(i)} = G_i \delta \chi_i \]  

(50)

The relationship among the variables of link and joint coordinates and their variations is summarized in Fig. 1.

![Fig. 1. Overview of the relationship among the variables used in the paper](image)

IV. COMPUTATION OF ARBITRARY JACOBINS

This section shows the different types of arbitrary partial derivatives \( \partial \mathbf{y}_{i,j}/\partial \mathbf{x}_j \) used in Eq. (2) by utilizing CMTM. Though \( \partial \mathbf{y}_{i,j}/\partial \mathbf{x}_j \) are strictly speaking not true Jacobian matrices as in the case of basic Jacobians, however, in this paper, let us call them Jacobians for descriptive purposes.

A. Jacobians of link posture, velocity and acceleration

Let us compute matrix \( J_i \) that converts variation \( \delta \chi_{all} \) of all joints to variation \( \delta \chi_j \) of link \( j \) as follows:

\[ \delta \chi_j = J_i \delta \chi_{all} = \sum_{k \in \mathcal{P}(j)} \delta X_{p(k)} \]  

(51)

As mentioned in the previous section, matrix \( X \) has the same features as \( A \), and matrix \( J \) can be computed in a similar manner when computing standard basic Jacobian matrix.

Now let us consider \( X_j \) of link \( j \). The following chain products hold among CMTMs:

\[ X_j = X_{p(k)} X_{p(k)}^{j} \]  

(52)

The variation of \( X_j \) can be computed according to chain products Eq. (52):

\[ \delta X_j = \sum_{k \in \mathcal{P}(j)} X_{p(k)} \delta X_{p(k)}^{j} X_{p(k)}^{j} \]  

(53)

By utilizing Eq. (28) and Eq. (32), the above equation can be transformed into:

\[ X_j (\delta \chi_{j})^* = \sum_{k \in \mathcal{P}(j)} X_{p(k)} X_{p(k)}^{j} (\delta \chi_{p(k)}) (54) \]

According to the above equation, the following equation holds identically.

\[ \delta \chi_j = \sum_{k \in \mathcal{P}(j)} X_k \delta \chi_k \]  

(55)

The coefficient matrix of Eq. (55) means the Jacobian matrix with respect to \( \delta \chi_j \), and each block matrix is equal to relative CMTM \( \delta \chi^{(i)} \).

Since the desired Jacobian matrix is the one with respect to \( \delta \chi_j \), let us compute it by transforming variations of Eq. (55) from \( \delta \chi_j \) to \( \delta \chi \) and from \( \delta \chi \) to \( \delta \chi_j \).

First, by utilizing Eq. (23), the following equation is obtained:

\[ \delta \chi_j = \sum_{k \in \mathcal{P}(j)} \delta \chi_k \]  

(56)

\[ \delta \chi_j \triangleq S_{j}^{-1} \delta \chi_{j}^{(i)} \]  

(57)
The next transformation can be performed with Eq. (50) as follows:

$$\delta x_j = \sum_k J_{(j,k)} \delta \chi_k^{(p)} \tag{58}$$

$$J_{(j,k)} \triangleq s_{(j,k)} \tilde{X}_j G_k$$

From the above, the Jacobian matrix shown in Eq. (51) was finally derived as Eq. (52). Since the direct computation of 18 × 18 matrix products in Eq. (57) is computationally inefficient, the solution of 6 × 6 block matrices in $\tilde{X}_j$ is written down in Appendix B.

**B. Jacobians of link inertial forces**

This subsection derives matrix $L_j$ which converts variation $\delta \chi_{all}$ of all joints to force variation $f_j$ of link $j$ as follows:

$$\delta f_j = L_j \delta \chi_{all} = \sum_k L_{(j,k)} \delta \chi_k \tag{60}$$

Let us first consider the variation of equations of motion of link $j$. According to Eq. (12), the following equations are obtained:

$$\delta f_j = H_j \delta x_j \tag{61}$$

$$H_j \triangleq \begin{bmatrix} O_b & D_j & M_j \end{bmatrix} \tag{62}$$

From Eq. (61) and Eq. (51), variation $\delta \chi_j$ can be transformed into $J_{(j,k)}$. The above equation can be also transformed into the followings:

$$\delta f_j = H_j \sum_i J_{(j,k)} \delta \chi_k \tag{63}$$

Therefore, the Jacobian matrix shown in Eq. (60) could be derived as followings:

$$L_{(j,k)} = H_j J_{(j,k)} \tag{64}$$

For the sake of forthcoming discussion, let us transform Eq. (61) to the linear form with respect to $\delta \xi_j$ by using Eq. (23).

$$\delta f_j = H_j S_j^{-1} \delta \xi_j \tag{65}$$

**C. Jacobians of joint constraint forces**

In this subsection, we compute matrix $N_j$ which converts variation $\delta \chi_{all}$ of all joints to force variation $f_j^{(p)}$ of link $j$ as follows:

$$\delta f_j^{(p)} = N_j \delta \chi_{all} = \sum_k L_{(j,k)} \delta \chi_k \tag{66}$$

From Eq. (59), the following equation about joint constraint force $f_j^{(p)}$ can be obtained.

$$\delta f_j^{(p)} = \delta f_j + \sum_{k \in C(j)} (A_k^{(p)} - T \delta f_k^{(p)} + \delta A_k^{(p)} - T f_k^{(p)}) \tag{67}$$

According to Eq. (65), the above equation can be transformed into:

$$\delta f_j^{(p)} = (H_j S_j^{-1}) \delta \xi_j + \sum_{k \in C(j)} (A_k^{(p)} - T \delta f_k^{(p)} + \delta A_k^{(p)} - T f_k^{(p)}) \tag{68}$$

On the other hand, the following equation holds from Eq. (53).

$$\delta \xi_j = X_p^{j(p)} \left( \sum_{i \in \beta(p)(j)} X_i^{(p)} \delta \xi_i^{(p)} \right) + \delta \xi_j^{(p)} \tag{69}$$

Since $A^{-T}$ and $X$ are transformation matrices, Eq. (69) and Eq. (68) have the same form of Eq. (90) and Eq. (91) in Appendix C respectively. According to the similar derivations of Eq. (92), Eq. (93), and Eq. (95) in Appendix C, the following recursive formula can be obtained.

$$\delta f_j^{(p)} = \begin{bmatrix} \delta f_j \\ \delta f_{j}^{(p)} \end{bmatrix} = \begin{bmatrix} \delta H_j^{(p)} j & \delta f_{j}^{(p)} \end{bmatrix} \begin{bmatrix} \delta \xi_j \\ \delta \xi_j^{(p)} \end{bmatrix} \tag{70}$$

where,

$$\delta f_{j}^{(p)} = \begin{bmatrix} \delta f_{j} \\ \delta f_{j}^{(p)} \end{bmatrix} = \begin{bmatrix} \delta \xi_j \\ \delta \xi_j^{(p)} \end{bmatrix} \tag{71}$$

$$\delta \xi_j = X_p^{j(p)} \left( \sum_{i \in \beta(p)(j)} X_i^{(p)} \delta \xi_i^{(p)} \right) + \delta \xi_j^{(p)} \tag{72}$$

It should be noted that, if the set of $A^{-T}$ and $X$ is replaced with $A^{-T}$ and $A$, and if there exists no bias terms like $\delta \xi_j^{(p)}$, the transformation based on Appendix C shows the same formula when introducing linear and angular momentum Jacobian.

Let us expand the term of $A_k^{(p)} - T \delta h_k$ in Eq. (72) by its recursive computation toward leaves-link side. In addition, by the transformation from $\delta \xi_j^{(p)}$ to $\delta x$ with Eq. (23), Eq. (72) can be transformed into:

$$\delta f_j^{(p)} = \begin{bmatrix} \delta f_j \\ \delta f_{j}^{(p)} \end{bmatrix} = \begin{bmatrix} \delta H_j^{(p)} j & \delta f_{j}^{(p)} \end{bmatrix} \begin{bmatrix} \delta \xi_j \\ \delta \xi_j^{(p)} \end{bmatrix} \tag{73}$$

There also exists the following conversion of variation $\delta A_k^{(p)} - T f_k^{(p)}$ as follows:

$$\delta A_k^{(p)} - T f_k^{(p)} = -A_k^{(p)} - T [\delta A_k^{(p)}] f_k^{(p)} = \begin{bmatrix} \delta A_k^{(p)} - T f_k^{(p)} \end{bmatrix} \tag{74}$$

According to Eq. (73) and equation $X_{(k,k)} = S_k^{(p)}$ (which is from Eq. (57)), Eq. (72) can be converted to:

$$\delta f_j^{(p)} = \begin{bmatrix} \delta f_j \\ \delta f_{j}^{(p)} \end{bmatrix} = \begin{bmatrix} \delta H_j^{(p)} j & \delta f_{j}^{(p)} \end{bmatrix} \begin{bmatrix} \delta \xi_j \\ \delta \xi_j^{(p)} \end{bmatrix} \tag{75}$$

where,

$$T \triangleq \begin{bmatrix} E_b & O_b & O_b \end{bmatrix} \tag{76}$$

By substituting Eq. (71) and Eq. (75) for Eq. (70), and with converting the variations from $\delta x_k^{(p)}$ to $\delta \chi_k^{(p)}$, the following
equation holds.

\[
\delta f_j^{(l)} = \sum_{k \in \mathcal{J}(j)} \hat{H}_j \mathcal{J}(j,k) \delta \mathbf{x}_k^{(l)} + \sum_{k \in \mathcal{J}(j)} \mathcal{A}_k^{-T} (\hat{H}_k \mathcal{J}(k,k) - [f_k^{(l)}] \mathcal{T} G_k) \delta \mathbf{x}_k^{(l)} 
\]

\begin{equation}
\mathbf{N}_{(j,k)} = \begin{cases} 
\hat{H}_j \mathcal{J}(j,k) & (k \in \mathcal{J}(j)) \\
\mathcal{A}_k^{-T} (\hat{H}_k \mathcal{J}(k,k) - [f_k^{(l)}] \mathcal{T} G_k) & (k \in \mathcal{C}(j)) \\
\mathbf{0}_{6 \times 18} & \text{(others)}
\end{cases}
\end{equation}

Given all the above, the Jacobian matrix shown in Eq. (66) could be finally shown as followings:

\[
\mathbf{N}_{(j,k)} = \begin{cases} 
\hat{H}_j \mathcal{J}(j,k) & (k \in \mathcal{J}(j)) \\
\mathcal{A}_k^{-T} (\hat{H}_k \mathcal{J}(k,k) - [f_k^{(l)}] \mathcal{T} G_k) & (k \in \mathcal{C}(j)) \\
\mathbf{0}_{6 \times 18} & \text{(others)}
\end{cases}
\]

The computation of Eq. (73) requires matrix \( \hat{H}_j \). It means that \( \hat{H}_j \) for all link \( j \) needs to be computed in advance according to recursive formula Eq. (71). After updating \( \hat{H}_j \), matrix \( \mathbf{N}_{(j,k)} \) for any \( j \) and \( k \) can be directly computed by Eq. (78). The direct computation of Eq. (71) with \( 18 \times 18 \) matrices is computationally inefficient. The final form of \( \hat{H}_j \) derived from Eq. (71) is written down in Appendix D.

V. NUMERICAL EXAMPLES

This section shows the comparison of computation time of Jacobian matrices between the two approaches: the proposed method and the traditional method in [25]. Since the formulations in [25] only handle 1-DOF joints, it was tested by using the serial manipulator which has \( N \) rotational joints. The Jacobian matrix of the joint torque of the first rotational joint was computed by changing the number of joints. It was tested by the computer with Intel(R) Xeon(R) CPU E3-1535M v5. The proposed method, of course, could generate the same Jacobian matrices as those from the classical method. Fig 2 shows the results of the computational time of the two methods. The computational complexity of the conventional method was \( O(N^2) \); on the other hand, the proposed method showed \( O(N) \), and the computational time was significantly improved in the large-DOF cases.

This section also validates the Jacobian in the case of spherical joints; we tested a simple example of motion optimization by using a serial manipulator with 5 spherical joints, as shown in Fig 3. This test assumes no gravity. Each segmented link has save physical quantities: \( \mathbf{p}^{(l)} = [0 0 0.2] \) [m], \( m_j = 3.0 \) [kg], \( \mathbf{c}_j = [0 0 0.1] \) [kgm], \( \mathbf{I}_j = \text{diag}([11.2 11.2 2.4]) \times 10^{-3} \) [kgm^2]. Let \( \mathbf{p}^{(l)} \) be the position of the end-effector at time instance \( t \), and \( \tau_{j,k} (1 \leq j \leq 5, 1 \leq k \leq 3) \) be \( k \)-th component of torques of \( j \)-th spherical joint.

In this planning, the total time length of the trajectory is assumed to be 2 [s], and its sampling rate is 1 [ms]. The variables of the spherical joints are represented by angle-axis vector, and their trajectories was modeled by using B-splines as shown in [25]. The number of B-spline basis is 24 and the timespan between two bases is 0.1 [s]. The number of the position of the end-effector starts from \( \mathbf{p}^0 = [0 0 1] \) with zero joint velocities and accelerations, then passes through \( \mathbf{p}^1 = [0 1 0] \) at \( t = 1.0 \) [s], and stops at \( \mathbf{p}^2 = [1 0 0] \) with zero joint velocities and accelerations. During the movement, the joint torque limitations are considered; \( |\tau_{1,k}| < 0.1 \) [N] and \( |\tau_{j,k}| < 20.0 \) [N] \( (2 \leq j \leq 5) \). In order to avoid self-collision, the distance constraints are also assumed; \( \|\mathbf{p}_k - \mathbf{p}_j\| \geq 0.1 (k \neq l) \). All the constraint conditions were converted to the penalty cost functions according to penalty function method. The optimization itself was solved by quasi-Newton method [9] by utilizing the proposed method. The generated trajectory is shown in Fig 4. The end-effector successfully passed through the targeted positions without self-collision. The result of joint torque trajectories of first joint is shown in Fig 4, which all satisfy the joint torque limitations. The other joint torques also satisfied the limitations.

Motion optimization framework is useful to solve several practical applications as mentioned in the introduction. More complicated cases as shown in [5] will be investigated in future works.

VI. CONCLUSION

This paper presents the comprehensive theory of differential kinematics and dynamics in order to derive the analytical partial derivative of both link/joint quantities with respect to joint coordinates and their derivatives. First, the \( 18 \times 18 \) comprehensive motion transformation matrix (CMTM) and \( 18 \) dimensional product operation is introduced for comprehensive kinematics formulation, which allows a simple chain product of the matrices to represent the standard and differential forward kinematics of a kinematics chain. They also have the same features as the rotation matrix and the \( 6 \times 6 \) transformation matrix of screws, and their product operations. By utilizing CMTM, the partial derivative of link coordinates and their derivatives were derived in the same manner as when introducing the basic Jacobians by just replacing the \( 6 \times 6 \) transformation matrices with CMTM in their formulations.

This novel theoretical framework brought the following contributions with respect to related work. The partial derivatives of each generalized force with respect to joint coordinates and their derivatives were also demonstrated. The procedure of derivation also has similarity with that of linear and angular momentum Jacobians. By utilizing the CMTM, each new Jacobians could be computed by \( O(N) \). The formulation can also handle different types of joints like spherical joints or free-floating base.
anticommutativity with a binary operation.

This work was partly supported by JSPS Grant-in-Aid for Scientific Research (A) Number 17H00768.

APPENDIX

A. Binary operation axioms

The bilinearity, alternativity, Jacobi identity axioms and anticommutativity with a binary operation \([\cdot, \cdot]\) are introduced:

\[
[(ax_1 + bx_2), x_3] = a[x_1, x_3] + b[x_2, x_3] \tag{79}
\]

\[
[x_3, (ax_1 + bx_2)] = a[x_3, x_1] + b[x_3, x_2] \tag{80}
\]

\[
[x_1, x_1] = 0 \tag{81}
\]

\[
[x_1, [x_2, x_3]] + [x_1, [x_2, x_3]] + [x_1, [x_2, x_3]] = 0 \tag{82}
\]

\[
[x_1, x_2] = -[x_2, x_1] \tag{83}
\]

(In this paper, \([\cdot, \cdot]\) corresponds with \([\cdot, \cdot, \cdot]\) or \([\cdot, \cdot]\).

B. Structure of \(18 \times 18\) matrix \(\dot{X}^j_k\) in Eq. (57)

Here are \(6 \times 6\) block matrices in \(\dot{X}^j_k\) of Eq. (57), when writing down with the notation of \(\ddot{X}^j_k\):

\[
\dot{X}^j_k \triangleq \begin{bmatrix}
A_k^{j-1} & O_6 & O_6 \\
X_{(j,k)}^{(2)} & A_k^{j-1} & O_6 \\
X_{(j,k)}^{(3)} & X_{(j,k)}^{(4)} & A_k^{j-1}^{-1}
\end{bmatrix} \tag{84}
\]

\[
\dot{X}_{(j,k)}^{(2)} = A_k^{j-1} [v_{p(k)}^j] \tag{85}
\]

\[
\dot{X}_{(j,k)}^{(3)} = A_k^{j-1} [-([v_{p(k)}^j - A_k^{j} v_j^k], \cdot)] \tag{86}
\]

\[
\dot{X}_{(j,k)}^{(4)} = A_k^{j-1} (-([v_{p(k)}^j, \cdot] + [A_k^{j} v_j^k], \cdot)) \tag{87}
\]

\[
\dot{v}_j^k \triangleq v_j - v_j^k \tag{88}
\]

\[
\dot{v}_j \triangleq v_j - v_j^k - [v_j^k, v_j^k] \tag{89}
\]

C. Recursive formulas of kinematic chain

Let \(S_j^i \) an arbitrary transformation matrix, which is non-singular and satisfies \(S_j^i = S_j^i S_k^j\). In regards to \(S_j^i\), physical quantity \(a_j \in \mathbb{R}^m\) has the following recursive formulas:

\[
a_j = S_j^i a_i + c_j \tag{90}
\]

where, \(c_j \) is a bias term. Similarly, let us consider another transformation matrix \(U_j^i\) and physical quantity \(b_i \in \mathbb{R}^n\), there also exists the following recursive formulas.

\[
b_i = P_j^i v_i + d_i + \sum_{j \in C(i)} U_j^i (P_j^i v_i c_j + \dot{d}_j) \tag{91}
\]

From the comparison between the terms of Eq. (92) and those of Eq. (93), the followings formulas are finally obtained.

\[
\dot{P}_i = P_i + \sum_{j \in C(i)} U_j^i (P_j^i v_i c_j + \dot{d}_j) \tag{94}
\]

\[
\dot{d}_i = d_i + \sum_{j \in C(i)} U_j^i (P_j^i v_i c_j + \dot{d}_j) \tag{95}
\]

D. Structure of \(6 \times 18\) matrix \(\dot{H}_j\) in Eq. (71)

Let us write down \(6 \times 6\) block matrices \(\dot{K}_j, \dot{D}_j, \dot{M}_j\) as follows:

\[
\dot{H}_j = \begin{bmatrix}
\dot{K}_j & \dot{D}_j & \dot{M}_j
\end{bmatrix} = \sum_{k \in \bar{C}(j)} A_k^{-1} [O_6, D_k, M_k] S_k^{-1} X_{p(k)}^{p(k)} S_{p(k)}^{-1} \tag{96}
\]

By substituting the each component of \(X\) and \(S\) according to Eq. (17) and Eq. (24), we can have:

\[
\dot{M}_j = M_j + \sum_{k \in C(j)} A_k^{-T} \dot{M}_k A_k^{-1} \tag{97}
\]

\[
\dot{D}_j = D_j + \sum_{k \in C(j)} A_k^{-T} (\dot{D}_k - \dot{M}_k (v_k^i, \cdot)) A_k^{-1} \tag{98}
\]

\[
\dot{K}_j = O_6 \tag{99}
\]
REFERENCES


