Comprehensive Gradient Computation Framework of PCS Model for Soft Robot Simulation

Taiki Ishigaki, *Graduate Student Member, IEEE*, Ko Ayusawa, and Ko Yamamoto, *Member, IEEE*

***Abstract*—Dynamic simulation of soft and flexible bodies is important for the motion planning and control of soft robot systems. This study presents a fast real-time simulation of the piecewise constant strain (PCS) model proposed in soft robotics research to calculate the dynamics of a beam or rod structure. We extend the theory of comprehensive motion transformation matrix (CMTM) to the PCS model, which allows us to systematically calculate the gradients in the equations of motion. Using the dynamic gradient, we perform a dynamic simulation of the PCS model via implicit integration. Compared to explicit integration, implicit integration allows us to use a larger time-step width for the integration without a flexible deformation divergence, which decreases the computational time. We present several examples of dynamic simulations, including a relatively rigid material such as carbon-fiber-reinforced plastics used in a leaf-spring-type sports prosthesis.**

***Index Terms*—Modeling, control, and learning for soft robots, dynamics, simulation and animation.**

I. Introduction

HUMAN body softness plays an important role in adaptive and dynamic motion control. Hand softness allows the adaptive grasping of unknown shapes. The flexibility of muscles and tendons can facilitate dynamic motions such as running and jumping. A soft robot made of flexible materials is a representative example of such functionality. Many studies have investigated the design of soft robots, and the modeling and control of their flexible and continuous deformations.

The finite element method [1] is often used for general-purpose deformation calculations of flexible objects. While for designing the soft robot hand or arm [2], various rod or beam models proposed [3], [4], [5]. Renda et al. [5] proposed a piecewise constant strain (PCS) model that discretizes the continuous Cosserat rod model [4] of a rod or beam structure into a finite number of segments and assumes a constant strain in each segment. The motion computation of the rigid-body link system was extended to calculate the PCS model dynamics. Furthermore, Renda et al. [6] proposed a *hybrid* model that integrates PCS and rigid multilink models. Ishigaki and Yamamoto [7] proposed a hybrid link system with a floating base link that considers the contact forces with the environment and achieved a forward dynamics simulation of a humanoid robot with a flexible structure in its legs. Using the hybrid link system, the motion analysis of an athlete with a leaf-spring-type prosthesis leg is performed [8], [9]. It is also expected to be applied to research on the handling of deformable linear objects such as cables, which are widely used in the medical or industrial fields with rigid link robots [10].

One of the challenges in simulating the dynamics of a soft and flexible object is the trade-off between the stability and computational cost of numerical calculations [11]. Particularly in numerical integration, a smaller time-step width stabilizes the calculation; however, it increases the computation time. Although a larger time step-width can save computational time, the numerical integration is often unstable. This trade-off becomes more critical when simulating the flexible deformation of a relatively rigid, lightweight material which causes the large acceleration by a small change of position. For example, the main material of a leaf-spring prosthesis leg is CFRP, whose density and Young's modulus are 1.81×10^3 kg/m^3 and 61.35 GPa, respectively. Therefore, a large acceleration is generated during dynamic deformation, which makes the numerical integration unstable with a small time-step width. An implicit integration method is used to ensure numerical stability with a large time step. In contrast to the explicit method, implicit integration is based on the backward difference approximation of velocity and acceleration, in which we need to solve a nonlinear algebraic equation with velocity and acceleration as unknown values. This equation is usually solved using numerical approaches such as the Newton-Raphson method, which computes *dynamic* gradient vectors that are partial derivatives of the quantities in the equations of motion by generalized coordinates, velocity, or acceleration.

Computation algorithms for kinematic gradients have traditionally been investigated in robotics as the computation of the Jacobian matrix [12]. In rigid link systems, the importance of dynamic gradients is increasing because of their applications in motion optimization, model predictive control and, reinforcement learning. Systematic algorithms for dynamic gradients were proposed in several studies [13], [14]. Ayusawa and Yoshida [13] proposed a comprehensive motion transformation matrix (CMTM), which is a Lie group of coordinate transformations of displacement, velocity, and acceleration. However, the computation of dynamic gradients becomes complicated during flexible deformation because the inertia matrix of a flexible object is not constant during motion.

In this study, the kinematics of the PCS model are reformulated using a spatial transformation matrix, and the CMTM

Manuscript received 23 December 2023; accepted 15 April 2024. Date of publication 7 May 2024; date of current version 16 May 2024. This letter was recommended for publication by Associate Editor Y. She and Editor L. Pallottino upon evaluation of the reviewers' comments. This work was supported by JSPS KAKENHI under Grant 21H01282. *(Corresponding author: Taiki Ishigaki.)*

Taiki Ishigaki and Ko Yamamoto are with the Department of Mechano-Informatics, The University of Tokyo, Tokyo 113-8656, Japan (e-mail: ishigaki@ynl.t.u-tokyo.ac.jp; yamamoto.ko@ynl.t.u-tokyo.ac.jp).

Ko Ayusawa is with the Human Augmentation Research Center, National Institute of Advanced Industrial Science and Technology, Chiba 277-0882, Japan (e-mail: k.ayusawa@aist.go.jp).

This letter has supplementary downloadable material available at https://doi.org/10.1109/LRA.2024.3397530, provided by the authors.

Digital Object Identifier 10.1109/LRA.2024.3397530

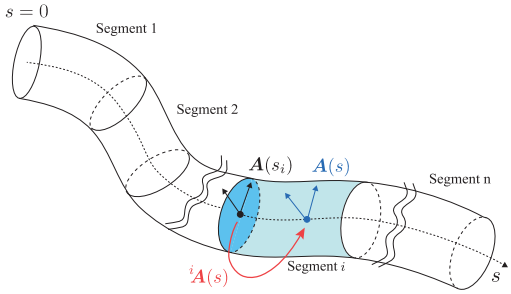

Fig. 1. Schematic view of PCS model. A continuum rod is divided into a finite number of segments. The configuration curve of the rod is expressed by the spatial transformation matrix $\boldsymbol{A}(s)$ at s.

framework is extended to the PCS model by deriving a matrix that maps the strain vector of the PCS model to the tangent vector of the CMTM. Finally, the analytical gradients of the kinematics and dynamics of the PCS model are derived. Using a dynamic gradient, we developed a method to simulate the dynamics of a PCS model through implicit integration. We demonstrated fast dynamic simulations assuming silicone rubber and CFRP as the materials. In particular, by assuming silicone rubber, we show that it is possible to achieve real-time or faster simulations.

In Section II, the flexible deformation of the PCS model is redefined using a spatial transformation matrix, and the kinematics and dynamics of the PCS model are explained. In Section III, the implicit integral method for forward dynamics calculations using the Newmark-β method is presented. In Section IV, we extend the CMTM to the PCS model, and proposed a gradient calculation method for kinematics and dynamics. In Section V, the forward dynamic simulations of the PCS model are presented. The letter ends with a conclusive section and an accompanying appendix.

II. Piecewise Constant Strain (PCS) Model

A. Kinematics of PCS Model

The *central axis coordinate* $s \in \mathbb{R}$ along the axis passing through the center of the rod or beam structure is defined, as shown in Fig. 1. Continuous deformation along s can be represented by a spatial transformation matrix $\boldsymbol{A}(s,t)$ defined as

$$\boldsymbol{A}(s,t) := \begin{bmatrix} \boldsymbol{R}(s,t) & \boldsymbol{O} \\ [\boldsymbol{p}(s,t)\times]\,\boldsymbol{R}(s,t) & \boldsymbol{R}(s,t) \end{bmatrix} \tag{1}$$

where $\boldsymbol{O}$ denotes the zero matrix, and $\boldsymbol{R}(s,t) \in \mathrm{SO}(3)$ and $\boldsymbol{p}(s,t) \in \mathbb{R}^3$ denote the orientation and position at s and time t, respectively. Hereafter, the simple notation $\boldsymbol{A}(s)$ is used and t is omitted when the meaning is clear. $\boldsymbol{A}(s)$ is defined as the as *configuration curve*.

Spatial transformation matrix $\boldsymbol{A}$ is an adjoint representation of the homogeneous transformation matrix. Set of $\boldsymbol{A}$ as a Lie group that satisfies the following properties:

1) Let $\boldsymbol{A}_i$ and ${}^i\boldsymbol{A}_{i+1}$ denote the transformation from the world frame to a local frame $\{i\}$, and the transformation from $\{i\}$ to another frame $\{i+1\}$. Thus, the following chain rule is satisfied:

$$\boldsymbol{A}_{i+1} = \boldsymbol{A}_i\,{}^i\boldsymbol{A}_{i+1} \tag{2}$$

2) Tangent vector $\delta\boldsymbol{\alpha} \in \mathbb{R}^6$ can be defined as

$$[(\delta\boldsymbol{\alpha})_\bullet] := \boldsymbol{A}^{-1}\delta\boldsymbol{A}. \tag{3}$$

where for any $\boldsymbol{x}_1, \boldsymbol{x}_2 \in \mathbb{R}^3$ and $\boldsymbol{x} = [\boldsymbol{x}_1^T \boldsymbol{x}_2^T]^T$, $[\boldsymbol{x}_\bullet]$ is defined as follows:

$$[\boldsymbol{x}_\bullet] := \begin{bmatrix} [\boldsymbol{x}_1\times] & \boldsymbol{O} \\ [\boldsymbol{x}_2\times] & [\boldsymbol{x}_1\times] \end{bmatrix}. \tag{4}$$

In the Cosserat Rod theory [15], [16], *strain* is defined as the deformation of infinitesimal solids. Given a configuration curve $\boldsymbol{A}(s)$, the strain vector $\boldsymbol{\xi}(s) \in \mathbb{R}^6$ is defined as follows:

$$[\boldsymbol{\xi}(s)_\bullet] := \boldsymbol{A}^{-1}(s)\frac{\partial \boldsymbol{A}(s)}{\partial s} \tag{5}$$

Note that the strain vector is defined w.r.t. the local frame.

In the PCS model, the configuration curve is divided into finite segments, as illustrated in Fig. 1. We label index i as a segment at $s \in [s_i, s_{i+1}]$ and call it Segment i. Subsequently, the strain in each segment was assumed to be constant for s:

$$\boldsymbol{\xi}_i := \boldsymbol{\xi}(s)\ (s \in [s_i, s_{i+1}]). \tag{6}$$

Multiplying both sides of (5) by $\boldsymbol{A}(s)$ yields the following first-order linear differential equation for $\boldsymbol{A}(s)$ with respect to s.

$$\frac{\partial \boldsymbol{A}(s)}{\partial s} = \boldsymbol{A}(s)\,[\boldsymbol{\xi}_{i\bullet}]\ \ (s \in [s_i, s_{i+1}]). \tag{7}$$

Given an initial value of $\boldsymbol{A}$ at $s = s_i$, we obtain the solution to (7) as follows:

$$\boldsymbol{A}(s) = \boldsymbol{A}_i\,{}^i\boldsymbol{A}(s)\ (s \in [s_i, s_{i+1}]) \tag{8}$$

$$\boldsymbol{A}_i := \boldsymbol{A}(s_i), \quad {}^i\boldsymbol{A}(s) := \exp\{(s - s_i)\,[\boldsymbol{\xi}_{i\bullet}]\} \tag{9}$$

where ${}^i\boldsymbol{A}(s)$ represents the transformation from the starting point s_i of Segement i to the point s. The matrix exponential of ${}^i\boldsymbol{A}(s)$ has a closed form and can be calculated without numerical method [5].

Substituting $s = s_{i+1}$ into (8) yields the chain rule (2), which represents the transformation from parent segment i into child segment $i+1$. By recursively applying (2) and (8), we can compute the configuration curve for a given value of $\boldsymbol{\xi}_i$ $(i = 1, \ldots, n)$.

B. Differential Kinematics of PCS Model

Replacing s in (5) with t yields the definition of spatial velocity $\boldsymbol{\eta}$:

$$[\boldsymbol{\eta}_\bullet] := \boldsymbol{A}^{-1}\dot{\boldsymbol{A}} \tag{10}$$

where $\dot{*}$ is the time derivative of physical quantity $*$. In a manner similar to strain $\boldsymbol{\xi}$, the spatial velocity is also defined w.r.t. the local frame, often used in robotics algorithms [17].

Given the time differentiation of the strain $\dot{\boldsymbol{\xi}}_i$ and velocity $\boldsymbol{\eta}(s_i)$, the velocity $\boldsymbol{\eta}(s)$ at $s \in [s_i, s_{i+1}]$ can be computed as

$$\boldsymbol{\eta}(s) = {}^i\boldsymbol{A}(s)^{-1}\boldsymbol{\eta}_i + {}^i\boldsymbol{C}(s)\dot{\boldsymbol{\xi}}_i,\ \boldsymbol{\eta}_i := \boldsymbol{\eta}(s_i) \tag{11}$$

where ${}^i\boldsymbol{C}(s)$ is defined as

$$ {}^i\boldsymbol{C}(s) := \int_{s_i}^{s} {}^i\boldsymbol{A}(x)^{-1}dx. \tag{12}$$

Note that ${}^i\boldsymbol{C}(s)$ can be analytically calculated.

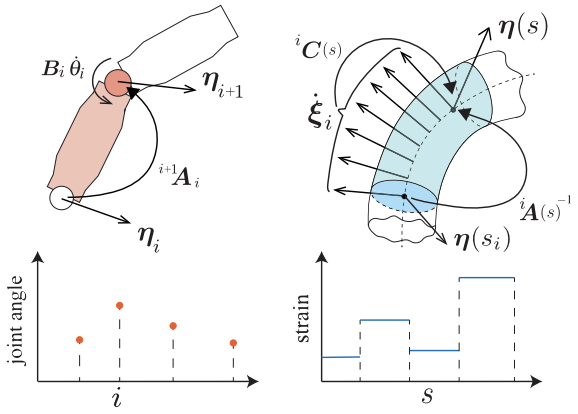

Fig. 2. Schematic view of the relationship between the strain velocity $\dot{\boldsymbol{\xi}}$ and the velocity $\boldsymbol{\eta}$.

Fig. 2 shows the physical meaning of (11) and the comparison with the case of a rigid-body link system. In the PCS model, $\boldsymbol{\eta}_i$ is transformed into the velocity at s by ${}^i\boldsymbol{A}^{-1}(s)$. Subsequently, $\dot{\boldsymbol{\xi}}_i$ is transformed into the velocity at s by ${}^i\boldsymbol{C}(s)$. In the case of a rigid-body link system consisting of rotational joints, as shown on the left side of Fig. 2, the transformation from parent link i into child link $i+1$ is represented as

$$\boldsymbol{\eta}_{i+1} = {}^i\boldsymbol{A}_{i+1}^{-1}\boldsymbol{\eta}_i + \boldsymbol{B}_i\dot{\theta}_i \tag{13}$$

where θ_i is the joint angle and $\boldsymbol{B}_i$ is a constant matrix defined according to the joint type. Compared with the PCS model, a rigid-body link system has a *discrete* deformation as the joint angle, and its mapping to the spatial velocity is easily obtained as a simple matrix $\boldsymbol{B}_i$ in (13). In the PCS model, the continuous deformation represented by strain is mapped by ${}^i\boldsymbol{C}(s)$, which is the spatial integration of the spatial transformation matrix.

By differentiating both sides of (11) and arranging the terms, we obtain the acceleration vector $\dot{\boldsymbol{\eta}}(s)$ at $s \in [s_i, s_{i+1}]$ as follows:

$$\begin{aligned}\dot{\boldsymbol{\eta}}(s) = {}&{}^i\boldsymbol{A}^{-1}(s)\dot{\boldsymbol{\eta}}_i + {}^i\boldsymbol{C}(s)\ddot{\boldsymbol{\xi}}_i \\ &+ \left(\left[\left({}^i\boldsymbol{A}^{-1}(s)\boldsymbol{\eta}_i\right)_\bullet\right]{}^i\boldsymbol{C}(s) + {}^i\dot{\boldsymbol{C}}(s)\right)\dot{\boldsymbol{\xi}}_i\end{aligned} \tag{14}$$

To derive the above equation, we use the following relationship:

$${}^i\dot{\boldsymbol{A}}^{-1}(s) = -\left[\left({}^i\boldsymbol{C}(s)\dot{\boldsymbol{\xi}}\right)_\bullet\right]{}^i\boldsymbol{A}(s)^{-1}. \tag{15}$$

In a previous study [5] ${}^i\dot{\boldsymbol{C}}(s)\dot{\boldsymbol{\xi}}_i$ was assumed to be zero and ignored in (14). However, it does not always hold that ${}^i\dot{\boldsymbol{C}}(s)\dot{\boldsymbol{\xi}}_i$ usually has a non-zero value. ${}^i\dot{\boldsymbol{C}}(s)\dot{\boldsymbol{\xi}}_i = 0$ only when the vectors $\dot{\boldsymbol{\xi}}_i$ and $\boldsymbol{\xi}_i$ are parallel. Therefore, we consider ${}^i\dot{\boldsymbol{C}}(s)\dot{\boldsymbol{\xi}}_i$ in (14).

C. Dynamics of PCS Model

The equation of motion of the PCS model is obtained by integrating that of an infinitesimal solid in Cosserat rod theory, which can be expressed as

$$\mathcal{M}(s)\dot{\boldsymbol{\eta}}(s) - [\boldsymbol{\eta}(s)_\bullet]^{\mathrm{T}}\mathcal{M}(s)\boldsymbol{\eta}(s) = \mathcal{F}(s) \tag{16}$$

where $\mathcal{M}(s) \in \mathbb{R}^{6\times6}$ and $\mathcal{F}(s) \in \mathbb{R}^6$ denote the screw inertia matrix and force vector [5], respectively.

In the PCS model, the equation of motion for Segment i at $s = s_i$ consists of the following two equations:

1) Equation on the internal force between segments.

$$\int_{s_i}^{s_{i+1}} {}^i\boldsymbol{A}(s)^{-\mathrm{T}}\mathcal{F}(s)ds = \boldsymbol{f}_i - {}^i\boldsymbol{A}_{i+1}^{-\mathrm{T}}\boldsymbol{f}_{i+1} \tag{17}$$

where $\boldsymbol{f}_i$ is the force applied to Segment i from the parent $i-1$. Segment i also receives $-\boldsymbol{f}_{i+1}$ from child $i+1$.

2) Equations on the strain and stress.

$$\int_{s_i}^{s_{i+1}} {}^i\boldsymbol{C}(s)^{\mathrm{T}}\mathcal{F}(s)ds = \boldsymbol{\tau}_i - {}^i\boldsymbol{C}_{i+1}^{T}\boldsymbol{f}_{i+1} \tag{18}$$

where $\boldsymbol{\tau}_i$ denotes the general force of the PCS model for Segment i.

Equations (17) and (18) are obtained from the spatial integration of (16) along s by multiplying it by ${}^i\boldsymbol{A}(s)^{-\mathrm{T}}$ and ${}^i\boldsymbol{C}(s)^{\mathrm{T}}$ as the dual mapping of the velocity, respectively.

Furthermore, (17) and (18) are combined into

$$\begin{bmatrix}\boldsymbol{M}_\eta & \boldsymbol{M}_{\eta\xi}\\ \boldsymbol{M}_{\eta\xi}^T & \boldsymbol{M}_\xi\end{bmatrix}\begin{bmatrix}\dot{\boldsymbol{\eta}}_i\\ \ddot{\boldsymbol{\xi}}_i\end{bmatrix} + \begin{bmatrix}\boldsymbol{b}_\eta\\ \boldsymbol{b}_\xi\end{bmatrix} = \begin{bmatrix}\boldsymbol{f}_i\\ \boldsymbol{\tau}_i\end{bmatrix} - \begin{bmatrix}{}^i\boldsymbol{A}_{i+1}^{-\mathrm{T}}\\ {}^i\boldsymbol{C}_{i+1}^{T}\end{bmatrix}\boldsymbol{f}_{i+1}. \tag{19}$$

See Appendix A for the definition of $\boldsymbol{M}_*$ and $\boldsymbol{b}_*$.

Equation (19) can be rewritten for all segments as Langrange's equation. The general coordinates $\boldsymbol{q}$ and general forces $\boldsymbol{\tau}$ are defined by combining all the strain vectors $\boldsymbol{\xi}_i$ and the corresponding force vectors $\boldsymbol{\tau}_i$ as follows:

$$\boldsymbol{q} := \begin{bmatrix}\boldsymbol{\xi}_1^T & \cdots & \boldsymbol{\xi}_n^T\end{bmatrix}^T, \boldsymbol{\tau} := \begin{bmatrix}\boldsymbol{\tau}_1^T & \cdots & \boldsymbol{\tau}_n^T\end{bmatrix}^T. \tag{20}$$

We can utilize the recursive formula given by (19) to compute the force $\boldsymbol{f}_i$ acting on each segment from the child-segment side and subsequently derive Lagrange's equation of motion for all segments

$$\boldsymbol{M}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \boldsymbol{b}(\boldsymbol{q}, \dot{\boldsymbol{q}}) = \boldsymbol{\tau} \tag{21}$$

where $\boldsymbol{M}(\boldsymbol{q})$ and $\boldsymbol{b}(\boldsymbol{q}, \dot{\boldsymbol{q}})$ denote the inertia matrix and bias terms, respectively. $\boldsymbol{\tau}$ includes the actuated force $\boldsymbol{\tau}_{\mathrm{act}}$ from such as cable actuators and the internal force $\boldsymbol{\tau}_{\mathrm{int}}$ from the flexible material's viscoelasticity as follows [5]:

$$\boldsymbol{\tau} = \boldsymbol{\tau}_{\mathrm{act}} + \boldsymbol{\tau}_{\mathrm{int}} \tag{22}$$

$\boldsymbol{\tau}_{\mathrm{int}}$ is expressed as

$$\boldsymbol{\tau}_{\mathrm{int}} = \boldsymbol{K}(\boldsymbol{q}_0 - \boldsymbol{q}) - \boldsymbol{D}\dot{\boldsymbol{q}} \tag{23}$$

where $\boldsymbol{K}$ and $\boldsymbol{D}$ are the stiffness and viscosity matrices, respectively. In this study, we focus on flexible objects with passive viscoelasticity, namely, we assume $\boldsymbol{\tau}_{\mathrm{act}} = 0$.

III. Forward Dynamics Calculation Using the Implicit Method

In previous studies [5], [7], explicit integral methods have been used for numerical time integration in a dynamic PCS model simulation. Explicit methods using forward differences tend to be unstable in numerical calculations. However, implicit methods using backward differences are less divergent but require solving algebraic equations consisting of backward difference equations and equations of motion. In this section, we explain the Newmark-β method, which is an implicit method, and demonstrate that the gradients of the general force are required.

The Newmark-β method uses the rate of change in acceleration $\Delta\ddot{\boldsymbol{q}}$ and the following backward difference equation:

$$\begin{cases} \ddot{\boldsymbol{q}}_{k+1} = \ddot{\boldsymbol{q}}_k + \Delta\ddot{\boldsymbol{q}}_{k+1} \\ \dot{\boldsymbol{q}}_{k+1} = \dot{\boldsymbol{q}}_k + \Delta t\ddot{\boldsymbol{q}}_k + \gamma\Delta t\Delta\ddot{\boldsymbol{q}}_{k+1} \\ \boldsymbol{q}_{k+1} = \boldsymbol{q}_k + \Delta t\dot{\boldsymbol{q}}_k + \dfrac{1}{2}\Delta t^2\ddot{\boldsymbol{q}}_k + \Delta t^2\beta\Delta\ddot{\boldsymbol{q}}_{k+1} \end{cases} \tag{24}$$

where Δt is the time step width of the simulation, and β and γ are the parameters. In this study, we use the average acceleration method ($\beta = 1/4$ and $\gamma = 1/2$), which leads to unconditional stability.

The difference between the left and right sides of (22) is defined as follows:

$$\begin{aligned} \boldsymbol{h}(\boldsymbol{q}, \dot{\boldsymbol{q}}, \ddot{\boldsymbol{q}}) &:= \boldsymbol{\tau} - (\boldsymbol{\tau}_{\mathrm{act}} + \boldsymbol{\tau}_{\mathrm{int}}) \\ &= \boldsymbol{M}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \boldsymbol{b}(\boldsymbol{q}, \dot{\boldsymbol{q}}) - (\boldsymbol{K}(\boldsymbol{q}_0 - \boldsymbol{q}) - \boldsymbol{D}\dot{\boldsymbol{q}}) \end{aligned} \tag{25}$$

The Newmark-β method determines $\Delta\ddot{\boldsymbol{q}}_{k+1}$ that satisfies

$$\boldsymbol{h}(\Delta\ddot{\boldsymbol{q}}_{k+1}) = \boldsymbol{0}, \tag{26}$$

which we obtain by substituting (24) into (25) as follows: The nonlinear (26) can be solved using the Newton-Raphson method with the following iterations:

$$\Delta\ddot{\boldsymbol{q}}_k^{\ell+1} = \Delta\ddot{\boldsymbol{q}}_k^{\ell} - \boldsymbol{J}^{-1}\boldsymbol{h}(\Delta\ddot{\boldsymbol{q}}_k^{\ell}) \tag{27}$$

where ℓ denotes the iteration steps for the Newton-Raphson method and k is the time step.

The Jacobian matrix $\boldsymbol{J}$ can be calculated as:

$$\boldsymbol{J} = \left.\frac{\partial \boldsymbol{h}}{\partial \boldsymbol{x}}\right|_{\boldsymbol{x}=\Delta\ddot{\boldsymbol{q}}^{l}} = \frac{\partial \boldsymbol{h}}{\partial \ddot{\boldsymbol{q}}} + \Delta t\gamma\frac{\partial \boldsymbol{h}}{\partial \dot{\boldsymbol{q}}} + \Delta t^2\beta\frac{\partial \boldsymbol{h}}{\partial \boldsymbol{q}} \tag{28}$$

In the above equation, $\frac{\partial \boldsymbol{h}}{\partial \ddot{\boldsymbol{q}}}$, $\frac{\partial \boldsymbol{h}}{\partial \dot{\boldsymbol{q}}}$, and $\frac{\partial \boldsymbol{h}}{\partial \boldsymbol{q}}$ are calculated as

$$\frac{\partial \boldsymbol{h}}{\partial \ddot{\boldsymbol{q}}} = \frac{\partial \boldsymbol{\tau}}{\partial \ddot{\boldsymbol{q}}}, \quad \frac{\partial \boldsymbol{h}}{\partial \dot{\boldsymbol{q}}} = \frac{\partial \boldsymbol{\tau}}{\partial \dot{\boldsymbol{q}}} + \boldsymbol{D}, \quad \frac{\partial \boldsymbol{h}}{\partial \boldsymbol{q}} = \frac{\partial \boldsymbol{\tau}}{\partial \boldsymbol{q}} + \boldsymbol{K}. \tag{29}$$

$\boldsymbol{D}$, and $\boldsymbol{K}$ are easily obtained. While $\frac{\partial \boldsymbol{\tau}}{\partial \ddot{\boldsymbol{q}}}$ is equal to $\boldsymbol{M}$ according to (21), $\frac{\partial \boldsymbol{\tau}}{\partial \boldsymbol{q}}$ and $\frac{\partial \boldsymbol{\tau}}{\partial \dot{\boldsymbol{q}}}$ have not been investigated for the PCS model. In the next section, we propose a comprehensive calculation framework for *dynamic* gradients.

IV. Comprehensive Motion Transformation Matrix of PCS Model

In this section, the CMTM [13] is reviewed, and its framework is extended to the PCS model. Based on this extension, we propose gradient computation techniques for the kinematics and dynamics of PCS models.

A. Comprehensive Motion Transformation Matrix [13]

The CMTM is defined as

$$\boldsymbol{X} := \begin{bmatrix} \boldsymbol{A} & \boldsymbol{O} & \boldsymbol{O} \\ \boldsymbol{A}[\boldsymbol{\eta}_\bullet] & \boldsymbol{A} & \boldsymbol{O} \\ \frac{1}{2}\boldsymbol{A}([\dot{\boldsymbol{\eta}}_\bullet] + [\boldsymbol{\eta}_\bullet]^2) & \boldsymbol{A}[\boldsymbol{\eta}_\bullet] & \boldsymbol{A} \end{bmatrix}. \tag{30}$$

This 18-dimensional square matrix includes the position/attitude $\boldsymbol{A}$, velocity $\boldsymbol{\eta}$, and acceleration $\dot{\boldsymbol{\eta}}$. The set of CMTM is also a Lie group and has properties similar to the spatial transformation matrix $\boldsymbol{A}$ given by (2) and (3).

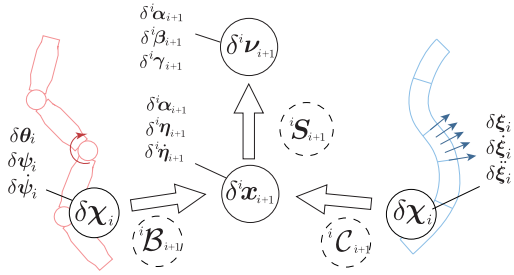

Fig. 3. Schematics of calculation path from joint variable or strain variable $\delta\boldsymbol{\chi}_i$ to CMTM tangent vector $\delta^i\boldsymbol{\nu}_{i+1}$ on the rigid-body link system and the PCS model.

1) Let $\boldsymbol{X}_i$ and ${}^i\boldsymbol{X}_{i+1}$ denote the transformation from the world frame to a local frame $\{i\}$ and from $\{i\}$ to another frame $\{i+1\}$. Thus, the following chain rule is satisfied:

$$\boldsymbol{X}_{i+1} = \boldsymbol{X}_i{}^i\boldsymbol{X}_{i+1}. \tag{31}$$

2) The tangent vector $\delta\boldsymbol{\nu}$ is defined as

$$[(\delta\boldsymbol{\nu})^\bullet] := \boldsymbol{X}^{-1}\delta\boldsymbol{X} \tag{32}$$

where $\delta\boldsymbol{\nu} \in \mathbb{R}^{18}$ and $[(\delta\boldsymbol{\nu})^\bullet]$ are defined as follows:

$$\delta\boldsymbol{\nu} := [\delta\boldsymbol{\alpha}^T\delta\boldsymbol{\beta}^T\delta\boldsymbol{\gamma}^T]^T, \tag{33}$$

$$\begin{cases} \delta\boldsymbol{\beta} := \delta\boldsymbol{\eta} + [(\delta\boldsymbol{\alpha})_\bullet]\boldsymbol{\eta} \\ \delta\boldsymbol{\gamma} := \frac{1}{2}(\delta\dot{\boldsymbol{\eta}} + [(\delta\boldsymbol{\alpha})_\bullet]\dot{\boldsymbol{\eta}} + [(\delta\boldsymbol{\beta})_\bullet]\boldsymbol{\eta}) \end{cases}, \tag{34}$$

$$[(\delta\boldsymbol{\nu})^\bullet] := \begin{bmatrix} [(\delta\boldsymbol{\alpha})_\bullet] & \boldsymbol{O} & \boldsymbol{O} \\ [(\delta\boldsymbol{\beta})_\bullet] & [(\delta\boldsymbol{\alpha})_\bullet] & \boldsymbol{O} \\ [(\delta\boldsymbol{\gamma})_\bullet] & [(\delta\boldsymbol{\beta})_\bullet] & [(\delta\boldsymbol{\alpha})_\bullet] \end{bmatrix}. \tag{35}$$

The sum of the tangent vectors $\delta\boldsymbol{\nu}$ can be expressed as

$$\delta\boldsymbol{\nu}_{i+1} = {}^{i+1}\boldsymbol{X}_i\delta\boldsymbol{\nu}_i + \delta^i\boldsymbol{\nu}_{i+1}. \tag{36}$$

The CMTM combines multiple and complicated transformations of position/orientation, velocity, and acceleration into a single (36), providing a simple expression.

This is illustrated in Fig. 3. The tangent vector $\delta\boldsymbol{\nu}$ can be mapped to another space $\delta\boldsymbol{x}$, which is defined as

$$\delta\boldsymbol{x} := \begin{bmatrix} \delta\boldsymbol{\alpha}^T & \delta\boldsymbol{\eta}^T & \delta\dot{\boldsymbol{\eta}}^T \end{bmatrix}^T. \tag{37}$$

Compared with $\delta\boldsymbol{\nu}$, $\delta\boldsymbol{x}$ has an intuitive physical meaning, replacing $\delta\boldsymbol{\beta}$ and $\delta\boldsymbol{\gamma}$ with the spatial velocity and acceleration, respectively. The transformation of $\delta\boldsymbol{x}$ into $\delta\boldsymbol{\nu}$ is expressed as:

$$\delta\boldsymbol{\nu} = \boldsymbol{S}\delta\boldsymbol{x} \tag{38}$$

$$\boldsymbol{S} := \begin{bmatrix} \boldsymbol{E} & \boldsymbol{O} & \boldsymbol{O} \\ -[\boldsymbol{\eta}_\bullet] & \boldsymbol{E} & \boldsymbol{O} \\ -\frac{1}{2}\left([\dot{\boldsymbol{\eta}}_\bullet] - [\boldsymbol{\eta}_\bullet]^2\right) & -\frac{1}{2}[\boldsymbol{\eta}_\bullet] & \frac{1}{2}\boldsymbol{E} \end{bmatrix} \tag{39}$$

where $\boldsymbol{E}$ is the identity matrix.

In the case of a rigid-body link system, $\delta\boldsymbol{x}$ stems from a joint variation, represented by the following vector:

$$\delta\boldsymbol{\chi}_i := \begin{bmatrix} \delta\boldsymbol{\theta}_i^T & \delta\boldsymbol{\psi}_i^T & \delta\dot{\boldsymbol{\psi}}_i^T \end{bmatrix}^T. \tag{40}$$

where $\delta\boldsymbol{\theta}_i$, $\delta\boldsymbol{\psi}_i$, and $\delta\dot{\boldsymbol{\psi}}_i$ are the displacement, velocity, and acceleration of Joint i, respectively. The transformation of $\delta\boldsymbol{\chi}$

into $\delta\boldsymbol{x}$ is expressed as follows:

$$\delta^i\boldsymbol{x}_{i+1} = {}^i\mathcal{B}_{i+1}\delta\boldsymbol{\chi}_i, \quad {}^i\mathcal{B}_{i+1} := \begin{bmatrix} \boldsymbol{B}_i & \boldsymbol{O} & \boldsymbol{O} \\ \boldsymbol{O} & \boldsymbol{B}_i & \boldsymbol{O} \\ \boldsymbol{O} & \boldsymbol{O} & \boldsymbol{B}_i \end{bmatrix} \tag{41}$$

where $\delta^i\boldsymbol{x}_{i+1}$ is the displacement of Link $i+1$ with respect to the parent link i, and ${}^i\mathcal{B}_{i+1}$ is a constant matrix determined by the type of joint. Based on the CMTM theory, Ayusawa and Yoshida [13] proposed a unified calculation method for gradient vectors in both kinematics and dynamics of a rigid-body link system.

B. Strain Variation Space PCS Model and CMTM

To extend the CMTM framework to the PCS model, we propose a new transformation matrix $\mathcal{C}$ instead of $\mathcal{B}$ as illustrated in Fig. 3. For the PCS model, we redefine $\delta\boldsymbol{\chi}$ as the following vector combining the strain $\delta\boldsymbol{\xi}$, its velocity $\delta\dot{\boldsymbol{\xi}}$, and its acceleration $\delta\ddot{\boldsymbol{\xi}}$:

$$\delta\boldsymbol{\chi}_i := \begin{bmatrix} \delta\boldsymbol{\xi}_i^T & \delta\dot{\boldsymbol{\xi}}_i^T & \delta\ddot{\boldsymbol{\xi}}_i^T \end{bmatrix}^T. \tag{42}$$

One of the main contributions of this study is the derivation of the following transformation formula for the PCS model:

$$\delta^i\boldsymbol{x}_{i+1} = {}^i\mathcal{C}_{i+1}\delta\boldsymbol{\chi}_i, \tag{43}$$

$${}^i\mathcal{C}_{i+1} := \begin{bmatrix} {}^i\boldsymbol{C}_{i+1} & \boldsymbol{O} & \boldsymbol{O} \\ \frac{\partial^i\boldsymbol{C}_{i+1}}{\partial\boldsymbol{\xi}_i}\dot{\boldsymbol{\xi}}_i & {}^i\boldsymbol{C}_{i+1} & \boldsymbol{O} \\ \frac{\partial^i\dot{\boldsymbol{C}}_{i+1}}{\partial\boldsymbol{\xi}_i}\dot{\boldsymbol{\xi}}_i + \frac{\partial^i\boldsymbol{C}_{i+1}}{\partial\boldsymbol{\xi}_i}\ddot{\boldsymbol{\xi}}_i & \frac{\partial^i\dot{\boldsymbol{C}}_{i+1}}{\partial\boldsymbol{\xi}_i}\dot{\boldsymbol{\xi}}_i + {}^i\dot{\boldsymbol{C}}_{i+1} & {}^i\boldsymbol{C}_{i+1} \end{bmatrix} \tag{44}$$

where $\mathcal{C}$ depends on $\boldsymbol{\xi}$, $\dot{\boldsymbol{\xi}}$, and $\ddot{\boldsymbol{\xi}}$. Whereas $\mathcal{B}$ in a rigid-link system is a constant block diagonal matrix, $\mathcal{C}$ is a lower-triangular matrix. The remainder of this subsection describes the derivation of matrix $\mathcal{C}$.

From (8), (11), and (14), the relationship between parent and child segments can be summarized as follows:

$${}^i\boldsymbol{A}_{i+1} = \exp\left((s_{i+1} - s_i)\left[\boldsymbol{\xi}_{i\bullet}\right]\right), \tag{45}$$

$${}^i\boldsymbol{\eta}_{i+1} = {}^i\boldsymbol{C}_{i+1}\dot{\boldsymbol{\xi}}_i, \tag{46}$$

$${}^i\dot{\boldsymbol{\eta}}_{i+1} = {}^i\dot{\boldsymbol{C}}_{i+1}\dot{\boldsymbol{\xi}}_i + {}^i\boldsymbol{C}_{i+1}\ddot{\boldsymbol{\xi}}_i. \tag{47}$$

Then, $\delta^i\boldsymbol{\alpha}_{i+1}$, $\delta^i\boldsymbol{\eta}_{i+1}$, and $\delta^i\dot{\boldsymbol{\eta}}_{i+1}$, which are the elements of $\delta^i\boldsymbol{x}_{i+1}$, can be computed as

$$\delta^i\boldsymbol{\alpha}_{i+1} = {}^i\boldsymbol{C}_{i+1}\delta\boldsymbol{\xi}_i, \tag{48}$$

$$\delta^i\boldsymbol{\eta}_{i+1} = \left(\frac{\partial^i\boldsymbol{C}_{i+1}}{\partial\boldsymbol{\xi}_i}\dot{\boldsymbol{\xi}}_i\right)\delta\boldsymbol{\xi}_i + {}^i\boldsymbol{C}_{i+1}\delta\dot{\boldsymbol{\xi}}_i, \tag{49}$$

$$\begin{aligned}\delta^i\dot{\boldsymbol{\eta}}_{i+1} = &\left(\frac{\partial^i\dot{\boldsymbol{C}}_{i+1}}{\partial\boldsymbol{\xi}_i}\dot{\boldsymbol{\xi}}_i + \frac{\partial^i\boldsymbol{C}_{i+1}}{\partial\boldsymbol{\xi}_i}\ddot{\boldsymbol{\xi}}_i\right)\delta\boldsymbol{\xi}_i \\ &+ \left(\frac{\partial^i\dot{\boldsymbol{C}}_{i+1}}{\partial\dot{\boldsymbol{\xi}}_i}\dot{\boldsymbol{\xi}}_i + {}^i\dot{\boldsymbol{C}}_{i+1}\right)\delta\dot{\boldsymbol{\xi}}_i + {}^i\boldsymbol{C}_{i+1}\delta\ddot{\boldsymbol{\xi}}_i.\end{aligned} \tag{50}$$

Given that $\delta\boldsymbol{A} = \sum_k \frac{\partial\boldsymbol{A}}{\partial\xi_k}\delta\xi_k$ where $\boldsymbol{\xi} = [\xi_1 \cdots \xi_6]^T$, $\delta\boldsymbol{A}$ can be calculated using the following equation:

$$\delta\boldsymbol{A} = \boldsymbol{A}\left[(\boldsymbol{C}(s)\delta\boldsymbol{\xi})_\bullet\right]. \tag{51}$$

Equation (48) can be derived by substituting (51) into the definition of $\boldsymbol{\alpha}$ in (3). Furthermore, to derive (49) and (50), using the variants of the relationships of (46) and (47), the variants of ${}^i\boldsymbol{C}_{i+1}$ for an arbitrary vector $\boldsymbol{x} \in \mathbb{R}^6$ are computed as follows:

$$\delta\boldsymbol{C}\boldsymbol{x} = \sum_k \left(\frac{\partial\boldsymbol{C}}{\partial\xi_k}\boldsymbol{x}\right)\delta\xi_k. \tag{52}$$

We can compute the variants of ${}^i\dot{\boldsymbol{C}}_{i+1}$ similar to (52).

C. Kinematics Gradient Computation of PCS Model

Using the CMTM framework, we computed the gradient of the kinematic quantities of the PCS model. Using the (36) and (38), $\delta\boldsymbol{x}_j$ can be calculated as follows:

From (36), the tangent vector of the CMTM $\delta\boldsymbol{\nu}_j$ at the starting point of Segment j can be calculated as the linear sum of the relative vector $\delta^i\boldsymbol{\nu}_{i+1}$ of each segment as follows:

$$\delta\boldsymbol{\nu}_j = \sum_{i<j} {}^i\boldsymbol{X}_j^{-1}\delta^i\boldsymbol{\nu}_{i+1}. \tag{53}$$

By converting $\delta\boldsymbol{\nu}$ into $\delta\boldsymbol{x}$ using (38), we can express (53) as

$$\delta\boldsymbol{x}_j = \boldsymbol{S}_j^{-1}\sum_{i<j} {}^i\boldsymbol{X}_j^{-1\,i}\boldsymbol{S}_{i+1}\delta^i\boldsymbol{x}_{i+1}. \tag{54}$$

Assuming $\boldsymbol{\chi} = [\boldsymbol{\chi}_1^T \cdots \boldsymbol{\chi}_n^T]^T$, from (54) and (43), the Jacobian matrix $\boldsymbol{J}_j$, mapping $\delta\boldsymbol{\chi}$ to $\delta\boldsymbol{x}_j$, can be computed as

$$\delta\boldsymbol{x}_j = \boldsymbol{J}_j\delta\boldsymbol{\chi} = \sum_{i<j}\boldsymbol{J}_{(j,i)}\delta\boldsymbol{\chi}_i, \tag{55}$$

$$\boldsymbol{J}_{(j,i)} := \begin{cases} \boldsymbol{S}_j^{-1j}\boldsymbol{X}_{i+1}{}^i\boldsymbol{S}_{i+1}{}^i\mathcal{C}_{i+1} & (i<j) \\ \boldsymbol{O} & (i \geq j) \end{cases}. \tag{56}$$

The block matrix $\boldsymbol{J}_{(j,i)}$ is related to the Segment i within $\boldsymbol{J}_j$.

The Jacobian matrix can be defined with respect to vector $\delta\boldsymbol{x}$ at the starting point of each segment. The Jacobian matrix $\boldsymbol{J}(s)$ at point s $(s_j < s < s_{j+1})$, which maps the variation $\delta\boldsymbol{\chi}$ onto $\delta\boldsymbol{x}(s)$, can be calculated as follows:

$$\delta\boldsymbol{x}(s) = \boldsymbol{J}(s)\delta\boldsymbol{\chi} = \sum_{i<=j}\boldsymbol{J}_i(s)\delta\boldsymbol{\chi}_i, \tag{57}$$

$$\boldsymbol{J}_i(s) := \begin{cases} \boldsymbol{S}(s)^{-1\,i}\boldsymbol{X}(s)^{-1\,i}\boldsymbol{S}_{i+1}{}^i\mathcal{C}_{i+1} & (i<j) \\ {}^i\mathcal{C}(s) & (i=j) \\ \boldsymbol{O} & (i>j) \end{cases}. \tag{58}$$

${}^i\mathcal{C}(s)$ denotes the matrix that holds $\delta^i\boldsymbol{x}(s) = {}^i\mathcal{C}(s)\delta\boldsymbol{\chi}_i$. The position, attitude, velocity, and acceleration variants at any point s can be calculated using the Jacobian matrix.

D. Dynamic Gradient Computation of PCS Model

From the Newton-Euler (17) and (18), we propose a method for calculating $\boldsymbol{Q}_j$ and $\boldsymbol{F}_j$ that satisfies the following equations with respect to force $\boldsymbol{f}$ and the general force $\boldsymbol{\tau}$:

$$\delta\boldsymbol{f}_j = \boldsymbol{Q}_j\delta\boldsymbol{\chi}, \quad \delta\boldsymbol{\tau}_j = \boldsymbol{F}_j\delta\boldsymbol{\chi}. \tag{59}$$

By considering the variations in (17) and (18), $\delta\boldsymbol{f}_i$ and $\delta\boldsymbol{\tau}_i$ can then be calculated. For transformation matrix $\boldsymbol{Y}(=\boldsymbol{A}^{-1}, \boldsymbol{C})$, $\delta\boldsymbol{f}_i$ and $\delta\boldsymbol{\tau}_i$ can be expressed as follows:

$$\delta\boldsymbol{f}_i = \delta\boldsymbol{h}_i(\boldsymbol{A}^{-1}), \quad \delta\boldsymbol{\tau}_i = \delta\boldsymbol{h}_i(\boldsymbol{C}) \tag{60}$$

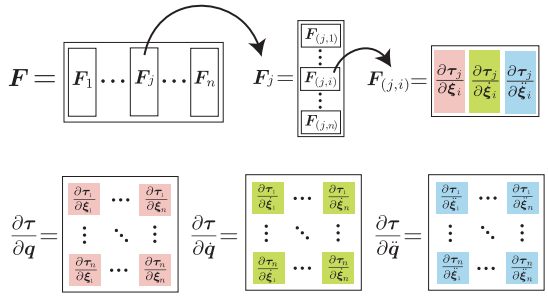

Fig. 4. Schematics of matrix elements of $\boldsymbol{F}$ which is the Jacobian matrix of general force $\boldsymbol{\tau}$ on the PCS model, relationship $\boldsymbol{F}$, and partial differentiation of general force $\boldsymbol{\tau}$.

$$\begin{aligned}\delta \boldsymbol{h}_i(\boldsymbol{Y}) &:= \int_{s_i}^{s_{i+1}} \delta {}^i\boldsymbol{Y}(s)^{\mathrm{T}} \mathcal{F}(s) + {}^i\boldsymbol{Y}(s)^{\mathrm{T}} \delta \mathcal{F}(s) ds \\ &+ \delta {}^i\boldsymbol{Y}_{i+1}^{\mathrm{T}} \boldsymbol{f}_{i+1} + {}^i\boldsymbol{Y}_{i+1}^{\mathrm{T}} \delta \boldsymbol{f}_{i+1} \\ &= \boldsymbol{N}_i(\boldsymbol{Y}) \delta \boldsymbol{\chi} + {}^i\boldsymbol{Y}_{i+1}^{\mathrm{T}} \delta \boldsymbol{f}_{i+1}. \end{aligned} \tag{61}$$

$\delta \mathcal{F}(s)$ is calculated as follows:

$$\begin{aligned}\delta \mathcal{F}(s) &= \boldsymbol{H}(s) \delta \boldsymbol{x}(s), \\ \boldsymbol{H}(s) &:= \begin{bmatrix} \boldsymbol{O} & \boldsymbol{L}(s) & \mathcal{M}(s) \end{bmatrix} \in \mathbb{R}^{6\times 18}, \\ \boldsymbol{L}(s) &:= -\left[\left(\mathcal{M}(s) \boldsymbol{\eta}(s) \right)_{\hat{\bullet}} \right] - \left[\boldsymbol{\eta}(s)_{\bullet} \right]^T \mathcal{M}(s) \in \mathbb{R}^{6\times 6} \end{aligned} \tag{62}$$

where $[\hat{\bullet}]$ satisfies $[\boldsymbol{\eta}_1{}_{\bullet}]^T \boldsymbol{\eta}_2 = [\boldsymbol{\eta}_2{}_{\hat{\bullet}}] \boldsymbol{\eta}_1$. From (57) and (62), we define $\boldsymbol{N}_i$ as follows using $\boldsymbol{H}$ and $\boldsymbol{J}$:

$$\begin{aligned}\boldsymbol{N}_i(\boldsymbol{Y}) &:= \int_{s_i}^{s_{i+1}} {}^i\boldsymbol{Y}(s)^{\mathrm{T}} \boldsymbol{H}(s) \boldsymbol{J}(s) ds \\ &+ \boldsymbol{T} \left(\int_{s_i}^{s_{i+1}} \frac{\partial {}^i\boldsymbol{Y}(s)^{\mathrm{T}}}{\partial \boldsymbol{\xi}_i} \mathcal{F}(s) ds + \frac{\partial {}^i\boldsymbol{Y}_{i+1}^{\mathrm{T}}}{\partial \boldsymbol{\xi}_i} \boldsymbol{f}_{i+1} \right) \end{aligned} \tag{63}$$

where $\boldsymbol{T} = [\boldsymbol{E}\ \boldsymbol{O}\ \boldsymbol{O}] \in \mathbb{R}^{6\times 18}$.

Consequently, from (61) we obtain $\boldsymbol{Q}_j$ and $\boldsymbol{F}_j$ have the following relationship:

$$\boldsymbol{Q}_j = \boldsymbol{N}_j(\boldsymbol{A}^{-1}) + {}^j\boldsymbol{A}_{j+1}^{-\mathrm{T}} \boldsymbol{Q}_{j+1}, \tag{64}$$

$$\boldsymbol{F}_j = \boldsymbol{N}_j(\boldsymbol{C}) + {}^j\boldsymbol{C}_{j+1}^{\mathrm{T}} \boldsymbol{Q}_{j+1}. \tag{65}$$

From the above, from (64) and (65), $\boldsymbol{Q}_j$ and $\boldsymbol{F}_j$ can be calculated using $\boldsymbol{Q}_{n+1} = \boldsymbol{O}$ when we assume that the number of segments is n.

The mapping matrix from $\delta\boldsymbol{\chi}$ to $\delta\boldsymbol{\tau}$ is defined as $\boldsymbol{F} = [\boldsymbol{F}_1^{\mathrm{T}} \cdots \boldsymbol{F}_n^{\mathrm{T}}]^{\mathrm{T}}$. $\frac{\partial \boldsymbol{\tau}}{\partial \boldsymbol{q}}$, $\frac{\partial \boldsymbol{\tau}}{\partial \dot{\boldsymbol{q}}}$, and $\frac{\partial \boldsymbol{\tau}}{\partial \ddot{\boldsymbol{q}}}$ can be obtained from the element of $\boldsymbol{F}$ as shown in Fig. 4, and implicit integration can be performed by using them for (28).

V. Verification Using Forward Dynamics Simulations

A. Setting up Numerical Experiments

We performed forward dynamic simulations using implicit integration with the proposed dynamic gradient calculation method for the PCS model and verified its effectiveness. Furthermore, simulations using an explicit integral (Euler integral) were performed to compare the computation times.

TABLE I
Material Parameters Set in Simulations

Parameter	Silicone Rubber	CFRP
Young's modulus E [kPa]	110	61.635×10^6
Shear viscosity μ [Pa·s]	300	20.5×10^6
Poisson's ratio ν [-]	0.5	0.3
Density ρ [kg/m^3]	1.08×10^3	1.81×10^3

The following simulations used a computer with an AMD Ryzen 9 5950 × 16 core 32 threads CPU and 128 GB of memory. Silicone rubber and CFRP were assumed as the materials, and the parameter sets are listed in Table I.

B. Verification Assuming Silicone Rubber

The flexible deformation of a cantilever beam owing to gravity was simulated, as shown in Fig. 5. A silicone rubber material is assumed, with a constant cross-sectional area, a total length of 0.9 m, and three segments.

Simulations were performed for the following four conditions to compare and verify the explicit and implicit integrals:

- E1 : explicit integration, time-step width is as small as possible $\Delta t = 4.0 \times 10^{-6}$ s,
- E2 : explicit integration, time-step width is as large as possible $\Delta t = 4.0 \times 10^{-4}$ s to the extent that the computation does not diverge,
- I1 : implicit integration, time-step width is as large as possible $\Delta t = 4.0 \times 10^{-3}$ s, and the threshold for the Newton-Raphson method is $\epsilon = 0.001$,
- I2 : implicit integration, the same time-step width as condition I1, and $\epsilon = 0.1$.

Compared to conditions E2, I1 and I2 can increase the time-step width by 10, indicating that the numerical stability of the implicit integral is higher than that of the explicit integration.

The snapshots of the simulation results for the implicit integral are presented in Fig. 5. It is possible to simulate deformations and bending in the direction of gravity that is applied vertically downwards. The time variation of the strain in the first segment on the root side with the largest change during the simulation is shown in Fig. 6. The upper part of Fig. 6 shows a graph of the linear strain, and the lower part shows a graph of the angular strain. The bending deformation is particularly large around the x-axis in the short-side direction, where bending is more likely to occur. The results for all methods were simulated without significant differences. In the explicit integration, when $\Delta t = 4.0 \times 10^{-3}$ s, which is same as in I1 and I2, the simulation diverges as it begins.

A comparison of the computation time required for a 10 s simulation is shown in Fig. 7. The explicit integral involves multi-core programming with parallel recursive inverse dynamic calculations in the unit vector method performed. The computation time for condition E1 was about 790 s, whereas for condition E2 the computation time was about 7.9 s. Furthermore, the computation time was about 5.6 s for condition I1 and about 2.6 s for condition I2.

We implemented an adaptive method that modulates the time-step width depending on whether the iterative computation converges. In the explicit integration, the equations of motion are linear with respect to the generalized acceleration, which does not require any iteration. Therefore, we implemented an adaptive method only in the implicit integration and performed

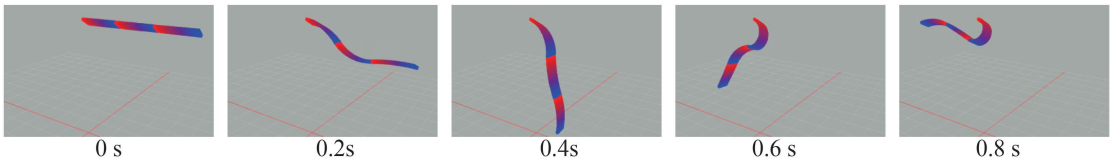

Fig. 5. Snapshots of forward dynamics simulation of the silicone rod with the PCS model which has three segments.

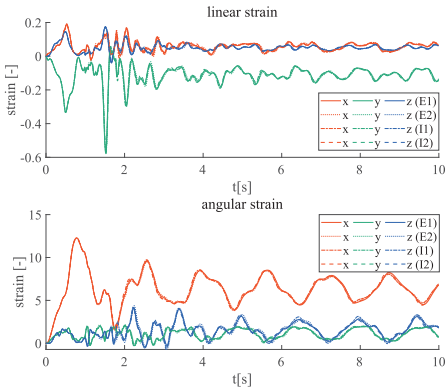

Fig. 6. The time plot of linear (top) and angular (bottom) strain of the root 1st segment, comparing explicit method (E1, E2) and implicit method (I1, I2).

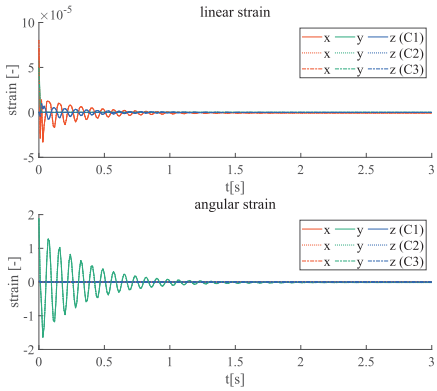

Fig. 8. The time plot of linear (top) and angler (bottom) strain of the root 1st segment, comparing the different time-step width (condition C1, C2, C3).

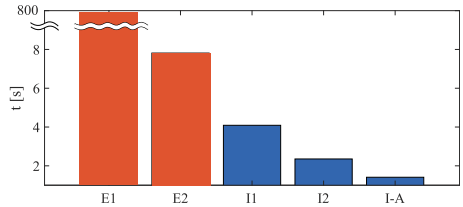

Fig. 7. Comparing the computation time between explicit method (E1, E2), implicit method (I1, I2) and adaptive time step width (I-A).

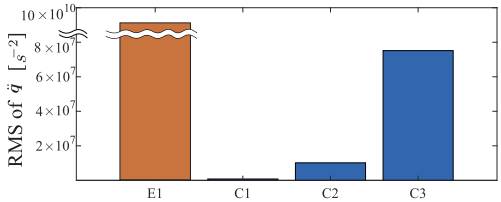

Fig. 9. Comparing the RMS of the general acceleration in the 1st step of the CFRP prosthesis leg's simulation between explicit method (same time-step width as E1) and implicit method (C1, C2, C3).

simulations by adaptively varying the time-step width in the range $\Delta t \in [0.004\ \mathrm{s}, 0.064\ \mathrm{s}]$ (Condition: I-A). As shown in Fig. 7, the computation time was about 1.4 s.

Note that a numerical differentiation is sometimes used for calculating a gradient in a simulation of an FEM or rigid-body link system although its accuracy is usually worse than that of analytical differentiation. We also tested a simulation of the PCS model by the implicit integration using numerical differentiation; however, the simulation did not converge, which implies that analytical gradient calculations are important.

C. Verification Assuming CFRP

The PCS model of the leaf-spring-shaped CFRP prosthesis (655 g) used in [8], [9] was validated for the simulation by implicit integration. As shown in Fig. 10, the number of segments is set to six, assuming a model with the prosthesis tip fixed to the ground. The initial strain was applied $+0.01$ for the linear strain component and $+1.0$ for the angular strain component, and the simulations were performed until convergence to the unloaded shape was achieved. The model assumes a material with light mass and high stiffness. In the explicit integration, the general acceleration in the first step of the simulation is large as shown in Fig. 9 and the simulation diverges as it begins with $\Delta t = 4.0 \times 10^{-6}$ s, similar to E1.

Simulations were performed for the following three conditions to compare and verify the differences in time-step width:

- C1 : implicit integration, time-step width is as large as possible, $\Delta t = 1.0 \times 10^{-3}$ s, and the threshold for the Newton-Raphson method is $\epsilon = 0.001$.
- C2 : implicit integration, time-step width is $\Delta t = 1.0 \times 10^{-4}$ s, and the threshold is same as C1.
- C3 : implicit integration, time-step width is $\Delta t = 1.0 \times 10^{-5}$ s, and the threshold is same as C1.

The snapshot of the forward dynamics simulation results under condition C1 is shown in Fig. 10. Under condition C1, the computation time for the 10 s simulation was 60 s. Under conditions C1-3, a graph of the change in the strain of the first segment fixed to the ground during the simulation is shown in Fig. 8. The upper part of the Fig. 8 is a graph of the linear strain, and the lower part is a graph of the angular strain. In particular, it deformed and oscillated significantly around the y-axis in the direction of the short side of the bendable cross-section. The

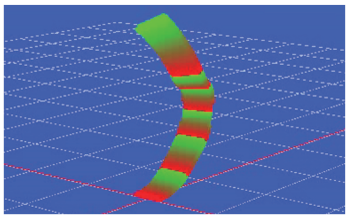

Fig. 10. Snapshot of the forward dynamics simulation of the prosthesis leg made from CFRP, which is a six-segment PCS model.

results remained nearly the same regardless of the difference in the time-step width for conditions C1-3.

VI. Conclusion

This study extends the CMTM framework to the PCS model and proposes a method for calculating its dynamic gradients. Furthermore, a gradient calculation method was used to simulate the PCS model using implicit integration.

1) We newly derived the transformation matrix with strain, which is the continuous deformation of the PCS model, and applied the CMTM framework to the PCS model by replacing the transformation matrix to comprehensively handle the strain velocities and accelerations.
2) A method is proposed to compute the kinematics and dynamics gradient of the PCS model. An implicit integration method with the Newmark-β method was used to compute the forward dynamics of the PCS model.
3) Numerical simulations were performed to compare the implicit integral with the explicit integral. For a flexible object such as silicone rubber, the implicit integral can compute 10 s physical phenomenon in about 2.6 s, which is a fast dynamics simulation. In the case of highly stiff materials such as CFRP, explicit integration resulted in divergent calculations, whereas implicit integration allowed simulations to be performed.

The fast dynamic simulation attained in this study will be a fundamental technology for soft robot development. The dynamic gradient calculation method is expected to be applied not only to implicit integration but also to optimization problems such as: model predictive control and motion planning. Considering the contact force from the environment, in future work, we will develop a method to calculate the forward dynamics and contact force simultaneously.

Appendix A
Newton-Euler Equations of Motion on PCS Model

For the transformation matrices $\boldsymbol{Y}, \boldsymbol{Z}(=\boldsymbol{A}^{-1}, \boldsymbol{C})$, the bias term $\boldsymbol{b}_*$ and the inertia matrix $\boldsymbol{M}_*$ of (19) are defined as

$$\boldsymbol{b}_{\boldsymbol{\eta}} = \boldsymbol{b}(\boldsymbol{A}^{-1}), \quad \boldsymbol{b}_{\boldsymbol{\xi}} = \boldsymbol{b}(\boldsymbol{C}) \tag{66}$$

$$\boldsymbol{M}_{\boldsymbol{\eta}} := \boldsymbol{M}(\boldsymbol{A}^{-1}, \boldsymbol{A}^{-1}), \boldsymbol{M}_{\boldsymbol{\eta\xi}} := \boldsymbol{M}(\boldsymbol{A}^{-1}, \boldsymbol{C}),$$

$$\boldsymbol{M}_{\boldsymbol{\xi}} := \boldsymbol{M}(\boldsymbol{C}, \boldsymbol{C}) \tag{67}$$

$$\boldsymbol{b}(\boldsymbol{Y}) := \left(\int_{s_i}^{s_{i+1}} {}^i\boldsymbol{Y}(s)^{\mathrm{T}} \mathcal{M}(s) \left({}^i\dot{\boldsymbol{A}}(s)^{-1} + {}^i\dot{\boldsymbol{C}}(s) \right) ds \right) \dot{\boldsymbol{\xi}}_i - \left(\int_{s_i}^{s_{i+1}} {}^i\boldsymbol{Y}(s)^{\mathrm{T}} \left[\boldsymbol{\eta}(s)_\bullet \right]^{\mathrm{T}} \mathcal{M}(s) \boldsymbol{\eta}(s) \right) ds \tag{68}$$

$$\boldsymbol{M}(\boldsymbol{Y}, \boldsymbol{Z}) := \int_{s_i}^{s_{i+1}} {}^i\boldsymbol{Y}(s)^{\mathrm{T}} \mathcal{M}(s) {}^i\boldsymbol{Z}(s) ds \tag{69}$$

References

[1] F. Faure et al., "Sofa: A multi-model framework for interactive physical simulation," in *Soft Tissue Biomechanical Modeling for Computer Assisted Surgery*. Berlin, Germany: Springer, 2012, pp. 283–321.

[2] J. Pinskier and D. Howard, "From bioinspiration to computer generation: Developments in autonomous soft robot design," *Adv. Intell. Syst.*, vol. 4, no. 1, 2022, Art. no. 2100086.

[3] I. S. Godage, G. A. Medrano-Cerda, D. T. Branson, E. Guglielmino, and D. G. Caldwell, "Dynamics for variable length multisection continuum arms," *Int. J. Robot. Res.*, vol. 35, no. 6, pp. 695–722, 2016.

[4] F. Renda, M. Giorelli, M. Calisti, M. Cianchetti, and C. Laschi, "Dynamic model of a multibending soft robot arm driven by cables," *IEEE Trans. Robot.*, vol. 30, no. 5, pp. 1109–1122, Oct. 2014.

[5] F. Renda, F. Boyer, J. Dias, and L. Seneviratne, "Discrete cosserat approach for multisection soft manipulator dynamics," *IEEE Trans. Robot.*, vol. 34, no. 6, pp. 1518–1533, Dec. 2018.

[6] F. Renda and L. Seneviratne, "A geometric and unified approach for modeling soft-rigid multi-body systems with lumped and distributed degrees of freedom," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2018, pp. 1567–1574.

[7] T. Ishigaki and K. Yamamoto, "Dynamics computation of a hybrid multi-link humanoid robot integrating rigid and soft bodies," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2021, pp. 2816–2821.

[8] S. Kim, T. Ishigaki, Y. Shimane, Y. Ikegami, and K. Yamamoto, "Inverse kinematics of hybrid multi-link system and its application to motion capture for athlete wearing sports prosthesis," in *Proc. IEEE-RAS 21st Int. Conf. Humanoid Robots*, 2022, pp. 837–842.

[9] Y. Shimane, T. Ishigaki, S. Kim, Y. Ikegami, and K. Yamamoto, "Application of piece-wise constant strain model to flexible deformation calculation of sports prosthesis and stiffness estimation," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2022, pp. 837–842.

[10] K. Almaghout, A. Cherubini, and A. Klimchik, "Robotic co-manipulation of deformable linear objects for large deformation tasks," *Robot. Auton. Syst.*, vol. 175, 2024, Art. no. 104652.

[11] Y. Jiang, "Soft robotics," in *Advanced Robotics for Manufacturing*. Clemson, SC, USA: Clemson Univ., ch. 8. Accessed: Apr. 15, 2024. [Online]. available: https://opentextbooks.clemson.edu/me8930/chapter/soft-robotics/

[12] O. Khatib, "A unified approach for motion and force control of robot manipulators: The operational space formulation," *IEEE J. Robot. Automat.*, vol. 3, no. 1, pp. 43–53, Feb. 1987.

[13] K. Ayusawa and E. Yoshida, "Comprehensive theory of differential kinematics and dynamics towards extensive motion optimization framework," *Int. J. Robot. Res.*, vol. 37, no. 13/14, pp. 1554–1572, 2018.

[14] J. Carpentier et al., "The Pinocchio C++ library: A fast and flexible implementation of rigid body dynamics algorithms and their analytical derivatives," in *Proc. IEEE/SICE Int. Symp. System Integration*, 2019, pp. 614–619.

[15] S. S. Antman, *Nonlinear Problems of Elasticity*, vol. 107. Berlin, Germany: Springer, 2005.

[16] F. Boyer and F. Renda, "Poincaré's equations for cosserat media: Application to shells," *J. Nonlinear Sci.*, vol. 27, no. 1, pp. 1–44, 2017.

[17] R. Featherstone, *Rigid Body Dynamics Algorithms*. Berlin, Germany: Springer, 2014.