An Edge-Based Computationally-Efficient Formulation of Saint Venant-Kirchhoff Tetrahedral Finite Elements

RYO KIKUUWE, HIROAKI TABUCHI, and MOTOJI YAMAMOTO
Kyushu University

This paper describes a computationally-efficient formulation and an algorithm for tetrahedral finite-element simulation of elastic objects subject to Saint Venant-Kirchhoff (StVK) material law. The number of floating point operations required by the algorithm is in the range of 15% to 27% for computing the vertex forces from a given set of vertex positions, and 27% to 38% for the tangent stiffness matrix, in comparison to a well-optimized algorithm directly derived from the conventional Total Lagrangian formulation. In the new algorithm, the data are associated with edges and tetrahedron-sharing edge-pairs (TSEPs), as opposed to tetrahedra, to avoid redundant computation. Another characteristic of the presented formulation is that it reduces to that of a spring-network model by simply ignoring all the TSEPs. The technique is demonstrated through an interactive application involving haptic interaction, being combined with a linearized implicit integration technique employing a preconditioned conjugate gradient method.

Categories and Subject Descriptors: I.3.5 [**Computer Graphics**]: Computational Geometry and Object Modeling—*Physically based modeling*

General Terms: Algorithms

Additional Key Words and Phrases: simulation, deformation, finite element, interactive, haptics, Green-Lagrange strain, Saint Venant-Kirchhoff material

1. INTRODUCTION

Fast simulation of deformable soft objects is an important issue for interactive applications such as computer games, virtual reality-based surgery training, and interactive 3D model editing, which can involve haptic interaction. Among many models for simulating deformable continuums, spring network (SN) models and finite element (FE) models are two of the most popular classes of deformation models. The SN models are generally faster and simpler in computation while FE models have better consistency with the continuum mechanics. There are many types of FE models ranging from simple tetrahedral linear models to many variations of higher-order nonlinear models. Appropriate modeling schemes should be chosen considering the balance between computational efficiency and numerical accuracy required by the application.

The linear FE models constitute the simplest class of FE models, but they are disadvantageous in that they distort the shape of the simulated object under rigid rotations and, worse, do not allow rotations of greater than $\pi/2$. Saint Venant-Kirchhoff (StVK) material law is the simplest continuum constitutive law that allows arbitrary rigid rotations without affecting the overall shape of the model. Arbitrary rigid rotation can be treated even with SN models, which are generally much faster, and some other material laws are more accurate, although computationally slower. Therefore, the StVK material law can be useful for some applications where a compromise between the speed and the accuracy is necessary, which include some classes of interactive applications such as surgery simulators. Although it is attracting attention from researchers, the nonlinear nature of StVK-based FE models tends to make themselves look complicated, having been hindering mathematical sophistication.

This paper presents an alternative formulation of the tetrahedral FE model subject to the StVK material law. The formulation analytically describes the connections from vertex positions to the vertex forces and to the global tangent stiffness matrix. The formulation is practically useful because its computation requires smaller number of floating point operations compared to previous formulations in the literature. Besides, by simply skipping a few computational loops, the presented algorithm becomes equivalent to the algorithm for simulating a network of nonlinear springs. This feature might be convenient in some situations where the user tentatively needs to accelerate the computation by sacrificing the accuracy.

The derivation of the new formulation starts by writing the strain energy of the whole mesh as a concise quadratic function of the

Authors' address: Department of Mechanical Engineering, Kyushu University, 744 Motooka, Nishi-ku, Fukuoka 819-0395, Japan. Hiroaki Tabuchi is now with Sumitomo Electric Hardmetal Corp., Japan.

© ACM, 2009. This is the author's version of the work. It is posted here by permission of ACM for your personal use. Not for redistribution. The definitive version was published in ACM Transactions on Graphics, Volume 28, Issue 1 (January 2009) http://doi.acm.org/10.1145/1477926.1477934

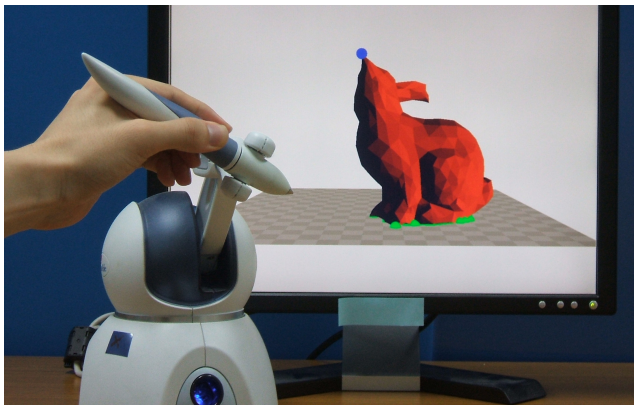


Fig. 1. Interactive simulation of a deformable object involving haptic feedback. The object is the mesh $\mathcal{M}_B$ (the Stanford Bunny composed of 4064 tetrahedra) introduced in section 5. The timestep size for the simulation was 30 ms while the sampling interval for the force feedback was 1 ms. Detailed information on this demonstration is found in section 5.

squared edge lengths. Such an approach is not novel with two-dimensional membranes [Delingette 2008] but cannot be found with three-dimensional solids in the literature. The energy is then analytically differentiated twice to yield vertex forces and a global tangent stiffness matrix. The obtained expressions employ a new class of geometric primitives named *tetrahedron-sharing edge pairs* (*TSEPs*). The material and geometric properties of the mesh are associated to the edges and the TSEPs, as opposed to the tetrahedra, to remove the redundancy in computation. The presented method is conceptually different from previous "edge-based" FE methods [Martins et al. 1997; Coutinho et al. 2001], which are for efficient storage and use of given stiffness matrices, not for efficient assembly of stiffness matrices.

The rest of this paper is organized as follows. Section 2 describes an overview of previous studies. Section 3, which is the main part of this paper, derives a new formulation and computational procedure for the StVK-based tetrahedral FE model. Section 4 shows that the fictitious equilibrium, which is a major drawback of the StVK material law, can be avoided by the conventional idea of using volume-preserving forces without introducing much increase in the number of floating-point operations. Section 5 demonstrates the presented method through an interactive application involving haptic interaction, as shown in Fig. 1. Section 6 provides the concluding remarks.

2. PREVIOUS STUDIES

2.1 FE Models and SN Models

Since the early work by Terzopoulos et al. [1987], there have been many models and techniques for computing object deformations: FE methods, SN models [Van Gelder 1998; Duysak and Zhang 2004; Lloyd et al. 2007], boundary element methods [James and Pai 1999], finite volume methods [Teran et al. 2003], and many types of mesh-free methods, as detailed in excellent reviews [Gibson and Mirtich 1997; Nealen et al. 2006]. The present paper mainly focuses on FE models except some remarks on their relations to SN models.

In both of SN models and FE models, the continuum of interest is discretized into a mesh composed of many edges and polyhedral elements (in the simplest case, tetrahedra), and the force and position are evaluated at the vertices of the mesh. In SN models, the edges are substituted by tensile springs that produce forces only in the direction of their lengths. In FE models, the polyhedral elements are considered as filled solids that apply forces on their vertices according to the theories of the continuum mechanics. An attractive feature of SN models is that they are conceptually simple and generally fast in computation. Attractive features of FE techniques are that they have better consistency with the continuum mechanics and that they can capture spatial inhomogeneity and anisotropy of continuums.

Several studies have been carried out to find better SN models to approximate FE models, but such models have been successful only in limited cases [Van Gelder 1998; Lloyd et al. 2007]. Delingette [2008] has presented a formal relation between an StVK material law and an SN model but his formulation is limited to two-dimensional triangular surface meshes like thin membranes or shells.

2.2 Linear and Nonlinear FE Models

In FE models, the vertex forces can be computed by taking the sums of the contributions of individual elements. In the linear FE techniques, however, element-by-element computation of the vertex forces is not necessary because the vertex forces are linearly related to the vertex positions through constant stiffness matrices of elements, which can be assembled into a global stiffness matrix in the precomputation phase. The global stiffness matrix is usually very large but sparse. In the precomputation phase, it can be condensed into a dense, smaller matrix to accelerate the runtime computation [Bro-Nielsen and Cotin 1996] and can be inverted (under appropriate constraint conditions) to allow implicit methods [Hirota and Kaneko 2001; Nakao et al. 2006]. Besides, the memory usage can be reduced by storing only non-zero off-diagonal blocks of the sparse stiffness matrix, which can be associated with individual edges [Martins et al. 1997; Coutinho et al. 2001]. Unfortunately, such elegant mathematical techniques do not apply straightforwardly to nonlinear FE techniques.

Green Lagrange (GL) strain tensor is a nonlinear strain measure that is invariant with respect to rigid rotation. An StVK material is a material in which GL strain tensor and a 2nd Piola-Kirchhoff (PK2) stress tensor are linearly related (i.e., materially linear although

geometrically nonlinear). Because the use of GL strain measure generally increases the computational cost of FE techniques, some approximation techniques have been presented. Barbič and James [2005] presented fast computation of StVK material based on precomputed reduced coordinates, exploiting the fact that the vertex forces are described as a cubic polynomial of the vertex displacements. Zhong et al. [2005] used the interpolation of precomputed relations between surface displacements and internal displacements to simulate materials described with GL strain measure and nonlinear material laws.

There are also many studies that avoid using strong approximations in geometrically-nonlinear interactive FE simulation. In most of previous studies, the computation is performed in element-by-element manners; i.e., the contributions of individual elements are computed and summed into the vertex forces. In O'Brien and Hodgins's work [1999], the vertex forces are computed through the PK2 stress tensor, which is obtained from the vertex positions. This method has been used in many scenarios such as cutting [Mendoza and Laugier 2003] and adapting mesh refinement [Debunne et al. 2001]. Miller et al. [2007] presented a similar methods with hexahedral meshes, also based on an element-by-element approach. One exception is Picinbono et al.'s work [2003], in which the vertex forces are computed through an expanded cubic polynomial of the vertex positions. They avoid using an element-by-element approach by associating the computational steps and data with different geometric primitives (vertices, edges, faces, and tetrahedra). Their method is reported to be five times slower than the linear FE method.

2.3 Tangent Stiffness Matrix

The simulation of the shape of an object, which changes over the time, requires numerical integration over the time of the vertex accelerations resulting from the vertex forces. There are many types of numerical integration schemes, including several types of explicit and implicit methods. Explicit integration schemes only require the vertex force vectors to be computed, but generally suffer from numerical instability unless the timestep size is set small enough. On the other hand, implicit integration schemes, such as the one used by Baraff and Witkin [1998] for cloth simulation, are rather stable. They however require the global tangent stiffness matrix to be assembled at every timestep except with linear FE models. Thus, many of previous studies on SN models and nonlinear FE models, such as those cited above and others [Zhuang and Canny 2000; Brouwer et al. 2007], employ explicit integration schemes.

In previous studies on interactive applications employing implicit integration, the runtime assembly of the global tangent stiffness matrix by element-by-element computation has not been preferred due to its computational cost. Capell et al.'s [2002a] paper provides an analytical expression of the tangent stiffness matrix in a fully-expanded polynomial of vertex displacements. They however are not proactive to use it in interactive applications due to its computational cost. Debunne et al. [2001] described a "semi-implicit" integration scheme, but they ignored all off-diagonal blocks. The technique described in [Müller et al. 2001] also seems to ignore all off-diagonal blocks of the global tangent stiffness matrix. Some researchers [Müller et al. 2002; Capell et al. 2002b; 2002a; Müller and Gross 2004] employed approximated tangent stiffness matrices obtained by rotating nonzero blocks of the precomputed initial stiffness matrices. Capell et al.'s [2002a] method is specific to models that have rigid skeletons. In contrast, the stiffness warping method [Müller and Gross 2004] is a versatile method in which all elements are rotated independently. This method is reported to be stable under large strain but is still an approximation of continuum mechanics.

2.4 Prevention of Element Inversion

A common drawback of the StVK material model and SN models is that it is invariant with respect to reflection and thus produces fictitious equilibrium at the inverted state. The use of volume-preserving forces has been introduced to SN models [Van Gelder 1998; Lloyd et al. 2007] and to StVK-based FE models [Picinbono et al. 2003]. Some researchers avoid using StVK material law and GL strain (e.g.,[Teran et al. 2003; Irving et al. 2006]) in applications where interactivity is not a primary concern. This problem will be discussed later in section 4.

3. DERIVATION OF A NEW FORMULATION

3.1 Problem Formulation

In the rest of this paper, $\mathcal{M}$ denotes the mesh of interest, and $\mathcal{T}$, $\mathcal{E}$, and $\mathcal{V}$ denote a tetrahedron, an edge, and a vertex, respectively. An edge is defined as an ordered pair of vertices. Different primitives are distinguished by subscripts, e.g., $\mathcal{E}_a$ and $\mathcal{E}_b$. A quantity associated to a primitive or a set of primitives is denoted as, e.g., $\boldsymbol{p}(\mathcal{V})$ and $L(\mathcal{E}_a, \mathcal{E}_b)$. The symbols $N_\mathcal{V}$, $N_\mathcal{E}$, $N_\mathcal{F}$, and $N_\mathcal{T}$ denote the numbers of vertices, edges, faces, and tetrahedra, respectively. Besides, $\boldsymbol{I}_n$ denotes the n-dimensional identity matrix, and $\boldsymbol{o}_n$ and $\boldsymbol{O}_{k\times n}$ denote the n-dimensional zero vector and the $k \times n$ zero matrix, respectively.

Let $\boldsymbol{p}(\mathcal{V})$ and $\boldsymbol{f}(\mathcal{V})$ ($\in \mathbb{R}^3$) denote the current position and the force, respectively, at a vertex $\mathcal{V}$. Let $\boldsymbol{p}$ and $\boldsymbol{f}$ ($\in \mathbb{R}^{3N_\mathcal{V}}$) be the global position and force vectors composed of the vectors $\boldsymbol{p}(\mathcal{V})$ and $\boldsymbol{f}(\mathcal{V})$, respectively, from all vertices $\mathcal{V} \in \mathcal{M}$. Let $\boldsymbol{\mathcal{F}} : \mathbb{R}^{3N_\mathcal{V}} \rightarrow \mathbb{R}^{3N_\mathcal{V}}$ be a function that relates $\boldsymbol{p}$ and $\boldsymbol{f}$ as follows:

$$\boldsymbol{f} = \boldsymbol{\mathcal{F}}(\boldsymbol{p}). \tag{1}$$

In an StVK material, $\boldsymbol{\mathcal{F}}(\boldsymbol{p})$ is a cubic polynomial of $\boldsymbol{p}$ [Capell et al. 2002a; Picinbono et al. 2003; Barbič and James 2005]. Besides, let $\boldsymbol{\mathcal{K}} : \mathbb{R}^{3N_\mathcal{V}} \rightarrow \mathbb{R}^{3N_\mathcal{V}\times 3N_\mathcal{V}}$ be defined as follows:

$$\boldsymbol{\mathcal{K}}(\boldsymbol{p}) \triangleq \partial\boldsymbol{\mathcal{F}}(\boldsymbol{p})/\partial\boldsymbol{p}. \tag{2}$$

The matrix $\boldsymbol{K} = \boldsymbol{\mathcal{K}}(\boldsymbol{p}) \in \mathbb{R}^{3N_\mathcal{V} \times 3N_\mathcal{V}}$ is referred to as the global tangent stiffness matrix. As shown in [Capell et al. 2002a], $\boldsymbol{\mathcal{K}}$ is a quadratic function in an StVK material.

This section focuses on the derivation of the analytical formulations and the algorithms for the functions $\boldsymbol{\mathcal{F}}$ and $\boldsymbol{\mathcal{K}}$. They will be derived through sections 3.2 to 3.5. After that, section 3.6 will compare the new algorithms to conventional methods in terms of the numbers of floating-point operations.

3.2 Strain Energy Density of StVK Material

Deformation of a continuum can be represented by a vector-valued vector function that maps the initial position $\boldsymbol{p}^{\mathrm{ini}}$ of a point in the continuum to its current position $\boldsymbol{p}$. The GL strain tensor is defined by using the partial derivative of the function as follows:

$$\boldsymbol{E} \triangleq \frac{1}{2}\left(\boldsymbol{F}^{\mathrm{T}}\boldsymbol{F} - \boldsymbol{I}_3\right) \in \mathbb{R}^{3\times 3}, \quad \boldsymbol{F} \triangleq \frac{\partial \boldsymbol{p}}{\partial \boldsymbol{p}^{\mathrm{ini}}} \in \mathbb{R}^{3\times 3}. \tag{3}$$

The tensor $\boldsymbol{F}$ is referred to as the deformation gradient tensor.

It follows from (3) that the GL strain tensor $\boldsymbol{E}$ is a symmetric tensor that possesses six independent entries. For the convenience of derivation, let us define the following vector:

$$\boldsymbol{\varepsilon} \triangleq \left[\begin{array}{cccccc} \varepsilon_{xx} & \varepsilon_{yy} & \varepsilon_{zz} & 2\varepsilon_{xy} & 2\varepsilon_{yz} & 2\varepsilon_{zx} \end{array}\right]^{\mathrm{T}} \in \mathbb{R}^6, \tag{4}$$

of which the entries are taken from the tensor $\boldsymbol{E}$, which is

$$\boldsymbol{E} = \left[\begin{array}{ccc} \varepsilon_{xx} & \varepsilon_{xy} & \varepsilon_{zx} \\ \varepsilon_{xy} & \varepsilon_{yy} & \varepsilon_{yz} \\ \varepsilon_{zx} & \varepsilon_{yz} & \varepsilon_{zz} \end{array}\right]. \tag{5}$$

Then, in an isotropic StVK material, the strain energy density w can be described as follows:

$$w \triangleq \boldsymbol{\varepsilon}^{\mathrm{T}}\boldsymbol{D}\boldsymbol{\varepsilon}/2 \tag{6}$$

where

$$\boldsymbol{D} \triangleq \left[\begin{array}{cccccc} \lambda+2\mu & \lambda & \lambda & 0 & 0 & 0 \\ \lambda & \lambda+2\mu & \lambda & 0 & 0 & 0 \\ \lambda & \lambda & \lambda+2\mu & 0 & 0 & 0 \\ 0 & 0 & 0 & \mu & 0 & 0 \\ 0 & 0 & 0 & 0 & \mu & 0 \\ 0 & 0 & 0 & 0 & 0 & \mu \end{array}\right] \in \mathbb{R}^{6\times 6}. \tag{7}$$

Here, λ and μ are Lamé constants, which are $\lambda = E\nu/(1+\nu)/(1-2\nu)$ and $\mu = E/2/(1+\nu)$ where E and ν are Young's modulus and Poisson's ratio, respectively. Anisotropy can be included by modifying the matrix $\boldsymbol{D}$ appropriately.

3.3 Formulation of Strain Energy

Consider a tetrahedron $\mathcal{T}$, which includes six edges ($\mathcal{E}$s) and four vertices ($\mathcal{V}$s), in the StVK material. Let $\boldsymbol{p}^{\mathrm{ini}}(\mathcal{V}) \in \mathbb{R}^3$ denote the initial position of a vertex $\mathcal{V}$. Let us define the following quantities:

$$\tilde{\boldsymbol{p}}^{\mathrm{ini}}(\mathcal{E}) \triangleq \boldsymbol{p}^{\mathrm{ini}}(\mathcal{V}_{\mathrm{e}}(\mathcal{E},0)) - \boldsymbol{p}^{\mathrm{ini}}(\mathcal{V}_{\mathrm{e}}(\mathcal{E},1)) \tag{8}$$

$$\tilde{\boldsymbol{p}}(\mathcal{E}) \triangleq \boldsymbol{p}(\mathcal{V}_{\mathrm{e}}(\mathcal{E},0)) - \boldsymbol{p}(\mathcal{V}_{\mathrm{e}}(\mathcal{E},1)) \tag{9}$$

where $\mathcal{V}_{\mathrm{e}}(\mathcal{E}, j)$ denotes the jth ($j = 0$ or 1) vertex of the edge $\mathcal{E}$. The vectors $\tilde{\boldsymbol{p}}^{\mathrm{ini}}(\mathcal{E})$ and $\tilde{\boldsymbol{p}}(\mathcal{E})$ are edge vectors at the initial and deformed states, respectively. Hereafter, quantities specific to the tetrahedron $\mathcal{T}$ are denoted with the subscript $\mathcal{T}$. Assuming the deformation gradient $\boldsymbol{F}$ takes a constant value $\boldsymbol{F}_\mathcal{T}$ in the region $\mathcal{T}$, it satisfies $\boldsymbol{F}_\mathcal{T}\tilde{\boldsymbol{p}}^{\mathrm{ini}}(\mathcal{E}) = \tilde{\boldsymbol{p}}(\mathcal{E})$. Therefore, from the definition of $\boldsymbol{E}$ in (3), the strain tensor $\boldsymbol{E}_\mathcal{T}$ in the tetrahedron $\mathcal{T}$ satisfies the following equation:

$$\begin{aligned} \tilde{\boldsymbol{p}}^{\mathrm{ini}}(\mathcal{E})^{\mathrm{T}}\boldsymbol{E}_\mathcal{T}\tilde{\boldsymbol{p}}^{\mathrm{ini}}(\mathcal{E}) &= \frac{1}{2}\tilde{\boldsymbol{p}}^{\mathrm{ini}}(\mathcal{E})^{\mathrm{T}}\left(\boldsymbol{F}_\mathcal{T}{}^{\mathrm{T}}\boldsymbol{F}_\mathcal{T} - \boldsymbol{I}_3\right)\tilde{\boldsymbol{p}}^{\mathrm{ini}}(\mathcal{E}) \\ &= \frac{1}{2}\left(\|\tilde{\boldsymbol{p}}(\mathcal{E})\|^2 - \|\tilde{\boldsymbol{p}}^{\mathrm{ini}}(\mathcal{E})\|^2\right). \end{aligned} \tag{10}$$

For the convenience of derivation, let us define

$$\tilde{\boldsymbol{q}}^{\mathrm{ini}}(\mathcal{E}) \triangleq \left[\begin{array}{cccccc} \tilde{x}^2, & \tilde{y}^2, & \tilde{z}^2, & \tilde{x}\tilde{y}, & \tilde{y}\tilde{z}, & \tilde{z}\tilde{x} \end{array}\right]^{\mathrm{T}} \in \mathbb{R}^6 \tag{11}$$

where $\tilde{x}$, $\tilde{y}$, and $\tilde{z}$ are taken from the vector $\tilde{\boldsymbol{p}}^{\mathrm{ini}}(\mathcal{E}) = [\tilde{x}, \tilde{y}, \tilde{z}]^{\mathrm{T}}$. Then, (10) can be rewritten as follows:

$$\tilde{\boldsymbol{q}}^{\mathrm{ini}}(\mathcal{E})^{\mathrm{T}}\boldsymbol{\varepsilon}_\mathcal{T} = \frac{1}{2}\left(\|\tilde{\boldsymbol{p}}(\mathcal{E})\|^2 - \|\tilde{\boldsymbol{p}}^{\mathrm{ini}}(\mathcal{E})\|^2\right), \tag{12}$$

in which $\tilde{\boldsymbol{p}}^{\mathrm{ini}}(\mathcal{E})$ and $\boldsymbol{E}_\mathcal{T}$ are replaced by $\tilde{\boldsymbol{q}}^{\mathrm{ini}}(\mathcal{E})$ and $\boldsymbol{\varepsilon}_\mathcal{T}$.

Let the six edges of the tetrahedron $\mathcal{T}$ be denoted by $\mathcal{E}_i$ ($i = 0, 1, \cdots, 5$). Stacking (12) with respect to the six $\mathcal{E}$s into a vector form yields $\boldsymbol{Q}_\mathcal{T}\boldsymbol{\varepsilon}_\mathcal{T} = \boldsymbol{\zeta}_\mathcal{T}/2$ where

$$\boldsymbol{Q}_\mathcal{T} \triangleq \left[\begin{array}{cccc} \tilde{\boldsymbol{q}}^{\mathrm{ini}}(\mathcal{E}_0) & \tilde{\boldsymbol{q}}^{\mathrm{ini}}(\mathcal{E}_1) & \cdots & \tilde{\boldsymbol{q}}^{\mathrm{ini}}(\mathcal{E}_5) \end{array}\right]^{\mathrm{T}} \in \mathbb{R}^{6\times 6} \tag{13}$$

$$\boldsymbol{\zeta}_\mathcal{T} \triangleq \left[\begin{array}{cccc} \zeta(\mathcal{E}_0) & \zeta(\mathcal{E}_1) & \cdots & \zeta(\mathcal{E}_5) \end{array}\right]^{\mathrm{T}} \in \mathbb{R}^6 \tag{14}$$

$$\zeta(\mathcal{E}) \triangleq \|\tilde{\boldsymbol{p}}(\mathcal{E})\|^2 - H(\mathcal{E}) \in \mathbb{R} \tag{15}$$

$$H(\mathcal{E}) \triangleq \|\tilde{\boldsymbol{p}}^{\mathrm{ini}}(\mathcal{E})\|^2 \in \mathbb{R}. \tag{16}$$

It can be easily proven that the matrix $\boldsymbol{Q}_\mathcal{T}$ has an inverse matrix if and only if the four vertices are not on a single plane. Thus, the strain vector $\boldsymbol{\varepsilon}$ can be obtained as follows:

$$\boldsymbol{\varepsilon}_\mathcal{T} = \frac{1}{2}\boldsymbol{Q}_\mathcal{T}{}^{-1}\boldsymbol{\zeta}_\mathcal{T}. \tag{17}$$

A similar expression, which linearly relates the GL strain tensor to the squared edge lengths, has already been known in two-dimensional (triangular) cases, such as thin shells [Gingold et al. 2004]. In three-dimensional, tetrahedral cases, such an expression cannot be found in the literature.

Based on (6) and (17), the strain energy in the tetrahedron $\mathcal{T}$ can be obtained as follows:

$$W_\mathcal{T} \triangleq \frac{C^{\mathrm{ini}}_\mathcal{T}}{6}w_\mathcal{T} = \frac{C^{\mathrm{ini}}_\mathcal{T}}{12}\boldsymbol{\varepsilon}_\mathcal{T}{}^{\mathrm{T}}\boldsymbol{D}_\mathcal{T}\boldsymbol{\varepsilon}_\mathcal{T} = \frac{1}{4}\boldsymbol{\zeta}_\mathcal{T}{}^{\mathrm{T}}\boldsymbol{L}_\mathcal{T}\boldsymbol{\zeta}_\mathcal{T} \in \mathbb{R} \tag{18}$$

where $C^{\mathrm{ini}}_\mathcal{T}$ is the six-fold initial volume of the tetrahedron $\mathcal{T}$ and

$$\boldsymbol{L}_\mathcal{T} \triangleq \frac{C^{\mathrm{ini}}_\mathcal{T}}{12}\boldsymbol{Q}_\mathcal{T}{}^{-T}\boldsymbol{D}_\mathcal{T}\boldsymbol{Q}_\mathcal{T}{}^{-1} \in \mathbb{R}^{6\times 6}. \tag{19}$$

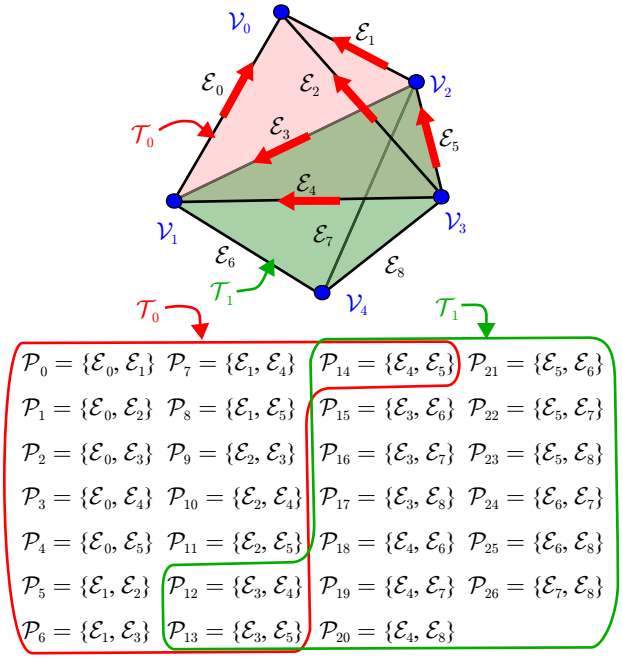


Fig. 2. A mesh $\mathcal{M}$ that contains two tetrahedra ($\mathcal{T}$s), nine edges ($\mathcal{E}$s), five vertices ($\mathcal{V}$s), and 27 TSEPs ($\mathcal{P}$s).

By letting $L_{\mathcal{T}}(\mathcal{E}_i, \mathcal{E}_j)$ denote the (i, j)-th entry of $\boldsymbol{L}_{\mathcal{T}}$, (18) can be rewritten as follows:

$$W_{\mathcal{T}} = \sum_{\mathcal{E}_a \in \mathcal{T}} \sum_{\mathcal{E}_b \in \mathcal{T}} \frac{1}{4} \zeta(\mathcal{E}_a) L_{\mathcal{T}}(\mathcal{E}_a, \mathcal{E}_b) \zeta(\mathcal{E}_b), \tag{20}$$

in which the strain energy is represented as a quadratic function of the squared edge lengths $\|\tilde{\boldsymbol{p}}(\mathcal{E})\|^2$, as is apparent from (15).

The total strain energy in the mesh $\mathcal{M}$, which is composed of multiple $\mathcal{T}$s, is written as follows:

$$W \triangleq \sum_{\mathcal{T} \in \mathcal{M}} W_{\mathcal{T}} = \sum_{\mathcal{E}_a \in \mathcal{M}} \sum_{\mathcal{E}_b \in \mathcal{M}} \frac{1}{4} \zeta(\mathcal{E}_a) \left(\sum_{\mathcal{T} \in \mathcal{M}} L_{\mathcal{T}}(\mathcal{E}_a, \mathcal{E}_b) \right) \zeta(\mathcal{E}_b). \tag{21}$$

Here, we used the fact that $\zeta(\mathcal{E})$ is an edge-specific quantity that does not depend on a specific tetrahedron. In contrast, $L_{\mathcal{T}}(\mathcal{E}_a, \mathcal{E}_b)$ depends on the tetrahedron $\mathcal{T}$, being unable to be taken out of the summation over $\mathcal{T} \in \mathcal{M}$. However, by letting $L(\mathcal{E}_a, \mathcal{E}_b) = \sum_{\mathcal{T} \in \mathcal{M}} L_{\mathcal{T}}(\mathcal{E}_a, \mathcal{E}_b)$, (21) can be simplified as follows:

$$W = \sum_{\mathcal{E}_a \in \mathcal{M}} \sum_{\mathcal{E}_b \in \mathcal{M}} \frac{1}{4} \zeta(\mathcal{E}_a) L(\mathcal{E}_a, \mathcal{E}_b) \zeta(\mathcal{E}_b). \tag{22}$$

The total strain energy of the whole mesh is now represented as a function of the edge lengths $\|\tilde{\boldsymbol{p}}(\mathcal{E})\|$, $\mathcal{E} \in \mathcal{M}$.

We can rewrite the expression (22) into a more convenient form. The definition of $L(\mathcal{E}_a, \mathcal{E}_b)$ implies that it is symmetric (i.e., $L(\mathcal{E}_a, \mathcal{E}_b) = L(\mathcal{E}_b, \mathcal{E}_a)$) and that it takes a non-zero value only if the edges $\mathcal{E}_a$ and $\mathcal{E}_b$ belong to at least one common tetrahedron. Thus, it is convenient to define such pairs of edges as a new class of geometric primitives, which we name *tetrahedron-sharing edge pairs* (*TSEP*s). A TSEP, denoted hereafter $\mathcal{P}$, is an ordered pair of edges. We hereafter denote the ith ($i = 0, 1$) edge of the TSEP $\mathcal{P}$ by $\mathcal{E}_{\mathrm{p}}(\mathcal{P}, i)$. One tetrahedron includes 15 TSEPs as illustrated in Fig. 2. Every non-zero $L(\mathcal{E}_a, \mathcal{E}_b)$ can be associated with either of an edge (if $\mathcal{E}_a = \mathcal{E}_b$) or a TSEP (otherwise), and therefore we can define the following notational conventions:

$$L(\mathcal{E}) \triangleq L(\mathcal{E}, \mathcal{E}), \quad L(\mathcal{P}) \triangleq L(\mathcal{E}_{\mathrm{p}}(\mathcal{P}, 0), \mathcal{E}_{\mathrm{p}}(\mathcal{P}, 1)). \tag{23}$$

By using them, the total strain energy W in (22) can be rewritten as follows:

$$W = \sum_{\mathcal{E} \in \mathcal{M}} \frac{1}{4} L(\mathcal{E}) \zeta(\mathcal{E})^2 + \sum_{\mathcal{P} \in \mathcal{M}} \frac{1}{2} L(\mathcal{P}) \zeta(\mathcal{E}_{\mathrm{p}}(\mathcal{P}, 0)) \zeta(\mathcal{E}_{\mathrm{p}}(\mathcal{P}, 1)), \tag{24}$$

which describes the contribution of edges and TSEPs separately.

Some useful physical interpretations can be drawn from the expression (24). It must be noted that Delingette [2008] has already arrived at a similar expression in formulating the deformation of two-dimensional objects, such as shells and membranes. For three-dimensional solids, however, similar expressions cannot be found in the literature. Recalling that $\zeta(\mathcal{E})$ is the change in the squared length of the edge $\mathcal{E}$ as defined in (15) and (16), one can show that the first term of (24) represents the sum of elastic energy stored in nonlinear springs coinciding with the edges. These springs correspond to what Delingette has referred to as "tensile biquadratic springs." The force produced by one of these springs is a cubic function of its length, as illustrated in Fig. 3. As long as the second term is left out of account, the energy W in (24) can be viewed as the elastic energy of an SN model whose springs are nonlinear.

The second term of (24), on the other hand, is not straightforward to physically interpret; it represents the energy produced by a coupled effect of paired edges. The difficulty in approximating an FE

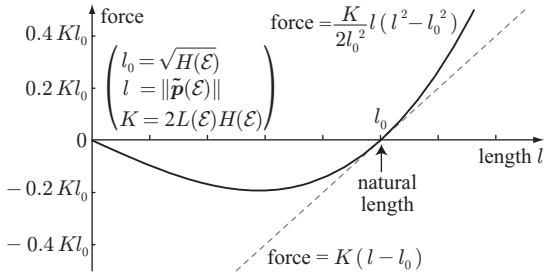


Fig. 3. The characteristic of the nonlinear springs that appear in (24) (solid curve) and its linear approximation (dashed line).

model by an SN model has been attracting attentions of researchers [Van Gelder 1998; Lloyd et al. 2007], but (24) shows that it can be attributed to the second term of (24). In the two-dimensional (triangular) case, Delingette [2008] attributes the coupled effects of paired edges to imaginary "angular biquadratic springs," each of which produces a force according to the angle between two intersecting edges and the lengths of them. In the three-dimensional (tetrahedral) case, however, the concept of "angular" springs does not apply because paired edges may or may not cross each other in a three-dimensional SN. His equations, although successful in two-dimensional cases, do not seem straightforward to be extended to deal with three-dimensional cases because they depend on many triangular-specific geometric formulas such as the sine law and the cosine law.

3.4 Formulation and Computation of Vertex Forces

Due to the principle of virtual work, the force $\boldsymbol{f}(\mathcal{V})$ ($\in \mathbb{R}^3$) can be obtained as a partial derivative of the strain energy W as follows:

$$\begin{aligned}\boldsymbol{f}(\mathcal{V}) = \frac{\partial W}{\partial \boldsymbol{p}(\mathcal{V})^{\mathrm{T}}} &= \sum_{\mathcal{E}_a \in \mathcal{M}} \sum_{\mathcal{E}_b \in \mathcal{M}} \frac{1}{2} \left(\frac{\partial \zeta(\mathcal{E}_a)}{\partial \boldsymbol{p}(\mathcal{V})^{\mathrm{T}}} \right) L(\mathcal{E}_a, \mathcal{E}_b) \zeta(\mathcal{E}_b) \\ &= \sum_{\mathcal{E} \in \mathcal{M}} \mathrm{t}(\mathcal{V}, \mathcal{E}) \tilde{\boldsymbol{p}}(\mathcal{E}) g(\mathcal{E}) \\ &= \sum_{\substack{\mathcal{E} \in \mathcal{M} \\ \mathcal{V}_{\mathrm{e}}(\mathcal{E},0)=\mathcal{V}}} \tilde{\boldsymbol{p}}(\mathcal{E}) g(\mathcal{E}) - \sum_{\substack{\mathcal{E} \in \mathcal{M} \\ \mathcal{V}_{\mathrm{e}}(\mathcal{E},1)=\mathcal{V}}} \tilde{\boldsymbol{p}}(\mathcal{E}) g(\mathcal{E}) \end{aligned} \quad (25)$$

where

$$g(\mathcal{E}) \triangleq \sum_{\mathcal{E}_b \in \mathcal{M}} L(\mathcal{E}, \mathcal{E}_b) \zeta(\mathcal{E}_b) \ \in \mathbb{R} \quad (26)$$

$$\mathrm{t}(\mathcal{V}, \mathcal{E}) \triangleq \begin{cases} 1 & \text{if } \mathcal{V} = \mathcal{V}_{\mathrm{e}}(\mathcal{E}, 0) \\ -1 & \text{if } \mathcal{V} = \mathcal{V}_{\mathrm{e}}(\mathcal{E}, 1) \\ 0 & \text{otherwise (if } \mathcal{V} \text{ is not included in } \mathcal{E}). \end{cases} \quad (27)$$

The definition (9) of $\tilde{\boldsymbol{p}}(\mathcal{E})$ implies $\partial \tilde{\boldsymbol{p}}(\mathcal{E}) / \partial \boldsymbol{p}(\mathcal{V}) = \mathrm{t}(\mathcal{V}, \mathcal{E}) \boldsymbol{I}_3$, which is used in (25).

By using the notations in (23), (26) can be written as follows:

$$\begin{aligned} g(\mathcal{E}) = L(\mathcal{E}) \zeta(\mathcal{E}) &+ \sum_{\substack{\mathcal{P} \in \mathcal{M} \\ \mathcal{E}_{\mathrm{p}}(\mathcal{P},1)=\mathcal{E}}} L(\mathcal{P}) \zeta(\mathcal{E}_{\mathrm{p}}(\mathcal{P}, 0)) \\ &+ \sum_{\substack{\mathcal{P} \in \mathcal{M} \\ \mathcal{E}_{\mathrm{p}}(\mathcal{P},0)=\mathcal{E}}} L(\mathcal{P}) \zeta(\mathcal{E}_{\mathrm{p}}(\mathcal{P}, 1)). \end{aligned} \quad (28)$$

The computation of the expression (25), which depends on (28), can be performed in the following procedure:

ALGORITHM algF [$\boldsymbol{p}$]
  // depending on $\{H(\mathcal{E}), L(\mathcal{E})\}_{\mathcal{E} \in \mathcal{M}}, \{L(\mathcal{P})\}_{\mathcal{P} \in \mathcal{M}}$
  FOR $\mathcal{V} \in \mathcal{M}$; $\boldsymbol{f}(\mathcal{V}) := \boldsymbol{o}_3$
  FOR $\mathcal{E} \in \mathcal{M}$
    $\tilde{\boldsymbol{p}}(\mathcal{E}) := \boldsymbol{p}(\mathcal{V}_{\mathrm{e}}(\mathcal{E}, 0)) - \boldsymbol{p}(\mathcal{V}_{\mathrm{e}}(\mathcal{E}, 1))$
    $\zeta(\mathcal{E}) := \|\tilde{\boldsymbol{p}}(\mathcal{E})\|^2 - H(\mathcal{E})$
    $g(\mathcal{E}) := L(\mathcal{E}) \zeta(\mathcal{E})$
  END FOR
  FOR $\mathcal{P} \in \mathcal{M}$
    $g(\mathcal{E}_{\mathrm{p}}(\mathcal{P}, 1)) \mathrel{+}= L(\mathcal{P}) \zeta(\mathcal{E}_{\mathrm{p}}(\mathcal{P}, 0))$
    $g(\mathcal{E}_{\mathrm{p}}(\mathcal{P}, 0)) \mathrel{+}= L(\mathcal{P}) \zeta(\mathcal{E}_{\mathrm{p}}(\mathcal{P}, 1))$
  END FOR
  FOR $\mathcal{E} \in \mathcal{M}$
    $\boldsymbol{f}_{\mathrm{tmp}} := \tilde{\boldsymbol{p}}(\mathcal{E}) g(\mathcal{E})$
    $\boldsymbol{f}(\mathcal{V}_{\mathrm{e}}(\mathcal{E}, 0)) \mathrel{+}= \boldsymbol{f}_{\mathrm{tmp}}$
    $\boldsymbol{f}(\mathcal{V}_{\mathrm{e}}(\mathcal{E}, 1)) \mathrel{-}= \boldsymbol{f}_{\mathrm{tmp}}$
  END FOR
  RETURN [$\boldsymbol{f}, \{\tilde{\boldsymbol{p}}(\mathcal{E}), g(\mathcal{E})\}_{\mathcal{E} \in \mathcal{M}}$]

The input to this algorithm is $\boldsymbol{p}$ (i.e., $\{\boldsymbol{p}(\mathcal{V})\}_{\mathcal{V} \in \mathcal{M}}$) and the main output is $\boldsymbol{f}$ (i.e., $\{\boldsymbol{f}(\mathcal{V})\}_{\mathcal{V} \in \mathcal{M}}$). The rest two of the outputs, $\tilde{\boldsymbol{p}}(\mathcal{E})$ and $g(\mathcal{E})$, are used to compute the tangent stiffness matrix in the next section.

3.5 Formulation and Computation of Tangent Stiffness Matrix

The global tangent stiffness matrix $\boldsymbol{K} = \partial \boldsymbol{f} / \partial \boldsymbol{p}$ is a $3N_{\mathcal{V}} \times 3N_{\mathcal{V}}$ symmetric matrix. Its 3×3 blocks are obtained by $\partial \boldsymbol{f}(\mathcal{V}_a) / \partial \boldsymbol{p}(\mathcal{V}_b)$. This partial derivative is not $\boldsymbol{O}_{3\times 3}$ only if $\mathcal{V}_a$ and $\mathcal{V}_b$ share a common edge (i.e., $\mathcal{V}_a = \mathcal{V}_b$ or $\exists \mathcal{E}$ s.t. $\{\mathcal{V}_a, \mathcal{V}_b\} \in \mathcal{E}$). Therefore, it can be associated with either of a vertex or an edge as follows:

$$\boldsymbol{K}(\mathcal{V}) \triangleq \frac{\partial \boldsymbol{f}(\mathcal{V})}{\partial \boldsymbol{p}(\mathcal{V})}, \ \boldsymbol{K}(\mathcal{E}) \triangleq \frac{\partial \boldsymbol{f}(\mathcal{V}_{\mathrm{e}}(\mathcal{E}, 0))}{\partial \boldsymbol{p}(\mathcal{V}_{\mathrm{e}}(\mathcal{E}, 1))} = \left(\frac{\partial \boldsymbol{f}(\mathcal{V}_{\mathrm{e}}(\mathcal{E}, 1))}{\partial \boldsymbol{p}(\mathcal{V}_{\mathrm{e}}(\mathcal{E}, 0))} \right)^{\mathrm{T}}. \quad (29)$$

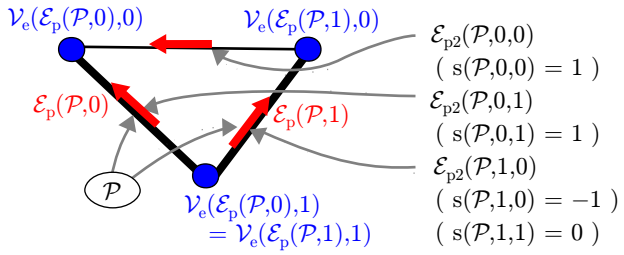
The matrices $\boldsymbol{K}(\mathcal{V})$, $\mathcal{V} \in \mathcal{M}$, are diagonal blocks of the global tangent matrix $\boldsymbol{K}$, and the matrices $\boldsymbol{K}(\mathcal{E})$, $\mathcal{E} \in \mathcal{M}$, are off-diagonal blocks. It can be easily verified (as in [Coutinho et al. 2001]) that $\boldsymbol{K}(\mathcal{V})$ and $\boldsymbol{K}(\mathcal{E})$ are constrained by the following relation:

$$\boldsymbol{K}(\mathcal{V}) = - \sum_{\substack{\mathcal{E} \in \mathcal{M} \\ \mathrm{t}(\mathcal{E},\mathcal{V})=1}} \boldsymbol{K}(\mathcal{E}) - \sum_{\substack{\mathcal{E} \in \mathcal{M} \\ \mathrm{t}(\mathcal{E},\mathcal{V})=-1}} \boldsymbol{K}(\mathcal{E})^{\mathrm{T}}, \quad (30)$$

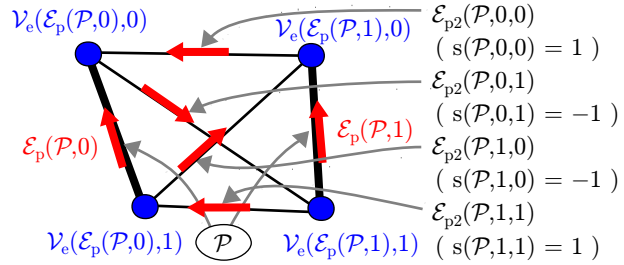
which allows us to focus on the problem of formulating $\boldsymbol{K}(\mathcal{E})$.

A tedious but straightforward derivation detailed in Appendix A shows that

$$\begin{aligned} \boldsymbol{K}(\mathcal{E}) = &-g(\mathcal{E}) \boldsymbol{I}_3 - 2 \tilde{\boldsymbol{p}}(\mathcal{E}) L(\mathcal{E}) \tilde{\boldsymbol{p}}(\mathcal{E})^{\mathrm{T}} \\ &+ \sum_{\substack{\mathcal{P} \in \mathcal{M} \\ \mathcal{E}_{\mathrm{p2}}(\mathcal{P},i,j)=\mathcal{E} \wedge \mathrm{s}(\mathcal{P},i,j)=1}} \sum_{i \in \{0,1\}} \sum_{j \in \{0,1\}} (-1)^{i+j} \boldsymbol{K}_p(\mathcal{P}) \\ &+ \sum_{\substack{\mathcal{P} \in \mathcal{M} \\ \mathcal{E}_{\mathrm{p2}}(\mathcal{P},i,j)=\mathcal{E} \wedge \mathrm{s}(\mathcal{P},i,j)=-1}} \sum_{i \in \{0,1\}} \sum_{j \in \{0,1\}} (-1)^{i+j} \boldsymbol{K}_p(\mathcal{P})^{\mathrm{T}} \end{aligned} \quad (31)$$



(a) Type-1 TSEP



(b) Type-2 TSEP

Fig. 4. The operators ${\rm s}(\mathcal{P},i,j)$ and $\mathcal{E}_{\rm p2}(\mathcal{P},i,j)$.

where

$$\boldsymbol{K}_p(\mathcal{P}) \triangleq 2\tilde{\boldsymbol{p}}(\mathcal{E}_{\rm p}(\mathcal{P},0))L(\mathcal{P})\tilde{\boldsymbol{p}}(\mathcal{E}_{\rm p}(\mathcal{P},1))^{\rm T} \tag{32}$$

$$\mathcal{E}_{\rm p2}(\mathcal{P},i,j) \triangleq \mathcal{E}_{\rm v}(\mathcal{V}_{\rm e}(\mathcal{E}_{\rm p}(\mathcal{P},0),i),\mathcal{V}_{\rm e}(\mathcal{E}_{\rm p}(\mathcal{P},1),j)) \tag{33}$$

$$ {\rm s}(\mathcal{P},i,j) \triangleq {\rm t}(\mathcal{V}_{\rm e}(\mathcal{E}_{\rm p}(\mathcal{P},0),i),\mathcal{E}_{\rm p2}(\mathcal{P},i,j)). \tag{34}$$

Fig. 4 illustrates the definitions (33) and (34). Here, $\mathcal{E}_{\rm v}(\mathcal{V}_a,\mathcal{V}_b)$ is the edge connecting the vertices $\mathcal{V}_a$ and $\mathcal{V}_b$ if such an edge exists. The integer flag ${\rm s}(\mathcal{P},i,j)$ in (34) indicates the orientation of the edge; it is 1 if $\mathcal{E}_{\rm p2}(\mathcal{P},i,j)$ is oriented from $\mathcal{E}_{\rm p}(\mathcal{P},1)$-side to $\mathcal{E}_{\rm p}(\mathcal{P},0)$-side, is -1 if reversed, and is 0 if $\mathcal{E}_{\rm p2}(\mathcal{P},i,j)$ does not exist. As is illustrated, TSEPs can be classified into two types by whether the edges in the TSEP share a common vertex or not. With $\mathcal{P}$ being a type-1 TSEP, $\mathcal{E}_{\rm p2}(\mathcal{P},i,j)$ may not exist due to the shared vertex; in such cases, the correspondent ${\rm s}(\mathcal{P},i,j)$ is set to be 0.

The computation of the expression (31) can be performed in the following procedures:

ALGORITHM algKE $[\,\{\tilde{\boldsymbol{p}}(\mathcal{E}),g(\mathcal{E})\}_{\mathcal{E}\in\mathcal{M}}\,]$
  // depending on: $\{L(\mathcal{E})\}_{\mathcal{E}\in\mathcal{M}},\{L(\mathcal{P})\}_{\mathcal{P}\in\mathcal{M}}$
  FOR $\mathcal{E}\in\mathcal{M}$
    $\boldsymbol{K}(\mathcal{E}) := -g(\mathcal{E})\boldsymbol{I}_3 - 2\tilde{\boldsymbol{p}}(\mathcal{E})L(\mathcal{E})\tilde{\boldsymbol{p}}(\mathcal{E})^{\rm T}$
  END FOR
  FOR $\mathcal{P}\in\mathcal{M}$
    $\boldsymbol{K}_p := 2\tilde{\boldsymbol{p}}(\mathcal{E}_{\rm p}(\mathcal{P},0))L(\mathcal{P})\tilde{\boldsymbol{p}}(\mathcal{E}_{\rm p}(\mathcal{P},1))^{\rm T}$
    FOR $i\in\{0,1\}$, $j\in\{0,1\}$
      IF ${\rm s}(\mathcal{P},i,j) = 1$
        $\boldsymbol{K}(\mathcal{E}_{\rm p2}(\mathcal{P},i,j)) \mathrel{+}= (-1)^{i+j}\boldsymbol{K}_p$
      ELSE IF ${\rm s}(\mathcal{P},i,j) = -1$
        $\boldsymbol{K}(\mathcal{E}_{\rm p2}(\mathcal{P},i,j)) \mathrel{+}= (-1)^{i+j}\boldsymbol{K}_p{}^{\rm T}$
      END IF
    END FOR
  END FOR
  RETURN $[\,\{\boldsymbol{K}(\mathcal{E})\}_{\mathcal{E}\in\mathcal{M}}\,]$.

This algorithm algKE must be run after the algorithm algF is run because it depends on $\tilde{\boldsymbol{p}}(\mathcal{E})$ and $g(\mathcal{E})$, which are produced by algF.

The diagonal blocks of the matrix $\boldsymbol{K}$ can be obtained by using the rule (30) in the following procedure:

ALGORITHM algKV $[\,\{\boldsymbol{K}(\mathcal{E})\}_{\mathcal{E}\in\mathcal{M}}\,]$
  FOR $\mathcal{V}\in\mathcal{M}$; $\boldsymbol{K}(\mathcal{V}) := \boldsymbol{O}_{3\times3}$
  FOR $\mathcal{E}\in\mathcal{M}$
    $\boldsymbol{K}(\mathcal{V}_{\rm e}(\mathcal{E},0)) \mathrel{-}= \boldsymbol{K}(\mathcal{E})$
    $\boldsymbol{K}(\mathcal{V}_{\rm e}(\mathcal{E},1)) \mathrel{-}= \boldsymbol{K}(\mathcal{E})^{\rm T}$
  END FOR
  RETURN $[\,\{\boldsymbol{K}(\mathcal{V})\}_{\mathcal{V}\in\mathcal{M}}\,]$.

Considering that $\boldsymbol{K}(\mathcal{V})$ are symmetric matrices each of which stores only six independent entries, the algorithm algKV only requires $12N_{\mathcal{E}}$ floating-point additions.

When the loops of "FOR $\mathcal{P}\in\mathcal{M}$" are removed from the algorithms algF and algKE, the algorithms reduce to those for simulating an SN model whose springs have the nonlinear characteristic illustrated in Fig. 3. This means that the user or the programmer can easily switch the FE model into the SN model by skipping this $\mathcal{P}$-loop if s/he needs to accelerate the computation by sacrificing the accuracy to some extent. Probably these nonlinear springs are computationally less expensive than the linear springs because they do not require square-root operations. The physical validity of such a nonlinear SN (NSN) model (without the $\mathcal{P}$-loop) is an open problem. However, considering that linear SN models have been practically accepted in spite of its having been questioned for their physical validity, some applications will accept the reduced NSN model in spite of its lack of physical validity. At least as long as the strain is sufficiently small, the nonlinear springs can be approximated by linear springs as illustrated in Fig. 3.

3.6 Comparison to previous methods

The computational cost of the presented method is now compared to those of previous methods in terms of the number of floating point operations (FLOPs). Although the number of FLOPs is not the only factor that determines the time of computation, it is indeed an important factor and its reduction is necessary before further optimization.

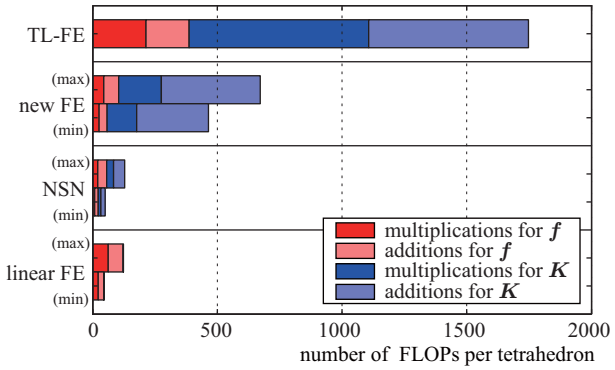


Fig. 5. Numbers of FLOPs per tetrahedron for computing $\boldsymbol{f}$ and $\boldsymbol{K}$; "TL-FE"=TL-FE algorithm, "new-FE"= new FE algorithm, "NSN"= the nonlinear SN model obtained by removing $\mathcal{P}$-loops from the "new" method, and "linear FE"=linear FE method.

The presented technique and many of previous nonlinear FE techniques reviewed in section 2 share a common theoretical basis with the conventional Total Lagrangian (TL) formulation [Bathe 1996], which employs GL strain tensor and PK2 stress tensor. In applications where the interactivity is not a concern, the forces and tangent stiffness matrices are usually computed by the numerical integration within polyhedral elements, which requires many FLOPs. Most of analytical expressions (with tetrahedral meshes) found in the literature are based on element-by-element formulations, and thus the number of FLOPs is proportional to $N_{\mathcal{T}}$. With an algorithm that can be straightforwardly derived from the standard TL formulation (hereafter, referred to as a "TL-FE algorithm"), the required number of FLOPs for computing $\boldsymbol{f}$ and $\boldsymbol{K}$ is $1747N_{\mathcal{T}}$ as detailed in Appendix B. This number is the sum of the numbers of multiplications and additions (including subtractions) for computing $\boldsymbol{f}$ and for computing $\boldsymbol{K}$, which are shown in the top bar of Fig. 5.

The number of FLOPs for the presented new FE method, on the other hand, is not proportional to $N_{\mathcal{T}}$ because it does not employ element-by-element computation. The numbers can be summarized as follows:[1]

$$N_{\mathrm{MLT},\boldsymbol{f},\mathrm{newFE}} = 7N_{\mathcal{E}} + 6N_{\mathcal{F}} + 6N_{\mathcal{T}} \tag{35a}$$
$$N_{\mathrm{ADD},\boldsymbol{f},\mathrm{newFE}} = 12N_{\mathcal{E}} + 6N_{\mathcal{F}} + 6N_{\mathcal{T}} \tag{35b}$$
$$N_{\mathrm{MLT},\boldsymbol{K},\mathrm{newFE}} = 9N_{\mathcal{E}} + 36N_{\mathcal{F}} + 36N_{\mathcal{T}} \tag{35c}$$
$$N_{\mathrm{ADD},\boldsymbol{K},\mathrm{newFE}} = 15N_{\mathcal{E}} + 81N_{\mathcal{F}} + 108N_{\mathcal{T}}. \tag{35d}$$

[1] Here, we used the fact that the numbers of type-1 and type-2 TSEPs ($N_{\mathcal{P}1}$ and $N_{\mathcal{P}2}$, respectively) satisfy $N_{\mathcal{P}1} = 3N_{\mathcal{F}}$ and $N_{\mathcal{P}2} = 3N_{\mathcal{T}}$. In addition, we assume that $2L(\mathcal{E})$ and $2L(\mathcal{P})$, which appear in algKE, are precomputed to avoid runtime multiplications between constants.

Here, $N_{\mathrm{MLT},\boldsymbol{f},\mathrm{newFE}}$ and $N_{\mathrm{ADD},\boldsymbol{f},\mathrm{newFE}}$ are the numbers of multiplications and additions, respectively, to compute $\boldsymbol{f}$. The numbers $N_{\mathrm{MLT},\boldsymbol{K},\mathrm{newFE}}$ and $N_{\mathrm{ADD},\boldsymbol{K},\mathrm{newFE}}$ are those to compute $\boldsymbol{K}$. The relations among $N_{\mathcal{V}}$, $N_{\mathcal{E}}$, $N_{\mathcal{F}}$, and $N_{\mathcal{T}}$ depend on the connectivity of the tetrahedra in the mesh. Among meshes composed of sufficiently many tetrahedra ($N_{\mathcal{T}} \gg 1$), the most sparsely connected one is that with an elongated structure in which all its edges are on its border, which satisfies $N_{\mathcal{V}} : N_{\mathcal{E}} : N_{\mathcal{F}} : N_{\mathcal{T}} \approx 1 : 3 : 3 : 1$. The opposite extreme is a densely connected mesh so that its surface-area-to-volume ratio is small. In such a mesh, a previous study [Gumhold et al. 1999] reports that the numbers typically become close to $N_{\mathcal{V}} : N_{\mathcal{E}} : N_{\mathcal{F}} : N_{\mathcal{T}} \approx 0.18 : 1.18 : 2 : 1$. Therefore, we can assume that the ratios for most of meshes normally used to model an object lie within the following ranges:

$$0.18 < \frac{N_{\mathcal{V}}}{N_{\mathcal{T}}} < 1, \quad 1.18 < \frac{N_{\mathcal{E}}}{N_{\mathcal{T}}} < 3, \quad 2 < \frac{N_{\mathcal{F}}}{N_{\mathcal{T}}} < 3. \tag{36}$$

Under the assumption (36), the numbers in (35) lie within the following ranges:

$$26.27N_{\mathcal{T}} < N_{\mathrm{MLT},\boldsymbol{f},\mathrm{newFE}} < 45N_{\mathcal{T}} \tag{37a}$$
$$32.18N_{\mathcal{T}} < N_{\mathrm{ADD},\boldsymbol{f},\mathrm{newFE}} < 60N_{\mathcal{T}} \tag{37b}$$
$$118.64N_{\mathcal{T}} < N_{\mathrm{MLT},\boldsymbol{K},\mathrm{newFE}} < 171N_{\mathcal{T}} \tag{37c}$$
$$287.73N_{\mathcal{T}} < N_{\mathrm{ADD},\boldsymbol{K},\mathrm{newFE}} < 396N_{\mathcal{T}}. \tag{37d}$$

These numbers are shown in the second and the third bars of Fig. 5. The figure shows that the new method is computationally much more efficient than the TL-FE algorithm at least in terms of the necessary numbers of FLOPs.

The FLOP counts for simpler but less accurate methods are now shown for the sake of comparison. In the linear FE methods (in the form of $\boldsymbol{f} = \boldsymbol{K}(\boldsymbol{p} - \boldsymbol{p}^{\mathrm{ini}})$ with a given constant stiffness matrix $\boldsymbol{K} \in \mathbb{R}^{3N_{\mathcal{V}} \times 3N_{\mathcal{V}}}$), the numbers of FLOPs are written as $N_{\mathrm{MLT},\boldsymbol{f},\mathrm{linFE}} = 9N_{\mathcal{V}} + 18N_{\mathcal{E}}$ and $N_{\mathrm{ADD},\boldsymbol{f},\mathrm{linFE}} = 6N_{\mathcal{V}} + 18N_{\mathcal{E}}$, where the sparsity of $\boldsymbol{K}$ is taken into account. Under the assumption (36), these numbers fall within the following ranges:

$$22.91N_{\mathcal{T}} < N_{\mathrm{MLT},\boldsymbol{f},\mathrm{linFE}} < 63N_{\mathcal{T}} \tag{38a}$$
$$22.36N_{\mathcal{T}} < N_{\mathrm{ADD},\boldsymbol{f},\mathrm{linFE}} < 60N_{\mathcal{T}}, \tag{38b}$$

which are shown in the bottom two bars in Fig. 5. The comparison between {(37a),(37b)} and (38) indicates that the inclusion of nonlinearity may or may not increase the numbers of FLOPs for computing $\boldsymbol{f}$, and that the increase is at most 17% (in the sum of multiplications and summations). As another example, the numbers for the NSN model, which is obtained by skipping all $\mathcal{P}$-loops from the algorithms algF and algKE, are shown in the fourth and fifth bars in Fig. 5. The figure shows that the NSN method is even faster than the linear FE method, although its physical validity is debatable.

A possible alternative to the algorithm algF is Picinbono et al.'s [2003] method. Although its precise number of FLOPs is not reported, the method requires larger number of FLOPs and longer

computational time than the linear FE method for computing $\boldsymbol{f}$ because it computes $\boldsymbol{f}$ by adding new quadratic and cubic terms to a standard linear term. In their implementation, the method is reported to require at least five times longer timestep size than the linear FE model. Another alternative to our method is the stiffness warping method [Müller and Gross 2004], which computes both $\boldsymbol{f}$ and $\boldsymbol{K}$. Their method can be viewed as a linear but rotationally-invariant approximation of an StVK-based FE method and its computation can be more stable than the StVK-based FE methods. Their method requires the polar decomposition of 3×3 matrices. In addition to this, it requires rotational coordinate transformation of all nonzero 3×3 blocks of the precomputed initial stiffness matrix and force vectors for every tetrahedron, which demands more than $54 \times 10 + 9 \times 4 = 576$ multiplications and $36 \times 10 + 6 \times 4 = 384$ additions per tetrahedron. This means that the stiffness warping method requires much more FLOPs than our presented method.

It must be noted that there are some other factors that influence the computational speed: e.g., the number of indirect addressing operations [Löhner 1994]. It must also be cautioned that most of the implicit integration schemes require iterative computation to solve algebraic equations after the computation of $\boldsymbol{f}$ and $\boldsymbol{K}$. This indeed can be a bottleneck for the acceleration of the whole simulation. However, within a given timestep size that is typically around 30 ms in interactive applications, the reduced time for computing $\boldsymbol{f}$ and $\boldsymbol{K}$ results in the increased accuracy of the iterative solver because it increases the maximum possible number of iterations. The severity of the bottleneck of the iterative solver depends on the necessary number of iterations, which highly depends on the level of accuracy needed by the application and the system to be simulated. For example, in simulations mainly dealing with slow motions or systems under high viscous resistance, the computation converges in a small number of iterations. The presented method could be beneficial in such cases. Nonetheless, many of iterative computational procedures are based on general-purpose, well-defined mathematical problems, which might be accelerated through future numerical techniques and/or hardware-level implementation techniques (e.g., [Bolz et al. 2003]).

The memory requirement might be another important issue in implementation. The presented method requires $H(\mathcal{E})$, $L(\mathcal{E})$, $2L(\mathcal{E})$, $L(\mathcal{P})$, and $2L(\mathcal{P})$ to be stored in the precomputation phase, which means that $3N_{\mathcal{E}} + 6N_{\mathcal{F}} + 6N_{\mathcal{T}}$ ($21.54N_{\mathcal{T}}$ to $33N_{\mathcal{T}}$) floating-point numbers must be stored. Picinbono et al's [2003] method, on the other hand, requires at least $10N_{\mathcal{V}} + 26N_{\mathcal{E}} + 18N_{\mathcal{F}} + 6N_{\mathcal{T}}$ ($74.48N_{\mathcal{T}}$ to $148N_{\mathcal{T}}$) floating-point numbers to be stored, according to Table 1 in [Picinbono et al. 2003], which is much larger than the presented technique. The requirement of the TL-FE algorithm is smaller than the presented FE method, being at least $3N_{\mathcal{V}} + 3N_{\mathcal{T}}$ ($3.54N_{\mathcal{T}}$ to $6N_{\mathcal{T}}$) floating-point numbers, at the price of increased number of FLOPs. The stiffness warping method [Müller and Gross 2004] requires a precomputed stiffness matrix and initial force vectors for every tetrahedra, which include more than $90N_{\mathcal{T}}$ numbers.

4. ADDITIONAL VOLUMETRIC STRAIN ENERGY

One disadvantage of the StVK material law is that it is invariant with respect to reflection, which is a drawback common to SN models. This means that these deformation models keep tetrahedra at fictitious equilibrium when they are inverted. One simple idea to avoid such a drawback is the use of additional forces and/or energy terms depending on volumetric (bulk) strain. Such an approach has been demonstrated in interactive applications [Picinbono et al. 2003; Teschner et al. 2004; Lloyd et al. 2007] and is supported by empirical and theoretical studies [Simo and Taylor 1991; Weiss et al. 1996; Doll and Schweizerhof 2000; Teran et al. 2003; Teran et al. 2005]. There are some different types of approaches [Irving et al. 2006; Irving et al. 2007], but they are not intended for interactive applications.

This conventional idea of additional volumetric forces is also useful in combination with our main contribution in section 3. Here, a precise algorithm based on an arbitrary volumetric strain energy function is described to ease future extension and to estimate the resultant increase in the number of FLOPs. With an additional volumetric strain energy term, the strain energy density w in (6), is rewritten as follows:

$$w = \boldsymbol{\varepsilon}^{\mathrm{T}} \boldsymbol{D} \boldsymbol{\varepsilon} / 2 + \psi_{\mathrm{vol}}(\theta) \tag{39}$$

where $\psi_{\mathrm{vol}}(\theta)$ is a positive definite scalar function and θ is the volumetric strain defined as $\theta = (\det(\boldsymbol{F}) - 1)/6$. The additional strain energy in a tetrahedron $\mathcal{T}$ can be described as follows:

$$W_{\mathcal{T}}^{\mathrm{vol}} = \Psi_{\mathcal{T}}^{\mathrm{vol}}(C_{\mathcal{T}} - C_{\mathcal{T}}^{\mathrm{ini}}) \tag{40}$$

where $\Psi_{\mathcal{T}}^{\mathrm{vol}}(c) \triangleq (C_{\mathcal{T}}^{\mathrm{ini}}/6)\psi_{\mathrm{vol}}(c/C_{\mathcal{T}}^{\mathrm{ini}})$ and $C_{\mathcal{T}}$ is the current sixfold volume of $\mathcal{T}$. Based on this function, a straightforward derivation detailed in Appendix C shows that the contributions of the additional volumetric strain energy can be added to $\boldsymbol{f}(\mathcal{V})$ and $\boldsymbol{K}(\mathcal{E})$ in the following procedure:

ALGORITHM $\mathrm{algB}_{\mathcal{T}}\,[\,\boldsymbol{f}, \{\tilde{\boldsymbol{p}}(\mathcal{E}), \boldsymbol{K}(\mathcal{E})\}_{\mathcal{E}\in\mathcal{T}}\,]$

$\boldsymbol{c}_0 := \tilde{\boldsymbol{p}}(\mathcal{E}_*) \times \tilde{\boldsymbol{p}}(\mathcal{E}_*)$

$\Delta C_{\mathcal{T}} := \tilde{\boldsymbol{p}}(\mathcal{E}_*)^{\mathrm{T}} \boldsymbol{c}_0 - C_{\mathcal{T}}^{\mathrm{ini}}$

$\Psi_1 := \Psi_{\mathcal{T}}^{\mathrm{vol}\prime}(\Delta C_{\mathcal{T}})$

$\Psi_2 := \Psi_{\mathcal{T}}^{\mathrm{vol}\prime\prime}(\Delta C_{\mathcal{T}})$

FOR $j \in \{1, 2\}$; $\quad \boldsymbol{c}_j := \tilde{\boldsymbol{p}}(\mathcal{E}_*) \times \tilde{\boldsymbol{p}}(\mathcal{E}_*)$

FOR $i \in \{0, 1, 2\}$; $\quad \boldsymbol{f}_{\mathcal{T}}^{\mathrm{vol}}(\mathcal{V}_j) := \boldsymbol{c}_j \Psi_1$

$\boldsymbol{f}_{\mathcal{T}}^{\mathrm{vol}}(\mathcal{V}_3) := -\boldsymbol{f}_{\mathcal{T}}^{\mathrm{vol}}(\mathcal{V}_0) - \boldsymbol{f}_{\mathcal{T}}^{\mathrm{vol}}(\mathcal{V}_1) - \boldsymbol{f}_{\mathcal{T}}^{\mathrm{vol}}(\mathcal{V}_2)$

FOR $j \in \{0, \cdots, 3\}$; $\boldsymbol{f}(\mathcal{V}_j) \mathrel{+}= \boldsymbol{f}_{\mathcal{T}}^{\mathrm{vol}}(\mathcal{V}_j)$

FOR $i \in \{0, 1, 2\}$; $\quad \boldsymbol{x}_i := \tilde{\boldsymbol{p}}(\mathcal{E}_i)\Psi_1$

FOR $i \in \{3, 4, 5\}$; $\quad \boldsymbol{x}_i := \boldsymbol{x}(\mathcal{E}_*) - \boldsymbol{x}(\mathcal{E}_*)$

FOR $j \in \{0, 1, 2\}$; $\quad \boldsymbol{y}_j := \boldsymbol{c}_j \Psi_2$

$\boldsymbol{y}_3 = -\boldsymbol{y}_0 - \boldsymbol{y}_1 - \boldsymbol{y}_2$

FOR $i \in \{0, \cdots, 5\}$; $\boldsymbol{K}(\mathcal{E}_i) \mathrel{+}= \boldsymbol{c}_* \boldsymbol{y}_*^{\mathrm{T}} + (-1)^* [\boldsymbol{x}_* \times]$
RETURN $[\boldsymbol{f}, \{\boldsymbol{K}(\mathcal{E})\}_{\mathcal{E}\in\mathcal{T}}]$.

Here, the subscripts $*$ denote integers appropriately chosen.

The algorithm $\mathrm{algB}_{\mathcal{T}}$ includes $(102 + n_{m\psi})$ multiplications and $(135 + n_{a\psi})$ additions, where $n_{m\psi}$ and $n_{a\psi}$ are the numbers of multiplications and additions, respectively, for $\Psi_{\mathcal{T}}^{\mathrm{vol}\prime}$ and $\Psi_{\mathcal{T}}^{\mathrm{vol}\prime\prime}$, which are typically a few. The comparison of these numbers to those in Fig. 5 indicates that the use of this element-by-element computation of $\mathrm{algB}_{\mathcal{T}}$ does not defeat the benefit of the method in section 3, because $\mathrm{algB}_{\mathcal{T}}$ introduces roughly $(237 + n_{m\psi} + n_{a\psi})/464.82 < 53\,\%$ of increase in the FLOP count while the conventional TL-FE method (without volume-preserving forces) requires more than 2.60 times as many. Besides, if the function $\psi_{\mathrm{vol}}(\theta)$ is chosen so that $\psi_{\mathrm{vol}}(\theta) = 0$ if $\theta > 0$, the amount of computation can be reduced by skipping expanded tetrahedra.

The whole procedure of the algorithm to obtain vertex forces $\boldsymbol{f}$ and the nonzero blocks of the global tangent stiffness matrix $\boldsymbol{K}$ from the vertex positions $\boldsymbol{p}$ is described as follows:

ALGORITHM $\mathrm{algALL}\,[\boldsymbol{p}]$
$[\boldsymbol{f}, \{\tilde{\boldsymbol{p}}(\mathcal{E}), g(\mathcal{E})\}_{\mathcal{E}\in\mathcal{M}}] := \mathrm{algF}\,[\boldsymbol{p}]$
$[\{\boldsymbol{K}(\mathcal{E})\}_{\mathcal{E}\in\mathcal{M}}] := \mathrm{algKE}\,[\{\tilde{\boldsymbol{p}}(\mathcal{E}), g(\mathcal{E})\}_{\mathcal{E}\in\mathcal{M}}]$
FOR $\mathcal{T} \in \mathcal{M}$
$[\boldsymbol{f}, \{\boldsymbol{K}(\mathcal{E})\}_{\mathcal{E}\in\mathcal{T}}] := \mathrm{algB}_{\mathcal{T}}\,[\boldsymbol{f}, \{\boldsymbol{K}(\mathcal{E}), \tilde{\boldsymbol{p}}(\mathcal{E})\}_{\mathcal{E}\in\mathcal{T}}]$
END FOR
$[\{\boldsymbol{K}(\mathcal{V})\}_{\mathcal{V}\in\mathcal{M}}] := \mathrm{algKV}\,[\{\boldsymbol{K}(\mathcal{E})\}_{\mathcal{E}\in\mathcal{M}}]$
RETURN $[\boldsymbol{f}, \boldsymbol{K}]$

where $\mathrm{algB}_{\mathcal{T}}$ must be used after algKE and before algKV.

5. EXAMPLES

The new FE algorithm (algALL) was implemented in a simulation environment which consisted of a desktop PC (Intel Core 2 Duo E6700 processor, 2.66 GHz, and 2GB RAM) and a SensAble PHANTOM Omni haptic device. For comparison, the TL-FE algorithm (in Appendix B) and the stiffness warping (SW) algorithm [Müller and Gross 2004][2] were also demonstrated. Besides, the NSN algorithm, which is obtained by skipping $\mathcal{P}$-loops from the new FE algorithm, was also demonstrated. Picinbono et al.'s [2003] method was not demonstrated because it does not include the computation of the global tangent stiffness matrix.

5.1 Implementation

An example implementation of the new FE algorithm is now described. We used four meshes listed in Table I and shown in Fig. 6(a), (d), (g), and (j), which were a cuboid ($\mathcal{M}_C$), a dinosaur ($\mathcal{M}_D$), a bunny ($\mathcal{M}_B$), and an Armadillo ($\mathcal{M}_A$), respectively. The material parameters for all the tetrahedra were chosen as $E = 0.05$ MPa and $\nu = 0.4$. The volumetric energy density function $\psi_{\mathrm{vol}}(\theta)$ used in $\mathrm{algB}_{\mathcal{T}}$ was chosen as $\psi_{\mathrm{vol}}(\theta) = K_{\mathrm{vol}} |\min(0, \theta)|^3 / 3$ where K_{vol} was chosen as $K_{\mathrm{vol}} = E/(4(1-2\nu))$ to produce the same amount of strain energy in the inverted state as the linear material. The cube

Table I. The numbers of the geometric primitives in the meshes.

	$N_{\mathcal{V}}$	$N_{\mathcal{E}}$	$N_{\mathcal{F}}$	$N_{\mathcal{T}}$
$\mathcal{M}_C$	441	1,984	2,744	1,200
$\mathcal{M}_D$	856	4,066	5,669	2,458
$\mathcal{M}_B$	1,171	6,153	9,047	4,064
$\mathcal{M}_A$	2,227	11,559	16,887	7,554

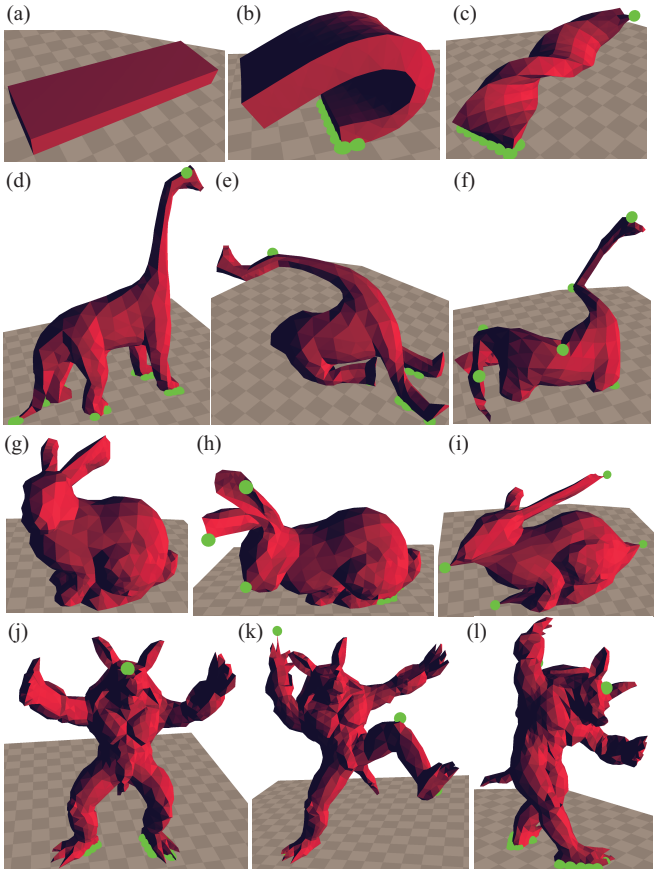


Fig. 6. Deformation of meshes realized with the new FE algorithm. (a)-(c) cuboid ($\mathcal{M}_C$); (d)-(f) dinosaur ($\mathcal{M}_D$); (g)-(i) bunny ($\mathcal{M}_B$); (j)-(l) Armadillo ($\mathcal{M}_A$). The green spheres are positions to each of which a vertex is elastically connected.

[2]The polar decomposition in the SW algorithm was performed employing the method of Higham and Schreiber [1990].

of θ is for holding $d\psi_{\rm vol}(\theta)/d\theta \approx 0$ when $\theta \approx 0$, not to influence the linear property of the material in the small-strain region.

The equation of motion of a whole mesh $\mathcal{M}$ can be described as follows:

$$\boldsymbol{M}\ddot{\boldsymbol{p}} + \boldsymbol{B}\dot{\boldsymbol{p}} + \boldsymbol{f} = \boldsymbol{h} \tag{41}$$

where $\boldsymbol{f} = \boldsymbol{\mathcal{F}}(\boldsymbol{p}) \in \mathbb{R}^{3N_\mathcal{V}}$, $\boldsymbol{M}$ and $\boldsymbol{B}$ are inertial and viscosity matrices of $\mathbb{R}^{3N_\mathcal{V}\times 3N_\mathcal{V}}$, respectively, and $\boldsymbol{h} \in \mathbb{R}^{3N_\mathcal{V}}$ is the vector composed of external force vectors acting on the vertices. For simplicity, $\boldsymbol{M}$ was set to be a diagonal matrix whose entries were chosen to approximate the distributed mass of density $\rho = 10^{-6}$ kg/mm^3. The conventional Rayleigh damping model $\boldsymbol{B} = \alpha\boldsymbol{M} + \beta\boldsymbol{K}$, where $\boldsymbol{K} = \boldsymbol{\mathcal{K}}(\boldsymbol{p})$, $\alpha = 0.5$ s^{-1}, and $\beta = 0.1$ s were used. External forces $\boldsymbol{h}$ were determined in a penalty-based manner, i.e., through virtual spring-damper elements each of which connects a vertex to the environment or to the haptic device. This means that the force $\boldsymbol{h}$ was computed through a function that can be described as $\boldsymbol{h} = \boldsymbol{\mathcal{H}}(\boldsymbol{p}, \dot{\boldsymbol{p}}, \boldsymbol{q})$ where $\boldsymbol{q}$ denotes the state (position and velocity) of the haptic device. Besides, the gravity of 9800 mm/s^2 was applied to all vertices. The stiffness and viscosity of the springs that connect vertices to the environment (the haptic device) were 10 N/mm and 0.1 Ns/mm (0.3 N/mm and 0 Ns/mm), respectively.

The differential equation (41) was integrated along time through a linearized implicit integration scheme, which is equivalent to the one described in [Baraff and Witkin 1998]. With T being the timestep size, the procedure of integration can be described as follows:

FOR $k \in \{1, 2, \cdots\}$
$\quad \boldsymbol{q}_{\rm device} := \text{GET\_DEVICE\_STATE}()$
$\quad \boldsymbol{h} := \boldsymbol{\mathcal{H}}\left(\boldsymbol{p}(k-1), \boldsymbol{v}(k-1), \boldsymbol{q}_{\rm device}\right)$
$\quad \boldsymbol{K}_H := \boldsymbol{\mathcal{H}}_p\left(\boldsymbol{p}(k-1), \boldsymbol{v}(k-1), \boldsymbol{q}_{\rm device}\right)$
$\quad \boldsymbol{B}_H := \boldsymbol{\mathcal{H}}_v\left(\boldsymbol{p}(k-1), \boldsymbol{v}(k-1), \boldsymbol{q}_{\rm device}\right)$
$\quad [\boldsymbol{f}, \boldsymbol{K}] := \text{algALL}\left[\boldsymbol{p}(k-1)\right]$
$\quad \boldsymbol{A} := \boldsymbol{K} + ((1/T + \alpha)\boldsymbol{M} - T\boldsymbol{K}_H - \boldsymbol{B}_H)/(T + \beta)$
$\quad \boldsymbol{b} := (\boldsymbol{h} - \boldsymbol{f} + (1/T)\boldsymbol{M}\boldsymbol{v}(k-1))/(T + \beta)$
$\quad \boldsymbol{v}(k) := \text{SOLVE\_BY\_PCGM}\ [\ \boldsymbol{A}\boldsymbol{v}(k) = \boldsymbol{b}\ ]$
$\quad \boldsymbol{p}(k) := \boldsymbol{p}(k-1) + T\boldsymbol{v}(k)$
END FOR.

Here, $\boldsymbol{\mathcal{H}}_p \triangleq \partial\boldsymbol{\mathcal{H}}/\partial\boldsymbol{p}$ and $\boldsymbol{\mathcal{H}}_v \triangleq \partial\boldsymbol{\mathcal{H}}/\partial\boldsymbol{v}$. The algebraic equation $\boldsymbol{A}\boldsymbol{v}(k) = \boldsymbol{b}$ was solved by the preconditioned conjugate gradient method (PCGM) with the simple diagonal preconditioner [Baraff and Witkin 1998]. The use of PCGM is based on the assumption that $\boldsymbol{A}$ is positive definite, which is not guaranteed as discussed in [Teran et al. 2005] but is mostly valid when the parameters are chosen appropriately. The iterations for the PCGM were performed for a fixed number of times, which was 20 for $\mathcal{M}_A$ and 80 for the others, as long as it was possible within the timestep size of $T = 30$ ms. The force from the device actuators was determined at every 1 ms through a virtual spring that connects the device to an interpolated position of the vertex to which the device is connected.

5.2 Results

The simulation was performed as shown in the photograph of Fig. 1. Fig. 6 shows deformed states of the meshes obtained by the new FE algorithm. Due to the property of the StVK material law, the meshes exhibited proper deformations even under large partial rotations as shown in (b), (c), (e), (h), (k), and (l) of Fig. 6. Extreme torsion [Fig. 6(c) and (f)] and elongation [Fig. 6(j)] were also successful due to the volume-preserving forces. The temporal behaviors of the meshes were visually plausible except with the largest mesh $\mathcal{M}_A$. The behavior of $\mathcal{M}_A$ was indeed realistic but was slow, as if it was moving in the water, probably because the number of the PCGM iterations (which was 20) was not sufficient to simulate this large mesh under the gravity and low viscous resistance. Increasing this number to 30, which was close to the limit within the timestep size $T = 30$ ms, slightly accelerated the motion, but it was still slower than the other meshes. The temporal behavior of the meshes strongly depends on the convergence speed of the iterative solver, which falls outside the scope of the paper.

The numbers of FLOPs required by the computation can be predicted as in Fig. 7 based on the discussion in section 3.6. In contrast, the time used for the computation in the experiment was as summarized in Fig. 8. For all meshes, the reduction of the FLOP count by the new method is about 70% from the TL-FE algorithm but the reduction of the time was only 28% to 53%. This is probably because of some overheads due to memory addressing. It must be noted again that the overall time of simulation includes the time for the iterative computation to solve the linear equation. The necessary number of the iterations depends on many factors such as the level of accuracy needed by the application and on the condition number of the matrix $\boldsymbol{A}$. Nevertheless, the reduced time for computing $\boldsymbol{f}$ and $\boldsymbol{K}$ contributes to the increased possible number of iterations, which leads to the increased accuracy in the dynamic response realizable within a given timestep size.

The TL-FE algorithm produced exactly the same results as the new FE algorithm because they are analytically equivalent to each other. Except them, the choice of the algorithms produced small differences in the deformation. Fig. 9 shows examples of such differences, in which the mesh $\mathcal{M}_C$ were twisted under the same constraint (indicated by the green spheres) and three different methods: the SW, the new FE, and the NSN algorithms. The difference mainly appears in the volumetric response resulted from the twist: the width in the twisted portion and the overall length (in the vertical direction of the figure) of the mesh. Fig. 9 shows that the mesh tends to preserve its volume more with the SW algorithm than with the new FE and the NSN algorithms. This can be attributed to the fact that the StVK and the NSN models become softer when they are compressed as illustrated in Fig. 3. Besides, the NSN model

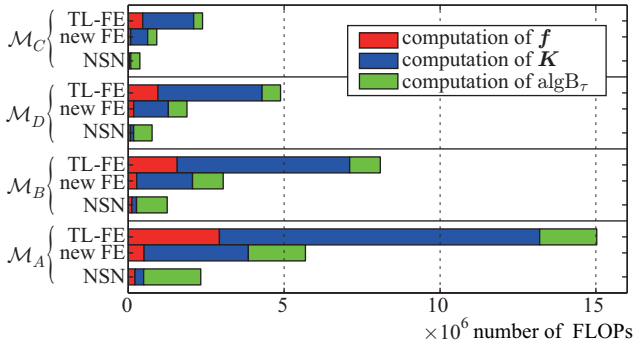


Fig. 7. Numbers of FLOPs to obtain $\boldsymbol{f}$ and $\boldsymbol{K}$ with different methods. The FLOP count for $\mathrm{algB}_{\mathcal{T}}$ is added to the numbers of each method because $\mathrm{algB}_{\mathcal{T}}$ is necessary for all methods to prevent element inversion. The numbers are obtained based on the discussions in section 3.6 and section 4. The numbers for SW are not shown because they are generally not constant due to the iterative computation in the polar decomposition.

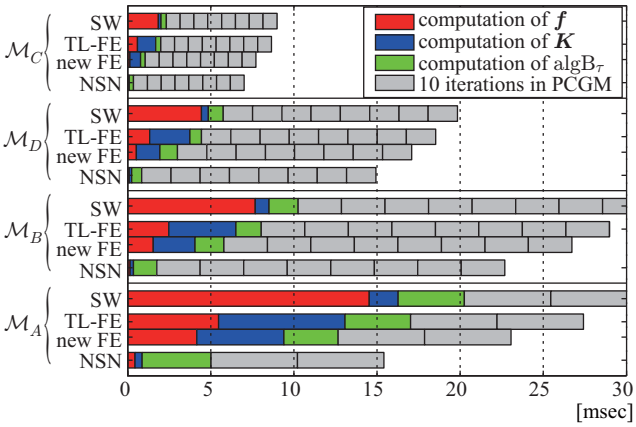


Fig. 8. Computational time spent for individual steps of the simulation in the experiment. The iteration count was 20 for $\mathcal{M}_A$ and 80 for others.

produced somewhat jaggy surface as compared to the other models, which might be because of its lack of representing continuum mechanics.

The conclusions drawn from these results are summarized as follows. The new FE algorithm produces the same result as but is faster than the TL-FE algorithm. The new FE algorithm is also faster than the SW algorithm although the SW algorithm is always stable as discussed in Müller and Gross's [2004] paper. The NSN model, on the other hand, is much faster. The new FE algorithm thus can be suited for some applications where a compromise between the accuracy/stability and the computational speed is necessary. Its better consistency with the well-established StVK material law, which is described by partial differential equations, might count as a strength from a theoretical point of view. The NSN model may be acceptable for some applications where physical royalty is not a primary concern.

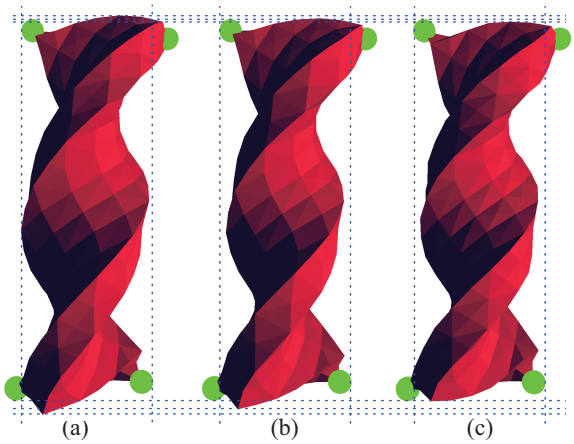


Fig. 9. Deformation of the mesh $\mathcal{M}_C$ produced by the (a) SW, (b) new FE, and (c) NSN algorithms. The TL-FE algorithm produced the same shape as the new FE algorithm. The green spheres are positions to each of which a vertex is elastically connected.

6. CONCLUSION AND FUTURE WORKS

This paper has presented a new formulation and an algorithm for FE simulation specific to tetrahedral meshes subject to the StVK material law. The algorithm computes the vertex forces and tangent stiffness matrices from given vertex positions. The number of FLOPs required for the computation is 62% to 73% smaller than a well-optimized algorithm derived from the conventional total Lagrangian formulation. A by-product of the presented formulation is a nonlinear spring network (NSN) model, which can be realized by skipping particular computational steps from the proposed FE algorithm. It is much faster than the FE models and produces visually plausible deformations although its physical royalty is debatable. The algorithms were demonstrated through an interactive application with haptic feedback combined with an implicit integration scheme. In our implementation, the presented FE algorithm was 28% faster than the conventional total Lagrangian algorithm and 42% faster than the stiffness warping method for computing vertex forces and tangent stiffness matrices in a mesh composed of 7554 tetrahedra.

Because the presented FE method is faster than more accurate or stable models but is still slower than less accurate models, it is suited for some applications where a compromise between the accuracy and the speed is necessary, such as interactive surgery simulators that require moderate accuracy. The presented FE method is probably useful also for applications that need runtime alteration

of the mesh structure, such as cutting and fracture, because it does not require as large amount of precomputation as some of the implicit schemes [Hirota and Kaneko 2001; Nakao et al. 2006]. It however needs some clarifications for an efficient redistribution of the material and geometric properties to edges and TSEPs when the connectivity of the tetrahedra is changed.

An important future topic of research is further optimization of the algorithms considering the efficiency of memory addressing. Parallelization of the algorithm will also be necessary for further acceleration using different types of computing devices such as multicore CPUs or GPUs. In addition, more optimized numerical techniques for solving algebraic equations will need to be combined with the presented formulation. Another interesting future topic is to include material nonlinearity. It is fortunate that the algorithm $\mathrm{algB}_{\mathcal{T}}$ for the volumetric force allows freedom in the choice of the energy function ψ_{vol}. Thus, it may be possible to design ψ_{vol} according to empirical data or well-established nonlinear material laws, making the model closer to real objects such as organs.

APPENDIX

A. DERIVATION OF TANGENT STIFFNESS MATRIX

This appendix section supplements section 3.5 by presenting the derivation of (31), which represents an off-diagonal block of the tangent stiffness matrix. From (29), when there exists an $\mathcal{E}$ connecting $\mathcal{V}_a$ and $\mathcal{V}_b$, $\partial \boldsymbol{f}(\mathcal{V}_a)/\partial \boldsymbol{p}(\mathcal{V}_b)$ can be rewritten as follows:

$$\begin{aligned}\boldsymbol{K}(\mathcal{E}) &= -g(\mathcal{E})\boldsymbol{I}_3 - 2\tilde{\boldsymbol{p}}(\mathcal{E})L(\mathcal{E})\tilde{\boldsymbol{p}}(\mathcal{E})^{\mathrm{T}} \\ &+ \sum_{\mathcal{P}\in\mathcal{M}} \mathrm{t}(\mathcal{V}_{\mathrm{e}}(\mathcal{E},0),\mathcal{E}_{\mathrm{p}}(\mathcal{P},0))\mathrm{t}(\mathcal{V}_{\mathrm{e}}(\mathcal{E},1),\mathcal{E}_{\mathrm{p}}(\mathcal{P},1))\boldsymbol{K}_p(\mathcal{P}) \\ &+ \sum_{\mathcal{P}\in\mathcal{M}} \mathrm{t}(\mathcal{V}_{\mathrm{e}}(\mathcal{E},1),\mathcal{E}_{\mathrm{p}}(\mathcal{P},0))\mathrm{t}(\mathcal{V}_{\mathrm{e}}(\mathcal{E},0),\mathcal{E}_{\mathrm{p}}(\mathcal{P},1))\boldsymbol{K}_p(\mathcal{P})^{\mathrm{T}} \quad (42)\end{aligned}$$

where $\boldsymbol{K}_p(\mathcal{P})$ is defined in (32). The definition of $\mathrm{t}(\mathcal{V},\mathcal{E})$ in (27) implies

$$\mathrm{t}(\mathcal{V},\mathcal{E}) = \sum_{\substack{i\in\{0,1\}\\ \mathcal{V}_{\mathrm{e}}(\mathcal{E},i)=\mathcal{V}}} (-1)^i. \quad (43)$$

Thus, the last two terms of (42) can be rewritten as follows:

$$\text{3rd term of (42)} = \sum_{\mathcal{P}\in\mathcal{M}} \sum_{i\in\{0,1\}} \sum_{\substack{j\in\{0,1\}\\ \mathrm{condA}(\mathcal{E},\mathcal{P},i,j)}} (-1)^{i+j}\boldsymbol{K}_p(\mathcal{P}) \quad (44a)$$

$$\text{4th term of (42)} = \sum_{\mathcal{P}\in\mathcal{M}} \sum_{i\in\{0,1\}} \sum_{\substack{j\in\{0,1\}\\ \mathrm{condB}(\mathcal{E},\mathcal{P},i,j)}} (-1)^{i+j}\boldsymbol{K}_p(\mathcal{P})^{\mathrm{T}} \quad (44b)$$

where the conditions under the summation operators are

$$\begin{aligned}\mathrm{condA}(\mathcal{E},\mathcal{P},i,j) \triangleq \{\ & \mathcal{V}_{\mathrm{e}}(\mathcal{E}_{\mathrm{p}}(\mathcal{P},0),i) = \mathcal{V}_{\mathrm{e}}(\mathcal{E},0) \\ & \wedge \mathcal{V}_{\mathrm{e}}(\mathcal{E}_{\mathrm{p}}(\mathcal{P},1),j) = \mathcal{V}_{\mathrm{e}}(\mathcal{E},1)\ \} \quad (45a)\end{aligned}$$

$$\begin{aligned}\mathrm{condB}(\mathcal{E},\mathcal{P},i,j) \triangleq \{\ & \mathcal{V}_{\mathrm{e}}(\mathcal{E}_{\mathrm{p}}(\mathcal{P},0),i) = \mathcal{V}_{\mathrm{e}}(\mathcal{E},1) \\ & \wedge \mathcal{V}_{\mathrm{e}}(\mathcal{E}_{\mathrm{p}}(\mathcal{P},1),j) = \mathcal{V}_{\mathrm{e}}(\mathcal{E},0)\ \}. \quad (45b)\end{aligned}$$

When $\mathcal{E}$ satisfies either of these conditions, $\mathcal{E}$ is the edge connecting the vertices $\mathcal{V}_{\mathrm{e}}(\mathcal{E}_{\mathrm{p}}(\mathcal{P},0),i)$ and $\mathcal{V}_{\mathrm{e}}(\mathcal{E}_{\mathrm{p}}(\mathcal{P},1),j)$. The orientation of $\mathcal{E}$ determines which of the two conditions is satisfied. By using the symbols $\mathcal{E}_{\mathrm{p2}}(\mathcal{P},i,j)$ and $\mathrm{s}(\mathcal{P},i,j)$, which are defined in (33) and (34) and illustrated in Fig. 4, (45) can be concisely rewritten as follows:

$$\mathrm{condA}(\mathcal{E},\mathcal{P},i,j) = \{\, \mathcal{E}_{\mathrm{p2}}(\mathcal{P},i,j)=\mathcal{E} \wedge \mathrm{s}(\mathcal{P},i,j)=1\} \quad (46a)$$

$$\mathrm{condB}(\mathcal{E},\mathcal{P},i,j) = \{\, \mathcal{E}_{\mathrm{p2}}(\mathcal{P},i,j)=\mathcal{E} \wedge \mathrm{s}(\mathcal{P},i,j)=-1\}. \quad (46b)$$

Substituting (44) by (46) yields (31).

B. TOTAL LAGRANGIAN FORMULATION

This appendix section supplements section 3.6 by showing the TL-FE algorithm, which is directly derived from the conventional total Lagrangian formulation. In the conventional derivation of FE formulations, both linear and nonlinear, geometric interpolation functions named "shape functions" are often used [Bathe 1996]. In tetrahedral meshes, the shape functions are linear functions described as $\phi_{\mathcal{T}\mathcal{V}}(\boldsymbol{x}) \triangleq (\boldsymbol{\alpha}_{\mathcal{T}}^{\mathrm{ini}}(\mathcal{V})^{\mathrm{T}}\boldsymbol{x} + a_{\mathcal{T}}(\mathcal{V})) \in \mathbb{R}$ where $\boldsymbol{\alpha}_{\mathcal{T}}^{\mathrm{ini}}(\mathcal{V}) \in \mathbb{R}^3$ and $a_{\mathcal{T}}(\mathcal{V}) \in \mathbb{R}$ are defined by the following:

$$\begin{bmatrix} \boldsymbol{\alpha}_{\mathcal{T}}^{\mathrm{ini}}(\mathcal{V}_0)^{\mathrm{T}} & a_{\mathcal{T}}^{\mathrm{ini}}(\mathcal{V}_0) \\ \boldsymbol{\alpha}_{\mathcal{T}}^{\mathrm{ini}}(\mathcal{V}_1)^{\mathrm{T}} & a_{\mathcal{T}}^{\mathrm{ini}}(\mathcal{V}_1) \\ \boldsymbol{\alpha}_{\mathcal{T}}^{\mathrm{ini}}(\mathcal{V}_2)^{\mathrm{T}} & a_{\mathcal{T}}^{\mathrm{ini}}(\mathcal{V}_2) \\ \boldsymbol{\alpha}_{\mathcal{T}}^{\mathrm{ini}}(\mathcal{V}_3)^{\mathrm{T}} & a_{\mathcal{T}}^{\mathrm{ini}}(\mathcal{V}_3) \end{bmatrix}^{\mathrm{T}} = \begin{bmatrix} \boldsymbol{p}^{\mathrm{ini}}(\mathcal{V}_0)^{\mathrm{T}} & 1 \\ \boldsymbol{p}^{\mathrm{ini}}(\mathcal{V}_1)^{\mathrm{T}} & 1 \\ \boldsymbol{p}^{\mathrm{ini}}(\mathcal{V}_2)^{\mathrm{T}} & 1 \\ \boldsymbol{p}^{\mathrm{ini}}(\mathcal{V}_3)^{\mathrm{T}} & 1 \end{bmatrix}^{-1}. \quad (47)$$

O'Brien and Hodgins [1999] use the vectors $\boldsymbol{\alpha}_{\mathcal{T}}^{\mathrm{ini}}(\mathcal{V}_i)$ to compute the force contribution to the vertices of the tetrahedron $\mathcal{T}$ as follows:[3]

$$\boldsymbol{f}_{\mathcal{T}}(\mathcal{V}_i) = \frac{C_{\mathcal{T}}^{\mathrm{ini}}}{6} \sum_{j=0}^{3} \boldsymbol{p}(\mathcal{V}_j) \left(\boldsymbol{\alpha}_{\mathcal{T}}^{\mathrm{ini}}(\mathcal{V}_j)^{\mathrm{T}}\boldsymbol{S}_{\mathcal{T}}\boldsymbol{\alpha}_{\mathcal{T}}^{\mathrm{ini}}(\mathcal{V}_i)\right) \in \mathbb{R}^3. \quad (48)$$

Here, $\boldsymbol{S}_{\mathcal{T}} \in \mathbb{R}^{3\times3}$ is a PK2 stress tensor, which can be obtained through the following procedure:

$$\boldsymbol{F}_{\mathcal{T}} := \textstyle\sum_{i=0}^{3} \boldsymbol{p}(\mathcal{V}_i)(\boldsymbol{\alpha}_{\mathcal{T}}^{\mathrm{ini}}(\mathcal{V}_i))^{\mathrm{T}} \in \mathbb{R}^{3\times3}. \quad (49a)$$

$$\boldsymbol{E}_{\mathcal{T}} := (\boldsymbol{F}_{\mathcal{T}}{}^{\mathrm{T}}\boldsymbol{F}_{\mathcal{T}} - \boldsymbol{I}_3)/2 \in \mathbb{R}^{3\times3} \quad (49b)$$

$$\boldsymbol{S}_{\mathcal{T}} := 2\mu\boldsymbol{E}_{\mathcal{T}} + (\lambda\mathrm{tr}(\boldsymbol{E}_{\mathcal{T}}))\boldsymbol{I}_3. \quad (49c)$$

The tangent stiffness matrix of a single tetrahedral element, on the other hand, can be expressed as follows:

$$\boldsymbol{K}_{\mathcal{T}} = \frac{C_{\mathcal{T}}^{\mathrm{ini}}}{6}\boldsymbol{B}_{\mathcal{T}}{}^{\mathrm{T}}\boldsymbol{D}_{\mathcal{T}}\boldsymbol{B}_{\mathcal{T}} + \begin{bmatrix} g_{\mathcal{T},00}\boldsymbol{I}_3 & \cdots & g_{\mathcal{T},03}\boldsymbol{I}_3 \\ \vdots & \ddots & \vdots \\ g_{\mathcal{T},03}\boldsymbol{I}_3 & \cdots & g_{\mathcal{T},33}\boldsymbol{I}_3 \end{bmatrix} \in \mathbb{R}^{12\times12}. \quad (50)$$

[3]For computing the forces, O'Brien and Hodgins [1999] use $C_{\mathcal{T}}^{\mathrm{ini}}/12$ instead of $C_{\mathcal{T}}^{\mathrm{ini}}/6$ in (48). This is because their definition of the strain tensor is double of the definition (49b).

The first and second terms of (50) are often referred to as the initial displacement stiffness matrix and the initial stress stiffness matrix, respectively. Here, $\boldsymbol{D}_{\mathcal{T}} \in \mathbb{R}^{6\times 6}$ is the one defined in (7) and $g_{\mathcal{T},ij} = (C_{\mathcal{T}}^{\mathrm{ini}}/6)\boldsymbol{\alpha}_{\mathcal{T}}^{\mathrm{ini}}(\mathcal{V}_j)^{\mathrm{T}}\boldsymbol{S}_{\mathcal{T}}\boldsymbol{\alpha}_{\mathcal{T}}^{\mathrm{ini}}(\mathcal{V}_i)$ $(i,j \in \{0,1,2,3\})$, which are used in computing $\boldsymbol{f}(\mathcal{V}_i)$ in (48). The matrix $\boldsymbol{B}_{\mathcal{T}} \in \mathbb{R}^{6\times 12}$ is the one that satisfies

$$\boldsymbol{B}_{\mathcal{T}} \triangleq \boldsymbol{B}_{\mathcal{T}}^{\mathrm{ini}}\mathrm{blockdiag}[\boldsymbol{F}_{\mathcal{T}}^{\mathrm{T}}, \boldsymbol{F}_{\mathcal{T}}^{\mathrm{T}}, \boldsymbol{F}_{\mathcal{T}}^{\mathrm{T}}, \boldsymbol{F}_{\mathcal{T}}^{\mathrm{T}}] \in \mathbb{R}^{6\times 12} \quad (51)$$

where $\boldsymbol{B}_{\mathcal{T}}^{\mathrm{ini}} \in \mathbb{R}^{6\times 12}$ is referred to as the strain-displacement matrix [Bathe 1996], which is a sparse matrix including the entries of $\boldsymbol{\alpha}_{\mathcal{T}}^{\mathrm{ini}}(\mathcal{V}_i)$.

For computing $\boldsymbol{f}$ from $\boldsymbol{p}$ through the procedure (49), one requires 214 multiplications and 173 additions, with taking the fact $\sum_{i=0}^{3}\boldsymbol{f}_{\mathcal{T}}(\mathcal{V}_i) = \boldsymbol{o}_3$ into account. For computing (50) and (51), one requires 720 multiplications and 574 additions, with taking the sparsity of $\boldsymbol{B}_{\mathcal{T}}^{\mathrm{ini}}$ and $\boldsymbol{D}_{\mathcal{T}}$ and the symmetry of $\boldsymbol{K}_{\mathcal{T}}$ into account and assuming that $g_{\mathcal{T},ij}$ are already obtained when $\boldsymbol{f}(\mathcal{V}_i)$ are computed. Moreover, one must perform 66 more additions to construct $\boldsymbol{K}_{\mathcal{T}}$s into the global tangent stiffness matrix. Thus, the computation of both $\boldsymbol{f}$ and $\boldsymbol{K}$ requires 934 multiplications and 813 additions.

C. DERIVATION OF VOLUMETRIC FORCES

This appendix section supplements section 4 by showing the derivation of the volumetric forces and the tangent stiffness matrices. By using the energy function $\Psi_{\mathcal{T}}^{\mathrm{vol}}$, a vertex force and an off-diagonal block of the tangent stiffness matrix can be respectively written as follows:

$$\boldsymbol{f}_{\mathcal{T}}^{\mathrm{vol}}(\mathcal{V}) = \frac{\partial W_{\mathcal{T}}^{\mathrm{vol}}}{\partial \boldsymbol{p}(\mathcal{V})^{\mathrm{T}}} = \Psi_{\mathcal{T}}^{\mathrm{vol}\prime}(\Delta C_{\mathcal{T}})\left(\frac{\partial C_{\mathcal{T}}}{\partial \boldsymbol{p}(\mathcal{V})^{\mathrm{T}}}\right) \quad (52)$$

$$\boldsymbol{K}_{\mathcal{T}}^{\mathrm{vol}}(\mathcal{E}) = \Psi_{\mathcal{T}}^{\mathrm{vol}\prime\prime}(\Delta C_{\mathcal{T}})\left(\frac{\partial C_{\mathcal{T}}}{\partial \boldsymbol{p}(\mathcal{V}_{\mathrm{e}}(\mathcal{E},0))^{\mathrm{T}}}\right)\left(\frac{\partial C_{\mathcal{T}}}{\partial \boldsymbol{p}(\mathcal{V}_{\mathrm{e}}(\mathcal{E},1))^{\mathrm{T}}}\right)^{\mathrm{T}} + \Psi_{\mathcal{T}}^{\mathrm{vol}\prime}(\Delta C_{\mathcal{T}})\frac{\partial^2 C_{\mathcal{T}}}{\partial \boldsymbol{p}(\mathcal{V}_{\mathrm{e}}(\mathcal{E},0))^{\mathrm{T}}\partial \boldsymbol{p}(\mathcal{V}_{\mathrm{e}}(\mathcal{E},1))} \quad (53)$$

where $\Delta C_{\mathcal{T}} = C_{\mathcal{T}} - C_{\mathcal{T}}^{\mathrm{ini}}$. Assuming that the orientations and the index numbers of the edges and vertices are chosen as in the tetrahedron $\mathcal{T}_0$ in Fig. 2, $C_{\mathcal{T}}$ is described as $C_{\mathcal{T}} = \tilde{\boldsymbol{p}}(\mathcal{E}_2)\cdot(\tilde{\boldsymbol{p}}(\mathcal{E}_5)\times\tilde{\boldsymbol{p}}(\mathcal{E}_4))$. Then, the partial derivatives in (52) and (53) are written as follows:

$$\frac{\partial C_{\mathcal{T}}}{\partial \boldsymbol{p}(\mathcal{V}_0)^{\mathrm{T}}} = \tilde{\boldsymbol{p}}(\mathcal{E}_5)\times\tilde{\boldsymbol{p}}(\mathcal{E}_4), \quad \frac{\partial^2 C_{\mathcal{T}}}{\partial \boldsymbol{p}(\mathcal{V}_0)^{\mathrm{T}}\partial \boldsymbol{p}(\mathcal{V}_1)} = [\tilde{\boldsymbol{p}}(\mathcal{E}_5)\times].$$

By using these relations, the contributions of the additional volume strain energy to $\boldsymbol{f}(\mathcal{V})$ and $\boldsymbol{K}(\mathcal{E})$ can be computed by using the algorithm $\mathrm{algB}_{\mathcal{T}}$ in section 4.

Acknowledgement

The authors are grateful with Prof. Xian Chen, Prof. Ken'ichi Morooka, Kyushu University, and Prof. Ryuji Shioya, Toyo University, for their helpful advices. The surface meshes of the bunny and the Armadillo were obtained from the website of Stanford Computer Graphics Laboratory[4]. The surface mesh of the dinosaur was obtained courtesy of an unknown creator from the AIM@SHAPE Shape Repository[5]. They were converted into tetrahedral meshes by using Adventure TetMesh provided by the ADVENTURE project[6].

REFERENCES

BARAFF, D. AND WITKIN, A. 1998. Large steps in cloth simulation. In *Proceedings of ACM SIGGRAPH 98*. 43–54.

BARBIČ, J. AND JAMES, D. L. 2005. Real-time subspace integration for St. Venant-Kirchhoff deformable models. *ACM Trans. Graph. 24,* 3, 982–990.

BATHE, K.-J. 1996. *Finite Element Procedures*. Prentice Hall.

BOLZ, J., FARMER, I., GRINSPUN, E., AND SCHRÖDER, P. 2003. Sparse matrix solvers on the GPU: Conjugate gradients and multigrid. *ACM Trans. Graph. 22,* 3, 917–924.

BRO-NIELSEN, M. AND COTIN, S. 1996. Real-time volumetric deformable models for surgery simulation using finite elements and condensation. In *Proceedings of Eurographics 96*. Vol. 15. 57–66.

BROUWER, I., MORA, V., AND LAROCHE, D. 2007. A viscoelastic soft tissue model for haptic surgical simulation. In *Proceedings of the Second Joint EuroHaptics Conference and Symposium on Haptic Interfaces for Virtual Environment and Teleoperator Systems*. 593–594.

CAPELL, S., GREEN, S., CURLESS, B., DUCHAMP, T., AND POPOVIĆ, Z. 2002a. Interactive skeleton-driven dynamic deformations. *ACM Trans. Graph. 21,* 3, 586–593.

CAPELL, S., GREEN, S., CURLESS, B., DUCHAMP, T., AND POPOVIĆ, Z. 2002b. A multiresolution framework for dynamic deformations. In *Proceedings of the 2002 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*. 41–47.

COUTINHO, A. L. G. A., MARTINS, M. A. D., ALVES, J. L. D., LANDAU, L., AND MORAES, A. 2001. Edge-based finite element techniques for non-linear solid mechanics problems. *International Journal for Numerical Methods in Engineering 50*, 2053–2068.

DEBUNNE, G., DESBRUN, M., CANI, M.-P., AND BARR, A. H. 2001. Dynamic real-time deformations using space and time adaptive sampling. In *Proceedings of ACM SIGGRAPH 2001*. 31–36.

DELINGETTE, H. 2008. Triangular springs for modeling nonlinear membranes. *IEEE Trans. Visualiz. Comput. Graph. 14,* 2, 329–341.

DOLL, S. AND SCHWEIZERHOF, K. 2000. On the development of volumetric strain energy functions. *Transactions of the ASME: Journal of Applied Mechanics 67,* 1, 17–21.

DUYSAK, A. AND ZHANG, J. J. 2004. Fast simulation of deformable objects. In *Proceedings of the Eighth International Conference on Information Visualisation*. 422–427.

GIBSON, S. F. F. AND MIRTICH, B. 1997. A survey of deformable modeling in computer graphics. Tech. rep. TR 1997-19, Mitsubishi Electric Research Laboratory.

[4]http://graphics.stanford.edu/data/3Dscanrep/

[5]http://shapes.aim-at-shape.net/

[6]http://adventure.q.t.u-tokyo.ac.jp/

GINGOLD, Y., SECORD, A., HAN, J. Y., GRINSPUN, E., AND ZORIN, D. 2004. A discrete model for inelastic deformation of thin shells. Tech. rep., Courant Institute of Mathematical Sciences, New York University. Available at http://www.mrl.nyu.edu/~ajsecord/fracture/.

GUMHOLD, S., GUTHE, S., AND STRASSER, W. 1999. Tetrahedral mesh compression with the cut-border machines. In *Proceedings of IEEE Visualization 1999*. 501–509.

HIGHAM, N. J. AND SCHREIBER, R. S. 1990. Fast polar decomposition of an arbitrary matrix. *SIAM Journal on Scientific and Statistical Computing 11,* 4, 648–655.

HIROTA, K. AND KANEKO, T. 2001. Haptic representation of elastic objects. *Presence 10,* 5, 525–536.

IRVING, G., SCHROEDER, C., AND FEDKIW, R. 2007. Volume conserving finite element simulations of deformable models. *ACM Trans. Graph. 26,* 3, 13:1–13:6.

IRVING, G., TERAN, J., AND FEDKIW, R. 2006. Tetrahedral and hexahedral invertible finite elements. *Graphical Models 68,* 2, 66–89.

JAMES, D. L. AND PAI, D. K. 1999. ArtDefo: accurate real time deformable objects. In *Proceedings of ACM SIGGRAPH 99*. 65–72.

LLOYD, B. A., SZÉKELY, G., AND HARDERS, M. 2007. Identification of spring parameters for deformable object simulation. *IEEE Trans. Visualiz. Comput. Graph. 13,* 5, 1081–1094.

LÖHNER, R. 1994. Edges, stars, superedges and chains. *Computer Methods in Applied Mechanics and Engineering 111*, 255–263.

MARTINS, M. A. D., COUTINHO, A. L. G. A., AND ALVES, J. L. D. 1997. Parallel iterative solution of finite element systems of equations employing edge-based data structures. In *Proceedings of the 8th SIAM Conference on Parallel Processing for Scientific Computing*.

MENDOZA, C. AND LAUGIER, C. 2003. Tissue cutting using finite elements and force feedback. In *Proceedings of the International Symposium in Surgery Simulation and Soft Tissue Modeling*. Lecture Notes in Computer Science, vol. 2673. Springer, 175–182.

MILLER, K., JOLDES, G., LANCE, D., AND WITTEK, A. 2007. Total Lagrangian explicit dynamics finite element algorithm for computing soft tissue deformation. *Communications in Numerical Methods in Engineering 23*, 121–134.

MÜLLER, M., DORSEY, J., MCMILLAN, L., AND JAGNOW, R. 2001. Real-time simulation of deformation and fracture of stiff materials. In *Proceedings of the Eurographics Workshop on Computer Animation and Simulation*. 113–124.

MÜLLER, M., DORSEY, J., MCMILLAN, L., JAGNOW, R., AND CUTLER, B. 2002. Stable real-time deformations. In *Proceedings of the 2002 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*. 49–54.

MÜLLER, M. AND GROSS, M. 2004. Interactive virtual materials. In *Proceedings of Graphics Interface*. 239–246.

NAKAO, M., KURODA, T., OYAMA, H., SAKAGUCHI, G., AND KOMEDA, M. 2006. Physics-based simulation of surgical fields for preoperative strategic planning. *Journal of Medical Systems 30,* 5, 371–380.

NEALEN, A., MÜLLER, M., KEISER, R., BOXERMAN, E., AND CARLSON, M. 2006. Physically based deformable models in computer graphics. *Computer Graphics Forum 25,* 4, 809–836.

O'BRIEN, J. F. AND HODGINS, J. K. 1999. Graphical modeling and animation of brittle fracture. In *Proceedings of ACM SIGGRAPH 99*. 137–146.

PICINBONO, G., DELINGETTE, H., AND AYACHE, N. 2003. Non-linear anisotropic elasticity for real-time surgery simulation. *Graphical Models 65,* 5, 305–321.

SIMO, J. C. AND TAYLOR, R. L. 1991. Quasi-incompressible finite elasticity in principal stretches: Continuum basis and numerical algorithms. *Computer Methods in Applied Mechanics and Engineering 85*, 273–310.

TERAN, J., BLEMKER, S., NG THOW HING, V., ANDFEDKIW, R. 2003. Finite volume methods for the simulation of skeletal muscle. In *Proceedings of the 2003 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*. 68–74.

TERAN, J., SIFAKIS, E., IRVING, G., AND FEDKIW, R. 2005. Robust quasistatic finite elements and flesh simulation. In *Proceedings of the 2005 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*. 181–190.

TERAN, J., SIFAKIS, E., NG THOW HING, V., LAU, C., AND FEDKIW, R. 2005. Creating and simulating skeletal muscle from the visible human data set. *IEEE Trans. Visualiz. Comput. Graph. 11,* 3, 317–328.

TERZOPOULOS, D., PLATT, J., BARR, A., AND FLEISCHER, K. 1987. Elastically deformable models. *Computer Graphics (Proceedings of ACM SIGGRAPH 87) 21,* 4, 205–214.

TESCHNER, M., HEIDELBERGER, B., MÜLLER, M., AND GROSS, M. 2004. A versatile and robust model for geometrically complex deformable solids. In *Proceedings of the Computer Graphics International 2004*. 312–319.

VAN GELDER, A. 1998. Approximate simulation of elastic membranes by triangulated spring meshes. *J. Graph. Tools 3,* 2, 21–42.

WEISS, J. A., MAKER, B. N., AND GOVINDJEE, S. 1996. Finite element implementation of incompressible, transversely isotropic hyperelasticity. *Computer Methods in Applied Mechanics and Engineering 135,* 1, 107–128.

ZHONG, H., WACHOWIAK, M. P., AND PETERS, T. M. 2005. A real time finite element based tissue simulation method incorporating nonlinear elastic behavior. *Computer Methods in Biomechanics and Biomedical Engineering 8,* 3, 177–189.

ZHUANG, Y. AND CANNY, J. 2000. Haptic interaction with global deformations. In *Proceedings of the 2000 IEEE International Conference on Robotics and Automation*. 2428–2433.

Received Month Year; revised Month Year; accepted Month Year