Visual Computer manuscript No.
(will be inserted by the editor)

A Time-Integration Method for Stable Simulation of Extremely Deformable Hyperelastic Objects

Ryo Kikuuwe

Received: date / Accepted: date

Abstract This paper presents a time integration method for realtime simulation of extremely deformable objects subject to geometrically nonlinear hyperelasticity. In the presented method, the equation of motion of the system is discretized by the backward Euler method, and linearly approximated through the first-order Taylor expansion. The approximate linear equation is solved with Quasi-Minimal Residual method (QMR), which is an iterative linear equation solver for non-symmetric or indefinite matrices. The solution is then corrected considering the nonlinear term that is omitted at the Taylor expansion. The method does not demand the constitutive law to guarantee the positive definiteness of the stiffness matrix. Experimental results show that the presented method realizes stable behavior of the simulated model under such deformation that the tetrahedral elements are almost flattened. It is also shown that QMR outperforms Biconjugate Gradient Stabilized method (BiCGStab) in this application.

Keywords QMR · finite elements · interactive simulation · hyperelasticity

1 Introduction

Realtime 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. In realtime simulation, the input from the user is obtained from input devices (e.g., mouse devices, keyboards, or haptic devices), the behavior of objects are simulated every timestep of a predetermined length (typically about 0.001 to 0.03 s), and the result is displayed through output devices (e.g., an LCD monitor and haptic devices), as exemplified in Fig. 1. Computational techniques for realtime simulation should assure the visual plausibility rather than quantitative accuracy or physical faithfulness. Such requirements are different from those for conventional applications of simulation software such as those for offline numerical analysis.

For the simulation of a continuum body represented by many vertices, the algorithm should typically consist of two stages per timestep. The first stage is for computing a constitutive law to provide the vertex forces according to the vertex positions. The second stage is the time integration, with which the vertex positions are updated according to the computed forces. Implicit integration schemes are often preferred because it realizes rather stable behavior of the simulated system. In such a scheme, a huge system of simultaneous equations must be solved at every timestep and, if the solver is of the iterative type, the computation must converge within the predetermined timestep size. In many studies in the field of computer graphics and interactive simulation, the equation is approximated by a linear equation of the form $\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}$, where $\boldsymbol{A}$ is a square matrix and $\boldsymbol{x}$ and $\boldsymbol{b}$ are vectors. The equation is usually numerically solved by the Conjugate Gradient Method (CGM) [1, 23, 28] because the method is known to be fast.

One concern in the use of CGM is that, if the matrix $\boldsymbol{A}$ is not a symmetric and positive definite matrix, it may result in the breakdown of the computation. The

This work was supported in part by Grant-in-Aid for Exploratory Research, No. 25540044, from Japan Society for the Promotion of Science.

R. Kikuuwe
Department of Mechanical Enginering
Kyushu University, Fukuoka 819-0395, Japan
Phone: +81-92-802-3174
Fax: +81-92-802-0001
E-mail: kikuuwe@ieee.org

positive definiteness of the matrix $\boldsymbol{A}$ can be violated especially when elements excessively deform. Moreover, in such cases, the validity of the linear approximation becomes questionable. In fact, in such an extreme deformation, the simulated object can be easily destabilized and the computation may diverge even with implicit integration schemes. Thus, possible options should be (i) to choose a computationally expensive constitutive law that maintains the positive definiteness of the matrix $\boldsymbol{A}$ and (ii) to choose a computationally expensive time integrator that allows the indefiniteness (non-positive-definiteness) of $\boldsymbol{A}$ and the nonlinearity of the simultaneous equation. Most of previous studies dealing with extreme deformation [17,22,39,42] explore the former approach with elaborately designed constitutive laws. As far as the author is aware, there have been no studies exploring the latter one.

The contribution of this paper is to provide an example of the latter approach to deal with extreme deformations. The presented technique employs the Saint Venant-Kirchhoff (StVK) constitutive law combined with a volumetric penalty [10,19,33], the Quasi-Minimum Residual method (QMR) [13,14] and a simple operation that corrects the error caused by the linear approximation. The StVK constitutive law is perhaps the simplest hyperelastic material law and its efficient computational method has already been presented [19]. The QMR is one of existing iterative solvers for linear equations with non-symmetric or indefinite matrices, which demands roughly twice the computational load of CGM. The presented technique realizes stable behavior of the simulated model under extreme deformation whereby the tetrahedral elements are flattened and stretched, as shown in Fig. 2.

The rest of this paper is organized as follows. Section 2 discusses a basic framework of the realtime simulation and related work. Section 3 describes the proposed method for time integration. Section 4 shows the results of implementation experiments. Section 5 provides the concluding remarks.

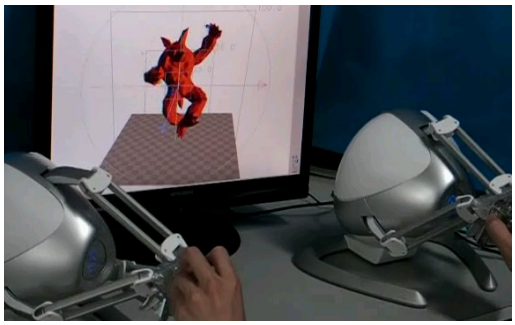

Fig. 1 Interactive simulation of an elastic object with haptic devices.

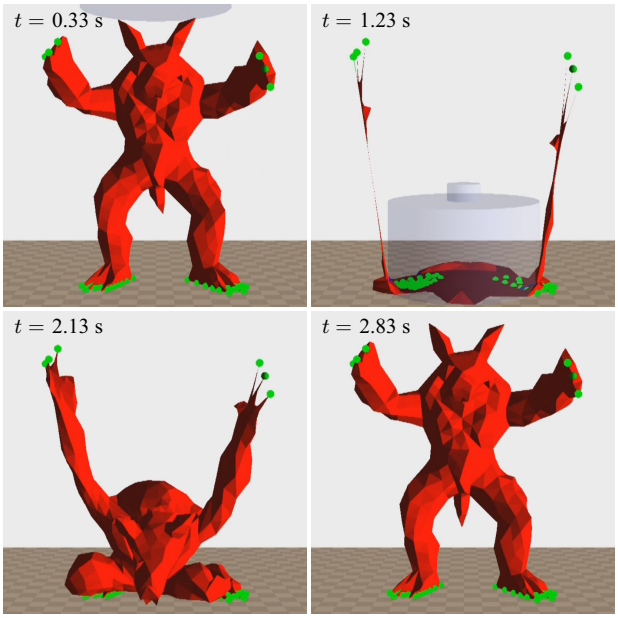

Fig. 2 Results provided by the proposed method. The simulated elastic object is elastically connected to the anchor points indicated by the green spheres. It is crushed with a cylindrical object from $t = 1.23$ s to $t = 1.9$ s. After the cylindrical object is removed, the elastic object resumes its original shape.

2 Preliminaries

2.1 Framework

Let us consider an elastic object represented by n vertices. Let $\boldsymbol{p} \in \mathbb{R}^{3n}$ and $\boldsymbol{v} \in \mathbb{R}^{3n}$ denote the vectors representing the positions and velocities, respectively, of the n vertices in the three dimensional space. Then, the equation of motion of the object can be described as follows:

$$\boldsymbol{M}\dot{\boldsymbol{v}} + \boldsymbol{B}\boldsymbol{v} + \boldsymbol{\mathcal{F}}(\boldsymbol{p}) = \boldsymbol{\mathcal{H}}(\boldsymbol{p}, \boldsymbol{v}) \tag{1}$$

$$\dot{\boldsymbol{p}} = \boldsymbol{v}. \tag{2}$$

Here, $\boldsymbol{\mathcal{F}} : \mathbb{R}^{3n} \to \mathbb{R}^{3n}$ is the constitutive law and $\boldsymbol{\mathcal{H}} : \mathbb{R}^{3n} \times \mathbb{R}^{3n} \to \mathbb{R}^{3n}$ is the external force as a function of $\boldsymbol{p}$ and $\boldsymbol{v}$. The matrices $\boldsymbol{M} \in \mathbb{R}^{3n\times 3n}$ and $\boldsymbol{B} \in \mathbb{R}^{3n\times 3n}$ are inertia and viscous matrices, respectively, which are symmetric and positive definite. Here, the mass matrix $\boldsymbol{M}$ can often be a diagonal matrix based on the so-called mass lumping technique.

For the convenience of further derivations, let us define the following functions:

$$\boldsymbol{\mathcal{K}}(\boldsymbol{p}) \triangleq \frac{\partial \boldsymbol{\mathcal{F}}(\boldsymbol{p})}{\partial \boldsymbol{p}} \in \mathbb{R}^{3n\times 3n} \tag{3}$$

$$\boldsymbol{\mathcal{K}}_H(\boldsymbol{p}, \boldsymbol{v}) \triangleq \frac{\partial \boldsymbol{\mathcal{H}}(\boldsymbol{p}, \boldsymbol{v})}{\partial \boldsymbol{p}} \in \mathbb{R}^{3n\times 3n} \tag{4}$$

$$\boldsymbol{\mathcal{B}}_H(\boldsymbol{p},\boldsymbol{v}) \triangleq \frac{\partial \boldsymbol{\mathcal{H}}(\boldsymbol{p},\boldsymbol{v})}{\partial \boldsymbol{v}} \in \mathbb{R}^{3n\times 3n}. \tag{5}$$

Based on the classical Rayleigh damping hypothesis, it is here assumed that the viscosity matrix $\boldsymbol{B}$ satisfies the following relation:

$$\boldsymbol{B} = \alpha \boldsymbol{M} + \beta \boldsymbol{\mathcal{K}}(\boldsymbol{p}) \tag{6}$$

where α and β are positive constants.

By using the backward Euler discretization, discrete-time approximations of (1) and (2) can be obtained as follows:

$$\boldsymbol{M}\frac{\boldsymbol{v}_{k+1}-\boldsymbol{v}_k}{T} + (\alpha\boldsymbol{M} + \beta\boldsymbol{\mathcal{K}}(\boldsymbol{p}_{k+1}))\boldsymbol{v}_{k+1} + \boldsymbol{\mathcal{F}}(\boldsymbol{p}_{k+1}) = \boldsymbol{\mathcal{H}}(\boldsymbol{p}_{k+1},\boldsymbol{v}_{k+1}) \tag{7}$$

$$\boldsymbol{p}_{k+1} = \boldsymbol{p}_k + T\boldsymbol{v}_{k+1} \tag{8}$$

where T is the timestep size. This is the set of simultaneous equations with respect to $\boldsymbol{p}_{k+1}$ and $\boldsymbol{v}_{k+1}$, and by eliminating $\boldsymbol{p}_{k+1}$ from (7) and (8), one can easily obtain an equation with an unknown vector $\boldsymbol{v}_{k+1}$. In order to deal with $\boldsymbol{v}_{k+1}$ in the arguments of the functions $\boldsymbol{\mathcal{F}}$, $\boldsymbol{\mathcal{K}}$, and $\boldsymbol{\mathcal{H}}$, let us consider the Taylor expansions of those functions as follows:

$$\boldsymbol{\mathcal{F}}(\boldsymbol{p}_k + T\boldsymbol{v}_{k+1}) \approx \boldsymbol{f}_k + T\boldsymbol{K}_k\boldsymbol{v}_{k+1} \tag{9}$$

$$\boldsymbol{\mathcal{K}}(\boldsymbol{p}_k + T\boldsymbol{v}_{k+1})\boldsymbol{v}_{k+1} \approx \boldsymbol{K}_k\boldsymbol{v}_{k+1} \tag{10}$$

$$\boldsymbol{\mathcal{H}}(\boldsymbol{p}_k + T\boldsymbol{v}_{k+1},\boldsymbol{v}_{k+1}) \approx \boldsymbol{h}_k + T\boldsymbol{K}_{H,k}\boldsymbol{v}_{k+1} + \boldsymbol{B}_{H,k}(\boldsymbol{v}_{k+1}-\boldsymbol{\phi}_k) \tag{11}$$

where

$$\boldsymbol{f}_k \triangleq \boldsymbol{\mathcal{F}}(\boldsymbol{p}_k) \in \mathbb{R}^{3n} \tag{12}$$

$$\boldsymbol{K}_k \triangleq \boldsymbol{\mathcal{K}}(\boldsymbol{p}_k) \in \mathbb{R}^{3n\times 3n} \tag{13}$$

$$\boldsymbol{h}_k \triangleq \boldsymbol{\mathcal{H}}(\boldsymbol{p}_k,\boldsymbol{\phi}_k) \in \mathbb{R}^{3n} \tag{14}$$

$$\boldsymbol{K}_{H,k} \triangleq \boldsymbol{\mathcal{K}}_H(\boldsymbol{p}_k,\boldsymbol{\phi}_k) \in \mathbb{R}^{3n\times 3n} \tag{15}$$

$$\boldsymbol{B}_{H,k} \triangleq \boldsymbol{\mathcal{B}}_H(\boldsymbol{p}_k,\boldsymbol{\phi}_k) \in \mathbb{R}^{3n\times 3n}. \tag{16}$$

Here, $\boldsymbol{\phi}_k \in \mathbb{R}^{3n}$ is an appropriate velocity vector that is used as the center of the Taylor expansion. In most cases, $\boldsymbol{\phi}_k$ can be the zero vector, though the optimal choice of the vector $\boldsymbol{\phi}_k$ is an open problem.

With the quantities defined in (12) to (16), (7) can be rewritten in the following form:

$$\boldsymbol{A}_k\boldsymbol{v}_{k+1} = \boldsymbol{b}_k \tag{17}$$

where

$$\boldsymbol{A}_k \triangleq \boldsymbol{K}_k + \frac{(1/T+\alpha)\boldsymbol{M} - T\boldsymbol{K}_{H,k} - \boldsymbol{B}_{H,k}}{T+\beta} \in \mathbb{R}^{3n\times 3n} \tag{18}$$

$$\boldsymbol{b}_k \triangleq \frac{\boldsymbol{h}_k - \boldsymbol{B}_{H,k}\boldsymbol{\phi}_k - \boldsymbol{f}_k + (\boldsymbol{M}/T)\boldsymbol{v}_k}{T+\beta} \in \mathbb{R}^{3n} \tag{19}$$

The vector $\boldsymbol{v}_{k+1}$ can be obtained by solving the linear equation (17). After that, the position $\boldsymbol{p}_{k+1}$ can be simply obtained by using (8). That is, the algorithm to update $\boldsymbol{p}$ and $\boldsymbol{v}$ can be obtained as follows:

Algorithm algU$(\boldsymbol{p}_k,\boldsymbol{v}_k)$

$\boldsymbol{f}_k \leftarrow \boldsymbol{\mathcal{F}}(\boldsymbol{p}_k)$
$\boldsymbol{K}_k \leftarrow \boldsymbol{\mathcal{K}}(\boldsymbol{p}_k)$
$\boldsymbol{h}_k \leftarrow \boldsymbol{\mathcal{H}}(\boldsymbol{p}_k,\boldsymbol{\phi}_k)$
$\boldsymbol{K}_{H,k} \leftarrow \boldsymbol{\mathcal{K}}_H(\boldsymbol{p}_k,\boldsymbol{\phi}_k)$
$\boldsymbol{B}_{H,k} \leftarrow \boldsymbol{\mathcal{B}}_H(\boldsymbol{p}_k,\boldsymbol{\phi}_k)$
$\boldsymbol{A}_k \leftarrow \boldsymbol{K}_k + \dfrac{(1/T+\alpha)\boldsymbol{M} - T\boldsymbol{K}_{H,k} - \boldsymbol{B}_{H,k}}{T+\beta}$
$\boldsymbol{b}_k \leftarrow \dfrac{\boldsymbol{h}_k - \boldsymbol{B}_{H,k}\boldsymbol{\phi}_k - \boldsymbol{f}_k + (\boldsymbol{M}/T)\boldsymbol{v}_k}{T+\beta}$
$\boldsymbol{v}_{k+1} \leftarrow \mathbf{slv}(\boldsymbol{A}_k,\boldsymbol{b}_k,\boldsymbol{v}_k)$
$\boldsymbol{p}_{k+1} \leftarrow \boldsymbol{p}_k + T\boldsymbol{v}_{k+1}$
Return $\{\boldsymbol{p}_{k+1},\boldsymbol{v}_{k+1}\}$.

Here, $\mathbf{slv}(\boldsymbol{A}_k,\boldsymbol{b}_k,\boldsymbol{v}_k)$ stands for the action of solving the equation $\boldsymbol{A}_k\boldsymbol{v} = \boldsymbol{b}_k$ with respect to $\boldsymbol{v}$ by using an iterative solver with the initial value of $\boldsymbol{v}$ being $\boldsymbol{v}_k$.

2.2 Related Work

There are many works that do not fall within the framework of the algorithm **algU**. Some works employ massive precomputation of the inverse of the stiffness matrix [15,25], the inverse of the matrix correspondent to $\boldsymbol{A}_k$ [36], and the deformation modes of the simulated object [3]. This paper will not consider such methods any further because they are not intended for realizing extreme deformation. Explicit integration methods have also been used [33], but this paper will not consider them either because they require a short timestep size for ensuring the stability. The projective dynamics approach [6,20,44], which is a recently developed approach, does not explicitly compute $\mathcal{F}$ or $\mathcal{K}$ but solves the problem (7)(8) (usually without the damping terms) by converting it into an optimization problem. This approach employs an auxiliary position vector that represents the perfect constraint, and thus, the constitutive law (more specifically, the elastic potential energy function) must be carefully designed. The present paper leaves this approach outside its scope by focusing on the use of conventional constitutive laws.

Some researchers employ algorithms similar to the algorithm **algU**, of which variations exist mainly in the constitutive law $\boldsymbol{\mathcal{F}}$ and its Jacobian $\boldsymbol{\mathcal{K}}$. The spring network (or mass-spring) models [1,11,20,26] and geometrically nonlinear finite element (FE) models [10,19] are two of the major classes of the constitutive laws. The spring-network models are usually faster and simpler in computation while FE models have better consistency

with the continuum mechanics. It has been recognized that, under large deformation, the stiffness matrix $\boldsymbol{K}_k$, and its resultant $\boldsymbol{A}_k$, can become nearly ill-conditioned or indefinite. Such a situation is not desirable because it prevents the use of CGM, which is fast but requires the positive definiteness, for solving the linear equation, i.e., the action of **slv** in **algU**. One simple remedy to this problem is to increase the inertia and the mass-proportional damping coefficient α. The efficacy, however, is limited especially under extreme deformation, and it can result in slow and damped response of the simulated object.

To deal with the possibility of the indefinite $\boldsymbol{A}_k$, it is reasonable to carefully design the constitutive law to maintain the positive definiteness of the stiffness matrix $\boldsymbol{K}_k$. A major approach to realize this is the corotational schemes, in which every element is regarded as a rotation of a particular deformation. The rotation is derived from the displacement gradient tensor, which is decomposed into a rotational component and another component by using, e.g., the polar decomposition [23,30], QR decomposition [27], and the singular value decomposition (SVD) [17,41]. The SVD is perhaps the most generic tool to find the rotations although it is computationally expensive. Recent improvements of corotational methods include those with potential energy [8], stiffness matrices considering the rotation gradients [2, 4,22], and an example-based scheme [39].

Complications are raised at element inversions, at which there is no unique way to extract a pure rotation from the 3×3 deformation gradient tensor. When the element is inverted, one of the three singular values of the deformation gradient tensor should be set negative, and its choice is not unique. In the pioneering work of Irving et al. [16,17], the singular value with the smallest magnitude is set negative and, for the computation of the vertex forces, the singular values are clipped above a small positive value. This strategy has been extended into "invertible" versions of existing constitutive laws such as an "invertible StVK" material law [38]. Schmedding and Teschner [37] questioned the use of the smallest-magnitude singular value through a careful observation of the force field generated by inverted elements, and proposed another strategy that minimizes the recovering motion of the tetrahedral element. Chao et al. [8] used the nearest rotation of the displacement gradient tensor defined in [24]. Civit-Flores and Susín [9] provided an improved method considering the vertex positions before and after the inversion. Stomakhin et al. [41] especially focused unnatural and unstable element inversions raised by extreme elongation of elements, and provided a new constitutive law depending on the singular values. Their constitutive laws are given as a form of a potential energy function, and its analytical derivatives, which are required for implementation, are composed of many terms.

If the indefiniteness of the matrix $\boldsymbol{K}_k$ is permitted, some simple constitutive laws without explicit consideration on element rotations are valid even at element inversions. The StVK material law [19,33] and spring-network model [1,11,20,21,26] are two of such constitutive laws, of which the potential energies are finite and smooth at any configurations, while Neo Hookean and Mooney Rivlin material laws have the potential energies that go to infinity as the material is compressed. One drawback of StVK and spring-network models is the fictitious equilibrium at the mirrored, inverted configuration, but it can be easily avoided by adding a volumetric penalty [19,21].

The indefinite $\boldsymbol{A}_k$, which can be caused by the indefinite $\boldsymbol{K}_k$, should be treated with linear equation solvers that are more computationally expensive than CGM. In the field of computer graphics and interactive simulation, such solvers are rarely used; the few examples include BiCGStab (Biconjugate gradient stabilized method) [43] for smoke animation [7], GMRES (Generalized minimal residual method) [35] for electrical discharge [5], and QMR for elastic objects [31]. As for the treatment of the nonlinearity, some researchers employ Newton-Raphson method to solve the nonlinear equation caused by the nonlinear constitutive laws [32, 45], but it is used at the precomputed stage, not in the realtime computation. As far as the author is aware, there have been no studies that consider the original nonlinear equation (7) in the realtime computation.

3 Proposed Method

This section presents an improvement of the algorithm **algU** for preventing unnatural behaviors of the simulated object. In the new algorithm, the approximate linear equation (17) is solved with QMR, which allows indefinite matrices, and then the obtained solution $\boldsymbol{v}_{k+1}$ is corrected by considering the error caused by the linear approximation. The central idea is to approximately solve the nonlinear equation (7) in such a way that large values could not be erroneously included in the solution $\boldsymbol{v}_{k+1}$.

3.1 Choice of Linear Equation Solver

There have been many iterative solvers proposed for linear equations involving indefinite matrices [34]. Biconjugate Gradient method (BiCG) [12] and Conjugate

Gradient Squared method (CGS) [40] can be considered as representative classical methods. One of their major drawbacks is their irregular convergent behavior; their plots of the residual norm versus the iteration count have many peaks during the convergence. BiCGStab [43] and QMR [14], which were proposed in the 1990s, are known to exhibit rather smooth convergence behaviors. It can be said that such a property is suited for realtime simulation because, in realtime simulation, the iterative computation is usually terminated within the limited timestep size, irrespective of convergence. In particular, QMR incorporates procedure to quasi-minimize the residual at every iteration while BiCGStab is mainly intended to stabilize the convergence. For this reason, this paper employs QMR and experimentally compares it with BiCGStab in Section 4.

3.2 Nonlinearity Correction

As will be shown in Section 4, the use of QMR is effective in preventing explosive behaviors of the simulated object. It is, however, not enough to prevent some unnatural behaviors, which are impulsively protruding motion of vertices that occur when the object is extremely deformed. Such erroneous behaviors are presumably caused by the inaccuracy of the linear approximation (9) of the highly nonlinear function $\boldsymbol{\mathcal{F}}(\cdot)$. One can infer that, during such behaviors, some entries of the solution $\boldsymbol{v}_{k+1}$ of the linear equation (17) are being excessively larger than the correct solution of the nonlinear equation (7).

Here we consider incorporating the error caused by the approximation (9), which is $\boldsymbol{\mathcal{R}}_k(\boldsymbol{v}_{k+1})$ where

$$\boldsymbol{\mathcal{R}}_k(\boldsymbol{v}) \stackrel{\Delta}{=} \boldsymbol{\mathcal{F}}(\boldsymbol{p}_k + T\boldsymbol{v}) - \boldsymbol{f}_k - T\boldsymbol{K}_k\boldsymbol{v}. \tag{20}$$

Eliminating $\boldsymbol{\mathcal{F}}(\boldsymbol{p}_k + T\boldsymbol{v}_{k+1})$ from (7) and (20) eventually yields

$$\boldsymbol{A}_k\boldsymbol{v}_{k+1} + \boldsymbol{\mathcal{E}}_k(\boldsymbol{v}_{k+1}) = \boldsymbol{b}_k \tag{21}$$

where

$$\boldsymbol{\mathcal{E}}_k(\boldsymbol{v}) \stackrel{\Delta}{=} \frac{\boldsymbol{\mathcal{R}}_k(\boldsymbol{v})}{T+\beta} = \frac{\boldsymbol{\mathcal{F}}(\boldsymbol{p}_k + T\boldsymbol{v}) - \boldsymbol{f}_k - T\boldsymbol{K}_k\boldsymbol{v}}{T+\beta}. \tag{22}$$

It would be possible to solve the nonlinear equation (21) with respect to $\boldsymbol{v}_{k+1}$ by using Newton-Raphson, quasi-Newton or steepest-descent methods [29, Chap.3]. It is however computationally expensive because the nonlinear equation must be solved with some iterations at every timestep, and, at each iteration, a $3n$-dimensional linear equation must be solved.

Instead of the problem (21), we consider setting a simplified problem. In the upcoming derivation, $\{\boldsymbol{X}\}_i$ denotes the ith entry of the vector $\boldsymbol{X} \in \mathbb{R}^{3n}$ and $[\boldsymbol{X}]_i$ denotes the ith 3-dimensional subvector of $\boldsymbol{X} \in \mathbb{R}^{3n}$ or the ith 3×3 diagonal block of $\boldsymbol{X} \in \mathbb{R}^{3n\times 3n}$. Recall that an estimate $\boldsymbol{A}_k^{-1}\boldsymbol{b}_k$ of the solution $\boldsymbol{v}_{k+1}$ can be obtained through a linear equation solver, and that the aim here is only to prevent large entries being included in the left-hand side of (21), not to accurately solve (21). Thus, we here consider obtaining $\boldsymbol{v}_{k+1}$ that satisfies

$$\left\| [\boldsymbol{A}_k\boldsymbol{v}_{k+1} + \boldsymbol{\mathcal{E}}_k(\boldsymbol{v}_{k+1})]_j \right\| \le \sqrt{\|[\boldsymbol{b}_k]_j\|^2 + Z_{k,j}^2}, \quad \forall j \in \{1, \cdots, n\} \tag{23}$$

$$\boldsymbol{v}_{k+1} = \boldsymbol{\Lambda}_k\boldsymbol{A}_k^{-1}\boldsymbol{b}_k \tag{24}$$

where $Z_{k,j} > 0$ ($j \in \{1, \cdots, n\}$) are appropriately chosen small positive scalars,

$$\boldsymbol{\Lambda}_k = \mathrm{blockdiag}\,[\lambda_{k,1}\boldsymbol{I}_3, \cdots, \lambda_{k,n}\boldsymbol{I}_3] \in \mathbb{R}^{3n\times 3n} \tag{25}$$

and $\lambda_{k,j}$ ($j \in \{1, \cdots, n\}$) are positive scalars smaller than 1. That is, we consider obtaining $\boldsymbol{v}_{k+1}$ through a subvector-wise scaling of $\boldsymbol{A}_k^{-1}\boldsymbol{b}_k$ so as to satisfy the condition (23).

Note that the condition (23) is a relaxation of the original condition (21), and if $\boldsymbol{\mathcal{E}}_k(\boldsymbol{v}_{k+1})$ is small enough, the relaxed problem provides the result $\boldsymbol{v}_{k+1} \approx \boldsymbol{A}_k^{-1}\boldsymbol{b}_k$. The positive scalars $Z_{k,j}$ in (23) have the same physical dimensions as that of $\boldsymbol{A}_k\boldsymbol{v}_{k+1}$, and they are used to prevent the velocity $\boldsymbol{v}_{k+1}$ from being excessively small when $\|[\boldsymbol{b}_k]_j\|$ are very small. Let $V > 0$ be an appropriately chosen velocity value that is permitted irrespective of the satisfaction of (21). Then,

$$Z_{k,j} \stackrel{\Delta}{=} \mathrm{tr}\,([\boldsymbol{A}_k]_j)\,V \tag{26}$$

can be considered as a possible choice for $Z_{k,j}$.

The relaxed problem (23)(24) still demands the computation of the nonlinear function $\boldsymbol{\mathcal{F}}(\cdot)$ for any iterative search of $\lambda_{k,i}$ values. To avoid this, here an assumption is introduced:

Assumption 1 *With any $\boldsymbol{v} \in \mathbb{R}^{3n}$, the matrix $\boldsymbol{A}_k$ and the function $\boldsymbol{\mathcal{E}}_k(\cdot)$ satisfy the following relations:*

$$\boldsymbol{A}_k\boldsymbol{\Lambda}\boldsymbol{v} \approx \boldsymbol{\Lambda}\boldsymbol{A}_k\boldsymbol{v} \tag{27}$$

$$\boldsymbol{\mathcal{E}}_k(\boldsymbol{\Lambda}\boldsymbol{v}) \approx \boldsymbol{\Lambda}^2\boldsymbol{\mathcal{E}}_k(\boldsymbol{v}) \tag{28}$$

where

$$\boldsymbol{\Lambda} = \mathrm{blockdiag}\,[\lambda_1\boldsymbol{I}_3, \cdots, \lambda_n\boldsymbol{I}_3] \in \mathbb{R}^{3n\times 3n} \tag{29}$$

with $0 < \lambda_j \le 1$ for all $j \in \{1, \cdots, n\}$.

The squared coefficient $\boldsymbol{\Lambda}^2$ in (28) can be justified by noticing

$$\boldsymbol{\mathcal{E}}_k(\boldsymbol{v}) \approx \frac{T^2}{2(T+\beta)} \begin{bmatrix} \boldsymbol{v}^T \left(\dfrac{\partial^2\{\boldsymbol{\mathcal{F}}(\boldsymbol{p})\}_1}{\partial\boldsymbol{p}\partial\boldsymbol{p}}\Big|_{\boldsymbol{p}=\boldsymbol{p}_k} \right)\boldsymbol{v} \\ \vdots \\ \boldsymbol{v}^T \left(\dfrac{\partial^2\{\boldsymbol{\mathcal{F}}(\boldsymbol{p})\}_{3n}}{\partial\boldsymbol{p}\partial\boldsymbol{p}}\Big|_{\boldsymbol{p}=\boldsymbol{p}_k} \right)\boldsymbol{v} \end{bmatrix}, \tag{30}$$

which is derived from the fact that $\boldsymbol{\mathcal{R}}_k(\boldsymbol{v})$ of (20) can be seen as a second-order term of the Taylor expansion of $\boldsymbol{\mathcal{F}}(\boldsymbol{v})$. This assumption can be satisfied when $\boldsymbol{A}_k$ and $\partial^2\{\boldsymbol{\mathcal{F}}(\boldsymbol{p})\}_i/\partial\boldsymbol{p}\partial\boldsymbol{p}$ ($i \in \{1, \cdots, 3n\}$) are sufficiently diagonally dominant.

By using Assumption 1, one can rewrite the left-hand side of (23) as follows:

$$\begin{aligned}\left\|[\boldsymbol{A}_k\boldsymbol{v}_{k+1} + \boldsymbol{\mathcal{E}}_k(\boldsymbol{v}_{k+1})]_j\right\| &\approx \left\|\left[\boldsymbol{\Lambda}_k\boldsymbol{b}_k + \boldsymbol{\Lambda}_k^2\boldsymbol{\mathcal{E}}_k(\boldsymbol{A}_k^{-1}\boldsymbol{b}_k)\right]_j\right\| \\ &= \lambda_{k,j}\left\|[\boldsymbol{b}_k]_j + \lambda_{k,j}\left[\boldsymbol{\mathcal{E}}_k(\boldsymbol{A}_k^{-1}\boldsymbol{b}_k)\right]_j\right\|. \quad (31)\end{aligned}$$

Thus, (23) can be approximated as follows:

$$\lambda_{k,j}\left\|[\boldsymbol{b}_k]_j + \lambda_{k,j}\left[\boldsymbol{\mathcal{E}}_k(\boldsymbol{A}_k^{-1}\boldsymbol{b}_k)\right]_j\right\| \le \sqrt{\|[\boldsymbol{b}_k]_j\|^2 + Z_{k,j}^2}, \quad \forall j \in \{1, \cdots, n\}. \quad (32)$$

Such a value of $\lambda_{k,j}$ that satisfies (32) can be obtained by reducing $\lambda_{k,j}$ from 1 until (32) is satisfied. In conclusions, $\boldsymbol{v}_{k+1}$ satisfying (23)(24) under Assumption 1 can be obtained by the following algorithm:

Algorithm algUNC$(\boldsymbol{p}_k, \boldsymbol{v}_k)$
  $\{\boldsymbol{p}_{k+1}, \boldsymbol{v}_{k+1}\} \leftarrow$ **algU**$(\boldsymbol{p}_k, \boldsymbol{v}_k)$
  $\{\boldsymbol{f}_k, \boldsymbol{K}_k, \boldsymbol{A}_k, \boldsymbol{b}_k\} \leftarrow \langle$Variables used in **algU**$(\boldsymbol{p}_k, \boldsymbol{v}_k)\rangle$
  $\boldsymbol{e}_{k+1} \leftarrow \dfrac{\boldsymbol{\mathcal{F}}(\boldsymbol{p}_{k+1}) - \boldsymbol{f}_k - T\boldsymbol{K}_k\boldsymbol{v}_{k+1}}{T+\beta}$
  for $j \in \{1, \cdots, n\}$
    $\lambda_{k,j} \leftarrow 1$
    $X_{k,j} \leftarrow \|[\boldsymbol{b}_k]_j\|^2 + \mathrm{tr}\left([\boldsymbol{A}_k]_j\right)^2 V^2$
    while $\lambda_{k,j}^2\|[\boldsymbol{b}_k]_j + \lambda_{k,j}[\boldsymbol{e}_{k+1}]_j\|^2 > X_{k,j}$
      $\lambda_{k,j} \leftarrow 0.9\lambda_{k,j}$
    end while
    $[\boldsymbol{p}_{k+1}]_j \leftarrow [\boldsymbol{p}_{k+1}]_j - (1-\lambda_{k,j})T[\boldsymbol{v}_{k+1}]_j$
    $[\boldsymbol{v}_{k+1}]_j \leftarrow \lambda_{k,j}[\boldsymbol{v}_{k+1}]_j$
  end for
Return $\{\boldsymbol{p}_{k+1}, \boldsymbol{v}_{k+1}\}$.

It should be noted that the algorithm **algUNC** requires the computation of the function $\boldsymbol{\mathcal{F}}(\cdot)$ twice in one timestep. Usually, the computation of the constitutive laws $\boldsymbol{\mathcal{F}}(\cdot)$ of nonlinear FE models is more expensive than that of linear FE models [33]. However, the algorithm for $\boldsymbol{\mathcal{F}}(\cdot)$ of the StVK FE model presented by the author and his colleagues [19] demands 70 % smaller amount of computation than a conventional algorithm for the StVK model, and it can be even faster than the linear FE method. Thus, one can say that the algorithm of $\boldsymbol{\mathcal{F}}(\cdot)$ in [19] is well suited for the use in the algorithm **algUNC**.

It should be cautioned that the algorithm **algUNC** is based on the relaxation (23) of (21) and the Assumption 1. Leaving theoretical validations of such a relaxation and an assumption for future work, this paper focuses on experimental validation of the presented algorithm. Section 4.2 will report some numerical results for the validation of Assumption 1.

4 Evaluation

4.1 Implementation

The presented algorithm was validated using an experimental simulation environment consisting of a desktop PC (Intel Core i7-X980, hexa-core, 3.33 GHz) and two Novint Falcon haptic devices. The simulation software was developed with Microsoft Visual C++ 2010. Major parts of the program were parallelized into five threads by using OpenMP.

The experiment mainly employed an elastic object model consisting of a tetrahedral mesh of Stanford Armadillo ($n = 975$ vertices, 2968 tetrahedra, approximately 150 mm high). The algorithms for $\boldsymbol{\mathcal{F}}$ and $\boldsymbol{\mathcal{K}}$ were those presented in [19], which are based on the StVK constitutive law. Young's modulus and Poisson's ratio were set as $E = 1.0$ MPa and $\nu = 0.49$, respectively. In order to prevent fictitious inverted equilibrium caused by the StVK constitutive law, additional volumetric penalty forces [19, Section 4] were also applied to the vertices. These forces were derived from the strain energy density function $\psi_{\mathrm{vol}}(\theta) = 0.1E\theta^2/(6(1-2\nu))$, where θ is the bulk strain. The mass matrix $\boldsymbol{M}$ was designed to approximate the distributed mass of density $\rho = 10^{-6}$ kg/mm^3. The damping parameters were chosen as $\alpha = 0.001$ s^{-1} and $\beta = 0.001$ s. Timestep size was chosen as $T = 0.03$ s. The gravity of 9800 mm/s^2 was applied to all vertices.

In the simulation, the object made contact with rigid horizontal surfaces, which are the fixed "floor" and the mobile "plate" moved along a predefined trajectory. The contact forces between the elastic object and the rigid surfaces were modeled by spring-damper forces acting to vertices. For simplicity, the friction force was modeled as a viscous force of which the viscous 1coefficient is proportional to the normal displacement. That is, the external force acting to a vertex was defined as follows:

$$\boldsymbol{\mathcal{H}}_v\left(\begin{bmatrix}p_N\\ \boldsymbol{p}_T\end{bmatrix}, \begin{bmatrix}\dot{p}_N\\ \dot{\boldsymbol{p}}_T\end{bmatrix}\right) = \begin{cases}\begin{bmatrix}K_N p_N + C_N(\dot{p}_N - V_N)\\ C_T p_N(\dot{\boldsymbol{p}}_T - \boldsymbol{V}_T)\end{bmatrix} & \text{if } p_N > 0\\ \boldsymbol{o}_3 & \text{otherwise}\end{cases} \quad (33)$$

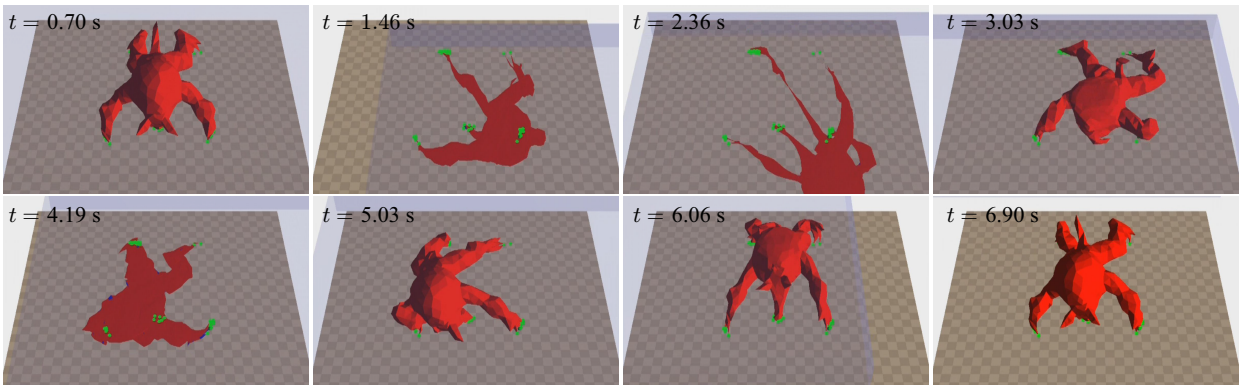

Fig. 3 Experiment A: Results with QMR+NC, $V = 100$ mm/s.

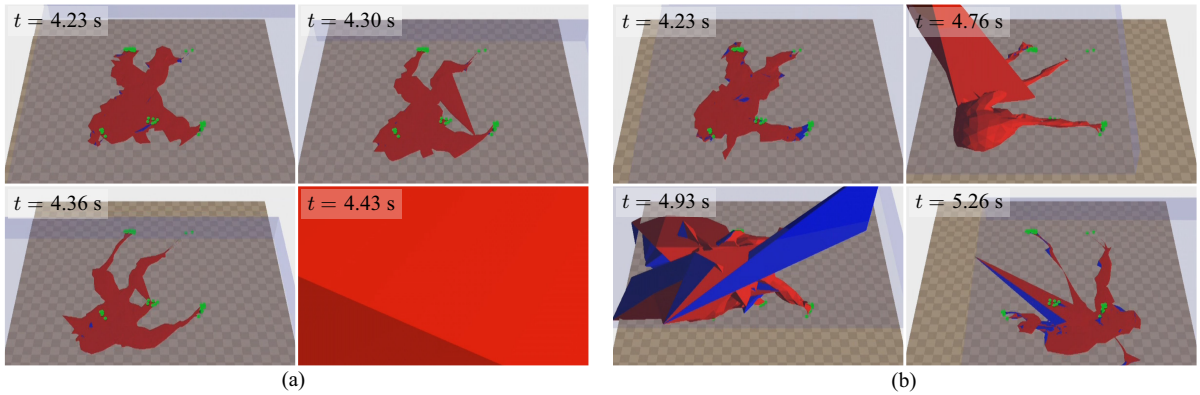

Fig. 4 Experiment A: (a) Explosive behavior of BiCGStab. (b) Erroneous impulsive behaviors of QMR. The blue faces are the inner sides of the faces of tetrahedral elements.

where $p_N \in \mathbb{R}$ and $\boldsymbol{p}_T \in \mathbb{R}^2$ are normal and tangential components of a vertex position, respectively. For notational simplicity, in (33), the origin of position vector is set on the surface, and p_N is positive into the surface. The quantities V_N and $\boldsymbol{V}_T$ are normal and tangential components, respectively, of the surface's velocity, which is zero as for the floor. The parameters were set as $K_N = 1000$ N/mm, $C_N = 10$ N·s/mm, and $C_T = 10$ N·s/mm^2. The Jacobian matrices $\boldsymbol{\mathcal{K}}_H$ and $\boldsymbol{\mathcal{B}}_H$ were defined accordingly. The velocity vector $\boldsymbol{\phi}_k$ in (9), which is the center of the Taylor expansion, was chosen based on V_N and $\boldsymbol{V}_T$.

Due to the parallelization of the computation, the simulation results were not exactly reproducible even when the number of iteration per timestep was fixed. This can be attributed to the non-associativity of floating-point additions [18] and to its effect magnified by the discontinuous mapping resulted from contact/non-contact transitions. In addition, fixing the number of iterations per timestep is not practically convenient because the possible number of iterations should be calibrated in advance and it varies by many factors such as the complexity of the mesh, contact conditions, and the background processes of the operating system. Thus, accepting a certain level of stochasticity of the results, an adaptive routine was employed to determine the number of iterations at every timestep so that as many iterations as possible are performed within the timestep size T.

4.2 Experiment A: Crush and Shear with Different Solvers

In the first set of experiments, the object (Stanford Armadillo) was initially placed in a posture shown in the left-top panel of Fig. 3. Some anchor points were fixed on the floor, as indicated by the green points in the figure, and they were connected to vertices of the model via virtual springs with the spring coefficients of 1000 N/mm and the damping coefficients of 10 N·s/mm. The mobile plate, which appears as a transparent object in Fig. 3, was moved along the predefined trajectory

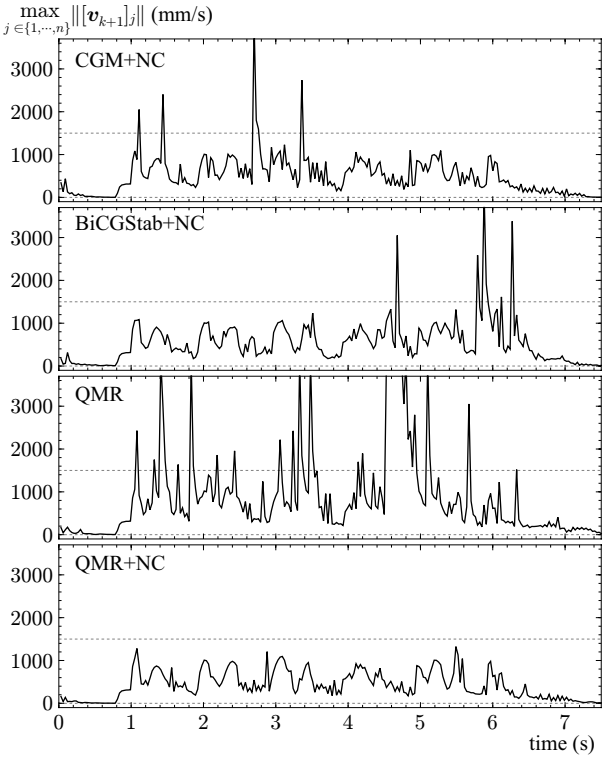

Fig. 5 Experiment A: The maximum vertex velocity obtained with the four schemes with which the simulation continued. The value 1500 mm/s, indicated by the horizontal dotted lines, is the threshold value used in Experiment B.

described by the following function of time:

$$\boldsymbol{p}_{\rm pl}(t) = \begin{cases} \begin{bmatrix} 0 \\ 20+300(1-t) \\ 0 \end{bmatrix} & \text{if } t \in [0\text{ s}, 1\text{ s}] \\ \begin{bmatrix} 100\sin(2\pi(t-1)/1.3) \\ \max(0, 20-100\sin(2\pi(t-1))) \\ -100\sin(2\pi(t-1)/0.7) \end{bmatrix} & \text{if } t \in [1\text{ s}, 6\text{ s}] \\ \begin{bmatrix} 100\sin(2\pi\cdot 5/1.3) \\ 20+300(t-6) \\ -100\sin(2\pi\cdot 5/0.7) \end{bmatrix} & \text{if } t \in [6\text{ s}, 8\text{ s}]. \end{cases} \quad (34)$$

That is, the plate was moved downward (i.e., in the negative y direction), moved in a Lissajous-like trajectory, and then moved upward. When the second entry of $\boldsymbol{p}_{\rm pl}$ is zero, the plate was in touch with the floor.

The following six schemes were compared in the experiments:

- CGM: **algU** with **slv** being CGM.
- CGM+NC: **algUNC** with **slv** being CGM.
- BiCGStab: **algU** with **slv** being BiCGStab.
- BiCGStab+NC: **algUNC** with **slv** being BiCGStab.
- QMR : **algU** with **slv** being QMR.
- QMR+NC : **algUNC** with **slv** being QMR.

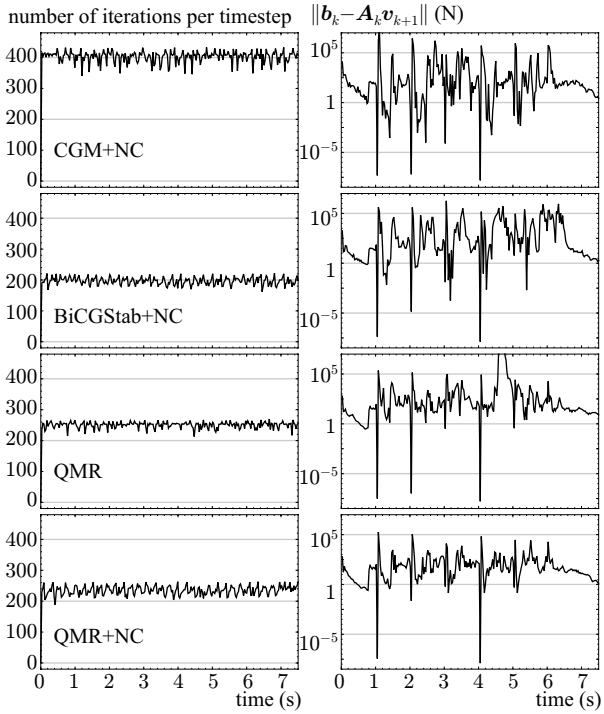

Fig. 6 Experiment A: The number of iterations and the residual error of the iterative solver **slv** within the timestep size $T = 0.03$ s at each timestep.

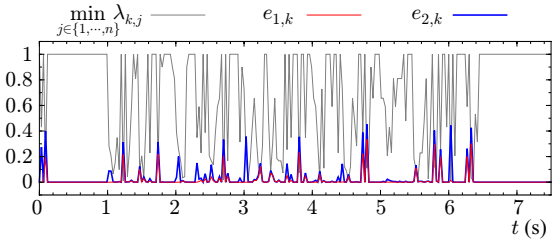

Fig. 7 The criteria $e_{1,k}$ and $e_{2,k}$ and the minimum of $\lambda_{k,j}$ at every timestep during the same scenario as Experiment A with QMR+NC, $V = 100$ mm/s.

Here, NC stands for the nonlinearity correction. In the algorithm **algUNC**, the V-value was chosen as 100 mm/s through some preliminary experiments.

The whole process of the simulation with QMR+NC is shown in Fig. 3. As can been seen here, the object is flattened and sheared by the friction force from the plate and the floor. It is shown that the simulation continues even when there is an extreme deformation. After the plate is removed, the object resumes its original shape.

In the cases of CGM and BiCGStab (without NC), the object showed explosive behaviors, as seen in Fig. 4(a), and the simulation did not continue. With the use of CGM+NC, BiCGStab+NC and QMR (without NC), the simulation did continue without explosions, but some

instantaneous impulsive behaviors were observed as seen in Fig. 4(b).

To evaluate the frequency of the erroneous impulsive behaviors, the maximum vertex velocity at each timestep was recorded. Fig. 5 shows the temporal profiles of the maximum vertex velocity in the mesh obtained from the four schemes, CGM+NC, BiCGStab+NC, QMR, and QMR+NC, with which the simulation continued without explosions. It is shown that, with any methods other than QMR+NC, there are some impulsive, high values in the maximum vertex speed. These results show that QMR is suited for this purpose and that NC is effective to enhance the stability.

Fig. 6 shows the number of iterations and the residual error achieved with each method. This figure shows that, although the numbers of iterations are smaller with QMR than with CG[1], the residual error of QMR is overall comparable with that of CG, and that its peak values are smaller than that of CG. In addition, from the comparison between QMR+NC and QMR, one can see that the nonlinearity correction contributes the reduction of peak values in the residual error but does not cost as much computational time as to result in much reduction of the number of iterations.

Numerical validation of Assumption 1 was also performed. A session of simulation was performed with QMR+NC and $V = 100$ mm/s in the same scenario as Fig. 3. In this session, the following criteria were computed at every timestep k:

$$e_{1,k} = \frac{\|\boldsymbol{A}_k\boldsymbol{\Lambda}_k\boldsymbol{v}_{k+1} - \boldsymbol{\Lambda}_k\boldsymbol{A}_k\boldsymbol{v}_{k+1}\|}{\|\boldsymbol{\Lambda}_k\boldsymbol{A}_k\boldsymbol{v}_{k+1}\|} \tag{35}$$

$$e_{2,k} = \frac{\|\boldsymbol{\mathcal{E}}_k(\boldsymbol{\Lambda}_k\boldsymbol{v}_{k+1}) - \boldsymbol{\Lambda}_k^2\boldsymbol{\mathcal{E}}_k(\boldsymbol{v}_{k+1})\|}{\|\boldsymbol{\Lambda}_k^2\boldsymbol{\mathcal{E}}_k(\boldsymbol{v}_{k+1})\|}, \tag{36}$$

which are for evaluating the accuracy of the approximations (27) and (28), respectively. The results are shown in Fig. 7[2]. This shows that, although $e_{1,k}$ and $e_{2,k}$ took the values around 0.5 at some peaks, they are mostly much smaller than 1 even when $\boldsymbol{\Lambda}_k$ has small components. These results imply that the approximations (27) and (28) are mostly valid.

The "rotational linear" method suggested by [17] combined with CGM was also tested in the same scenario. As can be seen in Fig. 8, it resulted in some unnatural behaviors especially when it is stretched, and it took a longer time (about $t = 13$ s) to eventually recover the original shape. The artifact of the corotational methods at the time of stretch has been pointed out by Stomakhin et al. [41], and such artifacts may be remedied by more sophisticated methods, e.g., [9, 37, 41]. This paper however does not consider such approaches any further because the focus of this paper is on the time integration scheme that can be combined with simple hyperelastic constitutive laws.

[1] This result is as expected because QMR performs two matrix-vector multiplications per iteration while CG performs only one.

[2] The criteria $e_{1,k}$ and $e_{2,k}$ were not computed in the experiments of Fig. 3, but was performed in another session because the computation of $e_{1,k}$ and $e_{2,k}$ resulted in non-negligible computational time. This computation resulted in roughly 10 % reduction in the number of QMR iterations.

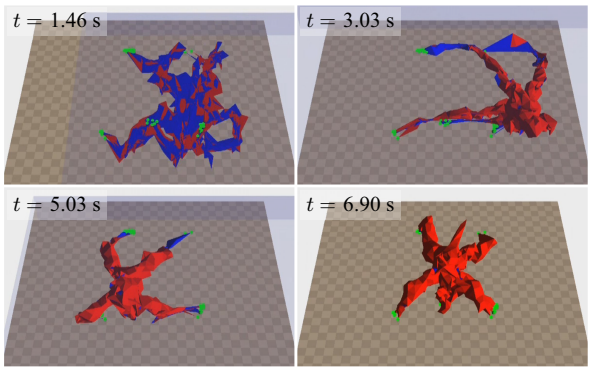

Fig. 8 Results of the "rotational linear" method in [17] combined with CGM. The blue faces are the inner sides of the faces of tetrahedral elements.

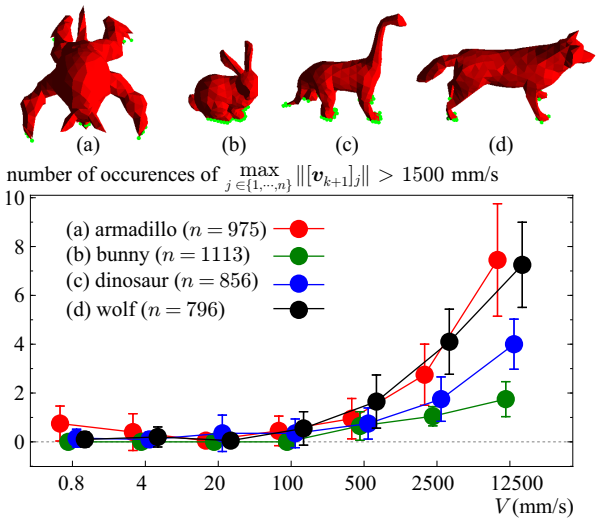

Fig. 9 Experiment B: The number of occurrence of $\max_j \|[\boldsymbol{v}_{k+1}]_j\| > 1500$ mm/s with QMR+NC with different V-values, during the same scenario as Experiment A.

4.3 Experiment B: Crush and Shear with Different V-Values

The effect of the V-value used in QMR+NC was evaluated by another set of experiments. The same procedure as Experiment A was followed with QMR+NC with six different V-values ranging from 0.8 mm/s to 12500 mm/s. Considering that the results are stochas-

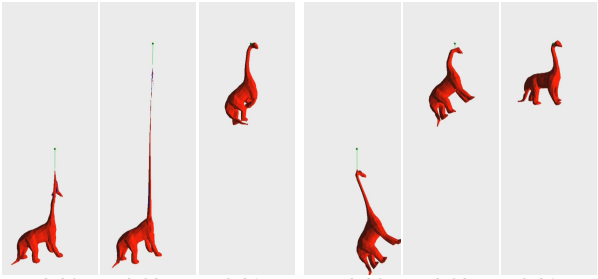

Fig. 10 Experiment C: QMR+NC with $V = 0.8$ mm/s and $V = 100$ mm/s.

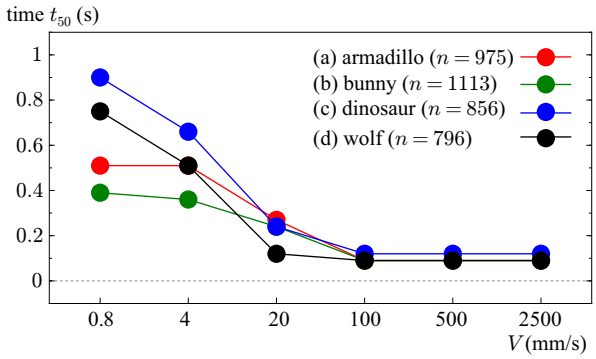

Fig. 11 Experiment C: Results with QMR+NC with different V-values and different meshes. The value t_{50} is the time required for the bottom most vertex of the object reaching 50 mm above its initial height.

tic, ten trials with each V-value were performed. It was performed with four different meshes indicated in the top of Fig. 9.

The program recorded the number of timesteps at which the maximum vertex speed is above 1500 mm/s during each trial. The averages and standard deviations are shown in the graph in Fig. 9. This shows that the probability of the high vertex velocity increases as V increases, and that V should be set appropriately low, being no higher than 100 mm/s.

4.4 Experiment C: Artifact of Excessively Low V-Values

Another set of experiments were performed to show a negative effect of excessively low V-values. The scenario of the experiments is shown in Fig. 10. Here, the object was initially hanged at an anchor point, which was connected to a vertex on the top part of the object through a virtual spring. At $t = 0$ s, the anchor point started to move up at a constant speed of 1000 mm/s, and at $t = 0.3$ s, it stopped. The procedure was also performed with four meshes indicated in the top of Fig. 9.

As seen in Fig. 10(a), when V is low (e.g., $V = 0.8$ mm/s), the bottom part remains stopped for a while after the beginning of the pulling-up, and it results in the stretching of the object. When V is high enough (e.g., $V = 100$ mm/s), such an artifact does not appear, and the object moves with its original shape being maintained, as seen in Fig. 10(b). The intensity of such an effect was evaluated with the time t_{50} required for the bottom most part of the object reaching 50 mm above the initial position. Fig. 11 shows the time t_{50} with different V values with the four meshes. These results suggest that V should be no lower than 100 mm/s to prevent the insensitivity to external forces.

Considering the results of Experiments II and III, it can be concluded that V should be low enough not to cause erroneous impulsive behaviors, and high enough not to cause the insensitivity to external forces. It has also be shown that there exist appropriate values in between, i.e., $V = 100$ mm/s in this case.

5 Conclusions

This paper has proposed a method for time integration of realtime simulation involving extreme hyperelastic deformation. The method is based on the linear approximation of the backward-Euler discretization of the equation of motion. The solution of the linear equation is corrected by an algorithm based on a rough approximation of the nonlinearity. The method has been validated through experiments. It has also been shown that, for solving the approximated linear equation, QMR is better than BiCGStab and CGM. The experiments have also investigated the effect of V, which is a parameter used in the algorithm of the nonlinearity correction.

Remaining problem is theoretical validation of the method. Optimal choice of the parameter V should also be clarified in the future study.

Acknowledgements The surface meshes of Stanford Armadillo and the "bunny" were obtained from the Web site of Stanford Computer Graphics Laboratory[3]. The surface mesh of the "dinosaur" was obtained courtesy of an unknown creator from the AIM@SHAPE Shape Repository[4]. They were converted into tetrahedral meshes by using Adventure TetMesh provided by the ADVENTURE project[5].

References

1. Baraff, D., Witkin, A.: Large steps in cloth simulation. In: Proceedings of ACM SIGGRAPH 98, pp. 43–54 (1998)

[3] http://graphics.stanford.edu/data/3Dscanrep/
[4] http://shapes.aim-at-shape.net/
[5] http://adventure.sys.t.u-tokyo.ac.jp/

2. Barbič, J.: Exact corotational linear FEM stiffness matrix. Tech. rep., University of Southern California (2012)
3. Barbič, J., James, D.L.: Real-time subspace integration for St. Venant-Kirchhoff deformable models. ACM Transactions on Graphics **24**(3), 982–990 (2005)
4. Barbič, J., Zhao, Y.: Real-time large-deformation substructuring. ACM Transactions on Graphics (2011)
5. Bickel, B., Wicke, M., Gross, M.: Adaptive simulation of electrical discharges. In: Proceedings of Vision, Modeling, and Visualization 2006, pp. 209–216 (2006)
6. Bouaziz, S., Martin, S., Li, T., Kavan, L., Pauly, M.: Projective dynamics: Fusing constraint projections for fast simulation. ACM Transactions on Graphics **33**(4), 154:1–154:11 (2014)
7. Brochu, T., Keeler, T., Bridson, R.: Linear-time smoke animation with vortex sheet meshes. In: Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation, pp. 87–95 (2012)
8. Chao, I., Pinkall, U., Sanan, P., Schröder, P.: A simple geometric model for elastic deformations. ACM Transactions on Graphics **29**(4), 38:1–38:6 (2010)
9. Civit-Flores, O., Susín, A.: Robust treatment of degenerate elements in interactive corotational FEM simulations. Computer Graphics Forum **33**(6), 298–309 (2014)
10. Delingette, H.: Biquadratic and quadratic springs for modeling St. Venant Kirchhoff materials. In: Proceedings of 4th International Symposium on Biomedical Simulation, *Lecture Notes in Computer Science*, vol. 5104, pp. 40–48. Springer (2008)
11. Desbrun, M., Schröder, P., Barr, A.H.: Interactive animation of structured deformable objects. In: Proceedings of the 1999 Conference on Graphics Interface, pp. 1–8 (1999)
12. Fletcher, R.: Conjugate gradient methods for indefinite systems. In: A. Dold, B. Eckmann (eds.) Numerical Analysis, *Lecture Notes in Mathematics*, vol. 506, pp. 73–89. Springer-Verlag (1976)
13. Freund, R.W., Nachtigal, N.M.: QMR: a quasi-minimal residual method for non-Hermitian linear system. Numerische Mathematik **60**(1), 315–339 (1991)
14. Freund, R.W., Nachtigal, N.M.: An implementation of the QMR method based on coupled two-term recurrences. SIAM Journal on Scientific Computing **15**(2), 313–337 (1994)
15. Hirota, K., Kaneko, T.: Haptic representation of elastic objects. Presence **10**(5), 525–536 (2001)
16. Irving, G., Teran, J., Fedkiw, R.: Invertible finite elements for robust simulation of large deformation. In: Proceedings of Eurographics/ACM SIGGRAPH Symposium on Computer Animation, pp. 131–140 (2004)
17. Irving, G., Teran, J., Fedkiw, R.: Tetrahedral and hexahedral invertible finite elements. Graphical Models **68**(2), 66–89 (2006)
18. Kapre, N., DeHon, A.: Optimistic parallelization of floating-point accumulation. Proceedings of 18th IEEE Symposium on Computer Arithmetic 2007 pp. 205–216 (2007)
19. Kikuuwe, R., Tabuchi, H., Yamamoto, M.: An edge-based computationally efficient formulation of Saint Venant-Kirchhoff tetrahedral finite elements. ACM Transactions on Graphics **28**(1), 8:1–8:13 (2009)
20. Liu, T., Bargteil, A.W., O'Brien, J.F., Kavan, L.: Fast simulation of mass-spring systems. ACM Transactions on Graphics **32**(6), 214:1–214:7 (2013)
21. Lloyd, B.A., Székely, G., Harders, M.: Identification of spring parameters for deformable object simulation. IEEE Transactions on Visualization and Computer Graphics **13**(5), 1081–1094 (2007)
22. McAdams, A., Zhu, Y., Selle, A., Empey, M., Tamstorf, R., Teran, J., Sifakis, E.: Efficient elasticity for character skinning with contact and collisions. ACM Transactions on Graphics **30**(4), Article 37 (2011)
23. Müller, M., Gross, M.: Interactive virtual materials. In: Proceedings of Graphics Interface, pp. 239–246 (2004)
24. Myronenko, A., Song, X.: On the closed-form solution of the rotation matrix arising in computer vision problems. arXiv:0904.161 (2009)
25. Nakao, M., Kuroda, T., Oyama, H., Sakaguchi, G., Komeda, M.: Physics-based simulation of surgical fields for preoperative strategic planning. Journal of Medical Systems **30**(5), 371–380 (2006)
26. Natsupakpong, S., Çavuşoğlu, M.C.: Determination of elasticity parameters in lumped element (mass-spring) models of deformable objects. Graphical Models **72**(6), 61–73 (2010)
27. Nesme, M., Payan, Y., Faure, F.: Efficient, physically plausible finite elements. In: Eurographics, Short Presentations, pp. 77–80 (2005)
28. Nienhuys, H.W., van der Stappen, F.A.: Combining finite element deformation with cutting for surgery simulations. In: Proceedings of Eurographics 2000, pp. 43–52 (2000)
29. Nocedal, J., Wright, S.J.: Numerical Optimization, 2 edn. Springer Series in Operations Research and Financial Engineering. Springer (2006)
30. Parker, E.G., O'Brien, J.F.: Real-time deformation and fracture in a game environment. In: Proceedings of 2009 ACM SIGGRAPH/Eurographics Symposium on Computer Animation, pp. 165–175 (2009)
31. Patterson, T., Mitchell, N., Sifakis, E.: Simulation of complex nonlinear elastic bodies using lattice deformers. ACM Transactions on Graphics **31**(6), 197:1–197:10 (2012)
32. Peterlík, I., Sedef, M., Basdogan, C., Matyska, L.: Real-time visio-haptic interaction with static soft tissue models having geometric and material nonlinearity. Computers & Graphics **34**(1), 43–54 (2010)
33. Picinbono, G., Delingette, H., Ayache, N.: Non-linear anisotropic elasticity for real-time surgery simulation. Graphical Models **65**(5), 305–321 (2003)
34. Saad, Y.: Iterative Methods for Sparse Linear Systems, 2nd edn. SIAM (2003)
35. Saad, Y., Schultz, M.H.: GMRES: A generalized minimal residual algorithm for solving nonsymmetric linear systems. SIAM Journal on Scientific Computing **7**(3), 856–869 (1986)
36. Saupin, G., Duriez, C., Cotin, S., Grisoni, L.: Efficient contact modeling using compliance warping. In: Proceedings of Computer Graphics International (2008)
37. Schmedding, R., Teschner, M.: Inversion handling for stable deformable modeling. Visual Computer **24**(7-9), 625–633 (2008)
38. Sin, F.S., Schroeder, D., Barbič, J.: Vega: Non-linear FEM deformable object simulator. Computer Graphics Forum **32**(1), 36–48 (2013)
39. Song, C., Zhang, H., Wang, X., Han, J., Wang, H.: Fast corotational simulation for example-driven deformation. Computers & Graphics **40**, 49–57 (2014)
40. Sonneveld, P.: CGS, a fast Lanczos-type solver for nonsymmetric linear systems. SIAM Journal on Scientific and Statistical Computing **10**(1), 36–52 (1989)
41. Stomakhin, A., Howes, R., Schroeder, C., Teran, J.M.: Energetically consistent invertible elasticity. In: Proceedings of ACM SIGGRAPH/Eurographics Symposium on Computer Animation, pp. 25–32 (2012)

42. Teran, J., Sifakis, E., Irving, G., Fedkiw, R.: Robust quasistatic finite elements and flesh simulation. In: Proceedings of the 2005 ACM SIGGRAPH/Eurographics Symposium on Computer Animation, pp. 181–190 (2005)
43. van der Vorst, H.A.: Bi-CGSTAB: A fast and smoothly converging variant of Bi-CG for the solution of nonsymmetric linear systems. SIAM Journal on Scientific Computing **13**(2), 631–644 (1992)
44. Wang, H.: A Chebyshev semi-iterative approach for accelerating projective and position-based dynamics. ACM Transactions on Graphics **34**(6), 246:1–246–9 (2015)
45. Zhong, H., Wachowiak, M.P., Peters, T.M.: A real time finite element based tissue simulation method incorporating nonlinear elastic behavior. Computer Methods in Biomechanics and Biomedical Engineering **8**(3), 177–189 (2005)