Contents lists available at ScienceDirect

Computer Aided Geometric Design

www.elsevier.com/locate/cagd

Coupling time-varying modal analysis and FEM for real-time cutting simulation of objects with multi-material sub-domains

Chen Yang[a], Shuai Li[a,*], Yu Lan[a], Lili Wang[a], Aimin Hao[a], Hong Qin[b]

[a] State Key Laboratory of Virtual Reality Technology and Systems, Beihang University, Beijing, China
[b] Stony Brook University, Stony Brook, NY, USA

ARTICLE INFO

Article history:
Available online 17 February 2016

Keywords:
Physical simulation
Multi-domains
Finite element methods
Modal analysis

ABSTRACT

Powerful global modal reduction techniques have received growing recognition towards significant performance gain in physical simulation, yet such numerical methods generally will fail when handling deformation of heterogeneous materials across multiple sub-domains involving cutting simulation. This is because the corresponding topological changes (due to cutting across multiple sub-domains) and/or drastic local deformations tend to invalidate the global subspace techniques. To ameliorate, this paper systematically advocates a novel deformation and arbitrary cutting simulation approach by adaptively integrating FEM-based fully-physical simulation and local deformation's modal reuse into a CUDA-enabled parallel computation framework. This paper's originality hinges upon the maximal reuse of the space–time-varying local modes from prior fully-physical simulations and the adaptively coupling of sub-domain behaviors, which give rise to great improvement of computational complexity while guaranteeing high-fidelity simulation effects. Other key advantages include, being independent of underlying physical models (e.g., either FEM or meshless methods), being flexibly accommodating sub-domains' heterogeneous material distributions, and being accurately responding to local user interactions. During the initialization stage, we partition the object into multiple sub-domains according to its material distributions and/or geometric structures, and respectively employ FEM for physics-based representation/simulation. During the dynamic stage, for each sub-domain, we leverage its local modal reduction in order to project complex deformations onto a low-dimensional subspace. We dynamically determine the sub-domain-specific switch between deformation reconstruction based on modal reuse and FEM-based physical simulation according to the physically-consistent error estimates, and couple all the sub-domains' physical behaviors together by imposing adjacent sub-domains' geometric-continuity constraints. To validate our method, we conduct extensive and quantitative evaluations over comprehensive and well-designed experiments, and all the experimental results have confirmed the advantages of our method in terms of efficiency, accuracy, and unconditional stableness in practical graphics applications.

* Corresponding author.
E-mail address: lishuai@buaa.edu.cn (S. Li).

http://dx.doi.org/10.1016/j.cagd.2016.02.014
0167-8396/© 2016 Elsevier B.V. All rights reserved.

1. Introduction and motivation

Real-time physically-realistic simulation has been one of the mainstreams that continues to attract a great deal of research efforts during the last two decades. So far, its widespread use has been enabling many downstream graphics applications such as computer games, virtual reality systems, computer animation, virtual surgery simulators, etc. To faithfully and efficiently simulate the object's physical behaviors of deformation and arbitrary cutting, many fundamental methodologies, ranging from accurate finite element methods (FEM) together with their GPU acceleration (Dick et al., 2011a) to various types of flexible mesh-free methods, have been well devised to accommodate the application-specific requirements.

In principle, except for the mathematically-rigorous and high-precision modeling of the underlying application-specific physics, most state-of-the-art methods are trying to pursue certain effective and flexible numerical approaches to make tradeoffs between physical accuracy and interactive efficiency, wherein global modal reduction based techniques (Barbič and James, 2005; Kim and James, 2009; Krysl et al., 2001) have gained growing momentum in recent years because of their powerful capabilities in significant simplification of computation complexity and still preserving dominant, global physical behaviors. Despite the great success of global modal reduction, when encountering complex heterogeneous objects, the engineering rationale of global modal reduction towards numerical gains tends to fall short in tackling new challenges, which requires material-sensitive physically-accurate modeling, adaptive handling of flexible topological changes, efficient local modal analysis and reuse, and sophisticated computational schemes in a much more intelligent way. In particular, the key technical challenges are highlighted as follows.

First of all, from the perspective of underlying physical modeling and simulation, most of FEM-based global modal reduction methods commonly discretize the physical domain into homogeneous elements. However, when handling complex homogeneous objects, a naively-transplanted way requires a large number of carefully-divided elements to accurately represent all the involved physical states, which gives rise to expensive computation expenses in modal reduction, and what is even worse is that, the above-documented extra efforts may not facilitate the corresponding global modal analysis towards physical realism due to massive elements from various sub-domains. Therefore, it naturally needs a divide-and-rule scheme (Barbič and Zhao, 2011; Kim and James, 2012) to independently model the involved heterogeneous physical domains in an approximated sense.

Second, from the perspective of the efficient utilization of modal analysis, even though global modal reduction methods have natural advantages in reducing computational expenses, its pre-computed modes may not have capabilities to represent any desirable motion outside the deformation subspace spanned by the global modes. Thus, global modal reduction is impossible to handle cutting and heterogeneous deformation because such behaviors naturally fall outside the space delineated by global modal reduction. Moreover, the topological changes due to arbitrary cutting require to frequently recompute the global modes, otherwise it will lead to un-neglected artifacts, which will inevitably diminish the apparent advantages associated with global modal reduction. Therefore, an effective way to conduct local modal analysis and accommodate local modal reuse is urgently needed.

Third, from the perspective of the numerical-computation efficiency and stableness towards practical applications, explicit integration based solvers (Fierz et al., 2012) indeed offer efficiency at the cost of being only conditionally stable, while implicit integration schemes (Allard et al., 2011) provide the assurance of unconditional stability and support large time steps, however, it usually needs to solve large systems of equations. Therefore, to achieve stable and interactive simulation, CUDA-based domain-parallel implicit solvers are urgently needed. Meanwhile, considering that user's interests in different sub-domains are distinct, in practical applications, it is required to simulate sub-domains of more interest with higher precision. And it is significant to effectively integrate spatially-varying FEM-based sub-domains with full-physical simulation capability and local deformation's modal reuse and to guarantee its correctness in physical and geometrical meanings.

To tackle the aforementioned challenges, our central idea is to extend the powerful global modal reduction method to handle efficient deformation and cutting simulation of dynamic models with heterogeneous material distribution. Specifically, this paper resorts to domain-parallelized physical simulation, subspace-independent physical modes reuse, and cross-domain adaptive behavior tight-coupling in a well-concerted CUDA-based parallel computational framework. The salient contributions of this paper include:

- We pioneer a space–time-varying local modal reuse method by introducing a divide-and-conquer scheme and coupling with the currently-available powerful global modal reduction, which enables spatially-varying deformation of different magnitude and arbitrary cutting simulation of heterogeneous object while guaranteeing high-fidelity simulation effects, and is independent of the underlying physical models (i.e., both FEM-based domain discretization and meshless method are supported).
- We propose to adaptively switch between integrating sub-domain-specific FEM-based simulation with full-physics capability and reconstructing deformation based on approximated, prior modal analysis according to the physics-geometry spatial-continuity error estimates. Such flexibility is dictated by sub-domain importance and sub-domain deformation amplitude, which can take full advantages of the accuracy of domain-specific physical simulation and the efficiency of local modal reduction methods.
- We design a CUDA-based parallel implicit integration framework, which supports material-aware and geometric-structure-aware sub-domain partitioning, dynamic evolution of sub-domain-specific deformation subspace and smooth switch between physical simulation and modal deformation for sub-domains. From the perspective of numerical anal-

ysis, the framework not only facilitates the local stiffness matrix pre-computation, dynamic topological updating and cutting surface reconstruction, but also guarantees the solvability of physically-correct local deformation modes, and satisfies the requirements of overall realtime efficiency and unconditional stableness simultaneously.

2. Related work

Closely relevant to the central theme of this paper, we now briefly review previous works in three aspects: FEM methods, coupling methods for physical simulation, and modal reduction methods.

FEM methods. FEM has proven to be a powerful approach to accurately model the physical and mechanical principles underlying deformation and cutting. For instance, Dick et al. (2011b) proposed a hexahedron based FEM method by conducting a multi-grid solving on GPU. Misztal et al. (2010) presented a novel approach to fluid simulation with an optimization-based, linear finite element method for solving the incompressible Euler equations. To enable large deformation, Müller et al. (2002) proposed a co-rotational formulation in the original space and employed a local coordinate frame to compute the linear force for each vertex. Similarly, Choi and Ko (2005) proposed a method based on node-wise rotation of the stiffness matrix and formulated in the modal space. Dick et al. (2011a) further introduced co-rotated strain formulation, which reduced nonlinear simulation to linear-time complexity. And Huang et al. (2006a) exploited domain decomposition method to handle large deformations using the linear elasticity model and a finite elements method. Based on the open-source framework (SOFA) (Allard et al., 2007), Allard et al. (2011) proposed a series of numerical methods to implicitly solve FEM-based deformation systems on GPU. Barbič and Zhao (2011) used co-rotational finite element method to accommodate large-scale deformations. And B. Wang et al. (2015) proposed to capture and model the deformations of soft objects by integrating FEM-based simulation into a physics-based tracking framework. As for FEM-based cutting simulation, it remains challenging in the aspects of dynamic topological update and dynamic cutting surface reconstruction (Jerabkova et al., 2010). Although Wicke et al. (2007) improved traditional subdivision based topological update method by introducing arbitrarily-convex finite elements, it is even more involved in numerical integration. Meanwhile, Sifakis et al. (2007) proposed a virtual node algorithm to avoid element splitting by duplicating the cut elements and redistributing the material properties. However, since each fragment is required to contain at least one FEM node, arbitrary cutting is strictly limited. Most recently, adaptive regular hexahedron approximation is used for cutting simulations (Dick et al., 2011b), however, it can only handle homogeneous objects.

Coupling methods for physical simulation. Although most of the current physical simulation methods are competitive, most of them are developed under different motivations. They are rather lonesome in some sense, and still require a trade-off among physical accuracy, material complexity, numerical stableness, and simulation efficiency. The domain decomposition methods (Kim and James, 2012; Barbič and Zhao, 2011; Toselli and Widlund, 2005) open up a new venue. These methods mainly transform the solution of the initial global problem to the solution of local sub-problems via splitting a global domain into several sub-domains. Rabczuk et al. (2006) gave a general overview of the coupling of mesh-free methods with finite elements. Besides, multi-domain subspace techniques provide a flexible solution to reduce the physical model. Kim and James (2012) proposed to couple the domains using penalty forces for character skinning. Xu et al. (2014) presented a semi-implicit integration method, which supports articulated structures. And Barbič and Zhao (2011) employed shape matching and mass lumping to handle the multi-domain subspace deformation coupling at the boundary interfaces. Yang et al. (2014) achieved precise cutting reconstruction by modeling the physical system with the material-aware local FEMs and modeling the domain-interlinking with meshless methods. Most recently, Yang et al. (2013b) proposed a boundary-aware modal construction method to characterize the deformation subspace of each domain, while Bosman et al. (2013) enabled cross-domain mechanical coupling and propagating of boundary conditions by representing the interface with 6-DOF mechanical points.

Modal reduction methods. Although fully physical-based methods are widely considered as reliable and accurate ways to simulate deformable objects, they are commonly computation-expensive. Modal reduction methods (also known as subspace methods) provide a novel perspective to reduce simulation complexity (Krysl et al., 2001), while guaranteeing dominant deformations. To solve slow convergence and numerical instability of iterative solver in nonlinear problem, Huang et al. (2006b) developed a subspace technique that projects deformation energy and constraints onto a coarse control mesh around the original mesh. Commonly, MR methods are mainly concentrated on linear deformation of finite element simulation (Pauly et al., 2005; Hildebrandt et al., 2011; Wang et al., 2014), however, they are not suitable for objects with non-linear relationship between strains and displacements. To alleviate, Barbič (2007) presented an automatic modal derivatives approach to select a set of quality low-dimensional basis vectors, and Xia et al. (2015) adopted a similar strategy to perform mesh interpolation in subspace spanned by these basis vectors. A MR method supporting arbitrary non-linear material models is proposed in An et al. (2008). Kim and James (2009) presented an online modal reduction method, wherein fully-physical computation is adaptively skipped and replaced with prior simulation subspace based deformation reconstruction. Most recently, von Tycowicz et al. (2013) proposed a new technique to approximate the reduced forces and construct object's deformation subspace. And Yang et al. (2013a) presented a novel physics-based volume segmentation by decomposing the tongue model into segments based on its deformation pattern with the computation of deformation subspaces and fitting the target deformation locally at each segment. Li et al. (2014a) proposed a space–time constraints involved method for elastic animation editing, which achieves better efficiency and scalability by conducting all the computations in a reduced rotation-strain space. Hahn et al. (2014) presented a new approach for clothing simulation in low-dimension linear

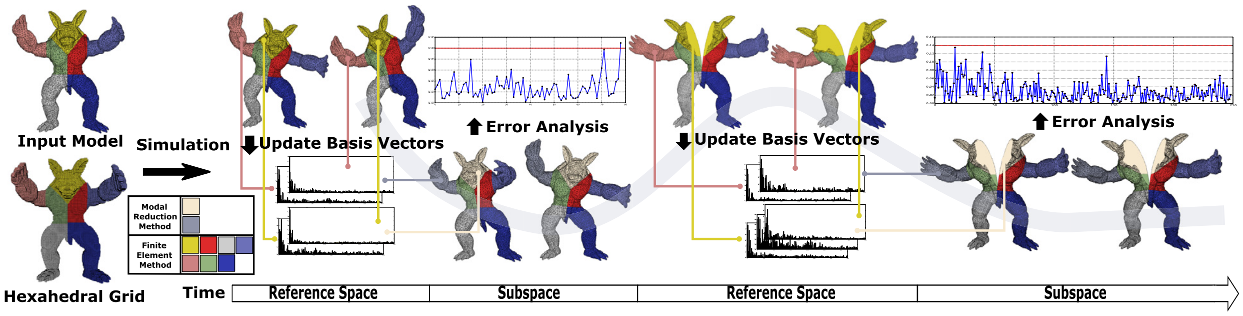

Fig. 1. The functional framework of our novel method.

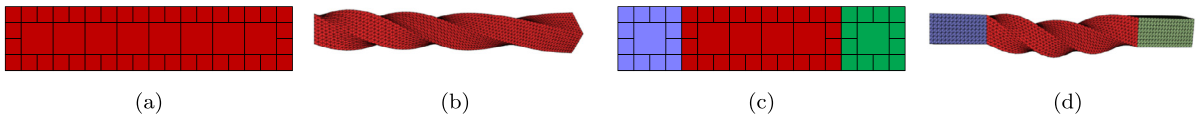

Fig. 2. Twisting comparison between homogeneous and heterogeneous bars. **(a)**: original bar with **homogeneous** material; **(b)**: twisted deformation of (a); **(c)**: original bar with **heterogeneous** materials; **(d)**: twisted deformation of (c). (For interpretation of the colors in this figure, the reader is referred to the web version of this article.)

subspaces with temporally-adaptive bases. And Y. Wang et al. (2015) designed linear deformation subspaces for realtime deformation by unifying linear blend skinning and the generalized barycentric coordinates. Besides, Wicke et al. (2009) partitioned fluids into multiple domains, and conducted modal reduction for each domain independently.

3. Method overview

As illustrated in Fig. 1, given an object to be simulated, we firstly overview the main functionalities of our method as follows.

Material-aware domain decomposition. We conduct material distribution analysis, decompose the already-labeled volume into independent homogeneous sub-domains (see Fig. 1 Input Model), and construct hierarchical octree-based FEM structure for each sub-domain (see Fig. 2(a), 2(c) Hexahedral Grid). Then we define the coupling constraints on the boundary interface of the connecting sub-domains (here we assume that grid is compatible on shared interface between adjacent sub-domains), and apply interface constraints into system equations by Lagrangian multiplier method. For example, in Fig. 2(c), the sub-domains marked by blue, red and green lattices are modeled as three independent FEM systems forming a geometry-space cross-domain coupling model, where blue and green denote rigid material and red denotes flexible material. Here we leverage the geometric continuity of the boundary vertices as cross-domain coupling constraints.

Sub-domain-specific physical modeling. For each sub-domain, we compute local matrices (stiffness, mass, and damping matrix) for Lagrangian equations of motion according to the material-specific parameters in an element-wise fashion. Benefiting from the structural regularity and uniformity of finite elements, we can pre-store the local stiffness matrices on GPU and access them according to the domain index in runtime. Please refer to Section 4.1 for details.

Sub-domain-specific deformation subspace generation. We dynamically generate and update sub-domain-specific deformation subspace by computing a set of time-varying vibrational modes based on the online modal reduction, which is expected to approximate the prior deformation space. The details are described in Section 4.2.

Cross-domain physical behavior coupling. Each sub-domain's behavior is simulated by adaptively switching between FEM and modal reuse. And we conduct cross-domain coupling by imposing neighboring sub-domains' geometric-continuity constraints to obtain the object's overall physical behaviors. Please refer to Section 4.3 for details.

State update due to cutting. The updating operation arising from cutting involves three main aspects: geometry, physics, and inter-domain constraints. In particular, when cross-domain cutting occurs, we need to update the constraints defined on the boundary interfaces of the affected sub-domains. We will detail them together with our numerical implementation in Section 5.

4. Sub-domain coupling of time-varying modal analysis and FEM simulation

4.1. Sub-domain-specific physical modeling

For each partitioned sub-domain, its dynamic behavior can be described by the displacements derived from Lagrangian equation of motion. For FE method, displacements are computed from the original deformation space, while for modal reduction method, they are computed from the deformation subspace as follows:

$$\mathbf{M}^i\ddot{\mathbf{u}}^i(t) + \mathbf{D}^i\dot{\mathbf{u}}^i(t) + \mathbf{K}^i\mathbf{u}^i(t) = \mathbf{f}^i_{ext}(t) + \mathbf{C}^{iT}\boldsymbol{\lambda}^i(t), \tag{1}$$

$$\mathbf{M}_r^i(t)\ddot{\mathbf{u}}_r^i(t) + \mathbf{D}_r^i(t)\dot{\mathbf{u}}_r^i(t) + \mathbf{K}_r^i(t)\mathbf{u}_r^i(t) = \mathbf{f}_{r,ext}^i(t) + \mathbf{C}_r^{iT}(t)\boldsymbol{\lambda}^i(t), \tag{2}$$

$$\mathbf{u}^i(t) = \boldsymbol{\Phi}^i(t)\mathbf{u}_r^i(t), \dot{\mathbf{u}}^i(t) = \boldsymbol{\Phi}^i(t)\dot{\mathbf{u}}_r^i(t), \ddot{\mathbf{u}}^i(t) = \boldsymbol{\Phi}^i(t)\ddot{\mathbf{u}}_r^i(t),$$

$$\mathbf{M}_r^i(t) = \boldsymbol{\Phi}^i(t)^T\mathbf{M}^i\boldsymbol{\Phi}^i(t), \mathbf{D}_r^i(t) = \boldsymbol{\Phi}^i(t)^T\mathbf{D}^i\boldsymbol{\Phi}^i(t), \mathbf{K}_r^i(t) = \boldsymbol{\Phi}^i(t)^T\mathbf{K}^i\boldsymbol{\Phi}^i(t),$$

$$\mathbf{f}_{r,ext}^i(t) = \boldsymbol{\Phi}^i(t)^T\mathbf{f}_{ext}^i(t), \mathbf{C}_r^{iT}(t) = \boldsymbol{\Phi}^i(t)^T\mathbf{C}^{iT}. \tag{3}$$

Here $\mathbf{M}^i$, $\mathbf{D}^i$ and $\mathbf{K}^i$ respectively represent the mass, damping, and stiffness matrix of sub-domain i in original deformation space. Because we adopt linear elasticity models, each element has linear shape functions, which should stay constant during simulation. $\mathbf{M}_r^i$, $\mathbf{D}_r^i$ and $\mathbf{K}_r^i$ respectively denote the Galerkin projection matrix of the mass, damping, and stiffness matrix of sub-domain i at time t (see Equation (3)). $\mathbf{u}^i$ represents the displacement vector of sub-domain i, and $\mathbf{u}_r^i$ represents the amplitude of the deformation modes in sub-domain i. Here we respectively use $\dot{\mathbf{u}}$ and $\ddot{\mathbf{u}}$ to indicate the first derivative and the second derivative with respect to time. Since we take a time-varying subspace constructing method, the basis matrix $\boldsymbol{\Phi}^i(t)$ of the reduced subspace of sub-domain i can be seen as a time-varying function, which makes projection matrix a time-varying function as well. And $\mathbf{f}_{ext}^i(t)$ is the external force of sub-domain i at time t, likewise, $\mathbf{f}_{r,ext}^i(t)$ is the reduced subspace projection of $\mathbf{f}_{ext}^i(t)$. $\mathbf{C}^i$ is the boundary interface constraint matrix of sub-domain i, and $\mathbf{C}_r^i(t)$ is the reduced subspace projection of $\mathbf{C}^i$. The boundary interface constraint matrix will be updated dynamically with cutting operations. $\boldsymbol{\lambda}^i(t)$ is the interface traction vector of sub-domain i at time t.

4.2. Sub-domain time-varying modal analysis

In this section, we mainly focus on the generation, the dynamic update of each sub-domain's time-varying deformation subspace, and the adaptive alternation scheme between sub-domain-specific finite element simulation and subspace simulation.

Sub-domain-specific deformation subspace generation. In our method, we do not use the displacement vectors from continuous perpendicular physical simulation's output to generate deformation subspace basis matrix. This is because that, such generating method for deformation subspace basis matrix usually requires the input data to be random samples from a stationary data source. However, in this paper, the generated data from physical simulation can be seen as data streams. Generally speaking, after the generation of deformation subspace basis matrix, the data output will be discarded and will not be embodied in deformation subspace. To solve this, we need to design a basis update strategy to dynamically embody the dominant deformation modes. The orthogonal vectors, generated from the aforementioned perpendicular basis matrix generation method, have no corresponding "dominant measurements" (e.g., eigenvalue or SVD's singular value), so they can not serve as a dominant measurement guidance to facilitate subspace update. To overcome these drawbacks, we conduct incremental SVD over the generated data streams to obtain a set of orthogonal vectors to construct the current deformation subspace basis matrix (see Kerschen and Golinvalas, 2002). As for dynamic simulation, we update deformation subspace via continuous fast low-rank modification to make it include current dominant deformation modes.

There are two reasons that inspire us to construct deformation subspace based on SVD. The first reason is, for the handling of sample data, SVD has been proved to approximate the data source with high quality (Frieze et al., 1998). The second reason is, according to De Moor et al. (1988), the resulted singular values can be defined as "oriented energy" to describe current system's major characteristics, so that they can account for the basis matrix's dominance during subspace update. More than that, because sub-domain deformation subspace basis matrix is constructed dynamically and is defined as a function of time, its deformation subspace's Galerkin projection operation (see Equation (3)) will be recalculated with the update of basis matrix. The time complexity of Galerkin projection is $O(N^3r^3)$, here N is the physical system's dimension of sub-domain i, and r is the deformation subspace's dimension of sub-domain i, it is still challenging for the adaptive and real-time switch between sub-domain physical simulation and subspace simulation. Thus, we need to design a gradual subspace generation method to reduce the time cost of Galerkin projection. We assume $\mathbf{X}_i = \begin{bmatrix}\mathbf{u}_i^0, & \mathbf{u}_i^1, & \cdots, & \mathbf{u}_i^n\end{bmatrix}$ is the physical simulation output set of sub-domain i, where column $\mathbf{u}_i^j$ denotes an output displacement vector of sub-domain i and n is the size of the current set. The SVD operation is solved by $\mathbf{X}_i = \mathbf{U}_i\mathbf{S}_i\mathbf{V}_i^T$, after which we can obtain the deformation basis vectors $\mathbf{U}_i$ and the corresponding oriented energy matrix $\mathbf{S}_i$ (also known as singular values) of current sub-domain i. When a newly-generated displacement vector $\mathbf{u}_i^{n+1}$ appends, the corresponding matrix operation is denoted as Equation (4). Then we transform the new dataset $\begin{bmatrix}\mathbf{X}_i & \mathbf{u}_i^{n+1}\end{bmatrix}$ via Equation (5), however, the generated $\begin{bmatrix}\mathbf{U}_i & \mathbf{u}_i^{n+1}\end{bmatrix}$ cannot be taken as the updated deformation subspace's basis matrix, because $\mathbf{U}_i$ and $\mathbf{u}_i^{n+1}$ are not orthogonal. Equation (6) further orthogonalizes $\mathbf{U}_i$ and $\mathbf{u}_i^{n+1}$ using Schimidt orthogonalization, and thus we can obtain Equation (7), wherein both the left matrix $\begin{bmatrix}\mathbf{U}_i & \mathbf{p}\end{bmatrix}$ and the right one $\begin{bmatrix}\mathbf{V}_i & \mathbf{e}_{n+1}\end{bmatrix}$ are orthogonal matrices. However, we cannot call Equation (7) a SVD, because the middle matrix $\mathbf{S}_i^{r+1}$ is not a diagonal matrix. So we use the method proposed in Brand (2006), Gu and Eisenstat (1993) to make a fast diagonalization to matrix $\mathbf{S}_i^{r+1}$ in Equation (7), whose time complexity is $O((r+1)^2)$. And we generate two $(r+1)\times(r+1)$ orthogonal matrices $\mathbf{U}_i'$, $\mathbf{V}_i'$ so that $\mathbf{S}_i^{r+1} = \mathbf{U}_i'\mathbf{S}'\mathbf{V}_i'^T$, where $\mathbf{S}'$ is a diagonal matrix, as shown in Equation (8). So far, we have transformed $\mathbf{X}_i = \mathbf{U}_i\mathbf{S}_i\mathbf{V}_i^T$ to the form of Equation (8). Here the benefit is that, its outermost matrix $\begin{bmatrix}\mathbf{U}_i & \mathbf{p}\end{bmatrix}$ is the augmentation of orthogonal matrix $\mathbf{U}_i$ from last time step. Furthermore, on the basis of Brand (2006) (see Equation (9)), we can leverage this characteristic to decompose Galerkin projection into multiple time steps ($\begin{bmatrix}(\mathbf{U}_i^0)^T, & (\mathbf{U}_i^1)^T, & \cdots, & (\mathbf{U}_i^r)^T\end{bmatrix}^T \mathbf{K}_i \begin{bmatrix}\mathbf{U}_i^0, & \mathbf{U}_i^1, & \cdots, & \mathbf{U}_i^r\end{bmatrix}$).

And we defer the operation $\mathbf{U}_i^{'(t)}\mathbf{U}_i^{'(t+1)}$ (called subspace rotations in numerical terms) in Equation (9) until switching physical simulation to subspace simulation, and this subspace rotation operation's time complexity is $O(r^3)$.

$$\mathbf{X}_i = \begin{bmatrix} \mathbf{X}_i & \mathbf{u}_i^{(n+1)} \end{bmatrix} = \begin{bmatrix} \mathbf{X}_i & \mathbf{0} \end{bmatrix} + \begin{bmatrix} \mathbf{0} & \mathbf{u}_i^{(n+1)} \end{bmatrix} \Leftrightarrow \mathbf{U}_i\mathbf{S}_i\mathbf{V}_i^T + \mathbf{u}_i^{(n+1)}\mathbf{e}_{n+1}^T, \mathbf{e}_{n+1}^T = \begin{bmatrix} 0, & \cdots, & 0, & 1 \end{bmatrix}, \tag{4}$$

$$\mathbf{U}_i\mathbf{S}_i\mathbf{V}_i^T + \mathbf{u}_i^{(n+1)}\mathbf{e}_{n+1}^T = \begin{bmatrix} \mathbf{U}_i & \mathbf{u}_i^{(n+1)} \end{bmatrix} \begin{bmatrix} \mathbf{S}_i & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} \mathbf{V}_i & \mathbf{e}_{n+1} \end{bmatrix}^T, \tag{5}$$

$$\mathbf{m} = \mathbf{U}_i^T\mathbf{u}_i^{(n+1)}, \mathbf{p} = \mathbf{u}_i^{(n+1)} - \mathbf{U}_i\mathbf{m}, \mathbf{p} = \|\mathbf{p}\|^{-1}\mathbf{p}, \tag{6}$$

$$\begin{bmatrix} \mathbf{X}_i & \mathbf{u}_i^{(n+1)} \end{bmatrix} = \begin{bmatrix} \mathbf{U}_i & \mathbf{p} \end{bmatrix} \begin{bmatrix} \mathbf{S}_i & \mathbf{m} \\ \mathbf{0} & \|\mathbf{p}\| \end{bmatrix} \begin{bmatrix} \mathbf{V}_i & \mathbf{e}_{n+1} \end{bmatrix}^T, \tag{7}$$

$$\begin{bmatrix} \mathbf{X}_i & \mathbf{u}_i^{(n+1)} \end{bmatrix} = \begin{bmatrix} \mathbf{U}_i & \mathbf{p} \end{bmatrix} \times \mathbf{U}_i' \times \mathbf{S}' \times \mathbf{V}_i'^T \times \begin{bmatrix} \mathbf{V}_i & \mathbf{e}_{n+1} \end{bmatrix}^T, \tag{8}$$

$$\mathbf{U}_i^{(t+1)}\mathbf{U}_i^{'(t+1)} = \begin{bmatrix} \mathbf{U}_i^{(t)} & \mathbf{p} \end{bmatrix} \mathbf{U}_i^{'(t)}\mathbf{U}_i^{'(t+1)}. \tag{9}$$

To verify the effectiveness of our subspace generation method, we have made linear model analyses (Shabana, 2012; Barbič and James, 2005) on the sub-domains of the FE mesh to obtain each sub-domain's LMA basis matrix (whether to cull rigid modes is not determined). Then we project our deformation basis matrix to the LMA basis matrix. As shown in Fig. 3, the low frequency modal vectors in LMA basis matrix can be well approximated by our basis matrix, meanwhile, the high frequency signals can also be well represented by our basis matrix. Due to the independence of the sub-domains, the assembled global system presents a block diagonal matrix fashion, wherein each matrix block denotes an independent sub-domain. As shown in Fig. 3(h), the matrix block's constitution is dependent on current system's status, either as a FE system (block matrix with larger size) or as a modal deformation system (block matrix with small size, and it reflects the user's attention as well as sub-domain's complexity).

Space–time-varying sub-domain modal updating. The update of sub-domain's deformation subspace will be triggered when current sub-domain's subspace cannot accurately depict the time-varying deformation status. We employ a strategy of maximally-reusing previous physical simulation results to make "rank 1 updating" for current deformation subspace. We add current physical simulation result into deformation subspace, according to subspace dimension's requirement, we determine to increase the subspace's dimension (see Equation (10)) or just maintain it (see Equation (11)). No matter the subspace's dimension is increased or not, the basis matrix in the updated subspace is the most dominant one in the current deformation space.

$$\mathbf{U}_i^{'(t+1)} = \begin{bmatrix} \mathbf{U}_i^{'(t)} & 0 \\ 0 & 1 \end{bmatrix} \mathbf{U}_i^{'(t+1)}, \mathbf{U}_i^{(t+1)} = \begin{bmatrix} \mathbf{U}_i^{(t)} & \mathbf{p} \end{bmatrix}, \tag{10}$$

$$\mathbf{U}_i^{'(t+1)} = \mathbf{U}_i^{'(t)}\mathbf{U}_{i,(1:r,1:r)}^{'(t+1)}, \mathbf{U}_i^{(t+1)} = \mathbf{U}_i^{(t)}. \tag{11}$$

Adaptive alternation scheme between sub-domain physical simulation and modal reuse. In simulation, to guarantee the simulation precision of each sub-domain, and guarantee the deformation subspace spanned by time-varying modals can continuously represent the physical deformation space, we resort to an online quality estimate. As shown in Fig. 2(d), we show an example of multiple-sub-domain simulation case, wherein each sub-domain's deformation degree and type are diverse, for example, blue and green sub-domains have significant elastic deformation, while the red one has a significant rigid displacement. So, to accurately depict the object's deformation process, we introduce a local error estimate method to support multiple sub-domains.

For each sub-domain, we need to consider two kinds of problems: the first is the time to switch from physical simulation to deformation subspace simulation, that is, the error metric problem between currently-generated deformation subspace and physical simulation space; the second is the time to switch from deformation subspace based simulation to physical simulation, that is, the prediction problem of deformation subspace's accumulative error. As mentioned in Rathinam and Petzold (2003), the total approximation error can be summarized as projection error $\mathbf{e}_\perp$ and integration error $\mathbf{e}_I$. In algebra, projection error is perpendicular to deformation subspace, i.e., $\mathbf{U}_i(t)\mathbf{U}_i^T(t)\mathbf{e}_\perp = \mathbf{0}$, while integration error is parallel to deformation subspace, i.e., $\mathbf{U}_i(t)\mathbf{U}_i^T(t)\mathbf{e}_I = \mathbf{e}_I$. For the first problem, we focus more on the computation of deformation subspace's projection error. As for the second one, we focus on the incremental estimation of the integration error in deformation subspace.

The projection error is defined as Equation (12), where $\mathbf{x}$ is the output of physical simulation and $\mathbf{U}_i(t)$ is the currently-generated deformation subspace basis matrix. The switch condition from physical simulation to subspace simulation is that current $\mathbf{U}_i(t)$ makes the $L_2 - norm$ of $\mathbf{Error}_{disp}$ less than threshold τ_{err}, as shown in Equation (13). In practice, each time we update deformation subspace's basis matrix $\mathbf{U}_i(t)$, we need to recompute $\mathbf{U}_i(t)\mathbf{U}_i^T(t)$ in order to compute Equation (12), and this will increase the global computation time. Our method adopts an optimized scheme, that is, we will not compute Equation (12) until the ratio (the singular values' summation of $\mathbf{U}_i(t-1)$ to that of new generated subspace basis vectors $\mathbf{U}_i(t)$) is more than threshold τ_{svd} ($0.9 < \tau_{svd} < 1.0$). This can avoid useless computation request so that it will not have

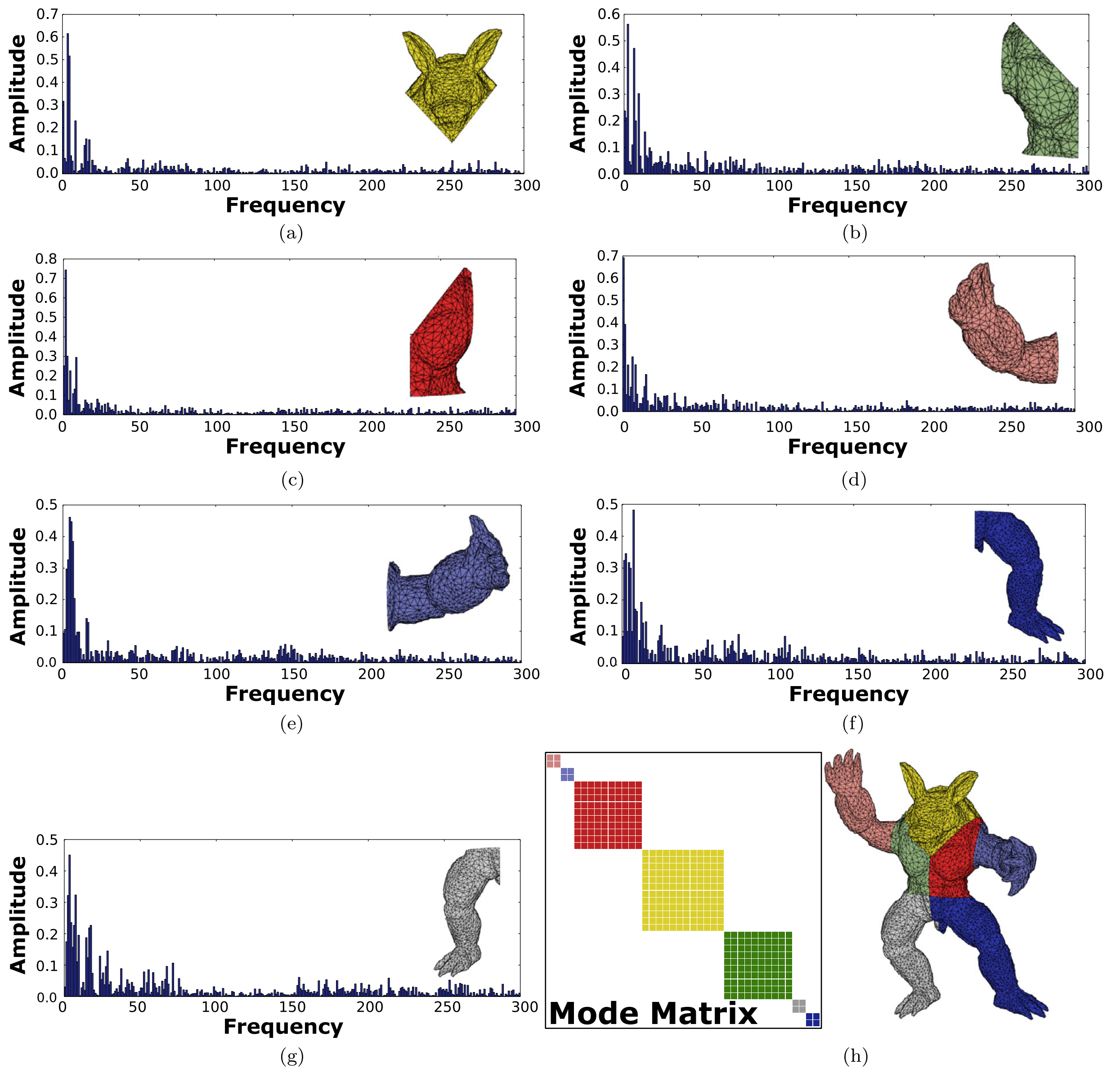

Fig. 3. Illustration of sub-domain-specific time-varying modal vibration frequencies. **(a)–(f)**: the vibration frequencies corresponding to different sub-domains; **(g)**: the reconstructed deformations by reusing the sub-domain vibration frequencies in (a)–(f); **(h)**: the pattern of global mode matrix.

significant influence on system's efficiency. For the computation of Equation (13), we resort to "small-sample statistical condition estimates" technique (Kenney and Laub, 1994). We do not solve Equation (13) directly, but leverage statistical methods to solve the expected value of the $L_2 - norm$ of $\mathbf{Error}_{disp}$. As shown in Equation (14), $\mathbf{u}_i^j$ is the column vector of current subspace basis matrix $\mathbf{U}_i(t)$, and W_N, W_r are called Wallis factors (Homescu et al., 2005). When the expected value of $L_2 - norm$ of $\mathbf{Error}_{disp}$ is less than threshold τ_{err}, suggesting that the approximation of physical space using current subspace basis matrix $\mathbf{U}_i(t-1)$ can satisfy system's requirement, it triggers the switch to subspace simulation. In another word, the significance of deformation subspace's basis matrix is controlled by threshold τ_{err} $(0.0 < \tau_{err} < 0.05\,\|\mathbf{x}\|_2)$. As for the second problem, considering we employ linear discretization to model physical system, we assume the system's error accumulation is also linear. Enlightened by Kim and James (2009), we project the physical simulation result to subspace basis matrix $\mathbf{U}_i(t)$ while switching from physical simulation to subspace simulation to obtain projection error, on that basis, the subspace system's error accumulation within s steps $(s = \lfloor \tau_{acc} / \|\mathbf{Error}_{disp}\| \rfloor)$ will not exceed system's requirement τ_{acc} $(0.0 < \tau_{acc} < 0.15\,\|\mathbf{x}\|_2)$.

$$\mathbf{Error}_{disp} = \mathbf{x} - \mathbf{U}_i(t)\mathbf{U}_i^T(t)\mathbf{x}, \tag{12}$$

$$\left\|\mathbf{Error}_{disp}\right\|_2 = \left\|\mathbf{x} - \mathbf{U}_i(t)\mathbf{U}_i^T(t)\mathbf{x}\right\|_2 < \tau_{err}, \tag{13}$$

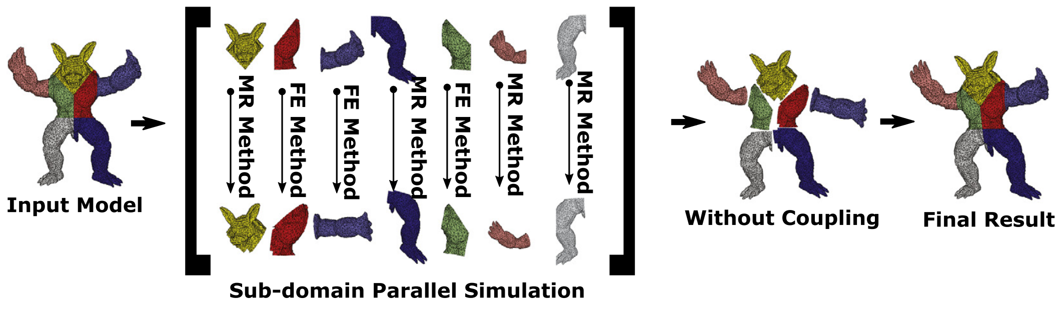

Fig. 4. Illustration of the independent selection of sub-domain-wise simulation schemes and their cross-domain tight-coupling.

$$E\left(\sqrt{\sum_{j=0}^{r}(\mathbf{u}_i^j\cdot\mathbf{Error}_{disp})^2}\right)=\frac{W_N}{W_r}\left\|\mathbf{Error}_{disp}\right\|,\ W_n=\begin{cases}\frac{1\cdot3\cdots(n-2)}{2\cdot4\cdots(n-1)} & ,n\in odd\\ \frac{2}{\pi}\frac{2\cdot4\cdots(n-2)}{1\cdot3\cdots(n-1)} & ,n\in even\end{cases}. \tag{14}$$

4.3. Cross-domain coupling among spatially-varying physical simulation and modal reuse

To accurately model the physics-geometry consistency among multiple sub-domains' interfaces, we employ Lagrangian multiplier method to depict the consistency of material points' displacements and traction among interfaces, and add these constraints into the system equations. For the convenience of depiction, we explain the interface constraints problem with two adjacent domains (α,β). The displacement vectors on the interface are denoted as $\mathbf{u}_\alpha^i$, $\mathbf{u}_\beta^i$ respectively, and the equilibrium condition of forces is $\mathbf{t}_\alpha=-\mathbf{t}_\beta$. We define interface traction as $\boldsymbol{\lambda}$, so $\mathbf{t}_\alpha=\boldsymbol{\lambda}$, $\mathbf{t}_\beta=-\boldsymbol{\lambda}$. Here, the model's system equations, including interface constraints, are defined as Equation (15), where $\mathbf{N}_\alpha$, $\mathbf{N}_\beta$ are displacement discretized shape function matrices on sub-domains' interfaces, and $\mathbf{N}_\lambda$ is the discretized shape function matrix of the interface's traction. When we have multiple sub-domains, we only need to assemble these interfaces' constraints into a constraint equation system and include it into our system equations. Over-constraining and locking problems may arise when applying constraints on interfaces, to solve this problem, we decrease the number of boundary constraint points according to Yang et al. (2010). After including interface consistency constraints into original system equations, the system equations become a so-called saddle point system. And the emergence of saddle point system brings new challenges for system solving, especially for a computation framework employing Krylov subspace methods (Dollar et al., 2006b), because the joint of constraints will transform the original symmetric positive definite system into an indefinite system while Krylov subspace method is not applicable to solve indefinite systems (for more details, see Subsection 5.2). Moreover, when the deformation subspace of sub-domain i satisfies accuracy requirements and triggers the switch operation for the sub-domain-wise simulation scheme, the system equations' size will change, for instance, $\mathbf{N}_\alpha+\mathbf{N}_\beta\rightarrow\mathbf{N}_\alpha+\mathbf{r}_\beta$. In practice, the system equations will continuously evolve independent of sub-domains (as shown in Fig. 4), which will give rise to severe challenges for any iterative solving method (see more details in Subsection 5.2).

$$\begin{bmatrix}\mathbf{K}_\alpha & \mathbf{0} & \mathbf{C}_\alpha^T\\ \mathbf{0} & \mathbf{K}_\beta & \mathbf{C}_\beta^T\\ \mathbf{C}_\alpha & \mathbf{C}_\beta & \mathbf{0}\end{bmatrix}\begin{bmatrix}\mathbf{u}_\alpha\\ \mathbf{u}_\beta\\ \boldsymbol{\lambda}\end{bmatrix}=\begin{bmatrix}\mathbf{f}_\alpha\\ \mathbf{f}_\beta\\ \mathbf{0}\end{bmatrix}$$

$$\mathbf{C}_\alpha=\int\limits_{\Gamma_I}\mathbf{N}_\alpha^T\mathbf{N}_\lambda d\Gamma,\mathbf{C}_\beta=\int\limits_{\Gamma_I}\mathbf{N}_\beta^T\mathbf{N}_\lambda d\Gamma. \tag{15}$$

5. CUDA-based numerical implementation

5.1. Large deformation handling

For the FE modeling of each sub-domain, we adopt widely-used Cauchy strain tensor. Although the generation of sub-domain's deformation subspace takes linear strain resulted deformations as input, the accuracy of Cauchy strain tensor is only applicable to moderate small deformation, because it tends to diverge from the correct solution under large deformation. In addition, in the deformation subspace's approximation, modal superposition can not represent the intermediate rotational displacement (Yang et al., 2013b; Guo and Qin, 2005). To solve the rotational deformation problems, we employ modal warp technique proposed in Müller et al. (2002), Choi and Ko (2005) to uniformly solve rotational deformation problems encountered in physical simulation and modal reduction.

$$\tilde{\mathbf{u}}(t)=\begin{cases}\tilde{\mathbf{R}}(t)\mathbf{u}(t) & ,\textit{using FE Method}\\ \tilde{\mathbf{R}}(t)\boldsymbol{\Phi}(t)\mathbf{u}_r(t) & ,\textit{using MR Method}\end{cases}, \tag{16}$$

$$\tilde{\mathbf{R}}_i(t)=\left[\mathbf{I}+(\hat{\mathbf{w}}_i(t)\times)\frac{1-\cos\|\mathbf{w}_i(t)\|}{\|\mathbf{w}_i(t)\|}+(\hat{\mathbf{w}}_i(t)\times)^2\left(1-\frac{\sin\|\mathbf{w}_i(t)\|}{\|\mathbf{w}_i(t)\|}\right)\right],\hat{\mathbf{w}}_i(t)=\frac{\mathbf{w}_i(t)}{\|\mathbf{w}_i(t)\|}. \tag{17}$$

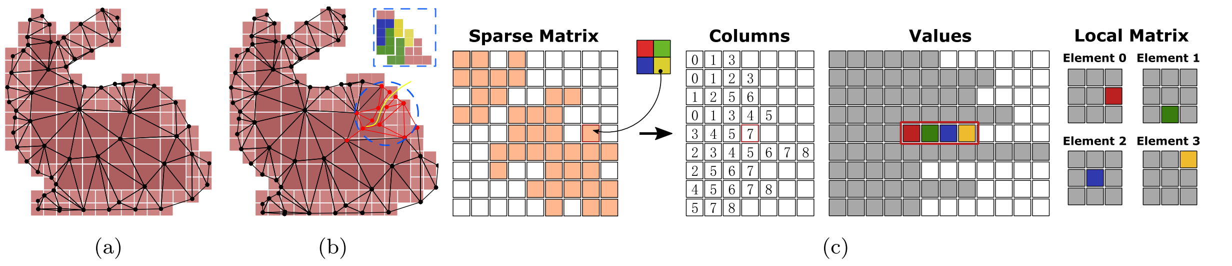

Fig. 5. Illustration of the trajectory due to cutting together with the specifically-designed data structure for efficient matrix operation. (**a**): initial volumetric grid; (**b**): volumetric grid after cutting; (**c**): incompletely-assembling sparse matrix data structure based on CUDA. (For interpretation of the colors in this figure, the reader is referred to the web version of this article.)

As shown in Equations (16), we extract vertices' rotation vectors in local coordinate system from the obtained resulting displacement field, and separate the rotation part from displacement field. Here t denotes current time step, $\tilde{\mathbf{R}}(t)$ is rotating separation operator, and $\tilde{\mathbf{R}}(t)$ is a block diagonal matrix, whose sub-matrix block is a 3×3 matrix ($\tilde{\mathbf{R}}(t) = \left[\delta_{ij}\tilde{\mathbf{R}}_i(t)\right]$) denoting a vertex's rotating separation operator. Equation (17) is the computation formula of vertex's rotating separation operator, where $\mathbf{w}_i$ is the local rotation vector derived from displacement curl of vertex i. $\hat{\mathbf{w}}_i(t)\times$ denotes the standard skew symmetric matrix of $\hat{\mathbf{w}}_i(t)$. In our method, $\mathbf{w}_i$ is the average of all the rotation vectors of the finite elements sharing one vertex i.

5.2. Cutting simulation

For cutting operation, we represent the cutting tool with a triangle mesh, and use the triangle mesh to track cutting trajectory. When cutting operation occurs, by making collision detection between hierarchical hexahedral mesh and triangle patches, we can obtain the hexahedral elements interacting with the cutting tool. For coarse elements, we need to refine them for simulation accuracy and clone the cut elements to satisfy the topology change (see Fig. 5). In Fig. 5(b), the range denoted by blue dashed line circle is the grid region related to cutting, and yellow curve represents the cutting trajectory. The blue dashed line box at the top right corner shows the topology change diagram of volume mesh. And in the diagram, pink means the element that is not involved in cutting; blue and green mean the refined elements (its predecessor is a coarse element), and the difference between them is that the green one is cloned further; yellow means the element only involves in cloning operation, because the yellow element is already in fine scale before cutting.

After handling the grid's geometry and topology update, for physical simulation system, we need to further redistribute physical attributes (such as velocity field, acceleration field, and gravity field) of the original grid onto new grid to guarantee a smooth transition of motion state. This process is achieved by assigning clone elements' nodes with nodes' physical attributes of source elements. When conducting cross-domain cutting operation, cutting will cause boundary interface constraints to change, and we assign the cloned nodes' constraint attributes (if any) to new generated nodes just as that in handling physical attributes. Benefited from the regularity of hexahedral elements, element-wise shape function matrix, element-wise mass matrix, element-wise damping matrix, and element-wise interface constraint matrix are consistent, and can be pre-computed, so new generated elements can be directly obtained without additional complex computation. For deformation subspace, cutting makes previous deformation subspace no longer well approximate current sub-domains' physical behavior, so we need recompute deformation subspace. This paper employs a strategy of maximally-reusing previous physical simulation results, and here we need to make augmentation adjustment on recorded prior physical simulation output $\mathbf{X}_i$ as follows: first enlarge the size of column vectors to meet current sub-domain's freedom; similar to the operation of redistributing physical attributes, copy source nodes' previous displacements onto clone nodes. The augmented displacement field can serve as the input for deformation subspace generation method. To accelerate the generation of new deformation subspace, we cut down original displacement field and only remain the nearest $r/2$ augmented displacements resulted from current time step, where r is the dimension of current deformation subspace.

5.3. Preconditioned conjugate gradient method for indefinite systems

According to the above description, the simulation method adopted by each sub-domain may change dynamically (see Subsection 4.2), and when cutting happens, system's size will enlarge further as well (see Subsection 5.2). With regard to global system equations, in each simulation loop, numerical solver may encounter a "brand new" system equation. When all sub-domains employ FE physical simulation method, system equations become a large sparse diagonal matrix. However, when all sub-domains employ deformation subspace simulation method, system equations become a small dense diagonal matrix. What's more, because of the constraints among sub-domains, system equations tend to become an indefinite system from a symmetric positive definite one. The frequent changes of sub-domains' systems put forward performance requirements for the assembly of global system matrix, and the difference in matrices and/or matrices' sizes also brings performance requirement for solver's preconditioning process. Because we may face a large sparse matrix, we choose iterative solver (conjugate gradient method) for this problem. Unfortunately, CG method is not able to solve an indefinite system, so we need to employ pre-conditioner technique to strengthen the ability of CG method. Because the problem has high performance demand for solver's preconditioning process, we urgently need a pre-conditioner that can be constructed fast and can effectively solve an indefinite system. We will detail our solution for this problem below.

We assume the target system matrix $\mathbf{S}(t)$ and the pre-conditioner $\mathbf{P}(t)$ are like Equation (18), considering they change with simulation time step, which are functions of time t. Here $\mathbf{A}(t)$ is a block diagonal matrix consisting of each sub-domain (see Subsection 4.2). $\mathbf{G}(t)$ is a diagonal matrix and its values on the principal diagonal are the same as those of $\mathbf{A}(t)$, i.e., $\mathbf{G}(t) = diag(\mathbf{A}(t))$. When solving indefinite systems using CG method, we need to solve the orthogonal projection onto the constraint term's null space brought by constraint $\mathbf{C}(t)$ (Rees and Scott, 2014). For constraint $\mathbf{C}(t)$ at each time step, the solution for its null space's projection operator is disastrous for solving system equations, we employ a close approximation of the orthogonal projection operator introduced in Dollar et al. (2006a) (also see Equation (15)). Thus, it avoids solving the projection onto the null space at each time step but introducing round-off error in iterative solving. In iterative solving, we need to introduce a modification to solution to offset the influence of round-off error, see Algorithm 2. By observing Equation (18), we can know that the pre-conditioner $\mathbf{P}(t)$ is an indefinite system as well. In order to solve pre-conditioner, we adapt $\mathbf{P}(t)$ to Equation (19), where $\mathbf{G}_{11}(t)$, $\mathbf{C}_1(t)$ and $\mathbf{C}_1^T(t)$ are $m \times m$ matrices and m is the row number of constraint matrix $\mathbf{C}(t)$. We employ Schilder's factorization technique (Dollar et al., 2006a) to decompose Equation (19), and we set $\mathbf{G}_{11}(t)$ as a zero matrix so that we can use Algorithm 3 to efficiently solve the pre-conditioner.

Besides, Algorithm 1 describes the process of solving global indefinite system equations at each time step. In practical applications, Algorithm 3 requires $\mathbf{C}_1^T(t)$ to be invertible. However, when handling complex problems, $\mathbf{C}_1^T(t)$ will be not invertible, in this case, we employ LU decomposition on constraint matrix $\mathbf{C}(t)$ to get its left permutation matrix $\mathbf{H}(t)$, and take it as a permutation operation to Equation (18) (also see Equation (20)). And because the pre-conditioner $\mathbf{P}(t)$ is obtained from Equation (18), we need to reconstruct pre-conditioner using Equation (20).

$$\mathbf{S}(t) = \begin{bmatrix} \mathbf{A}(t) & \mathbf{C}^T(t) \\ \mathbf{C}(t) & \mathbf{0} \end{bmatrix}, \mathbf{P}(t) = \begin{bmatrix} \mathbf{G}(t) & \mathbf{C}^T(t) \\ \mathbf{C}(t) & \mathbf{0} \end{bmatrix}, \tag{18}$$

$$\mathbf{P}(t) = \begin{bmatrix} \mathbf{G}_{11}(t) & \mathbf{0} & \mathbf{C}_1^T(t) \\ \mathbf{0} & \mathbf{G}_{22}(t) & \mathbf{C}_2^T(t) \\ \mathbf{C}_1(t) & \mathbf{C}_2(t) & \mathbf{0} \end{bmatrix}, \tag{19}$$

$$\hat{\mathbf{S}}(t) = \begin{bmatrix} \mathbf{H}^T(t)\mathbf{A}(t)\mathbf{H}(t) & \mathbf{H}^T(t)\mathbf{C}^T(t) \\ \mathbf{C}(t)\mathbf{H}(t) & \mathbf{0} \end{bmatrix} = \begin{bmatrix} \hat{\mathbf{A}}(t) & \hat{\mathbf{C}}^T(t) \\ \hat{\mathbf{C}}(t) & \mathbf{0} \end{bmatrix}. \tag{20}$$

Algorithm 1 PCG method for indefinite systems.

```
Input: $\mathbf{A}(t)$, $\mathbf{C}(t)$, $\mathbf{C}_1(t)$, $\mathbf{C}_2(t)$, $\mathbf{G}_{22}(t)$, $\mathbf{f}_{ext}(t)$;
Output: global displacement field at current time t;
Initialization: $n$ is the row number of $\mathbf{A}(t)$, $m$ is the row number of $\mathbf{C}(t)$;
    construct vectors: $\mathbf{x}(n)$, $\hat{\mathbf{x}}(n)$, $\mathbf{r}(n)$, $\mathbf{g}(n)$, $\mathbf{p}(n)$, $\mathbf{q}(n)$
                       $\hat{\mathbf{y}}(m)$, $\mathbf{w}(m)$, $\mathbf{v}(m)$, $\mathbf{a}(m)$, $\mathbf{t}(m)$, $\mathbf{h}(m)$, $\mathbf{l}(m)$
$\mathbf{w} = -\mathbf{C}(t)\mathbf{x}$
solve_preconditioner($\mathbf{G}_{22}(t), \mathbf{C}_1(t), \mathbf{C}_2(t), n, m, \hat{\mathbf{x}}, \hat{\mathbf{y}}, \mathbf{r}, \mathbf{w}$)
$\mathbf{x} = \mathbf{x} + \hat{\mathbf{x}}$
$\mathbf{r} = \mathbf{A}(t)\mathbf{x} + \mathbf{C}^T(t)\hat{\mathbf{y}} - \mathbf{f}_{ext}(t)$
solve_preconditioner($\mathbf{G}_{22}(t), \mathbf{C}_1(t), \mathbf{C}_2(t), n, m, \mathbf{g}, \mathbf{v}, \mathbf{r}, \mathbf{w}$)
residual_update($\mathbf{a}, \mathbf{g}, \mathbf{r}, \mathbf{v}, \mathbf{w}, \mathbf{C}(t), \mathbf{G}_{22}(t), \mathbf{C}_1(t), \mathbf{C}_2(t)$)
$\mathbf{t} = \mathbf{v} + \mathbf{a}$, $\mathbf{p} = -\mathbf{g}$, $\mathbf{h} = -\mathbf{t}$
$\mathbf{q} = \mathbf{A}(t)\mathbf{p}$
$sigma = \mathbf{r}\cdot\mathbf{g} + \mathbf{w}\cdot\mathbf{t}$, $gamma = \mathbf{p}\cdot\mathbf{q} + \mathbf{h}\cdot\mathbf{l}$, $r\_nrm2 = \mathbf{r}\cdot\mathbf{g}$
LOOP: termination condition $r\_nrm2 < (1e-6)\cdot\|\mathbf{f}_{ext}(t)\|$
    $alpha = sigma/gamma$
    $\mathbf{x} = \mathbf{x} + alpha \times \mathbf{p}$, $\mathbf{r} = \mathbf{r} + alpha \times \mathbf{q}$
    $\mathbf{a} = \mathbf{a} + alpha \times \mathbf{h}$, $\mathbf{w} = \mathbf{w} + alpha \times \mathbf{l}$
    solve_preconditioner($\mathbf{G}_{22}(t), \mathbf{C}_1(t), \mathbf{C}_2(t), n, m, \mathbf{g}, \mathbf{v}, \mathbf{r}, \mathbf{w}$)
    residual_update($\mathbf{a}, \mathbf{g}, \mathbf{r}, \mathbf{v}, \mathbf{w}, \mathbf{C}(t), \mathbf{G}_{22}(t), \mathbf{C}_1(t), \mathbf{C}_2(t)$)
    $\mathbf{t} = \mathbf{a} + \mathbf{v}$
    $old\_sigma = sigma$, $sigma = \mathbf{r}\cdot\mathbf{g} + \mathbf{w}\cdot\mathbf{t}$, $beta = sigma/old\_sigma$
    $\mathbf{p} = -\mathbf{g} + beta \times \mathbf{p}$, $\mathbf{h} = -\mathbf{t} + beta \times \mathbf{h}$
    $\mathbf{q} = \mathbf{A}(t)\mathbf{p}$
    $gamma = \mathbf{p}\cdot\mathbf{q} + \mathbf{h}\cdot\mathbf{l}$, $r\_nrm2 = \mathbf{r}\cdot\mathbf{g}$
```

Algorithm 2 Method for solving the pre-conditioner.

```
Input: $\mathbf{G}_{22}(t)$, $\mathbf{C}_1(t)$, $\mathbf{C}_2(t)$, $n$, $m$, $\mathbf{g}$, $\mathbf{v}$, $\mathbf{r}$, $\mathbf{w}$;
Output: solve preconditioner to obtain the result of displacement $\mathbf{g}$ and traction $\mathbf{v}$
Initialization:
    construct vectors: $\mathbf{g}_1(m)$, $\mathbf{g}_2(n-m)$, $\mathbf{r}_1(m)$, $\mathbf{r}_2(n-m)$
Implementation:
    $\begin{bmatrix}\mathbf{r}_1 & \mathbf{r}_2\end{bmatrix} = \mathbf{r}$, $\mathbf{v} = \mathbf{C}_1^{-1}(t)^T\mathbf{r}_1$
    $\mathbf{g}_2 = \mathbf{G}_{22}(t)^{-1}(\mathbf{r}_2 - \mathbf{C}_2(t)^T\mathbf{v})$, $\mathbf{g}_1 = \mathbf{C}_1^{-1}(t)(\mathbf{w} - \mathbf{C}_2(t)\mathbf{g}_2)$v
    $\mathbf{g} = \begin{bmatrix}\mathbf{g}_1 & \mathbf{g}_2\end{bmatrix}$
```

Algorithm 3 Residual update.

```
Input: $\mathbf{a}$, $\mathbf{g}$, $\mathbf{r}$, $\mathbf{v}$, $\mathbf{w}$, $\mathbf{C}(t)$, $\mathbf{G}_{22}(t)$, $\mathbf{C}_1(t)$, $\mathbf{C}_2(t)$;
Output: the update of vectors $\mathbf{a}$, $\mathbf{g}$, $\mathbf{r}$, $\mathbf{v}$, $\mathbf{w}$;
Implementation:
    $\mathbf{r} = \mathbf{r} - \mathbf{C}(t)^T\mathbf{v}$
    $\mathbf{a} = \mathbf{a} + \mathbf{v}$, $\mathbf{w} = \mathbf{0}$
    solve_preconditioner($\mathbf{G}_{22}(t), \mathbf{C}_1(t), \mathbf{C}_2(t), n, m, \mathbf{g}, \mathbf{v}, \mathbf{r}, \mathbf{w}$)
```

5.4. Sparse matrix data structure design for incomplete assembling

As described above, the frequent update of global system equations and the requirement to permute matrix's row and column in constructing process bring new demands for the fast assembly and permutation between matrix rows and columns. Therefore, we develop an incompletely-assembling Sparse Matrix Data Structure based on CUDA (Fig. 5(c)). First, for fast assembly, we only assemble matrix's rows while not assembling columns explicitly. The advantages of this design lie in that, it is more effective than matrix-free structure without total assembling because there involves no row locating in numeric indexing, and it is also more effective than matrix structure with total assembling because of no atomic accumulating for the same location and less bandwidth occupation (see Fig. 5(c), the red, green, blue, yellow elements are stored independently without summation operation). Second, as shown in Fig. 5(c), the matrix's row and column adopt bi-level indexing structure, so that permutation between rows and columns only needs to modify the index of rows and columns without moving data actually.

6. Experimental results

We have implemented a prototype system using C++ and CUDA. All the experiments are run on a desktop PC with NVIDIA GeForce GTX 770 GPU, Intel Core i7 3.60 GHz CPU and 4G RAM.

To verify the accuracy of our method, we simulate a Cactus model. First, we decompose the triangle mesh into 4 domains according to its structure as an input model (as shown in Fig. 6(a)). For the input model with multi-domain information, we construct a hierarchical hexahedral octree grid for every sub-domain respectively and construct a FE model for each sub-domain respectively. In order to explore our method in the performance of different resolution on the octree grid, we construct FE model for high-resolution (Fig. 6(b)) and low-resolution (Fig. 6(c)) grid respectively, and simulate high-resolution grid model with FE method as reference, as shown in Fig. 6(d), 6(g), 6(j). In the initial simulation phase, every sub-domain is simulated with FE method, and current system equations' size is shown in the sub-column "Max" of the Table 1's column "Before cutting". At the same time with physical simulation, subspace is constructed dynamically for every sub-domain respectively, and while the constructed subspace can satisfy the requirement of error precision (here, error precision is 95%), related sub-domain switches to subspace method to simulate (as Fig. 6(e), current system equations' size is $13\,855 \times 13\,855$, red: 26, yellow: 25, white: 31, green: 13 773, and Fig. 6(f), current system equations' size is 2867×2867, red: 10, yellow: 10, white: 16, green: 2831). After the occurrence of cutting operation, influenced sub-domains are simulated with FE method and other unaffected sub-domains still employ current simulation method and current system equations' size is shown in the sub-column "Max" of the Table 1's column "After cutting". As mentioned above, when our new constructed subspace can satisfy the requirement of error precision again, related sub-domain should switch to subspace method to simulate, as shown in Fig. 6(h), 6(i), 6(k), 6(l).

And to verify the effectiveness of processing with cutting operation, we process a steak model with complex material and corresponding domain partition. First, we conduct material-aware voxelization over the input model (as Fig. 7(a)) to generate a hierarchical hexahedral grid and implement FE modeling for every domain respectively (as shown in Fig. 7(b)). Note that, every independent sub-domain is assigned different material parameters due to different material information (bone: Young's Modulus 9.0E+9, muscle: Young's Modulus 3.0E+9, fat: Young's Modulus 1.0E+9). Zigzag-shaped and J-shaped cutting operation occurred in red, yellow and blue sub-domains. The grid generated after incision's elements' splitting and cloning as response to cutting is shown as Fig. 7(d), 7(f) and the final result of cutting is shown as Fig. 7(c), 7(e). Detailed Performance statistics can be found in Table 1.

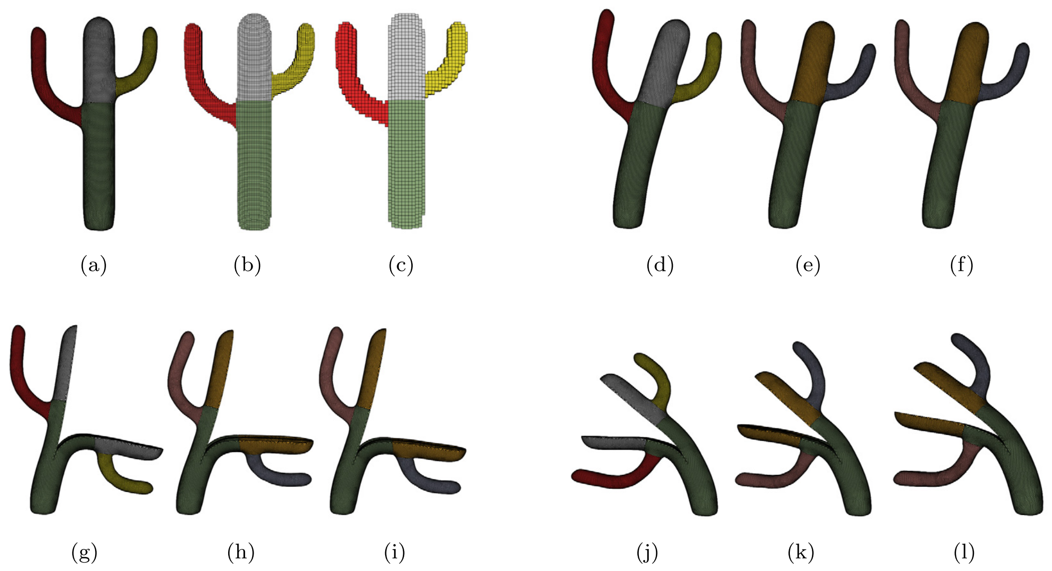

Fig. 6. Cutting simulation over the 4-domain Cactus model. **(a)**: the input model with already-labeled multiple domains; **(b)**: high-resolution octree-based representation; **(c)**: low-resolution octree-based representation; **(d)**: the simulation result based on the FEM model; **(e)**: the simulation result based on high-resolution grid using different simulation methods (red, yellow, and white regions with the subspace method and green region with the FE method); **(f)**: the simulation result based on low-resolution grid using different simulation methods (red, yellow, and white regions with the subspace method and green region with the FEM method); **(g)**: the simulation result of the FEM model after cross-domain cutting (i.e., cut across white and green sub-domains); **(h)**: the simulation result based on high-resolution grid using different simulation methods after cutting (red, yellow, and white regions with the subspace method and green region with the FE method); **(i)**: the simulation result based on low-resolution grid using different simulation methods after cutting (red, yellow, and white regions with the subspace method and green region with the FE method); **(j)**: the simulation result of the FEM model after cross-domain cutting; **(k)**: the simulation result on high-resolution grid using different simulation method after cutting (red, yellow, and white regions with the subspace method and green region with FE method); **(l)**: the simulation result based on low-resolution grid using different simulation methods after cutting (red, yellow, and white regions with the subspace method and green region with the FE method). (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

Table 1
Performance statistics.

Model	Elements	Domains	Before cutting		After cutting		FPS
			Max	Min	Max	Min	(Ours/FEM)
Cactus (low-res)	1246	4	6015 × 6015	54 × 54	6471 × 6471	58 × 58	41 / 25
Cactus (hi-res)	7401	4	29 349 × 29 349	115 × 115	30 711 × 30 711	136 × 136	35 / 16
Steak ("J" cuts)	2976	5	12 510 × 12 510	112 × 112	13 317 × 13 317	122 × 122	38 / 20
Steak ("Z" cuts)	2976	5	12 510 × 12 510	112 × 112	13 425 × 13 425	123 × 123	38 / 20
Cube	10 648	2	38 088 × 38 088	–	64 170 × 64 170	69 × 69	28 / 15
Armadillo	3657	7	16 083 × 16 083	161 × 161	18 675 × 18 675	163 × 163	29 / 19

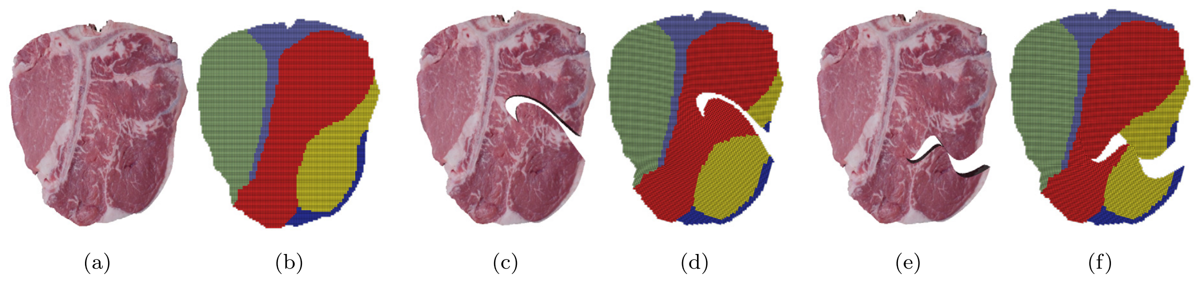

Fig. 7. Cutting simulation over the multi-material Steak model. **(a)**: the input Steak model with multiple materials; **(b)**: domain partition and voxelization according to the material attributes; **(c)**: J-shaped cutting simulation result; **(d)**: the updated grid after J-shaped cutting operation; **(e)**: Zigzag-shaped cutting simulation result; **(f)**: the updated grid after Zigzag-shaped cutting operation. (For interpretation of the colors in this figure, the reader is referred to the web version of this article.)

To verify our method's effectiveness of handling more complex cutting operation, we simulate a Cube model, as shown in Fig. 8. Fig. 8(a) shows the initial state of model, it is composed of two sub-domains (blue: Young's Modulus 6.0E+9 and yellow: Young's Modulus 1.0E+9). Because of the cross-domain cutting operation, the DOF of model increases to 64 170

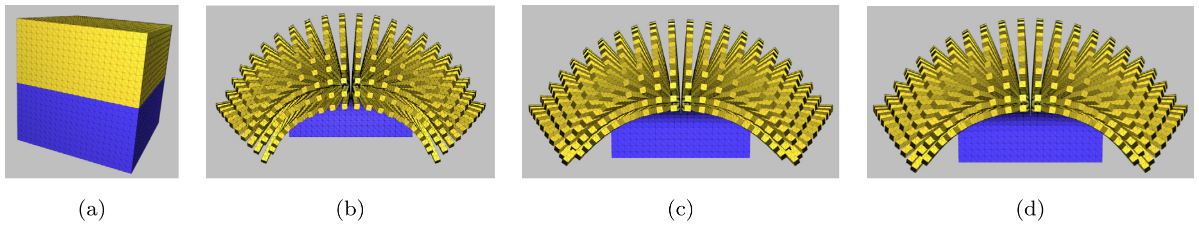

Fig. 8. Conducting complicated cutting over a cube model. **(a)**: original model with two sub-domains; **(b)**: the simulation output of all sub-domains with full FE method; **(c)**: the output of simulation using FE method in yellow region and using subspace method in blue region; **(d)**: the simulation output of all sub-domains with full subspace method. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

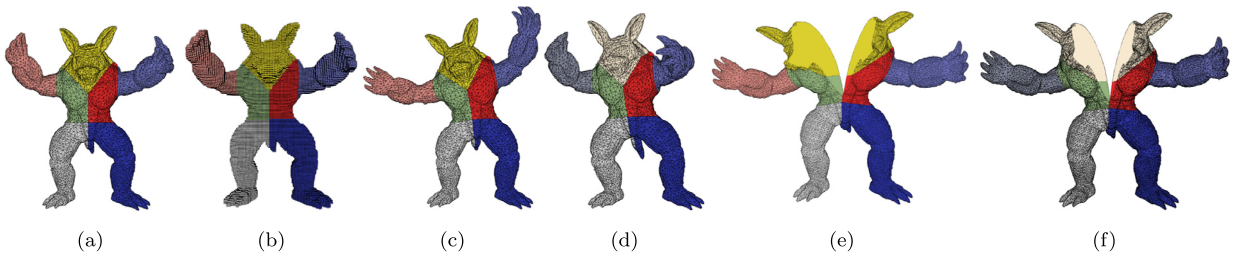

Fig. 9. Cutting simulation over multi-domain Armadillo model. **(a)**: original input model; **(b)**: hierarchical octree-based structure for each sub-domain; **(c)**: the simulation output of all sub-domains with full FE method; **(d)**: the simulation output with subspace method in head, right and left hand region; **(e)**: output with full FE method after cutting; **(f)**: the simulation output with subspace method in head, right and left hand region after cutting. (For interpretation of the colors in this figure, the reader is referred to the web version of this article.)

(yellow: 36 432, blue: 27 738) from 38 088 (yellow: 19 044, blue: 19 044), as shown in Fig. 8(b), and now both the two sub-domains are simulated with FE method (system equations' size is $64\,170 \times 64\,170$). And during physical simulation, subspace is constructed dynamically for every sub-domain and simulation switches to subspace method if the subspace satisfies the requirement of error precision, as shown in Fig. 8(c), where blue region employs subspace method to simulate, while yellow region still maintains FE method (system equations' size is $27\,772 \times 27\,772$). And as soon as the yellow region's subspace could satisfy error precision, simulation of the region switches to subspace method, as shown in Fig. 8(d), and current system equations' size is 69×69.

Finally, Fig. 9 shows a comprehensive experiment, wherein the Armadillo model is decomposed into 7 sub-domains (represented by different colors, as Fig. 9(a), 9(b)). Let us suppose the head and the upper body are the regions of interest, and we assign higher error precision (95%) to yellow, green and red regions (ROIs), while assign lower error precision (87%) to purple, pink, white and blue regions. Cutting operation occurs in the regions of interest. We refine, clone, and split the elements influenced by cutting, and simulate the influenced regions using FE method (as shown in Fig. 9(e)). In the same way, we construct subspace dynamically during physical simulation and we switch to subspace method when error precision is satisfied by constructed subspace, as shown in Fig. 9(f). Detailed Performance statistics can be found in Table 1.

7. Conclusion and discussion

In this paper, we have systematically presented a novel and versatile method to address a suite of research challenges encountered in modal reduction based real-time deformation and arbitrary cutting simulation of heterogeneous objects (with multiple sub-domains and large variations of material distribution). The most critical idea of our novel approach is to conduct a dynamic interchange between adaptively integrating material-aware and/or geometry-structure-aware simulation with full-physics capability and performing deformation reconstruction based on sub-domain-specific local modal' reuse, and all of the above numerical procedures have been implemented in a CUDA-centric parallel computation framework. The novel technical components within our new framework include: the space–time-varying local modal generation from previous-cycles' fully-physical simulation, the adaptive alternation scheme between sub-domain physical simulation and modal reuse, the cross-domain coupling of spatially-varying numerous deformations (i.e., some sub-domains' deformation is from physical simulation, and others are from modal-subspace reconstruction), and the sub-domain-level parallel implicit integration solvers supporting CUDA-enabled numerical computation, which collectively equip our method with remarkable advantages in terms of realtime efficiency, high-accuracy simulation, unconditional stableness, and practical versatility.

Currently, our prototype system is still a proof-of-concept only at the experimental stage, hence, it is not yet of practical use due to certain limitations. For example, we should introduce some collision detection function (Li et al., 2014b) into our current system to make it possible to support more complex interactions far beyond the simple cutting operations. And we should also incorporate certain mature techniques (Barbič and James, 2010; Teng et al., 2014) to accommodate self-collision caused by elastic deformation. In addition, our ongoing research efforts are concentrated on seeking an efficient optimization method to guarantee the physical accuracy in an absolute sense. Meanwhile, it also deserves our efforts to extend our method to handle more sophisticated fluid-elasticity coupling phenomena. In terms of limitations, it should be noted that, our domain coupling method's requirement for construction consistency of sub-domain interface may not be appropriate in practical applications and our current method is lack of flexibility in supporting more comprehensive adaptivity when handling much more complex boundary interfaces.

Acknowledgements

This research is supported in part by National Natural Science Foundation of China (Nos. 61190120, 61190121, 61190125, 61300067, and 61532002) and National Science Foundation of USA (Nos. IIS-0949467, IIS-1047715, and IIS-1049448).

References

Allard, J., Cotin, S., Faure, F., Bensoussan, P.-J., Poyer, F., Duriez, C., Delingette, H., Grisoni, L., 2007. SOFA – an open source framework for medical simulation. In: MMVR 15-Medicine Meets Virtual Reality. In: Stud. Health Technol. Inform., vol. 125. IOP Press, pp. 13–18.

Allard, J., Courtecuisse, H., Faure, F., 2011. Implicit FEM solver on GPU for interactive deformation simulation. In: Hwu, W. mei W. (Ed.), GPU Computing Gems Jade Edition. In: Applications of GPU Computing Series. Elsevier, pp. 281–294. https://hal.inria.fr/inria-00589200.

An, S.S., Kim, T., James, D.L., 2008. Optimizing cubature for efficient integration of subspace deformations. ACM Trans. Graph. 27 (5), 165. http://dx.doi.org/10.1145/1409060.1409118.

Barbič, J., 2007. Real-time reduced large-deformation models and distributed contact for computer graphics and haptics. Ph.D. thesis. Carnegie Mellon University, Pittsburgh, PA, USA.

Barbič, J., James, D.L., 2005. Real-time subspace integration for St. Venant–Kirchhoff deformable models. ACM Trans. Graph. 24 (3), 982–990. http://dx.doi.org/10.1145/1073204.1073300.

Barbič, J., James, D.L., 2010. Subspace self-collision culling. ACM Trans. Graph. 29 (4), 81. http://dx.doi.org/10.1145/1778765.1778818.

Barbič, J., Zhao, Y., 2011. Real-time large-deformation substructuring. ACM Trans. Graph. 30 (4), 91. http://dx.doi.org/10.1145/2010324.1964986.

Bosman, J., Duriez, C., Cotin, S., 2013. Connective tissues simulation on GPU. In: Bender, J., Dequidt, J., Duriez, C., Zachmann, G. (Eds.), Virtual Reality Interaction and Physical Simulation. Eurographics Association, pp. 41–50. http://dblp.uni-trier.de/db/conf/vriphys/vriphys2013.html#BosmanDC13.

Brand, M., 2006. Fast low-rank modifications of the thin singular value decomposition. In: Special Issue on Large Scale Linear and Nonlinear Eigenvalue Problems. Linear Algebra Appl. 415 (1), 20–30. http://dx.doi.org/10.1016/j.laa.2005.07.021. http://www.sciencedirect.com/science/article/pii/S0024379505003812.

Choi, M.G., Ko, H.-S., 2005. Modal warping: real-time simulation of large rotational deformation and manipulation. IEEE Trans. Vis. Comput. Graph. 11 (1), 91–101. http://dx.doi.org/10.1109/TVCG.2005.13.

De Moor, B., Staar, J., Vandewalle, J., 1988. Oriented energy and oriented signal-to-signal ratio concepts in the analysis of vector sequences and time series. In: Deprettere, E.F. (Ed.), SVD and Signal Processing. North-Holland Publishing Co., Amsterdam, The Netherlands, pp. 209–232. http://dl.acm.org/citation.cfm?id=107302.107311.

Dick, C., Georgii, J., Westermann, R., 2011a. A real-time multigrid finite hexahedra method for elasticity simulation using CUDA. Simul. Model. Pract. Theory 19 (2), 801–816.

Dick, C., Georgii, J., Westermann, R., 2011b. A hexahedral multigrid approach for simulating cuts in deformable objects. IEEE Trans. Vis. Comput. Graph. 17 (11), 1663–1675.

Dollar, H.S., Gould, N.I.M., Schilders, W.H.A., Wathen, A.J., 2006a. Implicit-factorization preconditioning and iterative solvers for regularized saddle-point systems. SIAM J. Matrix Anal. Appl. 28 (1), 170–189. http://dx.doi.org/10.1137/05063427X.

Dollar, H.S., Gould, N., Wathen, A., 2006b. On implicit-factorization constraint preconditioners. In: Di Pillo, G., Roma, M. (Eds.), Large-Scale Nonlinear Optimization. In: Nonconvex Optim. Appl., vol. 83. Springer US, pp. 61–82.

Fierz, B., Spillmann, J., Aguinaga, I., Harders, M., 2012. Maintaining large time steps in explicit finite element simulations using shape matching. IEEE Trans. Vis. Comput. Graph. 18 (5), 717–728.

Frieze, A., Kannan, R., Vempala, S., 1998. Fast Monte-Carlo algorithms for finding low-rank approximations. In: Proceedings of the 39th Annual IEEE Symposium on Foundations of Computer Science, pp. 370–378.

Gu, M., Eisenstat, S.C., 1993. A stable and fast algorithm for updating the singular value decomposition. Tech. Rep. YALEU/DCS/RR-966. Yale University, New Haven, CT. citeseer.nj.nec.com/gu94stable.html.

Guo, X., Qin, H., 2005. Real-time meshless deformation. Comput. Animat. Virtual Worlds 16 (3–4), 189–200. http://dx.doi.org/10.1002/cav.98.

Hahn, F., Thomaszewski, B., Coros, S., Sumner, R.W., Cole, F., Meyer, M., DeRose, T., Gross, M., 2014. Subspace clothing simulation using adaptive bases. ACM Trans. Graph. 33 (4), 105. http://dx.doi.org/10.1145/2601097.2601160.

Hildebrandt, K., Schulz, C., Tycowicz, C.V., Polthier, K., 2011. Interactive surface modeling using modal analysis. ACM Trans. Graph. 30 (5), 119. http://dx.doi.org/10.1145/2019627.2019638.

Homescu, C., Petzold, L.R., Serban, R., 2005. Error estimation for reduced-order models of dynamical systems. SIAM J. Numer. Anal. 43 (4), 1693–1714. http://dx.doi.org/10.1137/040603541.

Huang, J., Liu, X., Bao, H., Guo, B., Shum, H.-Y., 2006a. An efficient large deformation method using domain decomposition. Comput. Graph. 30 (6), 927–935. http://dx.doi.org/10.1016/j.cag.2006.08.014. http://www.sciencedirect.com/science/article/pii/S0097849306001488.

Huang, J., Shi, X., Liu, X., Zhou, K., Wei, L.-Y., Teng, S.-H., Bao, H., Guo, B., Shum, H.-Y., 2006b. Subspace gradient domain mesh deformation. ACM Trans. Graph. 25 (3), 1126–1134. http://dx.doi.org/10.1145/1141911.1142003.

Jerabkova, L., Bousquet, G., Barbier, S., Faure, F., Allard, J., 2010. Volumetric modeling and interactive cutting of deformable bodies. Prog. Biophys. Mol. Biol. 103 (2–3), 217–224.

Kenney, C.S., Laub, A.J., 1994. Small-sample statistical condition estimates for general matrix functions. SIAM J. Sci. Comput. 15 (1), 36–61. http://dx.doi.org/10.1137/0915003.

Kerschen, G., Golinvalas, J.C., 2002. Physical interpretation of the proper orthogonal modes using the singular value decomposition. J. Sound Vib. 249 (5), 849–865.

Kim, T., James, D.L., 2009. Skipping steps in deformable simulation with online model reduction. ACM Trans. Graph. 28 (5), 123. http://dx.doi.org/10.1145/1618452.1618469.

Kim, T., James, D.L., 2012. Physics-based character skinning using multidomain subspace deformations. IEEE Trans. Vis. Comput. Graph. 18 (8), 1228–1240.

Krysl, P., Lall, S., Marsden, J., 2001. Dimensional model reduction in non-linear finite element dynamics of solids and structures. Int. J. Numer. Methods Eng. 51 (4), 479–504.

Li, S., Huang, J., de Goes, F., Jin, X., Bao, H., Desbrun, M., 2014a. Space–time editing of elastic motion through material optimization and reduction. ACM Trans. Graph. 33 (4), 108. http://dx.doi.org/10.1145/2601097.2601217.

Li, S., Zhao, Q., Wang, S., Hao, A., Qin, H., 2014b. Interactive deformation and cutting simulation directly using patient-specific volumetric images. Comput. Animat. Virtual Worlds 25 (2), 155–169.

Misztal, M.K., Bridson, R., Erleben, K., Bærentzen, J.A., Anton, F., 2010. Optimization-based fluid simulation on unstructured meshes. In: Virtual Reality Interaction and Physical Simulation, pp. 11–20.

Müller, M., Dorsey, J., McMillan, L., Jagnow, R., Cutler, B., 2002. Stable real-time deformations. In: Proceedings of the 2002 ACM SIGGRAPH/Eurographics Symposium on Computer Animation. ACM, pp. 49–54.

Pauly, M., Keiser, R., Adams, B., Dutré, P., Gross, M., Guibas, L.J., 2005. Meshless animation of fracturing solids. ACM Trans. Graph. 24 (3), 957–964. http://dx.doi.org/10.1145/1073204.1073296.

Rabczuk, T., Xiao, S., Sauer, M., 2006. Coupling of mesh-free methods with finite elements: basic concepts and test results. Commun. Numer. Methods Eng. 22 (10), 1031–1065.

Rathinam, M., Petzold, L.R., 2003. A new look at proper orthogonal decomposition. SIAM J. Numer. Anal. 41 (5), 1893–1925. http://dx.doi.org/10.1137/S0036142901389049.

Rees, T., Scott, J., 2014. The null-space method and its relationship with matrix factorizations for sparse saddle point systems. Tech. Report RAL-TR-2014-016. STFC Rutherford Appleton Laboratory.

Shabana, A.A., 2012. Theory of Vibration: Volume II: Discrete and Continuous Systems. Springer Science & Business Media.

Sifakis, E., Der, K.G., Fedkiw, R., 2007. Arbitrary cutting of deformable tetrahedralized objects. In: Proceedings of Eurographics Symposium on Computer Animation, pp. 73–80.

Teng, Y., Otaduy, M.A., Kim, T., 2014. Simulating articulated subspace self-contact. ACM Trans. Graph. 33 (4), 106. http://dx.doi.org/10.1145/2601097.2601181.

Toselli, A., Widlund, O., 2005. Domain Decomposition Methods: Algorithms and Theory, vol. 3. Springer.

von Tycowicz, C., Schulz, C., Seidel, H.-P., Hildebrandt, K., 2013. An efficient construction of reduced deformable objects. ACM Trans. Graph. 32 (6), 213. http://dx.doi.org/10.1145/2508363.2508392.

Wang, R., Yang, Z., Liu, L., Deng, J., Chen, F., 2014. Decoupling noise and features via weighted l1-analysis compressed sensing. ACM Trans. Graph. 33 (2), 18. http://dx.doi.org/10.1145/2557449.

Wang, Y., Jacobson, A., Barbič, J., Kavan, L., 2015. Linear subspace design for real-time shape deformation. ACM Trans. Graph. 34 (4), 57. http://dx.doi.org/10.1145/2766952.

Wang, B., Wu, L., Yin, K., Ascher, U., Liu, L., Huang, H., 2015. Deformation capture and modeling of soft objects. ACM Trans. Graph. 34 (4), 94. http://dx.doi.org/10.1145/2766911.

Wicke, M., Botsch, M., Gross, M., 2007. A finite element method on convex polyhedra. Comput. Graph. Forum 26 (3), 355–364.

Wicke, M., Stanton, M., Treuille, A., 2009. Modular bases for fluid dynamics. ACM Trans. Graph. 28 (3), 39. http://dx.doi.org/10.1145/1531326.1531345.

Xia, Q., Li, S., Qin, H., Hao, A., 2015. Modal space subdivision for physically-plausible 4D shape sequence completion from sparse samples. In: Pacific Graphics Short Papers. The Eurographics Association, pp. 19–24.

Xu, H., Zhao, Y., Barbič, J., 2014. Implicit multibody penalty-baseddistributed contact. IEEE Trans. Vis. Comput. Graph. 20 (9), 1266–1279.

Yang, Y., Rong, G., Torres, L., Guo, X., 2010. Real-time hybrid solid simulation: spectral unification of deformable and rigid materials. Comput. Animat. Virtual Worlds 21 (3–4), 151–159. http://dx.doi.org/10.1002/cav.373.

Yang, Y., Guo, X., Vick, J., Torres, L., Campbell, T., 2013a. Physics-based deformable tongue visualization. IEEE Trans. Vis. Comput. Graph. 19 (5), 811–823. http://dx.doi.org/10.1109/TVCG.2012.174.

Yang, Y., Xu, W., Guo, X., Zhou, K., Guo, B., 2013b. Boundary-aware multidomain subspace deformation. IEEE Trans. Vis. Comput. Graph. 19 (10), 1633–1645. http://dx.doi.org/10.1109/TVCG.2013.12.

Yang, C., Li, S., Wang, L., Hao, A., Qin, H., 2014. Real-time physical deformation and cutting of heterogeneous objects via hybrid coupling of meshless approach and finite element method. Comput. Animat. Virtual Worlds 25 (3–4), 421–433. http://dx.doi.org/10.1002/cav.1594.