Boundary-Aware Multidomain Subspace Deformation

Yin Yang, Weiwei Xu, Xiaohu Guo, *Member*, *IEEE*,
Kun Zhou, and Baining Guo, *Fellow*, *IEEE*

Abstract—In this paper, we propose a novel framework for multidomain subspace deformation using node-wise corotational elasticity. With the proper construction of subspaces based on the knowledge of the boundary deformation, we can use the Lagrange multiplier technique to impose coupling constraints at the boundary without overconstraining. In our deformation algorithm, the number of constraint equations to couple two neighboring domains is not related to the number of the nodes on the boundary but is the same as the number of the selected boundary deformation modes. The crack artifact is not present in our simulation result, and the domain decomposition with loops can be easily handled. Experimental results show that the single-core implementation of our algorithm can achieve real-time performance in simulating deformable objects with around quarter million tetrahedral elements.

Index Terms—Model reduction, domain decomposition, FEM, deformable model

1 Introduction

Model reduction is an important technique for accelerating the physics-based simulation of deformable objects. The basic idea is to project the high-dimensional equation of motion to a carefully chosen low-dimensional subspace to construct a reduced model. Traditional global subspace methods, however, cannot handle the object's local deformation behaviors well unless a large number of basis vectors are used, which in turn would cancel out the benefit of acceleration. Multidomain subspace techniques provide a good solution to this problem by partitioning the deformable object into multiple domains and constructing reduced models for each domain independently. The advantages are twofold. First, local deformation behaviors can be well captured with a modest number of basis vectors for each domain. Second, local simulation of each domain can more flexibly handle deformable objects of complex, semantic geometries. Users can easily specify the number of degrees of freedom (DOFs) and the types of bases for each domain to accommodate hybrid simulation results.

The key challenge in applying multidomain subspace techniques to deformable object simulation is the seamless coupling of the neighboring domains at their boundary interface. Recently, two coupling methods have been developed for multidomain subspace deformations using the reduced nonlinear St.Venant-Kirchhoff (StVK) deformable model. Kim and James [1] proposed to couple the domains using penalty (spring) forces for character skinning. With an increasing number of basis vectors in the reduced model, the cracks become invisible visually. However, this method requires predetermined motions of local frames for each domain, and the length of the time step is usually small due to the possible large penalty forces at the boundary interface. Another multidomain subspace deformation method in [2] relies on shape matching and mass lumping at the boundary interfaces to handle the coupling issue. Nevertheless, cracks might still be visible if the deformation goes large and this problem can be remedied by geometric blending operations at the post-simulation stage. This method works well for the domain decomposition with tree-like hierarchies and small domain interfaces. However, seamless coupling of multidomain subspace deformations with arbitrary domain decomposition remains a technical challenge.

The goal of this paper is to develop a seamless coupling method for multidomain subspace deformation in the framework of nodewise corotational elasticity. To this end, we propose a boundary-aware mode construction method that characterizes the deformation subspace of each domain through its boundary deformations. Instead of posing coupling constraints on boundary node pairs, which can easily lead to overconstraints for large-scale meshes, our algorithm formulates these constraints with *rigid and soft boundary modes*, which form a compact representation of boundary deformations. In this way, we can apply the *Lagrange multiplier* technique to solve the boundary coupling constraints without overconstraining. The boundary modes are computed by solving the static equilibrium equations as in *component mode synthesis* (CMS) [3]. The large deformation of each domain is simulated using the *modal warping* technique [4].

With our algorithm, cracks are avoided in the simulation result, and the domain decomposition with loops can be

- *Y. Yang and X. Guo are with the Department of Computer Science, The University of Texas at Dallas, 800 W. Campbell Road, Richardson, TX 75080-3021. E-mail: {yinyang, xguo}@utdallas.edu.*
- *W. Xu is with Hangzhou Normal University, Hangzhou, China, and the State Key Lab of CAD and CG, Zhejiang University, 866 Yuhangtang Road, Hangzhou 310058, China. E-mail: weiwei.xu.g@gmail.com.*
- *K. Zhou is with the State Key Lab of CAD and CG, Zhejiang University, 866 Yuhangtang Road, Hangzhou 310058, China. E-mail: kunzhou@acm.org.*
- *B. Guo is with Microsoft Research Asia, Haidian District, Building 2, No. 5 Dan Ling Street, Beijing 100080, China. E-mail: bainguo@microsoft.com*

Manuscript received 26 Aug. 2012; revised 15 Jan. 2013; accepted 24 Jan. 2013; published online 13 Feb. 2013.
Recommended for acceptance by H.-S. Ko.
For information on obtaining reprints of this article, please send e-mail to: tvcg@computer.org, and reference IEEECS Log Number TVCG-2012-08-0170.
Digital Object Identifier no. 10.1109/TVCG.2013.12.

naturally handled as well. We show the capability of our multidomain subspace deformation algorithm by simulating a variety of large-scale deformable objects. User manipulation, such as directly constraining the node position or the rotation of the domain boundary, is also supported in our algorithm to ease the animation production.

The rest of the paper is organized as follows: Section 2 reviews the related work. Section 3 describes how to construct deformation modes from the boundary deformation. The deformation algorithm and direct manipulation methods are described in Sections 4 and 5. Experimental results and limitations are discussed in Section 6. Finally, we conclude in Section 7.

2 Related Work

Physics-based simulation of deformable objects has been an active research topic in computer graphics since 1980s. A comprehensive survey can be found in [5], [6]. Terzopoulos et al. [7], [8] proposed a fundamental framework to simulate 3D deformations based on the theory of elasticity. The ordinary differential equations describing the dynamics of deformable models can be numerically solved with the finite-element method (FEM) and generate realistic deformations. High computational cost is a drawback of the FEM especially for the finite-element meshes of large size. To make deformable models more practical for interactive applications, numerous works have been proposed and have greatly advanced related areas. Multiresolution [9] or adaptive simulation [10], [11] uses hierarchical or adaptive bases of the deformation to accelerate the computation. These types of techniques use the high-level bases to represent general deformations and the low-level or refined bases for more detailed deformations when necessary. Similarly, embedded mesh, mesh coarsening or controlling lattice [12], [13], [14], [15], [16] handle the deformation with auxiliary coarsened grids.

Corotational elasticity and its variations are widely used in computer graphics for fast simulation of large-scale deformations. It was first introduced by Müller et al. [17], [18] via *stiffness warping* and widely adopted in various applications [19], [20], [21], [22]. It has been extended to thin shell [23] and meshless [24] simulation. Warp-canceling corotation has recently been proposed to improve the approximation accuracy of stiffness warping to element-wise corotational elasticity [25]. Although the stiffness matrix can be kept constant in nodewise corotational methods as in [4], [17], it is still hard to directly apply it to the real-time simulation of a large-scale mesh as in our case, since the prefactorization of the large-scale stiffness matrix might take hours and sometimes not plausible on a desktop PC due to the memory limitations.

Another series of contributions are based on *modal analysis* (MA), which is a well-developed technique widely used in engineering areas. The MA utilizes the eigendecomposition to project the full deformation space to the vibrations of different frequencies [26], [27], [28], [29], [30]. The eigenvectors associated with low vibration energies are discarded as they are believed to have less contribution to the final deformation. To handle rotational deformations, Choi and Ko [4] proposed a technique called modal warping. The *curl* of the linear displacement field is used to estimate nodal rotation and warp the distortion induced by using linear modal bases. Alternatively, nonlinear deformation can also be captured with *modal derivatives* [31], which extends the linear deformation subspace to the parabolic subspace. On-the-fly subspace construction [32] provides another direction to accelerate the simulation. The subspace bases are the recently simulated displacements and vary during the simulation with the extra cost of examining residual error periodically. Geometry-based shape matching [33], [34] provides an alternative for fast computation of soft 3D volumes. Unfortunately, shape matching is not able to incorporate the material properties intuitively, and increasing the number of DOFs does not necessarily lead to a more accurate simulation.

Local subspace methods such as [1], [2], [35] can be categorized as *domain decomposition methods* (DDMs). The input mesh is decomposed into mutually disjoint domains. The model reduction is then applied to each domain. Because the domains are always connected and interacted with the neighbors, the domain's deformation is not unconstrained like a single deformable object. The key technical challenge of local subspace methods is to impose domain coupling with low costs when there exist a large number of boundary DOFs. Such interdomains constraints should be considered during the selection of deformation modes to construct a reasonable local subspace at each domain. Huang et al. [35] use node-pair position constraints (PCs) for domain coupling and a precomputed force-displacement matrix to accelerate matrix-vector multiplication. Barbič and Zhao [2] adopt a passive interdomain deformation mechanism with rigid interface fitting when the parent domain deforms. This framework works well for domain decomposition with tree-like hierarchies, and the precomputation time can be significantly reduced when a large number of domains are of the same geometry. However, due to the rigid interface assumption, it is not suitable for domains connected through large soft interfaces. Kim and James [1] use spring forces to avoid the stiffening induced by the subspace coupling. Meanwhile, additional damping forces are also provided to suppress boundary jigging. This coupling strategy is effective and simple to implement. It targets character skinning where the motion of the local frame of each domain must be predetermined. These DDM techniques unfortunately do not fully incorporate the boundary condition during the construction of the local subspace. As a result, the displacements at the duplicated domain interfaces may experience slight incompatibility and crack or interpenetration could appear. However, this problem can be easily fixed by a simple blending treatment at postsimulation stage as in [2].

Our algorithm complements existing methods. It is inspired by CMS [3] to construct the modes at the static equilibrium while we perform model reduction geometrically at the boundary DOFs to avoid the oversized boundary problem. The domain coupling constraint can be directly imposed on the reduced coordinates without the overconstraining problem. Boundary modes constitute the *static deformation* of the domains. The extra internal vibrational deformation is also captured with interior

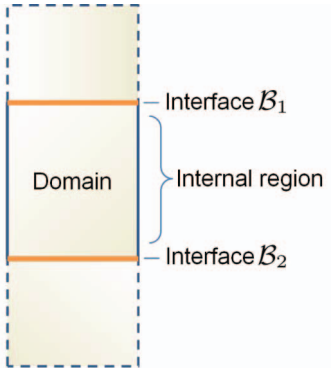


Fig. 1. A three-domain bar.

modes that do not participate in the domain coupling. Domains are always exactly coupled, and the postsimulation processing is avoided.

3 Boundary-Aware Mode Construction

The input finite-element mesh (tetrahedral mesh in this paper) is called a *host mesh* and is to be divided into multiple submeshes or *domains*. We enforce the face connectivity between a pair of neighboring domains such that they must share at least one triangle face if considered connected.[1] The *modes* are simply the precomputed domain displacements that serve as local subspace bases. Our boundary-aware mode construction characterizes the deformation subspace of each domain through its boundary deformation. Unlike the classic CMS method [3] that provides complete boundary freedom by assigning each boundary DOF an individual mode, we reduce the boundary freedom with the use of geometrically constructed bases. After that, the corresponding boundary modes of the domain are computed through solving the static equilibrium with linear elasticity. Besides boundary modes, internal modes are also incorporated for enriched local deformation.

3.1 DOF Classification

A domain with n nodes has total of $3n$ DOFs as each node has independent freedoms along x, y, and z. The DOFs that are shared with neighboring domains are called *boundary* DOFs denoted by set $\mathcal{B}$. All the other DOFs are called *internal* DOFs (even they may be located at the surface of the mesh) and are denoted by set $\mathcal{I}$. A DOF is either in $\mathcal{B}$ or in $\mathcal{I}$ and cannot be in $\mathcal{B}$ and $\mathcal{I}$ at the same time. If a domain has k neighboring domains, $\mathcal{B}$ is further grouped into k subsets, $\mathcal{B}_1, \mathcal{B}_2, \ldots, \mathcal{B}_k$. Each subset holds DOFs that are shared with the same neighbor and is called an *interface*. Fig. 1 shows an illustrative 2D example of a bar model. The middle domain has the boundary with two interfaces. Let $\mathbf{\Phi}$ denote the modes of a single domain. According to the DOF types, it can be decomposed into two parts $\mathbf{\Phi} = [\mathbf{\Phi}_{\mathcal{I}}^{\top} \mid \mathbf{\Phi}_{\mathcal{B}}^{\top}]^{\top}$. As illustrated in Table 1, in the following sections where the detailed mode computation is explained, we use subscript to denote the DOF type and superscript for mode type.

1. The face-connectivity avoids the singularity of the substiffness corresponding to internal DOFs, during the mode computation. Because the interface's displacement is always constrained during the computation of domain modes, as long as interface is able to determine the rigid body motion of the domain, the internal submatrix is always nonsingular.

TABLE 1
Matrix Notation Used in the Mode Computation

$\mathbf{\Phi}$	Mode matrix
$\mathbf{\Phi}^R$	Rigid (R) boundary mode matrix (Sec. 3.2)
$\mathbf{\Phi}^S$	Soft (S) boundary mode matrix (Sec. 3.2)
$\mathbf{\Phi}^N$	Normal (N) mode matrix (Sec. 3.3)
$\mathbf{\Phi}^I$	Inertia (I) mode matrix (Sec. 3.3)
$\mathbf{\Phi}_{\mathcal{I}}$	Mode submatrix corresponding to internal ($\mathcal{I}$) DOFs
$\mathbf{\Phi}_{\mathcal{B}}$	Mode submatrix corresponding to boundary ($\mathcal{B}$) DOFs
$\mathbf{\Phi}_{\mathcal{B}_k}$	Mode submatrix corresponding to DOFs on interface $\mathcal{B}_k$

3.2 Boundary Modes

The boundary modes are the domain's displacement at static equilibrium status when it is imposed to external boundary displacements (e.g., displacement at $\mathcal{B}$). Such equilibrium can be expressed in the form of $\mathbf{Ku} = \mathbf{f}$, where the domain's stiffness matrix is denoted by $\mathbf{K}$. $\mathbf{u}$ and $\mathbf{f}$ represent the displacement and forces of the domain, respectively. The unknown responding deformation at $\mathcal{I}$ is computed by solving the following equilibrium:

$$\begin{bmatrix} \mathbf{K}_{\mathcal{I}\mathcal{I}} & \mathbf{K}_{\mathcal{I}\mathcal{B}_1} & \cdots & \mathbf{K}_{\mathcal{I}\mathcal{B}_k} \\ \mathbf{K}_{\mathcal{B}_1\mathcal{I}} & \mathbf{K}_{\mathcal{B}_1\mathcal{B}_1} & \cdots & \mathbf{K}_{\mathcal{B}_1\mathcal{B}_k} \\ & & \vdots & \\ \mathbf{K}_{\mathcal{B}_k\mathcal{I}} & \mathbf{K}_{\mathcal{B}_k\mathcal{B}_1} & \cdots & \mathbf{K}_{\mathcal{B}_k\mathcal{B}_k} \end{bmatrix} \begin{bmatrix} \mathbf{\Phi}_{\mathcal{I}} \\ \mathbf{\Phi}_{\mathcal{B}} \end{bmatrix} = \begin{bmatrix} \mathbf{0} \\ \mathbf{F}_{\mathcal{B}} \end{bmatrix}. \quad (1)$$

The domain's stiffness matrix is grouped and ordered corresponding to the classification of DOFs (e.g., boundary or internal DOFs), and it is constant under linear elasticity. $\mathbf{f}_{\mathcal{B}}$ is the external force applied at the boundary to drive the boundary displacement. If a domain has more than one interface (e.g., $k > 1$), the modes are computed independently for each interface such that the imposed boundary displacement has nonzero values only at one interface each time, and all the other interfaces are fixed. Consequently, let $\mathbf{\Phi}_{\mathcal{B}_i}$ be the nonzero interface displacement for $\mathcal{B}_i$, and $\mathbf{\Phi}_{\mathcal{B}}$ has the block-diagonal-like form as

$$\mathbf{\Phi}_{\mathcal{B}} = \begin{bmatrix} \mathbf{\Phi}_{\mathcal{B}_1} & & & \\ & \mathbf{\Phi}_{\mathcal{B}_2} & & \\ & & \ddots & \\ & & & \mathbf{\Phi}_{\mathcal{B}_k} \end{bmatrix}.$$

Rigid boundary modes: The rigid boundary modes, denoted by $\mathbf{\Phi}^R$, represent the domain's deformation when its interfaces only have rigid body motion. Correspondingly, $\mathbf{\Phi}^R_{\mathcal{B}_i}$ has six columns, where the first three columns in $\mathbf{\Phi}^R_{\mathcal{B}_i}$ represent three linearly independent translational displacement fields over $\mathcal{B}_i$. A natural choice is to use unit displacements along each axis as $\mathbf{x} = [1,0,0]^{\top}$, $\mathbf{y} = [0,1,0]^{\top}$, and $\mathbf{z} = [0,0,1]^{\top}$ for each interface node. The other three columns are the rotational modes constructed by assigning each interface node the displacements along the tangent directions of the rotations around three linearly independent rotation axes (i.e., $\mathbf{x}$, $\mathbf{y}$, and $\mathbf{z}$). We set the centroid of the interface $\mathbf{c}_i$ as the rotation pivot. Such tangential rotation modes are only able to represent infinitesimal rotations for linear elasticity. We later show

in Section 5 that they can also represent large rotations with the help of modal warping. For an interface node p whose rest position is $\mathbf{p}$, the corresponding 3×6 block in $\mathbf{\Phi}^R_{\mathcal{B}_i}$ has the structure like

$$\mathbf{\Phi}^R_{p,p\in\mathcal{B}_i} = [\mathbf{x}|\mathbf{y}|\mathbf{z}|(\mathbf{p}-\mathbf{c}_i)\times\mathbf{x}|(\mathbf{p}-\mathbf{c}_i)\times\mathbf{y}|(\mathbf{p}-\mathbf{c}_i)\times\mathbf{z}]. \quad (2)$$

By expanding the first line of (1) (which corresponds to the internal DOFs of the domain) and substituting $\mathbf{\Phi}_{\mathcal{B}_i}$ with $\mathbf{\Phi}^R_{\mathcal{B}_i}$, $\mathbf{\Phi}^R_{\mathcal{I}}$ can be computed through

$$\mathbf{\Phi}^R_{\mathcal{I}} = -\mathbf{K}^{-1}_{\mathcal{I}\mathcal{I}}\left[\mathbf{K}_{\mathcal{I}\mathcal{B}_1}\mathbf{\Phi}^R_{\mathcal{B}_1}\middle|\mathbf{K}_{\mathcal{I}\mathcal{B}_2}\mathbf{\Phi}^R_{\mathcal{B}_2}\middle|\ldots\middle|\mathbf{K}_{\mathcal{I}\mathcal{B}_k}\mathbf{\Phi}^R_{\mathcal{B}_k}\right]. \quad (3)$$

Stacking $\mathbf{\Phi}^R_{\mathcal{I}}$ and $\mathbf{\Phi}^R_{\mathcal{B}}$ yields the rigid boundary modes:

$$\mathbf{\Phi}^R = \begin{bmatrix}\mathbf{\Phi}^R_{\mathcal{I}}\\ \mathbf{\Phi}^R_{\mathcal{B}}\end{bmatrix}. \quad (4)$$

The number of rigid boundary modes is $6k$, which only depends on the number of neighbor domains. The rigid boundary modes represent the general deformation of the domain, and they are always chosen as the domain's subspace bases in our implementation.

Soft boundary modes: Soft boundary modes complement the rigid boundary modes by incorporating the deformations that are induced by nonrigid interface displacements. Assigning each boundary DOF an individual boundary mode will lead to a very big system if there are many boundary nodes. This compromises the original purpose of using model reduction. Instead, the soft boundary modes are designed to only capture the most notable interface geometry, which is similar to regular MA of deformable model. For each interface, we compute its *manifold harmonic bases* [36] (denoted by $\mathbf{H}$), by solving the generalized eigen problem of the *Laplacian matrix* of the interface: $-\mathbf{QH} = \mathbf{\Lambda BH}$, where $\mathbf{Q}, \mathbf{B}$ are square symmetric matrices with size $n_{\mathcal{B}_i}$, the number of nodes on interface $\mathcal{B}_i$. Elements in $\mathbf{Q}, \mathbf{B}$ are computed with

$$\begin{cases} Q_{a,b} = \dfrac{cot(\beta_{a,b}) + cot(\beta'_{a,b})}{2} \\ Q_{a,a} = \sum_b Q_{a,b} \end{cases} \text{ and } \begin{cases} B_{a,b} = \dfrac{|t| + |t'|}{12} \\ B_{a,a} = \dfrac{\sum_{t\in St(a)}|t|}{6}, \end{cases}$$

where t and t' are two triangles that share the edge (a, b) with area $|t|$ and $|t'|$. $\beta_{a,b}$ and $\beta'_{a,b}$ denote the two angles opposite to the edge (a, b) in t and t'. $St(a)$ stands for the set of triangles incident to a.

Each Harmonics basis is a vector of size $n_{\mathcal{B}_i}$. It is spanned to represent the interface displacements in x, y, and z directions, respectively, as

$$\mathbf{\Phi}^S_{\mathcal{B}_i} = \mathbf{H}\otimes\mathbf{I}, \quad (5)$$

where $\otimes$ denotes the *Kronecker product* and $\mathbf{I}\in\mathbb{R}^{3\times3}$ is the identity matrix. The complete set of harmonic bases spans the full space of the interface displacement, and the harmonic bases corresponding to the large eigenvalues can be discarded to reduce the number of interface freedoms because they represent the detailed geometry features of the interface. Such property of harmonics is also utilized for mesh deformation with model reduction [37], [38].

Redundant bases could be induced if an interface displacement represented with harmonic bases is close to rigid. Therefore, for a given harmonic-spanned interface displacement $\phi^S_{\mathcal{B}_i}\in\mathbb{R}^{3n_{\mathcal{B}_i}\times1}$, we check the residual error of projecting $\phi^S_{\mathcal{B}_i}$ onto the existing rigid interface subspace spanned by $\mathbf{\Phi}^R_{\mathcal{B}_i}$:

$$e = \frac{\|[\mathbf{\Phi}^R_{\mathcal{B}_i}]([\mathbf{\Phi}^R_{\mathcal{B}_i}]^\top[\mathbf{\Phi}^R_{\mathcal{B}_i}])^{-1}[\mathbf{\Phi}^R_{\mathcal{B}_i}]^\top\phi^S_{\mathcal{B}_i} - \phi^S_{\mathcal{B}_i}\|}{\|\ \phi^S_{\mathcal{B}_i}\ \|}. \quad (6)$$

If the error e is small enough (e.g., <1 percent in our implementation), it indicates that a certain combination of the existing rigid interface displacements is able to represent $\phi^S_{\mathcal{B}_i}$ well, and $\phi^S_{\mathcal{B}_i}$ is opt out. It is noteworthy that the first harmonic basis has zero eigenvalue. The corresponding three bases computed by (5) are identical to the three translational modes. As a result, they are always excluded.

The unknown internal displacement $\mathbf{\Phi}^S_{\mathcal{I}}$ with respect to the interface harmonics can be computed similar to (3):

$$\mathbf{\Phi}^S_{\mathcal{I}} = -\mathbf{K}^{-1}_{\mathcal{I}\mathcal{I}}\left[\mathbf{K}_{\mathcal{I}\mathcal{B}_1}\mathbf{\Phi}^S_{\mathcal{B}_1}\middle|\mathbf{K}_{\mathcal{I}\mathcal{B}_2}\mathbf{\Phi}^S_{\mathcal{B}_2}\middle|\ldots\middle|\mathbf{K}_{\mathcal{I}\mathcal{B}_k}\mathbf{\Phi}^S_{\mathcal{B}_k}\right]. \quad (7)$$

By stacking boundary and internal parts computed in (5) and (7), the soft boundary modes are assembled as

$$\mathbf{\Phi}^S = \begin{bmatrix}\mathbf{\Phi}^S_{\mathcal{I}}\\ \mathbf{\Phi}^S_{\mathcal{B}}\end{bmatrix}. \quad (8)$$

3.3 Internal Vibrational Modes

Boundary modes are suitable for capturing static deformation driven by the unaccelerated boundary displacement. Enriched deformation at internal DOFs is necessary when the *inertia force* associated with boundary displacement is considered. Following an idea similar to [29], we include additional internal vibrational modes in the framework for detailed local deformation at internal DOFs ($\mathcal{I}$). Internal vibrational modes always have zero values at the boundary DOFs. Therefore, they are not responsible for domain coupling.

Normal modes: A natural choice for computing the vibrational mode is via solving the generalized eigen problem of internal DOFs (so, the modes look like fixed at the boundary):

$$\begin{cases} (\mathbf{K}_{\mathcal{I}\mathcal{I}} - \mathbf{\Lambda M}_{\mathcal{I}\mathcal{I}})\mathbf{\Phi}^N_{\mathcal{I}} &= 0 \\ \mathbf{\Phi}^N_{\mathcal{B}} &= 0. \end{cases} \quad (9)$$

Equation (9) is actually performing the linear mode analysis (LMA) over $\mathcal{I}$. It can be understood as choosing the deformation modes that increase the system's energy least [39] with the additional imposed constraints that the boundary DOFs are fixed.

Inertia modes: Using normal modes is a good choice when there is no clue about what internal deformation is going to happen. However, in multidomain deformation, the internal deformation occurs most likely as the consequence of the accelerated boundary movement. To better account for such boundary-triggered internal deformation, we precompute the corresponding internal deformation based on the knowledge of the boundary displacements (which are defined with boundary modes), and the resulting deformation modes are called inertia modes. They are called so

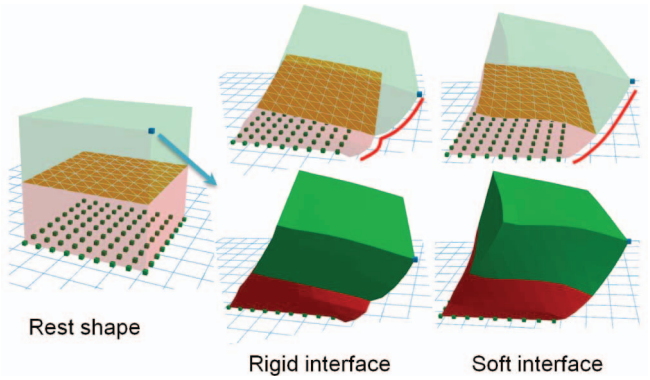


Fig. 2. The assumption of rigid interface [2] induces artifact when the interface is wide and the neighbor domains are soft. Using our soft modes yields smooth and natural deformation. Same number of modes (30 per domain) are used in the comparison while five soft boundary modes are used to capture the boundary deformation. The external force is shown as an arrow.

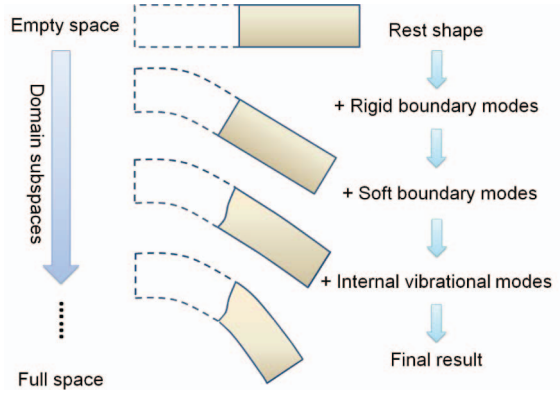


Fig. 3. Subspaces spanned by different types of modes constitute the layered deformation. Final deformation of the domain can be understood as the superposition of deformations from each subspace. Complete set of modes span the full space.

because inertia modes are computed by solving the equilibrium with inertia forces that correspond to the boundary acceleration:

$$\begin{bmatrix} \mathbf{K}_{\mathcal{I}\mathcal{I}} & \mathbf{K}_{\mathcal{I}\mathcal{B}} \\ \mathbf{K}_{\mathcal{B}\mathcal{I}} & \mathbf{K}_{\mathcal{B}\mathcal{B}} \end{bmatrix} \begin{bmatrix} \mathbf{\Phi}^I_{\mathcal{I}(1)} \\ \mathbf{0} \end{bmatrix} = \mathbf{M}\mathbf{\Phi}^R + \begin{bmatrix} \mathbf{0} \\ \mathbf{F}^I_{\mathcal{B}} \end{bmatrix}, \tag{10}$$

where $\mathbf{\Phi}^R$ represents the acceleration along the directions of the rigid boundary modes. Soft boundary modes can also be included in (10). However, from our experience, the inertia forces associated with soft boundary modes are often much smaller than the forces associated with rigid boundary modes. So, they are discarded for model reduction. The first order of inertia modes is computed as

$$\mathbf{\Phi}^I_{(1)} = \begin{bmatrix} \mathbf{\Phi}^I_{\mathcal{I}(1)} \\ \mathbf{0} \end{bmatrix} = \begin{bmatrix} \mathbf{K}^{-1}_{\mathcal{I}\mathcal{I}}\mathbf{M}_{\mathcal{I}\mathcal{I}}\mathbf{\Phi}^R_{\mathcal{I}} \\ \mathbf{0} \end{bmatrix}. \tag{11}$$

Similarly, if the acceleration associated with displacement $\mathbf{\Phi}^I_{(1)}$ is not neglected, another set of deformation can be computed with the same form of equilibrium as in (10). Successive blocks of higher order inertia modes are built with following recurrence relationship:

$$\mathbf{\Phi}^I_{(k+1)} = \begin{bmatrix} \mathbf{\Phi}^I_{\mathcal{I}(k+1)} \\ \mathbf{0} \end{bmatrix} = \begin{bmatrix} \mathbf{K}^{-1}_{\mathcal{I}\mathcal{I}}\mathbf{M}_{\mathcal{I}\mathcal{I}}\mathbf{\Phi}^I_{\mathcal{I}(k)} \\ \mathbf{0} \end{bmatrix}. \tag{12}$$

When the size of inertia subspace is close to full space size, the inertia modes could bring redundancy to the subspace bases. However, high-order inertia deformation has a much less contribution to the final internal deformation. Accordingly, the high-order inertia modes are discarded. For the sake of stability, one may apply a mass-based Gram-Schmidt orthonormalization over the inertia modes after each block iteration. Nevertheless, in our experiment, the system is stable even without mass orthonormalization.

Another advantage of inertia mode is its faster precomputation: The inverse of $\mathbf{K}_{\mathcal{I}\mathcal{I}}$ is shared in the computation of boundary modes (e.g., (3) and (7)). In fact, inertia mode $\mathbf{\Phi}^I_{\mathcal{I}}$ spans a *block Krylov subspace*:

$$\mathbf{\Phi}^I_{\mathcal{I}} = \left[\mathbf{A}\mathbf{\Phi}^R_{\mathcal{I}} \middle| \mathbf{A}^2\mathbf{\Phi}^R_{\mathcal{I}} \middle| \mathbf{A}^3\mathbf{\Phi}^R_{\mathcal{I}} \middle| \dots\right], \tag{13}$$

where $\mathbf{A} = \mathbf{K}^{-1}_{\mathcal{I}\mathcal{I}}\mathbf{M}_{\mathcal{I}\mathcal{I}}$. Krylov subspace is a well-known numerical method to compute the generalized eigen problem as defined in (9). We start the power iteration from the rigid boundary modes. From this point of view, inertia modes can be understood as a tuned version of normal modes as the inertia forces are preknown and boundary driven. In practice, we choose to use inertia modes instead of normal modes because they can be computed more efficiently than normal modes.

3.4 Discussion

Rigid interface versus soft interface: When the interface is small, a rigid body motion can well approximate its displacement. In this case, it is reasonable to make the assumption of the interface rigidity as in [2]. Such approximation can be achieved in our framework by only adopting rigid boundary modes and internal vibrational modes. However, in some situations where domains share broad and flexible interfaces, such assumption of interface rigidity could induce visual artifacts, because pure rigid interface is not sufficient for propagating enough deformation across domains. As shown in Fig. 2, the rigid interface fitting used in [2] leads to the discontinuous shape, while soft boundary modes generate much smoother deformation across the host mesh, and the sharp edges of the cube remain continuous. It may be possible to use higher order blending to smooth the deformation, but it requires extra postsimulation computation like moving least square embedding [40].

Layered subspaces: In our boundary-aware mode construction framework, the local domain's deformation is organized in layers. For each domain, we can consider its final deformation as the superposition of three subspace deformations as shown in Fig. 3. First, we slowly move the interface without changing its shape. Necessary domain deformation is generated to keep the domain coupled with its neighbors. This portion of deformation is represented with rigid boundary modes. After that, some nonrigid interface displacements are further generated to better accommodate the deformed boundary with soft boundary modes. Finally, internal vibrational modes capture additional local internal deformation. Complete set of boundary modes and internal vibrational modes

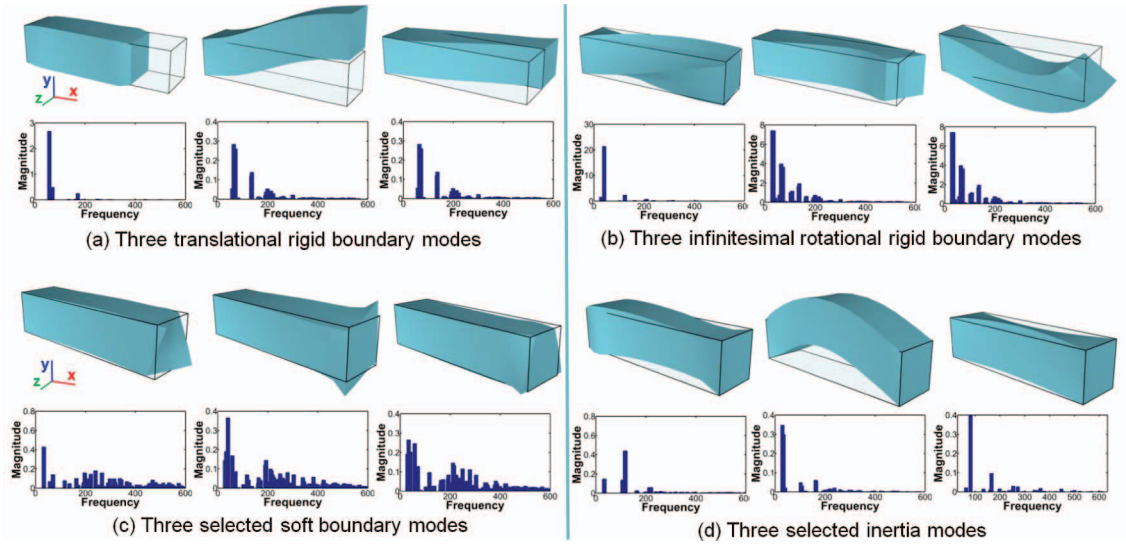


Fig. 4. The shapes of each type of warped modes associated with right interface of the domain. The left interface is assigned with zero displacements.

constitute the full space of the domain's deformation. Low-rank[2] boundary modes capture dominate interface displacements and low-rank internal vibrational modes describe majority internal vibrations with fixed boundary condition. From this figure, we can see that our boundary-aware mode construction strategy carefully performs model reduction on boundary modes and internal vibrational modes *separately*. By doing so, neighbor domains always have consistent low-dimensional interface displacement that is purely decided by the geometry feature of the interface (via computing its harmonic bases and rigid motion). We will see in Section 4.2 that such mechanism facilitates the subspace domain coupling so that the interface constraint can be directly enforced over the generalized reduced coordinate.

Our method versus free vibrational LMA: LMA provides the most natural deformation bases of the unconstrained vibration space in the case that the domain's deformation is completely unknown [39]. It is done by solving a generalized eigen problem ($\mathbf{K\Phi} = \mathbf{\Lambda M\Phi}$) defined over the entire domain [26]. Many model reduction techniques construct subspaces based on this method [4], [31], [41]. However, simply choosing the low-frequency LMA modes may not be the best solution for the multidomain deformable object, because domains are always coupled and not simple unconstrained free deformable bodies. Boundary modes are specially designed for the subspace domain coupling and, in general, they are not always the domain's vibrations of lowest frequency. In Fig. 4, we plot the frequency distribution of three types of modes mentioned above over the spectrum of the domain's free vibration. The y-axis in the plots is the projection of the modes onto the LMA bases of different frequency. We can see in the figure that the rigid boundary modes have more low-frequency components. However, some higher frequency components still exist. On the other hand, soft boundary modes have a wider distributed spectrum, which means that they have more high-frequency vibrational components. Similarly, internal vibrational modes have fixed boundary. As a result, they also have wide distribution over the free vibrational spectrum.

The boundary modes (both rigid and soft) are computed for each interface independently. That is, when imposing a certain displacement to an interface, we keep other domain interfaces fixed. This strategy has two advantages: 1) it guarantees that the subsets of boundary modes corresponding to different interfaces are linearly independent to each other; and 2) the displacement of an interface of the domain is determined only by the modes associated with the interface. The internal vibrational modes have vanishing values at the interfaces and only contribute to the deformation at $\mathcal{I}$. Thus, they are always linearly independent to the boundary modes and do not participate in the domain coupling.

4 Deformation Algorithm

In this section, we briefly describe the reduced Euler-Lagrange formulation for multidomain deformable bodies and the subspace domain coupling. Then, we introduce how to generalize the reduced nodewise corotational elasticity [4] for multidomain deformation.

4.1 Equation of Motion

For a single domain i, the Euler-Lagrange equation on the reduced coordinate $[\mathbf{q}]^i$ can be written as

$$[\mathbf{M}_q]^i[\ddot{\mathbf{q}}]^i + [\mathbf{C}_q]^i[\dot{\mathbf{q}}]^i + [\mathbf{K}_q]^i[\mathbf{q}]^i = [\mathbf{f}_q]^i, \tag{14}$$

where $[\mathbf{M}_q]^i$, $[\mathbf{C}_q]^i$, and $[\mathbf{K}_q]^i$ are the reduced mass, damping, and stiffness matrices. They are constant under linear elasticity. For the case of the commonly used *Rayleigh damping*, $[\mathbf{C}_q]^i$ is a linear combination of $[\mathbf{M}_q]^i$ and $[\mathbf{K}_q]^i$. $[\mathbf{f}_q]^i$ is the reduced external force. Domain displacements $[\mathbf{u}]^i$ and reduced coordinate $[\mathbf{q}]^i$ are related by the equation: $[\mathbf{u}]^i = [\mathbf{\Phi}]^i[\mathbf{q}]^i$, where $[\mathbf{\Phi}]^i$ contains the selected modes of the domain. It is noteworthy that since we are not using the

2. We say low rank that typically refers to the modes that are selected with high priority, such as rigid boundary modes and soft boundary modes associated with low-frequency harmonics. For internal vibrational modes, it means low-order inertia modes or normal modes with small eigenvalues.

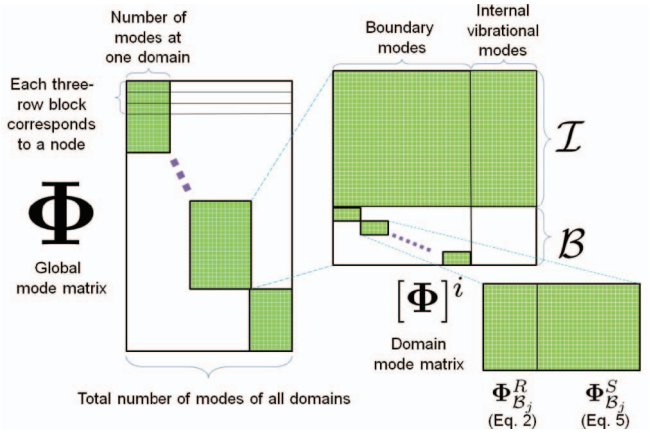


Fig. 5. The structures of global mode matrix and its nonzero diagonal submatrix at an individual domain. Shadowed blocks are nonzero submatrices.

regular eigenvectors as local subspace bases, the reduced equations are not decoupled and need to be solved using direct linear solver.

For a dynamic system with multiple domains, the global reduced mass, damping, and stiffness matrices are block diagonal: $\mathbf{M}_q = diag([\mathbf{M}_q]^i)$, $\mathbf{C}_q = diag([\mathbf{C}_q]^i)$, and $\mathbf{K}_q = diag([\mathbf{K}_q]^i)$. Similarly, the global mode matrix $\mathbf{\Phi}$ also has a block-diagonal-like structure as shown in Fig. 5. The global reduced displacement/velocity/acceleration $\mathbf{q}/\dot{\mathbf{q}}/\ddot{\mathbf{q}}$ is the column vector stacking $[\mathbf{q}]^i/[\dot{\mathbf{q}}]^i/[\ddot{\mathbf{q}}]^i$ at all domains. We do not apply mass orthogonalization to the modes as done in [31], because the boundary modes have clearly specified geometric properties that can be further utilized for interactive manipulation. We do not experience any stability issues in our experiment (with time step $1/30$ sec and implicit Newmark intergration).

4.2 Boundary Coupling

The locking issue and overconstraining problem are well resolved in our framework. First, neighboring subspaces are always compatible because the modes are computed based on the predefined boundary displacements, and the possible interface displacements at each subspace are the same. Second, the interface displacement is no longer a high-dimensional variable as it is expressed with a set of reduced geometric bases. Without loss of generality, assuming there are two domains α and β sharing an interface, a valid domain coupling requires keeping the duplicated interface DOFs at both domains overlapping during the simulation. Let $[\mathbf{\Phi}]^\alpha$ and $[\mathbf{\Phi}]^\beta$ denote the modes superset at the domains. Then, the interface constraint can be expressed as

$$[\mathbf{\Phi}_{\mathcal{B}_i}]^\alpha[\mathbf{q}]^\alpha - [\mathbf{\Phi}_{\mathcal{B}_i}]^\beta[\mathbf{q}]^\beta = \mathbf{0}, \tag{15}$$

where $[\mathbf{q}]^\alpha$ and $[\mathbf{q}]^\beta$ are the reduced modal displacement in domains. In general, the local interface index is different from a domain to another. To simplify the notation, we just use subscript $_{\mathcal{B}_i}$ for both domains that can be considered as a global index of the interfaces on the host mesh.

Equation (15) indicates that an interface displacement must be able to be represented with the reduced coordinates at both domains. In other words, the interface displacement has to be within the intersection of the subspaces spanned by $[\mathbf{\Phi}_{\mathcal{B}_i}]^\alpha$ and $[\mathbf{\Phi}_{\mathcal{B}_i}]^\beta$. If the intersection is only a small portion of the original subspaces, the regions nearby the boundary may appear to be "locking" as some deformations are filtered by the neighboring subspace. Time-varying subspace construction [32] may alleviate such an effect, but does not guarantee eliminating it. To have a natural deformation across the interface, the subspaces at the domains should be *compatible* at the interface DOFs. That is, any subspace interface displacements of α can also be represented within the subspace at β. Another problem of using (15) lies in the fact that the number of boundary constraints depends on the number of the domain's boundary DOFs. For a host mesh of large scale, a high-dimensional boundary constraint could easily turn the system into an overconstrained one.

In our framework, each interface is clearly associated with a subset of boundary modes, and the displacement of the interface is only determined by the corresponding reduced coordinates. Suppose domain d_0 neighbors to k domains $(d_1, d_2, \ldots, d_k, k \geq 1)$ at interfaces $\mathcal{B}_1, \mathcal{B}_2, \ldots, \mathcal{B}_k$, the boundary constraint for interdomain coupling can be directly enforced at the reduced domain coordinates that correspond to the interfaces for each pair of neighbor domains:

$$\begin{cases} [\mathbf{q}_{\mathcal{B}_1}]^{d_0} - [\mathbf{q}_{\mathcal{B}_1}]^{d_1} &= \mathbf{0} \\ [\mathbf{q}_{\mathcal{B}_2}]^{d_0} - [\mathbf{q}_{\mathcal{B}_2}]^{d_2} &= \mathbf{0} \\ & \vdots \\ [\mathbf{q}_{\mathcal{B}_k}]^{d_0} - [\mathbf{q}_{\mathcal{B}_k}]^{d_k} &= \mathbf{0}, \end{cases} \tag{16}$$

where the notation like $[\mathbf{q}_{\mathcal{B}_1}]^{d_0}$ stands for the subset of the reduced displacement at domain d_0 that corresponds to interface $\mathcal{B}_1$.

4.3 Large-Scale Deformation

The linear elasticity-based model reduction described above is not able to simulate large deformations. This is because 1) the *Cauchy's strain tensor* used is a linear strain tensor that generates inappropriate strain under rotations; and 2) the linear combination of the modes is not able to represent the intermediate rotational displacement. As a result, the rotational deformation must be specially handled. We adopt a nodewise corotational formulation with model reduction as in [4].

The *curl* of the linear deformation field is used to approximate the local rotation at each node. For finite elements with a linear shape function, it can be precomputed with the subspace modes. At each time step, we assemble a block-diagonal warping matrix $\tilde{\mathbf{R}}$. Each 3×3 diagonal block of $\tilde{\mathbf{R}}$ represents the current nodal warping. We refer the reader to the literature [4] for a detailed derivation of $\tilde{\mathbf{R}}$. Because of the domain decomposition, interface nodes are duplicated at the neighbor domains. This means that the number of rows of the global mode matrix $\mathbf{\Phi}$ is larger than the number of DOFs on the host mesh. Accordingly, we assemble an auxiliary elementary matrix $\mathbf{E}$, such that the rows of $\mathbf{\Phi}$ corresponding to the duplicated boundary nodes are picked only once. This operation implies that the computation of the *curl* at the interface

node takes all the neighbor domains into account, which ensures the smoothness of the warped deformation $\tilde{\mathbf{u}}$:

$$\tilde{\mathbf{u}} = \tilde{\mathbf{R}}\mathbf{E}\mathbf{\Phi}\mathbf{q}. \tag{17}$$

$\mathbf{E}$ is fixed when the domain decomposition is done and the product of $\mathbf{E}\mathbf{\Phi}$ can be precomputed. The update of nodal displacements is "local" as modes from other domains do not contribute to the final displacement of the node.[3] Accordingly, in real implementation, only local matrix-vector products are necessary to compute the displacement of the nodes.

5 Direct Manipulation

Manipulation of deformable objects is important for user interactivity and animation production. Our system enables the user to manipulate the deformable object through applying constraints to the nodes or the interfaces.

If a node p is constrained to a specified position. The corresponding PC equation is $\tilde{\mathbf{R}}_p\mathbf{\Phi}_p\mathbf{q} = \mathbf{c}$, where $\tilde{\mathbf{R}}_p$ and $\mathbf{\Phi}_p$ represent the warping matrix and three-row mode matrix at the constrained node. $\mathbf{c}$ is the desired node position. $\tilde{\mathbf{R}}_p$ is a time-varying matrix, and we use the warping matrix in the previous time step to approximate the current warping matrix. Therefore, a linear constraint equation can be used:

$$\mathbf{\Phi}_p\mathbf{q} - \tilde{\mathbf{R}}_p^{-1}\mathbf{c} = 0. \tag{18}$$

For subspace dynamics, the system could become over-constrained if multiple nodes are constrained by the user for interactive manipulation as each constrained node consumes three DOFs from the system. Therefore, a mechanism is needed to prevent the system from being overconstrained while keeping the system size small. In addition, subspace displacement may not be able to represent all the user-specified positions of the nodes, which could also lead to the locking problem.

To solve this issue, we design a new type of mode called the PC mode. The desired PC mode should be 1) linearly independent to the existing modes included at the domain and 2) able to represent any displacements for the constrained nodes. The latter requirement ensures that the compensating PC modes alone are able to fulfill the constraints so that other existing modes do not need to sacrifice their own freedom. We denote the constrained DOFs with set $\mathcal{C}$. Any DOFs in $\mathcal{I}$, if chosen in $\mathcal{C}$ are removed from $\mathcal{I}$. Imposing a unit displacement to each DOF in $\mathcal{C}$ while keeping the remaining DOFs in $\mathcal{C}$ as well as the ones in $\mathcal{B}$ fixed yields

$$\begin{bmatrix} \mathbf{K}_{\mathcal{I}\mathcal{I}} & \mathbf{K}_{\mathcal{I}\mathcal{C}} & \mathbf{K}_{\mathcal{I}\mathcal{B}} \\ \mathbf{K}_{\mathcal{C}\mathcal{I}} & \mathbf{K}_{\mathcal{C}\mathcal{C}} & \mathbf{K}_{\mathcal{C}\mathcal{B}} \\ \mathbf{K}_{\mathcal{B}\mathcal{I}} & \mathbf{K}_{\mathcal{B}\mathcal{C}} & \mathbf{K}_{\mathcal{B}\mathcal{B}} \end{bmatrix} \begin{bmatrix} \mathbf{\Phi}_{\mathcal{I}}^{PC} \\ \mathbf{I}_{\mathcal{C}}^{PC} \\ \mathbf{0} \end{bmatrix} = \begin{bmatrix} \mathbf{0} \\ \mathbf{F}_{\mathcal{C}}^{PC} \\ \mathbf{F}_{\mathcal{B}}^{PC} \end{bmatrix}, \tag{19}$$

and PC modes $\mathbf{\Phi}^{PC}$ can be computed through

3. Each three-row block in the global mode matrix ($\mathbf{\Phi}$) that corresponds to a node on the FE mesh is sparse (as shown in Fig. 5). If it is an internal node, the block has nonzeros at the columns corresponding to the domain that owns this node. If it is a boundary node, the block has nonzeros at the columns corresponding to its neighbor domains. In either case, the update only needs the reduced coordinates of related domains instead of the entire global $\mathbf{q}$.

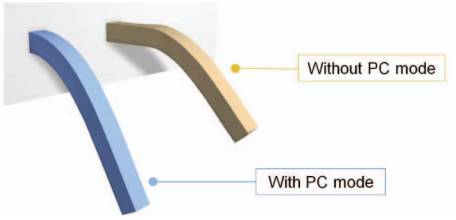


Fig. 6. Enforcing anchor nodes on the left end of the bar model using the Lagrange multiplier method: We can clearly see the locking region without using PC modes. Both bars are simulated with 100 modes in total.

$$\mathbf{\Phi}^{PC} = \begin{bmatrix} \mathbf{\Phi}_{\mathcal{I}}^{PC} \\ \mathbf{I}_{\mathcal{C}}^{PC} \\ \mathbf{0} \end{bmatrix} = \begin{bmatrix} -\mathbf{K}_{\mathcal{I}\mathcal{I}}^{-1}\mathbf{K}_{\mathcal{I}\mathcal{C}} \\ \mathbf{I}_{\mathcal{C}}^{PC} \\ \mathbf{0} \end{bmatrix}, \tag{20}$$

where $\mathbf{I}_{\mathcal{C}}^{PC}$ is the identity matrix standing for the unit displacement added at $\mathcal{C}$. The PC modes maintain the size of the subspace by introducing compensating modes to the system to preclude the system's DOFs from being "drained out" by the user constraints. It effectively resolves the locking problem induced by the position-constrained nodes (Fig. 6).

Our framework also supports direct manipulation of the interface by specifying its orientation. Such kind of manipulation enables the user to adjust the relative orientation among domains with deformation. Let $\mathbf{R}$ be the user-specified rotation for a boundary $\mathcal{B}_i$, and its axis-angle representation is $(\mathbf{a}, \theta)$. $\mathbf{p}$ denotes the relative position of an interface node p with respect to the rotation pivot (e.g., the interface centroid). The corresponding reduced coordinate $\mathbf{q}_{\mathcal{B}_i}^{rot} \in \mathbb{R}^{3\times 1}$ on the three tangential rotational modes is $\mathbf{q}_{\mathcal{B}_i}^{rot} = \theta\mathbf{a}$ and the unwarped displacement is $\mathbf{u} = \mathbf{\Phi}_p^{rot}\mathbf{q}_{\mathcal{B}_i}^{rot} = \theta[\mathbf{a}]_\times\mathbf{p}$, where $[\mathbf{a}]_\times$ is the skew-symmetric matrix of $\mathbf{a}$. With *Rodrigues's formulation*, the desired displacement $(\mathbf{R}\mathbf{p} - \mathbf{p})$ and the warped displacement $(\tilde{\mathbf{R}}_p\mathbf{u})$ can be written as

$$\mathbf{R}\mathbf{p} - \mathbf{p} = \left([\mathbf{a}]_\times^2 + sin(\theta)[\mathbf{a}]_\times - cos(\theta)[\mathbf{a}]_\times^2\right)\mathbf{p}, \tag{21}$$

and

$$\begin{aligned} \tilde{\mathbf{R}}_p\mathbf{u} &= \left(\mathbf{I} + [\mathbf{a}]_\times^2 + [\mathbf{a}]_\times\frac{1 - cos(\theta)}{\theta} - [\mathbf{a}]_\times^2\frac{sin(\theta)}{\theta}\right)\mathbf{u} \\ &= \left([\mathbf{a}]_\times^2 - cos(\theta)[\mathbf{a}]_\times^2 - sin(\theta)[\mathbf{a}]_\times^3\right)\mathbf{p}, \end{aligned} \tag{22}$$

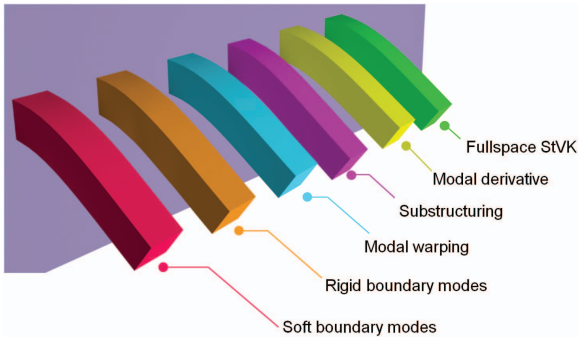
respectively. Note that $[\mathbf{a}]_\times = -[\mathbf{a}]_\times^3$ holds for skew-symmetric matrix $[\mathbf{a}]_\times$. Therefore, the left-hand sides of (21) and (22) are essentially equivalent to each other such as

$$\mathbf{R}\mathbf{p} - \mathbf{p} = \tilde{\mathbf{R}}_p\mathbf{\Phi}_p^{rot}\mathbf{q}_{\mathcal{B}_i}^{rot}. \tag{23}$$

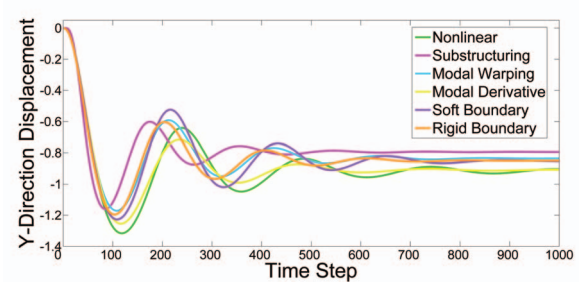
This means that, for the rotation $\mathbf{R}$ with axis-angle representation as $(\mathbf{a}, \theta)$, as long as $\mathbf{q}_{\mathcal{B}_i}^{rot} = \theta\mathbf{a}$, we can always have the desired rotational displacement with warping. Hence, the corresponding interface orientation constraint simply becomes

$$\mathbf{q}_{\mathcal{B}_i}^{rot} = \theta\mathbf{a}. \tag{24}$$

All the constraints including boundary constraint (16), position constraint (18) and interface constraint (24) are imposed through Lagrange multiplier method. The



(a)



(b)

Fig. 7. Comparative simulation among various simulators.

constraint equations form a constraint matrix, which is a constant if the constrained DOFs are not changed during the simulation. Using Newmark integration, the reduced Euler-Largange equations of all the domains can also be converted to a linear system that is solved at each time step based on the imposed constraint [42].

6 Experiment Results

Accuracy analysis: We first show a comparative experiment of simulating a bar model that is dropping under gravity ($-9.8\ \mathrm{m/s^2}$) with one end fixed. As shown in Fig. 7a, from right to left, the simulators used are nonlinear full space (single domain StVK, the ground truth), nonlinear single domain subspace integration with modal derivative [31], deformation substructuring [2] (nonlinear multidomain simulator), modal warping [4] (single domain, linear simulator with deformation warping), and our framework with and without soft boundary modes. The model is evenly divided into three domains for multidomain simulators. All the simulators use the same material parameters (e.g., Young's Modulus and Poisson's Ratio). To ensure the accuracy of the nonlinear simulator, we did not limit the maximum iteration number at each time step, and the threshold is set as $1e^{-6}$. The mass and stiffness damping coefficients are set as 0.5 and 0.08, respectively. The time step interval is 0.01 sec, and the *average implicit Newmark* time integration [43] is used for all simulators. The average vertical displacements of vertices at the free end of the model are recorded and shown in Fig. 7b. All the subspace simulators have the same number of bases (55 bases). The relative L_2 error for each simulator w.r.t. ground truth is 7.36 percent for modal derivative, 13.21 percent for modal warping, 13.04 percent for substructuring (the displacements from duplicated boundary DOFs are averaged), and 10.35 and 8.45 percent for our method (without and with soft boundary modes). Inducing interface flexibility to the system with soft boundary modes increases the accuracy of the simulation.

Coupling comparison: With corotational displacement correction, we can produce a similar deformation to other state-of-art nonlinear methods. One advantage of our method is that the multiplier-enforced coupling always guarantees a seamless domain connection, while the boundary-driven subspace avoids the locking and over-constraint problem. Fig. 8 illustrates the coupling effect using our algorithm and the penalty force-based method [1]. The elbow joint of the arm is a one DOF joint and rotating with angular velocity $0.2\ \mathrm{rad/sec}$. Both quasi-static nonlinear modes and modal derivatives are used for nonlinear subspace bases. The cracks at the elbow vanish when increasing the number of modes to 40. The simulation time step has to be limited to small values (e.g., 0.01 sec) because of the usage of highly stiff springs. On the other hand, our method excels with the larger time step (0.3 sec) and smaller size of subspace without any cracks or stability issue.

Another comparison is with deformation substructuring using rigid interface fitting [2] (Fig. 9). In this comparison, the tyrannosaurus model is decomposed into 17 domains and the rank for each domain is 30 (modal derivatives for substructuring and rigid/soft boundary modes for our method). We notice that near-rigid fitting used in substructuring [2] could also induce cracks at the domain

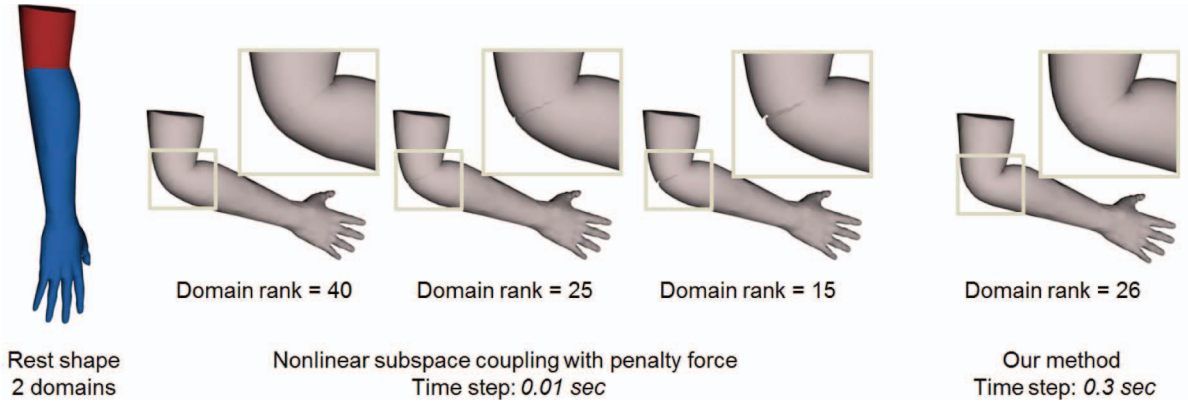


Fig. 8. Coupling comparison between our method and nonlinear multidomain with a penalty method [1].

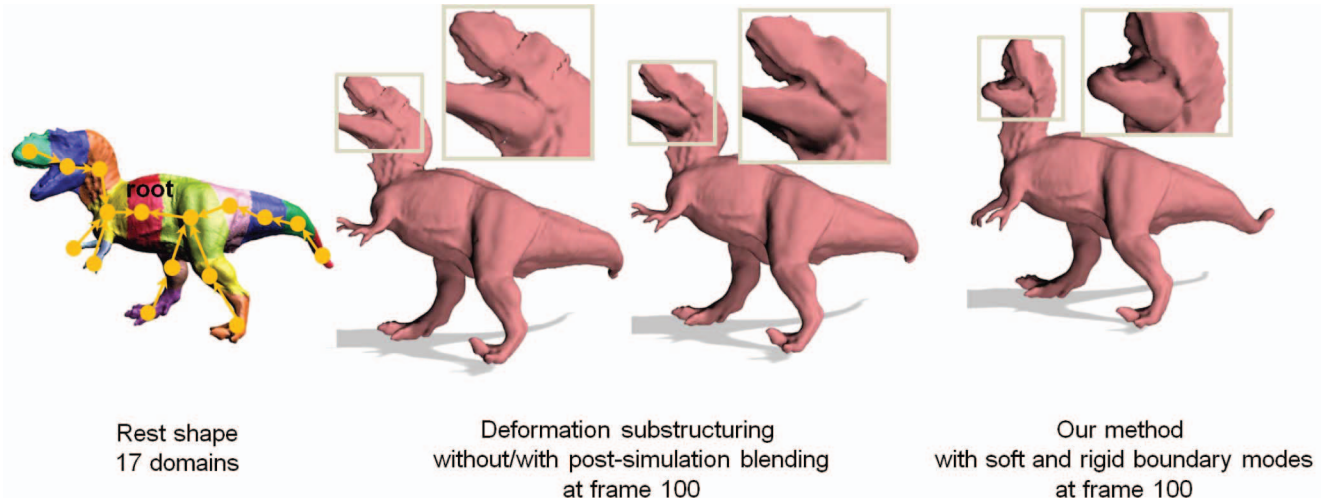


Fig. 9. Coupling comparison between our method and deformation substructuring using rigid interface fitting [2].

interfaces as the deformation and interface region goes large immediately after the simulation. Postsimulation processing with interface blending is able to relieve this artifact [2]. With soft boundary modes, our method avoids this problem and does not need any extra postsimulation amendments. More importantly, simple blending could still introduce artifacts of discontinuity as shown in Fig. 2 which requires more advanced geometrical blending at postsimulation stage in each time step.

Our method is essentially using linear elasticity, and the system matrix is constant during the simulation. The runtime performance is faster than nonlinear simulators: In the first comparison shown in Fig. 8, the overall frame per second (FPS) is 172 using our method and 89 using implicit springs [1]; in the second comparison as in Fig. 9, our overall FPS is 34 compared to 18 using substructuring [2]. In the implementation of [2], we perform three Newton-Raphson iterations at each time step. If we reduce this number to 1, the FPS for the tyrannosaurus model increases to 28 with higher residual error.

Simulation with manipulation: Large deformations are well handled in our framework. Fig. 11 shows a sunflower model with 47,164 elements and 13 domains. We apply external forces at the top of the flower. Large deformations are generated at the stalk, and the leaves have relatively smaller local vibrational deformations. Unlike deformation substructuring [2], our framework does not rely on hierarchical domain decomposition. Deformable objects with looped domain connection can be simulated with our method naturally. Fig. 10 shows the result of simulating a sailboat model with seven loops and 253,998 elements. We apply scripted forces at the masts and the body of the boat. Inertia modes of order 2 are assigned to the canvases to have enriched local deformation effects with the applied wind field. Domain loop can also be handled with spring coupling [1]. However, for looped multidomain deformation using substructuring [2], using implicit penalty force between domains could break the sequential simulation order of domains and bring more complexity to the system.

Supporting manipulations is straightforward with constraints in our framework. In Fig. 12, the fish model consists of 111,196 elements and 10 domains. It is manipulated by the user in real time. One constrained node is used to control the body motion of the fish, while the orientations of the interfaces connecting the rear fin and fish head are also

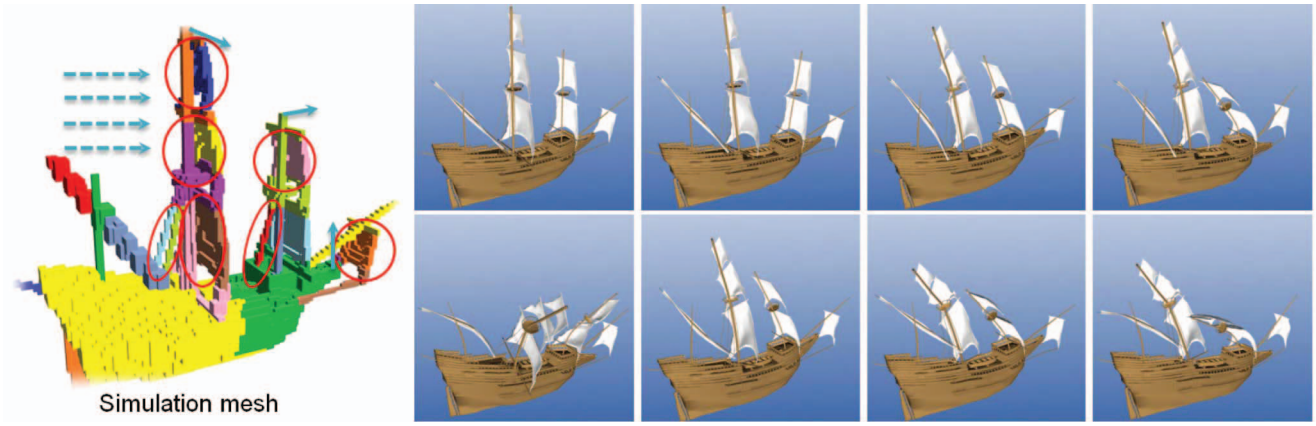


Fig. 10. Simulation of a sailboat model with 253,998 elements, 24 domains, and seven loops (highlighted with red circles) in real time (21 FPS). The total number of modes used is 830. The domains corresponding to the masts and the body of the boat have stiffer material, while the canvases have softer material. We applied scripted forces at the masts and the body of the boat and the wind force field at the canvases (shown as arrows in the figure).

Fig. 11. Simulation of a sunflower model with large deformation.

manipulated. In our implementation, a damped constraint equation [44] is used to avoid sharp impulse-like constraint forces and increase the stability of the system.

Performance: Detailed time performances can be found in Table 2. Our system is implemented with a Windows 7 PC equipped with a 2.26-GHz Intel Xeon CPU (only a single core is used for simulation) and 12-GB RAM. Because the domains are coupled with hard constraints, we must solve the system entirely (similar to coupling with implicit penalty forces as in [1]) instead of sequentially solving each domain individually [2]. However, linear elasticity with a constant system matrix needs only to be preconditioned once before simulation, and solving the system is not the bottleneck of the framework. For example, in our sailboat model, solving the prefactorized linear system takes less than 1 ms, which is only 5 percent computation time of one time step. The global mode matrix $\mathbf{\Phi}$ has a block-diagonal structure, and the runtime nodal displacement updates (e.g., using (17) to compute the nodal displacement on the mesh) are actually "local" as it only depends on the number of modes in the domain rather than the global subspace size. Hence, our framework is on average d times faster than modal warping [4], where d is the number of domains. In terms of precomputation, our method is significantly faster than the global subspace method, which generally requires solving a very large eigen problem. It could take up to several hours and consume considerable amount of memory, while local subspace bases precomputation is orders of magnitude faster and can be finished within seconds. In the table, we record the precomputation time for "fresh" computation of each type of mode including calculating the inverse the internal stiffness matrix ($\mathbf{K}_{II}$), which is the most time-consuming part in the mode computation. In fact, this only needs to be done once for each domain (if inertia modes are used). As a result, we can prefactorize $\mathbf{K}_{II}$ before mode computation, and the precomputation for domain modes can be further shortened.

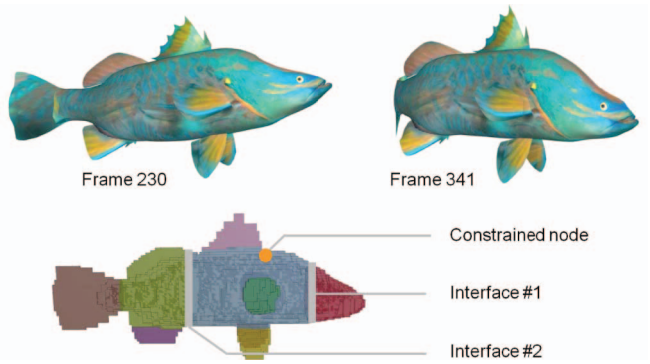


Fig. 12. Interactive manipulation on the fish model.

7 Conclusion

We have developed a seamless coupling method for multidomain subspace deformation in the framework of nodewise corotational elasticity. With boundary-aware mode construction method, the boundary coupling constraints can be directly imposed on the reduced coordinates, efficiently avoiding the over-constraining problem. Our algorithm can achieve real-time simulation performance for large-scale meshes and can support direct manipulation of deformable objects.

Limitation: When there are many boundary interfaces in the domain decomposition, the number of boundary deformation modes can be large. This leads to linear growth in the required number of multipliers and accordingly the dimension of the system matrix. It can be handled by separating the variables in the linear constraint equations into independent and dependent variables, and constructing the system matrix only using the independent variables to avoid explicit use of Lagrange multipliers. Nodewise corotational formulation produces *ghost forces* if the domains are unconstrained when local displacement at each node is warped. Fortunately, such effect is not seen in our experiment, because constraints always exist in the framework (e.g., interface constraints or PCs). The hard constraints suppress such artifact, and the results are physically plausible.

In subspace deformation techniques, the time for system resolving is small. We found that the displacement warping step, i.e., the estimation of rotation at each node, costs more than 50 percent of the time in one simulation step. We plan to reduce the time through adaptive rotation estimation. It can be done by only estimating the rotations of sparsely

TABLE 2
Time Performance

Model	Model statistics		Domains		Precomputation			Runtime Performance		
	# elements	# vertices	# r	# domains	boundary	inertia	conditioning	system	disp.	FPS
Sunflower	$47,164$	$12,659$	260	13	$3.46\ s$	$0.3\ s$	$0.03\ s$	$0.06\ ms$	$8\ ms$	67
Fish	$111,196$	$27,075$	340	10	$11.76\ s$	$6.3\ s$	$0.03\ s$	$0.08\ ms$	$18\ ms$	41
Tyrannosaurus	$121,976$	$32,009$	510	17	$15.3\ s$	$9.4\ s$	$0.14\ s$	$0.14\ ms$	$25\ ms$	34
Sailboat	$253,998$	$48,998$	830	24	$43.5\ s$	$6.2\ s$	$0.17\ s$	$0.6\ ms$	$30\ ms$	21

 # r: total number of modes; boundary*: precomputation time for boundary modes (rigid and soft ones);* inertia*: precomputation time for inertia modes;* conditioning*: time for a preconditioning linear system;* system*: time for a solving reduced linear system at each time step;* disp*: time for computing displacement from reduced coordinates;* FPS*: overall frame per second.*

sampled nodes and checking whether the rotations can be directly used at the remaining nodes.

Unconstrained deformable object is only partially supported with this framework. Because of the linear elasticity used, the underlying rigid body motions of the domains are only the first-order approximation of the real rigid body dynamics. Fortunately, modal warping is able to correct the distorted rotation. Another method is following the similar strategy as in [31], [45] to explicitly couple the rigid body motion and the deformation of the domains.

Future work: An interesting research direction is to investigate a way to apply the boundary-driven mode construction method to the coupling of multidomain subspace deformations using a nonlinear StVK deformable model in the spirit of modal derivative framework. It is possible to construct a first or higher order approximation of the change of boundary deformation modes using (1).

We also plan to investigate a parallel multidomain subspace deformation algorithms. This calls for coupling constraints to be handled in each domain separately. Another interesting research direction is to apply the multidomain subspace deformation technique to other application areas, such as fabrication-aware design, to provide fast simulation results when the user edits the geometry model.

Acknowledgments

The author would like to thank anonymous reviewers for their constructive comments and suggestions. Yin Yang and Xiaohu Guo are partially supported by the US National Science Foundation under Grants Nos. CNS-1012975 and IIS-1149737. Weiwei Xu is partially supported by NSFC (No. 61272392). Kun Zhou is supported by NSFC (No. 61272305).

References

[1] T. Kim and D.L. James, "Physics-Based Character Skinning Using Multi-Domain Subspace Deformations," *Proc. Symp. Computer Animation,* pp. 63-72, 2011.

[2] J. Barbič and Y. Zhao, "Real-Time Large-Deformation Substructuring," *Proc. ACM SIGGRAPH '11,* pp. 91:1-91:8, 2011.

[3] R.R. Craig, "A Review of Time-Domain and Frequency-Domain Component Mode Synthesis Methods," *Int'l J. Analytical and Experimental Modal Analysis,* vol. 2, pp. 59-72, 1987.

[4] M.G. Choi and H.-S. Ko, "Modal Warping: Real-Time Simulation of Large Rotational Deformation and Manipulation," *IEEE Trans. Visualization Computer Graphics,* vol. 11, no. 1, pp. 91-101, Jan. 2005.

[5] A. Nealen, M. Müller, R. Keiser, E. Boxerman, and M. Carlson, "Physically Based Deformable Models in Computer Graphics," *Computer Graphics Forum,* vol. 25, no. 4, pp. 809-836, 2006.

[6] M. Müller, J. Stam, D. James, and N. Thürey, "Real Time Physics: Class Notes," *Proc. ACM SIGGRAPH,* pp. 88:1-88:90, 2008.

[7] D. Terzopoulos, J. Platt, A. Barr, and K. Fleischer, "Elastically Deformable Models," *Proc. ACM SIGGRAPH '87,* pp. 205-214, 1987.

[8] D. Terzopoulos and K. Fleischer, "Deformable Models," *The Visual Computer,* vol. 4, pp. 306-331, 1988.

[9] S. Capell, S. Green, B. Curless, T. Duchamp, and Z. Popović, "A Multiresolution Framework for Dynamic Deformations," *Proc. Symp. Computer Animation,* pp. 41-47, 2002.

[10] X. Wu, M.S. Downes, T. Goktekin, and F. Tendick, "Adaptive Nonlinear Finite Elements for Deformable Body Simulation Using Dynamic Progressive Meshes," *Computer Graphics Forum,* vol. 20, pp. 349-358, 2001.

[11] E. Grinspun, P. Krysl, and P. Schröder, "Charms: A Simple Framework for Adaptive Simulation," *Proc. ACM SIGGRAPH '02,* pp. 281-290, 2002.

[12] J. Huang, X. Shi, X. Liu, K. Zhou, L.-Y. Wei, S.-H. Teng, H. Bao, B. Guo, and H.-Y. Shum, "Subspace Gradient Domain Mesh Deformation," *ACM Trans. Graphics,* vol. 25, pp. 1126-1134, July 2006.

[13] A.R. Rivers and D.L. James, "FastLSM: Fast Lattice Shape Matching for Robust Real-Time Deformation," *Proc. ACM SIGGRAPH '07,* 2007.

[14] L. Kharevych, P. Mullen, H. Owhadi, and M. Desbrun, "Numerical Coarsening of Inhomogeneous Elastic Materials," *Proc. ACM SIGGRAPH '09,* pp. 1-8, 2009.

[15] M. Nesme, P.G. Kry, L. Jeřábková, and F. Faure, "Preserving Topology and Elasticity for Embedded Deformable Models," *Proc. ACM SIGGRAPH '09,* pp. 1-9, 2009.

[16] P. Kaufmann, S. Martin, M. Botsch, and M. Gross, "Flexible Simulation of Deformable Models Using Discontinuous Galerkin FEM," *Graphical Models,* vol. 71, no. 4, pp. 153-167, July 2009.

[17] M. Müller, J. Dorsey, L. McMillan, R. Jagnow, and B. Cutler, "Stable Real-Time Deformations," *Proc. Symp. Computer Animation,* pp. 49-54, 2002.

[18] M. Müller and M. Gross, "Interactive Virtual Materials," *Proc. Graphics Interface,* pp. 239-246, 2004.

[19] O. Etzmuss, M. Keckeisen, and W. Strasser, "A Fast Finite Element Solution for Cloth Modelling," *Proc. 11th Pacific Conf. Computer Graphics and Applications,* pp. 244-251, Oct. 2003.

[20] Y. Zhu, E. Sifakis, J. Teran, and A. Brandt, "An Efficient Multigrid Method for the Simulation of High-Resolution Elastic Solids," *ACM Trans. Graphics,* vol. 29, no. 2, pp. 16:1-16:18, Apr. 2010.

[21] S. Martin, P. Kaufmann, M. Botsch, E. Grinspun, and M. Gross, "Unified Simulation of Elastic Rods, Shells, and Solids," *Proc. ACM SIGGRAPH,* 2010.

[22] J. Barbič, F. Sin, and E. Grinspun, "Interactive Editing of Deformable Simulations," *ACM Trans. Graphics,* vol. 31, no. 4, pp. 70:1-70:8, July 2012.

[23] M.G. Choi, S.Y. Woo, and H.-S. Ko, "Real-Time Simulation of Thin Shells," *Proc. Eurographics Conf.,* pp. 349-354, 2007.

[24] X. Guo and H. Qin, "Real-Time Meshless Deformation: Collision Detection and Deformable Objects," *Computer Animation Virtual Worlds,* vol. 16, nos. 3/4, pp. 189-200, 2005.

[25] F. Hecht, Y.J. Lee, J.R. Shewchuk, and J.F. O'Brien, "Updated Sparse Cholesky Factors for Corotational Elastodynamics," *ACM Trans. Graphics,* vol. 31, no. 5, pp. 1-13, Oct. 2012.

[26] A. Pentland and J. Williams, "Good Vibrations: Modal Dynamics for Graphics and Animation," *Proc. ACM SIGGRAPH '89,* 1989.

[27] J.F. O'Brien, P.R. Cook, and G. Essl, "Synthesizing Sounds from Physically Based Motion," *Proc. ACM SIGGRAPH '01,* pp. 529-536, 2001.

[28] D.L. James and D.K. Pai, "DyRT: Dynamic Response Textures for Real Time Deformation Simulation with Graphics Hardware," *Proc. ACM SIGGRAPH '02,* pp. 582-585, 2002.

[29] P.G. Kry, D.L. James, and D.K. Pai, "Eigenskin: Real Time Large Deformation Character Skinning in Hardware," *Proc. Symp. Computer Animation,* pp. 153-159, 2002.

[30] K. Hauser, C. Shen, and J. O'Brien, "Interactive Deformation Using Modal Analysis with Constraints," *Proc. Graphics Interface,* pp. 247-255, 2003.

[31] J. Barbič and D.L. James, "Real-Time Subspace Integration for St. Venant-Kirchhoff Deformable Models," *Proc. ACM SIGGRAPH '05,* pp. 982-990, Aug. 2005.

[32] T. Kim and D.L. James, "Skipping Steps in Deformable Simulation with Online Model Reduction," *ACM Trans. Graphics,* vol. 28, pp. 123:1-123:9, Dec. 2009.

[33] M. Müller, B. Heidelberger, M. Teschner, and M. Gross, "Meshless Deformations Based on Shape Matching," *Proc. ACM SIGGRAPH,* pp. 471-478, 2005.

[34] M. Müller and N. Chentanez, "Solid Simulation with Oriented Particles," *ACM Trans. Graphics,* vol. 30, no. 4, pp. 92:1-92:10, July 2011.

[35] J. Huang, X. Liu, H. Bao, B. Guo, and H.-Y. Shum, "An Efficient Large Deformation Method Using Domain Decomposition," *Computer Graphics,* vol. 30, no. 6, pp. 927-935, 2006.

[36] B. Vallet and B. Lévy, "Spectral Geometry Processing with Manifold Harmonics," *Computer Graphics Forum,* vol. 27, no. 2, pp. 251-260, 2008.

[37] G. Rong, Y. Cao, and X. Guo, "Spectral Mesh Deformation," *Vision Computer,* vol. 24, no. 7, pp. 787-796, July 2008.

[38] K. Hildebrandt, C. Schulz, C.V. Tycowicz, and K. Polthier, "Interactive Surface Modeling Using Modal Analysis," *ACM Trans. Graphics,* vol. 30, no. 5, pp. 119:1-119:11, Oct. 2011.
[39] E. Sifakis and J. Barbic, "FEM Simulation of 3D Deformable Solids: A Practitioner's Guide to Theory, Discretization and Model Reduction," *Proc. ACM SIGGRAPH,* pp. 20:1-20:50, 2012.
[40] P. Kaufmann, S. Martin, M. Botsch, and M. Gross, "Flexible Simulation of Deformable Models Using Discontinuous Galerkin FEM," *Graphical Models,* vol. 71, no. 4, pp. 153-167, July 2009.
[41] S.S. An, T. Kim, and D.L. James, "Optimizing Cubature for Efficient Integration of Subspace Deformations," *Proc. ACM SIGGRAPH '08,* pp. 165:1-165:10, 2008.
[42] D. Baraff, "Linear-Time Dynamics Using Lagrange Multipliers," *Proc. ACM SIGGRAPH '96,* pp. 137-146, 1996.
[43] T. Hughes, *The Finite Element Method: Linear Static and Dynamic Finite Element Analysis.* Dover Publications, 2000.
[44] J. Baumgarte, "Stabilization of Constraints and Integrals of Motion in Dynamical Systems," *Computer Methods in Applied Mechanics and Eng.,* vol. 1, no. 1, pp. 1-16, 1972.
[45] D. Terzopoulos and A. Witkin, "Physically Based Models with Rigid and Deformable Components," *IEEE Computer Graphics Applications,* vol. 8, no. 6, pp. 41-51, Nov. 1988.

Yin Yang is currently working toward the PhD degree and is a research assistant in the Department of Computer Science, The University of Texas at Dallas. His research focuses on physics-based animation/simulation and related applications, scientific visualization and medical imaging. For more information, please visit http://www.utdallas.edu/~yinyang.

Weiwei Xu received the BS and master's degrees in computer science from Hohai University in 1996 and 1999, respectively, and the PhD degree in computer graphics from Zhejiang University, Hangzhou. He is a professor at Hangzhou Normal University in the Department of Computer Science. Before that, he was a researcher in Internet Graphics Group, Microsoft Research Asia from October 2005 to June 2012. He has been a postdoc researcher at Ritsmeikan university in Japan around one year from 2004 to 2005. He has published more than 20 papers on international conference and journals, include nine papers on *ACM Transaction on Graphics*. His main research interests include digital geometry processing and computer animation techniques. He is now focusing on how to enhance the geometry design algorithm though the integration of physical properties.

Xiaohu Guo received the PhD degree in computer science from the State University of New York at Stony Brook in 2006. He is an associate professor of computer science at the University of Texas at Dallas. His research interests include computer graphics, animation, and visualization, with an emphasis on geometric, and physics-based modeling. His current researches at UT-Dallas include spectral geometric analysis, deformable models, centroidal Voronoi tessellation, GPU algorithms, 3D and 4D medical image analysis, and so on. He received the prestigious US National Science Foundation CAREER Award in 2012. He is a member of the IEEE. For more information, please visit http://www.utdallas.edu/~xguo.

Kun Zhou received the BS and PhD degrees in computer science from Zhejiang University in 1997 and 2002, respectively. He is currently a Cheung Kong distinguished professor in the Computer Science Department of Zhejiang University, and a member of the State Key Lab of CAD&CG. Before joining Zhejiang University, he was a leader researcher of the graphics group at Microsoft Research Asia. His research interests include shape modeling/editing, texture mapping/synthesis, real-time rendering, and GPU parallel computing.

Baining Guo received the BS degree in mathematics from Beijing University. He went to Cornell University for his graduate study from 1986 to 1991 and received the MS degree in computer science and the PhD degree in applied mathematics. He is the deputy managing director at Microsoft Research Asia, where he also leads the Graphics Lab. Prior to joining Microsoft Research in 1999, he was a senior staff researcher with the Microcomputer Research Labs of Intel Corporation in Santa Clara, California. His research interests include computer graphics, visualization, and natural user interface. He is particularly interested in studying light transmission and reflection in complex, textured materials, with applications to texture and reflectance modeling. He also worked on real-time rendering and geometry modeling. He was on the editorial boards of *IEEE Transactions on Visualization and Computer Graphics* (2006-2010) and *Computer and Graphics* (2007-2011). He is currently an associate editor of the *IEEE Computer Graphics and Applications*. He has served on the program committees of numerous conferences in graphics and visualization, including ACM Siggraph and IEEE Visualization. He holds more than 40 US patents. He is a fellow of the IEEE and ACM.