

Boundary-Aware Multi-Domain Subspace Deformation

Yin Yang, Weiwei Xu, Xiaohu Guo, Kun Zhou, Baining Guo

Abstract—In this paper, we propose a novel framework for multi-domain 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 over-constraining. 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. Multi-domain 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 two-fold. 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 multi-domain 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 multi-domain 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 pre-determined 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 multi-domain 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 multi-domain subspace deformations with arbitrary domain decomposition remains a technical challenge.

The goal of this paper is to develop a seamless coupling method for multi-domain subspace deformation in the framework of node-wise 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 over-constraints 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

- Yin Yang and Xiaohu Guo are with the Department of Computer Science, The University of Texas at Dallas, TX. E-mail: {yinyang|xguo}@utdallas.edu
- Weiwei Xu is with Hangzhou Normal University, Hangzhou, China. E-mail: weiwei.xu.g@gmail.com
- Kun Zhou is with State Key Lab of CAD and CG, Zhejiang University, Hangzhou, China. E-mail: kunzhou@acm.org
- Baining Guo is with Microsoft Research Aisa, Beijing, China. E-mail: bainguo@microsoft.com

coupling constraints without over-constraining. 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 naturally handled as well. We show the capability of our multi-domain 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. Sec. 2 reviews the related work. Sec. 3 describes how to construct deformation modes from the boundary deformation. The deformation algorithm and direct manipulation methods are described in Secs. 4 and 5. Experimental results and limitations are discussed in Sec. 6. Finally, we conclude in Sec. 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 (ODEs) 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 FEM especially for the finite element meshes of large size. In order to make deformable models more practical for interactive applications, numerous works have been proposed and have greatly advanced related areas. Multi-resolution [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 node-wise 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 pre-factorization 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 eigen decomposition 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. In order 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 inter-domains constraints should be considered during the selection of deformation modes in order to construct a reasonable local subspace at each domain. Huang *et al.* [35] use node-pair position constraints for domain coupling and a pre-computed force-displacement matrix to accelerate matrix-vector multiplication. Barbić and Zhao [2] adopt a passive inter-domain deformation mechanism with rigid interface fitting when the parent domain deforms. This framework works well for domain decomposition with tree-like hierarchies, and the pre-computation 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 jiggling. 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 pre-determined. 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 post-simulation 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 over-sized boundary problem. The domain coupling constraint can be directly imposed on the reduced coordinates without the over-constraining problem. Boundary modes constitute the *static deformation* of the domains. The extra internal vibrational deformation is also captured with interior modes that do not participate in the domain coupling. Domains are always exactly coupled and the post-simulation 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 sub-meshes 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 connected¹. The *modes* are simply the pre-computed domain displacements which 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] which 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.

1. The face-connectivity avoids the singularity of the sub-stiffness 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 sub matrix is always non-singular.

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, $\mathcal{B}_1, \mathcal{B}_2, \dots, \mathcal{B}_k$. 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.

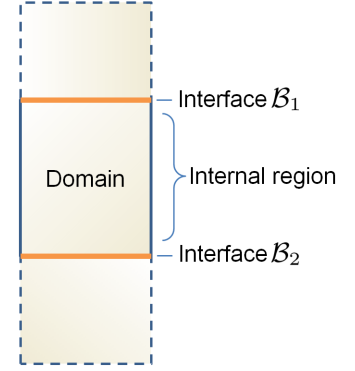


Fig. 1. A three-domain bar.

The middle domain has the boundary with two interfaces. Let Φ denote the modes of a single domain. According to the DOF types, it can be decomposed into two parts $\Phi = [\Phi_{\mathcal{I}}^T | \Phi_{\mathcal{B}}^T]^T$. As illustrated in Tab.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.

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

TABLE 1

Matrix notation used in the mode computation.

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 $\mathbf{K}\mathbf{u} = \mathbf{f}$, where the domain's stiffness matrix is denoted by \mathbf{K} . \mathbf{u} and \mathbf{f} represent the displacement and forces of the domain. The unknown responding deformation at \mathcal{I} is computed by solving the following equilibrium:

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

By stacking boundary and internal parts computed in Eq. 5 and Eq. 7, the soft boundary modes are assembled as:

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

3.3 Internal Vibrational Modes

Boundary modes are suitable for capturing static deformation driven by the un-accelerated 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):

$$\begin{cases} (\mathbf{K}_{\mathcal{I}\mathcal{I}} - \Lambda \mathbf{M}_{\mathcal{I}\mathcal{I}}) \Phi_{\mathcal{I}}^N = 0 \\ \Phi_{\mathcal{B}}^N = 0 \end{cases}, \quad (9)$$

Eq. 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 multi-domain 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 pre-compute 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 because inertia modes are computed by solving the equilibrium with inertia forces that correspond to the boundary acceleration:

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

where, Φ^R represents the acceleration along the directions of the rigid boundary modes. Soft boundary modes can also be included in Eq. 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:

$$\Phi_{(1)}^I = \begin{bmatrix} \Phi_{\mathcal{I}(1)}^I \\ \mathbf{0} \end{bmatrix} = \begin{bmatrix} \mathbf{K}_{\mathcal{I}\mathcal{I}}^{-1} \mathbf{M}_{\mathcal{I}\mathcal{I}} \Phi_{\mathcal{I}}^R \\ \mathbf{0} \end{bmatrix}. \quad (11)$$

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

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

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 mode 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 pre-computation: the inverse of $\mathbf{K}_{\mathcal{I}\mathcal{I}}$ is shared in the computation of boundary modes (e.g., Eq. 3 and 7). In fact, inertia mode $\Phi_{\mathcal{I}}^I$ spans a *block Krylov subspace*:

$$\Phi_{\mathcal{I}}^I = [\mathbf{A} \Phi_{\mathcal{I}}^R | \mathbf{A}^2 \Phi_{\mathcal{I}}^R | \mathbf{A}^3 \Phi_{\mathcal{I}}^R | \dots], \quad (13)$$

where $\mathbf{A} = \mathbf{K}_{\mathcal{I}\mathcal{I}}^{-1} \mathbf{M}_{\mathcal{I}\mathcal{I}}$. Krylov subspace is a well-known numerical method to compute the generalized eigen problem as defined in Eq. 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 pre-known 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.

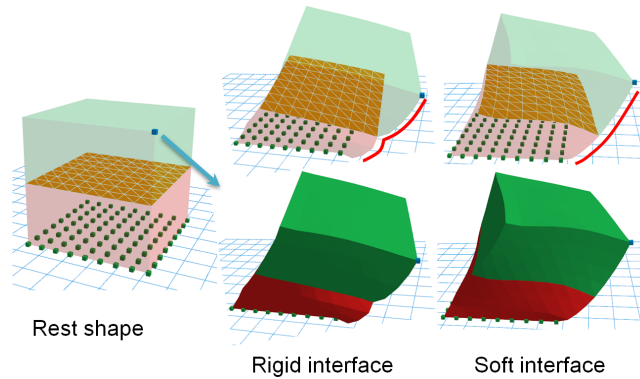


Fig. 2. The assumption of rigid interface [2] induces artifact when the interface is wide and the neighbor domains are soft. Using our soft modes yields smooth and natural deformation. Same number of modes (30 per domain) are used in the comparison while 5 soft boundary modes are used to capture the boundary deformation. The external force is shown as an arrow.

3.4 Discussion

Rigid interface vs. 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 generates 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 post-simulation computation like moving least square (MLS) embedding [40].

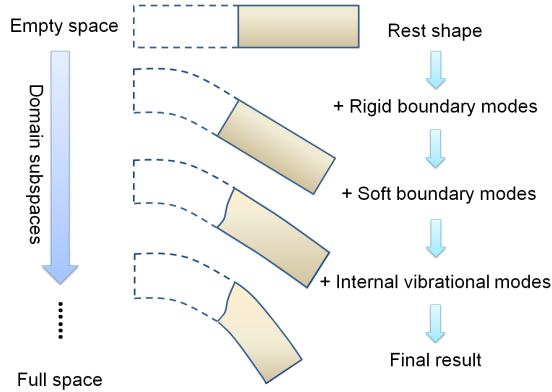


Fig. 3. Subspaces spanned by different types of modes constitute the layered deformation. Final deformation of the domain can be understood as the superposition of deformations from each subspace. Complete set of modes span the full space.

Layered subspaces In our boundary-aware mode construction framework, the local domain’s deformation 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 in order to keep the domain coupled with its neighbors. This portion of deformation is represented with rigid boundary modes. After that, some non-rigid 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 constitute the full space of the domain’s deformation.

Low-rank² 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 which is purely decided by the geometry feature of the interface (via computing its harmonic bases and rigid motion). We will see in Sec. 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 vs. 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 ($\mathbf{K}\Phi = \Lambda\mathbf{M}\Phi$) 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 multi-domain 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

2. We say low-rank which 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.

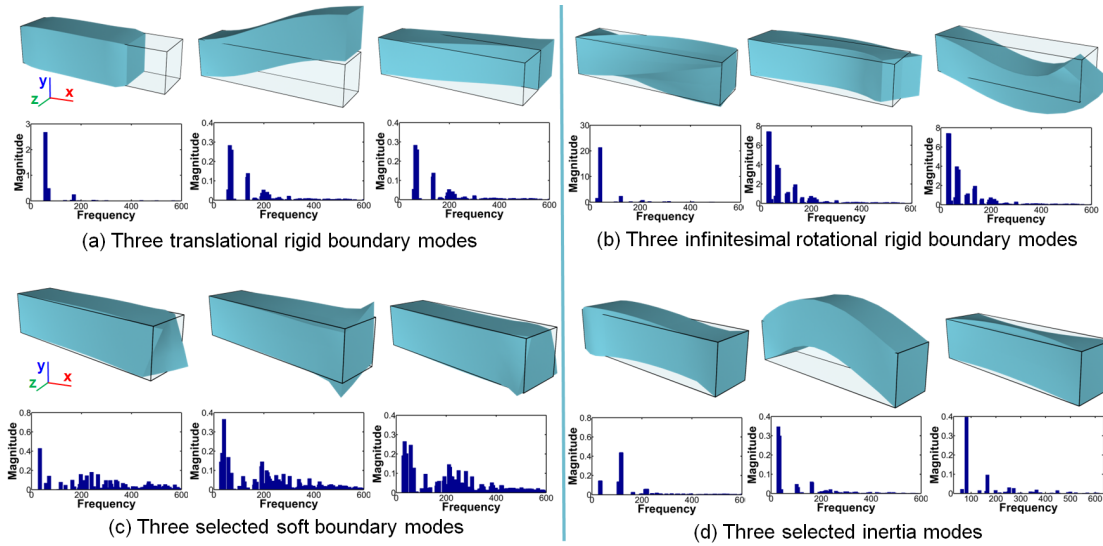


Fig. 4. The shapes of each type of warped modes associated with right interface of the domain. The left interface is assigned with zero displacements.

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 multi-domain deformable bodies and the subspace domain coupling. Then we introduce how to generalize the reduced node-wise corotational elasticity [4] for multi-domain 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:

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

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: $[\mathbf{u}]^i = [\Phi]^i [\mathbf{q}]^i$, where $[\Phi]^i$ contains the selected modes of the domain. It is noteworthy that since we are not using the 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: $\mathbf{M}_q = \text{diag}([\mathbf{M}_q]^i)$, $\mathbf{C}_q = \text{diag}([\mathbf{C}_q]^i)$ and $\mathbf{K}_q = \text{diag}([\mathbf{K}_q]^i)$. Similarly, the global mode matrix Φ also has block-diagonal-like structure

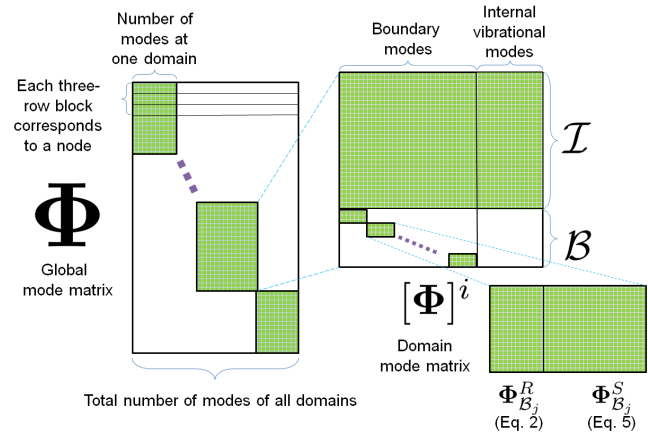


Fig. 5. The structures of global mode matrix and its nonzero diagonal submatrix at an individual domain. Shaded blocks are nonzero submatrices.

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 which can be further utilized for interactive manipulation. We do not experience any stability issues in our experiment (with time step $1/30 \text{ sec}$ and implicit Newmark integration).

4.2 Boundary Coupling

The locking issue and over-constraining problem are well resolved in our framework. First, neighboring subspaces are always compatible because the modes are computed based on the pre-defined 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 $[\Phi]^\alpha$ and $[\Phi]^\beta$ denote the modes superset at the domains. Then the interface constraint can be expressed as:

$$[\Phi_{\mathcal{B}_i}]^\alpha [\mathbf{q}]^\alpha - [\Phi_{\mathcal{B}_i}]^\beta [\mathbf{q}]^\beta = \mathbf{0}, \quad (15)$$

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 which can be considered as a global index of the interfaces on the host mesh.

Eq. 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 $[\Phi_{\mathcal{B}_i}]^\alpha$ and $[\Phi_{\mathcal{B}_i}]^\beta$. 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. In order 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 Eq. 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 over-constrained 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 ($d_1, d_2, \dots, d_k, k \geq 1$) at interfaces $\mathcal{B}_1, \mathcal{B}_2, \dots, \mathcal{B}_k$, the boundary constraint for inter-domain coupling can be directly enforced at the reduced domain coordinates that correspond to the interfaces for each pair of neighbor domains:

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

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 which 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 node-wise 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 pre-computed 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 Φ 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 Φ 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 $\tilde{\mathbf{u}}$:

$$\tilde{\mathbf{u}} = \tilde{\mathbf{R}}\mathbf{E}\Phi\mathbf{q}. \quad (17)$$

\mathbf{E} is fixed when the domain decomposition is done and the product of $\mathbf{E}\Phi$ can be pre-computed. The update of nodal displacements is "local" as modes from other domains do not contribute to the final displacement of the node³. 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 position constraint equation is: $\tilde{\mathbf{R}}_p\Phi_p\mathbf{q} = \mathbf{c}$, where $\tilde{\mathbf{R}}_p$ and Φ_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

3. Each three-row block in the global mode matrix (Φ) 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} .

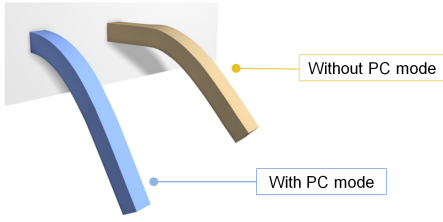


Fig. 6. Enforcing anchor nodes on the left end of the bar model using the Lagrange multiplier method: we can clearly see the locking region without using PC modes. Both bars are simulated with 100 modes in total.

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:

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

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 3 DOFs from the system. Therefore, a mechanism is needed to prevent the system from being over-constrained 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.

In order to solve this issue, we design a new type of mode called the *position constraint* (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 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:

$$\begin{bmatrix} \mathbf{K}_{II} & \mathbf{K}_{IC} & \mathbf{K}_{IB} \\ \mathbf{K}_{CI} & \mathbf{K}_{CC} & \mathbf{K}_{CB} \\ \mathbf{K}_{BI} & \mathbf{K}_{BC} & \mathbf{K}_{BB} \end{bmatrix} \begin{bmatrix} \Phi_I^{PC} \\ \mathbf{I}_C^{PC} \\ \mathbf{0} \end{bmatrix} = \begin{bmatrix} \mathbf{0} \\ \mathbf{F}_C^{PC} \\ \mathbf{F}_B^{PC} \end{bmatrix}, \quad (19)$$

and PC modes Φ^{PC} can be computed through:

$$\Phi^{PC} = \begin{bmatrix} \Phi_I^{PC} \\ \mathbf{I}_C^{PC} \\ \mathbf{0} \end{bmatrix} = \begin{bmatrix} -\mathbf{K}_{II}^{-1} \mathbf{K}_{IC} \\ \mathbf{I}_C^{PC} \\ \mathbf{0} \end{bmatrix}, \quad (20)$$

where \mathbf{I}_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}, θ) . \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 $\mathbf{u} = \tilde{\Phi}_p^{rot} \mathbf{q}_{\mathcal{B}_i}^{rot} = \theta [\mathbf{a}]_{\times} \mathbf{p}$, 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:

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

and

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

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

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

This means that, for the rotation \mathbf{R} with axis-angle representation as (\mathbf{a}, θ) , 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:

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

All the constraints including boundary constraint (Eq. 16), position constraint (Eq. 18) and interface constraint (Eq. 24) are imposed through Lagrange multiplier method. The 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-Lagrange equations of all the domains can also be converted to a linear system which 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 which is dropping under gravity (-9.8 m/s^2) with one end fixed. As shown in Fig. 7(a), 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

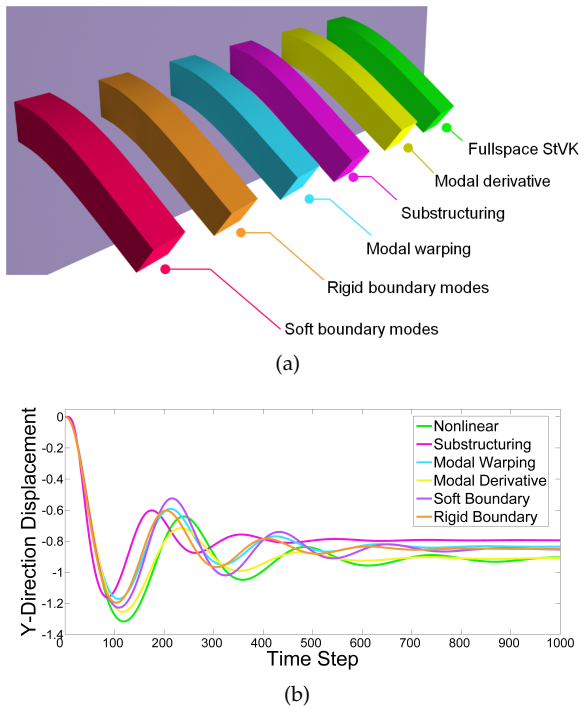


Fig. 7. Comparative simulation among various simulators.

substructuring [2] (nonlinear multi-domain 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 multi-domain simulators. All the simulators use the same material parameters (e.g., Young’s Modulus and Poisson’s Ratio). In order 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. 7(b). 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% for modal derivative, 13.21% for modal warping, 13.04% for substructuring (the displacements from duplicated boundary DOFs are averaged), and 10.35% and 8.45% 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 avoid-

s 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 *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. p

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 interfaces as the deformation and interface region goes large immediately after the simulation. Post-simulation 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 post-simulation amendments. More importantly, simple blending could still introduce artifacts of discontinuity as shown in Fig. 2 which requires more advanced geometrical blending at post-simulation 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 one, 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 two are assigned to the canvases to have enriched local deformation

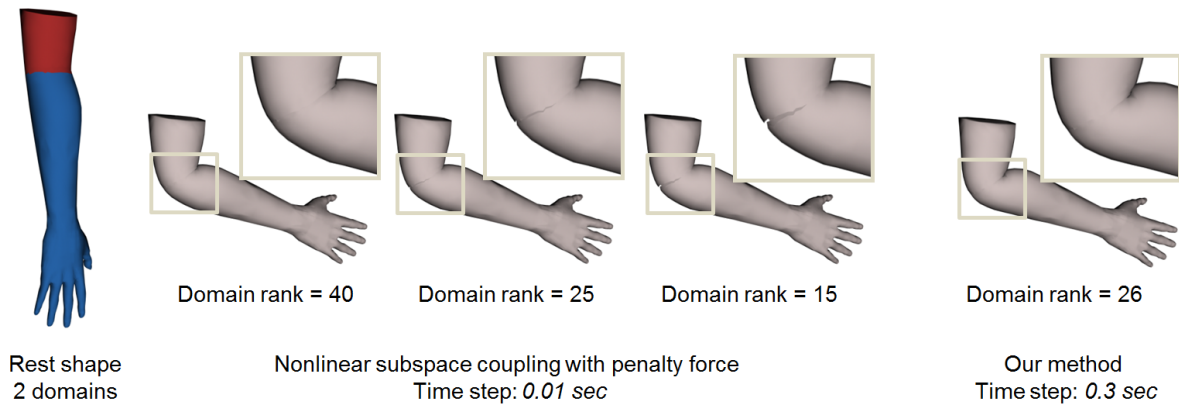


Fig. 8. Coupling comparison between our method and nonlinear multi-domain with penalty method [1].

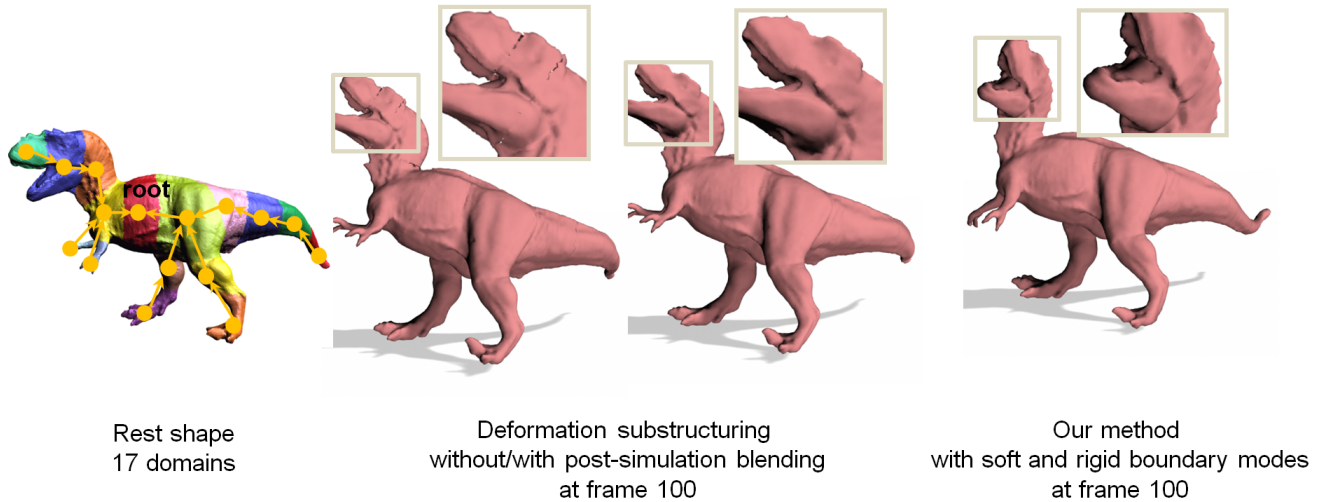


Fig. 9. Coupling comparison between our method and deformation substructuring using rigid interface fitting [2].

effects with the applied wind field. Domain loop can also be handled with spring coupling [1]. However, for looped multi-domain 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 ten 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 manipulated. In our implementation, a damped constraint equation [44] is used in order to avoid sharp impulse-like constraint forces and increase the stability of the system.

Performance Detailed time performances can be found in Tab. 2. Our system is implemented with a Windows 7 PC equipped with a 2.26GHz Intel Xeon CPU (only a single core is used for simulation) and 12GB RAM. Because the domains are coupled with hard constraints, we must solve the system entirely (similar to coupling with implicit penalty forces as



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

in [1]) instead of sequentially solving each domain individually [2]. However, linear elasticity with a constant system matrix needs only to be pre-conditioned once before simulation, and solving the system is not the bottleneck of the framework. For example in our sailboat model, solving the pre-factorized linear system takes less than $1ms$ which is only 5% computation time of one time step. The global mode matrix Φ has a block-diagonal structure and the run-time nodal displacement updates (e.g., using Eq. 17 to compute the nodal displacement on the mesh) are actually “local” as it only depends on the number of modes in the do-

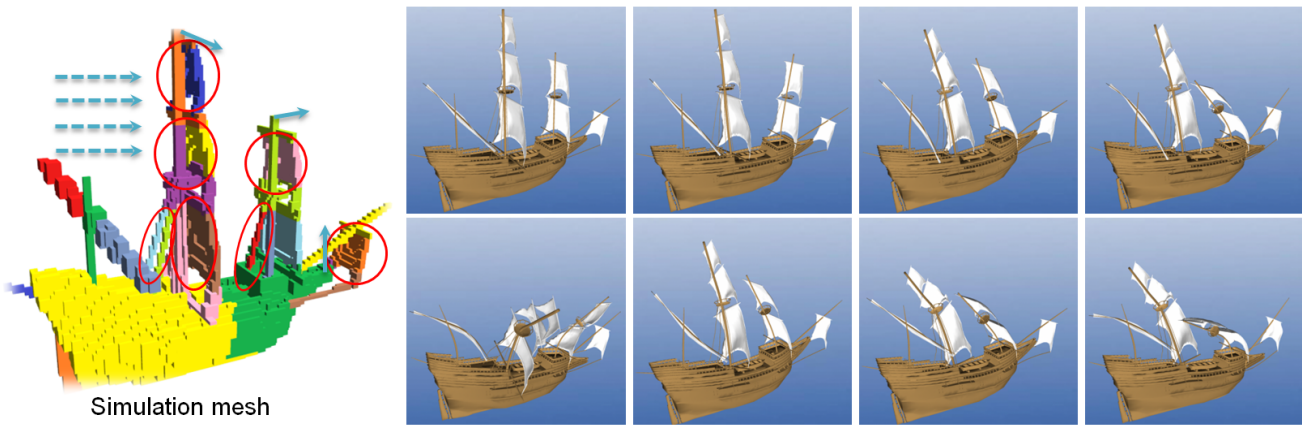


Fig. 10. Simulation of a sailboat model with 253,998 elements, 24 domains and seven loops (highlighted with red circles) in real-time (21 FPS). The total number of modes used is 830. The domains corresponding to the masts and the body of the boat have stiffer material while the canvases have softer material. 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).

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

TABLE 2

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

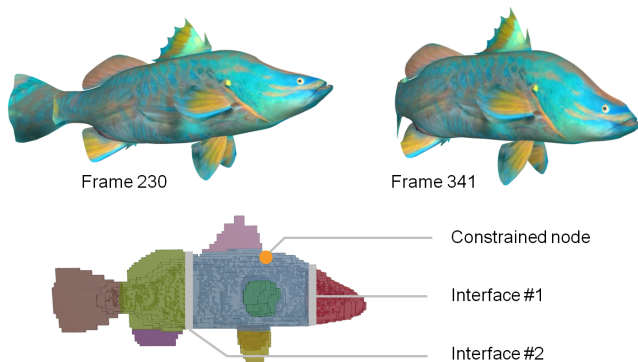


Fig. 12. Interactive manipulation on the fish model.

main 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 pre-computation, our method is significantly faster than 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 pre-

computation is orders of magnitude faster and can be finished within seconds. In the table, we record the pre-computation 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 pre-factorize \mathbf{K}_{II} before mode computation and the pre-computation for domain modes can be further shortened.

7 CONCLUSION

We have developed a seamless coupling method for multi-domain subspace deformation in the framework of node-wise 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. Node-wise 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 position constraints). 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% 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 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 multi-domain subspace deformations using 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 Eq. 1.

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

ACKNOWLEDGMENT

The author would like to thank anonymous reviewers for their constructive comments and suggestions. Yin Yang and Xiaohu Guo are partially supported by the NSF of USA 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).

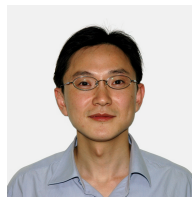
REFERENCES

- [1] T. Kim and D. L. James, "Physics-based character skinning using multi-domain subspace deformations," in *Symposium on Computer Animation*, 2011, pp. 63–72.
- [2] J. Barbič and Y. Zhao, "Real-time large-deformation substructuring," in *SIGGRAPH '11*, pp. 91:1–91:8.
- [3] R. R. Craig, "A review of time-domain and frequency-domain component mode synthesis methods," *International journal of 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. Visual. Comput. Graph.*, vol. 11, no. 1, pp. 91–101, 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," in *ACM SIGGRAPH 2008 classes*, ser. SIGGRAPH '08, 2008, pp. 88:1–88:90.
- [7] D. Terzopoulos, J. Platt, A. Barr, and K. Fleischer, "Elastically deformed models," in *SIGGRAPH '87*, 1987, pp. 205–214.
- [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," in *Symposium on Computer animation*, 2002, pp. 41–47.
- [10] X. Wu, M. S. Downes, T. Goktekin, and F. Tendick, "Adaptive nonlinear finite elements for deformable body simulation using dynamic progressive meshes," in *Computer Graphics Forum*, 2001, pp. 349–358.
- [11] E. Grinspun, P. Krysl, and P. Schröder, "Charms: a simple framework for adaptive simulation," in *SIGGRAPH '02*, pp. 281–290.
- [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. Graph.*, 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," in *SIGGRAPH '07*.
- [14] L. Kharevych, P. Mullen, H. Owjadi, and M. Desbrun, "Numerical coarsening of inhomogeneous elastic materials," in *SIGGRAPH '09*, 2009, pp. 1–8.
- [15] M. Nesme, P. G. Kry, L. Jeřábková, and F. Faure, "Preserving topology and elasticity for embedded deformable models," in *SIGGRAPH '09*, pp. 1–9.
- [16] P. Kaufmann, S. Martin, M. Botsch, and M. Gross, "Flexible simulation of deformable models using discontinuous galerkin fem," *Graph. Models*, vol. 71, no. 4, pp. 153–167, Jul. 2009.
- [17] M. Müller, J. Dorsey, L. McMillan, R. Jagnow, and B. Cutler, "Stable real-time deformations," in *Symposium on Computer animation*, 2002, pp. 49–54.
- [18] M. Müller and M. Gross, "Interactive virtual materials," in *Proc. of Graphics Interface '04*, 2004, pp. 239–246.
- [19] O. Eitzmuss, M. Keckeisen, and W. Strasser, "A fast finite element solution for cloth modelling," in *Computer Graphics and Applications*, 2003. *Proceedings. 11th Pacific Conference on*, oct. 2003, pp. 244 – 251.
- [20] Y. Zhu, E. Sifakis, J. Teran, and A. Brandt, "An efficient multigrid method for the simulation of high-resolution elastic solids," *ACM Trans. Graph.*, 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," *SIGGRAPH*, vol. 29, no. 4.
- [22] J. Barbič, F. Sin, and E. Grinspun, "Interactive editing of deformable simulations," *ACM Trans. Graph.*, vol. 31, no. 4, pp. 70:1–70:8, Jul. 2012.
- [23] M. G. Choi, S. Y. Woo, and H.-S. Ko, "Real-time simulation of thin shells," in *Proc. of Eurographics '07*, 2007, pp. 349–354.
- [24] X. Guo and H. Qin, "Real-time meshless deformation: Collision detection and deformable objects," *Comput. Animat. Virtual Worlds*, vol. 16, no. 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. Graph.*, vol. 31, no. 5, pp. X:1–13, Oct. 2012.
- [26] A. Pentland and J. Williams, "Good vibrations: modal dynamics for graphics and animation," *SIGGRAPH '89*, vol. 23, no. 3, pp. 207–214, 1989.
- [27] J. F. O'Brien, P. R. Cook, and G. Essl, "Synthesizing sounds from physically based motion," in *SIGGRAPH '01*, pp. 529–536.
- [28] D. L. James and D. K. Pai, "DyRT: dynamic response textures for real time deformation simulation with graphics hardware," in *SIGGRAPH '02*, 2002, pp. 582–585.
- [29] P. G. Kry, D. L. James, and D. K. Pai, "Eigenskin: real time large deformation character skinning in hardware," in *Symposium on Computer animation*, 2002, pp. 153–159.
- [30] K. Hauser, C. Shen, and J. O'Brien, "Interactive deformation using modal analysis with constraints," in *Proc. of Graphics Interface '03*, 2003, pp. 247–255.
- [31] J. Barbič and D. L. James, "Real-time subspace integration for St. Venant-Kirchhoff deformable models," *SIGGRAPH '05*, vol. 24, no. 3, pp. 982–990, Aug. 2005.
- [32] T. Kim and D. L. James, "Skipping steps in deformable simulation with online model reduction," *ACM Trans. Graph.*, vol. 28, pp. 123:1–123:9, December 2009.
- [33] M. Müller, B. Heidelberger, M. Teschner, and M. Gross, "Meshless deformations based on shape matching," in *ACM SIGGRAPH 2005 Papers*, ser. SIGGRAPH '05, pp. 471–478.
- [34] M. Müller and N. Chentanez, "Solid simulation with oriented particles," *ACM Trans. Graph.*, vol. 30, no. 4, pp. 92:1–92:10, Jul. 2011.
- [35] J. Huang, X. Liu, H. Bao, B. Guo, and H.-Y. Shum, "An efficient large deformation method using domain decomposition," *Comput. Graph.*, vol. 30, no. 6, pp. 927–935, 2006.
- [36] B. Vallet and B. Lévy, "Spectral geometry processing with manifold harmonics," *Computer Graphics Forum (Proceedings Eurographics)*, vol. 27, no. 2, pp. 251–260, 2008.
- [37] G. Rong, Y. Cao, and X. Guo, "Spectral mesh deformation," *Vis. Comput.*, vol. 24, no. 7, pp. 787–796, Jul. 2008.
- [38] K. Hildebrandt, C. Schulz, C. V. Tycowicz, and K. Polthier, "Interactive surface modeling using modal analysis," *ACM Trans. Graph.*, vol. 30, no. 5, pp. 119:1–119:11, Oct. 2011.
- [39] E. Sifakis and J. Barbič, "Fem simulation of 3d deformable solids: a practitioner's guide to theory, discretization and model reduction," in *ACM SIGGRAPH 2012 Courses*, 2012, pp. 20:1–20:50.
- [40] P. Kaufmann, S. Martin, M. Botsch, and M. Gross, "Flexible simulation of deformable models using discontinuous galerkin fem," *Graph. Models*, vol. 71, no. 4, pp. 153–167, Jul. 2009.
- [41] S. S. An, T. Kim, and D. L. James, "Optimizing cubature for efficient integration of subspace deformations," in *SIGGRAPH Asia '08*, pp. 165:1–165:10.
- [42] D. Baraff, "Linear-time dynamics using lagrange multipliers," ser. SIGGRAPH '96, 1996, pp. 137–146.
- [43] T. Hughes, *The finite element method: linear static and dynamic finite element analysis*, ser. Dover Civil and Mechanical Engineering Series. Dover Publications, 2000.
- [44] J. Baumgarte, "Stabilization of constraints and integrals of motion in dynamical systems," *Computer Methods in Applied Mechanics and Engineering*, vol. 1, no. 1, pp. 1 – 16, 1972.
- [45] D. Terzopoulos and A. Witkin, "Physically based models with rigid and deformable components," *IEEE Comput. Graph. Appl.*, vol. 8, no. 6, pp. 41–51, 1988.



Yin Yang is currently a Ph.D student and research assistant with 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>.



Dr. Weiwei Xu is a professor at Hangzhou Normal University in the department of computer science. Before that, Dr. Weiwei Xu was a researcher in Internet Graphics Group, Microsoft Research Asia from Oct. 2005 to Jun 2012. Dr. Weiwei Xu has been a post-doc researcher at Ritsmeikan university in Japan around one year from 2004 to 2005. He received Ph.D. Degree in Computer Graphics from Zhejiang University, Hangzhou, and B.S. Degree and Master Degree in Computer Science from Hohai University in 1996 and 1999 respectively. He has published more than 20 papers on international conference and journals, include 9 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, etc. He received the prestigious 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 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. He received his BS degree and PhD degree in computer science from Zhejiang University in 1997 and 2002, respectively. His research interests include shape modelling/editing, texture mapping/synthesis, real-time rendering and GPU parallel computing.



Dr. Baining Guo is a 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. Dr. Guo graduated from Beijing University with B.S. in mathematics. He went to Cornell University for his graduate study from 1986 to 1991 and obtained M.S. in Computer Science and Ph.D. in Applied Mathematics. Dr. Guo is an IEEE fellow and an ACM fellow.

Dr. Guo's 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. Dr. Guo 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 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. Dr. Guo holds over 40 US patents.