Available online at www.sciencedirect.com
ScienceDirect
Graphics Models 00 (2015) 1–12
Graphics Models

Interactive Design and Simulation of Tubular Supporting Structure

Ran Luo[b], Lifeng Zhu*[a], Weiwei Xu[c], Patrick Kelley[b], Vanessa Svihla[b], Yin Yang[b]

[a]*School of Instrument Science and Engineering, Southeast University*
[b]*University of New Mexico*
[c]*Hangzhou Normal University*

Abstract

This paper presents a system for design and simulation of supporting tube structure. We model each freeform tube component as a swept surface, and employ boundary control and skeletal control to manipulate its cross-sections and its embedding respectively. With the parametrization of the swept surface, a quadrilateral mesh consisting of nine-node general shell elements is automatically generated and the stress distribution of the structure is simulated using the finite element method. In order to accelerate the complex finite element simulation, we adopt a two-level subspace simulation strategy, which constructs a secondary complementary subspace to improve the subspace simulation accuracy. Together with the domain decomposition method, our system is able to provide interactive feedback for parametric freeform tube editing. Experiments show that our system is able to predict the structural character of the tube structure efficiently and accurately.

Keywords: , Thin shell, Swept surface, FEM, Model reduction, Domain decomposition

1. Introduction

Tubes serve as a type of important supporting structure and are commonly used in people's everyday life (Fig. 1). Traditionally, such supporting structures are often hollow to conserve the manufacture cost and self-weight. They mostly consist of regular cylinders, which are more budget-friendly for mass production with traditional manufacturing techniques. On the other hand, the rapid development of prototype technology (e.g. 3D printing) makes personalized and customized tube fabrication using generalized cylinders conveniently possible, which greatly expands the designing space of supporting tubes.

Image sources:
http://journeyeast.com/products/tubeline-stool
http://www.pinterest.com/robertpatterson/pipe-furniture

Figure 1. Two furniture designs using supporting tubes.

Although most existing computer-aided design (CAD) softwares (e.g. AutoCAD) well support the geometric design of such tubular structures. Users still need to manipulate many geometric degrees of freedom (DOFs) to model freeform tubes. Interpolatory and tangential controls at the boundary cross-sections are two widely-adopted mechanisms to control the shape of the tube. However, profile control [1] or solving higher-order differential equations [2] is still necessary to prevent shape distortion, which is often tedious or time-consuming. On the other hand, existing CAD packages merely focus on the aspect of shape editing while the structural properties of the 3D model remain unknown to novice users. Following the trend of design-simulation integration, current commercial produces start to enable user to analyze their design using finite element method. Unfortunately, an accurate simulation of the structural characteristic of a customized tubular structure is expensive because generalized shell element with high-order shape functions is usually required to avoid shear-locking artifact [3] and a simulator often possesses a large number DOFs that is prohibitive to regular desktop computers, not to mention performing interactive structural analysis of the 3D model being edited.

As a response to the aforementioned challenges, we present a system for interactive design and simulation of supporting

tubular structures. Tube components can be intuitively edited using *boundary* and *skeletal* controls and a complex tube system can be handily created by assembling tube components at their open interfaces. The underlying simulation is carried out using general quadratic nine-node quadrilateral element. A constraint subspace is constructed at each component, which serves as the primary subspace for the follow-up structural analysis. On the top of the constraint subspace, we build a load-dependent secondary subspace named residual subspace, which is able to precisely capture the detailed intra-component deflection due to the regional external loads without resorting to expensive full-space simulation. As a result, our system is able to provide interactive yet accurate structural analysis along with the editing operation of the tube.

Contribution In general, the contribution of our work can be briefly summarized as follows:

- This paper presents a system integrating parametric shape editing and finite element method based structural analysis into a unified environment for the design and simulation of freeform tubular supporting structures.
- We provide user an intuitive shape design mechanism with lower geometric DOFs by using the boundary and skeletal controls to manipulate the geometry of each tube component.
- A new simulation strategy is proposed based on the fact that the supporting tube is often of light self-weight comparing to its external loads. We use a two-level subspace simulation that is able to accurately capture the deflection induced by external loads while still keep the simulator compact.

2. Related Work

Swept surface is often used to model general cylinders [4] by transforming cross-section curves along a smooth rotation field on a swept trajectory. Topics such as how to design smooth rotation field on a given trajectory [5, 6], how to interpolate cross-section curves [7, 8] and how to support profile editing [9, 10] are all well studied in the literature. However, it is tedious to manipulate lots of control vertices of swept surfaces represented by standard tensor product spline surfaces. Recently, You et al. [1] suggest modeling swept surfaces by solving ordinary differential equations (ODE). They showed that interpolatory and tangential boundary controls are available by using fourth-order ODE, which leads to lower DOFs in controlling the shape of swept surfaces. They also derived analytical solutions to six-order partial differential equations and gave extra curvature control to swept surfaces [11]. The similar idea is also exploited in shape modeling using meshed surfaces. In [12, 2], the authors showed that generalized cylinders can be obtained by solving harmonic and higher-order harmonic equations. In our system, the geometric design of freeform supporting tubes is motivated by these existing studies.

Thin shell element is a natural choice for the tubular structure, which has a high width-thickness ratio. Such degeneracy motivates researchers, especially in graphics community, to seek for alternative energy models to capture the deformation of thin shell in a more efficient and intuitive manner such as spline/NURBS [13, 14, 15], hinge-based bending [16, 17, 18, 19], or meshless method [20, 21, 22], rather than resorting to classic strain theory [23]. Zhang et al. [24] proposed to use 1D orientated rod element with incremental strain theory to model the thin shell structure, which could be considered as an extended version of mass-spring system. While compelling results have been reported, these methods only produce physically plausible animations while we are looking for an accurate simulation that directly serves for potential follow-up fabrication (e.g. via 3D printing).

Design-simulation integration has received increased attention recently and fabrication-purposed design system becomes an active research topic. Simulation based optimization has been widely applied to make sure the fabricated object possesses the desired structural robustness [25, 26, 27], kinematic constraints [28, 29], and deformable behavior [30, 31, 32]. There are also many contributions trying to unify the simulation and the design processing. Umetani et al. [33] present a garment designing system that allows an interactive editing between 2D patterns and 3D simulated draped forms. Cirak et al. [34] propose to use subdivision surface for the design-simulation integration for thin-shell objects.

Simulation acceleration stands out a grand technical challenge for the integration of design simulation because an accurate finite element method (FEM) [3] simulation is often expensive while timely-coupled design-simulation environment is always favored. To accelerate the FEM simulation of thin shell, Seth et al. [35] employ a multi-resolution framework. In regular FEM simulation of 3D solid volume, subspace modal reduction is a widely used technique [36, 37, 38] and it can also be applied to accelerate thin-shell simulation [39].

Our method well complements exiting contributions by developing a design-simulation framework based on domain decomposition [40] and finite element tearing and interconnect (FETI) method [41], as we notice that tubular structures are often component-wise and the geometric symmetry commonly exists. The geometry of each tube component is dealt with using boundary and skeletal controls. The structural behavior is simulated using quadratic nine-node quadrilateral mesh automatically generated via the surface parametrization, which will assign five DOFs for each free node. To accelerate the simulation, we construct the component-level subspace based on an engineering technique named *component mode synthesis* (CMS) [42, 43]. Improved simulation accuracy is achieved by computing the residual deflection within a load-dependent secondary subspace. As a result, our system is able to provide accurate stress analysis while keeping the simulator compact and efficient.

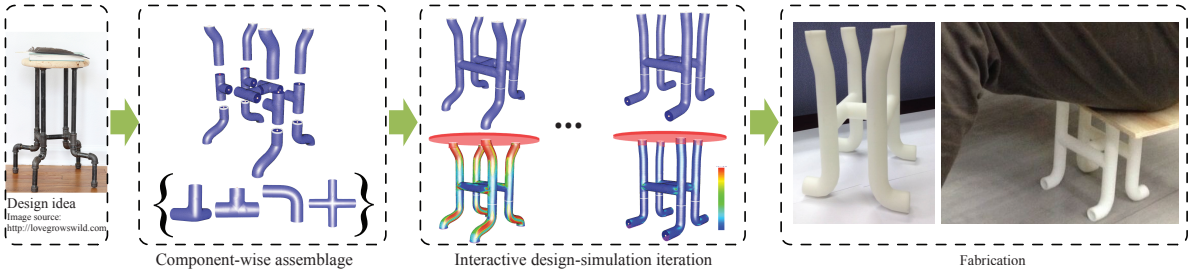

Figure 2. An overview of the proposed design-simulation system.

3. System Overview

Fig. 2 sketches an overview of the proposed design-simulation system. The entire structure is composed of multiple tubular *components* which are inter-connected at their interfaces. The shape of each component is modeled as a swept surface that can be freely edited with an intuitive interface that allows users to manipulate key cross-section curves and their trajectory (Sec. 4). Based on the parameterization, a quadrilateral finite element mesh is automatically generated. Each of its element is a nine-node quadratic shell element (Sec. 5), where the mid-edge nodes are determined using cubic Hermite interpolation. We adopt a two-level subspace simulation strategy to accelerate the simulation so that an interactive structural analysis is made possible (Sec. 6) and the stress distribution can be timely visualized by the designer to ensure the tubular model is robust and stable under the prescribed external loads.

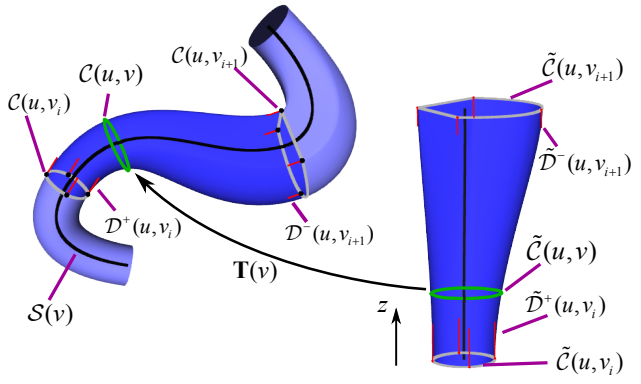

Figure 3. The boundary conditions and sweep trajectory of a freeform tube. The designed freeform tube is on the left and its shape space of cross-sections is shown on the right.

4. Geometric Design of Freeform Tubes

Our goal of the geometric design is to provide users intuitive and flexible controls of freeform tubular components, which are modeled using swept surfaces in our system. Similar to previous works [1], we use boundary constraints to manipulate geometric variation of the cross-section of the tube along the neutral axis instead of using control vertices for the tensor product spline surfaces, since it is intuitive and requires fewer control parameters. Instead of profile editing introduced in [1], we employ the sweep trajectory to control the design in our system, or namely *skeletal control*. Comparing to multiple profile curves, one freeform tube only have one sweep trajectory, which significantly eases the overall shape editing operation. Specifically, as shown in Fig. 3, our system allows users to control 1) the sweep trajectory $\mathcal{S}(v)$, 2) a few key cross-section curves $\mathcal{C}_i(u) = \mathcal{C}(u,v_i)$ at $\mathcal{S}(v_i)$, and 3) each key cross-section curve's variation at both extensions $\mathcal{D}^-(u,v_i)$ and $\mathcal{D}^+(u,v_i)$, such that $\mathcal{D}^-(u,v_i)$ and $\mathcal{D}^+(u,v_i)$ are the tangent directions. The output swept surface $\mathcal{C}(u,v)$ at $[v_i,v_{i+1}]$ must satisfy the following conditions:

$$\mathcal{C}(u,v_i) = \mathcal{C}_i(u), \qquad \frac{\partial \mathcal{C}(u,v_i)}{\partial v} = \mathcal{D}^+(u,v_i)$$
$$\mathcal{C}(u,v_{i+1}) = \mathcal{C}_{i+1}(u), \qquad \frac{\partial \mathcal{C}(u,v_{i+1})}{\partial v} = \mathcal{D}^-(u,v_{i+1})$$

You et al. [1] presented a formulation to construct the swept surface with the above constraints by solving a fourth-order ODEs. Unfortunately, we found that this approach yields unpleasant shape distortion if the boundary tangents are not in parallel, as shown in Fig.4 (a). To prevent such distortion, extra control information must be provided from the user such as profile control [1] or curvature control [2, 11], which inevitably induces more editing freedoms and could potentially make novice users confusing.

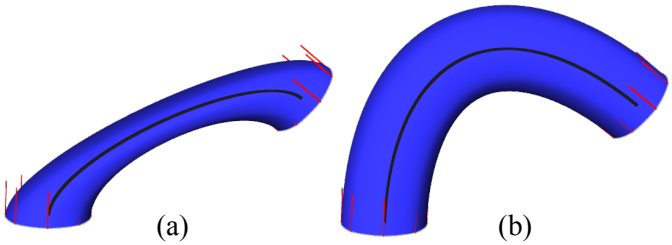

Figure 4. Boundary constrained swept surfaces using ODE-based techniques without (a) and with (b) sweep trajectory control.

Alternatively, we combine the *rotation minimizing frames* (RMF) [6] together with the existing ODE-based techniques. The geometry of the swept surface is decoupled into two parts: the shape defined by cross-sections and their 3D embedding defined by sweep trajectories. In our system, we constrain each

cross-section curve $\mathcal{C}(u,v_i)$ to be planar and the editing operations associated with cross-section curves are performed with an intuitive 2D user interface. Suppose a rigid transformation $\mathbf{T}(v)$ maps a planar curve $\tilde{\mathcal{C}}(u,v)$ to $\mathcal{C}(u,v)$, we call $\tilde{\mathcal{C}}(u,v_t)$ the shape of cross-section at $v=v_t$. Given a sparse set of key cross-section shapes $\tilde{\mathcal{C}}_i(u)=\tilde{\mathcal{C}}(u,v_i)$ for $i=1,\cdots,n$ and a sweep trajectory $\mathcal{S}(v)$, we are computing for interpolated cross-section shapes $\tilde{\mathcal{C}}(u,v)$ and transformations $\mathbf{T}(v)$ for all $v_i\in[v_1,v_n]$. The transformation $\mathbf{T}(v)$ is composed of a translation $\mathbf{t}(v)$ and a rotation $\mathbf{R}(v)$. We define the translation $\mathbf{t}(v)=\mathcal{S}(v)$, to make the cross-section curves sweep coincide with the specified trajectory $\mathcal{S}(v)$. The rotation $\mathbf{R}(v)$ is computed by using the RMF on $\mathcal{S}(v)$. Besides interpolatory constraints, we also allow users to control the tangent in the shape space of cross-sections. Specifically, we embed all planar cross-section shapes in 3D by defining v as the height (z direction) of $\tilde{\mathcal{C}}(u,v)$, as shown in Fig. 3. The tangential controls $\tilde{\mathcal{D}}^+(u,v_i)$ and $\tilde{\mathcal{D}}^-(u,v_i)$ in the shape space of cross-sections are also defined in this embedding. $\tilde{\mathcal{C}}(u,v)$ is solved with the following tangential constraints:

$$\tilde{\mathcal{C}}(u,v_i)=\mathcal{C}_i(u),\qquad \frac{\partial\tilde{\mathcal{C}}(u,v_i)}{\partial v}=\tilde{\mathcal{D}}^+(u,v_i)$$
$$\tilde{\mathcal{C}}(u,v_{i+1})=\mathcal{C}_{i+1}(u),\qquad \frac{\partial\tilde{\mathcal{C}}(u,v_{i+1})}{\partial v}=\tilde{\mathcal{D}}^-(u,v_{i+1}).$$

We take the shape on the plane $z=v_j$ as the interpolated cross-section at $v=v_j$. Note that in our system, $\tilde{\mathcal{D}}^-(u,v_i)$ and $\tilde{\mathcal{D}}^+(u,v_i)$ provide only tangential control in the shape space of cross-sections. They are not the tangents on the swept surface if the sweep trajectory is not straight. In our implementation, we set $\tilde{\mathcal{D}}^{\pm}(u,v_i)=\mathbf{R}^\top(v)\mathcal{D}^{\pm}(u,v_i)$, where $\mathbf{R}(v)$ is the rotation of the RMF at v.

As shown in Fig. 4 (b), with the same boundary conditions, swept surfaces using our method are free of distortion. In this example, the sweep trajectory is defined by a cubic Bezier curve whose end points and tangents are the same with the input boundary conditions. If profile control is still required, users can simply insert extra key cross-section curves to control the profile. Note that although existing commercial products support cross-section sketching followed by sweep path design, tangential control is mostly applied on the sweep trajectory but not on the surface. An alternative way is to directly manipulate on the control points, which leads to heavier data and interactions.

Since the cross-sections are interpolated analytically and the RMFs are computed explicitly, the computation associated with editing operations is negligible. Please note that our current system does not explicitly handle the self-intersection, which could be possible for highly curved tubes. An alert will be sent as soon as the user's editing leads to any self-collisions or intersections.

5. Formulation of General Shell Element

Shell element is a degenerated structural element. Unlike regular 3D solid volumetric elements such as brick or tetrahedra, the dimension along its thickness is much smaller (e.g. over 50 times) than the other two dimensions. Such degeneracy leads serious numerical stability issue. As a result, necessary geometric/kinematic constraints and simplifications must be assumed. This section will briefly explain the finite element formulation of the nine-node quadrilateral shell element. We refer readers to the related literature [3, 44], for a more detailed derivation.

The parametrization of the swept surface handily generates a quadrilateral mesh. Each four-node quadrilateral will be converted into a nine-node quadratic shell element. The extra mid-edge nodes are determined by using cubic Hermite interpolation so that the resulting nine-node element has smoothly-curved edges. For the ease of derivation, we adopt the *isoparametric formulation*, which begins with a standard shell element (Fig. 5) defined in a virtual *rst* coordinate frame or *natural coordinate frame*. This element spans from -1 to 1 in both r and s directions. For elements with arbitrary location and geometry, we take use of the Jacobian matrix to map it back to the real coordinate system.

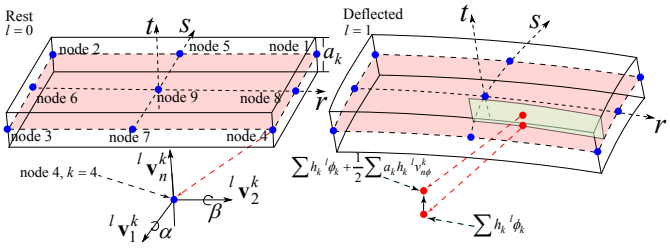

Figure 5. A standard nine-node element in natural coordinate frame.

For an arbitrary mass point within the element, all of its kinematic terms are interpolated using nodal shape functions. Shape functions always have ones at their host nodes and vanished values at the other nodes. High-order shape function can be considered as the superposition of the scaled low-order ones. Based on this property, the shape function of this standard element can be easily written as:

$$\begin{aligned}
h_1 &= \tfrac{1}{4}(1+r)(1+s)-\tfrac{1}{2}h_5-\tfrac{1}{2}h_8-\tfrac{1}{4}h_9\\
h_2 &= \tfrac{1}{4}(1+r)(1+s)-\tfrac{1}{2}h_5-\tfrac{1}{2}h_6-\tfrac{1}{4}h_9\\
h_3 &= \tfrac{1}{4}(1+r)(1+s)-\tfrac{1}{2}h_6-\tfrac{1}{2}h_7-\tfrac{1}{4}h_9\\
h_4 &= \tfrac{1}{4}(1+r)(1+s)-\tfrac{1}{2}h_7-\tfrac{1}{2}h_8-\tfrac{1}{4}h_9\\
h_5 &= \tfrac{1}{2}(1+r)(1+s)-\tfrac{1}{2}h_9\\
h_6 &= \tfrac{1}{2}(1+r)(1+s)-\tfrac{1}{2}h_9\\
h_7 &= \tfrac{1}{2}(1+r)(1+s)-\tfrac{1}{2}h_9\\
h_8 &= \tfrac{1}{2}(1+r)(1+s)-\tfrac{1}{2}h_9\\
h_9 &= (1-r^2)(1-s^2).
\end{aligned}\qquad (1)$$

Fig. 5 shows a nine-node shell element in the rest (left) and deflected (right) configurations respectively, indicated with the superscript l. When $l=0$, the corresponding variable is at the rest configuration; when $l=1$, the corresponding variable is at the deflected configuration. A unit vector ${}^l\mathbf{v}_n^k=[{}^lv_{nx}^k,{}^lv_{ny}^k,{}^lv_{nz}^k]^\top$ is defined in the regular xyz coordinate frame at node k, which corresponds to the tangent direction of t axis in *rst* frame. ${}^l\mathbf{v}_1^k$ and ${}^l\mathbf{v}_2^k$ are two mutually perpendicular unit vectors sitting the plane normal to ${}^l\mathbf{v}_n^k$. The infinitesimal rotations around ${}^l\mathbf{v}_1^k$ and ${}^l\mathbf{v}_2^k$ are denoted as α and β, which serve as extra two DOFs of

node k[1]. Therefore, each node possesses five DOFs i.e. x_k, y_k, z_k, α_k and β_k. Let $p(r,s,t)$ be an arbitrary mass point within the element. Its rest/deflected position can be interpolated as:

$$ {}^{l}\phi(r,s,t) = \sum_k h_k {}^{l}\phi_k + \frac{1}{2}\sum_k a_k h_k {}^{l}v^k_{n\phi}, \quad \phi = x,y \text{ or } z, \tag{2} $$

where a_k is the thickness of the shell at node k. The first term in Eq. 2 corresponds to the regular shape function interpolation and the second term (i.e. $\frac{1}{2}\sum a_k h_k {}^{l}v^k_{n\phi}$) assumes that r and s displacements at $p(r,s,t)$ are linearly proportional to its t coordinate when r and s are fixed. The displacement vector $\mathbf{u}(r,s,t) = [u,v,w]^\top$ at p can be interpolated in a similar fashion:

$$ \begin{aligned} u(r,s,t) &= \sum_k h_k u_k + \frac{1}{2}\sum_k a_k h_k v^k_{nx} \\ v(r,s,t) &= \sum_k h_k v_k + \frac{1}{2}\sum_k a_k h_k v^k_{ny} \\ w(r,s,t) &= \sum_k h_k w_k + \frac{1}{2}\sum_k a_k h_k v^k_{nz} \end{aligned} \tag{3} $$

where v^k_{nx}, v^k_{ny} and v^k_{nz} are the three components of the displacement vector $\mathbf{v}^k_n$ from ${}^{0}\mathbf{v}^k_n$ to ${}^{1}\mathbf{v}^k_n$ (i.e. $\mathbf{v}^k_n \triangleq {}^{1}\mathbf{v}^k_n - {}^{0}\mathbf{v}^k_n$). It can be expressed using nodal DOFs α_k and β_k such that:

$$ \mathbf{v}^k_n = -{}^{0}\mathbf{v}^k_2\alpha_k + {}^{0}\mathbf{v}^k_1\beta_k. \tag{4} $$

Substituting Eqs. 3 and 4 into Eq. 2 leads to the final interpolation of the shell element:

$$ \begin{bmatrix} u \\ v \\ w \end{bmatrix} = \sum_k \mathbf{H}_k \begin{bmatrix} u_k \\ v_k \\ w_k \\ \alpha_k \\ \beta_k \end{bmatrix}, \tag{5} $$

where

$$ \mathbf{H}_k = \left[\begin{array}{ccc|c|c} h_k & & & -h_k\tfrac{t}{2}a_k {}^{0}v^k_{2x} & h_k\tfrac{t}{2}a_k {}^{0}v^k_{1x} \\ & h_k & & -h_k\tfrac{t}{2}a_k {}^{0}v^k_{2y} & h_k\tfrac{t}{2}a_k {}^{0}v^k_{1y} \\ & & h_k & -h_k\tfrac{t}{2}a_k {}^{0}v^k_{2z} & h_k\tfrac{t}{2}a_k {}^{0}v^k_{1z} \end{array}\right]. \tag{6} $$

The element stiffness matrix is in the format of:

$$ \mathbf{K}_e = \int_{V_e} \mathbf{B}^\top \mathbf{C}\mathbf{B} \mathrm{d}v, \tag{7} $$

where $\mathbf{B}$ is the strain-displacement matrix obtained by applying the partial derivative to nodal displacements using Eq. 5. $\mathbf{C}$ is the stress-strain matrix determined by the material property. In this paper, linear elasticity is assumed and $\mathbf{C}$ is constant.

Due to the degeneracy of the shell element along its t direction, the stress components normal to the midsurface must always be zero. As a result, we need transforms the $\mathbf{C}$ matrix to make it aligned with the orientation of the shell element. As shown in Fig. 6, let $\mathbf{r}$, $\mathbf{s}$, $\mathbf{t}$ be the unit vectors corresponding to three axes in the natural coordinate frame. They may be distorted and no longer orthogonal to each other when being viewed from the regular xyz coordinate frame. Let $\mathbf{r}'$, $\mathbf{s}'$, $\mathbf{t}'$ be the unit vectors at the tangent directions of the corresponding axis (note that $\mathbf{t}' = \mathbf{t}$). We can extract a new set of basis vectors $\bar{\mathbf{r}}$, $\bar{\mathbf{s}}$ and $\bar{\mathbf{t}}$ such that: $\bar{\mathbf{r}} = \frac{\mathbf{s}'\times\mathbf{t}'}{||\mathbf{s}'\times\mathbf{t}'||_2}$, $\bar{\mathbf{s}} = \frac{\mathbf{t}'\times\bar{\mathbf{r}}}{||\mathbf{t}'\times\bar{\mathbf{r}}||_2}$, and $\bar{\mathbf{t}} = \frac{\mathbf{t}'}{||\mathbf{t}'||_2}$. Afterwards, a transformation matrix $\mathbf{Q}_{sh} \in \mathbb{R}^{6\times6}$ can be constructed from the direction cosines of the $\bar{\mathbf{r}}$, $\bar{\mathbf{s}}$ and $\bar{\mathbf{t}}$ measured in xyz coordinate frame, such that:

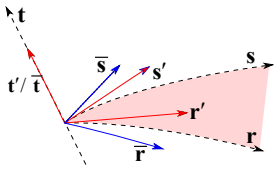

Figure 6. Extracting orthogonal basis vectors $\bar{\mathbf{r}}$, $\bar{\mathbf{s}}$ and $\bar{\mathbf{t}}$.

$$ \mathbf{Q}_{sh} = \begin{bmatrix} l_1^2 & m_1^2 & n_1^2 & l_1m_1 & m_1n_1 & n_1l_1 \\ l_2^2 & m_2^2 & n_2^2 & l_2m_2 & m_2n_2 & n_2l_2 \\ l_3^2 & m_3^2 & n_3^2 & l_3m_3 & m_3n_3 & n_3l_3 \\ ll_{1,2} & mm_{1,2} & nn_{1,2} & lm_{1,2} & mn_{1,2} & nl_{1,2} \\ ll_{2,3} & mm_{2,3} & nn_{2,3} & lm_{2,3} & mn_{2,3} & nl_{2,3} \\ ll_{3,1} & mm_{3,1} & nn_{3,1} & lm_{3,1} & mn_{3,1} & nl_{3,1} \end{bmatrix}, \tag{8} $$

where

$$ \begin{aligned} l_1 &= cos(\mathbf{x},\bar{\mathbf{r}}); \quad m_1 = cos(\mathbf{y},\bar{\mathbf{r}}); \quad n_1 = cos(\mathbf{z},\bar{\mathbf{r}}); \\ l_2 &= cos(\mathbf{x},\bar{\mathbf{s}}); \quad m_2 = cos(\mathbf{y},\bar{\mathbf{s}}); \quad n_2 = cos(\mathbf{z},\bar{\mathbf{s}}); \\ l_3 &= cos(\mathbf{x},\bar{\mathbf{t}}); \quad m_3 = cos(\mathbf{y},\bar{\mathbf{t}}); \quad n_3 = cos(\mathbf{z},\bar{\mathbf{t}}). \end{aligned} \tag{9} $$

The notation $ab_{i,j}$ denotes $a_ib_j + a_jb_i$, for instance $lm_{1,2} = l_1m_2 + l_2m_1$. The transformed $\mathbf{C}$ matrix is computed as:

$$ \mathbf{C}_{sh} = \mathbf{Q}_{sh}^\top \mathbf{C}\mathbf{Q}_{sh}. \tag{10} $$

The volumetric integral in Eq. 7 is evaluated numerically at 18 sampling points $p_i(r_i,s_i,t_i)$ according to *Gauss-Legendre* integration strategy:

$$ \mathbf{K}_e \approx \sum_i det(\mathbf{J}_i) w_i \mathbf{B}^\top \mathbf{C}_{sh}\mathbf{B}, \tag{11} $$

where $\mathbf{J}_i$ is the Jacobian matrix encoding how rst frame is transformed to xyz frame at p_i. w_i is the constant sampling weight of p_i. The r, s and t coordinates of p_i are selected from the combinations of $r = \pm0.77460, 0$, $s = \pm0.77460, 0$ and $t = \pm0.57735$.

6. Subspace Simulation Acceleration

We project the FEM simulator into a pre-constructed subspace to accelerate the associated computation in order to provide an interactive design-simulation interplay. It is well-known that the cost of subspace simulation acceleration is the accuracy compromise as the system response beyond the pre-defined subspace can not be captured. This problem is dealt with by using a secondary residual subspace, in our system, to accurately obtain the necessary intra-component deflection due to the external load.

The Choice of Subspace Our subspace construction strategy is devised based on the following three important observations/assumptions:

[1] α and β are small because the thickness (t dimension) of the shell is much smaller than its r and s dimensions.

♦ *The entire tube system is composed of multiple small-size components, many of which are of the same geometry because of the geometric symmetry.*
Accordingly, we compute the subspace basis vectors or *modes* at each of the tubular components individually so that the expensive computation associated with global finite element mesh is avoided. For tubular components of the same geometry but different locations/orientations, the modes can be directly synthesized by applying the corresponding rotation and translation.

♦ *Only a static equilibrium structural analysis is required in our case while vibrational response under highly accelerated velocity field (e.g. a launching rocket) is not our concern.*
Correspondingly, we safely ignore the dynamical structural analysis and only focus on the static equilibrium analysis with the format of $\mathbf{Ku} = \mathbf{f}$.

♦ *The supporting tubular structure often undertakes a much larger external regional loads comparing with its self-weight.*
Consequently, we build a complementary subspace at run-time to capture the residual deformation that is not included in the primary subspace. It can be mathematically proven that such two-level subspace simulation strategy is able to produce the same result as using the full-space[2].

6.1. Constraint Subspace – the Primary Subspace

Our primary subspace construction method is inspired by the *boundary mode* [45], which is essentially an extension of the classic CMS technique [43]. The basis vectors are computed per component by solving a static equilibrium system. For a given tubular component, we classify all of its DOFs into two categories namely, the *internal* DOF set and *boundary* DOF set, which are denoted using subscripts i and b respectively in the following formulation. We impose one unit displacement to each boundary DOF while restrain the rest boundary DOFs anchored, which leads to:

$$\begin{bmatrix} \mathbf{K}_{ii}^L & \mathbf{K}_{ib}^L \\ \mathbf{K}_{ib}^{L\top} & \mathbf{K}_{bb}^L \end{bmatrix} \begin{bmatrix} \Phi_i^L \\ \Phi_b^L \end{bmatrix} = \begin{bmatrix} \mathbf{F}_i^L \\ \mathbf{F}_b^L \end{bmatrix}, \tag{12}$$

where superscript L denotes the variables are local. Φ_b^L is an identity matrix corresponding to the unit boundary excitement imposed. $\mathbf{F}_i^L = \mathbf{0}$ as no external loads are applied at internal DOFs. The unknown internal response Φ_i^L can be easily computed by expanding the first line of Eq. 12:

$$\Phi_i^L = -\mathbf{K}_{ii}^{L^{-1}} \mathbf{K}_{ib}^L, \tag{13}$$

and mode vectors at the component are assembled by concatenating the Φ_i^L and Φ_b^L such that $\Phi^L = [\Phi_i^{L\top} | \Phi_b^{L\top}]^\top$. We notice that the formulation of Φ^L is consistent with the *constraint mode* in CMS [42]. Therefore, we refer the subspace spanned by Φ^L as the *constraint subspace*. Fig. 7 shows the shapes of five constraint modes associated with a boundary node while other boundary nodes (red nodes in the figure) are fixed.

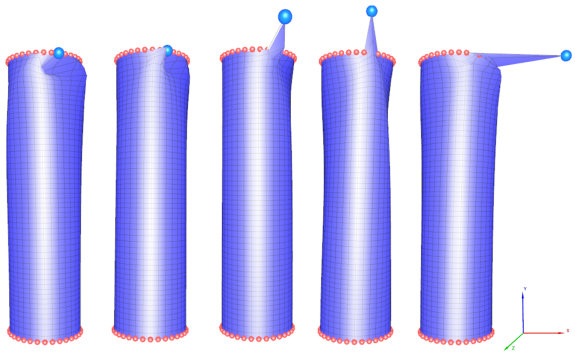

Figure 7. The shapes of five constraint modes associated with a boundary node (highlighted as blue node). Other restrained boundary DOFs are marked as red nodes.

With constraint modes, the system's displacement can be expressed using the reduced coordinates at each component such that:

$$\mathbf{u} = \bar{\Phi}\bar{\mathbf{q}}, \tag{14}$$

where $\bar{\Phi} = \mathrm{diag}(\Phi^1, \Phi^2, ...\Phi^k)$ and $\bar{\mathbf{q}} = [\mathbf{q}^{1\top}, \mathbf{q}^{2\top}, ...\mathbf{q}^{k\top},]$ are the *global* subspace matrix and generalized displacement vector of the system with k tubular components.

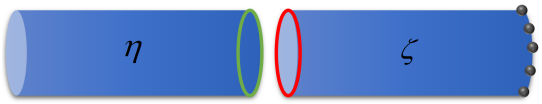

Figure 8. An illustrative example showing the coupling of tubular components η and ζ assuming that an appropriate boundary condition has been specified at the highlighted nodes.

6.2. Multiplier-Free Component Coupling

All the tubular components are mutually connected at their interfaces. Such coupling can be formulated as the *interface constraint* (IC) between a pair of adjacency components. As shown in Fig. 8, IC requires that duplicated boundary DOFs from components η and ζ must always have identical values e.g. $\mathbf{u}_b^\eta = \mathbf{u}_b^\zeta$. It can be re-written using the reduced coordinate in constraint subspace:

$$\mathbf{E}_b^\eta \Phi^\eta \mathbf{q}^\eta = \mathbf{E}_b^\zeta \Phi^\zeta \mathbf{q}^\zeta, \tag{15}$$

where $\mathbf{E}_b^\eta$ and $\mathbf{E}_b^\zeta$ are two elementary matrices extracting boundary DOFs from each component. A commonly-adopted approach to enforce Eq. 15 is to use the Lagrange Multiplier method, which explicitly formulates the interface forces as the unknown multipliers [41]. While our simulator can also be handled this way, there are two obvious drawbacks associated with multiplier-based solution: 1) the dimension of the resulting linear system is greatly increased due to the existence of the multipliers and

[2] In fact, even if the the structure's self-weight is not neglectable, we can further incorporate the *inertia-relief mode set* (extra six modes per component) to make the local primary subspace a *statically complete mode set*, which is able to accurately capture the static system response due to the rigid body acceleration as proven in [42].

the redundancy of the boundary DOFs and 2) the system matrix is no longer a symmetric positive definite (SPD) matrix as we have vanished diagonal elements at locations corresponding to the IC. Consequently, the effectiveness of subspace acceleration is compromised.

Alternatively, we enforce the interface constraint without replying on the Lagrange Multiplier method to maintain a more compact and better-conditioned subspace solver. Note that ICs are a set of linear constraints, which can be re-written as:

$$\mathbf{C}\bar{\mathbf{q}} = \mathbf{0}, \tag{16}$$

where $\bar{\mathbf{q}} = [\mathbf{q}^{\eta\top}, \mathbf{q}^{\zeta\top}]^\top$ and $\mathbf{C} = [\mathbf{E}_b^\eta \Phi^\eta | -\mathbf{E}_b^\zeta \Phi^\zeta]$ as in the case shown in Fig. 8. $\mathbf{C} \in \mathbb{R}^{c\times d}$ is a rectangular matrix, where c is the number of ICs of the system and d is the total number of the reduced coordinates at components η and ζ including the duplicated interface DOFs. Obviously, $d > c$, therefore $\mathbf{C}$ can be further split into two parts:

$$\mathbf{C} = [\mathbf{C}_1 | \mathbf{C}_2], \tag{17}$$

such that $\mathbf{C}_1 \in \mathbb{R}^{c\times c}$ is a full-rank square matrix. Eq. 16 can be re-written as:

$$\mathbf{C}_1\mathbf{q}_d + \mathbf{C}_2\mathbf{q}_f = \mathbf{0}, \tag{18}$$

where $\mathbf{q}_f$ represents a subset of $\bar{\mathbf{q}}$ consisting of only independent or free DOFs and $\mathbf{q}_d$ represents a subset of dependent DOFs. In the example shown in Fig. 8, if the DOFs on the red interface are free DOFs, the DOFs on the green interface are dependent ones and vice versa.

Since $\mathbf{C}_1$ is full-rank, we can use $\mathbf{q}_f$ to represent the other "redundant" DOFs $\mathbf{q}_d$:

$$\mathbf{q}_d = -\mathbf{C}_1^{-1}\mathbf{C}_2\mathbf{q}_f, \tag{19}$$

as well as the complete $\bar{\mathbf{q}}$ vector:

$$\bar{\mathbf{q}} = \begin{bmatrix} \mathbf{q}_f \\ \mathbf{q}_d \end{bmatrix} = \begin{bmatrix} \mathbf{I} \\ -\mathbf{C}_1^{-1}\mathbf{C}_2 \end{bmatrix} \mathbf{q}_f. \tag{20}$$

Substituting Eq. 20 into Eq. 14 yields

$$\mathbf{u} = \bar{\Phi} \begin{bmatrix} \mathbf{I} \\ -\mathbf{C}_1^{-1}\mathbf{C}_2 \end{bmatrix} \mathbf{q}_f = \Phi\mathbf{q}, \tag{21}$$

where $\Phi \triangleq \bar{\Phi} \begin{bmatrix} \mathbf{I} \\ -\mathbf{C}_1^{-1}\mathbf{C}_2 \end{bmatrix}$ and $\mathbf{q} \triangleq \mathbf{q}_f$. The full-space equilibrium $\mathbf{K}\mathbf{u} = \mathbf{f}$[3] can be directly projected onto the new subspace spanned by Φ where IC is implicitly encoded:

$$\mathbf{K}_q\mathbf{q} = \mathbf{f}_q. \tag{22}$$

Here, $\mathbf{K}_q = \Phi^\top\mathbf{K}\Phi$ and $\mathbf{f}_q = \Phi^\top\mathbf{f}$. This derivation can be easily extended for multiple components.

[3]Note that here $\mathbf{K}$ is a diagonal block sparse matrix, each diagonal block is the component's local stiffness matrix.

6.3. Residual Subspace – the Secondary Subspace

Using constraint modes is able to significantly improve the performance, yet it also sacrifices the accuracy of the simulation. Tubular supporting structure is often exerted concentrated regional loads of large magnitude. While the deflections at load-free components are accurately captured (when the gravity effect can be ignored) because all the inter-component stress propagations are losslessly passed via the interface whose DOFs are fully preserved within constraint subspace, deflection at the *loading components* where the forces are applied is not able to be well represented with constraint modes. To resolve this issue, a local secondary subspace is built to capture the residual deflection and improve the simulation accuracy at loading components, which is detailed in this subsection. As all the formulation is for a certain loading component, the superscript L is omitted.

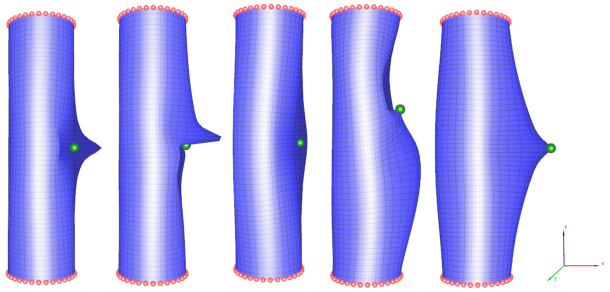

Figure 9. The shapes of five residual modes associated with an exciting node (green).

We denote all the internal DOFs undertaking the external loads as the *exciting* DOFs while all the other internal DOFs as *passive* DOFs. They are symbolized using subscripts e and p respectively. Similar to constraint mode, a unit displacement is imposed to the each of the exciting DOF while keep other exciting DOFs and boundary DOFs fixed. We restrain the boundary DOFs so that the complementary deflection computed will not affect the status of other load-free components. A equilibrium system can be listed accordingly:

$$\begin{bmatrix} \mathbf{K}_{ee} & \mathbf{K}_{ep} & \mathbf{K}_{eb} \\ \mathbf{K}_{ep}{}^\top & \mathbf{K}_{pp} & \mathbf{K}_{pb} \\ \mathbf{K}_{eb}{}^\top & \mathbf{K}_{pb}{}^\top & \mathbf{K}_{bb} \end{bmatrix} \begin{bmatrix} \Psi_e \\ \Psi_p \\ \Psi_b \end{bmatrix} = \begin{bmatrix} \mathbf{F}_e \\ \mathbf{F}_p \\ \mathbf{F}_b \end{bmatrix}, \tag{23}$$

where $\Psi_e = \mathbf{I}$ and $\Psi_b = \mathbf{0}$ correspond to the imposed unit displacement at exciting DOFs and anchored boundary DOFs. $\mathbf{F}_p = \mathbf{0}$ as no forces are applied at the passive DOFs. Again, the superscript L indicating the local variables is omitted here. The unknown system response at passive DOFs can be computed by expanding the second line of Eq. 23:

$$\Psi_p = -\mathbf{K}_{pp}^{-1}\mathbf{K}_{ep}^\top. \tag{24}$$

We use $span(\Psi)$ to represent the subspace spanned by Ψ, and name it as the *residual subspace* whose basis vectors are *residual modes*. Fig. 9 shows the shapes of five residual modes associated with an internal exciting node.

The residual modes are employed based on the fact that the response of a linear system of a composite input is equivalent to the superposition of the system's responses with respect to each individual input. In fact, it can be proven that the superset of Φ and Ψ is able to completely capture the deflection at loading components. We refer readers to Appendix A for mathematical proof details.

In other words, it means that an accurate result will be obtained if the system is solved within $span(\Phi) \cup span(\Psi)$:

$$\begin{bmatrix} \Phi^\top \\ \Psi^\top \end{bmatrix} \mathbf{K}[\Phi|\Psi] \begin{bmatrix} \mathbf{q} \\ \mathbf{p} \end{bmatrix} = \begin{bmatrix} \Phi^\top \mathbf{f} \\ \Psi^\top \mathbf{f} \end{bmatrix},$$

or

$$\begin{bmatrix} \mathbf{K}_{\Phi\Phi} & \mathbf{K}_{\Phi\Psi} \\ \mathbf{K}_{\Psi\Phi} & \mathbf{K}_{\Psi\Psi} \end{bmatrix} \begin{bmatrix} \mathbf{q} \\ \mathbf{p} \end{bmatrix} = \begin{bmatrix} \mathbf{f}_\Phi \\ \mathbf{f}_\Psi \end{bmatrix}, \tag{25}$$

where

$$\begin{cases} \mathbf{K}_{\Phi\Phi} &= \Phi^\top \mathbf{K} \Phi \\ \mathbf{K}_{\Phi\Psi} &= \Phi^\top \mathbf{K} \Psi \\ \mathbf{K}_{\Psi\Phi} &= \Psi^\top \mathbf{K} \Phi \\ \mathbf{K}_{\Psi\Psi} &= \Psi^\top \mathbf{K} \Psi \end{cases}. \tag{26}$$

Here, $\mathbf{q}$ and $\mathbf{p}$ are the reduced coordinates of constraint modes and residual modes. While Φ can be pre-computed for each component, Ψ is a load-dependent matrix as different external loads would specify different exciting DOF sets and therefore, yield different mode matrices. It implies that as soon as the external loads are changed, the entire system must be recomputed, which significantly downgrades the useability of the system.

We notice that if the off-diagonal blocks (e.g. $\mathbf{K}_{\Phi\Psi}$ and $\mathbf{K}_{\Phi\Psi}^\top$) in Eq. 25 are zero, the constraint subspace and residual subspace will be decoupled and the computation of $\mathbf{q}$ and $\mathbf{p}$ are isolated. Therefore, we apply the *modified Gram-Schmidt process* (MGS) [46] towards Ψ with respect to $\mathbf{K}\Phi$, which yields a new set of residual modes $\tilde{\Psi}$ such that $\tilde{\Psi} \perp \mathbf{K}\Phi$ or $(\mathbf{K}\Phi)^\top \tilde{\Psi} = \mathbf{0}$. Boundary DOFs are always fixed during the computation of Ψ. On the contrary, there always exists one non-zero boundary DOF in the constraint mode. Such properties of constraint modes and residual modes guarantee that $span(\Phi) \cap span(\Psi) = \varnothing$. Therefore, $span(\Psi) = span(\tilde{\Psi})$ and $span(\Phi) \cup span(\Psi) = span(\Phi) \cup span(\tilde{\Psi})$. Substituting Ψ with $\tilde{\Psi}$, Eq. 25 is simplified to:

$$\begin{cases} \mathbf{K}_{\Phi\Phi}\mathbf{q} &= \mathbf{f}_\Phi \\ \mathbf{K}_{\tilde{\Psi}\tilde{\Psi}}\mathbf{p} &= \mathbf{f}_{\tilde{\Psi}}. \end{cases} \tag{27}$$

The final component deflection is computed using

$$\begin{aligned} \mathbf{u} &= [\Phi|\tilde{\Psi}][\mathbf{q}^\top|\mathbf{p}^\top]^\top \\ &= \Phi\mathbf{q} + \Psi\mathbf{p} \\ &\triangleq \mathbf{u}_\Phi + \mathbf{u}_{\tilde{\Psi}}. \end{aligned} \tag{28}$$

Note that $\mathbf{u}_\Phi$ is the constraint subspace displacement. Therefore, we only need to calculate an incremental displacement $\mathbf{u}_{\tilde{\Psi}}$ in order to obtain the exact full-space result at loading components. Fig. 10 summarizes the major computational procedures in our system.

7. Experimental Results

We report and discuss experiments we have conducted in this section. Please refer to the accompanying video for more results including the test use of the fabricated tubular models.

7.1. Hardware & Software Platform

Our experiments are carried out on a Dell Optiplex 9010 workstation computer equipped with an Intel i7-3770, 3.40 Ghz CPU and 16G memory. The proposed system is implemented on 64-bit Microsoft Windows 7 using Visual Studio 2010. We use Eigen numerical library [47] for most linear system related calculations. Note that we only use the single-core implementation however, many computations (e.g. per-component subspace construction) can be trivially parallelized using multi-threading.

7.2. Four-node Element vs. Nine-node Element

Existing FEM literatures [3, 44] have mentioned that linear four-node element is not a good choice for general shell simulation. It is partially because the governing stress equilibrium is characterized using a second-order partial differential equation. The adoption of weak form (well-known as virtual work principle in the context of continuum mechanics) allows the usage of linear interpolation functions (e.g. linear element) however, the accuracy of the simulation is compromised. The adoption of the linear element also leads to the *shear locking* artifact, which is shown in Fig. 11. The regular cylinder-shaped tube is simulated using the same number of four-node shell elements (left) and nine-node shell elements (right), respectively. The external forces are applied at the highlighted nodes in the positive y direction. The bending deformation can be well observed with nine-node element which is however, "locked" with four-node element.

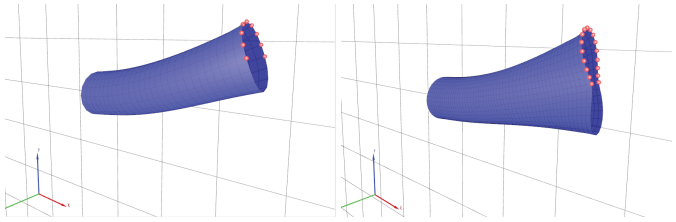

Figure 11. Shear locking using linear four-node element (left) which eliminates the bending deflection while quadratic nine-node element (right) does not have such artifact.

7.3. User Interface & Implementation Details

Fig. 12 shows a screen capture of the user interface of the proposed design-simulation system. Right to the main 3D view, our system provides an intuitive interface for the user to specify the boundary (top) and skeletal (bottom) controls of tube components. Our system maintains a tube library shown blew the main 3D view. The geometry of each tube component can be freely edited. Immediately after the geometric edit of a component is committed, the updated component will be inserted into

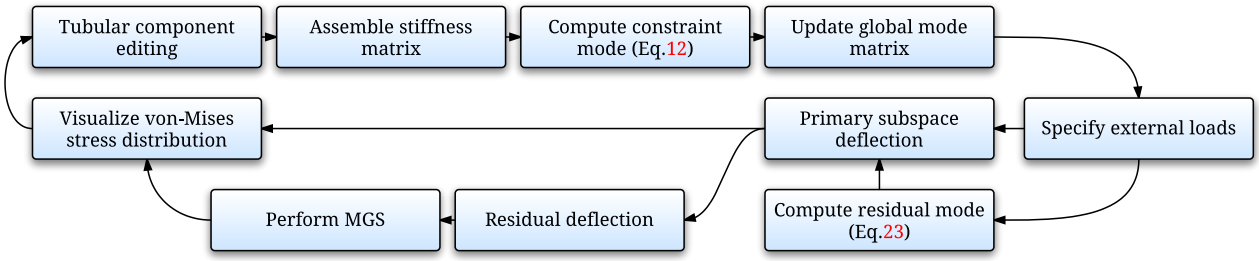

Figure 10. Overall computational procedures of the two-level subspace simulation method.

the library as a new component. All the related pre-computation is also carried out at this stage. On average, a tube component holds about 4,000 to 7,000 DOFs. The related computations like assembling the stiffness matrix $\mathbf{K}$, computing the constraint modes Φ as well as calculating the matrix-vector product of $\mathbf{K}\Phi$, which is for the potential MGS to be applied if this component is a loading component, can be done in real-time. During the model assemblage, each component can be freely copied and pasted. User is also able to specify the geometric symmetry during the editing so that the shape edits applied at a component will be automatically mapped to its geometrically symmetric counterparts like the stool legs shown in Fig. 2.

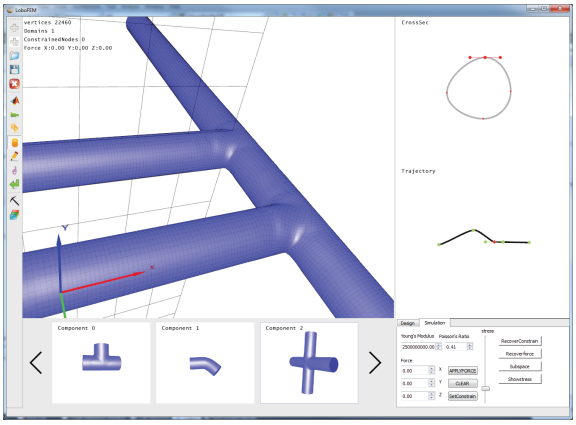

Figure 12. The user interface of the proposed system.

As soon as the entire structure is assembled, we need to build the global Φ matrix (Eq. 21) as the primary subspace basis vectors. Therefore matrices $\mathbf{C}_1$ and $\mathbf{C}_2$ (Eq. 17) must be identified. They can be efficiently found as each column in the original $\mathbf{C}$ matrix corresponds to a system DOF while all the row vectors in $\mathbf{C}$ are guaranteed to be linearly independent (as long as the IC are not redundantly defined in $\mathbf{C}$). Therefore, we only need to construct another elementary matrix encoding the necessary column permutation to move all the columns corresponding to independent DOFs to the left-end of the matrix. As long as the topology of the tubular structure is not altered, this elementary matrix remains the same.

After the displacement is computed. The strain vector $\varepsilon \in \mathbb{R}^{6\times 1}$ can be easily evaluated using the strain-displacement matrix $\mathbf{B}$, which is further converted to the stress vector σ according to the assumed linear elasticity: $\sigma = \mathbf{C}\varepsilon$. Finally, we visualize the von Mises stress using the GLSL shader. The von Mises stress is a scalar and can be computed as:

$$\sigma_{von}^2 = \frac{\sigma_{1,2}^2 + \sigma_{2,3}^2 + \sigma_{3,1}^2 + 6(\sigma_{12}^2 + \sigma_{23}^2 + \sigma_{31}^2)}{2}, \qquad (29)$$

where $\sigma_{i,j} = \sigma_{ii} - \sigma_{jj}$ and σ_{ij} is the i,j component in the tensor representation of σ. We simulate the tubular structure of stereolithography (SLA) material with Young Modulus of $2.5e9$ and Poisson Ration of 0.41. Regions with high von Mises stress are likely to fail under the prescirbed external loads as shown in Fig. 14.

Fig. 13 shows the snapshots of using our system for designing and simulating various tubular structures. Since authors are not professional designers, we just follow some design ideas searched from internet shown in the leftmost column of the figure. When high-stress regions is observed, we apply some further geometric edits to the model including altering the shapes of the cross-sections at critical region (row 1, the lamp stand model), reducing the curvature connecting neighbor components (row 2, the laptop holder model), adding extra supporting components (row 3, the bookshelf model) and reducing the force moment (row 4, the camera rack model). The edited regions are highlighted in the rightmost column in the figure.

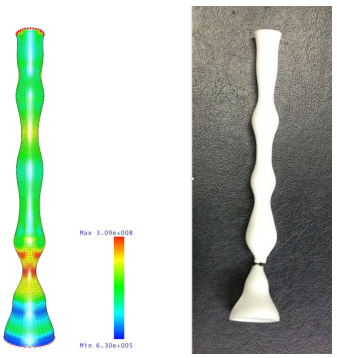

Figure 14. The 3D printed lamp stand fails and the failure location matches the area where high stress distribution is observed.

7.4. Time Performance

Tab. 1 reports the detailed statistic of the 3D models we have tested. We compare the time performance of the proposed two-level subspace simulation method with the full-space simulation as well as the subspace simulator using the Lagrange

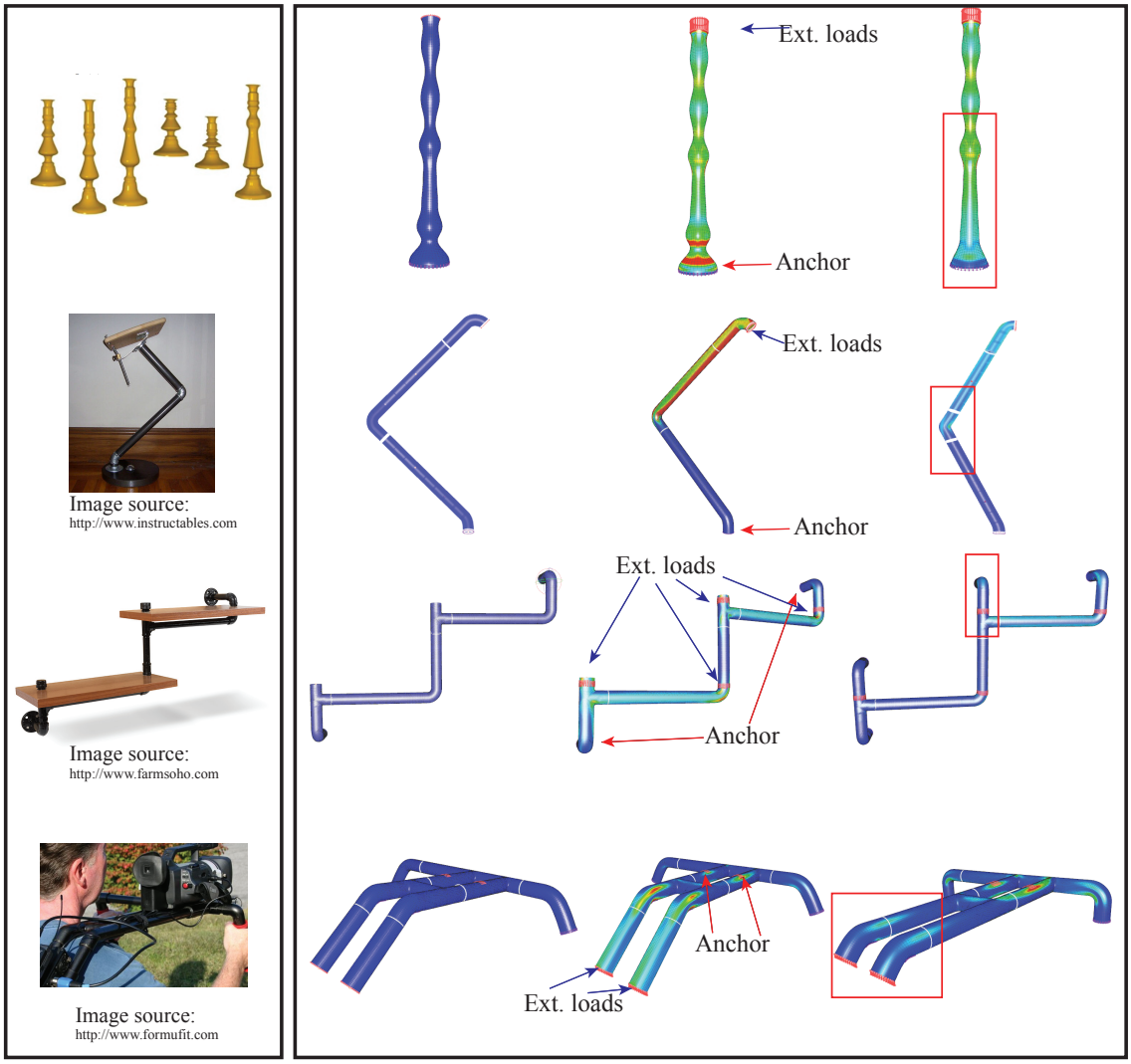

Figure 13. Snapshots of using the proposed design-simulation system.

multiplier method. While the full-space system can be solved using the sparse Cholesky solver (the built-in `SimplicialLLT` routine in Eigen library), the simulation is not interactive along with the design operations with lags of seconds. On the other hand, the multiplier based subspace solver often has doubled or tripled size comparing with our method, due to the dulcification of the boundary DOFs as well as the explicit formulation of unknown multipliers. In addition, the resulting system matrix is no longer SPD either and can not be handled with LLT decomposition. Therefore, the performance data listed in Tab. 1 is the one using the LU decomposition (the built-in `PartialPivLU` solver in Eigen library), which is slower than LLT decomposition in most cases. As a result, even with much fewer DOFs, Lagrange multiplier based subspace solver could be even slower than the full-space solver.

The proposed multiplier-free coupling mechanism will have a dense SPD matrix of much smaller size. Therefore, it is much more efficient than the above-mentioned two solvers. For 3D models with over 100k full-space DOFs, our method is still able to perform the accurate structural analysis at an interactive rate. We would like to remind the reader that unlike most existing subspace model reduction methods, *our method does not compromise simulation accuracy* while making the simulation an order of magnitude faster. Extra computations are required for applying the MGS, constructing the residual subspace and solving the secondary deflection. However, such computations are light-weight as they are conducted at the component level. In most cases, they can be finished within milliseconds.

8. Conclusion and Limitation

In this paper, we present a system for design and simulation of tubular supporting structure. Besides an intuitive shape control mechanism, our system is able to interactively perform the structural analysis using a two-level subspace FEM simulation. We show the interactive and accurate design and simulation of large 3D model of over 100k DOFs. However, there still exist several limitations in the current version of the system. First

Model	#Ele.	#Nodes	#DOFs	S(L)DOFs	SDOFs	FT (s)	S(L)T (s)	ST (s)	#Com.
Lamp stand	5,184	5,216	25,920	1,120	480	0.406	0.093	0.01	3
Stool legs	6,456	25,946	129,090	5,160	2,040	2.108	7.278	0.633	14
Bookshelf	25,824	19,547	97,750	5,120	2080	1.532	6.96	0.66	11
Bookshelf edited	25,280	22,942	114,710	5,760	1,920	1.861	9.484	0.51	13
Camera rack	22,400	22,170	110,850	4,920	2040	2.060	6.301	0.62	13
Laptop holder	12,096	12,128	60,480	3,040	1,120	0.784	1.472	0.103	7

Table 1. Time performance of our method, full-space simulator as well as subspace simulator using Lagrange Multiplier method. **#Ele.**: the number of elements; **#Nodes**: the number of free nodes; **#DOFs**: the number of full-space DOFs: **#S(L)DOFs** the size of the simulator using constraint mode and Lagrange Multiplier method; **SDOFs**: the size of the simulator using the propose multiplier-free coupling method; **FT**: time used to solve the system in full-space; **S(L)T**: time used to solve the multiplier-based subspace system; **ST**: time for our method; **#Com**: the number of the tube components.

of all, our design system is only able to handle simple tubular components with two open interfaces. More complex tubes of T-shape or Y-shape are not able to be intuitively edited. A possible solution is to deform a template with boundary control by solving higher-order harmonic equations. As a future work, we will study how to use fundamental solutions to accelerate the geometric design of multi-interface tube components. During the shape editing, self-intersection could occur at highly curved regions. However, our current system cannot automatically fix such problem and it only sends out an alert to the user. Feature modeling is not supported in our system. Different domain decomposition strategies are required to accelerate the simulation if features are added to tubular structures, which gives another direction of our future work. In the simulation part, we ignore the deflection induced by component's self-weight, which could also induce simulation inaccuracy when the tubular component is fabricated using material of high density. We plan to incorporate the *inertia-relief modes* [42] to fully accommodate the gravity effect to the system in the future. Another interest future direction is to make simulation active meaning the simulator will provide the user potential solutions to fix a "faulty" geometric edit as in many recent design-simulation systems [26].

9. Acknowledgement

The authors would like to thank anonymous reviewers for their constructive comments. Ran Luo and Yin Yang were partially supported by National Science Foundation (NSF) under award No. IIS-1464306. Yin Yang was also partially supported by UNM RAC award and OVPR award. Weiwei Xu is partially supported by National Science Foundation of China (NSFC) under grants No.61272392 and No.61322204.

References

[1] L. H. You, X. S. Yang, M. Pachulski, J. J. Zhang, Boundary constrained swept surfaces for modelling and animation, Computer Graphics Forum 26 (3) (2007) 313–322.
[2] A. Jacobson, E. Tosun, O. Sorkine, D. Zorin, Mixed Finite Elements for Variational Surface Modeling, Computer Graphics Forum 29 (5) (2010) 1565–1574.
[3] K. Bathe, Finite Element Procedures, Klaus-Jurgen Bathe, 2007.
[4] U. Shani, D. H. Ballard, Splines as embeddings for generalized cylinders, Computer Vision, Graphics, and Image Processing 27 (2) (1984) 129 – 156.
[5] F. Klok, Two moving coordinate frames for sweeping along a 3d trajectory, Computer Aided Geometric Design 3 (3) (1986) 217 – 229.
[6] W. Wang, B. Jüttler, D. Zheng, Y. Liu, Computation of rotation minimizing frames, ACM Trans. Graph. 27 (1) (2008) 2:1–2:18.
[7] T.-I. Chang, J.-H. Lee, M.-S. Kim, S. J. Hong, Direct manipulation of generalized cylinders based on b-spline motion, The Visual Computer 14 (5-6) (1998) 228–239.
[8] J. K. Johnstone, J. P. Williams, A rational model of the surface swept by a curve., Comput. Graph. Forum 14 (3) (1995) 77–88.
[9] S. Coquillart, A control-point-based sweeping technique, Computer Graphics and Applications, IEEE 7 (11) (1987) 36–45.
[10] G. Wang, J. Sun, Shape control of swept surface with profiles, Computer-Aided Design 33 (12) (2001) 893 – 902.
[11] L. You, J. Chang, X. Yang, J. J. Zhang, Solid modelling based on sixth order partial differential equations, Computer-Aided Design 43 (6) (2011) 720 – 729.
[12] M. Botsch, L. Kobbelt, An intuitive framework for real-time freeform modeling, ACM Trans. Graph. 23 (3) (2004) 630–634.
[13] H. Qin, D. Terzopoulos, D-nurbs: A physics-based framework for geometric design, IEEE Transactions on Visualization and Computer Graphics 2 (1) (1996) 85–96.
[14] D. Terzopoulos, H. Qin, Dynamic nurbs with geometric constraints for interactive sculpting, ACM Trans. Graph. 13 (2) (1994) 103–136.
[15] K. Wang, Y. He, X. Guo, H. Qin, Spline thin-shell simulation of manifold surfaces, in: Proceedings of the 24th International Conference on Advances in Computer Graphics, CGI'06, Springer-Verlag, Berlin, Heidelberg, 2006, pp. 570–577.
[16] D. Baraff, A. Witkin, Large steps in cloth simulation, in: Proceedings of the 25th Annual Conference on Computer Graphics and Interactive Techniques, SIGGRAPH '98, ACM, New York, NY, USA, 1998, pp. 43–54.
[17] A. Garg, E. Grinspun, M. Wardetzky, D. Zorin, Cubic shells, in: Proceedings of the 2007 ACM SIGGRAPH/Eurographics Symposium on Computer Animation, SCA '07, Eurographics Association, Aire-la-Ville, Switzerland, Switzerland, 2007, pp. 91–98.
[18] E. Grinspun, A. N. Hirani, M. Desbrun, P. Schröder, Discrete shells, in: Proceedings of the 2003 ACM SIGGRAPH/Eurographics Symposium on Computer Animation, SCA '03, Eurographics Association, Aire-la-Ville, Switzerland, Switzerland, 2003, pp. 62–67.
[19] M. Wardetzky, M. Bergou, D. Harmon, D. Zorin, E. Grinspun, Discrete quadratic curvature energies, Comput. Aided Geom. Des. 24 (8-9) (2007) 499–518.
[20] X. Guo, X. Li, Y. Bao, X. Gu, H. Qin, Meshless thin-shell simulation based on global conformal parameterization, IEEE Transactions on Visualization and Computer Graphics 12 (3) (2006) 375–385.
[21] S. Martin, P. Kaufmann, M. Botsch, E. Grinspun, M. Gross, Unified simulation of elastic rods, shells, and solids, Vol. 29, ACM, 2010.
[22] M. Wicke, D. Steinemann, M. Gross, Efficient animation of point-sampled thin shells, in: Computer Graphics Forum, Vol. 24, Wiley Online Library, 2005, pp. 667–676.
[23] W. M. Lai, D. Rubin, E. Krempl, Introduction to Continuum Mechanics, Fourth Edition, 4th Edition, Elsevier, Amsterdam ; Boston, 2009.
[24] N. Zhang, H. Qu, R. Sweet, Orientation-preserving rod elements for real-time thin-shell simulation, Visualization and Computer Graphics, IEEE Transactions on 17 (6) (2011) 822–835.

[25] O. Stava, J. Vanek, B. Benes, N. Carr, R. Měch, Stress relief: Improving structural strength of 3d printable objects, ACM Trans. Graph. 31 (4) (2012) 48:1–48:11.
[26] N. Umetani, T. Igarashi, N. J. Mitra, Guided exploration of physically valid shapes for furniture design, ACM Trans. Graph. 31 (4) (2012) 86:1–86:11.
[27] Q. Zhou, J. Panetta, D. Zorin, Worst-case structural analysis, ACM Trans. Graph. 32 (4) (2013) 137:1–137:12.
[28] S. Coros, B. Thomaszewski, G. Noris, S. Sueda, M. Forberg, R. W. Sumner, W. Matusik, B. Bickel, Computational design of mechanical characters, ACM Trans. Graph. 32 (4) (2013) 83:1–83:12.
[29] L. Zhu, W. Xu, J. Snyder, Y. Liu, G. Wang, B. Guo, Motion-guided mechanical toy modeling, ACM Trans. Graph. 31 (6) (2012) 127:1–127:10.
[30] M. Bächer, B. Bickel, D. L. James, H. Pfister, Fabricating articulated characters from skinned meshes, ACM Trans. Graph. 31 (4) (2012) 47:1–47:9.
[31] J. Calì, D. A. Calian, C. Amati, R. Kleinberger, A. Steed, J. Kautz, T. Weyrich, 3d-printing of non-assembly, articulated models, ACM Trans. Graph. 31 (6) (2012) 130:1–130:8.
[32] X. Chen, C. Zheng, W. Xu, K. Zhou, An asymptotic numerical method for inverse elastic shape design, ACM Trans. Graph. 33 (4) (2014) 95:1–95:11.
[33] N. Umetani, D. M. Kaufman, T. Igarashi, E. Grinspun, Sensitive couture for interactive garment modeling and editing, ACM Trans. Graph. 30 (4) (2011) 90:1–90:12.
[34] F. Cirak, M. J. Scott, E. K. Antonsson, M. Ortiz, P. Schröder, Integrated modeling, finite-element analysis, and engineering design for thin-shell structures using subdivision, Computer-Aided Design 34 (2) (2002) 137–148.
[35] S. Green, G. Turkiyyah, D. Storti, Subdivision-based multilevel methods for large scale engineering simulation of thin shells, in: Proceedings of the Seventh ACM Symposium on Solid Modeling and Applications, SMA '02, ACM, New York, NY, USA, 2002, pp. 265–272.
[36] S. S. An, T. Kim, D. L. James, Optimizing cubature for efficient integration of subspace deformations, in: ACM SIGGRAPH Asia 2008 Papers, SIGGRAPH Asia '08, ACM, New York, NY, USA, 2008, pp. 165:1–165:10.
[37] J. Barbič, D. L. James, Real-time subspace integration for st. venant-kirchhoff deformable models, in: ACM SIGGRAPH 2005 Papers, SIGGRAPH '05, ACM, New York, NY, USA, 2005, pp. 982–990.
[38] M. G. Choi, H.-S. Ko, Modal warping: Real-time simulation of large rotational deformation and manipulation, IEEE Transactions on Visualization and Computer Graphics 11 (1) (2005) 91–101.
[39] M. G. Choi, S. Yong Woo, H.-S. Ko, Real-time simulation of thin shells, Computer Graphics Forum 26 (3) (2007) 349–354.
[40] R. Kornhuber, R. W. Hoppe, J. Periaux, O. Pironneau, O. Widlund, J. Xu (Eds.), Domain Decomposition Methods in Science and Engineering, 2005th Edition, Springer, Berlin, 2004.
[41] C. Farhat, F.-X. Roux, A method of finite element tearing and interconnecting and its parallel solution algorithm, International Journal for Numerical Methods in Engineering 32 (6) (1991) 1205–1227.
[42] R. R. Craig, A review of time-domain and frequency-domain component mode synthesis method, 1985, 00298.
[43] W. C. HURTY, Dynamic analysis of structural systems using component modes, AIAA Journal 3 (4) (1965) 678–685, 00961.
[44] O. C. Zienkiewicz, R. L. Taylor, The Finite Element Method for Solid and Structural Mechanics, Sixth Edition, 6th Edition, Butterworth-Heinemann, Amsterdam ; Boston, 2005.
[45] Y. Yang, W. Xu, X. Guo, K. Zhou, B. Guo, Boundary-aware multidomain subspace deformation, Visualization and Computer Graphics, IEEE Transactions on 19 (10) (2013) 1633–1645.
[46] L. N. Trefethen, D. B. III, Numerical Linear Algebra, SIAM: Society for Industrial and Applied Mathematics, Philadelphia, 1997.
[47] G. Guennebaud, B. Jacob, et al., Eigen v3, http://eigen.tuxfamily.org (2010).

Appendix A.

Proof. We show that the subspace component deflection computed using Eq. 25 is identical to the solution of the full-space equilibrium (e.g. $\mathbf{Ku} = \mathbf{f}$). Noticing that for a loading component, external forces are only applied to the exciting DOFs and the corresponding component equilibrium becomes:

$$\begin{bmatrix} \mathbf{K}_{ee} & \mathbf{K}_{ep} & \mathbf{K}_{eb} \\ \mathbf{K}_{ep}{}^{\top} & \mathbf{K}_{pp} & \mathbf{K}_{pb} \\ \mathbf{K}_{eb}{}^{\top} & \mathbf{K}_{pb}{}^{\top} & \mathbf{K}_{bb} \end{bmatrix} \begin{bmatrix} \mathbf{u}_e \\ \mathbf{u}_p \\ \mathbf{u}_b \end{bmatrix} = \begin{bmatrix} \mathbf{f}_e \\ \mathbf{0}_p \\ \mathbf{f}_b \end{bmatrix}. \tag{A.1}$$

Expanding the second line of Eq. A.1 yields:

$$\mathbf{u}_p = -\mathbf{K}_{pp}^{-1}\mathbf{K}_{ep}^{\top}\mathbf{u}_e - \mathbf{K}_{pp}^{-1}\mathbf{K}_{pb}\mathbf{u}_b. \tag{A.2}$$

Similarly, we also expand the equilibrium of the passive DOFs through the definition of constraint mode and residual mode (e.g. Eqs. 12 and 23), which gives:

$$\mathbf{K}_{ep}^{\top}\Phi_e + \mathbf{K}_{pp}\Phi_p + \mathbf{K}_{pb}\mathbf{I}_b = \mathbf{0}, \tag{A.3}$$

and

$$\mathbf{K}_{pp}\Psi_p + \mathbf{K}_{ep}^{\top} = \mathbf{0}. \tag{A.4}$$

By substituting Eqs. A.3 and A.4 into Eq. A.2, we obtain:

$$\mathbf{u}_p = \Psi_p\mathbf{u}_e + (\Phi_p - \Psi_p\Phi_e)\,\mathbf{u}_b,$$

which leads to:

$$\begin{bmatrix} \mathbf{u}_e \\ \mathbf{u}_p \\ \mathbf{u}_b \end{bmatrix} = \begin{bmatrix} \Phi_e & \Psi_e \\ \Phi_p & \Psi_p \\ \mathbf{I}_b & \Psi_b \end{bmatrix} \begin{bmatrix} \mathbf{u}_b \\ \mathbf{u}_e - \Phi_e\mathbf{u}_b \end{bmatrix} = [\Phi|\Psi] \begin{bmatrix} \mathbf{q} \\ \mathbf{p} \end{bmatrix}.$$

□