Real-Time Nonlinear Shape Interpolation

CHRISTOPH VON TYCOWICZ
Zuse Institute Berlin
and
CHRISTIAN SCHULZ
Max Planck Institute for Informatics
and
HANS-PETER SEIDEL
Max Planck Institute for Informatics
and
KLAUS HILDEBRANDT
Delft University of Technology
Max Planck Institute for Informatics

We introduce a scheme for real-time nonlinear interpolation of a set of shapes. The scheme exploits the structure of the shape interpolation problem, in particular, the fact that the set of all possible interpolated shapes is a low-dimensional object in a high-dimensional shape space. The interpolated shapes are defined as the minimizers of a nonlinear objective functional on the shape space. Our approach is to construct a reduced optimization problem that approximates its unreduced counterpart and can be solved in milliseconds. To achieve this, we restrict the optimization to a low-dimensional subspace that is specifically designed for the shape interpolation problem. The construction of the subspace is based on two components: a formula for the calculation of derivatives of the interpolated shapes and a Krylov-type sequence that combines the derivatives and the Hessian of the objective functional. To make the computational cost for solving the reduced optimization problem independent of the resolution of the example shapes, we combine the dimensional reduction with schemes for the efficient approximation of the reduced nonlinear objective functional and its gradient. In our experiments, we obtain rates of 20-100 interpolated shapes per second even for the largest examples which have 500k vertices per example shape.

Categories and Subject Descriptors: I.3.5 [**Computer Graphics**]: Computational Geometry and Object Modeling—*Physically based modeling*

General Terms: XXX, YYY

Additional Key Words and Phrases: shape interpolation, shape averaging, geometric modeling, geometry processing, interactive tools, geometric optimization, model reduction.

ACM Reference Format:

von Tycowicz, C., Schulz, C., Seidel, H.-P., and Hildebrandt, K. 2014. Real-time Nonlinear Shape Interpolation. ACM Trans. Graph. VV, N, Article XXX (Month YYYY), X pages.
DOI = 10.1145/XXXXXXX.YYYYYYY
http://doi.acm.org/10.1145/XXXXXXX.YYYYYYY

1. INTRODUCTION

Efficient algorithms for the interpolation (or averaging) of shapes are important for various tasks in graphics, including morphing, pose and animation transfer, controlling deformable object simulations using example-based materials, example-based shape editing, and shape exaggeration. These applications pose high demands on a shape interpolation scheme. On one hand, "good" interpolated shapes that match our intuition of how shapes deform are desired. On the other hand, the computation must be fast since many weighted average shapes have to be computed or even real-time rates are required for interactive applications. Recently, nonlinear geometric and physically-based approaches that produce very pleasing interpolated shapes have been proposed. However, the nonlinear modeling comes at high computational costs.

We propose an approach for real-time nonlinear shape interpolation that exploits the following structural property of the shape interpolation problem. An accurate representation of a detailed shape requires a large number of triangles (or tets). Hence, the space of all possible variations of the shape is of high dimension. In contrast, the number m of example poses to be interpolated is typically very small. Since an interpolated shape is specified by $m-1$ weights, the set of all possible interpolated shapes of m example poses is an $(m-1)$-parameter family. Consequently, the optimization problem we use to model the shape interpolation is high-dimensional—but has only $m-1$ input parameters.

Our approach is based on an offline-online framework. In the offline phase, we construct a reduced optimization problem that approximates the high-dimensional problem and can be efficiently solved. In particular, the computational cost for solving the reduced problem is independent of the resolution of the example poses; it depends on the geometry of the example poses and the desired accuracy of the approximation. We set up the reduced problem in two steps. First, the space of possible shape variations is restricted to a low-dimensional affine subspace. To do this, we propose an efficient construction of reduced spaces for nonlinear shape interpolation based on differential properties of the objective functional. By restricting the shape interpolation problem to this subspace, we obtain a low-dimensional optimization problem. However, the cost for evaluating the objective functional and gradient is still too high for real-time performance since it depends on the resolution of the example poses. To design schemes for the fast evaluation or approximation of the reduced objective functional and its gradient, one can use the fact that the functional and gradient are only evaluated at points in the subspace and that only the projections of the gradients to the subspace are needed. In the second step, we set up the approximation of the reduced functional using quartic polynomials [Barbič and James 2005], optimized cubature [An et al.

2008], and mesh coarsening [Hildebrandt et al. 2011]. Our examples demonstrate that fairly low-dimensional reduced spaces (less than 70-dimensional in all our examples) suffice to produce interpolated shapes that well-approximate the unreduced solution. Since the computational cost for solving the reduced optimization problem mainly depends on the size of the reduced space, we obtain real-time performance even for meshes that represent detailed surfaces. In our experiments, we obtain rates of 20-100 interpolated shapes per second in all experiments—even for interpolating between meshes with 500k vertices.

We implemented the reduction technique for a broad framework for nonlinear shape interpolation based on the approach for elastic shape averaging introduced by Rumpf and Wirth [2009]. In this approach, every shape is assigned material properties, which are represented by a deformation energy. In the simplest case, all shapes are given the same homogeneous and isotropic material. For an arbitrary shape, one can compute the energy stored in the object when it is deformed into this shape. To interpolate, we assign positive weights to each of the example shapes (such that the weights sum to one) and consider the weighted sum of the stored energies. The weighted average shape is the minimizer of this weighted sum over all admissible shapes. In [Rumpf and Wirth 2009], elastic shape averaging is formulated for implicitly described shapes and the optimization is performed over all possible correspondences between the shapes. We adapt the approach to our setting in which the shapes are explicitly described (*e.g.* triangle or tet meshes) and correspondences between the shapes are given. This setting, where correspondences between the shapes are given, is typically considered in graphics. For example, correspondences are available when the example shapes are poses of one character or object. Furthermore, it is important to preserve the correspondences (*e.g.*, if the example shapes have textures). For particular choices of deformation energies, we obtain optimization problems comparable to those treated in [Fröhlich and Botsch 2011] and [Martin et al. 2011] and for interpolating between two shapes, we obtain the interpolation scheme considered in [Chao et al. 2010].

2. RELATED WORK

Many shape interpolation schemes are based on the following procedure. Select a number of geometric quantities of a shape that determine the shape (often up to rigid motion). Then, for all example shapes to be interpolated, compute and average the quantities. Finally, reconstruct the shape that best matches the averaged quantities. The differences between the methods lie in the choice of the geometric quantities and in the way the quantities are averaged. The reconstruction is typically done in a least-squares sense. Depending on whether the quantities depend linearly or nonlinearly on the vertex positions, the reconstruction is a linear or nonlinear least-squares problem. Examples of the first kind are the schemes of Sumner and Popović [2004], Sumner et al. [2005], and Xu et al. [2005]. The geometric quantities used are the deformation gradients of the triangles (or tetrahedra) and the reconstruction is a Poisson problem. The averaging is nonlinear—the rotational components of the deformation gradients are extracted and nonlinearly blended (*e.g.*, by taking the shortest path in the rotation group). Since the matrix in the Poisson problem does not change, a sparse factorization of the matrix can be computed and used to solve the systems, which makes the scheme very fast. A problem, however, is that the blending of the rotations is done separately for each triangle. This leads to undesirable interpolation results with large distortions. We refer to [Kircher and Garland 2008; Chu and Lee 2009; Winkler et al. 2010] for examples and a thorough discussion of this problem. To compensate for these effects, Kirchner and Garland [2008] modify the averaging process and describe the rotations relative to coordinate frames that vary over the surface. With the same goal, Chu and Lee [2009] use machine learning for improving the consistency of the blending of the rotations over the surface. For both methods, the price to pay to improve the interpolation quality is a loss in computational speed. These methods run at real-time rates only for coarse meshes. Another example of the first kind uses the *linear rotation invariant* coordinates proposed by Lipman et al. [2005]. To represent a surface mesh in a rotation-invariant way, they combine coordinate frames located at the vertices and connection maps, which describe transformations between neighboring frames. These coordinates are well-suited to represent deformations with rotational components (*e.g.* twists), but have problems dealing with deformations that include stretching. We refer to the survey [Botsch and Sorkine 2008] for an in-depth discussion and experimental comparison of various linear deformation models. Linear rotation invariant coordinates are used for *semantic deformation transfer* by Baran et al. [2009]. Their approach allows for transferring deformations between shapes without establishing an explicit mapping between the shapes. Instead poses of the models are matched and shape interpolation is used to transfer the deformations.

Examples of the second kind are the methods of Winkler et al. [2010] and Martin et al. [2011]. The geometric quantities used in the first scheme are the edge lengths and dihedral angles of triangles of a surface mesh and in the second the strain tensors of the tetrahedra of a volume mesh. Due to the nonlinear optimization needed for the reconstruction, the methods run at real-time rates only for very coarse meshes. Fröhlich and Botsch [2011] introduce a fast approximation scheme for the interpolation based on edge lengths and dihedral angles. Their scheme interpolates between simplified meshes and uses *deformation transfer* [Botsch et al. 2006] to map the coarse interpolated shapes to a fine mesh. In Section 6, we compare results and timings of this scheme with our approach. Another example of nonlinear coordinates are the *Pyramid coordinates* introduced by Sheffer and Kraevoy [2004].

Related to shape interpolation are Riemannian metrics on shape spaces. Such metrics measure the metric distortion or the viscous dissipation required to physically deform an elastic object [Kilian et al. 2007; Chao et al. 2010; Wirth et al. 2011; Heeren et al. 2012]. The corresponding Riemannian distance between two shapes is the infimum of the length over all curves connecting the shapes. A curve realizing this distance (a shortest geodesic) naturally interpolates between the two shapes in a shape space with a Riemannian metric. Weighted averages of more than two shapes could be defined using the *Riemannian center of mass* [Grove and Karcher 1973]. This is a very promising direction, but computationally much more involved than the direction we take here.

In addition to the interpolation of curved surfaces and volumes, the interpolation of planar shapes is an active research topic. For an overview, we refer to the recent paper of Chen et al. [2013] and references therein.

Creating continuous motions of objects and characters is an important problem in computer animation and is linked to shape interpolation. Typically, motions are created from keyframes (poses and times) that are interpolated. However, in contrast to shape interpolation usually wiggly (instead of smooth) motions, which exhibit secondary motion effects, are desired and the control of velocities is essential. Furthermore, the keyframes are ordered by their interpolation times and are successively interpolated—hence interpolation between more than two shapes is not considered. Traditionally, splines are used to interpolate between the poses. The space-

Fig. 1. Snapshots from an interactive shape interpolation session with five example poses of an armadillo shell model with 332k triangles. After a preprocess, our framework for nonlinear shape interpolation computes weighted average poses in real-time (see Table I).

time constraints paradigm, introduced by Witkin and Kass [1988], is closer to our work. The goal is to simplify the creation of realistic motion by combining physical simulation with keyframe interpolation. Motions are computed by solving nonlinear constrained spacetime optimization problems and model reduction techniques are used to accelerate the computations, see [Barbič et al. 2009; Hildebrandt et al. 2012]. Recently, spacetime optimization of the dynamics of elastic objects has been used for interactive editing of simulations and animations [Barbič et al. 2012; Li et al. 2014] and for creating motions that interpolate a set of partial keyframes [Schulz et al. 2014].

3. BACKGROUND: DEFORMATION ENERGIES

One ingredient to the shape interpolation approach we describe in the next section is a deformation energy, which is a measure of the energy stored in any deformed configuration of a shape away from its reference configuration. In our experiments, we focused on deformation energies derived from elasticity. However, other types of deformation energies may be used as well.

The presentation of the *elastic shape averaging* in the next section simplifies if we use a differential geometric approach to describe elastic potentials. In particular, we consider a manifold $\mathcal{M}$ and represent the reference and deformed configuration of an elastic object using maps from $\mathcal{M}$ to $\mathbb{R}^3$. Whereas this setting is commonly used for describing deformations of elastic shells, for elastic solids, a simpler setting in which the reference configuration is a domain in $\mathbb{R}^3$ and a deformation is given by a mapping from this domain to $\mathbb{R}^3$ is often considered. Therefore, we briefly review basics concerning potentials of elastic solids and introduce our notation. For a general introduction to the theory of elasticity from a differential geometric point of view, we refer to [Marsden and Hughes 1994] and the introductory book [Ciarlet 2005].

Consider an elastic body whose reference configuration is described by a (regular enough) map $X{:}\mathcal{M} \to \mathbb{R}^3$, where $\mathcal{M}$ is a three-dimensional manifold. A deformed configuration of the body is given by a map $Y{:}\mathcal{M} \to \mathbb{R}^3$. Here, $\mathcal{M}$ can be a domain in $\mathbb{R}^3$, e.g., the reference configuration, or an abstract manifold. In the case that $\mathcal{M}$ is the reference configuration, the map X is the embedding of $\mathcal{M}$ in $\mathbb{R}^3$. Our first step is to define the strain tensor $\mathcal{E}$, which provides a complete description of the strain (*e.g.* stretching, shearing, twisting) induced by a deformation. The definition involves the Riemannian metrics g_X and g_Y on $\mathcal{M}$ induced by the maps X and Y. For every point $p \in \mathcal{M}$, g_X (and analogously g_Y) provides a scalar product on $T_p\mathcal{M}$ that is given by

$$g_X(\phi, \psi) = \langle dX(\phi), dX(\psi) \rangle ,$$

where $\phi, \psi \in T_p\mathcal{M}$ are tangential vectors, dX is the differential of X, and $\langle \cdot, \cdot \rangle$ is the standard scalar product of $\mathbb{R}^3$. The *Cauchy–Green tensor* is the unique g_X-symmetric tensor field $\mathcal{C}$ that for every $p \in \mathcal{M}$ satisfies

$$g_X(\phi, \mathcal{C}\psi) = g_Y(\phi, \psi)$$

for all $\phi, \psi \in T_p\mathcal{M}$. Then, the strain tensor

$$\mathcal{E} = \frac{1}{2}(\mathcal{C} - \mathcal{I}),$$

induced by the deformation from X to Y measures the deviation of $\mathcal{C}$ from the identity $\mathcal{I}$. For example, a deformation is isometric if g_X equals g_Y. Then, $\mathcal{C}$ is the identity and the strain tensor vanishes.

The stresses in an object resulting from a deformation depend on properties of its material. In an *elastic* material, the stresses depend only on the reference and deformed configurations X, Y and are independent of the deformation path and speed. We consider *hyperelastic* materials, for which additionally the energy stored in a deformation Y is independent of the deformation path. For such materials, the energy stored in the deformation from the reference configuration X to the configuration Y is given by a function $\Omega(X, Y)$, which is called the potential. The stresses in a hyperelastic material can be described using the derivatives of the potential.

A material is called *objective* if considering the object from a rotated point of view, results in stresses that are transformed with the same rotation. For an objective material, there is an energy density function W that depends on $\mathcal{M}$ and $\mathcal{E}$ such that the potential can be written as an integral

$$\Omega(X, Y) = \int_{\mathcal{M}} W(p, \mathcal{E}(p)) \mathrm{d}vol_X , \tag{1}$$

where $\mathrm{d}vol_X$ is the volume form on $\mathcal{M}$ induced by X. A material is *isotropic* if $W(p, \cdot)$ depends only on the eigenvalues $\lambda_1, \lambda_2, \lambda_3$ of $\mathcal{E}(p)$ instead of $\mathcal{E}(p)$ itself. For such materials the potentials $\Omega(X, Y)$ are invariant under rigid transformations of X and Y. The material is called *homogeneous* if the energy density does not depend on $\mathcal{M}$. Then all material parameters are constant on $\mathcal{M}$. An example is the homogeneous St. Venant–Kirchhoff (StVK) material with density function

$$W_{\mathrm{StVK}}(\lambda_1, \lambda_2, \lambda_3) = \frac{\alpha}{2}(\lambda_1 + \lambda_2 + \lambda_3)^2 + \beta\,(\lambda_1^2 + \lambda_2^2 + \lambda_3^2).$$

Here α and β are material parameters.

4. ELASTIC SHAPE AVERAGING

Elastic shape averaging is a concept for computing weighted averages of a set of deformable objects. It was introduced by Rumpf and Wirth [2009] in the context of image processing and was formulated for implicitly defined objects in 2D and 3D images. Here we use this concept for interpolating between explicitly defined objects, *e.g.* immersed manifolds (in the continuous case) and triangle or tet meshes (in the discrete case). Since the example shapes are immersions of the same manifold (or meshes with the same connectivity in the discrete setting), correspondences between them are implicitly given and the interpolation is defined with respect to these correspondences. In contrast, in the work of Rumpf and Wirth the optimization is performed over all possible correspondences.

We consider m deformable objects whose reference configurations are given by maps $X_k{:}\mathcal{M} \to \mathbb{R}^3$. Then, we have m elastic potentials

$$\Omega_k(\cdot) = \Omega(X_k, \cdot). \tag{2}$$

For any configuration $Y{:}\mathcal{M} \to \mathbb{R}^3$, $\Omega_k(Y)$ measures the energy stored in the k^{th} object when it is deformed to the configuration Y. For a vector $\omega = (\omega_1, \omega_2, ..., \omega_m)$ of m positive weights, we consider the weighted sum of the elastic potentials Ω_k. The weighted average shape $A(\omega)$ corresponding to a weight vector ω is defined as the configuration that minimizes the weighted sum of the Ω_k

$$A(\omega) = \arg\min_{Y} \sum_{k=1}^{m} \omega_k \, \Omega_k(Y). \tag{3}$$

The idea is to deform m objects X_k to the same configuration Y and we search for the configuration whose total energy (weighted with the weights ω_k) required to deform all m objects is minimal.

In the following, we discuss some properties of (this variant of) elastic shape averaging:

—The weighted average shape is *independent of the choice of a base manifold* $\mathcal{M}$: If $\tilde{\mathcal{M}}$ is a manifold and $\phi : \tilde{\mathcal{M}} \to \mathcal{M}$ a diffeomorphism, then the weighted average shapes $A(\omega) : \mathcal{M} \to \mathbb{R}^3$ and $\tilde{A}(\omega) : \tilde{\mathcal{M}} \to \mathbb{R}^3$ satisfy $\tilde{A}(\omega) = A(\omega) \circ \phi$. The reason is that the elastic potentials are independent of the base manifold. For configurations $X, Y : \mathcal{M} \to \mathbb{R}^3$ and corresponding configurations $\tilde{X}, \tilde{Y} : \tilde{\mathcal{M}} \to \mathbb{R}^3$, the potentials agree

$$\Omega(X, Y) = \tilde{\Omega}(\tilde{X}, \tilde{Y}).$$

As a consequence the weighted sum of the potentials (3) is independent of the choice of the base manifold.

—*Homogeneity*. The map A, which maps a weight vector ω to the weighted average shape $A(\omega)$, is *homogeneous*

$$A(\omega_1, \omega_2, ..., \omega_m) = A(\xi\omega_1, \xi\omega_2, ..., \xi\omega_m) \text{ for any } \xi \in \mathbb{R}^+.$$

The reason is that when ω is scaled by a factor $\xi \in \mathbb{R}^+$, the objective functional in (3), which is the weighted sum of the potentials, scales with ξ as well. Since scaling the objective functional does not change the minimizers, the weighted average shapes remain the same. A consequence of homogeneity is that it suffices to consider weights ω_k that sum to one.

—*Invariance under rigid motion.* The weighted average shape is only determined up to rigid motion and it does not change if any of the example shapes X_k is transformed by a rigid motion. This property is a consequence of the fact that the elastic potentials, $\Omega(X, Y)$, are invariant under rigid transformations of X and Y.

—*Lagrange property.* The weighted average shape $A(e_k)$ equals X_k (up to rigid motion), where $e_k = (0, 0, ..., 0, 1, 0, ..., 0)$ is the k^{th} standard basis vector of $\mathbb{R}^m$. This means that the weighted average shape of the weight vector e_k is the k^{th} example shape. The reason is that in this case the objective functional in (3) agrees with the potential Ω_k and the minimizer of Ω_k is the reference configuration X_k.

—The weighted average shape is determined by the reference configurations $\{X_1, X_2, \ldots, X_m\}$, their material properties, and the weight vector ω. It is independent of any ordering on the set of reference shapes. This independence follows from the fact that the order of summands in (3) can be altered while the sum remains the same.

4.1 Elastic shape averaging for meshes

In the discrete setting, the reference configurations and the weighted average shapes are tet or triangle meshes that have the same connectivity. The abstract manifold $\mathcal{M}$ is replaced by an abstract simplicial manifold and the reference and weighted average shapes are given by embeddings of the simplicial manifold in $\mathbb{R}^3$. For the implementation this means that we have only one indexed face list (encoding the abstract simplicial manifold) which is used for all meshes. Then, the reference configurations and the weighted average shapes are given by $3n$-vectors (where n is the number of vertices) that list the coordinates in $\mathbb{R}^3$ for every vertex. We use lower-case letters to denote the discrete reference shapes x_k and deformations y.

In addition, we will need elastic potentials for the discrete shapes. Since the maps that describe deformations between meshes with the same connectivity are continuous and piecewise linear, we can directly use finite element discretizations with linear Lagrange elements. For details on finite element discretizations for elasticity, we refer to [Bonet and Wood 2008] and for open source implementations of different materials to the VegaFEM library [Sin et al. 2013]. In our experiments, we used finite element discretizations of elastic solids (tet meshes) with homogeneous *St. Venant–Kirchhoff* (StVK) and *Mooney–Rivlin* (MR) materials. For elastic shells (triangle meshes), we used *Discrete Shells* (DS) [Grinspun et al. 2003; Hildebrandt et al. 2010; Heeren et al. 2012] and *As-Rigid-As-Possible* (ARAP) [Sorkine and Alexa 2007; Chao et al. 2010; Jacobson et al. 2012]. The resulting discrete deformation energies are nonlinear functions $\mathcal{W} : \mathbb{R}^{3n} \times \mathbb{R}^{3n} \to \mathbb{R}$, where the first argument is the list of vertex positions of the reference configuration and the second argument specifies the deformed configuration.

Fig. 2. Weighted elastic averages of three Neptune solids with 691k tets using the nonlinear Mooney–Rivlin material computed at rates around 100 interpolated shapes per second (see Table I).

We define the discrete energies

$$\mathcal{W}_k(\cdot) = \mathcal{W}(x_k, \cdot)$$

and the weighted average shapes

$$a(\omega) = \arg\min_y \sum_{k=1}^{m} \omega_k \, \mathcal{W}_k(y) \tag{4}$$

analogous to (2) and (3). To compute a weighted average shape, a nonlinear optimization problem in $\mathbb{R}^{3n}$ has to be solved.

5. REAL-TIME SHAPE INTERPOLATION

In this section, we introduce an efficient scheme for the approximation of weighted average shapes that achieves interactive response times even for complex geometries and general nonlinear potential energies. The idea is to replace the complex optimization problem (4) by a reduced problem that can be solved much faster. One design principle is that the reduced problem should be independent of the resolution of the shapes that are to be averaged. We achieve this goal by combining two components: a low-dimensional affine subspace of the shape space, specifically designed for the interpolation problem, and a scheme for the fast approximation of the reduced objective functional and its gradient. The motivation for using this strategy is the special structure of the shape interpolation problem. To represent the geometries accurately, detailed meshes are needed. Compared to this, the number m of example shapes is very small. Therefore, the variation within the set of interpolated shapes is of low rank: all possible weighted average shapes can be parametrized by $m-1$ parameters. We want to exploit the resulting correlations between the degrees of freedom in the high-dimensional optimization problem.

5.1 Subspace construction

The first ingredients to construct the affine subspace are the example configurations x_k. We use $\tau = x_1$ as a reference point for the affine subspace and collect the vectors $x_k - x_1$ for $k \in \{2, 3, ..., m\}$ in a set $\mathcal{V}$. The affine space will be the linear span of $\mathcal{V}$ attached to the reference point τ. Including the difference vectors in $\mathcal{V}$ ensures that the resulting space includes all the example shapes.

For the second ingredient, we use the derivatives of the weighted average $a(\omega)$ with respect to the variation of ω. For any variation of ω around a fixed ω_0, the corresponding derivative of the weighted average shape is a vector in $\mathbb{R}^{3n}$. The linear span of all these vectors attached to the shape $a(\omega_0)$ is the $(m-1)$-dimensional affine subspace of $\mathbb{R}^{3n}$ that locally around $a(\omega_0)$ best approximates the set of the weighted average shapes. One can think of this affine space as the tangent space at $a(\omega_0)$ of the submanifold of $\mathbb{R}^{3n}$ consisting of all interpolated shapes. We want to compute the smallest affine subspace of $\mathbb{R}^{3n}$ that contains the union of the tangent spaces of the m example shapes. To do this, we compute vectors spanning the tangent spaces at the m example shapes and add these vectors to the set $\mathcal{V}$. Since the set $\mathcal{V}$ also contains the difference vectors $x_k - x_1$, the linear span of $\mathcal{V}$ attached to the reference point τ is the desired affine space.

To compute the tangent space at an interpolated shape, we need the derivatives of the energies $\mathcal{W}_k$: the forces F_k and the tangential stiffness matrices (or Hessians) K_k. They are defined as

$$F_k = -\mathrm{D}\mathcal{W}_k \quad \text{and} \quad K_k = -\mathrm{D}F_k = \mathrm{D}^2\mathcal{W}_k.$$

The vector $F_k(y)$ is the interior force resulting from deforming x_k into y. By definition, at a weighted average $a(\omega)$ the weighted sum of these forces is balanced

$$0 = \sum_k \omega_k F_k(a(\omega)). \tag{5}$$

By differentiating this equation, we obtain the derivative of the weighted average shape, which we summarize in the following lemma. We postpone the proof to the appendix.

LEMMA 1. *If all $\mathcal{W}_k$s are twice differentiable at $a(\omega)$ and a is differentiable at ω, then the derivative of $a(\omega)$ in direction $\nu \in \mathbb{R}^m$ satisfies*

$$\sum_k \omega_k \, K_k(a(\omega)) D_\nu a(\omega) = \sum_k \nu_k F_k(a(\omega)). \tag{6}$$

The matrix $\sum \omega_k \, K_k(a(\omega))$ is the Hessian of the objective functional (4) at $a(\omega)$ and $F_k(a(\omega))$ is the stress in the shape x_k when it is deformed to the configuration $a(\omega)$. For the construction of the subspace, we compute the derivatives at the example shape x_k. Using (6), we get

$$K_k(a(e_k))\mathrm{D}_{e_j} a(e_k) = F_j(a(e_k)).$$

The linear subspace of $\mathbb{R}^{3n}$ spanned by all the derivatives at $a(e_k)$ is spanned by the $m-1$ vectors $\mathrm{D}_{e_j} a$, where $j \in \{1, 2, ..., k-1, k+1, ..., m\}$. The vector $\mathrm{D}_{e_k} a(e_k)$ vanishes because $F_k(a(e_k)) = 0$. We denote this tangent space by T_k. We compute all $m-1$ derivatives at the m example shapes and add the resulting vectors to $\mathcal{V}$. For efficiency, we compute sparse factorizations of the matrices K_k and use each factorization to obtain all $m-1$ derivatives at the corresponding $a(e_k)$. The set $\mathcal{V}$ contains $m^2 - 1$ vectors.

In the following, we describe a strategy for extending the set $\mathcal{V}$, which can be used to improve the approximation of the set of interpolated shapes. Such a strategy is particularly helpful when the number m of shapes to be averaged is small. For example, for interpolating between two shapes, we get three-dimensional spaces. In such a case, we want to be able to add more variability to the subspaces. Still, when experimenting with the three-dimensional spaces, we found that they can already contain surprisingly good approximations of the unreduced interpolated shapes (compare Table III). One strategy for extending the subspace would be to compute samples, *e.g.*, interpolated shapes for some random weight vectors. The drawback of this approach is that computing the samples is costly, in particular, for the case of shells and highly-resolved meshes.

We propose a strategy that adapts the concept of *Ritz vectors*, which was introduced by Wilson et al. [1982], to our setting and involves only the forces F_k and the tangent stiffness matrices K_k at the configutations x_k. Ritz vectors are an established technique for the dimension reduction of physical systems (see [Lülf et al. 2013] for a recent comparison of dimension reduction techniques). To compute Ritz vectors, a set of loads has to be specified. In our setting, these loads are the forces $F_l(x_k)$ acting on the shape x_l when it is deformed to the configuration x_k. The Ritz vectors provide us with shape deformations corresponding to these forces. In other words, the Ritz vectors allow us to use information about the shape interpolation problem to construct degrees of freedom for interpolating between a specific set of example shapes.

For each of the m tangent spaces T_k, we compute the first $\rho + 1$ elements of the Krylov-type sequence

$$\{T_k, K_k^{-1} M_k T_k, \left(K_k^{-1} M_k\right)^2 T_k, ...\},$$

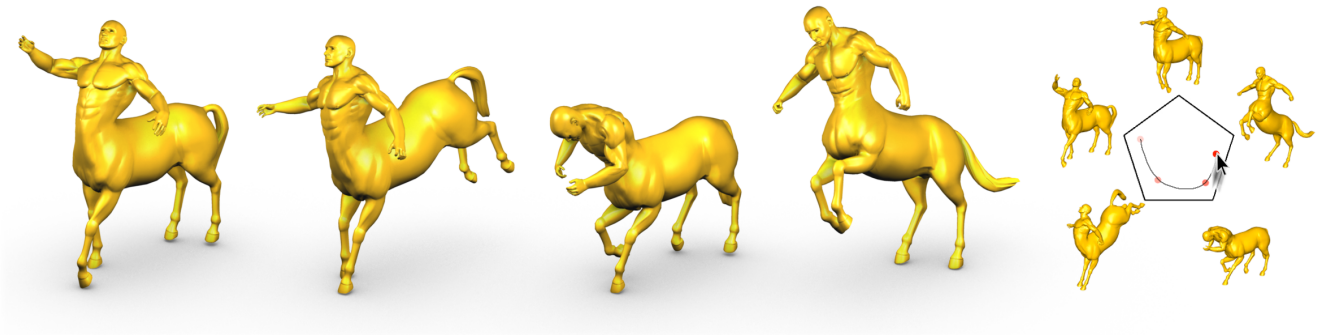

Fig. 3. Weighted elastic average shapes computed in real-time. On the right is a user-interface for exploring the space of shapes spanned by five example poses of the centaur shell model.

where M_k is the mass matrix of the example shape x_k, and add bases of these spaces to the set $\mathcal{V}$. This can be done by computing the corresponding Krylov sequences for all the tangent vectors $\mathrm{D}_{e_j} a(e_k)$ and adding them to $\mathcal{V}$.

Directly computing this sequence is numerically unstable, so we use an algorithm that produces a sequence that has the same linear span. The construction of the subspace basis is listed in Algorithm 1. The computation of the Krylov sequences is the inner *for*-loop running over the index "l". The vectors produced by the inner *for*-loop are pairwise M_k-orthogonal. To ensure this, it suffices to orthogonalize only against the last two elements of the sequences. This technique is borrowed from the Hermitian Lanczos algorithm (see [Saad 1992]). The computation of the Krylov sequences is fast because we can re-use the sparse factorizations we already used to determine the tangent vectors. In addition to the sequences generated from the tangent vectors, one can use the sequence generated from the difference vectors $x_k - x_j$ between the positions of the shapes. After we collected the difference vectors, the tangent vectors, and Krylov sequences in the set $\mathcal{V}$, we use an SVD to compute an orthonormal basis of the linear span of $\mathcal{V}$. Then the affine subspace we use for the shape interpolation is the linear span of $\mathcal{V}$ attached to the reference point τ.

Algorithm: Subspace construction
Data: Example poses $\{x_k\}_{k\in\{1,2,\dots,m\}}$
Result: Subspace basis $\{b_1, b_2, \dots, b_d\}$
$\mathcal{V} \leftarrow \{\}$; $v_0 \leftarrow \vec{0}$;
for $k = 2, 3, \dots, m$ **do**
  $\mathcal{V} \leftarrow \mathcal{V} \cup \{x_k - x_1\}$;
for $j, k = 1, 2, \dots, m; j \neq k$ **do**
  Solve $K_k\, w = F_j$;
  $\beta_1 \leftarrow \|w\|_{M_k}$;
  $v_1 \leftarrow w/\beta_1$;
  for $l = 1, 2, \dots \rho$ **do**
    Solve $K_k\, w = M_k\, v_l$;
    $\alpha_l \leftarrow \langle w, v_l \rangle_{M_k}$;
    $w \leftarrow w - \alpha_l v_l - \beta_l v_{l-1}$;
    $\beta_{l+1} \leftarrow \|w\|_{M_k}$;
    $v_{l+1} \leftarrow w/\beta_{l+1}$;
  $\mathcal{V} \leftarrow \mathcal{V} \cup \{v_1, \dots, v_{\rho+1}\}$;
$\{b_1, b_2, \dots, b_d\} \leftarrow$ compute basis of span$(\mathcal{V})$;

Algorithm 1: Construction of the subspace for shape interpolation.

5.2 Reduced functional and gradient approximation

Once a subspace basis $\{b_1, b_2, \dots, b_d\}$ and reference point τ are computed, we can restrict the optimization problem (4) to this affine subspace of $\mathbb{R}^{3n}$. Let $U = [b_1, \dots, b_d]$ be the matrix whose columns are the basis vectors. Then, the reduced optimization problem is

$$\tilde{a}(\omega) = \operatorname*{arg\,min}_{q\in\mathbb{R}^d} \sum_{k=1}^{m} \omega_k\, \tilde{\mathcal{W}}_k(q), \tag{7}$$

where $\tilde{\mathcal{W}}_k(q) = \mathcal{W}_k(Uq + \tau)$ and q is the vector of reduced coordinates. Solving the interpolation problem in the low-dimensional reduced space holds the promise of superior runtime performance. However, to attain real-time rates even for larger meshes, we need a strategy to evaluate the reduced objective functional (and its derivatives) at costs independent of the resolution of the example shapes. This, in turn, requires an efficient evaluation or approximation of the reduced energies $\tilde{\mathcal{W}}_k$ which itself is a challenging problem for general complex materials and arbitrary geometries.

A method that achieves resolution independence, due to Barbič and James [2005], used the fact that for the special case of linear materials, the (reduced) energy is a multivariate quartic polynomial in the (reduced) coordinates. By precomputing the coefficients of the reduced polynomial, an exact evaluation of the reduced energy is achieved at a cost complexity that scales $O(d^4)$. While we can employ this scheme for the materially linear St. Venant–Kirchhoff model, we have to resort to approximation for general nonlinear materials. To this end, we adopt the optimized cubature by An et al. [2008]. Their approach solves a best subset selection problem to determine a set of cubature points and weights that minimize a fitting error of the reduced forces to training data. The number of cubature points necessary was found to grow linearly with the subspace dimension, thus furnishing fast reduced evaluation at $O(d^2)$ cost. In our implementation, we apply the automatic training pose generation by von Tycowicz et al. [2013] to each example pose and then compute separate cubatures for the $\tilde{\mathcal{W}}_k$s. We also employ their NN-HTP solver for fast cubature optimization. An option for the shell models is the coarsening-based approach presented in [Hildebrandt et al. 2011]. This technique constructs a low-resolution version of $\mathcal{M}$ (called a *ghost*) together with a reduced space. By construction, both reduced spaces are isomorphic and we use the isomorphism to pull the energy from the subspace of the ghost to the subspace of the full mesh thus achieving approximation costs independent of the resolution of the full mesh. Due to the lower preprocessing costs (potentially at the price of approximation quality),

we prefer the coarsening approach over the cubature for the shell models in our experiments.

5.3 Solver

Even though the reduced problem is low-dimensional, it is still challenging to solve at real-time rates due to the nonlinear terms in the objective functional. An efficient family of solvers for this type of problem is the Broyden class of quasi-Newton methods. Members of the Broyden class construct a model of the objective functional that is good enough to produce superlinear convergence by measuring changes in gradients and hence do not require second derivatives to be supplied at each iteration. In particular, we opt for the BFGS method (see [Nocedal and Wright 2006]) that directly approximates the inverse Hessian of the objective functional avoiding costly linear system solves. To achieve a warm start of our solver, we compute the inverse of the reduced Hessian at the reduced mean shape once during the offline phase and use it as an initial inverse Hessian approximation for the BFGS solver in the online phase.

6. RESULTS AND DISCUSSION

We tested the shape interpolation scheme with different elastic potentials: (finite element discretizations of) St. Venant–Kirchhoff and Mooney–Rivlin solids, Discrete Shells, and the As-Rigid-As-Possible energy. For the shells, we employed the coarsening-based energy approximation; for the St. Venant–Kirchhoff and Mooney–Rivlin solids we used polynomial-based and cubature-based approaches, respectively. To set up the energy approximations, we used the configurations from the original publications, *i.e.*, we computed cubatures with about $3d$ (d being the subspace dimension) cubature points on 1k training poses, and, for the ghosts, we used coarse meshes with 500 to 1k vertices. In all of our experiments, we used homogeneous elastic materials. Furthermore, we use diagonal (or lumped) mass matrices M_k for which the diagonal entry is the mass of the corresponding vertex, which is a third of the combined area of the adjacent triangles and a fourth of the combined volume of the adjacent tets for shells and solids, respectively. We carried out experiments for both unconstrained models and models with equality constrained vertices that stay fixed throughout the interpolation. In the latter case, we color-coded the fixed vertices in gray in the figures and accompanying video.

In Table I we list statistics for both the offline and online phase of our framework for various geometries and parameter settings. This includes timings for the construction of the low-dimensional subspace and for setting up the quartic polynomial, cubature, and coarse mesh. Both steps dominate the offline phase and their computational cost depends mostly on the resolution and number of example shapes, and the approximation strategy.

To evaluate the runtime performance of our framework, we implemented a user interface that provides an intuitive metaphor for exploring the shapes contained in the span of the example poses. The user can drag a point in a two-dimensional control polygon for which each vertex corresponds to an example shape. Whenever the point is dragged, the framework computes an interpolated shape using the barycentric coordinates of the point as interpolation weights (see Fig. 3 for an illustration). Due to the resolution independence of our framework, the interpolated shapes are computed in fractions of a second yielding interactive feedback to the user. The accompanying video contains screen captured sequences of live shape exploration sessions. Additionally, we list runtime statistics for all our experiments in Table I. The armadillo experiment shown in Fig. 1 demonstrates the runtime performance. By restricting the 498k-dimensional optimization problem to a 64-dimensional subspace and employing energy approximation, our framework can determine reduced interpolated shapes in about 36 ms. Using a jCUDA-based matrix-vector multiplication we can map the reduced coordinates to the full 498k-dimensional space in 4 ms yielding response times of about 40 ms.

Table I. Performance statistics

Model	#v	W	m	d	$\mathcal{V}$	$\tilde{\mathcal{W}}$	BFGS
Armadillo	166k	DS	5	64	420s	8s	6ms (6)
		DS	2	41	178s	4s	3ms (4)
		ARAP	2	41	177s	4s	15ms (4)
Centaur	16k	DS	5	64	36s	2s	6ms (3)
Ch. dragon	130k	DS	2	17	121s	3s	1ms (2)
Dinosaur	28k	StVK	2	17	8s	338s	0.2ms (6)
Elephant	42k	DS	3	20	59s	2s	5ms (2)
Face	500k	DS	5	64	1456s	6s	7ms (3)
Hand	6k	DS	6	65	17s	0.5s	12ms (4)
Neptune	178k	MR	3	26	100s	1076s	2ms (5)
Livingstone elephant	40k	DS	2	39	41s	2s	3ms (3)
		ARAP	2	39	41s	2s	19ms (3)

Statistics measured on a 2012 MacBook Pro Retina 2.6GHz. From left to right: number of vertices, material, number of poses, subspace dimension, time for computation of subspace, time for construction of energy approximation, and time for one BFGS iteration (average number of iterations until convergence).

In addition to the shells, we also present experiments with tetrahedral meshes. We model the dinosaur (see Fig. 7 top row) as a St. Venant–Kirchhoff deformable object and the Neptune in Fig. 2 as an elastic solid with Mooney–Rivlin material, *i.e.*, a nonlinear constitutive model. For the latter, we computed cubatures for each

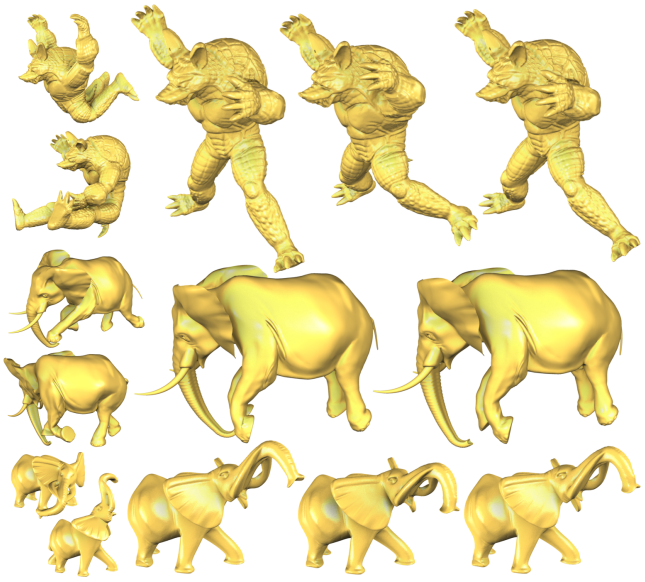

Fig. 4. **Comparison to previous work** from [Winkler et al. 2010] (top row), [Kircher and Garland 2008] (middle row), and [Fröhlich and Botsch 2011] (bottom row). From left to right in each row: example poses, mean shape from previous work and our method using the DS energy (for the armadillo and the Livingstone elephant we also provide our results using the ARAP energy).

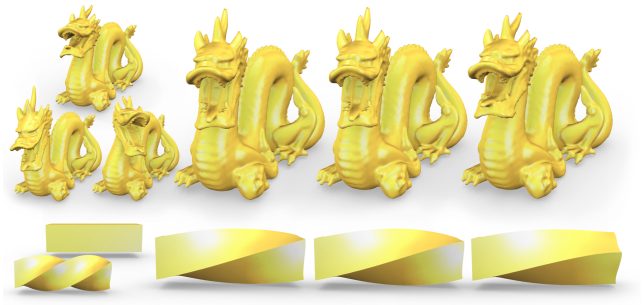

Fig. 5. **Comparison of subspace constructions**. From left to right in each row: example poses, unreduced mean shape, and mean shapes in reduced spaces based on our construction and the one from [Barbič et al. 2009] (see Tbl. II for details).

energy comprising 81 out of the 691k tetrahedra furnishing rates of about 100 interpolations per second.

Comparison to previous work. We compare our method to the state-of-the-art nonlinear interpolation schemes in [Kircher and Garland 2008], [Winkler et al. 2010], and [Fröhlich and Botsch 2011] for which results were kindly provided by their respective authors. Fig. 4 shows example poses and mean shapes obtained by our and previous work. The comparison illustrates that the proposed scheme is able to produce visually comparable interpolation results while featuring superior runtime performance. For example, for the interpolation of the armadillo model, Winkler et al. and Fröhlich and Botsch report timings of about 18.9 and 1.4 seconds per interpolated shape, respectively, while our framework computes interpolated shapes and maps them to the full space in only 15 milliseconds.

Additionally to the results with the *Discrete Shells* energy, we present interpolated shapes based on the *As-Rigid-As-Possible* energy for both the armadillo example by Winkler et al. and the Livingstone elephant example by Fröhlich and Botsch. These examples demonstrate that the choice of deformation energy affects the interpolation results. All shown mean shapes between the armadillo with straight legs and the armadillo with bent legs, for example, differ slightly by the extend to which the legs are bent. We provide a more detailed juxtaposition of the armadillo and elephant comparison (for which the authors provided densely sampled trajectories) in the accompanying video.

We are presenting the first subspace construction for the shape interpolation problem. However, constructions of subspaces have been introduced in other contexts. We are comparing results for shape interpolation in subspaces constructed with our method to the results obtained in subspaces using the construction introduced in [Barbič et al. 2009].

Before discussing the comparison, we want to emphasize that the construction in [Barbič et al. 2009] has not been introduced for the shape interpolation problem, but for generating realistic motion from keyframes. Therefore a different physical system is considered. Their goal is to modify the trajectory of an elastic solid such that it interpolates keyframes and exhibits secondary motion effects. Therefore, they consider an elastic object with a (fixed) reference configuration. In contrast, we are using elasticity to model shape interpolation and consider a fictitious force field that is derived from a combination of elastic potentials with different reference configurations (which are the example shapes). On a structural level there is a similarity between the constructions: both are

Table II. Comparison of subspace constructions

Model	#v	m	d	Our	Barbič et al.
Aorta	10k	2	3	1.61	15.48
			9	0.06	2.48
			17	0.06	0.66
Bar (90° twist)	4k	2	3	0.70	5.70
			9	0.14	3.20
			17	0.14	1.38
Bar (180° twist)	4k	2	3	5.84	26.75
			9	1.44	7.30
			17	0.59	6.80
Dragon	26k	3	8	0.18	5.65
			14	0.11	2.88
			32	0.05	0.38

From left to right: number of vertices, number of poses, subspace dimension, and relative L^2-errors (in 10^{-4}) for interpolation results obtained in reduced spaces using our construction and the one from [Barbič et al. 2009]. For each model, we compare subspaces constructed from differences and tangents only (1st row) and extended spaces of increasing dimension (2nd and 3rd row).

using the difference vectors between the shapes and certain tangent vectors. However, the tangents used are fundamentally different. In [Barbič et al. 2009] the tangents of so-called "deformation curves" are computed. These tangents (and curves) depend on the reference configuration of the elastic object considered (which is important for their application). In contrast, we are combining m elastic potentials whose rest configurations are the example poses to get the derivatives of the weighted elastic average shapes. The computation of the derivatives always mixes the Hessians and the gradients of elastic potentials with different rest configurations, which do not exist in the problem setting considered in [Barbič et al. 2009].

Our experiments comprise comparisons for interpolation problems of elastic (StVK) solids with different number of poses, geometric complexity, and magnitude of deformation. For each setting, we provide comparisons of the solutions obtained in reduced spaces of varying size. In particular, we employ spaces from difference and tangent vectors and spaces enriched with Ritz vectors and linear vibration modes, as proposed here and in [Barbič et al. 2009], respectively. The number of Ritz vectors and vibration modes have been chosen such that the resulting subspaces are of equal dimension. For example, adding three vibration modes around each ex-

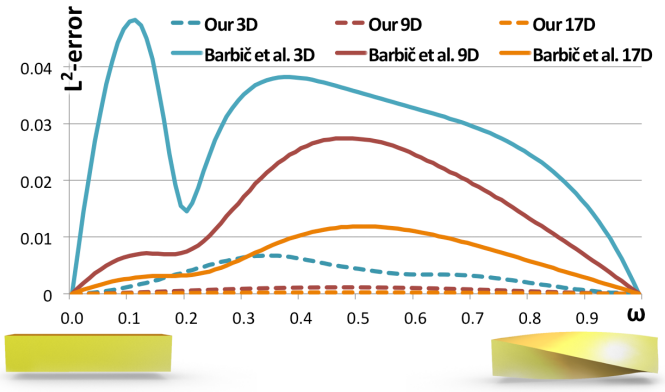

Fig. 6. **Approximation errors** for interpolating a straight and 90° twisted bar in reduced spaces of varying size using our construction and the one from [Barbič et al. 2009] (see also 2nd row in Tbl. II for relative L^2-errors).

ample pose (or three Ritz vectors for each tangent), we obtain 9-dimensional spaces for the two-pose examples. In Table II, we list details of the interpolation problems together with relative L^2-errors to the unreduced reference solutions (computed on about 100 interpolating shapes obtained by uniformly sampling the parameter domain). The results show that our proposed spaces are able to provide approximation errors that are an order of magnitude lower than those obtained by employing the construction from [Barbič et al. 2009]. A more detailed comparison of the approximation error is shown in Figure 6 which visualizes the L^2-error as a function of the interpolation weight ω_1 for the 90° twisted bar example. Additionally, in Figure 5 we show unreduced mean shapes in comparison to mean shapes computed in reduced spaces spanned only by differences and our tangent vectors and the ones from [Barbič et al. 2009].

Subspace accuracy. To further evaluate the fidelity of the reduced interpolation problem, we measured the deviation of the reduced from the reference-full solution for a solid (the dinosaur) and a shell (the Chinese dragon) model. In Table III, we list Hausdorff distances for solutions obtained in subspaces of varying size. The examples show that the proposed scheme is able to closely match the unreduced interpolated shapes even in very low-dimensional subspaces. In particular, the reduced Chinese dragon mean shape computed in a three-dimensional subspace is visually comparable to the unreduced one (see Fig. 7 bottom). The dinosaur, on the contrary, is an example where a three-dimensional space does not resolve the nonlinearities sufficiently illustrating the effect of our Krylov-type extension—adding Ritz vectors significantly improves the approximation performance of the reduced spaces (see Fig. 7 top). Though the result in the three-dimensional space is not a good approximation in terms of the Hausdorff distance, it is a smooth shape.

7. CONCLUSION

We introduce a scheme for nonlinear shape interpolation that is based on an offline-online framework. In the online phase, it allows for real-time computation of interpolated shapes. This makes the method well-suited for real-time applications that involve shape interpolation and for tools that need to compute a large number of interpolated shapes. The scheme is based on a novel construction of low-dimensional shape spaces that use the specific structure of the interpolation problem. The dimensional reduction is combined with schemes for the fast approximation of the nonlinear objective functional and gradient. The scheme is implemented for a broad approach to shape interpolation that we obtain by translating the approach for elastic shape averaging by Rumpf and Wirth [2009] from implicitly to explicitly described shapes.

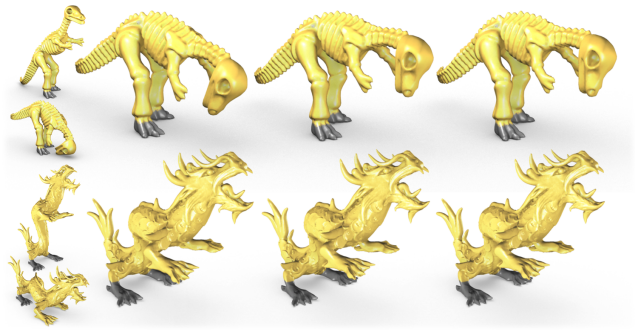

Fig. 7. **Subspace fidelity.** Mean shapes of the dinosaur and Chinese dragon model computed in spaces of varying size. From left to right: the two example poses, mean shapes obtained in a 3 dim., 17 dim., and unreduced space. (See Tbl. III for Hausdorff distances.)

Table III. Subspace fidelity

Model	#v	d (#rv)	Hausdorff distance 0.25	0.5	0.75
Dinosaur	28k	3 (0)	1.87	7.96	9.17
		9 (6)	0.49	0.87	1.63
		17 (14)	0.38	0.69	0.49
Chinese dragon	130k	3 (0)	1.16	1.73	1.46
		9 (6)	0.59	0.82	0.59
		17 (14)	0.18	0.24	0.17

From left to right: number of vertices, subspace dimension (number of Ritz vectors), and Hausdorff distances to the unreduced reference solution (divided by the length of the bounding box diagonal) for various weights. Example poses and mean shapes are shown in Fig. 7.

Limitations and Challenges. Our current construction of the subspaces leaves room for improvement. Instead of a straightforward computation of the tangents and Krylov sequences on the large meshes, one could try to develop efficient approximation techniques. In addition, the computation of the basis vectors could be done in parallel for all m shapes (or even for all Krylov sequences). In our present framework, there is no control over the approximation error. An interesting problem would be to integrate an accuracy control into the subspace construction. Another direction for future work is to use the subspace construction for accelerating the computation of geodesics in shape space. Furthermore, it would be interesting to test the averaging approach with geometrically motivated energies, *e.g.*, to get weighted averages that are invariant under affine (or Möbius) transformations of the example shapes. Recent schemes for deformation-based modeling [Gal et al. 2009; Wu et al. 2014] preserve automatically detected or user-specified structure of a shape, *e.g.* symmetries or planarity constraints, during modeling. An interesting question is how such structural information could be integrated to shape interpolation.

ACKNOWLEDGMENTS

We would like to thank Marc Alexa, Mario Botsch, Jens Driesenberg, Stefan Fröhlich, Michael Garland, Kai Hormann, Scott Kircher, and Tim Winkler for sharing results of their methods, Mimi Tsuruga for proofreading the text, and the anonymous reviewers for their comments and suggestions. This work was supported by the DFG Project *KneeLaxity: Dynamic multi-modal knee joint registration* (EH 422/1-1) and the Max Planck Center for Visual Computing and Communication (MPC-VCC).

APPENDIX

A. TANGENT SPACES

In this appendix, we derive the formula (6) for the derivative of the weighted average deformation with respect to variation of the weights ω.

PROOF OF LEMMA 1. At any weighted average deformation the forces pulling towards the shapes x_k are balanced, *i.e.*, Equation (5) is satisfied. Hence, the directional derivative of the ω-weighted

sum of the forces must vanish and we get

$$\begin{aligned} 0 &= \mathrm{D}_\nu \sum_k \omega_k F_k(a(\omega)) \\ &= \sum_k \nu_k F_k(a(\omega)) + \sum_k \omega_k \mathrm{D}_\nu F_k(a(\omega)) \\ &= \sum_k \nu_k F_k(a(\omega)) - \sum_k \omega_k K_k(a(\omega)) \mathrm{D}_\nu a(\omega), \end{aligned}$$

which verifies Equation (6). □

REFERENCES

AN, S. S., KIM, T., AND JAMES, D. L. 2008. Optimizing cubature for efficient integration of subspace deformations. *ACM Trans. Graph. 27,* 5, 1–10.

BARAN, I., VLASIC, D., GRINSPUN, E., AND POPOVIĆ, J. 2009. Semantic deformation transfer. *ACM Trans. Graph. 28,* 3, 36:1–36:6.

BARBIČ, J., DA SILVA, M., AND POPOVIĆ, J. 2009. Deformable object animation using reduced optimal control. *ACM Trans. Graph. 28,* 3, 53:1–53:9.

BARBIČ, J. AND JAMES, D. L. 2005. Real-time subspace integration for St. Venant-Kirchhoff deformable models. *ACM Trans. Graph. 24,* 3, 982–990.

BARBIČ, J., SIN, F., AND GRINSPUN, E. 2012. Interactive editing of deformable simulations. *ACM Trans. Graph. 31,* 4.

BONET, J. AND WOOD, R. D. 2008. *Nonlinear Continuum Mechanics for Finite Element Analysis*. Cambridge University Press. Cambridge University Press.

BOTSCH, M. AND SORKINE, O. 2008. On linear variational surface deformation methods. *IEEE Transactions on Visualization and Computer Graphics 14,* 1, 213–230.

BOTSCH, M., SUMNER, R., PAULY, M., AND GROSS, M. 2006. Deformation transfer for detail-preserving surface editing. In *Proceedings of Vision, Modeling & Visualization*. 357–364.

CHAO, I., PINKALL, U., SANAN, P., AND SCHRÖDER, P. 2010. A simple geometric model for elastic deformations. *ACM Trans. Graph. 29*, 38:1–38:6.

CHEN, R., WEBER, O., KEREN, D., AND BEN-CHEN, M. 2013. Planar shape interpolation with bounded distortion. *ACM Trans. Graph. 32,* 4, 108:1–108:12.

CHU, H.-K. AND LEE, T.-Y. 2009. Multiresolution mean shift clustering algorithm for shape interpolation. *IEEE Transactions on Visualization and Computer Graphics 15,* 5, 853–866.

CIARLET, P. G. 2005. *An Introduction to Differential Geometry with Applications to Elasticity*. Springer.

FRÖHLICH, S. AND BOTSCH, M. 2011. Example-driven deformations based on discrete shells. *Comp. Graph. Forum 30,* 8, 2246–2257.

GAL, R., SORKINE, O., MITRA, N., AND COHEN-OR, D. 2009. iWires: An analyze-and-edit approach to shape manipulation. *ACM Trans. Graph. 28,* 3.

GRINSPUN, E., HIRANI, A. N., DESBRUN, M., AND SCHRÖDER, P. 2003. Discrete shells. In *Symp. on Comp. Anim.* 62–67.

GROVE, K. AND KARCHER, H. 1973. How to conjugate C^1-close group actions. *Mathematische Zeitschrift 132*, 11–20.

HEEREN, B., RUMPF, M., WARDETZKY, M., AND WIRTH, B. 2012. Time-discrete geodesics in the space of shells. *Comp. Graph. Forum 31,* 5, 1755–1764.

HILDEBRANDT, K., SCHULZ, C., VON TYCOWICZ, C., AND POLTHIER, K. 2010. Eigenmodes of surface energies for shape analysis. In *Proceedings of Geometric Modeling and Processing*. 296–314.

HILDEBRANDT, K., SCHULZ, C., VON TYCOWICZ, C., AND POLTHIER, K. 2011. Interactive surface modeling using modal analysis. *ACM Trans. Graph. 30,* 5, 119:1–119:11.

HILDEBRANDT, K., SCHULZ, C., VON TYCOWICZ, C., AND POLTHIER, K. 2012. Interactive spacetime control of deformable objects. *ACM Trans. Graph. 31,* 4, 71:1–71:8.

JACOBSON, A., BARAN, I., KAVAN, L., POPOVIĆ, J., AND SORKINE, O. 2012. Fast automatic skinning transformations. *ACM Trans. Graph. 31,* 4, 77:1–77:10.

KILIAN, M., MITRA, N. J., AND POTTMANN, H. 2007. Geometric modeling in shape space. *ACM Trans. Graph. 26,* 3, 64:1–64:8.

KIRCHER, S. AND GARLAND, M. 2008. Free-form motion processing. *ACM Trans. Graph. 27,* 2, 12:1–12:13.

LI, S., HUANG, J., DE GOES, F., JIN, X., BAO, H., AND DESBRUN, M. 2014. Space-time editing of elastic motion through material optimization and reduction. *ACM Trans. Graph. 33,* 4, 108:1–108:10.

LIPMAN, Y., SORKINE, O., LEVIN, D., AND COHEN-OR, D. 2005. Linear rotation-invariant coordinates for meshes. *ACM Trans. Graph. 24,* 3, 479–487.

LÜLF, F. A., TRAN, D.-M., AND OHAYON, R. 2013. Reduced bases for nonlinear structural dynamic systems: A comparative study. *Journal of Sound and Vibration 332,* 15, 3897 – 3921.

MARSDEN, J. E. AND HUGHES, T. J. R. 1994. *Mathematical Foundations of Elasticity*. Dover Publications.

MARTIN, S., THOMASZEWSKI, B., GRINSPUN, E., AND GROSS, M. 2011. Example-based elastic materials. *ACM Trans. Graph. 30,* 4, 72:1–72:8.

NOCEDAL, J. AND WRIGHT, S. J. 2006. *Numerical Optimization (2nd edition)*. Springer.

RUMPF, M. AND WIRTH, B. 2009. A nonlinear elastic shape averaging approach. *SIAM J. on Imaging Science 2,* 3, 800–833.

SAAD, Y. 1992. *Numerical Methods for Large Eigenvalue Problems*. Manchester University Press.

SCHULZ, C., VON TYCOWICZ, C., SEIDEL, H.-P., AND HILDEBRANDT, K. 2014. Animating deformable objects using sparse spacetime constraints. *ACM Trans. Graph. 33,* 4, 109:1–109:10.

SHEFFER, A. AND KRAEVOY, V. 2004. Pyramid coordinates for morphing and deformation. In *Proceedings of Symposium on 3D Data Processing, Visualization, and Transmission*. 68–75.

SIN, F. S., SCHROEDER, D., AND BARBIČ, J. 2013. Vega: Non-linear FEM deformable object simulator. *Computer Graphics Forum 32,* 1, 36–48.

SORKINE, O. AND ALEXA, M. 2007. As-rigid-as-possible surface modeling. In *Proceedings of Eurographics/ACM SIGGRAPH Symposium on Geometry Processing*. 109–116.

SUMNER, R. W. AND POPOVIĆ, J. 2004. Deformation transfer for triangle meshes. *ACM Trans. Graph. 23,* 3, 399–405.

SUMNER, R. W., ZWICKER, M., GOTSMAN, C., AND POPOVIĆ, J. 2005. Mesh-based inverse kinematics. *ACM Trans. Graph. 24,* 3, 488–495.

VON TYCOWICZ, C., SCHULZ, C., SEIDEL, H.-P., AND HILDEBRANDT, K. 2013. An efficient construction of reduced deformable objects. *ACM Trans. Graph. 32,* 6, 213:1–213:10.

WILSON, E. L., YUAN, M.-W., AND DICKENS, J. M. 1982. Dynamic analysis by direct superposition of Ritz vectors. *Earthquake Engineering & Structural Dynamics 10,* 6, 813–821.

WINKLER, T., DRIESEBERG, J., ALEXA, M., AND HORMANN, K. 2010. Multi-scale geometry interpolation. *Comp. Graph. Forum 29,* 2, 309–318.

WIRTH, B., BAR, L., RUMPF, M., AND SAPIRO, G. 2011. A continuum mechanical approach to geodesics in shape space. *Int. J. Comput. Vision 93,* 3, 293–318.

WITKIN, A. AND KASS, M. 1988. Spacetime constraints. *Proc. of ACM SIGGRAPH 22*, 159–168.

WU, X., WAND, M., HILDEBRANDT, K., KOHLI, P., AND SEIDEL, H.-P. 2014. Real-time symmetry-preserving deformation. *Computer Graphics Forum 33,* 7, 229–238.

XU, D., ZHANG, H., WANG, Q., AND BAO, H. 2005. Poisson shape interpolation. In *Symp. on Solid and Phys. Mod.* 267–274.