Somigliana Coordinates: an elasticity-derived approach for cage deformation

JIONG CHEN, LIX, Ecole Polytechnique, IP Paris, France
FERNANDO DE GOES, Pixar Animation Studios, USA
MATHIEU DESBRUN, Inria / Ecole Polytechnique, France

In this paper, we present a novel cage deformer based on elasticity-derived matrix-valued coordinates. In order to bypass the typical shearing artifacts and lack of volume control of existing cage deformers, we promote a more elastic behavior of the cage deformation by deriving our coordinates from the Somigliana identity, a boundary integral formulation based on the fundamental solution of linear elasticity. Given an initial cage and its deformed pose, the deformation of the cage interior is deduced from these Somigliana coordinates via a corotational scheme, resulting in a matrix-weighted combination of both vertex positions and face normals of the cage. Our deformer thus generalizes Green coordinates, while producing physically-plausible spatial deformations that are invariant under similarity transformations and with interactive bulging control. We demonstrate the efficiency and versatility of our method through a series of examples in 2D and 3D.

CCS Concepts: • **Computing methodologies** → **Volumetric models**.

Additional Key Words and Phrases: Cage deformation, elasticity, volume control, free-form modeling.

ACM Reference Format:

Jiong Chen, Fernando de Goes, and Mathieu Desbrun. 2023. Somigliana Coordinates: an elasticity-derived approach for cage deformation. In *Special Interest Group on Computer Graphics and Interactive Techniques Conference Conference Proceedings (SIGGRAPH '23 Conference Proceedings), August 6–10, 2023, Los Angeles, CA, USA.* ACM, New York, NY, USA, 8 pages. https://doi.org/10.1145/3588432.3591519

1 INTRODUCTION

Cage deformers are ubiquitously used to generate high-quality spatial deformations driven by the articulation of an enclosing "cage" mesh. The majority of cage deformers are based on precomputed or closed-form generalized barycentric coordinates that interpolate the cage vertices while reproducing affine transformations [Hormann and Sukumar 2017]. However, these interpolatory coordinates tend to induce undesirable shearing artifacts. This motivated Lipman et al. [2008] to define the so-called Green coordinates using both cage vertices and normals as a means to favor quasi-conformal maps, thus reducing shearing and leading to better detail preservation. Unfortunately, the quasi-conformality of Green coordinates is notorious for causing significant and undesirable volume scaling, curbing their practical appeal. While subsequent work has partially alleviated these volume changes by combining Green coordinates with distortion minimization [Ben-Chen et al. 2009; Martin et al. 2009], it comes at the cost of iterative global solves, negating the attractive efficiency of cage deformers.

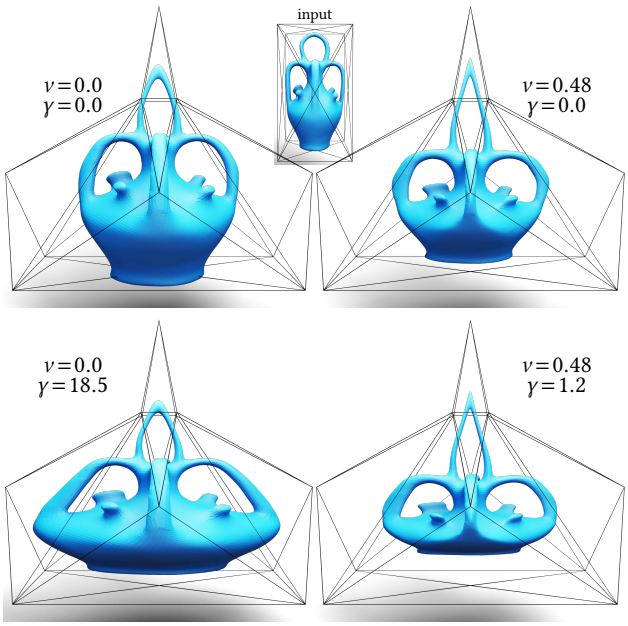


Fig. 1. **Somigliana coordinates.** Given an initial cage (top inset) and its deformed pose, our novel cage deformer promotes a more elastic behavior of the cage deformation than previous works by leveraging an elasticity-derived matrix-weighted combination of both vertex positions and face normals of the cage. A Poisson ratio ν and bulging scale γ can be adjusted to offer control over local and global volume change.

The usual lack of bulging and volume control in cage deformers also conflicts with the common desire to produce physically plausible deformation. For instance, many authors have employed quasi-static elastic simulations to provide more natural poses [McAdams et al. 2011; Smith et al. 2018]. However, these approaches are computationally expensive since they require volume discretizations and non-convex numerical optimizations. Model reduction techniques can accelerate elastic simulations by precomputing deformation modes [Barbič and James 2005] and multi-domain subspaces [Barbič and Zhao 2011] at the cost of increased memory footprint. The boundary element method (BEM) offers instead a surface-only representation

SIGGRAPH '23 Conference Proceedings, August 6–10, 2023, Los Angeles, CA, USA

to efficiently derive volume deformations of elastic solids [James and Pai 1999, 2003], but its use of dense linear solves is incompatible with real-time deformation. Recently, the design of sculpting brushes providing elastic deformation with interactive volume control through regularized Kelvinlets [de Goes and James 2017] has quickly gained popularity [Adobe 2022; Blender 2022], but these deformations are only local and oblivious to boundaries. While this limitation can be alleviated by constraining multiple regularized Kelvinlets, it comes again at the cost of a dense linear solve.

In this work, we introduce new cage coordinates that produce spatial deformation with interactive bulging and volume control. Our method is based on a corotational extension to an integral form of the static solution to the linear elasticity equation over a closed domain. Since this integral formulation is often referred to as the Somigliana identity, we name our new cage coordinates *Somigliana coordinates*. Our approach leads to a novel set of matrix-valued coordinates that mimic the elasticity response to displacements of the boundary, while still reproducing global similarity transformations. We show that these matrices can be efficiently precomputed from vertices and faces of a cage mesh and rapidly evaluated at arbitrary space locations, thus making our technique free of any volumetric discretization, numerical solve, or large memory footprint. We also exploit local and global geometric estimates based on the rest and posed cages in order to deduce boundary tractions and provide interactive bulging control. As a result, our Somigliana coordinates complement existing elasticity techniques by generating physically-plausible deformations entirely driven by the cage pose.

2 RELATED WORK

Generating spatial deformations quickly and intuitively has received considerable attention over the past four decades. For conciseness, we now review prior methods that are closely related to ours.

Generalized Barycentric Coordinates. Ever since the work of [Meyer et al. 2002], there has been growing interest in constructing generalizations of Möbius' simplicial barycentric coordinates to polygons and polyhedra for graphics applications [Floater 2015; Hormann and Sukumar 2017]. While many coordinates only apply to the restricted case of convex polytopes [Warren et al. 2007; Budninskiy et al. 2016], mean value coordinates are far more versatile since they can be computed in closed-form for cages of arbitrary shape [Floater 2003; Ju et al. 2005; Thiery et al. 2018]. Alternative approaches have also been proposed to enforce positive coordinates, hence reducing artifacts near concavities [Joshi et al. 2007; Lipman et al. 2007; Hormann and Sukumar 2008]; but they require either large memory consumption or customized numerical schemes. While generalized barycentric coordinates are interpolatory and affine-invariant by definition, our approach favors non-interpolatory coordinates that offer instead elasticity-derived volume control.

Green coordinates. To remove the usual shearing artifacts of interpolatory methods, Lipman et al. [2008] introduced Green coordinates by combining contributions of both cage vertices and faces. This approach was later extended to quadrangulated cage meshes [Thiery and Boubekeur 2022]. The resulting deformations are conformal in 2D and quasi-conformal in 3D and, thus, tend to present unreasonable volume scaling. Subsequent work [Ben-Chen et al. 2009; Martin et al. 2009] have mitigated these volume changes based on distortion optimizations at the cost of added computational complexity. In contrast, we incorporate volume control by deriving new coordinates based on the fundamental solutions of linear elasticity.

Complex Coordinates. As shown in [Weber et al. 2009], Green coordinates in 2D can be expressed as complex-valued coordinates assigned solely to vertex positions of a cage polygon. This complex representation led to a series of planar deformation techniques that account for boundary angle and pointwise constraints as well as bounded distortion—see [Weber 2017] and references therein. However, complex coordinates are limited to 2D and thus cannot express the bulging effects exhibited by elastic deformations in 3D.

Elastostatics. Our work draws inspiration from the quasi-static methods for linear elasticity. In [James and Pai 1999, 2003] for instance, the boundary element method (BEM) was leveraged to solve the equation of linear elasticity by deducing boundary tractions from surface displacements while avoiding volumetric discretization. However, BEM-based schemes require dense linear solves and repeated matrix updates, hampering their use for real-time editing. A faster alternative is the use of regularized Kelvinlets [de Goes and James 2017], which are sculpting brushes offering local and natural-looking elastic deformation. More recently, regularized Kelvinlets were extended to elasto-dynamics [de Goes and James 2018], sharp falloffs [de Goes and James 2019], and anisotropic materials [Chen and Desbrun 2022]. Unfortunately, these sculpting brushes ignore boundaries and, therefore, are poorly adapted to cage-based editing.

As we will show, our approach forms a bridge between prior methods by designing a cage deformer with vertex and face-based coordinates that inherently produce elasticity-derived deformation.

3 BACKGROUND

We begin the exposition of our contributions by reviewing the basics of linear elastostatics upon which our formulation is built — readers can refer to [Slaughter 2012] for more details about linear elasticity. Hereafter, we consider a domain $\Omega \subset \mathbb{R}^d$ $(d = 2, 3)$ describing a homogeneous and isotropic elastic material bounded by a piecewise-smooth manifold $\partial\Omega$. By convention, we use $\boldsymbol{n} : \partial\Omega \to \mathbb{R}^d$ to denote the outward unit normal field over the domain boundary $\partial\Omega$.

3.1 Elastostatics

Equilibrium in linear elasticity is determined by a displacement $\boldsymbol{u} : \Omega \to \mathbb{R}^d$ satisfying the Navier-Cauchy equation:

$$\mu \Delta \boldsymbol{u} + \frac{\mu}{1 - 2\nu} \nabla \left(\nabla \cdot \boldsymbol{u} \right) = \boldsymbol{b}, \tag{1}$$

where elastic shear modulus $\mu > 0$ indicates material stiffness, Poisson ratio $\nu < 1/2$ dictates material compressibility, and $\boldsymbol{b} : \Omega \to \mathbb{R}^d$ is the external body load. Given a solution of Eq. (1), we can also quantify the force distribution at any point $\boldsymbol{x} \in \Omega$ using the Cauchy stress tensor field $\boldsymbol{\sigma} : \Omega \to \mathbb{R}^{d \times d}$, which is expressed as

$$\boldsymbol{\sigma}(\boldsymbol{x}) = \mu \left[\nabla \boldsymbol{u}(\boldsymbol{x}) + \nabla \boldsymbol{u}(\boldsymbol{x})^t \right] + \frac{2\mu\nu}{1 - 2\nu} \left[\nabla \cdot \boldsymbol{u}(\boldsymbol{x}) \right] \boldsymbol{I}. \tag{2}$$

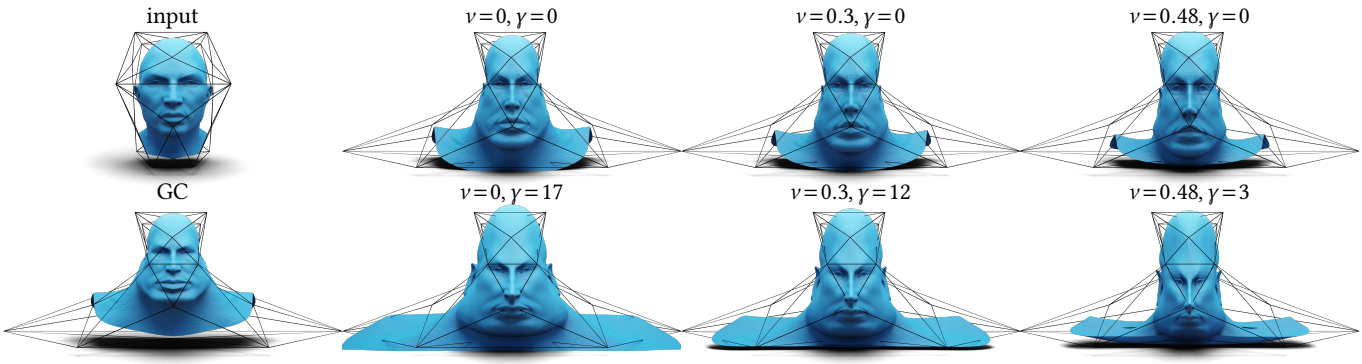


Fig. 2. **Extreme face lift**. For a simple rest cage enclosing a face mesh (top left), the same posed cage can produce a diverse set of results depending on the user-selected combination of Poisson ratio ν and bulging scale γ of our cage deformer — here using the global variant.

A traction vector at a point $\boldsymbol{x} \in \Omega$ is then obtained by evaluating the stress along any desired direction; in particular, the traction at a boundary point $\boldsymbol{x} \in \partial\Omega$ is defined via $\boldsymbol{\tau}(\boldsymbol{x}) = \boldsymbol{\sigma}(\boldsymbol{x})\boldsymbol{n}(\boldsymbol{x})$.

3.2 Fundamental Solutions

An important special case of Eq. (1) is when the domain is unbounded (i.e., $\Omega \equiv \mathbb{R}^d$) and is under a body load of the form $\boldsymbol{b}(\boldsymbol{x}) = \delta(\boldsymbol{x}-\boldsymbol{x}_0)\boldsymbol{f}$ centered at point $\boldsymbol{x}_0 \in \Omega$ for a given force vector $\boldsymbol{f}$. The response of Eq. (1) to this point load is then $\boldsymbol{u}(\boldsymbol{x}) = \mathcal{K}(\boldsymbol{x}, \boldsymbol{x}_0)\boldsymbol{f}$, where the kernel function $\mathcal{K}$ defines the fundamental solution of linear elasticity, also known as the Kelvinlet [Kelvin 1848]. Denoting the relative position vector $\boldsymbol{r} = \boldsymbol{x}-\boldsymbol{x}_0$ and its norm $r = \|\boldsymbol{r}\|$, the Kelvinlet solution is a matrix-valued function $\mathcal{K} : \Omega\times\Omega \to \mathbb{R}^{d\times d}$ expressed as:

$$\mathcal{K}(\boldsymbol{x}, \boldsymbol{x}_0) = \begin{cases} \dfrac{(a-b)}{r}\boldsymbol{I} + \dfrac{b}{r^3}\boldsymbol{r}\boldsymbol{r}^t & \text{in 3D,} \\ (a-b)\ln(1/r)\boldsymbol{I} + \dfrac{b}{r^2}\boldsymbol{r}\boldsymbol{r}^t & \text{in 2D,} \end{cases} \tag{3}$$

with coefficients $a = 1/\mu(2^{d-1}\pi)$ and $b = a/4(1-\nu)$ which concisely encapsulate the terms dependent on the material parameters μ and ν in dimension $d = 2, 3$.

We can also employ Eq. (2) to evaluate the traction associated with the Kelvinlet solution, thus defining the traction fundamental solution as a matrix-valued function $\mathcal{T} : \Omega\times\Omega \to \mathbb{R}^{d\times d}$ written as:

$$\mathcal{T}(\boldsymbol{x}, \boldsymbol{x}_0) = \frac{\mu(a-2b)}{r^d}\left[(\boldsymbol{n}^t\boldsymbol{r})\boldsymbol{I} + \boldsymbol{n}\boldsymbol{r}^t - \boldsymbol{r}\boldsymbol{n}^t\right] + \frac{2\mu bd}{r^{d+2}}(\boldsymbol{n}^t\boldsymbol{r})\boldsymbol{r}\boldsymbol{r}^t. \tag{4}$$

Notice that the first terms in Eqs. (3) and (4) resemble, respectively, the fundamental solution of the Laplacian equation and its normal derivative, but they are now combined with extra terms that control volume compression. Moreover, Eq. (3) makes use of both material parameters μ and ν, while Eq. (4) depends only on the Poisson ratio ν, since μ gets canceled when multiplied by a or b.

3.3 Somigliana identity

The fundamental solutions of linear elasticity given by Eqs. (3) and (4) play a central role in the evaluation of the elastostatic solution over *bounded* domains. Using the so-called *Somigliana identity* [Somigliana 1885; Cruse and Suwito 1993], we can express the solution $\boldsymbol{u}(\boldsymbol{x})$ of Eq. (1) at any point $\boldsymbol{x} \in \Omega$ in integral form as

$$\boldsymbol{u}(\boldsymbol{x}) = \int_{\partial\Omega}\left[\mathcal{T}(\boldsymbol{y}, \boldsymbol{x})\boldsymbol{u}(\boldsymbol{y}) + \mathcal{K}(\boldsymbol{y}, \boldsymbol{x})\boldsymbol{\tau}(\boldsymbol{y})\right] \mathrm{d}\sigma_{\boldsymbol{y}} + \int_{\Omega}\mathcal{K}(\boldsymbol{y}, \boldsymbol{x})\boldsymbol{b}(\boldsymbol{y})\mathrm{d}\sigma_{\boldsymbol{y}}. \tag{5}$$

It is worth pointing out that the first term of Eq. (5) is a boundary integral that involves both displacements and traction vectors, while the second term is a volume integral accounting for body loads. Note also that the boundary integral in Eq. (5) depends solely on the Poisson ratio ν, while the stiffness μ is simply a global scaling that counteracts the body loads.

4 OUR CAGE DEFORMER

Equipped with the Somigliana identity, we can now delve into the formulation of our elasticity-derived cage deformer.

4.1 Cage Discretization

We discretize the domain Ω as a simplicial mesh (the "cage") forming a closed triangulated manifold mesh in 3D (or a simple polygon in 2D) with n vertices and m faces (where $m = n$ in 2D). Such a cage mesh can be created manually using typical polygonal modeling tools or automatically [Xian et al. 2012; Le and Deng 2017]. We denote the positions of cage vertices by $\{\boldsymbol{v}_i\}_{i=1..n}$ and the normals of cage faces by $\{\boldsymbol{n}_j\}_{j=1..m}$. Any point $\boldsymbol{x}$ *on* the cage boundary $\partial\Omega$ can then be expressed using a linear combination of nearby cage vertices, i.e., $\boldsymbol{x} = \sum_i \phi_i(\boldsymbol{x})\,\boldsymbol{v}_i$, where ϕ_i is piecewise-linear basis function associated with cage vertex i. Similarly, we can write the outward unit normal at $\boldsymbol{x} \in \partial\Omega$ as $\boldsymbol{n}(\boldsymbol{x}) = \sum_j \psi_j(\boldsymbol{x})\,\boldsymbol{n}_j$, where ψ_j is a piecewise-constant basis function corresponding to cage face j. Finally, we will denote by $\widetilde{\Omega}$ the deformed cage with vertices and normals indicated through $\{\widetilde{\boldsymbol{v}}_i\}$ and $\{\widetilde{\boldsymbol{n}}_j\}$, respectively.

4.2 Somigliana Coordinates

We begin our formulation by noticing that, when considering the interior of cage Ω as an elastic domain, the identity map $\boldsymbol{u}(\boldsymbol{x}) = \boldsymbol{x}$ is a trivial solution of Eq. (1) in the absence of body load, i.e., when $\boldsymbol{b}(\boldsymbol{x}) = \boldsymbol{0}$. As a result, we can employ the Somigliana identity from Eq. (5) to represent any point $\boldsymbol{x}$ *inside* the cage Ω as a linear combination of boundary points and normals, yielding:

$$\boldsymbol{x} = \int_{\partial\Omega}\left[\mathcal{T}(\boldsymbol{y}, \boldsymbol{x})\boldsymbol{y} + c\,\mathcal{K}(\boldsymbol{y}, \boldsymbol{x})\boldsymbol{n}(\boldsymbol{y})\right] \mathrm{d}\sigma_{\boldsymbol{y}}, \tag{6}$$

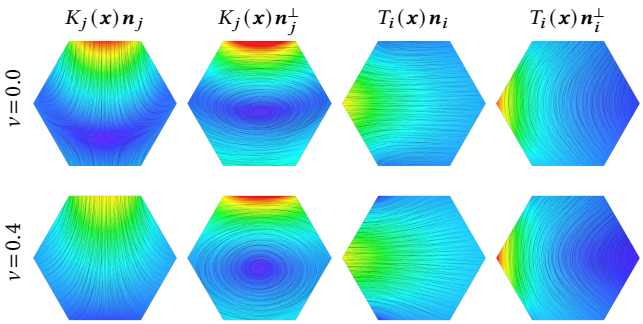


Fig. 3. **Visualizing 2D kernels K_j & T_i.** We visualize these two elastostatic kernels via the vector fields obtained by multiplying the edge-based matrix function $K_j(x)$ (resp., the vertex-based matrix function $T_i(x)$) by the normal n_j of its associated cage edge (resp., the normal n_i of its associated cage vertex) and its 90° rotation, for two different Poisson ratios. Colors indicate magnitude, while directions of the vector fields are conveyed via particle tracing, exhibiting clear differences depending on compressibility.

where the constant $c = 2\mu\,[1 + d\nu/(1-2\nu)]$ comes from the boundary traction $\tau(y)$ given by Eq. (2) for $u(x) = x$ in $\mathbb{R}^d$ ($d = 2, 3$). By expanding Eq. (6) in terms of cage vertices $\{v_i\}$ and normals $\{n_j\}$, we can rewrite the coordinates of any point x inside the cage as

$$\boxed{x = \sum_i T_i(x)\, v_i + \sum_j K_j(x)\,(c\, n_j)\,,} \tag{7}$$

with matrix coefficients per cage vertex i and face j defined through

$$\begin{cases} T_i(x) = \displaystyle\int_{\partial\Omega} \mathcal{T}(y,x)\phi_i(y)\mathrm{d}\sigma_y, & \text{(8a)} \\ K_j(x) = \displaystyle\int_{\partial\Omega} \mathcal{K}(y,x)\psi_j(y)\mathrm{d}\sigma_y. & \text{(8b)} \end{cases}$$

Since these matrix functions are derived from the Somigliana identity in Eq. (6), we name them *Somigliana coordinates*. To illustrate their built-in elastic nature, Fig. 3 visualizes in 2D the vector fields $(K_j(x)n_j, K_j(x)n_j{}^\perp)$ and $(T_i(x)n_i, T_i(x)n_i{}^\perp)$ inside a polygonal cage for different Poisson ratios. Notice that these pairs of vector fields clearly exhibit anisotropic falloffs and form smoothly-varying frames oriented towards their respective cage elements, induced by their dependence on the fundamental solution of elastostatics. This should not come as a surprise since these matrix functions are in fact used in BEM to find the internal deformation once all boundary tractions have been computed through a dense linear system (see, e.g., [James and Pai 1999, 2003]).

Moreover, by noting that any constant displacement field is a trivial solution of Eq. (1) when $b(x) = 0$, we have $\sum_i T_i(x) = I$ for any point $x \in \Omega$, thus ensuring that the vertex-based matrices $\{T_i\}$ form a partition of unity. With this property, Eq. (7) can be rewritten as

$$\sum_i T_i(x)\,(v_i - x) + \sum_j K_j(x)\,(c\, n_j) = 0. \tag{9}$$

Therefore, we can interpret the Somigliana coordinates of a rest cage Ω evaluated at any point $x \in \Omega$ as a balance between the tractions produced by the vectors $v_i - x$ and the Kelvinlet solution induced by the cage's boundary tractions $c\, n_j$.

4.3 Cage Deformer via Corotational Formulation

While the Somigliana coordinates in Eqs. (8) inherit the local elastic behavior we seek out, their tensorial nature makes them dependent on the global orientation and scaling of the undeformed cage and, consequently, they can not reproduce similarity transformations. To address this issue, we propose to factor out the rotational component of the cage deformations by adapting the usual corotational formulation of elasticity to Somigliana coordinates.

In addition to the rest and posed cage Ω and $\widetilde{\Omega}$, assume for now that a rotation matrix R_j and a symmetric strain matrix S_j are given for each cage face j — we will detail different options to compute these matrices from the deformed cage later in §4.4. From the face rotations, we can deduce a rotation matrix R_i per cage vertex i by averaging its neighboring face rotations weighted by their areas. The only condition we impose to the face rotation-strain matrices $\{R_j, S_j\}$ is that, if the posed cage is a similarity transformation of the rest cage (i.e., $\widetilde{\Omega} = sR\Omega + \mathbf{t}$ with R being a rotation, $\mathbf{t}$ a translation, and s a scalar), then $R_j = R\ \forall j$ and $S_j = sI\ \forall j$, i.e., all rotations match and all strains are the same uniform scaling. This condition will guarantee similarity invariance of our cage deformer.

We define the deformed position $\widetilde{x}$ of a rest point $x \in \Omega$ with Somigliana coordinates $\{T_i(x), K_j(x)\}$ via a corotational extension of Eq. (9):

$$\sum_i R_i T_i(x) R_i^t\,(\widetilde{v}_i - \widetilde{x}) + \sum_j R_j K_j(x) R_j^t \widetilde{\tau}_j = 0, \tag{10}$$

where $\widetilde{\tau}_j$ is the imposed traction of the deformed cage face j, leading to the following closed-form update rule:

$$\boxed{\widetilde{x} = T(x)^{-1}\left[\sum_i R_i T_i(x) R_i^t\, \widetilde{v}_i + \sum_j R_j K_j(x) R_j^t \widetilde{\tau}_j\right],} \tag{11}$$

with $T(x) = \sum_i R_i T_i(x) R_i^t$.

Furthermore, we can expand the corotated traction $\widetilde{\tau}_j$ in terms of the matrices R_j and S_j from each cage face j (see, e.g., [Sifakis and Barbic 2012]), yielding the expression:

$$\widetilde{\tau}_j = 2\mu R_j \left[S_j + \left(\frac{\nu}{1-2\nu}\right)\mathrm{tr}(S_j) I\right] n_j. \tag{12}$$

The strain S_j associated with cage face j is meant to encode a stretch along the undeformed unit normal n_j as well as in-plane stretches along the undeformed face. Since tangential stretches contribute to the corotated traction only via the trace of S_j in Eq. (12), we can represent S_j without loss of generality as

$$S_j = \eta_j n_j n_j^t + \lambda_j (I - n_j n_j^t), \tag{13}$$

where η_j is the stretch along the rest normal and λ_j is the average in-plane stretch. With this expression for S_j, Eq. (12) simplifies to

$$\widetilde{\tau}_j = \underbrace{2\mu\left[\eta_j + \frac{\nu}{1-2\nu}\left(\eta_j + (d-1)\lambda_j\right)\right]}_{s_j} R_j n_j = s_j R_j n_j, \tag{14}$$

where we factored out all the elastic material coefficients and strain stretches into a single scalar s_j for legibility. Observe that the traction vector is, up to a scaling factor, the rotated rest normal, which may be different than the normal of the posed cage depending on our choice of rotation R_j. This difference will allow a greater space of possible deformations as we will see in §4.4. Also notice that,

because of our assumption on the cage rotations and strains in the case of a similarity transformation between rest and posed cages, the expression of our elasticity-derived deformer in Eq. (11) guarantees similarity invariance, an important property to have in order to ensure an intuitive behavior of the resulting deformation.

4.4 Setting Rotations and Stretches

Now that we have derived our cage deformation, we describe next how to compute, for every cage face j, a rotation R_j as well as tangential and normal stretches (λ_j, η_j) that define a strain matrix S_j in order to complete the construction of our cage deformer. The choice of these degrees of freedom is crucial to the look and feel of the resulting cage deformer. While one could explore various deformation effects by tweaking these values, we only discuss two representative variants (one local, one global) which offer complementary benefits.

Global variant. Arguably the simplest choice for face rotations is through a registration process [Umeyama 1991] that returns the optimal similarity transformation mapping the rest cage to the posed cage in the least-squares sense. From the resulting global rotation R and global scaling s, we then set every cage face j with rotation $R_j = R$ and tangential stretch $\lambda_j = s$. This approach is inspired by the type of surface tractions that BEM (e.g., [James and Pai 1999]) generates from displacements, which do not exhibit significant change in both orientation and magnitude under small to moderate deformations. Our use of a global registration can thus be seen as a rough approximation of the BEM deformation, but without any expensive linear solve.

Local variant. In contrast to the global approach, we can create more localized deformation by setting R_j as the rotation between the rest and deformed cage face j. Similarly, a per-face tangential stretch can be derived in 3D by computing the two singular values of the linear map between the rest and posed cage face j and setting λ_j to their average — in 2D, we simply assign λ_j to the length ratio between the rest and posed edge j.

Normal stretch. Once either the local or the global variant has been selected, we still need to choose the stretch η_j along the normal direction of each cage face j. However, there is technically no information on how to set this value based simply on the shapes of the rest and posed cage unless we include application-specific requirements. For instance, Lipman et al. [2008] advocated for $\eta_j = \lambda_j$ so as to ensure (quasi-)conformal deformations. Since our goal is to resemble elastic deformations, we consider the normal stretch as a "knob" to offer bulging control for the cage deformer. More concretely, we draw inspiration from the observation that a swelling deformation of a surface patch broadens the range of its normal field, while a contraction narrows it. Therefore, we estimate the local bulging amount β_j of a cage face j by measuring the difference between two solid angles: the one spanned by the incident vertex normals on the deformed cage face and the one computed on the rest face, with vertex normals evaluated by averaging the area-weighted normals of the incident faces. We then set the normal stretch as $\eta_j = \lambda_j \exp(\gamma\beta_j/(2^{d-1}\pi))$, where γ is a global user parameter that controls the bulging scale while the denominator normalizes the solid angle difference. Note that this normal stretch maintains the similarity invariance property by construction since one gets $\beta_j = 0$ and thus $\eta_j = \lambda_j$ if rest and posed cages are the same up to a similarity transformation. Fig. 5 compares the resulting bulging effects for a range of values of γ in 2D and 3D.

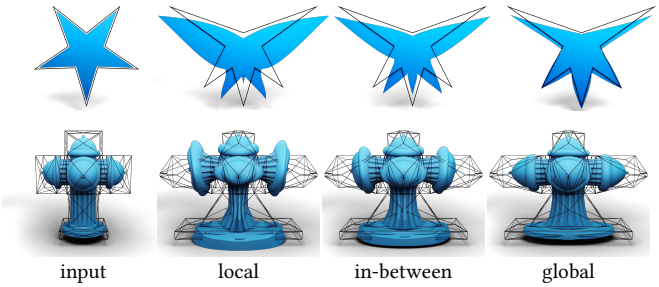
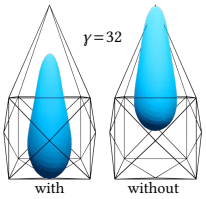


Fig. 4. **Global vs. local variant.** On a 2D Star mesh (top, with $\nu = 0.38$) and a 3D FireHydrant mesh (bottom, with $\nu = 0.3$), we compare the global vs. local choice of boundary conditions (rotations and stretches) when no bulging is requested ($\gamma = 0$). One can easily create in-between deformation by simply averaging the boundary conditions.

Finally, we found useful to add a post-optimization of the traction scaling values s_j from Eq. (14) to avoid a possible "floating" effect of the deformation in case of large cage displacements or large bulging scales (see inset). Since linear elasticity expects a zero sum of internal forces, we alter boundary tractions by finding new traction scaling values s'_j so that the sum of each corotated cage traction $\widetilde{\boldsymbol{\tau}}_j$ weighted by its respective cage face area A_j (or length in 2D) is zero. This correction, which again does not affect similarity invariance, is found via the following constrained least-square minimization:



$$\min_{\{s'_j\}} \sum_j A_j \left(s'_j - s_j\right)^2 \quad \text{s.t.} \quad \sum_j A_j s'_j R_j \boldsymbol{n}_j = \mathbf{0}, \tag{15}$$

which is efficiently performed through the Schur complement by solving a small $d \times d$ linear system.

4.5 Discussion

Our cage deformer has a number of unusual properties that deserve mentioning. First, it uses vertex and normal coordinates to generate non-interpolatory deformation, a property shared in 3D only with

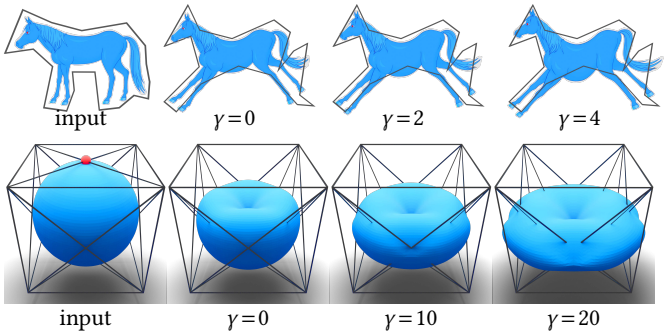


Fig. 5. **Bulging control.** By varying γ, one changes the amount of bulging created by our cage deformer (here, using the global variant and $\nu = 0$).

Green coordinates thus far [Lipman et al. 2008]. Second, our coordinates are derived from an integral of the fundamental solution of elasticity over a bounded domain, unlike the Kelvinlets of de Goes and James [2017] which disregard all boundary terms. Third, the actual deformation induced by our cage deformer involves matrix-valued coefficients, thus leading to greater expressiveness. Finally, our contribution offers interactive volume and bulging control by simply adjusting the normal stretch, which is a unique feature.

It also bears discussing our two variants for setting rotation and stretches, since they have their own benefits and flaws. In our experiments (see for instance Figs. 4 and 10), the local approach appeared more "lively" creating results reminiscent of squash and stretch deformations due to its purely local nature. However, it may cause fold-overs in the cage interior more often, and its deformation is also more sensitive to the triangulation of the cage mesh. Conversely, the global approach partially dampens the deformation, thus offering more robust and physically-plausible results under large cage displacements. Although our two variants already offer a richer gamut of deformation and control compared to previous work, other alternatives could be easily devised; e.g., Fig. 4 shows how simply linearly blending rotations and stretches provides a middle ground between our two variants. Even bulging control could be handled via different estimates based on specific applications, e.g., we could measure the "swept volume" formed by a cage face between its rest and posed positions. Finally, we point out that our formulation offers volume control of the cage deformer in two complementary ways: the Poisson ratio ν affects both the precomputation of the matrix-valued coordinates in Eqs. (8) and the corotated traction in Eq. (12) to imbue the deformer with global compressibility, while the bulging parameter γ is used only at runtime in the normal stretch η_j to control local shape swelling.

5 RESULTS

In this section, we provide further details about our implementation, show shape editing examples generated by our cage deformer, and discuss these results compared to existing cage deformers. Our source code is available at `https://gitlab.inria.fr/geomerix/public/somi-cage`.

Implementation. Given a query point $\boldsymbol{x}$ inside the rest cage Ω, we evaluate its associated Somigliana coordinates by expressing Eqs. (8) as a sum of integrals over cage triangles. Although simple closed-form formulas are available in 2D, it becomes far more difficult in 3D. We thus turn to quadrature rules to evaluate these integrals on each triangle, as detailed in Sec. A of the supplemental material. In our implementation, we simply subdivide each triangle into three quads and apply standard Gauss-Legendre quadratures on each of them (see inset). By default, we take 128 quadratures for each edge in 2D and 7500 for each triangle in 3D. In Fig. 6, we plot the partition of unity residual caused by approximating $T_i(\boldsymbol{x})$, confirming its rapid improvement as we increase quadrature count; we found in our tests that a residual below 10^{-2} results in visually artifact-free deformations. We implemented our 3D version using CUDA on a Geforce RTX 3060 GPU to benefit from massive parallelism. While

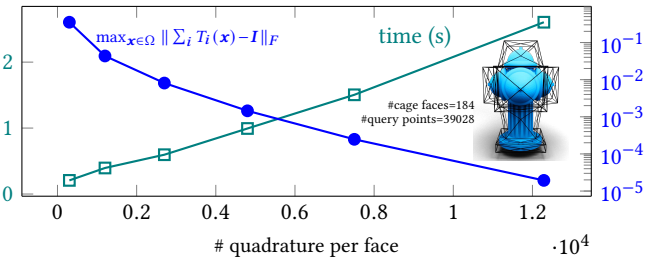


Fig. 6. **Quadrature evaluation.** On the FireHydrant mesh, for instance, the time complexity of computing Somigliana coordinates based on the rest cage is roughly linear in the number of quadrature points per cage face. Failure to satisfy partition of unity measures the numerical accuracy of our quadrature evaluation, which decreases quickly with the number of quadrature points, with no noticeable artifacts for errors below 10^{-2}.

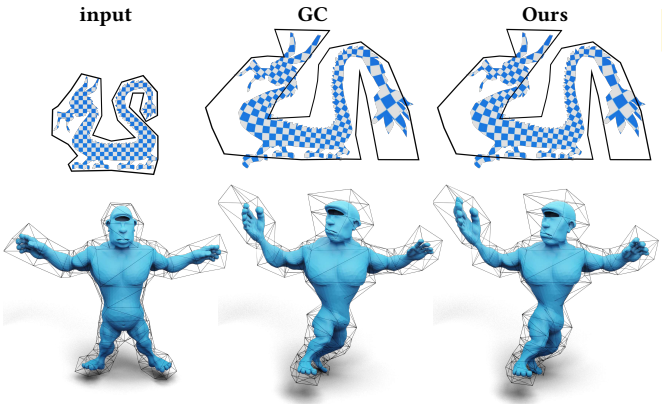


Fig. 7. **Infinite Poisson ratio.** For $\nu = \pm\infty$ and $\gamma = 0$, our cage deformer (right) exactly matches Green coordinates (middle) for a 2D cage deformation of a dragon (top), but a 3D deformation of the Ogre mesh exhibits different results. Note that our method produces reduced quasi-conformality in this example, with a right hand better matching the cage.

our use of a non-adaptive scheme is less efficient than more advanced quadratures, our precomputation timings are around 1.5*s* for the 39*k* FireHydrant mesh of Fig. 6, which is comparable to recent quadrature-based deformers such as [Thiery et al. 2018]. Once the Somigliana coordinates are built, generating our cage deformation is real-time, taking only 24*ms* for the same FireHydrant mesh for instance.

Bulging and volume control. The ease with which a model will experience volume change during a cage deformation is controlled by the Poisson ratio ν. Figs. 1 and 2 show how changing the Poisson ratio value affects deformation. Note that we prevent a recomputation of both $T_i(\boldsymbol{x})$ and $K_j(\boldsymbol{x})$ each time the user changes ν by storing the $\boldsymbol{r}\boldsymbol{r}^t$ part of Eq. (3) and the two matrix parts of Eq. (4); we then cheaply reassemble the matrix coordinates when ν changes. To control the propensity for the cage deformation to bulge out locally, one can act instead on the bulging scale γ: Fig. 5 compares the intuitive effect of this value on a 2D and a 3D cage example.

Comparisons. For the unphysical case of infinite Poisson ratio ($\nu = \pm\infty$), our 2D Somigliana coordinates match exactly the Green coordinates of [Lipman et al. 2008] and the complex Cauchy-Green coordinates of [Weber et al. 2009] when no bulging factor ($\gamma = 0$) is

used — see Fig. 7 (top) for a visual confirmation and Sec. B of the supplemental material for a proof; but this is no longer true in 3D and our cage deformer behaves quite differently from all previous works. In Fig. 11, we provide additional comparisons of 3D cage deformations generated by Mean Value, Green, or Somigliana coordinates. Notice that, for physically realistic Poisson ratios $0 \leq \nu < 1/2$, our results differ significantly from other coordinates by exhibiting compensatory bulging with a magnitude dependent on ν — for instance, see WireSphere or Botijo models. Fig. 10 shows 2D examples instead, where again the elastic behavior of our coordinates is clear in comparison to Mean Value or Green coordinates. For completeness, we also compare our method with the real-time deformer of James [2020] that tetrahedralizes the cage and computes an embedded Phong deformation. Since this approach is only C^0 continuous across tetrahedra, it suffers from non-smooth artifacts which remain visible even if a finer cage is used, as seen in Fig. 8.

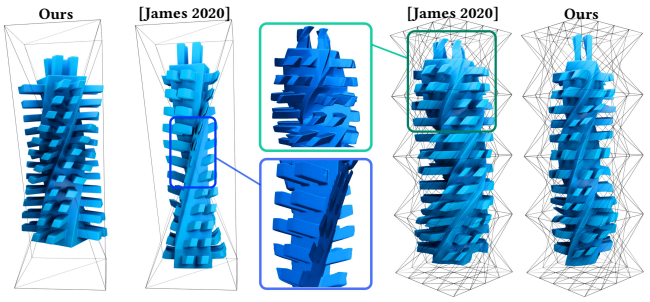


Fig. 8. C^∞ **smoothness.** While a Phong deformer [James 2020] can also provide real-time deformation of a model with a coarse (left) or fine (right) cage, its C^0 continuity across internal tetrahedra induces severe visual artifacts. Our cage deformer (here, using the local variant and $\nu=0$) does not require an internal 3D tesselation and guarantees a smooth embedding.

Limitations. At this point, it should be explicitly stated that our Somigliana coordinates are *not* generating deformation satisfying elastostatics from Eq. (1). While small to moderate displacements of the cage vertices may capture deformations very much in line with elastostatics simulation by its very foundations (see Fig. 9), Somigliana coordinates are only derived from Eq. (1), very much like Green coordinates were derived from the Laplace equation. They are, in a sense, infused with the elasticity solutions, hence their useful bulging and volume behavior during cage editing. Additionally, our approach relies on quadrature evaluations as simple closed-form

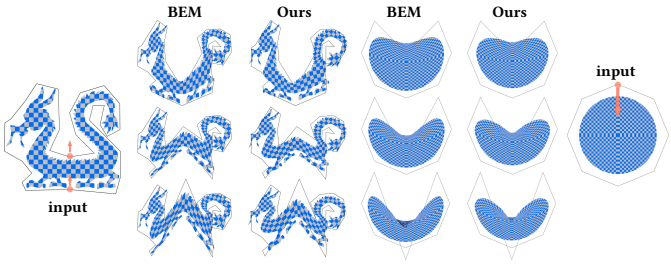


Fig. 9. **BEM vs. Somigliana.** Our global variant for setting boundary tractions (here with $\nu=0, \gamma=2$) mimics point-collocated BEM [Pozrikidis 2002] well for small deformations without requiring dense equation solves, but it does deviate from BEM solutions for larger deformations.

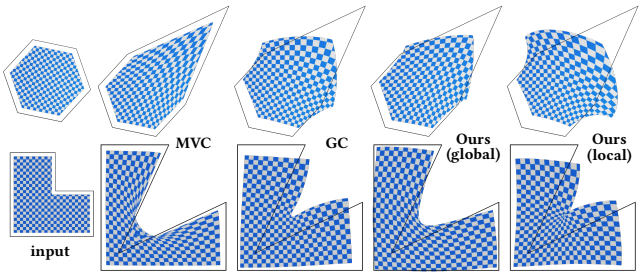
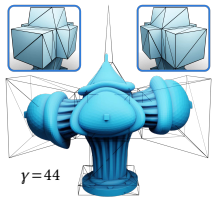


Fig. 10. **2D comparisons.** We compare Mean Value (MVC [Floater 2003]), Green (GC [Lipman et al. 2008]), and global & local Somigliana coordinates for a deformed hexagonal (top) and an L-shaped cage (bottom), for $\gamma=2$, and $\nu=0.1$. The global variant shows a more muted deformation than the local one, but both exhibit an expected elastic deformation.

expressions are currently not known in 3D, although this is hardly a limitation in practice — in fact, our use of quadratures prevents the few numerical singularities (e.g., division by zero) that the analytical formula from [Lipman et al. 2008] generates. When a large bulging factor γ is used, our cage deformer may also exhibit a lack of



symmetry induced by its dependence on the mesh connectivity of the cage (see inset), which is a common limitation for prior work too as addressed in [Thiery et al. 2018; Thiery and Boubekeur 2022]. One might explore approaches that are less sensitive to mesh connectivity, based for instance on the aforementioned "swept volume" enclosed between rest and deformed facets. Finally, our results may contain, like most existing cage deformers, fold-overs — especially for the local variant with large Poisson ratio.

6 CONCLUSION

In this paper, we introduced a novel cage deformer based on vertex and normal matrix-valued coordinates. Due to their derivation from an integral form of fundamental solutions to elastostatics, we showed that the resulting cage deformations capture the typical local bulging and volume changes expected from elasticity. As a result, our cage deformer significantly increases the space of plausible deformations that an artist can control via cage editing.

For future work, we can tackle a variety of desirable improvements or extensions. For instance, finding simple closed-form expressions for our 3D Somigliana coordinates or an adaptive strategy for quadratures based on near- or far-field evaluations are likely to improve computational efficiency. One could also extend our approach to quad or tri-quad meshes to offer more symmetric deformation in the spirit of [Thiery et al. 2018; Thiery and Boubekeur 2022]. Finally, vertex *and* normal coordinates, be they scalar or matrix-valued, remain vastly unexplored, so we hope that our work will spark further exploration, based on other linear operators for instance.

ACKNOWLEDGMENTS

JC would like to thank Jean-Marc Thiery for early discussions on integrating Eqs. (8), and acknowledges the support from the Uber chair, while MD acknowledges the generous support of Ansys, Inc., Adobe,

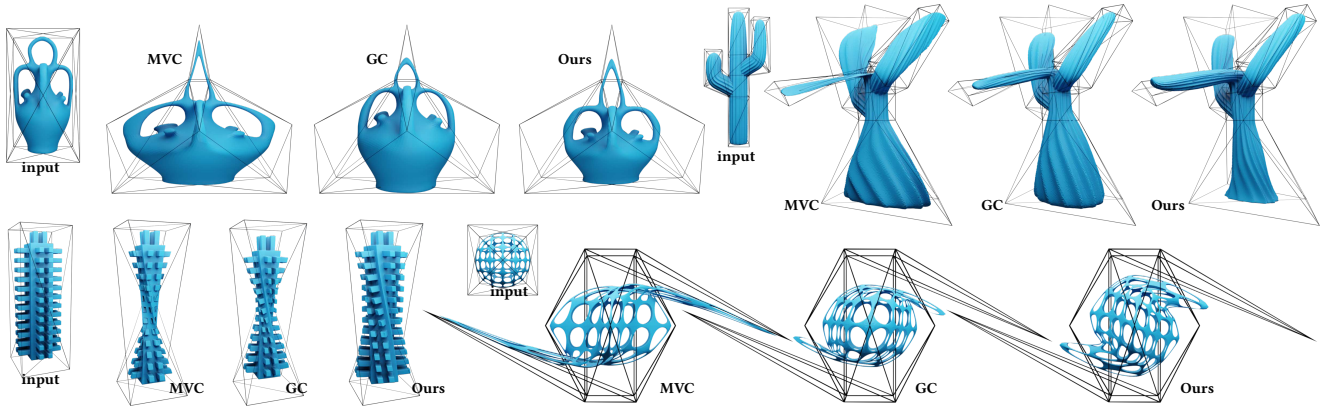


Fig. 11. **3D Comparisons.** We compare Mean Value (MVC [Ju et al. 2005]), Green (GC [Lipman et al. 2008]), and Somigliana coordinates for a variety of 3D cages using our global variant, for $\gamma = 1$ and $\nu = 0.3$. Note how our cage deformer generates elastic-looking shapes that follow the cage trend (be it bending, twisting, or creasing) quite closely even in the case of extreme deformation (e.g., WireSphere, bottom right) or very coarse cage (e.g., SpikyBox, bottom left).

and a Choose France Inria chair. The 3D models used in this paper are courtesy of Jean-Marc Thiery and Tamy Boubekeur [Thiery et al. 2018] (FireHydrant, SpikyBox, WireSphere and Cactus), Kiaran Ritchie (Ogre), and Pixologic (ManHead).

REFERENCES

Adobe. 2022. *Substance 3D Modeler.* http://www.adobe.com/products/substance3d-modeler.html

Jernej Barbič and Doug L. James. 2005. Real-Time Subspace Integration for St. Venant-Kirchhoff Deformable Models. *ACM Trans. Graph.* 24, 3 (2005), 982–990.

Jernej Barbič and Yili Zhao. 2011. Real-time Large-deformation Substructuring. *ACM Trans. Graph.* 30, 4, Article 91 (2011).

Mirela Ben-Chen, Ofir Weber, and Craig Gotsman. 2009. Variational Harmonic Maps for Space Deformation. *ACM Trans. Graph.* 28, 3, Article 34 (2009).

Blender. 2022. *A 3D modelling and rendering package.* Blender Online Community. http://www.blender.org

Max Budninskiy, Beibei Liu, Yiying Tong, and Mathieu Desbrun. 2016. Power Coordinates: A Geometric Construction of Barycentric Coordinates on Convex Polytopes. *ACM Trans. Graph.* 35, 6, Article 241 (2016).

Jiong Chen and Mathieu Desbrun. 2022. Go Green: General Regularized Green's Functions for Elasticity. In *ACM SIGGRAPH Proceedings.* Article 6.

Thommas A. Cruse and Wan Suwito. 1993. On the Somigliana Stress Identity in Elasticity. *Computational Mechanics* 11 (1993), 1–10.

Fernando de Goes and Doug L. James. 2017. Regularized Kelvinlets: Sculpting Brushes Based on Fundamental Solutions of Elasticity. *ACM Trans. Graph.* 36, 4, Article 40 (2017).

Fernando de Goes and Doug L. James. 2018. Dynamic Kelvinlets: Secondary Motions Based on Fundamental Solutions of Elastodynamics. *ACM Trans. Graph.* 37, 4, Article 81 (2018).

Fernando de Goes and Doug L. James. 2019. Sharp Kelvinlets: Elastic Deformations with Cusps and Localized Falloffs. In *Proceedings of the Digital Production Symposium.* Article 2.

Michael S. Floater. 2003. Mean Value Coordinates. *Comp. Aided Geom. Design* 20, 1 (2003), 19–27.

Michael S. Floater. 2015. Generalized barycentric coordinates and applications. *Acta Numerica* 24 (2015), 161–214.

Kai Hormann and N. Sukumar. 2008. Maximum Entropy Coordinates for Arbitrary Polytopes. In *Symp. Geo. Proc.* 1513–1520.

Kai Hormann and N. Sukumar. 2017. *Generalized barycentric coordinates in computer graphics and computational mechanics.* CRC Press.

Doug L. James. 2020. Phong deformation: a better C^0 interpolant for embedded deformation. *ACM Trans. Graph.* 39, 4 (2020), 56–1.

Doug L. James and Dinesh K. Pai. 1999. ArtDefo: Accurate Real Time Deformable Objects. In *Proceedings of the 26th Annual Conference on Computer Graphics and Interactive Techniques.* 65–72.

Doug L. James and Dinesh K. Pai. 2003. Multiresolution Green's Function Methods for Interactive Simulation of Large-Scale Elastostatic Objects. *ACM Trans. Graph.* 22, 1 (2003), 47–82.

Pushkar Joshi, Mark Meyer, Tony DeRose, Brian Green, and Tom Sanocki. 2007. Harmonic Coordinates for Character Articulation. *ACM Trans. Graph.* 26, 3, Article 71 (2007).

Tao Ju, Scott Schaefer, and Joe Warren. 2005. Mean Value Coordinates for Closed Triangular Meshes. *ACM Trans. Graph.* 24, 3 (2005), 561–566.

Lord Kelvin. 1848. Note on the Integration of the Equations of Equilibrium of an Elastic Solid. *Cambridge and Dublin Mathematical Journal* 3 (1848), 87–89.

Binh Huy Le and Zhigang Deng. 2017. Interactive Cage Generation for Mesh Deformation. In *Symposium on Interactive 3D Graphics and Games.* Article 3.

Yaron Lipman, Johannes Kopf, Daniel Cohen-Or, and David Levin. 2007. GPU-assisted Positive Mean Value Coordinates for Mesh Deformations. In *Symposium on Geometry Processing.*

Yaron Lipman, David Levin, and Daniel Cohen-Or. 2008. Green Coordinates. *ACM Trans. Graph.* 27, 3 (2008), 1–10.

Sebastian Martin, Christopher O. Huber, Peter Kaufmann, and Markus H. Gross. 2009. Shape-Preserving Animation of Deformable Objects. In *International Workshop on Vision, Modeling, and Visualization.* 65–72.

Aleka McAdams, Yongning Zhu, Andrew Selle, Mark Empey, Rasmus Tamstorf, Joseph Teran, and Eftychios Sifakis. 2011. Efficient Elasticity for Character Skinning with Contact and Collisions. *ACM Trans. Graph.* 30, 4, Article 37 (2011).

Mark Meyer, Alan Barr, Haeyoung Lee, and Mathieu Desbrun. 2002. Generalized Barycentric Coordinates on Irregular Polygons. *J. Graph. Tools* 7, 1 (2002), 13–22.

Constantine Pozrikidis. 2002. Practical Guide to Boundary Element Methods with the Software Library Bemlib.

Eftychios Sifakis and Jernej Barbic. 2012. FEM Simulation of 3D Deformable Solids. In *ACM SIGGRAPH courses.*

William S Slaughter. 2012. *The Linearized Theory of Elasticity.* Springer.

Breannan Smith, Fernando de Goes, and Theodore Kim. 2018. Stable Neo-Hookean Flesh Simulation. *ACM Trans. Graph.* 37, 2, Article 12 (2018), 15 pages.

Carlo Somigliana. 1885. Sopra l'Equilibrio di un Corpo Elastico Isotropo. *Nuovo Cimento* 3 (1885), 140–148.

Jean-Marc Thiery and Tamy Boubekeur. 2022. Green Coordinates for Triquad Cages in 3D. In *SIGGRAPH Asia 2022 Conference Papers.* Article 38.

Jean-Marc Thiery, Pooran Memari, and Tamy Boubekeur. 2018. Mean Value Coordinates for Quad Cages in 3D. *ACM Trans. Graph.* 37, 6, Article 229 (2018).

Shinyi Umeyama. 1991. Least-squares Estimation of Transformation Parameters Between Two Point Patterns. *IEEE Trans. PAMI* 13, 4 (1991), 376–380.

Joe Warren, Scott Schaefer, Anil Hirani, and Mathieu Desbrun. 2007. Barycentric Coordinates for Convex Sets. *Adv. Comput. Math.* 27, 3 (2007), 319–338.

Ofir Weber. 2017. Planar Shape Deformation. In *Generalized Barycentric Coordinates in Computer Graphics and Computational Mechanics*, Kai Hormann and N. Sukumar (Eds.). CRC Press, Chapter 7, 109–133.

Ofir Weber, Mirela Ben-Chen, and Craig Gotsman. 2009. Complex Barycentric Coordinates with Applications to Planar Shape Deformation. *Comp. Graph. Forum* 28, 2 (2009).

Chuhua Xian, Hongwei Lin, and Shuming Gao. 2012. Automatic Cage Generation by Improved OBBs for Mesh Deformation. *Vis. Comp.* 28, 1 (2012), 21–33.

Supplemental Material

Somigliana Coordinates: an elasticity-derived approach for cage deformation

JIONG CHEN, LIX, Ecole Polytechnique, IP Paris, France
FERNANDO DE GOES, Pixar Animation Studios, USA
MATHIEU DESBRUN, Inria / Ecole Polytechnique, France

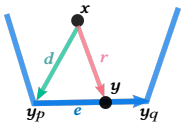
A EVALUATING SOMIGLIANA COORDINATES

As explained in our main paper [Chen et al. 2023], Somigliana coordinates for each face i and node j are defined as

$$\begin{cases} T_i(\boldsymbol{x}) = \displaystyle\int_{\partial\Omega} \mathcal{T}(\boldsymbol{y},\boldsymbol{x})\phi_i(\boldsymbol{y})\,\mathrm{d}\sigma_{\boldsymbol{y}}, & \text{(1a)} \\ K_j(\boldsymbol{x}) = \displaystyle\int_{\partial\Omega} \mathcal{K}(\boldsymbol{y},\boldsymbol{x})\psi_j(\boldsymbol{y})\,\mathrm{d}\sigma_{\boldsymbol{y}}. & \text{(1b)} \end{cases}$$

Finding efficiently-computable closed-form expressions for these integrals is challenging. Instead, we apply quadratures to numerically evaluate them in practice. Here, we provide details on integrating these integrals for the 2D and 3D cases. Given a point $\boldsymbol{x} \in \Omega$, we compute both $T_i(\boldsymbol{x})$ and $K_j(\boldsymbol{x})$ by iterating over each element of the domain boundary.

2D case. Given an edge $L_{pq} = (\boldsymbol{y}_p, \boldsymbol{y}_q)$, a point $\boldsymbol{y}$ on this edge can be expressed as $\boldsymbol{y} = \boldsymbol{y}_p + \alpha(\boldsymbol{y}_q - \boldsymbol{y}_p)$ for a barycentric coordinate $\alpha \in [0,1]$. We define $\boldsymbol{e} = \boldsymbol{y}_q - \boldsymbol{y}_p$, $\boldsymbol{d} = \boldsymbol{y}_p - \boldsymbol{x}$, and $\boldsymbol{r} = \alpha\boldsymbol{e} + \boldsymbol{d}$ (see inset), while $r = \|\boldsymbol{r}\|$ denotes the norm of $\boldsymbol{r}$. Since $\psi_j(\boldsymbol{y})$ are piecewise constant per edge, face coordinates are easily computed through quadrature as:



$$\begin{aligned} K_{L_{pq}}(\boldsymbol{x}) &= \int_{L_{pq}} \mathcal{K}(\boldsymbol{y},\boldsymbol{x})\,\mathrm{d}\sigma_{\boldsymbol{y}} = \|\boldsymbol{e}\| \int_0^1 \mathcal{K}(\boldsymbol{y}(\alpha),\boldsymbol{x})\,\mathrm{d}\alpha \\ &= \|\boldsymbol{e}\| \int_0^1 (a-b)\ln\left(\frac{1}{r(\alpha)}\right)\boldsymbol{I} + \frac{b}{r^2(\alpha)}\boldsymbol{r}(\alpha)\boldsymbol{r}^t(\alpha)\,\mathrm{d}\alpha \\ &\approx \|\boldsymbol{e}\| \sum_k w_k \left((a-b)\ln\left(\frac{1}{r(\alpha_k)}\right)\boldsymbol{I} + \frac{b}{r^2(\alpha_k)}\boldsymbol{r}(\alpha_k)\boldsymbol{r}^t(\alpha_k)\right), \end{aligned}$$

where $a = 1/\mu(2^{d-1}\pi)$ and $b = a/4(1-\nu)$, while $\{(w_k,\alpha_k)\}_k$ denotes the pairs of weights and (barycentric coordinates of) points of a Gauss-Legendre quadrature over the interval $[0,1]$.

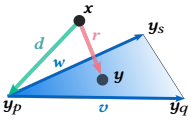
As for the vertex coordinates, integrating over the line segment L_{pq} contributes $T_p^{L_{pq}}$ (resp. $T_q^{L_{pq}}$) to $T_p(\boldsymbol{x})$ (resp. $T_q(\boldsymbol{x})$) with:

$$\begin{aligned} T_p^{L_{pq}}(\boldsymbol{x}) &= \|\boldsymbol{e}\| \int_0^1 \mathcal{T}(\boldsymbol{y}(\alpha),\boldsymbol{x})(1-\alpha)\,\mathrm{d}\alpha \\ &= \|\boldsymbol{e}\| \int_0^1 \left\{ \frac{\mu(a-2b)}{r^2(\alpha)}\left[(\boldsymbol{n}^t\boldsymbol{r}(\alpha))\boldsymbol{I} + \boldsymbol{n}\boldsymbol{r}^t(\alpha) - \boldsymbol{r}(\alpha)\boldsymbol{n}^t\right] \right. \\ &\qquad \left. + \frac{4\mu b}{r^4(\alpha)}(\boldsymbol{n}^t\boldsymbol{r}(\alpha))\boldsymbol{r}(\alpha)\boldsymbol{r}^t(\alpha)\right\}(1-\alpha)\mathrm{d}\alpha. \end{aligned}$$

Similarly, we have $T_q^{L_{pq}}(\boldsymbol{x}) = \|\boldsymbol{e}\| \int_0^1 \mathcal{T}(\boldsymbol{y}(\alpha),\boldsymbol{x})\alpha\,\mathrm{d}\alpha$, and both of them are approximated using a Gauss-Legendre quadrature. We then assemble each $T_i(\boldsymbol{x})$ from its two neighboring edges L_{pi} and L_{iq} via:

$$T_i(\boldsymbol{x}) = T_i^{L_{pi}}(\boldsymbol{x}) + T_i^{L_{iq}}(\boldsymbol{x}).$$

3D case. The 3D case proceeds similarly: given a triangular cage facet $\triangle_{pqs} = (\boldsymbol{y}_p, \boldsymbol{y}_q, \boldsymbol{y}_s)$, a point inside this triangle can be expressed using barycentric coordinates as $\boldsymbol{y} = \boldsymbol{y}_p + \alpha(\boldsymbol{y}_q - \boldsymbol{y}_p) + \beta(\boldsymbol{y}_s - \boldsymbol{y}_p)$, where $\alpha \in [0,1]$, and $\beta \in [0, 1-\alpha]$. Denote $\boldsymbol{v} = \boldsymbol{y}_q - \boldsymbol{y}_p, \boldsymbol{w} = \boldsymbol{y}_s - \boldsymbol{y}_p$, $\boldsymbol{d} = \boldsymbol{y}_p - \boldsymbol{x}$ (see inset), so that $\boldsymbol{r} = \boldsymbol{d} + \alpha\boldsymbol{v} + \beta\boldsymbol{w}$, with still $r = \|\boldsymbol{r}\|$. Then one has:



$$\begin{aligned} K_{\triangle_{pqs}}(\boldsymbol{x}) &= \int_{\triangle_{pqs}} \mathcal{K}(\boldsymbol{y},\boldsymbol{x})\mathrm{d}\sigma_{\boldsymbol{y}} \\ &= 2|\triangle_{pqs}| \int_0^1 \mathrm{d}\alpha \int_0^{1-\alpha} \mathrm{d}\beta\, \mathcal{K}(\boldsymbol{y}(\alpha,\beta),\boldsymbol{x}) \\ &= 2|\triangle_{pqs}| \int_0^1 \mathrm{d}\alpha \int_0^{1-\alpha} \mathrm{d}\beta\, \frac{a-b}{r(\alpha,\beta)}\boldsymbol{I} + \frac{b}{r^3(\alpha,\beta)}\boldsymbol{r}(\alpha,\beta)\boldsymbol{r}^t(\alpha,\beta) \\ &\approx 2|\triangle_{pqs}| \sum_k w_k \left(\frac{a-b}{r(\alpha_k,\beta_k)}\boldsymbol{I} + \frac{b}{r^3(\alpha_k,\beta_k)}\boldsymbol{r}(\alpha_k,\beta_k)\boldsymbol{r}^t(\alpha_k,\beta_k)\right), \end{aligned}$$

where $|\triangle_{pqs}|$ is the area of $\triangle_{pqs}$, while $\{w_k\}_k$ are the quadrature weights of their associated 2D quadrature points $\{(\alpha_k,\beta_k)\}_k$ on a canonical triangle in $\mathbb{R}^2$ with nodes located at $(0,0)$, $(0,1)$ and $(1,0)$. Taking into account both the cost and the generality of quadrature calculations to support high-order precision, we choose to subdivide each triangle into three quads to apply standard Gauss-Legendre quadratures — please refer to Sec. 5 of our paper for an illustration. Now, since the basis function $\phi_i(\boldsymbol{y})$ is a hat function, $T_i(\boldsymbol{x})$ is a sum of components coming from all its neighboring triangle facets, i.e., $T_i(\boldsymbol{x}) = \sum_{\triangle\in\mathcal{N}_i} T_i^{\triangle}(\boldsymbol{x})$. The contribution to $T_i(\boldsymbol{x})$ of a neighboring triangle $\triangle_{ipq}$, for instance, reads:

Authors' addresses: J. Chen, LIX, Ecole Polytechnique (IP Paris), 1 rue Honoré d'Estienne d'Orves, 91120 Palaiseau, France; F. de Goes, Pixar Animation Studios, 1200 Park Ave, Emeryville, CA 94608, USA; M. Desbrun, Inria Saclay/Ecole Polytechnique (IP Paris), 1 rue Honoré d'Estienne d'Orves, 91120 Palaiseau, France.

$$
\begin{aligned}
T_i^{\triangle ipq} &= \int_0^1 \mathrm{d}\alpha \int_0^{1-\alpha} \mathrm{d}\beta\, \mathcal{T}(\boldsymbol{y}(\alpha,\beta),\boldsymbol{x})(1-\alpha-\beta) \\
&= 2|\triangle_{ipq}| \int_0^1 \mathrm{d}\alpha \int_0^{1-\alpha} \mathrm{d}\beta \left\{ \frac{\mu(a-2b)}{r^3(\alpha,\beta)} \big[(\boldsymbol{n}^t \boldsymbol{r}(\alpha,\beta))\boldsymbol{I} + \boldsymbol{n}\boldsymbol{r}^t(\alpha,\beta) \right. \\
&\quad \left. - \boldsymbol{r}(\alpha,\beta)\boldsymbol{n}^t \big] + \frac{6\mu b}{r^5(\alpha,\beta)} (\boldsymbol{n}^t \boldsymbol{r}(\alpha,\beta))\boldsymbol{r}(\alpha,\beta)\boldsymbol{r}(\alpha,\beta)^t \right\} (1-\alpha-\beta).
\end{aligned}
$$

Similarly, we have

$$
\begin{cases}
T_j^{\triangle pjq} = \displaystyle\int_0^1 \mathrm{d}\alpha \int_0^{1-\alpha} \mathrm{d}\beta\, \mathcal{T}(\boldsymbol{y}(\alpha,\beta),\boldsymbol{x})\alpha, \\
T_k^{\triangle pqk} = \displaystyle\int_0^1 \mathrm{d}\alpha \int_0^{1-\alpha} \mathrm{d}\beta\, \mathcal{T}(\boldsymbol{y}(\alpha,\beta),\boldsymbol{x})\beta.
\end{cases}
$$

B CONDITIONAL EQUIVALENCE TO CAUCHY-GREEN COORDINATES (2D GREEN COORDINATES)

In the special 2D case when $\nu = \pm\infty$ and $\gamma = 0$, the traction terms given by Eq. (14) from our paper is zero, so we only have the deformed boundary $\widetilde{\boldsymbol{y}}$ with traction kernel $\mathcal{T}_i(\boldsymbol{y},\boldsymbol{x})$ that contributes to the actual deformation $\widetilde{\boldsymbol{x}}$. Since $\lim_{\nu\to\pm\infty} b = 0$, traction kernel becomes $\mathcal{T}(\boldsymbol{r}) = \frac{1}{2\pi r^2}[(\boldsymbol{n}^t\boldsymbol{r})\boldsymbol{I} + \boldsymbol{n}\boldsymbol{r}^t - \boldsymbol{r}\boldsymbol{n}^t]$. As a result, the deformation computed by summing over all edges reduces to

$$
\begin{aligned}
\widetilde{\boldsymbol{x}}(\boldsymbol{x}) &= \frac{1}{2\pi} \sum_e \int_0^{L_e} \left(\frac{\boldsymbol{r}^t\boldsymbol{n}}{r^2}\boldsymbol{I} + \frac{1}{r^2}(\boldsymbol{n}\boldsymbol{r}^t - \boldsymbol{r}\boldsymbol{n}^t) \right) \widetilde{\boldsymbol{y}}\, \mathrm{d}\sigma_{\boldsymbol{y}} \\
&= \frac{1}{2\pi} \sum_e \int_0^{L_e} \frac{1}{r^2} \left\{ \begin{pmatrix} r_1n_1+r_2n_2 & 0 \\ 0 & r_1n_1+r_2n_2 \end{pmatrix} \right. \\
&\qquad \left. + \begin{pmatrix} 0 & r_2n_1-r_1n_2 \\ r_1n_2-r_2n_1 & 0 \end{pmatrix} \right\} \widetilde{\boldsymbol{y}}\, \mathrm{d}\sigma_{\boldsymbol{y}}.
\end{aligned} \tag{2}
$$

For each edge, we can rewrite the above integral using complex numbers denoted with 𝕙𝕠𝕝𝕝𝕠𝕨 letters, yielding

$$
\begin{aligned}
&\frac{1}{2\pi} \int_0^{L_e} \frac{r_1n_1 + r_2n_2 + \mathbb{i}(r_1n_2 - r_2n_1)}{r^* r} \widetilde{\mathbb{y}}\, \mathrm{d}\sigma_{\mathbb{y}} \\
=&\frac{1}{2\pi} \int_0^{L_e} \frac{(r_1 - \mathbb{i}r_2)(n_1 + \mathbb{i}n_2)}{r^* r} \widetilde{\mathbb{y}}\, \mathrm{d}\sigma_{\mathbb{y}} \\
=&\frac{1}{2\pi} \int_0^{L_e} \frac{r^* n}{r^* r} \widetilde{\mathbb{y}}\, \mathrm{d}\sigma_{\mathbb{y}} = \frac{1}{2\pi\mathbb{i}} \int_0^{L_e} \frac{\widetilde{\mathbb{y}}}{r} \mathbb{i}\cdot n\, \mathrm{d}\sigma_{\mathbb{y}} \\
=&\frac{1}{2\pi\mathbb{i}} \int_L \frac{\widetilde{\mathbb{y}}}{r}\, \mathrm{d}\mathbb{y},
\end{aligned} \tag{3}
$$

where r^* denotes the conjugate of r. This proves that when $\nu = \pm\infty$ and $\gamma = 0$, our 2D coordinates exactly reproduce the Cauchy-Green coordinates [Weber et al. 2009] derived from Cauchy's integral.

REFERENCES

Jiong Chen, Fernando de Goes, and Mathieu Desbrun. 2023. Somigliana Coordinates: an elasticity-derived approach for cage deformation. In *Special Interest Group on Computer Graphics and Interactive Techniques Conference Conference Proceedings (SIGGRAPH '23 Conference Proceedings)*. 8 pages. https://doi.org/10.1145/3588432.3591519

Ofir Weber, Mirela Ben-Chen, and Craig Gotsman. 2009. Complex Barycentric Coordinates with Applications to Planar Shape Deformation. *Comp. Graph. Forum* 28, 2 (2009).