











A Survey on Cage-based Deformation of 3D Models

D. Ströter¹, J. M. Thiery², K. Hormann³, J. Chen⁴, Q. Chang³, S. Besler^{1,5},
J. S. Mueller-Roemer^{1,5}, T. Boubekur², A. Stork^{1,5} and D. W. Fellner^{1,5,6}

¹Technical University of Darmstadt, Germany ²Adobe Research, France ³Università della Svizzera italiana, Lugano, Switzerland
⁴Inria, France ⁵Fraunhofer IGD, Germany ⁶Graz University of Technology, Institute of Computer Graphics and Knowledge Visualization, Austria

Abstract

Interactive deformation via control handles is essential in computer graphics for the modeling of 3D geometry. Deformation control structures include lattices for free-form deformation and skeletons for character articulation, but this report focuses on cage-based deformation. Cages for deformation control are coarse polygonal meshes that encase the to-be-deformed geometry, enabling high-resolution deformation. Cage-based deformation enables users to quickly manipulate 3D geometry by deforming the cage. Due to their utility, cage-based deformation techniques increasingly appear in many geometry modeling applications. For this reason, the computer graphics community has invested a great deal of effort in the past decade and beyond into improving automatic cage generation and cage-based deformation. Recent advances have significantly extended the practical capabilities of cage-based deformation methods. As a result, there is a large body of research on cage-based deformation. In this report, we provide a comprehensive overview of the current state of the art in cage-based deformation of 3D geometry. We discuss current methods in terms of deformation quality, practicality, and precomputation demands. In addition, we highlight potential future research directions that overcome current issues and extend the set of practical applications. In conjunction with this survey, we publish an [application](#) to unify the most relevant deformation methods. Our report is intended for computer graphics researchers, developers of interactive geometry modeling applications, and 3D modeling and character animation artists.

CCS Concepts

• *Computing methodologies* → *Shape modeling*; • *Mathematics of computing* → *Interpolation*;

1. Introduction

Interactive deformation of geometry is a fundamental computer graphics task. For user interaction, a set of control handles corresponding to points in space is used. Deformation applications visualize these handles for user interaction via mouse. First, the user selects certain handles. Second, the user relocates them by moving the mouse. During relocation, each handle locally deforms a part of the geometry in the relocation direction. When the user performed the intended deformation, they release the control handle. Finally, the application visualizes the deformed geometry.

For intuitive deformation control, applications organize their handles in a sophisticated control structure, because point-based deformation control provides no clear indication of how the influence on the geometry is shared by different control handles. In a pioneering work, SEDERBERG and PARRY [SP86] arrange control points in a control lattice to allow for free-form deformation of geometrical objects included by the lattice. Since then, the computer graphics community devised various sophisticated control structures for deformation control (see Section 3). Skeletal control structures arrange control handles in a graph so that the influence of control points is interpolated along edges [MLT88]. This structure is well-suited for character articulation.

For more detailed deformation control, a cage control structure organizes control handles as a manifold polygonal mesh encasing the geometry. As the polygonal mesh can adapt to surface details, the advantage of cage-based deformation is its ability to intuitively express high-resolution deformation relocating a part of the cage at once. For this reason, cage-based deformation is a versatile approach. Today, cage-based deformation is used in various fields including character animation [JZvdP*08; CMT*12; KSKL14], image deformation [MSW*09], mesh modeling [TMB18; TB22], virtual prototyping [AAN12; DJS16], 3D motion capture [SF11; TTB12], and recently even virtual environments [SZG*20; LLH22] and neural networks [YAK*20; PYL*22; XH22].

Due to the ongoing challenges, many research papers [ZDL*14; XLX15; WJBK15; TMB18; TB22; CDH23; CDD23] have addressed cage-based deformation in the past decade. More than a decade has passed since the last survey on cage-based deformation [NS12]. Thus, the last survey no longer adequately reflects the state of the art. We intend to use the comprehensive state-of-the-art report (STAR) format to present the advances in 3D cage-based deformation in detail. As 3D deformation is one of the most challenging but ground-breaking research directions, our report focuses on 3D rather than 2D. We evaluate the zoo of cage-based deformation methods to discuss their advantages and disadvantages.

1.1. Outline

We first provide the necessary preliminaries and notation used throughout this report, then guide the reader through the process of cage-based deformation while highlighting advantages and disadvantages of individual methods, and finally present key conclusions and potential future avenues for research:

- Section 2 presents foundational concepts of and notation for cage-based deformation.
- Section 3 reviews related interactive deformation methods and highlights their differences to cage-based deformation.
- Section 4 describes, categorizes, and systematically discusses the methods to construct a cage for deformation control.
- Section 5 details the construction of coordinates with closed-form expressions to enable deformation control via cage vertices.
- Section 6 presents the computation of coordinates by energy minimization for more local deformation control.
- Section 7 covers the use of probabilistic models for the computation of coordinates for cage-based deformation.
- Section 8 presents coordinates not only based on cage vertices but also on normals for better feature and volume preservation.
- Section 9 discusses the coordinates for cage-based deformation in a systematic way.
- Section 10 presents methods that enlarge the facilities and the scope of applications of cage-based deformation.
- Section 11 draws the final key conclusions and suggests future research directions.

2. Preliminaries & Notation

This STAR relies on a common notation for the presented deformation methods. A cage \mathcal{C} is a manifold surface mesh consisting of vertices $\mathbf{c}_i \in \mathbb{R}^3, i = 1, \dots, N_C$ and polygons $f_i, i = 1, \dots, N_F$. The matrix $\mathbf{C} = (\mathbf{c}_1, \dots, \mathbf{c}_{N_C})^T \in \mathbb{R}^{N_C \times 3}$ includes all the vertices of \mathcal{C} . The polygonal faces $f_i \in \mathbb{N} \times \dots \times \mathbb{N}$ include integers of cage vertices into \mathbf{C} , form the boundary of the cage $\partial\mathcal{C} \ni f_i$ and define the interior space of the cage $\text{Int}\mathcal{C}$, where $\partial\mathcal{C} \cap \text{Int}\mathcal{C} = \emptyset$. If f is a triangle or quadrilateral (quad), we denote it as t or q , respectively. The to-be-deformed geometry \mathcal{T} consists of geometrical primitives \mathcal{P} and vertices $\mathbf{v}_i \in \mathbb{R}^3, i = 1, \dots, N_V$. The matrix $\mathbf{V} = (\mathbf{v}_1, \dots, \mathbf{v}_{N_V})^T \in \mathbb{R}^{N_V \times 3}$ contains all the vertices of \mathcal{T} . The prime symbol indicates deformed vertices \mathbf{v}' . Deformations are limited to a certain domain Ω . Typically, it holds that $\Omega = \mathcal{C}$ but some deformation methods allow for extrapolation from the cage. A point in the domain is denoted as $\mathbf{x} \in \Omega$.

In general, the cage-based deformation workflow follows three steps (see Fig. 1). First, the cage \mathcal{C} for the geometry \mathcal{T} must be constructed, which can be a laborious task. To bind the cage \mathcal{C} to the geometry \mathcal{T} for deformation control, cage-based deformation methods commonly use generalized barycentric coordinates. Altering the positions of the cage vertices \mathbf{c} relocates the vertices \mathbf{v} of the geometry \mathcal{T} , while its connectivity remains unchanged.

2.1. Generalized Barycentric Coordinates

Since MÖBIUS [Möb27] first formulated barycentric coordinates, the concepts of barycentric coordinates and interpolation have

been widely used and extended. Applications of such generalized barycentric coordinates (GBC) are manifold including color interpolation [MLBD02], Gouraud and Phong shading [HF06], rendering of quadrilaterals [HT04], texture mapping [DMA02], texture synthesis [TSNI10], image warping [HF06; WSHD07; SHF13; MT23], image composition [FHL*09], mesh parameterization [Flo97], shape deformation [JMD*07; JSW05; LS08; LKCL07; WG10], deformation transfer [BWG09], gradient mesh simplification [LJH13], generalized Bézier surfaces [LS07; LD89], surface design [SV18], and finite element applications [AO06; MP07; SM06; TS08; WBG07]. While this STAR focuses on the essentials for cage-based deformation, the reader is referred to the survey paper by FLOATER [Flo15] and the book by HORMANN and SUKUMAR [HS17] for an in-depth overview on GBC.

Early work on GBC focused on the 2D setting and many constructions have been proposed over the last 50 years. *Wachspress coordinates* [Wac75] are rational functions and have a simple closed form [MLBD02], but they are not well-defined for arbitrary simple polygons. The same holds for *discrete harmonic coordinates* [PP93; EDD*95], which arise from the standard piecewise linear finite element approximation of the Laplace equation. FLOATER [Flo03] uses the circumferential mean value property of harmonic functions to derive *mean value coordinates*, which also have a simple closed form and are well-defined for any concave polygon [HF06]. They are positive inside the kernel of star-shaped polygons, satisfy the Lagrange property, and are smooth everywhere in the plane, except at the vertices of the polygon, where they are only C^0 continuous. Other closed-form GBC that are well-defined for concave polygons are *metric* [MLD05; SM06], *moving least squares* [MS10], *Poisson* [LH13], *cubic mean value* [LJH13], and *Gordon–Wixom* [GW74; Bel06]. There even exists a whole family of coordinates that are well-defined for degenerate polygons [YS19], but all these constructions can be negative inside the domain. Positive coordinates for arbitrary concave polygons include *positive mean value* [LKCL07], *positive Gordon–Wixom coordinates* [MLS11], and *power coordinates* [BLTD16], but they are not smooth. *Blended barycentric coordinates* [APH17] overcome this limitation, but they depend on an initial triangulation of the domain. Coordinates that are at least C^1 and non-negative for arbitrary polygons include *harmonic* [JMD*07], *maximum entropy* [HS08], *maximum likelihood* [CDH23], *positive and smooth Gordon–Wixom* [WLMD19], *iterative* [DCH20], and *local barycentric coordinates* [ZDL*14; TDZ19], but they do not have a closed form and must be approximated by some numerical procedure.

Some of the 2D constructions can be extended to the 3D setting, which is required for cage-based deformation, and we review these constructions in detail in Sections 5 to 8. In the 3D setting, GBC provide a set of functions $\lambda_i: \mathbb{R}^3 \rightarrow \mathbb{R}, i = 1, \dots, N_C$ to interpolate the interior of \mathcal{C} . With the use of these functions, one can express any point inside \mathcal{C} as an affine sum of cage vertices:

$$\sum_{i=1}^{N_C} \lambda_i(\mathbf{v}) \mathbf{c}_i = \mathbf{v}, \quad \text{where} \quad \sum_{i=1}^{N_C} \lambda_i(\mathbf{v}) = 1.$$

Cage-based deformation capitalizes on these interpolation functions, in order to determine the vertex positions of \mathcal{T} , whenever the user deforms \mathcal{C} . Thus, calculating the coefficients $\lambda_i(\mathbf{v}_j)$,

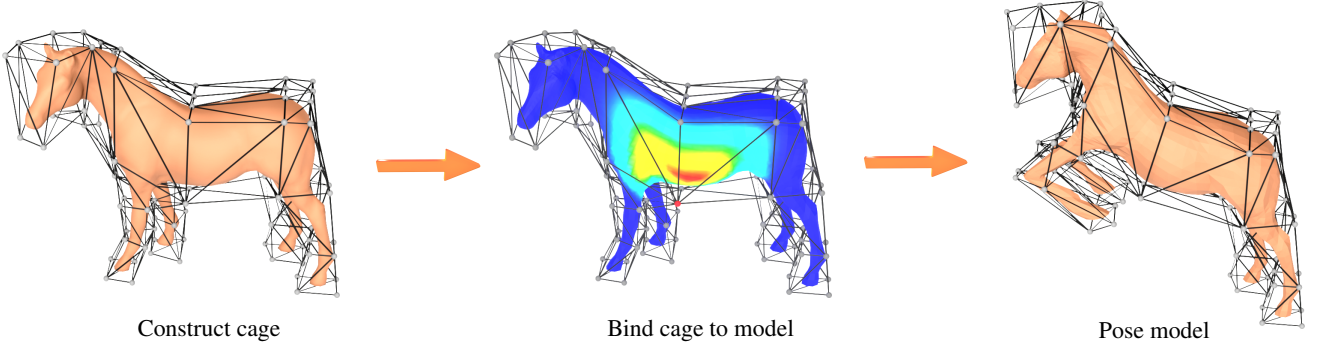


Figure 1: Cage-based deformation workflow: First, designers construct a cage. Calculating coordinates for the vertices of the model (see influence of the red vertex) binds the cage to the model. As a result, each cage vertex influences a nearby part of the model and users can intuitively pose the model.

$i = 1, \dots, N_C$ for every vertex \mathbf{v}_j , $j = 1, \dots, N_V$ of \mathcal{T} is an essential pre-processing step of cage-based deformation. This stage is frequently denoted as bind time, because it binds the to-be-deformed model to \mathcal{C} . After bind time, the vertex positions of \mathcal{T} can be adjusted to the deformed cage:

$$\sum_{i=1}^{N_C} \lambda_i(\mathbf{v}_j) \mathbf{c}'_i = \mathbf{v}'_j. \quad (1)$$

For brevity, λ_{ij} denotes the coefficient $\lambda_i(\mathbf{v}_j)$. Several properties govern the resulting deformation quality. Non-negativity is an important property, because it guarantees that the deformed vertices are located inside \mathcal{C}' . In addition, the coefficients λ_{ij} should vary smoothly, in order to provide shape preservation. On the boundary of \mathcal{C} , the functions λ_i should be linear over the polygons f . In summary, high-quality cage-based deformation control is achieved, if the functions λ_i satisfy the following properties:

- **Reproduction:** $\sum_{i=1}^{N_C} \lambda_i(\mathbf{x}) \mathbf{c}_i = \mathbf{x}, \forall \mathbf{x} \in \Omega$
- **Partition of unity:** $\sum_{i=1}^{N_C} \lambda_i(\mathbf{x}) = 1, \forall \mathbf{x} \in \Omega$
- **Non-negativity:** $\lambda_i(\mathbf{x}) \geq 0, \forall \mathbf{x} \in \Omega$
- **Smoothness:** $\lambda_i \in C^\infty, \forall i = 1, \dots, N_C$
- **Linearity on $\partial\mathcal{C}$:** λ_i is linear on each cage polygon f
- **Lagrange property:** $\lambda_i(\mathbf{c}_j) = \delta_{ij}$, where δ_{ij} is the Kronecker delta
- **Locality:** λ_i only influences regions nearby \mathbf{c}_i , i.e., $\lambda_i(\mathbf{x})$ vanishes if \mathbf{x} is far away from \mathbf{c}_i

After a set of GBC λ_{ij} is determined, the user can pose the model by relocating the cage vertices, i.e., control handles. Let us suppose that $\mathbf{W}_{ij} = \lambda_{ij}$ so that $\mathbf{W} \in \mathbb{R}^{N_C \times N_V}$. Then, posing the model follows a simple deformation scheme:

$$\mathbf{V}' = \mathbf{W}^T \mathbf{C}'. \quad (2)$$

Most deformation approaches using cages only rely on Eq. (2) to pose a model in accordance to \mathcal{C} . However, if cages are combined with other control structures, other update schemes such as linear blend skinning are used.

2.2. Linear Blend Skinning

Linear blend skinning (LBS) is typically used for deformation using skeletal control structures. In LBS, several transformations are applied at once as a “blend” to a point in space. Let us consider the coefficients λ_{ij} as interpolating weights. A set of N transformations is applied to a vertex \mathbf{v}_j as follows:

$$\mathbf{v}'_j = \sum_{i=1}^N \lambda_{ij} (\mathbf{L}_i \mathbf{v}_j + \mathbf{t}_i) = \sum_{i=1}^N \lambda_{ij} \mathbf{T}_i \begin{pmatrix} \mathbf{v}_j \\ 1 \end{pmatrix}, \quad (3)$$

where \mathbf{L}_i and \mathbf{t}_i are the linear and translation parts of the transformation \mathbf{T}_i , respectively.

As JACOBSON [Jac14] pointed out, cage-based deformation using GBC is a special case of LBS, if transformations applied to cage vertices are restricted to translations. Through re-arranging, Eq. (1) for adjusting the model vertices to the deformed cage matches Eq. (3):

$$\begin{aligned} \mathbf{v}'_j &= \sum_{i=1}^{N_C} \lambda_{ij} \mathbf{c}'_i = \sum_{i=1}^{N_C} \lambda_{ij} \mathbf{c}'_i + \lambda_{ij} \mathbf{c}_i - \lambda_{ij} \mathbf{c}_i = \sum_{i=1}^{N_C} \mathbf{v}_j + \lambda_{ij} (\mathbf{c}'_i - \mathbf{c}_i) \\ &= \sum_{i=1}^{N_C} \lambda_{ij} (\mathbf{I} \mathbf{v}_j + \mathbf{t}_i). \end{aligned}$$

With the use of LBS, a convenient update scheme for cage-based deformation can be obtained. When the coordinates λ_{ij} are determined after bind time, the LBS matrix $\mathbf{M} \in \mathbb{R}^{N_V \times 4N_C}$ can be assembled:

$$\mathbf{M} = \begin{bmatrix} \lambda_{11}(\mathbf{v}_1^T, 1) & \dots & \lambda_{N_C 1}(\mathbf{v}_1^T, 1) \\ \vdots & \ddots & \vdots \\ \lambda_{1 N_V}(\mathbf{v}_{N_V}^T, 1) & \dots & \lambda_{N_C N_V}(\mathbf{v}_{N_V}^T, 1) \end{bmatrix}.$$

Whenever the user deforms \mathcal{C} at pose time, it suffices to assemble a transformation matrix $\mathbf{T} = ((\mathbf{I}, \mathbf{t}_1)^T, \dots, (\mathbf{I}, \mathbf{t}_{N_C})^T)^T \in \mathbb{R}^{4N_C \times 3}$. The model deformation using LBS can then be calculated as a simple matrix product:

$$\mathbf{V}' = \mathbf{M} \mathbf{T}.$$

3. Related Deformation Methods

Besides cage-based deformation, other deformation methods using control points to manipulate geometry have been presented.

3.1. Free-form Deformation using Lattices

One such method is free-form deformation. The deformation space Ω is defined by a volumetric control mesh, the so-called lattice, which allows the user manipulation by means of relocating its control points. Like in cage-based deformation, the vertices of \mathcal{T} are expressed as an affine sum of the control points. The weights of the sum are defined by the location of the vertex in the parameter space of the lattice geometry. Due to the expression of the vertices in the parameter space, the computation of the updated vertices is equal to the evaluation of the lattice geometry at predefined parameter values. However, mapping a point in Euclidean space to the parameter space requires a known mapping between the spaces [SP86; MJ96].

Various geometry representations have been shown to be suitable as lattices, such as tensor product trivariate Bernstein polynomials by SEDERBERG and PARRY [SP86], trivariate B-Splines by GRIESSMAIR and PURGATHOFER [GP89], and Catmull–Clark volumes by MACCRACKEN and JOY [MJ96]. Depending on the geometry representation, algebraic, numerical, or approximating methods may be best suited [SP86; MJ96] to find the parameter values of each vertex. As the deformation is based on the re-evaluation of the lattice, the mathematical properties of the deformation, such as continuity and locality, are defined by the basis functions of the chosen geometry representation. Furthermore, the deformation space may only include parts of the model. This allows for the definition of multiple lattices on a model to achieve local deformations using varying degrees of control [SP86].

Similar to cage-based methods, free-form deformation techniques use a mesh for deformation control, which contains the to-be-deformed geometry. However, the lattice geometry is inherently volumetric and typically imposes additional restrictions on mesh connectivity, which complicates the construction of a viable lattice for deformation control. Cages are more simple to handle, as they are less restrictive.

3.2. Skeletal-based Deformation

While cages are good control structures for detailed shape manipulation, skeletal-based deformation introduced by MAGNENAT-THALMANN et al. [MLT88] excel at expressing character motion. The user either generates the skeleton manually or automatically [LW07; VF09]. In an intuitive way, the user configures the joints of the skeleton to specify the character's pose [CHP89]. Whenever the joint configuration changes, the surface mesh, i.e., the character's skin, is deformed accordingly. The coupled use of skeletons and cages allows for motion expression with the skeleton, while details can be modeled with the cage [CTL*20].

Deformation of the skin can be efficiently implemented on a GPU for real-time interaction with the application of LBS (see Section 2.2), where transformations are associated with each joint. Although LBS is frequently used due to its simplicity and efficiency, it can produce artifacts, e.g., candy wrapper artifacts, which can

be prevented by using an advanced skinning scheme such as dual quaternions [KCŽ008]. Another method to avoid these artifacts is the animation space by MERRY et al. [MMG06], which is a larger family of linear deformation methods with many benefits such as improved fitting to example poses and advanced distance computations. As the computation of high-quality skinning weights can be expensive, WANG and SOLOMON [WS21] recently presented quasi-harmonic weights for near real-time computation.

3.3. Linear Subspaces

As the generation and re-meshing of cages for deformation control is tedious, WANG et al. [WJBK15] efficiently calculate linear subspaces in Ω allowing users to interactively define point and region handles without using cages. A point handle is represented by a point in space, whereas a region handle is a manipulator object with control vertices to deform a selected subdomain of Ω , where the undeformed parts of Ω are blended smoothly [BK04]. Deformation by handle control uses a set of weights that allow for deformation using LBS (see Section 2.2). The weights are computed numerically by discretizing Ω with an embedding mesh \mathcal{E} .

For smooth deformation, computation of the weights minimizes a squared Laplacian energy. As the fairness component of this energy contains $\mathbf{1}$ in its null space, minimization can be implemented efficiently by solving a sparse linear system with a right-hand side for each handle. Due to the efficiency of weight calculation, users are able to add or remove handles at interactive rates and deform the model applying handle translations and rotations, albeit at the cost of negative weights and the need to generate \mathcal{E} .

3.4. Radial-Basis-Function-based Deformation

In contrast to the mesh-based definition of the deformation space, the deformation space may be manipulated by arbitrary, unconnected control points. Each control point is assigned a radial basis function, which defines the influence of the control point by means of its distance to the vertices. The influence reduces with increasing distance from the control point. This approach allows for accurate control over the deformation locality and poses no restriction on the placement of the control points. The choice of the specific radial basis function (RBF) determines the deformation behavior when manipulating a control point.

The specific coefficients of each RBF may be found automatically by solving a linear system [dBvdSB07; PGWB21]. The deformation of the model is computed by the interpolation defined by the previously specified RBFs. The interpolation computes multiple scalar fields, containing the coordinates of the deformed mesh. Each scalar field interpolates a component of the deformed mesh [dBvdSB07].

Due to the universal nature of the RBF interpolation, the same deformation may be applied to multiple meshes and may extrapolate or interpolate [PGWB21]. However, RBF-based techniques only offer point handles for deformation control. Contrary to offering only point handles, cage-based deformation enables users to define the local influence of handles based on the topology of the cage.

4. Cages

The term cage generally refers to a polygonal mesh enclosing input primitives. While enclosing meshes are needed for many different applications [MCA15], we focus on cages for deformation control. The creation of a cage is a significant part of the cage-based deformation workflow.

4.1. Cage Quality

Cages for interactive modeling should exhibit certain properties for convenient deformation control [Jac14]. Users wish to deform the model with only few control vertex relocations. Thus, the cage should include reasonably few control vertices for the user to manage. At the same time, the cage should wrap the model tightly with control vertices near the semantic parts of the model. A self-intersecting cage can lead to erroneous deformation results and should be avoided [SVJ15]. In addition, many coordinate types do not work if the cage intersects the model. In summary, the properties of a high-quality cage are:

- Reasonably low number of control vertices
- No self-intersections
- No intersections with the model (conservative)
- Wrap the model tightly
- Control vertices should be close to the to-be-deformed parts of the model
- The cage should provide symmetric structures, where the model exhibits symmetric features

As these properties only loosely define the requirements for a high-quality cage, the quality of the cage needs to be evaluated in light of each use case. In order to allow for the comparison of cages, XIAN et al. [XLX15] present a quality metric for cages:

$$E_C = \left(1 - \frac{N_C}{N_V}\right) S(\mathcal{C}, \mathcal{T}) e^{1 - (\text{vol}(\mathcal{C}) / \text{vol}(\mathcal{T}))},$$

where $\text{vol}(\cdot)$ represents the volume of a mesh and $S(\cdot, \cdot) \in [0, 1]$ represents a shape similarity measure, such as the similarity measure by ELAD et al. [ETA02], which enables user-guided evaluation of shape similarity.

4.2. Cage Generation

Especially for novice users, the design of a high-quality cage is laborious and time-consuming. Even for experienced users, cage generation can take several hours [LD17]. Therefore, the literature includes several approaches for automatic and semi-automatic cage generation. Previous work presented methods for fast generation of cages. LAUBE and UMLAUF [LU16] present a survey on the early automatic cage generation methods that are briefly addressed in the following.

4.2.1. Offset Surface Simplification Methods

One of the early efforts to automatically generate an enclosing cage is the progressive hull generation by SANDER et al. [SGG*00]. Building upon the progressive meshes by HOPPE [Hop96], the hull is generated by collapsing edges of the input mesh such that the

coarser mesh includes every vertex of the input mesh. After a sequence of edge collapses, a coarse cage is obtained from the input mesh.

SHEN et al. [SOS04] create an implicit surface enclosing an input polygon soup using a constrained least-squares formulation. They detail an iso-surface extraction method to ensure that all input points are enclosed by the extracted iso-surface.

For deformation transfer, BEN-CHEN et al. [BWG09] construct an offset surface for an input model with repeated simplification until the number of cage faces is lower than a user-defined threshold. As the offset surface is formed by relocating vertices along the normals of the previous offset surface, some models require heterogeneous step sizes for computing offset positions to avoid artifacts.

To avoid self-intersections, DENG et al. [DLM11] decimate \mathcal{T} , prioritizing edge collapses with an error metric, and perform a clean-up step. They incorporate a fidelity function for shape preservation and a function penalizing opposing face normals for triangle quality. Throughout decimation, they position vertices along surface normals to ensure an enveloping cage. Self-intersections are detected and removed by re-meshing the intersecting parts.

SACHT et al. [SVJ15] generate nested cages in layers so that high-resolution cages tightly bound the model and coarser cages wrap the finer cages. Each cage layer is the result of the previous finer layer after passes of decimation, flow pushing the cage inside the previous layer and re-inflation until obtaining the bounding condition. The cage layers offer a nested collection of viable cages for modeling.

4.2.2. Voxelization-based Methods

XIAN et al. [XLG09] voxelize the oriented bounding box (OBB) of the model to obtain a cage. The use of principal component analysis (PCA) quickly provides a tight OBB for the model [DKKR06]. After voxelization of the OBB, they calculate the max-norm distances [VKK*03] of mesh triangles to voxel centers to determine the feature voxels intersecting the mesh surface. Extracting and triangulating the outer feature voxel faces yields a cage bounding the model. A final smoothing step relocates the cage vertices towards the model and achieves a smoothed tightly bounding cage.

To construct a coarse enclosing grid of highly detailed linear elastic deformable objects, NESME et al. [NKJF09] organize an initial fine hexahedral embedding into an octree structure, where each hexahedron is associated with the local material properties of the model. The material properties of a parent voxel can be deduced from its children [NF*06]. As a result, a coarse bounding cage enables the quick retrieval of deformation properties at any desired level of the octree hierarchy.

As coarse cages are convenient for deformation control, XIAN et al. [XLG11] automatically generate cages by optimizing an OBB tree with a slicing rule and subsequent mesh improvement. First, a voxelization of the model's OBB is decomposed into the voxel types feature, inside and outside. The mesh vertices and barycenters of inner voxels form a point set, whose PCA provides the root OBB of the tree. Recursively, each OBB is split according to a termination criterion considering local shape variation and OBB edge

lengths. Each OBB split bisects the point set at its barycenter perpendicular to the longest edge and applies PCA to the two new point sets. Finally, boolean union operations merge the OBBs and mesh improvement ensures a high quality triangular cage.

For fast computation of coarse cages, XIAN et al. [XLX15] present a voxelization-based region decomposition with subsequent simplification. The method relies on a voxel grid of the model's OBB. A scanline method categorizes the voxels into inside, surface, and outside. The use of flood filling divides the inside voxels into disconnected groups. Dilating the voxels of inner groups yields surface voxels that envelope the input model. The surface voxels dilating the inner voxels constitute the broad model parts, while the other surface voxels constitute the narrow parts. Determining the outside voxel faces of broad regions and the OBBs of narrow regions yields a set of partial cages, which forms a single cage through successive application of boolean operations. Finally, a simplification step merges co-planar voxel faces and collapses edges so that the cage still envelops the model.

4.2.3. Template-based Methods

To enable the reuse of skinning behaviors across similar models, JU et al. [JZvdP*08] construct cages from a library of skinning templates. A skinning template is a specific skeleton configuration that can be applied to characters with a similar joint structure. The system first selects a template matching the character and subsequently constructs a cage fitting the character's shape. For each bone or joint, a polygon represents the cross section along the skeleton. Cage generation scales the cross section vertices radially from its bone or joint so that the cage does not intersect the model. Users may interactively adjust cage vertices to achieve a better cage embedding. To enable cage-based deformation, the system binds the cage to the skeleton to enable cage deformation, whenever the user loads a new skinning template.

As retargeting muscles of a character requires surface deformation, YANG et al. [YCSZ12] use cages for muscle construction from skeletons. Joint by joint, their method generates a cage applying different rules for the number of bones at a joint. For joints with two bones, cage generation constructs a cross section polygon like JU et al. [JZvdP*08]. For joints with one, three or four bones, ray casting finds positions for polygon vertices at intersection points with the skin. To avoid self-intersections, their method shoots several rays to detect overshooting. In addition, a spherical mapping of the cage to the unit sphere enables optimization of vertex positions according to an energy that penalizes self-intersections.

4.2.4. Interactive Cage Generation Methods

As the automatic construction of cages oftentimes requires subsequent adjustments of the cage, many research papers present methods, where prior user interaction affects the manner of cage construction.

CHEN and FENG [CF14] construct cages adapted from skeletons for animated meshes. Users first sketch a skeleton on a silhouette of the model. From the sketched skeleton, CHEN and FENG [CF14] extract prominent cross-sections [SMK*10] along the joints. The cage construction interpolates offsets of simplified cross-sections

and connects the vertices to adjacent offset cross-sections to construct partial cages. Finally, the partial cages are stitched together into a single cage. Due to the use of the skeleton for cage construction, a sequence of cages can be generated for an animated model.

As automatic cage computation can produce unintuitive results for models with high topological complexity, LE and DENG [LD17] propose an interactive cage generation method for designing cages. Their method enables users to specify the semantic model parts with cut slide planes. These slides are transformed into cross sections spanned by quadrilaterals, whose vertices are part of the cage. As the quadrilaterals should be well aligned to the model and consistently oriented, an optimization step rectifies orientations in a least squares manner. Subsequently, Delaunay meshing followed by edge flips for surface smoothness improvement produce a cage. The final step optimizes cage vertex positions using least squares fitting respecting a user-specified offset. The method of LE and DENG [LD17] allows for quick interactive cage design, while it may produce self-intersecting cages.

To enable quick, interactive generation of coarse cages, CALDERON and BOUBEKEUR [CB17] combine an initial parallel voxelization step with subsequent mesh coarsening. After the user chooses a global voxel scale for cage design, the user can interactively brush the regions of the cage that require a finer or coarser bounding approximation. For cage construction, a voxelization forms a 3D rasterization of the input object using a GPU-parallel conservative voxelization method [SS10]. With the use of a 3D structuring element, a morphological closing of the voxels intersecting the object is obtained, performing morphological dilation followed by erosion. From the resulting voxels, a quad-dominant mesh can be extracted that is further simplified using constrained edge collapse operations, guaranteeing that the cage bounds the model with respect to the voxel scale.

For semi-automatic generation using a skeleton, CASTI et al. [CLM*19] guide cage construction by placing bending points along the skeleton curve. A heuristic pre-calculates bending points based on abrupt changes in the thickness of \mathcal{T} . Volumetric meshing of \mathcal{T} allows for computing a harmonic field that emanates radially from the skeleton. After circular sampling of the field around the bending points, their method extracts cross sections of \mathcal{T} orthogonal to the skeleton. Connecting polygons representing the cross sections obtains an initial tight cage that is inflated to avoid intersections with the model. While the method of CASTI et al. [CLM*19] enables fast generation of high-quality symmetric cages, it requires an input skeleton, a prior volumetric meshing step, and a sufficiently dense set of bending points.

4.3. Embedding the Cage

Many cage-based deformation methods (see Section 6) not only depend on a polygonal cage \mathcal{C} but also a volumetric grid $\mathcal{E}_{\mathcal{C}}$ embedding the cage, because these methods solve partial differential equations to construct a set of GBCs. Due to its robustness, unstructured tetrahedral meshing is frequently used to generate $\mathcal{E}_{\mathcal{C}}$. Although fast and robust tetrahedral meshing algorithms are available [CDS12; HZG*18; HSW*20], the dependency on an embedding complicates the workflow, because the quality of the deformations is affected by the properties of $\mathcal{E}_{\mathcal{C}}$.

Table 1: Comparison of cage generation methods ordered by category (top to bottom: offset surfaces simplification, voxelization-based, template-based, and interactive) and year of publication. Being conservative expresses the ability to generate \mathcal{C} not intersecting with \mathcal{T} .

method	no self-intersections	conservative	tightness	symmetry	comment
[SGG*00]	Local	✗	✗	✗	Basic building block for cage generation
[SOS04]	Resolution-dependent	Resolution-dependent	✓	✗	Relies on iso-surfaces
[BWG09]	Local	✓	✓	✗	Intended for deformation transfer
[DLM11]	Global	Vertices of \mathcal{C} only	✓	✗	Features post-hoc cage repair
[SVJ15]	Global if the input has no self-intersections	✓	✓	✗	Generates a sequence of nested cages
[XLG09]	Global w/o smoothing local with smoothing	Vertices of \mathcal{C} only	✓	✗	Smooth and tightly fitting cages
[NKJF09]	Global	✓	✗	✗	Intended for linear elasticity deformable objects
[XLG11]	Global w/o improvement local with improvement	✓	✓	✗	Quick generation of coarse cages for complex models
[XLX15]	Global w/o collapsing local with collapsing	✓	✓	✗	Provides a high-quality cage for a suitable voxel size
[JZvdP*08]	Local	✓	✓	✗	Usability depends on a large template library
[YCSZ12]	Global if input is of genus 0	✓	✓	✗	Intended for muscle design
[CF14]	Local	✓	✓	✗	Generates a sequence of cages from skeletons
[LD17]	Local due to using unbounded cut slides	✓	✓	✓	Expressive interaction through user-defined cuts
[CB17]	Global	✓	✓	✓	Highly efficient and robust
[CLM*19]	Global for a sufficient number of bending nodes	✓	✓	✓	Generates a cage for a skeleton by bending node control

Generally, the more points \mathcal{E}_C includes, the better the accuracy of the resulting GBCs becomes. Thus, a finer resolution of \mathcal{E}_C improves deformation quality, whereas the run time performance of calculating the GBCs benefits from a coarse \mathcal{E}_C , because each element costs computationally. It can be difficult to estimate an appropriate resolution for \mathcal{E}_C , if the intended deformations are unclear at bind time. An alternative to increasing the resolution of \mathcal{E}_C is the use of higher-order elements, while current cage-based deformation methods only use linear elements. In addition, the deformation quality also depends on the shape quality of the tetrahedral elements [She02] in \mathcal{E}_C . Constructing a constrained tetrahedral mesh without elements of ill shape quality is an ongoing research topic [Lo15]. Although scalable mesh improvement methods are available [RPPS17; SMWF22], they do not provide guarantees on the resulting element quality. To decouple deformation quality from element shape quality, one could use the technique of SCHNEIDER et al. [SHD*18] at the cost of decreased run time performance and

implementing a more complex construction of the finite element system.

In order to extract the deformed model from \mathcal{E}_C , two different methods can be used:

1. Insert the vertices \mathbf{V} into \mathcal{E}_C as constrained points and extract them after deformation
2. Interpolate the weights of \mathbf{V} from \mathcal{E}_C and pose the model with interpolated weights

As meshing tools such as TetGen [Han15] typically provide meshes with constrained points in consecutive order, the first method is simple to implement. However, it complicates the meshing of \mathcal{E}_C , because insertion of constrained points leads to additional mesh refinement to avoid elements of low shape quality. The second method provides more convenient meshing and reusability of \mathcal{E}_C , as it can be used for any model enclosed by the cage, given that the resolution of \mathcal{E}_C is fine enough, albeit at the cost of performing

a point lookup step that should be accelerated with a spatial data structure [MWUP22; SMSF20].

4.4. Comparing Cage Generation Methods

We discuss the methods for cage generation and present a systematic comparison in Table 1. The first automatic cage generation methods fall into the offset surface simplification category. Progressively collapsing edges is inherently a sequential procedure, which imposes low run time performance for cage generation. Self-intersections are either resolved within the local topological vicinity or globally across \mathcal{C} . The avoidance of global self-intersections or intersections with \mathcal{T} requires intersection tests for all cage faces [DLM11] or the application of sophisticated offset surface handling [SVJ15], which significantly reduces run time performance. Thus, these methods are not suitable for applications, where users expect a cage immediately after loading a high-resolution model. Especially for high-resolution models, situations may occur, where the simplification scheme is not able to decimate the fine-grained details of the offset surface leading to robustness issues [CB17]. Nonetheless, offset surface simplification methods are successful in achieving a low number of control vertices leading to improved run time performance at bind time. Consequently, cage simplification is an important optimization step in many current cage generation methods.

The voxelization-based approaches offer simple and efficiently parallelizable methods for cage generation. Self-intersections and intersections with \mathcal{T} can be avoided easily, as the cage is formed from non-intersecting voxel faces respecting an offset distance to \mathcal{T} . Without post-optimization the resulting cages are even globally free of self-intersections. Users can control the resolution of the cage by specifying the voxel size. As it is not intuitively predictable which voxel size leads to the intended results, users typically need to run the cage generation method multiple times to obtain a suitable cage. Additionally, the majority of the fully automatic methods only allow for a homogeneous voxel size, whereas some details might need finer sampling. Thus, voxelization-based methods excel at quick and robust cage-generation but lack intuitive control.

The template-based methods are intended specifically for re-using skinning configurations across several animated characters. The key advantage of template-based methods is that they assist users in posing the model, because users only need to perform minor posing for the motion details not covered by the template. Template-based methods come with the drawback that users require a large library to find similar characters with suitable templates. Moreover, template-based methods inherently depend on a skeleton for cage generation. Cage generation constructs an offset surface along the skeleton curve, which is prone to self-intersections, unless the methods of YANG et al. [YCSZ12] are applied.

Since the cages generated by automated methods typically need manual adjustment to be suitable for the intended deformation, the interactive cage generation methods are most useful. While the fully automatic methods typically do not guarantee to provide symmetrically structured cages for symmetric features, many interactive methods provide symmetric cages. The methods by CHEN and FENG [CF14] and CASTI et al. [CLM*19] additionally depend on

a skeleton. Among the interactive methods, LE and DENG [LD17] and CASTI et al. [CLM*19] provide the most fine-grained user control of the cage generation. As CALDERON and BOUBEKEUR [CB17] offer a heterogeneous voxel size that can be interactively controlled, their method benefits from the efficiency and robustness of parallel voxelization.

5. Barycentric Coordinates with Explicit Formulas

The simplest class of coordinates that can be used for cage-based deformation are GBC with a simple closed-form expression. While several such GBC were proposed for convex cages [War96; JSWD05; WSHD07; JLW07], only mean value coordinates are well-defined for arbitrary cages. We review them in detail in Section 5.1 and then discuss several variations of mean value coordinates that overcome some of their limitations.

5.1. Mean Value Coordinates

Mean value coordinates (MVC) were derived by FLOATER [Flo03] in an attempt to find an alternative discretization of harmonic functions in 2D. The classical finite element approach is based on minimizing the Dirichlet energy of a piecewise linear function over a triangulation of the domain and leads to the well-known cotangent weights [Mac49; Duf59; PP93], which constitute the set of discrete harmonic coordinates in 2D [EDD*95; FHK06]. Instead, Floater considers the mean value property of harmonic functions and applies it to piecewise linear functions. The resulting 2D MVC turn out to be non-negative over the kernel of a polygon, well-defined in the whole plane (even in the case of nested polygons), and smooth, except at the vertices of the polygon, where they are only C^0 [HF06].

In order to provide a generalized construction of GBC, FLOATER et al. [FHK06] show that MVC belong to a unifying family of 2D GBC for convex polygons. YAN and SCHAEFER [YS19] present a modification of this family, which leads to GBC that support arbitrary dimensions and non-manifold simplicial control structures, but neither allow for quad cages nor guarantee to be positive inside non-convex shapes. Consequently, this STAR focuses on the evolution of MVC that overcome these limitations.

One way of showing the reproduction property of 2D MVC is by using the fact that the integral of the unit normals of a circle is zero, a property that generalizes to higher dimensions and has been used to derive MVC in 3D [FKR05; JSW05]. Given a cage \mathcal{C} with triangular faces $f \in \partial\mathcal{C}$ and a point $\mathbf{x} \in \mathcal{C}$, we first project the cage to the unit sphere S around \mathbf{x} , giving the cage \mathcal{C}' with spherical triangular faces $f' \in \partial\mathcal{C}'$ and vertices $\mathbf{c}'_i = \mathbf{x} + \mathbf{e}_i$, where $\mathbf{e}_i = (\mathbf{c}_i - \mathbf{x})/r_i$ and $r_i = \|\mathbf{c}_i - \mathbf{x}\|$. For every face f , we then let

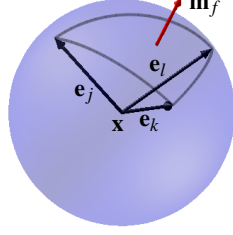
$$\mathbf{m}_f = \int_{f'} (\mathbf{y} - \mathbf{x}) \, d\mathbf{y}$$

be the mean vector of f , defined as the integral of the outward-pointing unit normals over the projected face f' . We then have

$$\sum_{f \in \partial\mathcal{C}} o_f \mathbf{m}_f = \int_S (\mathbf{y} - \mathbf{x}) \, d\mathbf{y} = \mathbf{0}, \quad (4)$$

where o_f is the orientation of f' (i.e., $o_f = 1$, if the outward-pointing normal of f is pointing away from \mathbf{x} and $o_f = -1$, otherwise), because the integral of the unit normals of a sphere is zero.

The key idea now is to “extract” the MVC from the mean vectors \mathbf{m}_f . To this end, let us assume that f is the triangle formed by the cage vertices $\mathbf{c}_j, \mathbf{c}_k, \mathbf{c}_l$, oriented counter-clockwise. It is then clear (see inset) that \mathbf{m}_f lies in the cone spanned by $\mathbf{e}_j, \mathbf{e}_k, \mathbf{e}_l$, so there must exist non-negative weights $\mu_j^f, \mu_k^f, \mu_l^f$, such that



$$\mathbf{m}_f = \mu_j^f \mathbf{e}_j + \mu_k^f \mathbf{e}_k + \mu_l^f \mathbf{e}_l. \quad (5)$$

Inserting this and the definition of \mathbf{e}_i into Eq. (4), we get

$$\sum_{f \in \partial \mathcal{C}} \frac{o_f \mu_j^f}{r_j} (\mathbf{c}_j - \mathbf{x}) + \frac{o_f \mu_k^f}{r_k} (\mathbf{c}_k - \mathbf{x}) + \frac{o_f \mu_l^f}{r_l} (\mathbf{c}_l - \mathbf{x}) = \mathbf{0}$$

and, after rearranging the terms so as to sum over the cage vertices,

$$\sum_{i=1}^{N_C} \sum_{f \ni i} w_i^f (\mathbf{c}_i - \mathbf{x}) = \mathbf{0}, \quad \text{where} \quad w_i^f = \frac{o_f \mu_i^f}{r_i}. \quad (6)$$

It remains to define the values of the MVC $\lambda_1, \dots, \lambda_{N_C}$ at \mathbf{x} as

$$\lambda_i(\mathbf{x}) = \frac{w_i}{\sum_{j=1}^{N_C} w_j}, \quad \text{where} \quad w_i = \sum_{f \ni i} w_i^f. \quad (7)$$

To compute $\lambda_i(\mathbf{x})$, we recall that the mean vector of f can be expressed as

$$\mathbf{m}_f = \frac{1}{2} (\theta_{k,l} \mathbf{n}_{k,l} + \theta_{l,j} \mathbf{n}_{l,j} + \theta_{j,k} \mathbf{n}_{j,k}),$$

where $\theta_{r,s}$ is the angle between \mathbf{e}_r and \mathbf{e}_s or, equivalently, the length of the circular arc between the vertices \mathbf{c}_r' and \mathbf{c}_s' of the spherical triangle f' , and $\mathbf{n}_{r,s} = o_f (\mathbf{e}_r \times \mathbf{e}_s) / \|\mathbf{e}_r \times \mathbf{e}_s\|$ denotes the unit normal of the triangle $(\mathbf{x}, \mathbf{c}_r, \mathbf{c}_s)$, pointing into the tetrahedron $(\mathbf{x}, \mathbf{c}_j, \mathbf{c}_k, \mathbf{c}_l)$ [FKR05; JSW05]. Note that the orientation is defined as

$$o_f = \text{sgn det}(\mathbf{e}_j, \mathbf{e}_k, \mathbf{e}_l)$$

and takes care of faces that flip orientation during projection. Given \mathbf{m}_f , the weights in Eq. (5) can then be computed as

$$\mu_j^f = \frac{\mathbf{n}_{k,l} \cdot \mathbf{m}_f}{\mathbf{n}_{k,l} \cdot \mathbf{e}_j}, \quad \mu_k^f = \frac{\mathbf{n}_{l,j} \cdot \mathbf{m}_f}{\mathbf{n}_{l,j} \cdot \mathbf{e}_k}, \quad \mu_l^f = \frac{\mathbf{n}_{j,k} \cdot \mathbf{m}_f}{\mathbf{n}_{j,k} \cdot \mathbf{e}_l}.$$

The resulting code for evaluating MVC at any $\mathbf{x} \in \mathcal{C}$ is straightforward, but can suffer from numerical instabilities if one or more projected triangles have a small projected area or if \mathbf{x} lies in the plane defined by one of the triangles of \mathcal{C} . Ju et al. discuss how to overcome both problems and provide pseudocode for the robust and efficient evaluation of the mean value interpolant [JSW05, Figure 4].

The definition of the MVC λ_i in Eq. (7) implies that they form a partition of unity and the reproduction property follows from Eq. (6). They are well-defined for any $\mathbf{x} \in \mathbb{R}^3$, smooth, except at the cage vertices \mathbf{c}_i , linear on the triangular faces of the cage, and

they satisfy the Lagrange property. However, they can be negative if \mathbf{x} lies outside the kernel of \mathcal{C} .

5.2. Positive Mean Value Coordinates

The fact that MVC can be negative in certain parts of a non-convex cage may lead to non-intuitive and unwanted deformation results. To overcome this limitation, LIPMAN et al. [LKCL07] propose a modification of MVC which guarantees the positivity of the coordinates at any point inside the cage and eliminates the aforementioned deformation artifacts.

To understand their approach, recall that MVC can be expressed alternatively as

$$\lambda_i(\mathbf{x}) = \int_S \sum_{k=1}^{N_y} \frac{(-1)^{k+1}}{\|\mathbf{y}_k - \mathbf{x}\|} \xi_i(\mathbf{y}_k) d\mathbf{y} \bigg/ \int_S \sum_{k=1}^{N_y} \frac{(-1)^{k+1}}{\|\mathbf{y}_k - \mathbf{x}\|} d\mathbf{y},$$

where S is the unit sphere around \mathbf{x} , the points $\mathbf{y}_1, \dots, \mathbf{y}_{N_y} \in \partial \mathcal{C}$ are the N_y intersections of the ray $\rho_{\mathbf{x}}(\mathbf{y}) = \{\mathbf{x} + t(\mathbf{y} - \mathbf{x}) : t \geq 0\}$ from \mathbf{x} in the direction $\mathbf{y} - \mathbf{x}$ (sorted by increasing distance), for any $\mathbf{y} \in S$, with the cage boundary, and $\xi_i: \partial \mathcal{C} \rightarrow \mathbb{R}$ is the continuous function that is linear over each triangle $f \in \partial \mathcal{C}$ with values $\xi_i(\mathbf{c}_j) = \delta_{i,j}$ at the cage vertices [JSW05; Bel06]. Note that the alternating signs in the numerators of the fractions in both sums take care of the orientation of the faces (after projection onto S) which the intersection points \mathbf{y}_k belong to, because odd indices k correspond to faces with an outward-pointing normal and vice versa for even k .

The idea of LIPMAN et al. [LKCL07] is to simply ignore all intersection points with index $k > 1$ and to define the Positive Mean Value Coordinates (PMVC) as

$$\lambda_i(\mathbf{x}) = \int_S \frac{1}{\|\mathbf{y}_1 - \mathbf{x}\|} \xi_i(\mathbf{y}_1) d\mathbf{y} \bigg/ \int_S \frac{1}{\|\mathbf{y}_1 - \mathbf{x}\|} d\mathbf{y}.$$

By definition, PMVC are positive for any $\mathbf{x} \in \text{Int } \mathcal{C}$, and they are identical to MVC for convex cages, or more generally at any \mathbf{x} inside the kernel of \mathcal{C} , because $N_y = 1$ in these cases. Moreover, it can be shown that they form a partition of unity, satisfy the reproduction as well as the Lagrange property, and that they are linear on the triangular faces of the cage.

LIPMAN et al. [LKCL07] describe an efficient algorithm for approximating PMVC using the GPU, which turns out to be accurate and fast enough for interactive shape deformation. One limitation of PMVC is that they are only C^0 across the supporting planes of the triangles adjacent to non-convex cage vertices, but the resulting artifacts are often negligible in practice.

5.3. Mean Value Coordinates for Planar n -gons using Spherical Barycentric Coordinates

LANGER et al. [LBS06] introduced spherical barycentric coordinates (SBC). While detailing those in the general case is beyond the scope of this STAR, we explain their geometric construction for the derivation of GBC for planar n -gon cages. The construction relies on the observation by JU et al. [JSW05], that valid GBC can be obtained from per-face weights w_i^f as long as

$$\sum_{i \in f} w_i^f (\mathbf{c}_i - \mathbf{x}) = \mathbf{m}_f, \quad \sum_{i \in f} w_i^f = w_i, \quad \sum_{f \in \partial \mathcal{C}} \mathbf{m}_f = \mathbf{0}. \quad (8)$$

One can see that this leads directly to

$$\sum_{i=1}^{N_F} \mathbf{m}_{f_i} = \sum_{i=1}^{N_C} \sum_{f \ni i} w_i^f (\mathbf{c}_i - \mathbf{x}) = \sum_{i=1}^{N_C} \left(\sum_{f \ni i} w_i^f \right) (\mathbf{c}_i - \mathbf{x}) = 0,$$

if and only if $\frac{\sum_{i=1}^{N_C} w_i^f \mathbf{c}_i}{\sum_{i=1}^{N_C} w_i^f} = \mathbf{x}$, provided that $\sum_{i=1}^{N_C} w_i^f \neq 0$. By defining the per-face mean-vector \mathbf{m}_f as the integral of the unit normal over the portion of the sphere covered by the projection of the face f [JSW05] one obtains $\sum_{f \in \partial C} \mathbf{m}_f = 0$. This results in a mean-vector \mathbf{m}_f that can be computed as

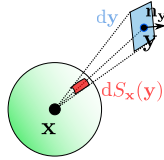
$$\mathbf{m}_f = - \sum_{i \in f} \frac{\theta_i^f \mathbf{n}_i^f}{2},$$

which extends the triangle mean-vector of the standard MVC. Note that Eq. (8) admits an infinite number of solutions. This is related to the existence of an infinite number of ways to parameterize a smooth surface interpolating the $|f|$ vertices of f using barycentric positive basis functions Γ_f^i . One could define such basis functions Γ_f^i , expressing the geometry of the facet as $\mathbf{y} = \sum_{i=1}^{N_C} \Gamma_f^i(\mathbf{y}) \mathbf{c}_i$, and trying to compute the per-face unnormalized weight as

$$w_i^f = \int_{S_x(f)} \frac{\Gamma_f^i(\mathbf{y})}{\|\mathbf{y} - \mathbf{x}\|} dS_x(\mathbf{y}), \quad (9)$$

where $S_x(s)$ is the projection of a surface element s , e.g. f , onto the unit sphere and $dS_x(\mathbf{y})$ is the infinitesimal solid angle element spanned by $d\mathbf{y}$ from \mathbf{x} using the notation by JU et al. [JSW05] (see inset):

$$dS_x(\mathbf{y}) = \frac{(\mathbf{y} - \mathbf{x}) \cdot \mathbf{n}_y}{\|\mathbf{y} - \mathbf{x}\|^3} d\mathbf{y}. \quad (10)$$

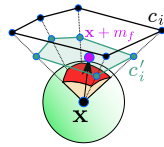


Using this notation, one can read all surface integrals of the type $\int_{S_x(s)} f(\mathbf{y}) dS_x(\mathbf{y})$ as

$$\int_{S_x(s)} f(\mathbf{y}) dS_x(\mathbf{y}) = \int_s f(\mathbf{y}) \frac{(\mathbf{y} - \mathbf{x}) \cdot \mathbf{n}_y}{\|\mathbf{y} - \mathbf{x}\|^3} d\mathbf{y}.$$

Defining weights from Eq. (9) would directly imply Eq. (8). While this approach (used for example to derive QMVC) follows the original MVC approach, it requires integrating complex functions on complex geometries, which is not always feasible in practice.

Instead, LANGER et al. [LBS06] propose a concise and purely geometric construction and derive valid weights interpolating any desired set of per-face basis functions Γ_f^i . Once \mathbf{m}_f is computed, the vertices \mathbf{c}_i are projected towards \mathbf{x} by intersecting the lines $(\mathbf{x}\mathbf{c}_i)$ with the tangent plane going through $\mathbf{x} + \mathbf{m}_f$ orthogonal to \mathbf{m}_f (see inset), thus constructing a planar polygon $f' = \{\mathbf{c}'_i\}$ (with $\mathbf{c}'_i - \mathbf{x} = \lambda_i^f (\mathbf{c}_i - \mathbf{x})$) that contains $\mathbf{x} + \mathbf{m}_f$.



2D barycentric coordinates ω_i^f for $\mathbf{x} + \mathbf{m}_f$ w.r.t. $\{\mathbf{c}'_i\}$ can therefore be computed (using, e.g., 2D MVC) as

$$\sum_{i \in f} \omega_i^f \mathbf{c}'_i = \mathbf{x} + \mathbf{m}_f, \quad \sum_{i \in f} \omega_i^f = 1.$$

This leads directly to

$$\sum_{i \in f} \omega_i^f (\mathbf{c}'_i - \mathbf{x}) = \sum_{i \in f} \omega_i^f \lambda_i^f (\mathbf{c}_i - \mathbf{x}) = \mathbf{m}_f,$$

demonstrating that the re-weighted coordinates $w_i^f = \omega_i^f \lambda_i^f$ are valid per-face unnormalized weights for arbitrary planar n -gons.

The constructed coordinates interpolate the chosen per-face 2D barycentric coordinates ω_i^f and match the standard MVC for triangles. While the two techniques differ in the way they derive the coordinates, the constructed per-face unnormalized coordinates w_i^f both fulfill $\sum_{i \in f} w_i^f (\mathbf{c}_i - \mathbf{x}) = \mathbf{m}_f$, for which the solution is unique.

5.4. Mean Value Coordinates for Tri-Quad Cages

While SBC enable the use of n -gon cages, they are of limited practical use, as the restriction to planar polygons imposes constraints on cage design. MVC for tri-quad cages were devised by THIERY et al. [TMB18] to allow for interpolating coordinates coping with non-planar quad cages, for which *bilinear quads* were used as a smooth, underlying geometric model. While the derivation is similar to other MVC-based methods, the key challenge lies in integrating the mean-value kernels on curved geometries. Following once again the remark of JU et al. [JSW05] that valid coordinates can be derived if per-face unnormalized coordinates w_i^f satisfy the mean-vector property

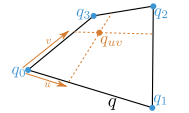
$$\sum_{i \in f} w_i^f (\mathbf{c}_i - \mathbf{x}) = \mathbf{m}_f, \quad \sum_{f \in \partial C} \mathbf{m}_f = 0,$$

a set of MVC $\{\lambda_i | \sum_{i=1}^{N_C} \lambda_i \mathbf{c}_i = \mathbf{x}; \sum_i \lambda_i = 1\}$ was derived for tri-quad cages as

$$w_i^f = \int_{S_x(f)} \frac{\Gamma_f^i(\mathbf{y})}{\|\mathbf{y} - \mathbf{x}\|} dS_x(\mathbf{y}), \quad w_i = \sum_{f \ni i} w_i^f, \quad \lambda_i = w_i / \sum_{j=1}^{N_C} w_j.$$

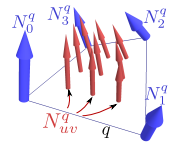
For planar polygons, the parameterization of the underlying geometric model *only impacts the way the space will be deformed after a cage edit*, and it does not impact the geometry of the binding cage itself. For non-planar polygons, the choice of the underlying geometric model impacts the definition of the facets at binding time, and therefore the partition of the space into the interior and the exterior of the cage as well.

THIERY et al. [TMB18] use *bilinear quads* that possess a set of elementary geometric properties. A quad $q = (\mathbf{q}_0, \mathbf{q}_1, \mathbf{q}_2, \mathbf{q}_3)$ is equipped with bilinear coordinates $\{b_{uv}^0, b_{uv}^1, b_{uv}^2, b_{uv}^3\} = \{(1-u)(1-v), u(1-v), uv, (1-u)v\}$ at parameters $(u, v) \in \mathbb{R}^2$ (see inset) and is a smooth geometry given by $\mathbf{q}_{uv} = \sum_{k=0}^3 b_{uv}^k \mathbf{q}_k$.



Noting \mathbf{N}_k^q the 4 non-normalized corner normals defined by $\mathbf{N}_k^q = (\mathbf{q}_{k+1} - \mathbf{q}_k) \times (\mathbf{q}_{k+3} - \mathbf{q}_k)$ (all indices modulo 4), the non-normalized normal vector \mathbf{N}_{uv}^q at parameter (u, v) is given by

$$\mathbf{N}_{uv}^q = \sum_{k=0}^3 b_{uv}^k \mathbf{N}_k^q.$$



Note that the *surface element* corresponding to the (u, v) -parameterization is given by $d\mathbf{q}_{uv} = \|\mathbf{N}_{uv}^q\| du dv$ and surface integrals over q can be written as

$$\int_q f dq = \int_0^1 \int_0^1 f(\mathbf{q}_{uv}) \|\mathbf{N}_{uv}^q\| du dv. \quad (11)$$

Equipped with the bilinear quad's geometric model, one can see that

$$\sum_{i=0}^3 w_{\mathbf{q}_i}^q (\mathbf{q}_i - \mathbf{x}) = \mathbf{m}_q,$$

where $\mathbf{m}_q = \sum_{i=0}^3 \frac{-\theta_i^q \mathbf{n}_i^q}{2}$ (see inset), which can be written in matrix form as

$$\mathbf{A}_q \mathbf{w}^q = \mathbf{m}_q, \quad (12)$$

where $\mathbf{w}^q = (w_{\mathbf{q}_0}^q, w_{\mathbf{q}_1}^q, w_{\mathbf{q}_2}^q, w_{\mathbf{q}_3}^q)^\top \in \mathbb{R}^4$ are the 4 unknown unnormalized coordinates, and the j -th column of \mathbf{A}_q is given by $(\mathbf{q}_j - \mathbf{x})$.

This equation is valid for non-planar geometry interpolating the quad edges. As \mathbf{A}_q cannot be full-rank, the tri-quad MVC λ_i cannot be deduced from it alone. Similarly, as non-planar quads are targeted, the construction of LANGER et al. [LBS06] does not yield valid coordinates in this case.

After having admittedly looked in vain for an exact solution, THIERY et al. [TMB18] resorted to building an efficient approximate scheme resulting in smooth, bilinearly interpolating, and valid coordinates. Their main observation is that the general solution to Eq. (12) takes the form of the 4D “minimal-norm solution” (given by pseudo-inversion of Eq. (12)) and an additional 4D vector κ aligned with its one-dimensional kernel:

$$\mathbf{w}^q = \bar{\mathbf{w}}^q + \alpha \kappa, \quad \bar{\mathbf{w}}^q = \mathbf{A}_q^\dagger \mathbf{m}_q, \quad \mathbf{A}_q \kappa = 0.$$

While κ and $\bar{\mathbf{w}}^q$ can be computed from a singular value decomposition of \mathbf{A}_q , closed-form expressions can be derived geometrically. As $\kappa = (k_0, k_1, k_2, k_3)^\top$ is the null-space vector of \mathbf{A}_q , it follows that $\sum_i k_i (\mathbf{q}_i - \mathbf{x}) = 0$. This means that k_i are the barycentric coordinates of \mathbf{x} in the tetrahedron $(\mathbf{q}_0, \mathbf{q}_1, \mathbf{q}_2, \mathbf{q}_3)$:

$$\kappa = (k_0, k_1, k_2, k_3)^\top, \quad k_i = (-1)^i |\mathbf{q}_{i+1} - \mathbf{x}; \mathbf{q}_{i+2} - \mathbf{x}; \mathbf{q}_{i+3} - \mathbf{x}|.$$

The minimal-norm solution fulfills

$$\begin{pmatrix} \mathbf{A}_q \\ \kappa^\top \end{pmatrix} \cdot \bar{\mathbf{w}}^q = \begin{pmatrix} \mathbf{m}_q \\ 0 \end{pmatrix}$$

Setting $\mathbf{p}_i = \mathbf{q}_i - \mathbf{x}$, one obtains

$$\bar{\mathbf{w}}^q = \frac{1}{D} \begin{pmatrix} k_1 |\mathbf{p}_2; \mathbf{p}_3; \mathbf{m}_q| + k_2 |\mathbf{p}_3; \mathbf{p}_1; \mathbf{m}_q| + k_3 |\mathbf{p}_1; \mathbf{p}_2; \mathbf{m}_q| \\ k_0 |\mathbf{p}_3; \mathbf{p}_2; \mathbf{m}_q| + k_2 |\mathbf{p}_0; \mathbf{p}_3; \mathbf{m}_q| + k_3 |\mathbf{p}_2; \mathbf{p}_0; \mathbf{m}_q| \\ k_0 |\mathbf{p}_1; \mathbf{p}_3; \mathbf{m}_q| + k_1 |\mathbf{p}_3; \mathbf{p}_0; \mathbf{m}_q| + k_3 |\mathbf{p}_0; \mathbf{p}_1; \mathbf{m}_q| \\ k_0 |\mathbf{p}_2; \mathbf{p}_1; \mathbf{m}_q| + k_1 |\mathbf{p}_0; \mathbf{p}_2; \mathbf{m}_q| + k_2 |\mathbf{p}_1; \mathbf{p}_0; \mathbf{m}_q| \end{pmatrix}$$

where $D = k_0^2 + k_1^2 + k_2^2 + k_3^2 = \|\kappa\|^2$.

Exposing this decomposition allows casting the problem to the definition of this single α -coordinate. The authors proceed by building a *smooth approximant* $\bar{\mathbf{w}}^q \simeq \mathbf{w}^q$ of the coordinates, and using this estimate to build an associated approximate α -coordinate.

Doing so ensures valid barycentric coordinates and simplifies the problem to finding a robust estimate of a single scalar coordinate using an adaptive Riemann summation strategy, which approximates the integral in Eq. (11).

6. Energy Minimization-based Barycentric Coordinates

Besides closed-form coordinates, it is also possible to define GBC as a solution of a suitable minimization problem, subject to certain constraints. Typically, the numerical discretization of a partial differential equation (PDE) provides an energy term to compute energy minimization-based coordinates (EMC). We discuss two such GBC in Sections 6.1 and 6.3, and a related approach in Section 6.2, which does not yield GBC as defined in Section 2.1, but provides an intuitive deformation tool.

6.1. Harmonic Coordinates

FLOATER et al. [FHK06] noticed that GBC with the desired properties can be obtained as the unique solution to the Laplace equation subject to suitable boundary conditions. While the resulting harmonic coordinates (HC) do not have a closed-form solution, JOSHI et al. [JMD*07] show that numerical approximations of the exact solutions are sufficient in the context of cage-based deformation. They suggest the following procedure to define each coordinate function λ_i .

Let $\xi_i: \partial\mathcal{C} \rightarrow \mathbb{R}$ be the continuous function that is defined piecewise over the polygonal faces $f \in \partial\mathcal{C}$ of \mathcal{C} , such that ξ_i is linear along the edges of \mathcal{C} , harmonic inside each face f , and $\xi_i(\mathbf{c}_j) = \delta_{i,j}$ for $j = 1, \dots, N_{\mathcal{C}}$. Note that this definition implies that ξ_i vanishes over all faces that are not adjacent to \mathbf{c}_i and that ξ_i is linear over the triangular and bilinear over the quadrilateral faces of \mathcal{C} . We then define λ_i as the solution to the Laplace equation

$$\Delta \lambda_i(\mathbf{x}) = 0, \quad \mathbf{x} \in \text{Int } \mathcal{C}$$

subject to the Dirichlet boundary condition

$$\lambda_i(\mathbf{x}) = \xi_i(\mathbf{x}), \quad \mathbf{x} \in \partial\mathcal{C}.$$

Classical approaches for computing an approximation of the exact solution of this PDE include finite difference methods [JMD*07], the finite element method [MKB*08], the boundary element method [Rus07], and the method of fundamental solutions [FK98].

Since constant and linear functions are harmonic, it follows that HC form a partition of unity and satisfy the reproduction property. Moreover, the boundary condition implies the Lagrange property and the correct behavior on the polygonal faces of the cage, the maximum principle ensures that they are non-negative, and HC are smooth over $\text{Int } \mathcal{C}$, because they are solutions to the Laplace equation. However, while these properties hold for the exact solutions, some of them are lost due to approximation. The finite difference and finite element approximations are not smooth and only the finite element method guarantees the correct behavior over $\partial\mathcal{C}$.

6.2. Bounded Biharmonic Weights

Modeling objects using either cages or skeletons can be tedious, because each control structure has its merits and shortcomings. For

instance, skeletons are well-suited for articulated character motion, but are restrictive when it comes to shape modeling, e.g., thickening the arms of a character. In addition, the construction of a fully enclosing cage is a difficult task (see Section 4.2), which makes the local use of cages and skeletons attractive. Thus, JACOBSON et al. [JBPS11] present bounded biharmonic weights (BBW) to bind a model to several cages, skeletons, and point handles. Strictly speaking, BBW are the result of minimizing piecewise-linear functions w_j , which do not provide coordinates for the domain Ω of the deformation. In order to combine cages with other control structures, JACOBSON et al. [JBPS11] exploit the fact that cage-based deformation is a special case of LBS (see Section 2.2). For a number $N \geq N_C$ of handles, the computation of BBW performs variational weight optimization minimizing Laplacian energy subject to a multitude of constraints that enforce interpolation:

$$\begin{aligned} & \arg \min_{w_1, \dots, w_N} \sum_{j=1}^N \frac{1}{2} \int_{\Omega} \|\Delta w_j\|^2 dV \\ & \text{subject to } w_j(\mathbf{h}_k) = \delta_{jk}, \\ & \quad \forall f \in \partial \mathcal{C}, w_j \text{ is linear on } f, \\ & \quad \forall \mathbf{x} \in \Omega, \sum_{j=1}^N w_j(\mathbf{x}) = 1, \\ & \quad \forall \mathbf{x} \in \Omega, w_j(\mathbf{x}) \in [0, 1], j = 1, \dots, N. \end{aligned} \quad (13)$$

As shape preservation during deformation is an important property, the deformation should allow to preserve a user-specified sub-region $\Pi \subset \Omega$. For this reason, the BBW formulation admits the incorporation of a rigidity mask represented by functions $\rho: \Pi \rightarrow \mathbb{R}^+$ and the addition of the following least squares term:

$$\sum_{j=1}^N \frac{1}{2} \int_{\Pi} \rho \|\Delta w_j\|^2 dV.$$

An embedding mesh \mathcal{E} discretizes the domain Ω to numerically solve the optimization problem. The standard linear finite element method (FEM) is a suitable numerical method for computing BBW. The typical approach to solve Laplacian energy offers a discretization of Eq. (13):

$$\sum_{j=1}^N \frac{1}{2} \int_{\Omega} \|\Delta w_j\|^2 dV \approx \frac{1}{2} \sum_{j=1}^N \mathbf{w}_j^T (\mathbf{L} \mathbf{M}_{\text{lump}}^{-1} \mathbf{L}) \mathbf{w}_j, \quad (14)$$

where \mathbf{M}_{lump} is the lumped mass matrix and \mathbf{L} is the cotangential Laplacian matrix. A quadratic programming solver such as Mosek [AA00] can be customized to compute BBW based on the matrices in Eq. (14), the constraints in Eq. (13), and \mathcal{E} . Libigl [JP*24] provides an open source implementation of a BBW solver using the active set method (see, e.g., [NW06]).

6.3. Local Barycentric Coordinates

Since cage-based deformation expresses the geometry as an affine sum of cage vertices, the relocation of a single control point can lead to a global change by propagation into the entire domain Ω , which is not intended by the user. In order to overcome this limitation, ZHANG et al. [ZDL*14] present local barycentric coordinates (LBC). For each interior point, LBC only include a small set of

nearby cage vertices for the affine combination, while the coordinates of distant cage vertices vanish, falling below $\epsilon \approx 0$.

The key strategy to compute LBC is to minimize the total variation of the functions λ_i . Total variation energies offer two crucial benefits for deformation control. First, total variation provides a metric for oscillation [CV01]. Thus, reducing total variation diminishes oscillation. Second, the total variation of a set equals the perimeter of the set [EG15], which can be used to achieve locality. Given a strict super level set $L_s^+(\lambda_i) = \{\lambda_i < s\} = \{\mathbf{x} \in \Omega \mid \lambda_i(\mathbf{x}) > s\}$ of an arbitrary but fixed s and a GBC function λ_i , the locality of λ_i increases if the area/volume of $L_s^+(\lambda_i)$ decreases. With the use of total variation of λ_i , one can minimize the perimeter $P(L_s^+(\lambda_i); \Omega)$ to increase locality:

$$\int_{-\infty}^{+\infty} P(L_s^+(\lambda_i); \Omega) ds = \int_{\Omega} |\nabla \lambda_i| dV.$$

Minimizing the sum of total variations for all N_C functions λ_i yields LBC for the cage. In order to allow for control of locality, the penalty coefficients $\hat{\phi}_i$ adjust the locality of the functions λ_i . As a result, solving the following variational convex optimization problem subject to constraints enforcing the desired interpolation properties produces LBC:

$$\arg \min_{\lambda_1, \dots, \lambda_{N_C}} \sum_{i=1}^{N_C} \int_{\Omega} \hat{\phi}_i \|\lambda_i\| dV \quad (15a)$$

$$\text{subject to } \sum_{i=1}^{N_C} \lambda_i(\mathbf{x}) \mathbf{c}_i = \mathbf{x}, \sum_{i=1}^{N_C} \lambda_i(\mathbf{x}) = 1, \lambda_i(\mathbf{x}) \geq 0, \forall \mathbf{x} \in \Omega \quad (15b)$$

$$\lambda_i(\mathbf{c}_j) = \delta_{ij}, \forall i, j \in \{1, \dots, N_C\} \quad (15c)$$

$$\lambda_i \text{ is linear on all } f, \forall i \in \{1, \dots, N_C\} \quad (15d)$$

The coefficients $\hat{\phi}_i$ penalize the gradient norm based on the geodesic distance $g_i(\cdot)$ to the cage vertex \mathbf{c}_i and a continuous function $\tau: [0, 1] \rightarrow [0, 1]$:

$$\hat{\phi}_i = \tau \left(\frac{g_i(\mathbf{x})}{\arg \max_{\mathbf{y} \in \Omega} g_i(\mathbf{y})} \right), \quad \text{with } \mathbf{x} \in \Omega.$$

When choosing a monotonically increasing function for τ , points more distant from \mathbf{c}_i receive larger penalty coefficients, which leads to more local support. Analogously, the choice of a monotonically decreasing function for τ reduces local support.

For computing a solution of Eq. (15a), an embedding tetrahedral grid \mathcal{E}_C discretizes Ω . As the numerical computation of LBC imposes low run time performance for high-resolution models, TAO et al. [TDZ19] simplify the discretized formulation of the continuous problem in Eq. (15a) while incurring only negligible deviation from the original discretization. This simplification primarily capitalizes on the removal of the non-negativity constraint to reformulate the problem so that costly computations for solving global linear systems can be omitted. Their numerical scheme first computes a solution for the gradients $\nabla_s \lambda_i$ of the tetrahedra $s \in \mathcal{E}_C$ and subsequently integrate the LBC λ_i from the gradients propagating from the boundary.

In order to enforce Lagrange and linearity properties at the boundary, the gradient \mathbf{g}_b^i of the function λ_i at the boundary triangle b should be co-planar to b . The following Neumann boundary

condition ensures this: $\mathbf{n}_b \times (\mathbf{g}_b^i - \mathbf{h}_b^i) = 0$, where \mathbf{n}_b is the normal of b and \mathbf{h}_b^i is the directional derivative of \mathbf{g}_b^i on b . In addition, the final integration step for the functions λ_i demands gradients \mathbf{g}_f^i at interior faces $f \in \mathcal{F}$ to be co-linear for the two adjacent tetrahedra s and s' : $\mathbf{n}_f \times (\mathbf{g}_{f_s}^i - \mathbf{g}_{f_{s'}}^i) = 0$, where \mathbf{n}_f is the normal of f and $\mathbf{g}_{f_s}^i$ as well as $\mathbf{g}_{f_{s'}}^i$ represent the local gradients at the face f from s or s' , respectively.

The discretized problem is formulated with auxiliary variables to obtain a separable target function that can be minimized with the alternating direction method of multiplier (ADMM) [Boy10]. The final formulation uses each auxiliary variable for a specific part of the optimization problem. $\mathbf{Z} \in \mathbb{R}^{3N_t \times N_C}$ helps enforce total variation for the gradients \mathbf{g}_f^i for each of the N_t tetrahedra in \mathcal{E}_C , $\mathbf{X} \in \mathbb{R}^{3N_b \times N_C}$ assists enforcing the boundary conditions for the N_b boundary faces of \mathcal{E}_C , and $\mathbf{Y} \in \mathbb{R}^{6N_f^{\text{int}} \times N_C}$ assists enforcing the integrability condition for the N_f^{int} interior faces of \mathcal{E}_C . For brevity, $\mathbf{G} \in \mathbb{R}^{3N_t \times N_C}$ denotes all gradient variables. As a result, the following optimization problem is a discretization of Eq. (15a):

$$\arg \min_{\mathbf{G}, \mathbf{Z}, \mathbf{X}, \mathbf{Y}} \sum_{s \in \mathcal{E}_C} \sum_{i=1}^{N_C} \hat{\phi}_i^s V_s \|\mathbf{z}_s^i\| + \hat{\sigma}_1(\mathbf{G}) + \hat{\sigma}_2(\mathbf{X}) + \hat{\sigma}_3(\mathbf{Y})$$

$$\text{subject to } \mathbf{Z} = \mathbf{G}, \mathbf{X} = \mathbf{G}\mathbf{S}_X, \mathbf{Y} = \mathbf{G}\mathbf{S}_Y,$$

where indicator functions $\hat{\sigma}_1$, $\hat{\sigma}_2$, and $\hat{\sigma}_3$ enforce the conditions in Eq. (15a) and \mathbf{S}_X as well as \mathbf{S}_Y are sparse selection matrices that choose the gradient variables for the boundary and the integrability conditions, respectively. $\hat{\sigma}_1$ ensures the condition in Eq. (15b), $\hat{\sigma}_2$ ensures the condition in Eq. (15c), and $\hat{\sigma}_3$ ensures the condition in Eq. (15d).

7. Probability-based Coordinates

One major advantage of EMC is that they are guaranteed to be non-negative even for non-convex cages, but this comes at the price of having to solve a global non-linear optimization problem. In fact, it is not possible to evaluate the exact coordinates locally at an arbitrary point $\mathbf{x} \in \mathcal{C}$. Instead, an approximation of the coordinates must first be computed globally by discretizing and solving a PDE. An alternative construction of non-negative GBC that overcomes this limitation is related to statistical concepts. In this approach, the GBC of \mathbf{x} are seen as the probabilities of discrete random events that are associated with the cage vertices \mathbf{c}_i , and we are looking for a probability distribution, such that the expected value of the random variable $\mathbf{c} = (\mathbf{c}_1, \dots, \mathbf{c}_{N_C})$ is $\mathbb{E}[\mathbf{c}] = \sum_{i=1}^{N_C} \lambda_i(\mathbf{x}) \mathbf{c}_i = \mathbf{x}$. This approach typically leads to an optimization problem that is local in the sense that it determines the GBC $\lambda_i(\mathbf{x})$ of \mathbf{x} independently of the coordinates $\lambda_i(\mathbf{y})$ of all other points $\mathbf{y} \neq \mathbf{x}$.

7.1. Maximum Entropy Coordinates

SUKUMAR [Suk04] proposes to consider the probability distribution with least-biased statistical inference. The resulting maximum entropy coordinates (MEC) $\lambda = (\lambda_1(\mathbf{x}), \dots, \lambda_{N_C}(\mathbf{x}))$ of $\mathbf{x} \in \mathcal{C}$ can be found by maximizing the *Shannon entropy*

$$H_{\mathbf{x}}(\lambda) = - \sum_{i=1}^{N_C} \lambda_i(\mathbf{x}) \log \lambda_i(\mathbf{x}) \quad (16)$$

subject to the constraints

$$\sum_{i=1}^{N_C} \lambda_i(\mathbf{x}) = 1, \quad \sum_{i=1}^{N_C} \lambda_i(\mathbf{x}) \mathbf{c}_i = \mathbf{x}. \quad (17)$$

This works well for convex cages, but the coordinates λ do not satisfy the Lagrange property at concave cage vertices. To overcome this limitation, HORMANN and SUKUMAR [HS08] propose to replace the functional in Eq. (16) with the *Shannon–Jaynes entropy*

$$H_{\mathbf{x}}(\lambda) = - \sum_{i=1}^{N_C} \lambda_i(\mathbf{x}) \log \frac{\lambda_i(\mathbf{x})}{m_i(\mathbf{x})}, \quad (18)$$

which can also be seen as the negative Kullback–Leibler divergence of λ from the reference distribution $m = (m_1(\mathbf{x}), \dots, m_{N_C}(\mathbf{x}))$. HORMANN and SUKUMAR [HS08] suggest to define the *prior functions* $m_i: \mathcal{C} \rightarrow \mathbb{R}$ as

$$m_i(\mathbf{x}) = \frac{\pi_i(\mathbf{x})}{\sum_{j=1}^{N_C} \pi_j(\mathbf{x})}, \quad \pi_i(\mathbf{x}) = \frac{1}{\prod_{f \ni i} \rho_f(\mathbf{x})}, \quad (19)$$

where the product in the denominator of π_i ranges over all faces adjacent to \mathbf{c}_i . Assuming that $f \in \partial \mathcal{C}$ is a polygon with k vertices $\mathbf{c}_1, \dots, \mathbf{c}_k$, the function $\rho_f: \mathcal{C} \rightarrow \mathbb{R}$ is defined as

$$\rho_f(\mathbf{x}) = \sum_{i=1}^k A(\mathbf{x}, \mathbf{c}_i, \mathbf{c}_{i+1}) - A(\mathbf{c}_1, \dots, \mathbf{c}_k),$$

where $A(\mathbf{y}_1, \dots, \mathbf{y}_n)$ denotes the area of the polygon with vertices $\mathbf{y}_1, \dots, \mathbf{y}_n$, so that $\rho_f(\mathbf{x})$ is non-negative and vanishes if and only if $\mathbf{x} \in f$. Consequently, the prior function m_i in Eq. (19) vanishes over $\partial \mathcal{C}$, except on the faces adjacent to \mathbf{c}_i , and satisfies $m_i(\mathbf{c}_j) = \delta_{i,j}$. This is exactly the behavior that we would like the coordinates to have and MEC inherit it from these prior functions.

Maximizing $H_{\mathbf{x}}(\lambda)$ in Eq. (18) under the constraints in Eq. (17) turns out to be equivalent to first finding the unique vector $\boldsymbol{\eta} \in \mathbb{R}^3$ that minimizes the strictly convex function

$$F(\boldsymbol{\eta}) = \log \sum_{i=1}^{N_C} Z_i(\boldsymbol{\eta}), \quad Z_i(\boldsymbol{\eta}) = m_i(\mathbf{x}) e^{-\boldsymbol{\eta}^\top (\mathbf{c}_i - \mathbf{x})},$$

which can be done efficiently with a few iterations of Newton's method and then defining the coordinates as

$$\lambda_i(\mathbf{x}) = \frac{Z_i(\boldsymbol{\eta})}{\sum_{j=1}^{N_C} Z_j(\boldsymbol{\eta})}, \quad i = 1, \dots, N_C.$$

The resulting MEC have all the properties listed in Section 2.1, except for locality. However, the particular choice of prior functions in Eq. (19) is not “geometry-aware” and can sometimes lead to badly shaped coordinate functions and undesired deformation artifacts.

7.2. Maximum Likelihood Coordinates

An alternative approach, which is motivated by the concept of maximum likelihood estimation, was suggested by CHANG et al. [CDH23] who propose to maximize the product of the coordinates $\lambda_i(\mathbf{x})$ instead of the *Shannon entropy* in Eq. (16). In order to extend this method to non-convex cages, they do not rely on prior functions, but rather pre-process the cage with a series of projection

and averaging steps, which were inspired by the construction of 2D iterative coordinates [DCH20].

To find the maximum likelihood coordinates (MLC) of $\mathbf{x} \in \mathcal{C}$, we start by shifting the cage \mathcal{C} by $-\mathbf{x}$ and then projecting the shifted cage vertices onto the unit sphere around the origin, resulting in the spherical cage $\hat{\mathcal{C}}$ with vertices

$$\hat{\mathbf{c}}_i = \frac{\mathbf{c}_i - \mathbf{x}}{\|\mathbf{c}_i - \mathbf{x}\|}.$$

We then smooth this cage with two averaging steps to get the cage $\hat{\mathcal{C}}$ with vertices $\hat{\mathbf{c}}_i$. In the first averaging step, we determine the mean vectors \mathbf{m}_f for the spherical faces of $\hat{\mathcal{C}}$, and in the second averaging step, we accumulate the normalized mean vectors of the spherical faces adjacent to $\hat{\mathbf{c}}_i$ and normalize the result to obtain

$$\hat{\mathbf{c}}_i = \mathbf{t}_i / \|\mathbf{t}_i\|, \quad \mathbf{t}_i = \sum_{f \ni i} \mathbf{m}_f / \|\mathbf{m}_f\|.$$

Since the mean vectors \mathbf{m}_f can be expressed as non-negative linear combinations of the shifted cage vertices (cf. Eq. (5) in Section 5.1), the matrix that includes all the vertices of $\hat{\mathcal{C}}$ can be expressed in terms of the matrix that includes all vertices of \mathcal{C} as

$$\hat{\mathbf{C}} = \mathbf{M}(\mathbf{C} - \mathbf{e}\mathbf{x}^T), \quad (20)$$

where $\mathbf{e} = (1, \dots, 1)^T \in \mathbb{R}^n$ and $\mathbf{M} \in \mathbb{R}^{N_C \times N_C}$ is a non-negative matrix.

Suppose now that we are given some non-negative GBC $\hat{\lambda} = (\hat{\lambda}_1, \dots, \hat{\lambda}_{N_C})$ of the origin with respect to the cage $\hat{\mathcal{C}}$, that is, $\hat{\lambda}\hat{\mathbf{C}} = 0$, then we can define the GBC $\lambda = (\lambda_1(\mathbf{x}), \dots, \lambda_{N_C}(\mathbf{x}))$ of \mathbf{x} with respect to the original cage \mathcal{C} as

$$\lambda = \frac{\hat{\lambda}\mathbf{M}}{\hat{\lambda}\mathbf{M}\mathbf{e}}. \quad (21)$$

These coordinates are non-negative, because $\hat{\lambda}$ and \mathbf{M} are non-negative, they form a partition of unity, because $\lambda\mathbf{e} = 1$, and the reproduction property follows from Eqs. (20) and (21), because

$$\lambda\mathbf{C} = \frac{\hat{\lambda}\mathbf{M}\mathbf{C}}{\hat{\lambda}\mathbf{M}\mathbf{e}} = \frac{\hat{\lambda}\hat{\mathbf{C}} + \hat{\lambda}\mathbf{M}\mathbf{e}\mathbf{x}^T}{\hat{\lambda}\mathbf{M}\mathbf{e}} = \mathbf{x}^T.$$

This pre-processing step reduces the problem of finding GBC λ of a point \mathbf{x} inside an arbitrary cage \mathcal{C} to the problem of finding GBC $\hat{\lambda} = (\hat{\lambda}_1, \dots, \hat{\lambda}_{N_C})$ of the origin with respect to the vertices of a spherical cage $\hat{\mathcal{C}}$, and CHANG et al. [CDH23] propose to find the latter by maximizing the function

$$\ell(\hat{\lambda}) = \log \prod_{i=1}^{N_C} \hat{\lambda}_i = \sum_{i=1}^{N_C} \log \hat{\lambda}_i,$$

subject to the constraints

$$\sum_{i=1}^{N_C} \hat{\lambda}_i = 1, \quad \sum_{i=1}^{N_C} \hat{\lambda}_i \hat{\mathbf{c}}_i = \mathbf{x}.$$

Using Lagrangian multipliers, this optimization problem with N_C variables can be converted into an optimization problem with only $d = 3$ variables, namely finding the unique minimum of the strictly convex function

$$F(\eta) = - \sum_{i=1}^{N_C} \log(N_C + \eta^T(\hat{\mathbf{c}}_i - \mathbf{x})).$$

The global minimizer $\eta \in \mathbb{R}^3$ of F can be determined efficiently with Newton's method in a few iterations, and the barycentric coordinates of the origin with respect to $\hat{\mathcal{C}}$ are then defined as

$$\hat{\lambda}_i = \frac{1}{N_C + \eta^T(\hat{\mathbf{c}}_i - \mathbf{x})}.$$

These coordinates are non-negative by construction, and it remains to plug them into Eq. (21). The resulting MLC have all desired properties, and CHANG et al. [CDH23] further explain how the global minimizer η of F can be used to compute the derivatives of MLC at \mathbf{x} . However, if \mathbf{x} is close to a cage face, then it may happen that the spherical cage $\hat{\mathcal{C}}$ does not contain the origin, and then the procedure above does not give proper GBC of \mathbf{x} with respect to \mathcal{C} .

8. Coordinates with Normal Control

Another category of methods extends the notion of GBC by adding normal control. Although these methods are no longer interpolatory, they have the advantage of producing more realistic deformation. Both Green coordinates (Sections 8.1 and 8.2) and Somigliana coordinates (Section 8.3) introduced in this section are based on the boundary integral formulations of PDEs. Therefore, they inherit the simplicity of point-wise evaluation as MVC (see Section 5.1), while also incorporating smoothness and shape preservation of EMC (see Section 6), yet without the need for solving linear systems.

8.1. Green Coordinates

Green coordinates (GC), introduced by LIPMAN et al. [LLC08] for triangular cages, differ from all previously presented coordinates in the sense that the deformation function is defined as a blending of the cage vertices as well as the cage triangle normals. This allows inferring local shape rotations from pure cage vertex translations.

Those are based on Green's third identity that allows expressing any harmonic function $u(\mathbf{y}), \mathbf{y} \in \mathbb{R}^3$ inside a volumetric domain Ω from the diffusion of associated Dirichlet and Neumann conditions set on its boundary $\partial\Omega$:

$$u(\mathbf{x}) = \int_{\partial\Omega} u(\mathbf{y}) \frac{\partial_1 G(\mathbf{y}, \mathbf{x})}{\partial \mathbf{n}_y} d\mathbf{a}_y - \int_{\partial\Omega} G(\mathbf{y}, \mathbf{x}) \frac{\partial u(\mathbf{y})}{\partial \mathbf{n}_y} d\mathbf{a}_y, \quad (22)$$

where $d\mathbf{a}_y$ is the area element on $\partial\Omega$ and G is the fundamental solution to the Laplace equation in \mathbb{R}^3 (i.e., $\Delta_1 G(\mathbf{y}, \mathbf{x}) = \Delta_2 G(\mathbf{y}, \mathbf{x}) = \delta_0(\|\mathbf{x} - \mathbf{y}\|)$):

$$G(\mathbf{y}, \mathbf{x}) = G(\mathbf{x}, \mathbf{y}) = \frac{-1}{4\pi\|\mathbf{x} - \mathbf{y}\|},$$

$$\nabla_1 G(\mathbf{y}, \mathbf{x}) = -\nabla_2 G(\mathbf{y}, \mathbf{x}) = \frac{\mathbf{y} - \mathbf{x}}{4\pi\|\mathbf{y} - \mathbf{x}\|^3}.$$

Thus, we observe that those coordinates can only be expressed for cages that do not self-intersect, as they have to be expressed as “the surface boundary of a volumetric compact space”. Note that this derivation allows deriving deformation only for the interior of the domain. Though we do not detail it, an algorithm to derive GC for points outside the cage was presented by LIPMAN et al. [LLC08].

While it is straightforward to express the Dirichlet condition on a triangle t as

$$\mathbf{u}(\mathbf{y}) = \sum_{i \in t} \Gamma_i^t(\mathbf{y}) \mathbf{c}_i'$$

for deformed triangle corners \mathbf{c}_i' (corresponding to mapping a 3D flat triangle to a 3D flat triangle), it is unclear which Neumann condition should be set for the triangle t .

To compute the solution to the Laplace equation

$$\begin{aligned} \Delta u &= 0 \text{ in } \Omega, \\ u &\text{ given on } \partial\Omega, \end{aligned}$$

one should ensure that not only the Dirichlet condition but also the Neumann condition holds. Unfortunately, finding the Neumann condition associated with the Dirichlet condition is as difficult as solving the Laplace equation, and no analytical solution has ever been found for non-trivial domains. Note that setting up this Neumann condition would lead to analytical formulas reproducing the deformation behavior of HC (see Section 6.1), as HC discretize the Laplace equation.

Instead, LIPMAN et al. [LLC08] set a Neumann condition that leads to the completion of the 2D linear map inferred by the Dirichlet condition into a 3D linear map that best approximates a quasi-conformal map onto the triangle t , by mapping the input normal \mathbf{n}_t to the deformed normal \mathbf{n}_t' and accounting for the stretch factor of the 2D linear map $t \mapsto t'$:

$$\frac{\partial u(\mathbf{y})}{\partial \mathbf{n}_y} = \sigma_t \mathbf{n}_t', \quad \forall \mathbf{y} \in t.$$

Using two edges $(\mathbf{e}_1, \mathbf{e}_2)$ of the input triangle t , the stretch factor σ_t is computed as

$$\sigma_t = \sqrt{\frac{\|\mathbf{e}_1'\|^2 \|\mathbf{e}_2'\|^2 + \|\mathbf{e}_2'\|^2 \|\mathbf{e}_1'\|^2 - 2(\mathbf{e}_1 \cdot \mathbf{e}_2)(\mathbf{e}_1' \cdot \mathbf{e}_2')}{2\|\mathbf{e}_1 \times \mathbf{e}_2\|^2}}. \quad (23)$$

Setting this Neumann condition for all triangles leads empirically to 3D quasi-conformal deformation, which is, to the best of our knowledge, almost unique. We are not aware of any work other than Green coordinates allowing for non-trivial quasi-conformal 3D deformation obtained from closed-form expressions.

The final deformation formula reads:

$$\mathbf{x}' = \sum_{i=1}^{N_C} \phi_i(\mathbf{x}) \mathbf{c}_i' + \sum_{j=1}^{N_F} \psi_j(\mathbf{x}) \sigma_{t_j} \mathbf{n}_{t_j}', \quad (24)$$

with coordinates ϕ_i and ψ_j :

$$\phi_i^t(\mathbf{x}) = \int_t \Gamma_i^t(\mathbf{y}) \frac{\partial_1 G(\mathbf{y}, \mathbf{x})}{\partial \mathbf{n}_y} d\mathbf{y},$$

$$\phi_i(\mathbf{x}) = \sum_{i \in t} \phi_i^t(\mathbf{x}),$$

$$\psi_j(\mathbf{x}) = - \int_{t_j} G(\mathbf{y}, \mathbf{x}) d\mathbf{y}.$$

Computation LIPMAN et al. [LLC08] provide expressions for the integrals, which impose precision and efficiency issues. BEN-CHEN et al. [BWG09] noted that expressions for the computation

of GC are given by URAGO [Ura00], along with their gradients and Hessians. In their paper [BWG09], GC are used as a smooth harmonic subspace for as-rigid-as-possible shape deformation, by optimizing for least-squared deviations from positional constraints, rotational/similarity constraints on the medial axis and smoothness constraints. *We recommend using the expressions in their paper [BWG09] instead of the original computation scheme.*

8.2. Green Coordinates for Tri-Quad Cages

Green coordinates were recently extended to tri-quad cages by THIERY and BOUBEKEUR [TB22]. Their work borrows important ingredients from THIERY et al. [TMB18], such as

- the derivation of validity equations, ensuring exact reproduction of identity,
- the use of an optimized Riemann summation allowing for efficient approximation of the ground truth coordinates,
- the casting of the resulting approximation to the null-space of the validity equations.

As emphasized in Section 8.1, the key to the quasi-conformal property of GC is the suitable definition of the Dirichlet and Neumann conditions. For a non-planar quad q , the Dirichlet condition defined by the bilinear interpolant $(\mathbf{q}_{uv} \mapsto \mathbf{q}_{uv}')$ results in **i**) a non-constant stretch factor $\sigma_q(u, v)$ over the quad and **ii**) a non-constant face normal \mathbf{n}_{uv}' . Following the construction of GC leads to the following Neumann condition:

$$\frac{\partial u(\mathbf{q}_{uv})}{\partial \mathbf{n}_{uv}^q} = \sigma_q(u, v) \mathbf{n}_{uv}', \quad \forall (u, v) \in [0, 1]^2, \quad (25)$$

where $\sigma_q(u, v)$ is computed from Eq. (23), using the tangent vectors $(\partial_u \mathbf{q}_{uv}, \partial_v \mathbf{q}_{uv}, \partial_u \mathbf{q}_{uv}', \partial_v \mathbf{q}_{uv}')$ instead of the triangle edges $(\mathbf{e}_1, \mathbf{e}_2, \mathbf{e}_1', \mathbf{e}_2')$.

Applying these steps using the following contributions of q to the Dirichlet and Neumann conditions in combination with Eqs. (22) and (25) results in

$$\text{Dirichlet:} \quad u_D^q(\mathbf{x}) = \int_0^1 \int_0^1 \mathbf{q}_{uv}' \frac{(\mathbf{q}_{uv} - \mathbf{x}) \cdot \mathbf{n}_{uv}^q}{4\pi \|\mathbf{q}_{uv} - \mathbf{x}\|^3} du dv,$$

$$\text{Neumann:} \quad u_N^q(\mathbf{x}) = \int_0^1 \int_0^1 \frac{\sigma_q(u, v) \mathbf{n}_{uv}^q}{4\pi \|\mathbf{q}_{uv} - \mathbf{x}\|} \|\mathbf{n}_{uv}^q\| du dv.$$

This leads to integrals that seem unfortunately impossible to formulate as simple expressions of geometrical quantities.

THIERY and BOUBEKEUR [TB22] propose instead to deviate slightly from this definition and to use an auxiliary Neumann boundary condition $\sigma_q^{\text{aux}}(u, v)$:

$$\sigma_q^{\text{aux}}(u, v) = \frac{\|\mathbf{n}_{uv}'\|}{\|\mathbf{n}_{uv}^q\|}.$$

Introducing this quantity in $u_N^q(\mathbf{x})$ leads to

$$\begin{aligned} u_N^q(\mathbf{x}) &= \int_0^1 \int_0^1 \frac{\sigma_q(u, v) \mathbf{n}_{uv}^{q'}}{4\pi \|\mathbf{q}_{uv} - \mathbf{x}\|} \frac{\sigma_q^{\text{aux}}(u, v)}{\sigma_q^{\text{aux}}(u, v)} \|\mathbf{N}_{uv}^q\| du dv \\ &= \int_0^1 \int_0^1 \frac{\mathbf{N}_{uv}^{q'}}{4\pi \|\mathbf{q}_{uv} - \mathbf{x}\|} \frac{\sigma_q(u, v)}{\sigma_q^{\text{aux}}(u, v)} du dv \\ &= \sum_{k=0}^3 \int_0^1 \int_0^1 \frac{b_{uv}^k}{4\pi \|\mathbf{q}_{uv} - \mathbf{x}\|} \frac{\sigma_q(u, v)}{\sigma_q^{\text{aux}}(u, v)} du dv \mathbf{N}_k^{q'} \\ &\simeq \sum_{k=0}^3 \psi_q^k(\mathbf{x}) \sigma_q^k \mathbf{N}_k^{q'} \end{aligned} \quad (26)$$

with

$$\psi_q^k(\mathbf{x}) = \int_0^1 \int_0^1 \frac{b_{uv}^k}{4\pi \|\mathbf{q}_{uv} - \mathbf{x}\|} du dv, \quad (27)$$

appearing as a *quad-corner Neumann coordinate* and σ_q^k a *quad-corner stretch factor* approximating the ratio $\sigma_q(u, v)/\sigma_q^{\text{aux}}(u, v)$ over the quad, importance-sampled by b_{uv}^k .

Taking the contribution of q to the Dirichlet condition, one obtains

$$u_D^q(\mathbf{x}) = \int_0^1 \int_0^1 \mathbf{q}_{uv}' \frac{(\mathbf{q}_{uv} - \mathbf{x}) \cdot \mathbf{N}_{uv}^q}{4\pi \|\mathbf{q}_{uv} - \mathbf{x}\|^3} du dv = \sum_{k=0}^3 \phi_{qk}^q(\mathbf{x}) \mathbf{q}_k'$$

with

$$\phi_{qk}^q(\mathbf{x}) = \int_0^1 \int_0^1 b_{uv}^k \frac{(\mathbf{q}_{uv} - \mathbf{x}) \cdot \mathbf{N}_{uv}^q}{4\pi \|\mathbf{q}_{uv} - \mathbf{x}\|^3} du dv \quad (28)$$

appearing as a *quad corner Dirichlet coordinate*.

Accumulating the per-face Dirichlet coordinates together with the Neumann coordinates for triangular faces, the following deformation scheme is obtained:

$$\mathbf{x}' = \sum_{i=1}^{N_C} \phi_i(\mathbf{x}) \mathbf{c}_i' + \sum_{t \in T} \psi_t(\mathbf{x}) \sigma_t \mathbf{n}_t' + \sum_{q \in Q} \sum_{k=0}^3 \psi_q^k(\mathbf{x}) \sigma_q^k \mathbf{N}_k^{q'},$$

which extends the Green coordinates deformation function for triangular cages (cf. Eq. (24)).

Validity equations As disclaimed, closed-form expressions for the coordinates were not derived by THIERY and BOUBEKEUR [TB22], and a strategy similar to the computation of QMVC was used. Despite the approximation, QGC are guaranteed to be valid, as they are constrained to reproduce the identity.

To obtain such constraints, one can consider the contribution of a quad q to the reproduction of the identity equation. If a tessellation of q using N_{Tes} triangles $t_j = (\mathbf{t}_0^j, \mathbf{t}_1^j, \mathbf{t}_2^j) \in \mathbb{R}^{3 \times 3}$ is considered, the contributions of q and t_j to the reproduction of identity should match:

$$\sum_{k=0}^3 \phi_{qk}^q(\mathbf{x}) \mathbf{q}_k + \psi_q^k(\mathbf{x}) \mathbf{N}_k^q = \sum_{j=1}^{N_{\text{Tes}}} \sum_{k=0}^2 \phi_{t_k}^j(\mathbf{x}) \mathbf{t}_k^j + \sum_{j=1}^{N_{\text{Tes}}} \psi_j(\mathbf{x}) \mathbf{n}_{t_j}. \quad (29)$$

Considering the specific nature of the coefficients ϕ , one can see that summing them up leads to the solid angle $\omega_q(\mathbf{x})$ spanned by q from the point of view of \mathbf{x} , since

$$\frac{(\mathbf{q}_{uv} - \mathbf{x}) \cdot \mathbf{N}_{uv}^q}{\|\mathbf{q}_{uv} - \mathbf{x}\|^3} du dv = dS_{\mathbf{x}}(\mathbf{q}_{uv}) \quad (\text{see Eq. (10)})$$

establishes a link between GC and MVC through their respective kernels. This additional constraint therefore reads

$$\sum_{k=0}^3 \phi_{qk}^q(\mathbf{x}) = \sum_{j=1}^{N_{\text{Tes}}} \sum_{k=0}^2 \phi_{t_k}^{t_j}(\mathbf{x}) \left(= \frac{\omega_q(\mathbf{x})}{4\pi} \right). \quad (30)$$

Noting that $\Phi^q = (\phi_{q0}^q, \phi_{q1}^q, \phi_{q2}^q, \phi_{q3}^q, \psi_{q0}^q, \psi_{q1}^q, \psi_{q2}^q, \psi_{q3}^q)$ and using both validity Eqs. (29) and (30) leads to the following matrix expression constraint:

$$\mathbf{A}_q \Phi^q(\mathbf{x}) = \mathbf{m}_q(\mathbf{x}) \in \mathbb{R}^4, \quad (31)$$

$$\mathbf{A}_q = \begin{pmatrix} \mathbf{q}_0 & \mathbf{q}_1 & \mathbf{q}_2 & \mathbf{q}_3 & \mathbf{N}_0^q & \mathbf{N}_1^q & \mathbf{N}_2^q & \mathbf{N}_3^q \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \end{pmatrix} \in \mathbb{R}^{4 \times 8}.$$

As proven by THIERY and BOUBEKEUR [TB22], the null space of this equation is *always* 4D (as long as q is not degenerate), and the solution to Eq. (31) therefore takes (see Section 5.4) the form of the 4D “minimal-norm solution” (given by pseudo-inversion of Eq. (31)) and a combination of additional 4D vectors \mathbf{k}_i aligned with its (four-dimensional) kernel:

$$\Phi^q = \bar{\Phi}^q + \sum_{i=0}^3 \alpha_i \mathbf{k}_i, \quad \bar{\Phi}^q = \mathbf{A}_q^\dagger \mathbf{m}_q, \quad \mathbf{A}_q \mathbf{k}_i = 0.$$

Further following the strategy of THIERY et al. [TMB18] to compute smooth approximate QMVC, a *smooth and robust estimate* $\bar{\Phi}^q$ is computed using a Riemann summation approximating Eqs. (27) and (28).

Note on quasi-conformality Like the original GC, *quasi-conformality was not formally proven, but merely empirically observed*. Note that the approximation introduced by considering the ratio $\sigma_q(u, v)/\sigma_q^{\text{aux}}(u, v)$ as constant over the quad q (see Eq. (26)) makes QGC only an approximation that one would obtain by tessellating the bilinear quads (at the limit) and using the original GC approach. The approximation becomes formally null when all quads are scaled uniformly.

8.3. Somigliana Coordinates

CHEN et al. [CDD23] recently introduced Somigliana coordinates (SC) with normal control. SC draw inspiration from linear elasticity, specifically the Navier–Cauchy equation, which extends GC to better volume control. In contrast to the harmonic equation, linear elasticity measures both the material’s compressibility and the smoothness of deformation. This leads to a displacement field \mathbf{u} that satisfies

$$\mu \Delta \mathbf{u} + \frac{\mu}{1-2\nu} \nabla (\nabla \cdot \mathbf{u}) = \mathbf{b}, \quad (32)$$

where $\mu > 0$ is the shear modulus, ν is the Poisson ratio, and \mathbf{b} is the external body load. If no external load is exerted ($\mathbf{b} = \mathbf{0}$), the displacement field within the domain is entirely determined by the boundary conditions, including the boundary displacement \mathbf{u} and the associated traction $\boldsymbol{\tau}$. This relationship is given by the *Somigliana identity* [Som85] written as

$$\mathbf{u}(\mathbf{x}) = \int_{\partial\Omega} [\mathcal{T}(\mathbf{y}, \mathbf{x}) \mathbf{u}(\mathbf{y}) + \mathcal{K}(\mathbf{y}, \mathbf{x}) \boldsymbol{\tau}(\mathbf{x})] d\mathbf{a}_{\mathbf{y}}, \quad \forall \mathbf{x} \in \Omega. \quad (33)$$

Table 2: Comparison of cage coordinate types ordered by category (top to bottom: GBC with explicit formulas, EMC, probability-based coordinates, and coordinates with normal control) and year of publication.

coordinates	Lagrange property	pointwise evaluation	face type		extrapolation	comment
			triangle	quad		
MVC [FKR05]	✓	✓	✓	✗	✓	Simple and efficient
SBC [LBS06]	✓	✓	✓	planar	✓	Support for planar n -gon faces
PMVC [LKCL07]	✓	✓	✓	✗	✗	Non-negative MVC on the GPU
QMVC [TMB18]	✓	✓	✓	✓	✗	Use of tri-quad cages
HC [JMD*07]	✓	✗	✓	planar	✗	Local deformation influence
BBW [JBPS11]	✓	✗	✓	✗	✗	Enables the joint use of cages, skeletons, and point handles
LBC [ZDL*14]	✓	✗	✓	✗	✗	Enables to control locality
MEC [HS08]	✓	✓	✓	✓	✗	Direct evaluation for arbitrary polygon-cages
MLC [CDH23]	✓	✓	✓	✗	✗	Better shape awareness than MEC
GC [LLC08]	✗	✓	✓	✗	✓	Conformal mapping
QGC [TB22]	✗	✓	✓	✓	✓	Symmetry preserving
SC [CDD23]	✗	✓	✓	✗	✓	Volume control

In fact, this equation is a generalization of Green's third identity (see Eq. (22)) to linear elasticity. Here, \mathcal{K} represents the fundamental solutions of linear elasticity, also known as the Kelvin solution, and \mathcal{T} corresponds to the boundary traction derived from it. The expressions for \mathcal{K} and \mathcal{T} [CDD23] contain terms resembling, respectively, the fundamental solution of the Laplacian equation and its normal derivative, but they are further combined with extra terms that control volume compression. The strength of this control is determined by the Poisson ratio ν .

Much like the derivation of GC, SC are obtained by recognizing that the identity mapping $\mathbf{u}(\mathbf{x}) = \mathbf{x}$ also satisfies Eq. (32). When substituting the mapping to the boundary integral Eq. (33), it yields a reproduction of the rest pose:

$$\mathbf{x} = \sum_{i=1}^{N_C} \mathbf{T}_i(\mathbf{x}) \mathbf{c}_i + \sum_{j=1}^{N_F} \mathbf{K}_j(\mathbf{x}) (c \mathbf{n}_j), \quad (34)$$

where the constant $c = 2\mu(1+\nu)/(1-2\nu)$ is the magnitude of the boundary traction induced by $\mathbf{u}(\mathbf{x}) = \mathbf{x}$, and \mathbf{T}_i , \mathbf{K}_{t_i} are the *matrix-valued* SC respectively defined for cage vertices and facets, computed through

$$\mathbf{T}_i(\mathbf{x}) = \int_{\partial\Omega} \mathcal{T}(\mathbf{y}, \mathbf{x}) \Gamma_i(\mathbf{y}) d\sigma_{\mathbf{y}},$$

$$\mathbf{K}_{t_i}(\mathbf{x}) = \int_{\partial\Omega} \mathcal{K}(\mathbf{y}, \mathbf{x}) \Pi_{t_i}(\mathbf{y}) d\sigma_{\mathbf{y}},$$

where Γ_i is the piecewise linear basis function for the cage vertex and Π_{t_i} is piecewise constant across the cage face. Since deriving the closed-form expressions for the above integrals is technically

challenging, CHEN et al. [CDD23] use quadrature rules to compute these integrals over cage triangles. As a result, computation of SC can be massively parallelized, enabling effective use of the computational power of GPUs.

Corotational formulation A significant distinction between matrix-valued SC and scalar coordinates lies in their tensorial nature, which renders them dependent on the global orientation and scaling of the undeformed cage. Consequently, they cannot reproduce similarity transformations, inheriting the limitations of linear elasticity. This flaw would cause significant volume inflation artifacts when dealing with large deformations. To address this issue, CHEN et al. [CDD23] further proposed an improvement to SC using a corotational formulation to achieve similarity invariance. Specifically, each cage facet is associated with a rotation matrix \mathbf{R}_{t_i} representing a local frame. From face rotations, a nodal rotation \mathbf{R}_i can be computed by averaging the rotations of its neighboring faces. As a result, the deformed mesh \mathcal{T}' is interpolated using a corotational extension of Eq. (34):

$$\mathbf{v}' = \mathbf{T}(\mathbf{v})^{-1} \left[\sum_{i=1}^{N_C} \mathbf{R}_i \mathbf{T}_i(\mathbf{v}) \mathbf{R}_i^T \mathbf{c}'_i + \sum_{j=1}^{N_F} \mathbf{R}_{t_j} \mathbf{K}_{t_j}(\mathbf{v}) \mathbf{R}_{t_j}^T \boldsymbol{\tau}'_{t_j} \right] \quad (35)$$

with $\mathbf{T}(\mathbf{v}) = \sum_{i=1}^{N_C} \mathbf{R}_i \mathbf{T}_i(\mathbf{v}) \mathbf{R}_i^T$. Here, $\boldsymbol{\tau}'_{t_i}$ is the Neumann boundary condition for the *corotated* traction. As shown by CHEN et al. [CDD23], this traction simplifies to a scaled and rotated rest normal, i.e., $\boldsymbol{\tau}'_{t_i} = s_{t_i} \mathbf{R}_{t_i} \mathbf{n}_{t_i}$, with the elastic material coefficients and strain stretches being factored out into a single scalar s_{t_i} for legibility.

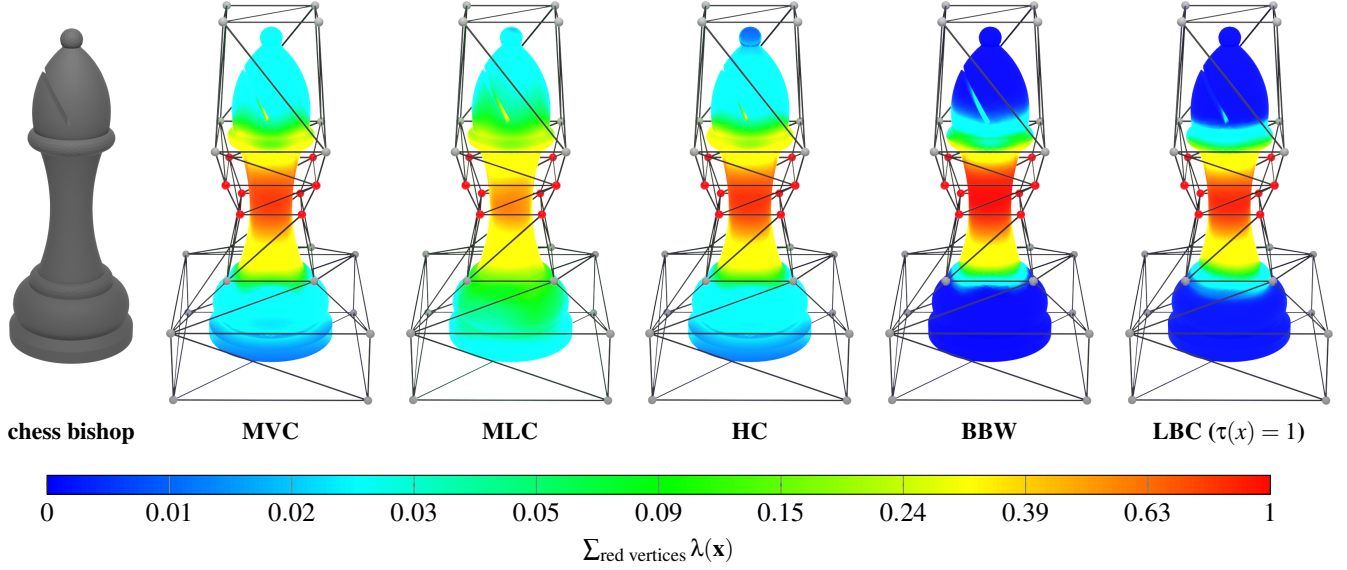


Figure 2: Visualization of the local influence of red control vertices for different coordinate types with a logarithmically scaled color map.

According to Eq. (35), the final deformation is determined by both cage positions and the choice of s_{t_i} and \mathbf{R}_{t_i} . This selection is crucial for the appearance and behavior of the cage deformer. Among the available options, CHEN et al. [CDD23] recommended two representative variants which offer complementary benefits. The global approach derives scaling and rotation using a registration process that returns the optimal similarity transformation aligning the rest cage with the deformed cage. In this approach, all faces share the same rotation and scaling values, resulting in deformation that resembles more computationally expensive boundary element methods and achieving a higher level of realism. In contrast, the local approach estimates rotation and scaling on a per-face basis. This approach typically tends to produce more artistic and creatively exaggerated results, featuring localized deformation. While both variants incorporate volume control to account for the material compressibility, they also contain a set of scaling factors based on solid angles in the estimation of each s_{t_i} . These scaling factors enable real-time generation of localized bulges, adding a level of detail that respects the curvature changes of the cage and thus complementing the volume control. These variants offer great flexibility for interactive shape editing, allowing for intuitive adjustments and fine-tuning of deformation.

9. Comparing Coordinate Types

As the various coordinate types possess different advantages and disadvantages, we provide a comparison. Table 2 presents a comparative overview of the presented coordinate types.

9.1. Locality

Since the influence of each cage vertex should be local for intuitive deformation control, our discussion involves the locality of coordinate types. Prototypical for our experiments, Fig. 2 plots the

local influence of various GBC types for a selected set of control vertices. Generally, all the coordinate types in this report provide local deformation but the degree of the locality highly varies across coordinate types.

MVC provide high influence near the control vertices, while the GBC functions decay only slowly in the more distant regions. The probability-based coordinates such as MLC are less local and decay slower than MVC. EMC (see Section 6) exhibit the most local deformation control. HC are more local than MVC. A significant increase of locality is achieved using BBW for cage-based deformation. The use of LBC (with $\tau(x) = 1$) inherently provides the most local set of GBC. Our experiments confirm that even more local GBC are obtained by setting τ to a monotonically increasing function. The locality of coordinates with normal control is more difficult to plot, because face normals and stretch factors also influence the locality of deformation. Typically, these coordinates provide less locality than the other coordinate types.

9.2. Cage Connectivity

As quad-layouts are more common in industry, many applications use quads for cages. Consequently, a cage needs to be triangulated, if the coordinate type only supports triangles. However, deforming a model with a triangulated cage can lead to artifacts (see Fig. 3). Instead of triangulating cage polygons, one should use suitable coordinates to avoid artifacts in the deformed geometry.

The probability-based coordinate types conceptionally support pointwise evaluation for arbitrary polygon cages, while MLC has only been evaluated for triangle cages up to now. Local deformation for planar polygon cages can be obtained with the use of HC, though the numerical solver of use needs to provide an implementation for the polygon type of the cage. Considering that the most commonly used solvers for HC are implemented for trian-

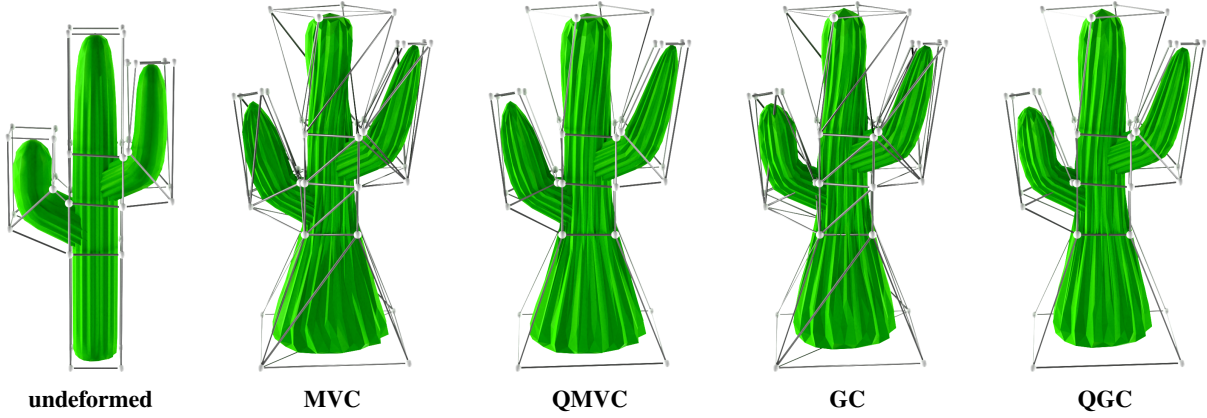


Figure 3: Cage triangulation can lead to artifacts. The use of tri-quad enabled coordinates such as QGC resolves this issue.

gular cages, users cannot easily benefit from the support of arbitrary planar polygons. Users can easily deform models with quad or tri-quad cages using QMVC or QGC and benefit from simple and efficient pointwise evaluation. An advantage of QGC is its extrapolation capabilities, which enable users to only wrap the model parts of interest with cages while each cage still deforms the exterior parts.

9.3. Shape Preservation

As users typically wish to preserve surface features, cage-based deformation should avoid significant distortion of surface features. However, large deformation can lead to unintended shape distortion. Thus, we discuss the ability of different coordinate types to preserve the input shape under large deformation. Figure 4 presents the resulting geometries by substantially deforming an Ogre model.

The application of simple MVC to large deformation can impose sharp artifacts, because MVC is prone to inflating the volume more at the parts near the cage vertices. This can lead to asymmetric shapes such as the chest of the Ogre and sharp bumps such as the distortion at the Ogre's wrist and shoulder. Since the locality of MVC is governed by the euclidean distance between a point $\mathbf{x} \in \Omega$ and a cage vertex, the occurrence of these artifacts is difficult to avoid. Due to their increased locality, the EMC suffer from the same problem. Significant artifacts are observed with the use of BBW, whereas one can easily preserve certain areas by incorporating an additional energy term, e.g., preserving the Ogre's right wrist. Among the EMC, the application of LBC seems to most reliably prevent artifacts, because their calculation effectively minimizes total variation and the locality is controlled by geodesic distances.

Deformation using MLC provides good shape awareness in the interior of the cage, which results in good volume preservation. For instance, see the Ogre's right arm after deformation with MLC. However, the probability-based GBC also incur artifacts especially at the regions close to the cage boundary. The best feature and volume preservation is provided by the cage coordinates with normal control, because they deform the model considering the shape and

scaling of cage faces besides the cage vertex positions. As a result, coordinates with normal control better avoid the introduction of asymmetries and sharp features (see the Ogre's chest and shoulder). While GC provide good feature preservation in general, improved control of volume and shape preservation can be achieved using SC. Cage deformation using SC can restrict the volume inflation of the model (see the Ogre's right hand). The only drawback of coordinates with normal control is that after large deformation the cage potentially intersects the model. In such a situation, the influence of the cage on the model is not as intuitive and the user needs to gauge which cage vertices to relocate to achieve the intended deformation.

9.4. Computational Complexity

The coordinate types for cage-based deformation differ in their computational overheads at bind time. The EMC impose the most pre-calculation demands. One needs to obtain a sufficient embedding \mathcal{E}_C (see Section 4.3) to initiate the calculation of coordinates. Additionally, a numerical scheme is required to calculate the coordinates, which typically cannot be trivially parallelized. As a result, the calculation of high quality EMC such as BBW or LBC can impose low run time performance. For instance, the calculation of BBW or LBC for an \mathcal{E}_C with over one million tetrahedra took several days on an Intel i9-11900K CPU with 64 GB of RAM in our experiments. Thus, it is advised to offload the calculation of EMC on a remote machine and save the coordinates for later use.

The other coordinate types can be calculated efficiently for high-resolution models in a couple of seconds or even milliseconds. The actual run time performance depends on the implementation. Especially, the GBC with explicit formulas and the coordinates with normal control admit efficient parallelization on massively parallel GPUs. The probability-based coordinates require an additional numerical optimization step that can typically be performed efficiently.

9.5. Summary of Coordinate Type Comparison

The barycentric coordinates with explicit formulas offer a simple construction that is easy to apply to various use cases, whereas

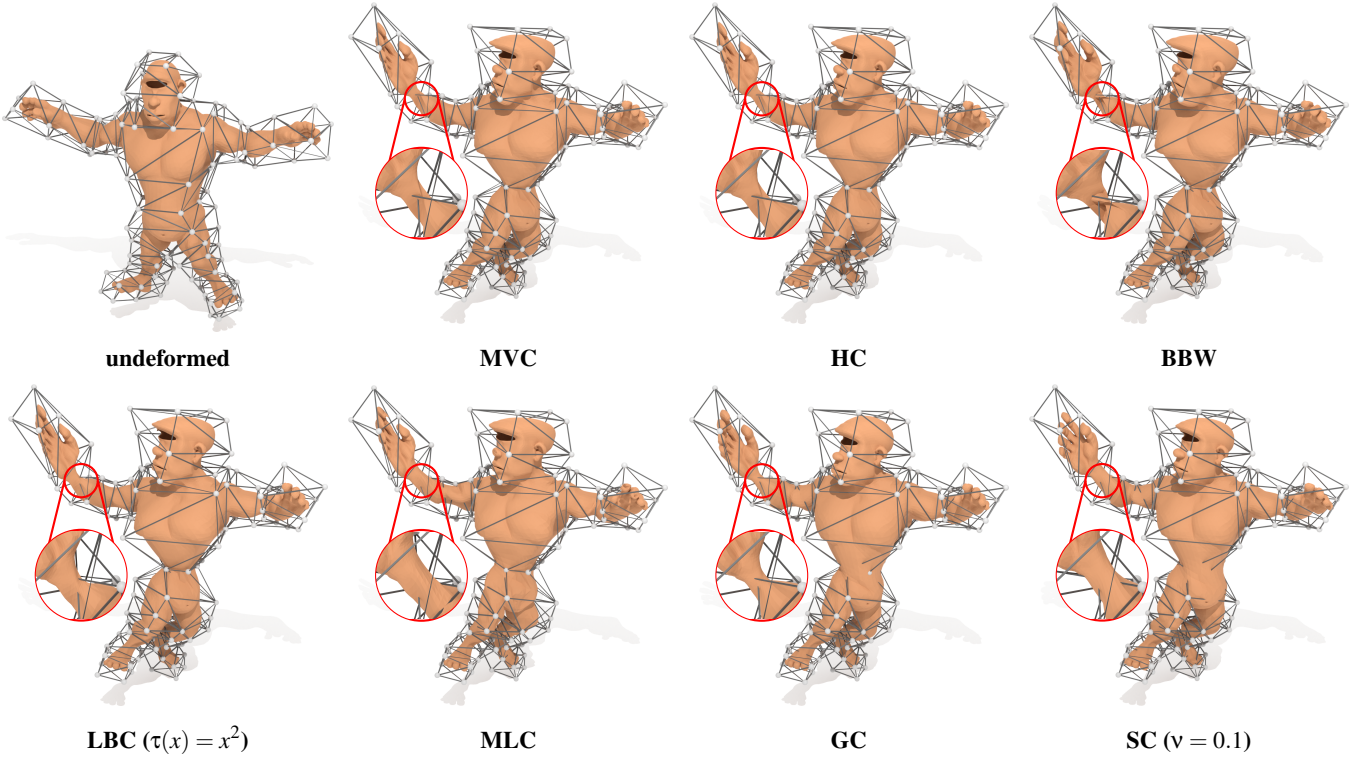


Figure 4: Comparison of the results of large deformation of an Ogre model using different coordinate types.

these coordinates impose several limitations. Many of these limitations have been overcome by other coordinate types. The EMC offer improved locality for deformation control and should be used for applications where small deformation is needed to adjust local details while preserving the global shape. Additionally, suitable applications for EMC are tolerant to long pre-calculation times. For quick and shape preserving modeling applications, the coordinates with normal control are most shape-preserving under large deformation. The probability-based coordinates offer pointwise evaluation for topologically arbitrary cages, while MLC have only been evaluated for triangle cages up to now. However, they do not offer the locality of EMC or the shape preservation of coordinates with normal control. We will publish source code to unify all the current cage coordinate types in one framework.

10. Deforming the Model

After a set of GBC is determined, users can relocate cage vertices to deform the model at interactive rates. The deformation of the model then either uses Eq. (2), Eq. (3), or a specific scheme involving polygon normals (see Section 8). Each of these deformation schemes can easily be parallelized for GPUs to achieve real-time performance for deforming high-resolution models. Over the years, academics devised cage-based deformation systems to support important use cases and various coordinate types.

*Cages is a hierarchical multilevel cage-based deformation system by GONZÁLEZ GARCÍA et al. [GPCP13] that allows the com-

bination of coordinate types across a single cage. In *Cages, the union of non-intersecting small leaf-cages span the enclosing cage for deformation control. Each leaf-cage can use its own coordinate type independent of other leaf cages. In between adjacent leaf cages *Cages C^1 -blends the coordinates to produce a smooth deformation, which allows using multiple leaf-cages for one deformation. As a result, deformation control is localized for a user-defined subpart of the cage. This improves run times and memory consumption.

CASTI et al. [CCLS18] provide the extendable CageLab tool to the academic community. CageLab enables users to visually evaluate cage quality and the local influence of a specific type of GBC. Using CageLab, the user is able to load a set of GBC or calculate different coordinate types. At the time of writing, CageLab supported GC and MVC. To allow for character animation, CageLab features an FBX file [Aut24] (3D scene format) importer and a key-framing system.

As cages are complementary to skeletons, CORDA et al. [CTL*20] present a deformation system to couple cages with skeletons. The system supports typical linear cage-based deformation (see Eq. (2)). After the determination of GBC, barycentric coordinates for the skeleton-joints are calculated using cage-coordinates. Whenever the user deforms the skeleton, the system produces \mathcal{T}' using LBS and finds new cage vertex positions \hat{C}' by solving a linear system:

$$\mathbf{W}\hat{C}' = \mathbf{V}'.$$

Whenever the user deforms the cage, instead of using Eq. (2) to deform the model, the system reflects the changes of C' to the rest cage C , such that the barycentric coordinates for the skeleton produce T' . This concept extends to non-linear coordinate types such as GC, though it requires more complex algorithms for coupling the cage with the skeleton.

11. Conclusion

In summary, we have presented the current state of the art of cage-based deformation in a systematic way, addressing advantages and disadvantages. We have contributed a report that comprehensively presents and reviews cage-based deformation. With this report, we have organized the cage generation methods and coordinate types into categories. Our categorization bundles approaches with commonalities to provide a comparative overview reflecting the most relevant advantages and disadvantages for practical use. As a result, our report facilitates the identification of suitable approaches for specific use cases and illuminates ongoing challenges. Our application can be retrieved under <https://github.com/DanStroeter/CageModeler.git>.

11.1. Future Research Directions

Due to the many recent advancements, future research potentially enables leveraging cage-based deformation in important applications. As recent advances enable efficient point-wise evaluation for quad and tri-quad cages, mesh modeling applications such as shape design or virtual prototyping might be able to benefit from deformation by cage control. For VR/AR, the use of interactive cage-based deformation allows the modeling of geometry within a virtual environment. Interactive cage generation for VR/AR is a potentially interesting research direction.

In the light of machine learning, cage-based deformation is an approach to span a coordinate system, which is amenable to neural networks and enables the application of machine learning to geometric data and its deformation. Our evaluation shows that the use of the newer coordinate types provides crucial advantages, whereas current work, e.g. [YAK*20; PYL*22; XH22], on coupling neural networks with cage-based deformation use the earlier coordinate types. As machine learning methods typically rely on extensive data sets, cage-based deformation can generate variations of existing models to improve coverage of training data. In addition, neural networks may be used to parameterize a set of GBC such as recently proposed by DODIK et al. [DSSS23]. Up to now, the advanced cage construction methods and coordinate types have been only sparsely used in the field of machine learning. We hope that our report will motivate future efforts to extend the field of cage-based deformation.

Acknowledgements

Q. Chang was supported by the Swiss National Science Foundation (SNF) under project number 188577. J. Chen was partially supported by an INRIA chair (Desbrun/GeomeriX). S. Besler was supported by the Fraunhofer Cluster of Excellence Programmable Materials (CPM). We thank the anonymous reviewers for their valu-

able comments, which helped us to improve the quality of our paper.

References

- [AA00] ANDERSEN, ERLING D. and ANDERSEN, KNUD D. “The MOSEK interior point optimizer for linear programming: An implementation of the homogeneous algorithm”. *High Performance Optimization*. Ed. by FRENK, HANS, ROOS, KEES, TERLAKY, TAMÁS, and ZHANG, SHUZHONG. Vol. 33. Applied Optimization. Boston: Springer, 2000, 197–232. DOI: [10.1007/978-1-4757-3216-0_8](https://doi.org/10.1007/978-1-4757-3216-0_8) 12.
- [AAN12] ANDERSON, GEORGE, AFTOSMIS, MICHAEL, and NEMEC, MARIAN. “Parametric deformation of discrete geometry for aerodynamic shape design”. *Proceedings of the 50th AIAA Aerospace Sciences Meeting including the New Horizons Forum and Aerospace Exposition*. Nashville, Jan. 2012, Article 965, 18 pages. DOI: [10.2514/6.2012-965](https://doi.org/10.2514/6.2012-965) 1.
- [AO06] ARROYO, MARINO and ORTIZ, MICHAEL. “Local maximum-entropy approximation schemes: A seamless bridge between finite elements and meshfree methods”. *International Journal for Numerical Methods in Engineering* 65.13 (Mar. 2006), 2167–2202. DOI: [10.1002/nme.1534](https://doi.org/10.1002/nme.1534) 2.
- [APH17] ANISIMOV, DMITRY, PANOZZO, DANIELE, and HORMANN, KAI. “Blended barycentric coordinates”. *Computer Aided Geometric Design* 52–53 (Mar. 2017), 205–216. DOI: [10.1016/j.cagd.2017.02.007](https://doi.org/10.1016/j.cagd.2017.02.007) 2.
- [Aut24] AUTODESK. *Autodesk FBX SDK*. 2024. URL: <https://www.autodesk.com/developer-network/platform-technologies/fbx-sdk-2020-2-1> (visited on 01/31/2024) 20.
- [Bel06] BELYAEV, ALEXANDER. “On transfinite barycentric coordinates”. *Proceedings of the 4th Symposium on Geometry Processing*. SGP '06. Cagliari, June 2006, 89–99. DOI: [10.2312/SGP/SGP06/089-099](https://doi.org/10.2312/SGP/SGP06/089-099) 2, 9.
- [BK04] BOTSCH, MARIO and KOBELT, LEIF. “An intuitive framework for real-time freeform modeling”. *ACM Transactions on Graphics* 23.3 (Aug. 2004), 630–634. DOI: [10.1145/1015706.1015772](https://doi.org/10.1145/1015706.1015772) 4.
- [BLTD16] BUDNINSKIY, MAX, LIU, BEIBEI, TONG, YIYING, and DESBRUN, MATTHIEU. “Power coordinates: A geometric construction of barycentric coordinates on convex polytopes”. *ACM Transactions on Graphics* 35.6 (Nov. 2016), Article 241, 11 pages. DOI: [10.1145/2980179.2982441](https://doi.org/10.1145/2980179.2982441) 2.
- [Boy10] BOYD, STEPHEN. “Distributed optimization and statistical learning via the alternating direction method of multipliers”. *Foundations and Trends® in Machine Learning* 3.1 (July 2010), 1–122. DOI: [10.1561/2200000016](https://doi.org/10.1561/2200000016) 13.
- [BWG09] BEN-CHEN, MIRELA, WEBER, OFIR, and GOTSMAN, CRAIG. “Spatial deformation transfer”. *Proceedings of the Symposium on Computer Animation*. SCA '09. New Orleans, Aug. 2009, 67–74. DOI: [10.1145/1599470.1599479](https://doi.org/10.1145/1599470.1599479) 2, 5, 7, 15.
- [CB17] CALDERON, STÉPHANE and BOUBEKEUR, TAMY. “Bounding proxies for shape approximation”. *ACM Transactions on Graphics* 36.4 (July 2017), Article 57, 13 pages. DOI: [10.1145/3072959.3073714](https://doi.org/10.1145/3072959.3073714) 6–8.
- [CCLS18] CASTI, SARA, CORDA, FABRIZIO, LIVESU, MARCO, and SCATENI, RICCARDO. “CageLab: An interactive tool for cage-based deformations”. *Proceedings of the Italian Chapter Conference on Smart Tools and Apps for Graphics*. STAG '18. Brescia, 2018, 65–74. DOI: [10.2312/stag.20181299](https://doi.org/10.2312/stag.20181299) 20.
- [CDD23] CHEN, JIONG, DE GOES, FERNANDO, and DESBRUN, MATTHIEU. “Somigliana coordinates: An elasticity-derived approach for cage deformation”. *SIGGRAPH 2023 Conference Proceedings*. Los Angeles, July 2023, Article 52, 8 pages. DOI: [10.1145/3588432.3591519](https://doi.org/10.1145/3588432.3591519) 1, 16–18.

- [CDH23] CHANG, QINGJUN, DENG, CHONGYANG, and HORMANN, KAI. “Maximum likelihood coordinates”. *Computer Graphics Forum* 42.5 (Aug. 2023), Article e14908, 13 pages. DOI: [10.1111/cgf.14908](https://doi.org/10.1111/cgf.14908) 1, 2, 13, 14, 17.
- [CDS12] CHENG, SIU-WING, DEY, TAMAL K., and SHEWCHUK, JONATHAN. “Tetrahedral meshing of PLCs”. *Delaunay Mesh Generation*. Ed. by CHENG, SIU-WING, DEY, TAMAL K., and SHEWCHUK, JONATHAN. Boca Raton: Chapman and Hall/CRC, 2012. Chap. 8, 185–204. DOI: [10.1201/b12987-10.6](https://doi.org/10.1201/b12987-10.6).
- [CF14] CHEN, XUE and FENG, JIEQING. “Adaptive skeleton-driven cages for mesh sequences”. *Computer Animation & Virtual Worlds* 25.3–4 (May 2014), 445–453. DOI: [10.1002/cav.1577](https://doi.org/10.1002/cav.1577) 6–8.
- [CHP89] CHADWICK, JOHN E., HAUMANN, DAVID R., and PARENT, RICHARD E. “Layered construction for deformable animated characters”. *Proceedings of the 16th Annual Conference on Computer Graphics and Interactive Techniques*. SIGGRAPH ’89. Boston, July 1989, 243–252. DOI: [10.1145/74333.743584](https://doi.org/10.1145/74333.743584).
- [CLM*19] CASTI, SARA, LIVESU, MARCO, MELLADO, NICOLAS, et al. “Skeleton based cage generation guided by harmonic fields”. *Computers & Graphics* 81 (June 2019), 140–151. DOI: [10.1016/j.cag.2019.04.004](https://doi.org/10.1016/j.cag.2019.04.004) 6–8.
- [CMT*12] COROS, STELIAN, MARTIN, SEBASTIAN, THOMASZEWSKI, BERNHARD, et al. “Deformable objects alive!”. *ACM Transactions on Graphics* 31.4 (Aug. 2012), Article 69, 9 pages. DOI: [10.1145/2185520.2185565](https://doi.org/10.1145/2185520.2185565) 1.
- [CTL*20] CORDA, FABRIZIO, THIERY, JEAN-MARC, LIVESU, MARCO, et al. “Real-time deformation with coupled cages and skeletons”. *Computer Graphics Forum* 39.6 (Jan. 2020), 19–32. DOI: [10.1111/cgf.13900](https://doi.org/10.1111/cgf.13900) 4, 20.
- [CV01] CHAN, TONY F. and VESE, LUMINITA A. “Active contours without edges”. *IEEE Transactions on Image Processing* 10.2 (Feb. 2001), 266–277. DOI: [10.1109/83.902291](https://doi.org/10.1109/83.902291) 12.
- [dBvdSB07] De BOER, AUKJE, van der SCHOOT, MARTIJN S., and BIJL, HESTER. “Mesh deformation based on radial basis function interpolation”. *Computers & Structures* 85.11–14 (June 2007), 784–795. DOI: [10.1016/j.compstruc.2007.01.013](https://doi.org/10.1016/j.compstruc.2007.01.013) 4.
- [DCH20] DENG, CHONGYANG, CHANG, QINGJUN, and HORMANN, KAI. “Iterative coordinates”. *Computer Aided Geometric Design* 79 (May 2020), Article 101861, 13 pages. DOI: [10.1016/j.cagd.2020.101861](https://doi.org/10.1016/j.cagd.2020.101861) 2, 14.
- [DJS16] DAVIA-ARACIL, MIGUEL, JIMENO-MORENILLA, ANTONIO, and SALAS, FAUSTINO. “A new methodological approach for shoe sole design and validation”. *The International Journal of Advanced Manufacturing Technology* 86.9–12 (Oct. 2016), 3495–3516. DOI: [10.1007/s00170-016-8427-5](https://doi.org/10.1007/s00170-016-8427-5) 1.
- [DKKR06] DIMITROV, DARKO, KNAUER, CHRISTIAN, KRIEGLER, KLAUS, and ROTE, GÜNTER. “On the bounding boxes obtained by principal component analysis”. *Proceedings of the 22nd European Workshop on Computational Geometry*. EWCG ’06. Delphi, Mar. 2006, 193–196 5.
- [DLM11] DENG, ZHENG-JIE, LUO, XIAO-NAN, and MIAO, XIAO-PING. “Automatic cage building with quadric error metrics”. *Journal of Computer Science and Technology* 26.3 (May 2011), 538–547. DOI: [10.1007/s11390-011-1153-4](https://doi.org/10.1007/s11390-011-1153-4) 5, 7, 8.
- [DMA02] DESBRUN, MATHIEU, MEYER, MARK, and ALLIEZ, PIERRE. “Intrinsic parameterizations of surface meshes”. *Computer Graphics Forum* 21.3 (Sept. 2002), 209–218. DOI: [10.1111/1467-8659.00580](https://doi.org/10.1111/1467-8659.00580) 2.
- [DSSS23] DODIK, ANA, STEIN, ODED, SITZMANN, VINCENT, and SOLOMON, JUSTIN. “Variational barycentric coordinates”. *ACM Transactions on Graphics* 42.6 (Dec. 2023), Article 255, 16 pages. DOI: [10.1145/3618403](https://doi.org/10.1145/3618403) 21.
- [Duf59] DUFFIN, RICHARD J. “Distributed and lumped networks”. *Journal of Mathematics and Mechanics* 8.5 (1959), 793–926. DOI: [10.1512/IUMJ.1959.8.58051](https://doi.org/10.1512/IUMJ.1959.8.58051) 8.
- [EDD*95] ECK, MATTHIAS, DEROSE, TONY, DUCHAMP, TOM, et al. “Multiresolution analysis of arbitrary meshes”. *Proceedings of the 22nd Annual Conference on Computer Graphics and Interactive Techniques*. SIGGRAPH ’95. Los Angeles, Sept. 1995, 173–182. DOI: [10.1145/218380.218440](https://doi.org/10.1145/218380.218440) 2, 8.
- [EG15] EVANS, LAWRENCE C. and GARIEPY, RONALD F. *Measure Theory and Fine Properties of Functions*. New York: Chapman and Hall/CRC, 2015. DOI: [10.1201/b18333](https://doi.org/10.1201/b18333) 12.
- [ETA02] ELAD, MICHAEL, TAL, AYELET, and AR, SIGAL. “Content based retrieval of VRML objects — An iterative and interactive approach”. *Multimedia 2001*. Ed. by JORGE, JOAQUIM, CORREIA, NUNO, JONES, HUW, and BLATTNER KAMEGAI, MEERA. Vienna: Springer, 2002, 107–118. DOI: [10.1007/978-3-7091-6103-6_12](https://doi.org/10.1007/978-3-7091-6103-6_12) 5.
- [FHK06] FLOATER, MICHAEL S., HORMANN, KAI, and KÓS, GÉZA. “A general construction of barycentric coordinates over convex polygons”. *Advances in Computational Mathematics* 24.1–4 (Jan. 2006), 311–331. DOI: [10.1007/s10444-004-7611-6](https://doi.org/10.1007/s10444-004-7611-6) 8, 11.
- [FHL*09] FARBMAN, ZEEV, HOFFER, GIL, LIPMAN, YARON, et al. “Coordinates for instant image cloning”. *ACM Transactions on Graphics* 28.3 (Aug. 2009), Article 67, 9 pages. DOI: [10.1145/1576246.1531373](https://doi.org/10.1145/1576246.1531373) 2.
- [FK98] FAIRWEATHER, GRAEME and KARAGEORGHIS, ANDREAS. “The method of fundamental solutions for elliptic boundary value problems”. *Advances in Computational Mathematics* 9.1–2 (Sept. 1998), 69–95. DOI: [10.1023/A:1018981221740](https://doi.org/10.1023/A:1018981221740) 11.
- [FKR05] FLOATER, MICHAEL S., KÓS, GÉZA, and REIMERS, MARTIN. “Mean value coordinates in 3D”. *Computer Aided Geometric Design* 22.7 (Oct. 2005), 623–631. DOI: [10.1016/j.cagd.2005.06.004](https://doi.org/10.1016/j.cagd.2005.06.004) 8, 9, 17.
- [Flo03] FLOATER, MICHAEL S. “Mean value coordinates”. *Computer Aided Geometric Design* 20.1 (Mar. 2003), 19–27. DOI: [10.1016/S0167-8396\(03\)00002-5](https://doi.org/10.1016/S0167-8396(03)00002-5) 2, 8.
- [Flo15] FLOATER, MICHAEL S. “Generalized barycentric coordinates and applications”. *Acta Numerica* 24 (Apr. 2015), 161–214. DOI: [10.1017/s0962492914000129](https://doi.org/10.1017/s0962492914000129) 2.
- [Flo97] FLOATER, MICHAEL S. “Parameterization and smooth approximation of surface triangulations”. *Computer Aided Geometric Design* 14.3 (Apr. 1997), 231–250. DOI: [10.1016/S0167-8396\(96\)00031-3](https://doi.org/10.1016/S0167-8396(96)00031-3) 2.
- [GP89] GRIESSMAIR, JOSEF and PURGATHOFER, WERNER. “Deformation of solids with trivariate B-splines”. *Eurographics ’89 Conference Proceedings*. Hamburg, Sept. 1989, 137–148. DOI: [10.2312/egtp.19891010](https://doi.org/10.2312/egtp.19891010) 4.
- [GPCP13] GONZÁLEZ GARCÍA, FRANCISCO, PARADINAS, TERESA, COLL, NARCÍS, and PATOW, GUSTAVO. “Cages: A multilevel, multi-cage-based system for mesh deformation”. *ACM Transactions on Graphics* 32.3 (June 2013), Article 24, 13 pages. DOI: [10.1145/2487228.2487232](https://doi.org/10.1145/2487228.2487232) 20.
- [GW74] GORDON, WILLIAM J. and WIXOM, JAMES A. “Pseudo-harmonic interpolation on convex domains”. *SIAM Journal on Numerical Analysis* 11.5 (1974), 909–933. DOI: [10.1137/0711072](https://doi.org/10.1137/0711072) 2.
- [Han15] HANG, SI. “TetGen, a Delaunay-based quality tetrahedral mesh generator”. *ACM Transactions Mathematical Software* 41.2 (Jan. 2015), Article 11, 36 pages. DOI: [10.1145/2629697](https://doi.org/10.1145/2629697) 7.
- [HF06] HORMANN, KAI and FLOATER, MICHAEL S. “Mean value coordinates for arbitrary planar polygons”. *ACM Transactions on Graphics* 25.4 (Oct. 2006), 1424–1441. DOI: [10.1145/1183287.1183295](https://doi.org/10.1145/1183287.1183295) 2, 8.
- [Hop96] HOPPE, HUGUES. “Progressive meshes”. *Proceedings of the 23rd Annual Conference on Computer Graphics and Interactive Techniques*. SIGGRAPH ’96. New Orleans, Aug. 1996, 99–108. DOI: [10.1145/237170.237216](https://doi.org/10.1145/237170.237216) 5.

- [HS08] HORMANN, KAI and SUKUMAR, NATARAJAN. “Maximum entropy coordinates for arbitrary polytopes”. *Computer Graphics Forum* 27.5 (July 2008), 1513–1520. DOI: [10.1111/j.1467-8659.2008.01292.x](https://doi.org/10.1111/j.1467-8659.2008.01292.x) 2, 13, 17.
- [HS17] HORMANN, KAI and SUKUMAR, NATARAJAN. *Generalized Barycentric Coordinates in Computer Graphics and Computational Mechanics*. Boca Raton: Taylor & Francis/CRC, 2017. DOI: [10.1201/9781315153452.2](https://doi.org/10.1201/9781315153452.2).
- [HSW*20] HU, YIXIN, SCHNEIDER, TESEO, WANG, BOLUN, et al. “Fast tetrahedral meshing in the wild”. *ACM Transactions on Graphics* 39.4 (Aug. 2020), Article 117, 18 pages. DOI: [10.1145/3386569.3392385](https://doi.org/10.1145/3386569.3392385) 6.
- [HT04] HORMANN, KAI and TARINI, MARCO. “A quadrilateral rendering primitive”. *Proceedings of Graphics Hardware*. GH ’04. Grenoble, Aug. 2004, 7–14. DOI: [10.2312/EGGH/EGGH04/007-014.2](https://doi.org/10.2312/EGGH/EGGH04/007-014.2).
- [HZG*18] HU, YIXIN, ZHOU, QINGNAN, GAO, XIFENG, et al. “Tetrahedral meshing in the wild”. *ACM Transactions on Graphics* 37.4 (Aug. 2018), Article 60, 14 pages. DOI: [10.1145/3197517.3201353](https://doi.org/10.1145/3197517.3201353) 6.
- [Jac14] JACOBSON, ALEC. “Automatic skinning via constrained energy optimization”. *Skining: Real-Time Shape Deformation*. SIGGRAPH 2014 Course Notes. July 2014. Chap. 2. DOI: [10.1145/2614028.2615427](https://doi.org/10.1145/2614028.2615427) 3, 5.
- [JBPS11] JACOBSON, ALEC, BARAN, ILYA, POPOVIĆ, JOVAN, and SORKINE, OLGA. “Bounded biharmonic weights for real-time deformation”. *ACM Transactions on Graphics* 30.4 (July 2011), Article 78, 8 pages. DOI: [10.1145/2010324.1964973](https://doi.org/10.1145/2010324.1964973) 12, 17.
- [JLW07] JU, TAO, LIEPA, PETER, and WARREN, JOE. “A general geometric construction of coordinates in a convex simplicial polytope”. *Computer Aided Geometric Design* 24.3 (Apr. 2007), 161–178. DOI: [10.1016/j.cagd.2006.12.001](https://doi.org/10.1016/j.cagd.2006.12.001) 8.
- [JMD*07] JOSHI, PUSHKAR, MEYER, MARK, DEROSE, TONY, et al. “Harmonic coordinates for character articulation”. *ACM Transactions on Graphics* 26.3 (July 2007), Article 71, 9 pages. DOI: [10.1145/1276377.1276466](https://doi.org/10.1145/1276377.1276466) 2, 11, 17.
- [JP*24] JACOBSON, ALEC, PANOZZO, DANIELE, et al. *libigl: A simple C++ geometry processing library*. 2024. URL: <https://libigl.github.io/> (visited on 01/31/2024) 12.
- [JSW05] JU, TAO, SCHAEFER, SCOTT, and WARREN, JOE. “Mean value coordinates for closed triangular meshes”. *ACM Transactions on Graphics* 24.3 (July 2005), 561–566. DOI: [10.1145/1073204.1073229](https://doi.org/10.1145/1073204.1073229) 2, 8–10.
- [JSWD05] JU, TAO, SCHAEFER, SCOTT, WARREN, JOE, and DESBRUN, MATHIEU. “A geometric construction of coordinates for convex polyhedra using polar duals”. *Proceedings of the 3rd Symposium on Geometry Processing*. SGP ’05. Vienna, July 2005, 181–186. DOI: [10.2312/SGP/SGP05/181-186.8](https://doi.org/10.2312/SGP/SGP05/181-186.8).
- [JZvdP*08] JU, TAO, ZHOU, QIAN-YI, van de PANNE, MICHIEL, et al. “Reusable skinning templates using cage-based deformations”. *ACM Transactions on Graphics* 27.5 (Dec. 2008), Article 122, 10 pages. DOI: [10.1145/1409060.1409075](https://doi.org/10.1145/1409060.1409075) 1, 6, 7.
- [KCŽO08] KAVAN, LADISLAV, COLLINS, STEVEN, ŽÁRA, JIŘÍ, and O’SULLIVAN, CAROL. “Geometric skinning with approximate dual quaternion blending”. *ACM Transactions on Graphics* 27.4 (Oct. 2008), Article 105, 23 pages. DOI: [10.1145/1409625.1409627](https://doi.org/10.1145/1409625.1409627) 4.
- [KSKL14] KIM, JONGMIN, SEOL, YEONGHO, KWON, TAESOO, and LEE, JEHEE. “Interactive manipulation of large-scale crowd animation”. *ACM Transactions on Graphics* 33.4 (July 2014), Article 83, 10 pages. DOI: [10.1145/2601097.2601170](https://doi.org/10.1145/2601097.2601170) 1.
- [LBS06] LANGER, TORSTEN, BELYAEV, ALEXANDER, and SEIDEL, HANS-PETER. “Spherical barycentric coordinates”. *Proceedings of the 4th Symposium on Geometry Processing*. SGP ’06. Cagliari, June 2006, 81–88. DOI: [10.2312/SGP/SGP06/081-088](https://doi.org/10.2312/SGP/SGP06/081-088) 9–11, 17.
- [LD17] LE, BINH HUY and DENG, ZHIGANG. “Interactive cage generation for mesh deformation”. *Proceedings of the 21st Symposium on Interactive 3D Graphics and Games*. I3D ’17. San Francisco, Feb. 2017, Article 3, 9 pages. DOI: [10.1145/3023368.3023369](https://doi.org/10.1145/3023368.3023369) 5–8.
- [LD89] LOOP, CHARLES T. and DEROSE, TONY D. “A multisided generalization of Bézier surfaces”. *ACM Transactions on Graphics* 8.3 (July 1989), 204–234. DOI: [10.1145/77055.77059](https://doi.org/10.1145/77055.77059) 2.
- [LH13] LI, XIAN-YING and HU, SHI-MIN. “Poisson coordinates”. *IEEE Transactions on Visualization and Computer Graphics* 19.2 (Feb. 2013), 344–352. DOI: [10.1109/TVCG.2012.109](https://doi.org/10.1109/TVCG.2012.109) 2.
- [LJH13] LI, XIAN-YING, JU, TAO, and HU, SHI-MIN. “Cubic mean value coordinates”. *ACM Transactions on Graphics* 32.4 (July 2013), Article 126, 10 pages. DOI: [10.1145/2461912.2461917](https://doi.org/10.1145/2461912.2461917) 2.
- [LKCL07] LIPMAN, YARON, KOPF, JOHANNES, COHEN-OR, DANIEL, and LEVIN, DAVID. “GPU-assisted positive mean value coordinates for mesh deformations”. *Proceedings of the 5th Symposium on Geometry Processing*. SGP ’07. Barcelona, July 2007, 117–123. DOI: [10.2312/SGP/SGP07/117-123](https://doi.org/10.2312/SGP/SGP07/117-123) 2, 9, 17.
- [LLC08] LIPMAN, YARON, LEVIN, DAVID, and COHEN-OR, DANIEL. “Green coordinates”. *ACM Transactions on Graphics* 27.3 (Aug. 2008), Article 78, 10 pages. DOI: [10.1145/1360612.1360677](https://doi.org/10.1145/1360612.1360677) 14, 15, 17.
- [LLH22] LAM, KIT YUNG, LEE, LIK-HANG, and HUI, PAN. “3DeformR: Freehand 3D model editing in virtual environments considering head movements on mobile headsets”. *Proceedings of the 13th Multimedia Systems Conference*. MMSys ’22. Athlone, June 2022, 52–61. DOI: [10.1145/3524273.3528180](https://doi.org/10.1145/3524273.3528180) 1.
- [Lo15] LO, DANIEL S. H. *Finite Element Mesh Generation*. Boca Raton: Taylor & Francis/CRC, 2015. DOI: [10.1201/b17713](https://doi.org/10.1201/b17713) 7.
- [LS07] LANGER, TORSTEN and SEIDEL, HANS-PETER. “Mean value Bézier surfaces”. *Mathematics of Surfaces XII*. Ed. by MARTIN, RALPH, SABIN, MALCOLM, and WINKLER, JOAB. Vol. 4647. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer, 2007, 263–274. DOI: [10.1007/978-3-540-73843-5_16](https://doi.org/10.1007/978-3-540-73843-5_16) 2.
- [LS08] LANGER, TORSTEN and SEIDEL, HANS-PETER. “Higher order barycentric coordinates”. *Computer Graphics Forum* 27.2 (Apr. 2008), 459–466. DOI: [10.1111/j.1467-8659.2008.01143.x](https://doi.org/10.1111/j.1467-8659.2008.01143.x) 2.
- [LU16] LAUBE, PASCAL and UMLAUF, GEORG. “A short survey on recent methods for cage computation”. *Proceedings of the 3rd BW-CAR Symposium on Information and Communication Systems*. SInCom ’16. Karlsruhe, Dec. 2016, 37–42 5.
- [LW07] LI, JITUO and WANG, YANGSHENG. “Automatically constructing skeletons and parametric structures for polygonal human bodies”. *Proceedings of the 25th Computer Graphics International Conference*. CGI ’07. Pétopolis, May 2007 4.
- [Mac49] MACNEAL, RICHARD H. “The Solution of Partial Differential Equations by Means of Electrical Networks”. PhD thesis. California Institute of Technology, 1949. DOI: [10.7907/PZ04-5290](https://doi.org/10.7907/PZ04-5290) 8.
- [MCA15] MANDAD, MANISH, COHEN-STEINER, DAVID, and ALLIEZ, PIERRE. “Isotopic approximation within a tolerance volume”. *ACM Transactions on Graphics* 34.4 (July 2015), Article 64, 12 pages. DOI: [10.1145/2766950](https://doi.org/10.1145/2766950) 5.
- [MJ96] MACCRACKEN, RON and JOY, KENNETH I. “Free-form deformations with lattices of arbitrary topology”. *Proceedings of the 23rd Annual Conference on Computer Graphics and Interactive Techniques*. SIGGRAPH ’96. New Orleans, Aug. 1996, 181–188. DOI: [10.1145/237170.237247](https://doi.org/10.1145/237170.237247) 4.
- [MKB*08] MARTIN, SEBASTIAN, KAUFMANN, PETER, BOTSCH, MARIO, et al. “Polyhedral finite elements using harmonic basis functions”. *Computer Graphics Forum* 27.5 (July 2008), 1521–1529. DOI: [10.1111/j.1467-8659.2008.01293.x](https://doi.org/10.1111/j.1467-8659.2008.01293.x) 11.
- [MLBD02] MEYER, MARK, LEE, HAEYOUNG, BARR, ALAN, and DESBRUN, MATHIEU. “Generalized barycentric coordinates on irregular polygons”. *Journal of Graphics Tools* 7.1 (2002), 13–22. DOI: [10.1080/10867651.2002.10487551](https://doi.org/10.1080/10867651.2002.10487551) 2.

- [MLD05] MALSCH, ELISABETH A., LIN, JOHN J., and DASGUPTA, GAUTAM. “Smooth two dimensional interpolants: A recipe for all polygons”. *Journal of Graphics Tools* 10.2 (2005), 27–39. DOI: [10.1080/2151237X.2005.10129192.2](#).
- [MLS11] MANSON, JOSIAH, LI, KUIYU, and SCHAEFER, SCOTT. “Positive Gordon–Wixom coordinates”. *Computer-Aided Design* 43.11 (Nov. 2011), 1422–1426. DOI: [10.1016/j.cad.2011.08.019.2](#).
- [MLT88] MAGNENAT-THALMANN, NADIA, LAPERRIÈRE, RICHARD, and THALMANN, DANIEL. “Joint-dependent local deformations for hand animation and object grasping”. *Proceedings of Graphics Interface*. GI ’88. Edmonton, June 1988, 26–33. DOI: [10.20380/GI1988.04.1.4](#).
- [MMG06] MERRY, BRUCE, MARAIS, PATRICK, and GAIN, JAMES. “Animation space: A truly linear framework for character animation”. *ACM Transactions on Graphics* 25.4 (Oct. 2006), 1400–1423. DOI: [10.1145/1183287.1183294.4](#).
- [Möb27] MÖBIUS, AUGUST F. *Der barycentrische Calcul*. Leipzig: Johann Ambrosius Barth Verlag, 1827.2.
- [MP07] MILBRADT, PETER and PICK, TOBIAS. “Polytope finite elements”. *International Journal for Numerical Methods in Engineering* 73.12 (July 2007), 1811–1835. DOI: [10.1002/nme.2149.2](#).
- [MS10] MANSON, JOSIAH and SCHAEFER, SCOTT. “Moving least squares coordinates”. *Computer Graphics Forum* 29.5 (July 2010), 1517–1524. DOI: [10.1111/j.1467-8659.2010.01760.x2](#).
- [MSW*09] MENG, WEILIANG, SHENG, BIN, WANG, SHANDONG, et al. “Interactive image deformation using cage coordinates on GPU”. *Proceedings of the 8th International Conference on Virtual Reality Continuum and its Applications in Industry*. VRCAI ’09. Yokohama, Dec. 2009, 119–126. DOI: [10.1145/1670252.1670279.1](#).
- [MT23] MICHEL, ÉLIE and THIERY, JEAN-MARC. “Polynomial 2D Green coordinates for polygonal cages”. *SIGGRAPH 2023 Conference Proceedings*. Los Angeles, July 2023, Article 23, 9 pages. DOI: [10.1145/3588432.3591499.2](#).
- [MWUP22] MORRICAL, NATE, WALD, INGO, USHER, WILL, and PASCUCCI, VALERIO. “Accelerating unstructured mesh point location with RT cores”. *IEEE Transactions on Visualization and Computer Graphics* 28.8 (Aug. 2022), 2852–2866. DOI: [10.1109/tvcg.2020.3042930.8](#).
- [NF*06] NESME, MATTHIEU, FAURE, FRANÇOIS, et al. “Animating shapes at arbitrary resolution with non-uniform stiffness”. *Proceedings of the 3rd Workshop on Virtual Reality Interactions and Physical Simulations*. VRIPHYS ’06. Madrid, Nov. 2006, 17–24. DOI: [10.2312/PE/vrphys/vrphys06/017-024.5](#).
- [NKJF09] NESME, MATTHIEU, KRY, PAUL G., JEŘÁBKOVÁ, LENKA, and FAURE, FRANÇOIS. “Preserving topology and elasticity for embedded deformable models”. *ACM Transactions on Graphics* 28.3 (Aug. 2009), Article 52, 9 pages. DOI: [10.1145/1531326.1531358.5.7](#).
- [NS12] NIETO, JESÚS R. and SUSÍN, ANTONIO. “Cage based deformations: A survey”. *Deformation Models*. Ed. by HIDALGO, MANUEL G., TORRES, ARNAU M., and GÓMEZ, JAVIER V. Vol. 7. Lecture Notes in Computational Vision and Biomechanics. Dordrecht: Springer, 2012, 75–99. DOI: [10.1007/978-94-007-5446-1_3.1](#).
- [NW06] NOCEDAL, JORGE and WRIGHT, STEPHEN J. *Numerical Optimization*. Second. Springer Series in Operations Research and Financial Engineering. New York: Springer, 2006. DOI: [10.1007/978-0-387-40065-5.12](#).
- [PGWB21] PORZIANI, STEFANO, GROTH, CORRADO, WALDMAN, WITOLD, and BIANCOLINI, MARCO E. “Automatic shape optimisation of structural parts driven by BGM and RBF mesh morphing”. *International Journal of Mechanical Sciences* 189 (Jan. 2021), Article 105976, 11 pages. DOI: [10.1016/j.ijmecsci.2020.105976.4](#).
- [PP93] PINKALL, ULRICH and POLTHIER, KONRAD. “Computing discrete minimal surfaces and their conjugates”. *Experimental Mathematics* 2.1 (1993), 15–36. DOI: [10.1080/10586458.1993.10504266.2.8](#).
- [PYL*22] PENG, YICONG, YAN, YICHAO, LIU, SHENGQI, et al. “CageNeRF: Cage-based neural radiance field for generalized 3D deformation and animation”. *Proceedings of the Conference on Neural Information Processing Systems*. NeurIPS ’22. New Orleans, Nov. 2022, 31402–31415.1,21.
- [RPPS17] RABINOVICH, MICHAEL, PORANNE, ROI, PANOZZO, DANIELE, and SORKINE-HORNUNG, OLGA. “Scalable locally injective mappings”. *ACM Transactions on Graphics* 36.2 (Apr. 2017), Article 16, 16 pages. DOI: [10.1145/2983621.7](#).
- [Rus07] RUSTAMOV, RAIF M. *Boundary Element Formulation of Harmonic Coordinates*. Tech. rep. Department of Mathematics, Purdue University, Nov. 2007.11.
- [SF11] SAVOYE, YANN and FRANCO, JEAN-SÉBASTIEN. “Cage-based tracking for performance animation”. *Computer Vision – ACCV 2010*. Ed. by KIMMEL, RON, KLETTE, REINHARD, and SUGIMOTO, AKIHIRO. Vol. 6494. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer, 2011, 599–612. DOI: [10.1007/978-3-642-19318-7_47.1](#).
- [SGG*00] SANDER, PEDRO V., GU, XIANFENG, GORTLER, STEVEN J., et al. “Silhouette clipping”. *Proceedings of the 27th Annual Conference on Computer Graphics and Interactive Techniques*. SIGGRAPH ’00. New Orleans, July 2000, 327–334. DOI: [10.1145/344779.344935.5.7](#).
- [SHD*18] SCHNEIDER, TESEO, HU, YIXIN, DUMAS, JÉRÉMIE, et al. “Decoupling simulation accuracy from mesh quality”. *ACM Transactions on Graphics* 37.6 (Dec. 2018), Article 280, 14 pages. DOI: [10.1145/3272127.3275067.7](#).
- [She02] SHEWCHUK, JONATHAN R. “What is a good linear element? Interpolation, conditioning, and quality measures”. *Proceedings of the 11th International Meshing Roundtable*. IMR ’02. Ithaca, Sept. 2002, 115–126.7.
- [SHF13] SCHNEIDER, TESEO, HORMANN, KAI, and FLOATER, MICHAEL S. “Bijective composite mean value mappings”. *Computer Graphics Forum* 32.5 (Aug. 2013), 137–146. DOI: [10.1111/cgf.12180.2](#).
- [SM06] SUKUMAR, NATARAJAN and MALSCH, ELISABETH A. “Recent advances in the construction of polygonal finite element interpolants”. *Archives of Computational Methods in Engineering* 13.1 (Mar. 2006), 129–163. DOI: [10.1007/BF02905933.2](#).
- [SMK*10] SELLAMANI, SUBRAMANI, MUTHUGANAPATHY, RAMANATHAN, KALYANARAMAN, YAGNANARAYANAN, et al. “PCS: Prominent cross-sections for mesh models”. *Computer-Aided Design and Applications* 7.4 (2010), 601–620. DOI: [10.3722/cadaps.2010.601-620.6](#).
- [SMSF20] STRÖTER, DANIEL, MUELLER-ROEMER, JOHANNES S., STORK, ANDRÉ, and FELLNER, DIETER W. “OLBVH: Octree linear bounding volume hierarchy for volumetric meshes”. *The Visual Computer* 36.10–12 (Oct. 2020), 2327–2340. DOI: [10.1007/s00371-020-01886-6.8](#).
- [SMWF22] STRÖTER, DANIEL, MUELLER-ROEMER, JOHANNES S., WEBER, DANIEL, and FELLNER, DIETER W. “Fast harmonic tetrahedral mesh optimization”. *The Visual Computer* 38.9–10 (Sept. 2022), 3419–3433. DOI: [10.1007/s00371-022-02547-6.7](#).
- [Som85] SOMIGLIANA, CARLO. “Sopra l’equilibrio di un corpo elastico isotropo”. *Il Nuovo Cimento* 17 (Dec. 1885), 140–148. DOI: [10.1007/BF02817783.16](#).
- [SOS04] SHEN, CHEN, O’BRIEN, JAMES F., and SHEWCHUK, JONATHAN R. “Interpolating and approximating implicit surfaces from polygon soup”. *ACM Transactions on Graphics* 23.3 (Aug. 2004), 896–904. DOI: [10.1145/1015706.1015816.5.7](#).

- [SP86] SEDERBERG, THOMAS W. and PARRY, SCOTT R. “Free-form deformation of solid geometric models”. *Proceedings of the 13th Annual Conference on Computer Graphics and Interactive Techniques. SIGGRAPH '86*. Dallas, Aug. 1986, 151–160. DOI: [10.1145/15922.15903](https://doi.org/10.1145/15922.15903) 1, 4.
- [SS10] SCHWARZ, MICHAEL and SEIDEL, HANS-PETER. “Fast parallel surface and solid voxelization on GPUs”. *ACM Transactions on Graphics* 29.6 (Dec. 2010), Article 179, 10 pages. DOI: [10.1145/1882261.1866201](https://doi.org/10.1145/1882261.1866201) 6.
- [Suk04] SUKUMAR, NATARAJAN. “Construction of polygonal interpolants: A maximum entropy approach”. *International Journal for Numerical Methods in Engineering* 61.12 (Nov. 2004), 2159–2181. DOI: [10.1002/nme.1193](https://doi.org/10.1002/nme.1193) 13.
- [SV18] SALVI, PÉTER and VÁRADY, TAMÁS. “Multi-sided Bézier surfaces over concave polygonal domains”. *Computers & Graphics* 74 (Aug. 2018), 56–65. DOI: [10.1016/j.cag.2018.05.006](https://doi.org/10.1016/j.cag.2018.05.006) 2.
- [SVJ15] SACHT, LEONARDO, VOUGA, ETIENNE, and JACOBSON, ALEC. “Nested cages”. *ACM Transactions on Graphics* 34.6 (Nov. 2015), Article 170, 14 pages. DOI: [10.1145/2816795.2818093](https://doi.org/10.1145/2816795.2818093) 5, 7, 8.
- [SZG*20] SCALAS, ANDREAS, ZHU, YUANJU, GIANNINI, FRANCA, et al. “A first step towards cage-based deformation in virtual reality”. *Proceedings of the Italian Chapter Conference on Smart Tools and Apps for Graphics. STAG '20*. Online, Nov. 2020, 119–130. DOI: [10.2312/STAG.20201246](https://doi.org/10.2312/STAG.20201246) 1.
- [TB22] THIERY, JEAN-MARC and BOUBEKEUR, TAMY. “Green coordinates for triquad cages in 3D”. *SIGGRAPH Asia 2022 Conference Proceedings*. Daegu, Dec. 2022, Article 38, 8 pages. DOI: [10.1145/3550469.3555400](https://doi.org/10.1145/3550469.3555400) 1, 15–17.
- [TDZ19] TAO, JIONG, DENG, BAILIN, and ZHANG, JUYONG. “A fast numerical solver for local barycentric coordinates”. *Computer Aided Geometric Design* 70 (Mar. 2019), 46–58. DOI: [10.1016/j.cagd.2019.04.006](https://doi.org/10.1016/j.cagd.2019.04.006) 2, 12.
- [TMB18] THIERY, JEAN-MARC, MEMARI, POORAN, and BOUBEKEUR, TAMY. “Mean value coordinates for quad cages in 3D”. *ACM Transactions on Graphics* 37.6 (Dec. 2018), Article 229, 14 pages. DOI: [10.1145/3275127.3275063](https://doi.org/10.1145/3275127.3275063) 1, 10, 11, 15–17.
- [TS08] TABARRAEI, ALIREZA and SUKUMAR, NATARAJAN. “Extended finite element method on polygonal and quadtree meshes”. *Computer Methods in Applied Mechanics and Engineering* 197.5 (Jan. 2008), 425–438. DOI: [10.1016/j.cma.2007.08.013](https://doi.org/10.1016/j.cma.2007.08.013) 2.
- [TSN10] TAKAYAMA, KENSHI, SORKINE, OLGA, NEALEN, ANDREW, and IGARASHI, TAKEO. “Volumetric modeling with diffusion surfaces”. *ACM Transactions on Graphics* 29.6 (Dec. 2010), Article 180, 8 pages. DOI: [10.1145/1882261.1866202](https://doi.org/10.1145/1882261.1866202) 2.
- [TTB12] THIERY, JEAN-MARC, TIERNY, JULIEN, and BOUBEKEUR, TAMY. “CageR: Cage-based reverse engineering of animated 3D shapes”. *Computer Graphics Forum* 31.8 (Oct. 2012), 2303–2316. DOI: [10.1111/j.1467-8659.2012.03159.x](https://doi.org/10.1111/j.1467-8659.2012.03159.x) 1.
- [Ura00] URAGO, MASATAKA. “Analytical integrals of fundamental solution of three-dimensional Laplace equation and their gradients”. *Transactions of the Japan Society of Mechanical Engineers Series A* 66.642 (Feb. 2000), 254–261. DOI: [10.1299/kikaia.66.254](https://doi.org/10.1299/kikaia.66.254) 15.
- [VF09] VASILAKIS, ANDREAS and FUDOS, IOANNIS. “Skeleton-based rigid skinning for character animation”. *Proceedings of the 4th International Conference on Computer Graphics Theory and Applications. GRAPP '09*. Lisboa, Feb. 2009, 302–308. DOI: [10.5220/0001799803020308](https://doi.org/10.5220/0001799803020308) 4.
- [VKK*03] VARADHAN, GOKUL, KRISHNAN, SHANKAR, KIM, YOUNG J., et al. “Efficient max-norm distance computation and reliable voxelization”. *Proceedings of the 1st Symposium on Geometry Processing. SGP '03*. Aachen, June 2003, 116–126. DOI: [10.2312/SGP/SGP03/116-126](https://doi.org/10.2312/SGP/SGP03/116-126) 5.
- [Wac75] WACHSPRESS, EUGENE L. *A Rational Finite Element Basis*. Vol. 114. Mathematics in Science and Engineering. New York: Academic Press, 1975 2.
- [War96] WARREN, JOE. “Barycentric coordinates for convex polytopes”. *Advances in Computational Mathematics* 6.1 (Dec. 1996), 97–108. DOI: [10.1007/BF02127699](https://doi.org/10.1007/BF02127699) 8.
- [WBG07] WICKE, MARCO, BOTSCH, MARIO, and GROSS, MARKUS. “A finite element method on convex polyhedra”. *Computer Graphics Forum* 26.3 (Sept. 2007), 355–364. DOI: [10.1111/j.1467-8659.2007.01058.x](https://doi.org/10.1111/j.1467-8659.2007.01058.x) 2.
- [WG10] WEBER, OFIR and GOTSMAN, CRAIG. “Controllable conformal maps for shape deformation and interpolation”. *ACM Transactions on Graphics* 29.4 (July 2010), Article 78, 11 pages. DOI: [10.1145/1778765.1778815](https://doi.org/10.1145/1778765.1778815) 2.
- [WJBK15] WANG, YU, JACOBSON, ALEC, BARBIČ, JERNEJ, and KAVAN, LADISLAV. “Linear subspace design for real-time shape deformation”. *ACM Transactions on Graphics* 34.4 (Aug. 2015), Article 57, 11 pages. DOI: [10.1145/2766952](https://doi.org/10.1145/2766952) 1, 4.
- [WLMD19] WANG, ZHIHAO, LI, YAJUAN, MA, WEIYIN, and DENG, CHONGYANG. “Positive and smooth Gordon–Wixom coordinates”. *Computer Aided Geometric Design* 74 (Oct. 2019), Article 101774, 9 pages. DOI: [10.1016/j.cagd.2019.101774](https://doi.org/10.1016/j.cagd.2019.101774) 2.
- [WS21] WANG, YU and SOLOMON, JUSTIN. “Fast quasi-harmonic weights for geometric data interpolation”. *ACM Transactions on Graphics* 40.4 (Aug. 2021), Article 73, 15 pages. DOI: [10.1145/3450626.3459801](https://doi.org/10.1145/3450626.3459801) 4.
- [WSHD07] WARREN, JOE, SCHAEFER, SCOTT, HIRANI, ANIL N., and DESBRUN, MATHIEU. “Barycentric coordinates for convex sets”. *Advances in Computational Mathematics* 27.3 (Oct. 2007), 319–338. DOI: [10.1007/s10444-005-9008-6](https://doi.org/10.1007/s10444-005-9008-6) 2, 8.
- [XH22] XU, TIANHAN and HARADA, TATSUYA. “Deforming radiance fields with cages”. *Computer Vision – ECCV 2022*. Ed. by AVIDAN, SHAI, BROSTOW, GABRIEL, Cissé, MOUSTAPHA, et al. Vol. 13693. Lecture Notes in Computer Science. Cham: Springer, 2022, 159–175. DOI: [10.1007/978-3-031-19827-4_10](https://doi.org/10.1007/978-3-031-19827-4_10) 1, 21.
- [XLG09] XIAN, CHUHUA, LIN, HONGWEI, and GAO, SHUMING. “Automatic generation of coarse bounding cages from dense meshes”. *Proceedings of the International Conference on Shape Modeling and Applications. SMI '09*. Beijing, June 2009, 21–27. DOI: [10.1109/smi.2009.5170159](https://doi.org/10.1109/smi.2009.5170159) 5, 7.
- [XLG11] XIAN, CHUHUA, LIN, HONGWEI, and GAO, SHUMING. “Automatic cage generation by improved OBBs for mesh deformation”. *The Visual Computer* 28.1 (Apr. 2011), 21–33. DOI: [10.1007/s00371-011-0595-6](https://doi.org/10.1007/s00371-011-0595-6) 5, 7.
- [XLX15] XIAN, CHUHUA, LI, GUIQING, and XIONG, YUNHUI. “Efficient and effective cage generation by region decomposition”. *Computer Animation & Virtual Worlds* 26.2 (Mar. 2015), 173–184. DOI: [10.1002/cav.1571](https://doi.org/10.1002/cav.1571) 1, 5–7.
- [YAK*20] YIFAN, WANG, AIGERMAN, NOAM, KIM, VLADIMIR G., et al. “Neural cages for detail-preserving 3D deformations”. *Proceedings of the Conference on Computer Vision and Pattern Recognition. CVPR '20*. Seattle, June 2020, 72–80. DOI: [10.1109/CVPR42600.2020.00015](https://doi.org/10.1109/CVPR42600.2020.00015) 1, 21.
- [YCSZ12] YANG, XIAOSONG, CHANG, JIAN, SOUTHERN, RICHARD, and ZHANG, JIAN J. “Automatic cage construction for retargeted muscle fitting”. *The Visual Computer* 29.5 (June 2012), 369–380. DOI: [10.1007/s00371-012-0739-3](https://doi.org/10.1007/s00371-012-0739-3) 6–8.
- [YS19] YAN, ZHIPEI and SCHAEFER, SCOTT. “A family of barycentric coordinates for co-dimension 1 manifolds with simplicial facets”. *Computer Graphics Forum* 38.5 (Aug. 2019), 75–83. DOI: [10.1111/cgf.13790](https://doi.org/10.1111/cgf.13790) 2, 8.
- [ZDL*14] ZHANG, JUYONG, DENG, BAILIN, LIU, ZISHUN, et al. “Local barycentric coordinates”. *ACM Transactions on Graphics* 33.6 (Nov. 2014), Article 188, 12 pages. DOI: [10.1145/2661229.2661255](https://doi.org/10.1145/2661229.2661255) 1, 2, 12, 17.