

Chapter 2

Sparse Constraints Over Animatable Subspaces

In this chapter, we present a cage-based inversion process for modeling tasks. This novel core methodology estimates coherent enclosed model deformation via sparse positional constraints. In particular, we investigate a new technique mixing surface-and-space deformation in order to fit input feature correspondences. Moreover, user-specified constraints are transferred into the cage domain with a well-adapted and detached Laplacian-type regularization. Thus, we suggest a novel algorithm to easily edit shape deformation using the *inverse kinematics* paradigm applied in geometric space. Finally, an intuitive user interface is proposed to allow users to easily specify desired screen-space constraints.

2.1 Introduction

Inverse Mesh Modeling. Nowadays, mesh editing and inverse modeling techniques play a decisive role in *Computer Graphics*. This research domain has been intensively studied over the years. Nevertheless, the relentless increase in demand of industry has motivated researchers to exhibit new coordinate systems as-well-as new optimization frameworks. Building accessible pipelines able to provide flexible output for animation re-use is a challenging issue. In nature, deformation techniques can be seen as an energy minimization process (defined locally or globally) that measures how much the object has been deformed from its initial pose given a support domain (for instance surface or volume representation). Approximating global shape characteristics of the surface aims to enforce specific surface resistance properties (such as rigidity, flexibility or elasticity). One significant challenge is to find a fast framework to achieve plausible and scalable boneless inverse kinematics that produces pleasing deformations and preserves the global properties of the surface. Avoiding the articulated skeleton parametrization to achieve large deformation allows us to considerate a large number of highly non-rigid deformations that occur in the physical world, also known as *free-form deformation*.

Constraint-based Mesh Optimality. With the explosion in polygons, mesh deformation tools definitively take advantage of parametrizations demonstrating independence toward the resolution of the surface-of-interest. Meanwhile, cage-based deformation techniques are increasingly used to control the deformation of an enclosed fine-detailed mesh in commercial software. Achieving deformation based on vertex constraints has been extensively studied for the case of pure meshes, but a few works specifically examine how such vertex constraints can be used to efficiently deform the models and to estimate the corresponding cage-pose. In this chapter, we show that mesh optimality can be efficiently reached within a framework employing *Cage-based Deformation* enhanced by *Laplacian Mesh Editing* principles. First, we provide a linear estimation framework for cage-vertex coordinates. Second, our deformation regularization is expressed exclusively on the cage connectivity rather than the enclosed mesh, yielding a computationally efficient solution which fully benefits from cage-based parameterization. We demonstrate a practical use of this scheme for dense mesh editing from sparse screen-space constraints reprojected into the world-space coordinates.

Contributions. In this chapter, we combine surface and space deformation techniques into an inverse variational modeling approach. Indeed, we focus on estimating an enclosed model that fits some sparse surfel displacement constraints, specified by the user (not limited to artists) over the enclosed mesh surface. We explore the least-squares cage geometry as an intermediate, transparent and off-the-shell tool, that is not directly editable by users. In other words, the model is embedded in a volumetric bounding cage using generalized barycentric coordinates with local properties. The optimal geometric solution is obtained by a single minimization within a linear black-box solver.

However, the key contribution of this chapter lies in the use of sparse linear surface constraints to estimate the global cage parameters reproducing the desired deformation of the enclosed model. Besides, such constraints are expressed on the enclosed model and transferred into the cage-based subspace (offered by the bounding cage). Consequently, we take advantage of optimal reduced parameters provided by the given coarse cage surrounding the surface. In addition, the cage structure is enhanced with Laplacian regularization leading to an effective over-determined system. The Laplacian cage maintains a spatial deformation of the mesh geometry, more powerful than applying *surface-based* or *cage-based* techniques separately. Finally, we show how Laplacian regularizations can be operated to non-manifold animatable subspaces such as traditional articulated kinematic skeletons.

Chapter Reminder. The rest of the chapter is organized as follows. First, we briefly review most of the relevant work on interactive deformable surface and shape parametrization in Sect. 2.2. In particular, we propose a clear understanding concerning *surface-based*, *skeleton-based*, *space-based* and *cage-based* parametrization. Then, we give a conceptual and mathematical description of our inverse cage-based system in Sect. 2.3, and we show the effectiveness of our method on various datasets. Next, a generalization of the employed variational strategy to skeletal subspace structures is offered in Sect. 2.4. In this specific case, we found that Laplacian regularization

breaks the rigidity of skeletal subspaces by discretizing an elastic energy on their connected joints. Finally, this chapter is concluded in Sect. 2.5 with some discussions about benefits and limitations of our system.

2.2 Related Works on Interactive Deformable Surface

In this section, we present various relevant parametrizations in the area of *Interactive Deformable Surfaces* that are useful from digitized model to animatable surface. In particular, we focus on the representation of animatable discrete surface with reduced and reusable parameters. More importantly, we overview the large body of recent work on techniques addressing the problem of interactive mesh deformation using underlying deformable structure, with the aim to be used for video-based animation purpose. In particular, we are interested in three different levels: *surface-based*, *skeleton-based*, and *space-based* parameterization. Finally, our focus will be extensively oriented on *cage-based* paradigm.

2.2.1 Deformable Models

Theory of Deformable Models. Deformable models are efficient and popular tools in both *Computer Graphics* and *Computer Vision* where the main idea is to encode and manipulate a variety of flexible shape, at the heart of many applications. For instance, deformable models have been used in animation and simulation. Generally, deformable models can be described as a class of physics-based models that deform under the laws of Newton mechanics. Another advantage of attractive deformable models is to reduce the dependence to the object representation by providing information on its global behavior. In addition, deformable models provide promising techniques for registration. Simulating an animation can be seen as deforming a model across time. In case of dense meshes, a discrete deformable model is required

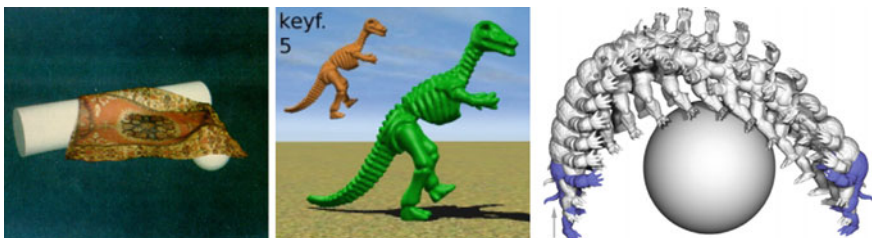


Fig. 2.1 Deformable models: (from left to right) an elastic deformable model (image courtesy of [2]), a reduced deformable model (image courtesy of [3]), a physically-based meshless deformable model (image courtesy of [4])

2.2.2 Surface-based Parameterization

Differential Geometry. In recent years, differential geometry became an important topic of interest where manifold geometry is proceeded through linear algebra and differential calculus. Many efforts have been expanded on surface-based deformation. Consequently, *differential mesh editing* techniques are becoming a prominent suitable tool for preserving surface details during mesh operations. For instance, Laplacian coordinates provide an efficient intrinsic representation of the surface mesh to encode its local geometry as shown by Wardetzky et al. [8]. These editing systems represent local features using differential coordinates defined in a global coordinate system.

There are several types of highly impacting approaches exploiting a differential descriptor of the edited surface in terms of Laplacian and differential coordinates for mesh editing [9, 10]. Differential information as local intrinsic feature descriptors have been massively used for mesh editing in various frameworks over the decade. For instance, the proposed method of Lipman et al. [11] allows the reconstruction of the edited surface by solving a linear system that satisfies the reconstruction of the local details in a least-squares sense. In addition, the differential coordinates are not invariant with respect of rotation and scaling, and it is definitively a potential drawback for animated geometry. Nevertheless, these deformation techniques mimic elastic deformation via a Laplacian framework. Another interesting modeling technique using a functional optimization system is presented by Nealen et al. [12] where a set of curves is used to create sophisticated models. Another powerful editing framework also based on a descriptive set of wires is offered by Gal et al. [13].

Dual Laplacian. *Dual Laplacian* systems describe a collection of approaches where the Laplacian is not directly expressed on mesh vertices. Early approaches similar to the work of Au et al. [15] motivated the use of *Dual Laplacian* system by reducing distortion in parametrization and geometry. Another mesh editing method working in the dual domain for regions of interest can be found in the work of Qiong et al. [16]. Unfortunately, Laplacians cannot satisfy all natural properties. The observation of the local surface behavior has been proposed recently by Sorkine et al. [14], where *as-rigid-as-possible* surface modeling is performed by the minimization of the deformed surface under local rigidity transformation constraints (see Fig. 2.2).

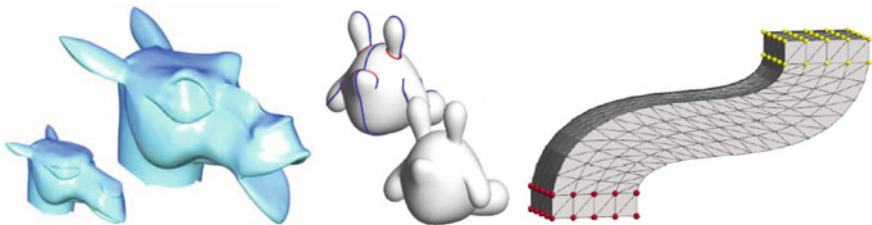


Fig. 2.2 Differential geometry: (from left to right) Laplacian mesh editing (image courtesy of [11]), fibermesh (image courtesy of [12]), and as-rigid-as possible surface modeling (image courtesy of [14])

2.2.3 Skeleton-based Parameterization

Articulation-Driven Skins. Introduced by the pioneering work of Magnenat-Thalmann et al. [17], *skeleton-driven mesh deformation* is a popular method thanks to its efficiency and simplicity. This class of deformation exploits the benefit of a generic hierarchy of kinematic articulation to conduct a local parametrization for a variety of enveloping skinned mesh. This underlying structure allows user intervention to control large scale motion with fewer characteristic of the model, but does not provide realistic muscle movement and skin motion. In skeleton-based animation, the bone length is naturally preserved because the parametrization is purely kinematic and relies on non-independent parameters. Besides, articulated deformation ensures rigidity by blending affine per-joint transformation in hierarchical manner. We also notice that this type of parametrization is well-adapted for humanoid, quadruped and vertebral living animals that are naturally articulated, but it is difficult to maintain the illusion of an underlying bone structure in presence of additional non-rigid surrounding layers like clothing or musculature.

Rigging and Skinning. Generally, skeleton animation is often associated with a rigging and skinning function. Firstly, the rigging step attaches the skeleton to the mesh by computing a per-joint influence weight distribution for all mesh vertices. Secondly, at each frame the skinning step deforms the mesh from its default pose according to the blending of joint transformation. The first automatic rigging method for skeletal deformation is introduced by Baran et al. [18] with two principal steps: *skeleton embedding* and *skin attachment*. The skeleton embedding phase focuses on placing all skeleton joints in the optimal location according to the enclosed volume of the input character shape. The skeleton structure adjustment is then performed by estimating a refined embedding thanks to a complex multivariate objective function with respect to the mesh medial axis and a discrete penalty function. Next, the skin attachment process finds a bone weight distribution from a single mesh pose for attaching mesh vertices to skeleton. The per-vertex weight distributions generated from the joint proximity and a surface-based diffusion equilibrium equation use

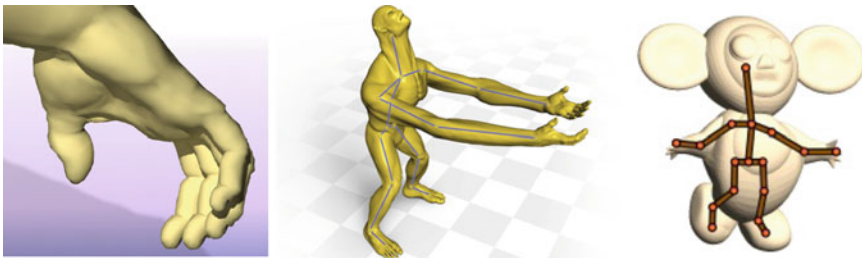


Fig. 2.3 Skeletal-based deformation: (from *left to right*) context-aware skeletal shape deformation (image courtesy of [20]), stretchable bone skinning (image courtesy of [21]), a skeleton is automatically adjusted and attached to a given skin using bone weight diffusion (image courtesy of [18])

Laplace’s diffusion. Finally, this interesting technique seems limited to single component mesh and required the character to be input in the same orientation than the mesh model with approximately same scale. Notwithstanding, the problem of building character rigs from multiple sources was already attempted by Miller et al. [19].

Advanced Skinning. Skinning techniques can be seen as a formulation for mesh deformation resulting of control points displacement. Another well-known skinning algorithm to avoid the undesired candy-wrapper effect by employing a dual quaternion formulation is proposed in Kavan et al. [22]. The skin-skeleton paradigm is also tackled by Weber et al. [20] with a skeletal-based system producing naturally-looking skin deformation that runs according the context of input shape examples (see Fig. 2.3). More recently, Jacobson et al. [21] extend the space of deformations for skinning for bending, stretching and twisting with only slight variant of existing skinning equations. More advanced physically-based skinning techniques have been also proposed recently for collision-free flesh-like character. For instance, a near-interactive simulation skinning technique is developed by McAdams et al. [23], to make physically-based elasticity practical for production of character rigging for skeleton-driven elasticity models with necessary control. The complexity of these techniques could be a potential drawback for representing complex surfaces deformation.

2.2.4 Space-based Parameterization

Free-Form Deformation. Introduced in 1986 by Sederberg and Parry [24], *Free-Form Deformation* (also known as FFD) is a pioneering class of classical solid modeling techniques where a rubber-like volumetric grid embeds an original shape of large-scale mesh or point-cloud to be altered. This first computer-assisted tool is based on space warping by blending the deformation at the control points. Conceptually explained by Coquillart et al. [25], the three dimensional lattice contains a rather small number of control points that encloses the model and the associated basis deforming functions should be smooth for aesthetics result. Manipulating the control points smoothly deforms the space enclosed in the lattice, and the embedded geometry deforms accordingly. Thus, the global-to-local deformation of the object is an intuitive appreciation with respect to the deformation. In addition, the original volume can be preserved using trivariate Bernstein polynomials. As seen in the work of Mac Cracken et al. [26] and Kobayashi et al. [27], the original FFD is generalized to various deformation approaches (arbitrary and cylindrical lattice), and various control point weighting factors.

Volumetric Preservation. Space-based parametrization offers improvements over surface-based approaches by allowing volumetric constraints. For instance, the *Volumetric Laplacian* studied by Zhou et al. [30] preserves volumetric properties for large deformations. Volume preservation is also addressed by Huang et al. [29] with a non-linear deformation energy projected onto the control mesh vertices.

Furthermore, there has been a terrific deal of work using an analogy to the traditional skeleton-based inverse kinematics but applied on boneless surfaces. Clearly, the important work of Der et al. [31] and Sumner et al. [28] introduce elegant mesh-based inverse kinematics techniques (see Fig. 2.4).

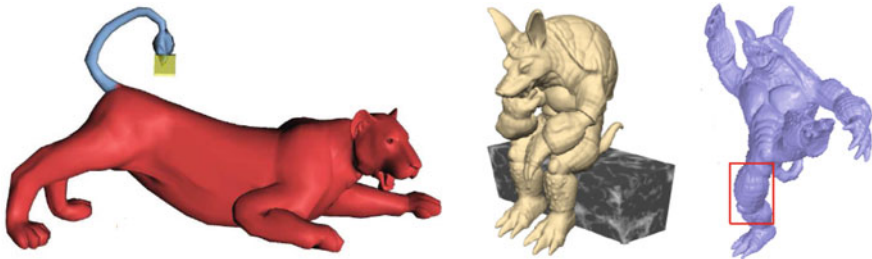


Fig. 2.4 Space-based deformation: (from *left to right*) mesh-based inverse kinematics (image courtesy of [28]), gradient domain mesh deformation (image courtesy of [29]), and volumetric Laplacian (image courtesy of [30])

2.2.5 Cage-based Parameterization

Cage-based Paradigm. In recent years, we notice an increasing interest for cages to manipulate 3D models efficiently. In this section, we review algorithms and properties of various cage-based deformation techniques. In order to deform a mesh, the cage is enhanced by a coordinate system, mainly based on the generalization of barycentric coordinates but also developed from more elaborated mathematics like Green’s theory. By cage-based deformation techniques, we mean emerging deformation techniques where a bounding cage deforms an enclosed triangular mesh in a volumetric fashion, according to a particular coordinate system (such as *Mean Value Coordinates*, *Harmonic Coordinates* and *Green Coordinates*). Additionally, a coordinate system expresses the influence of the control cage on the deformable enclosed model. In the following, we focus on studying the peculiarity of cage-based paradigm to encode surfaces. Thus, we detail now the *automatic cage building* process and impacting coordinate systems (see Fig. 2.5).

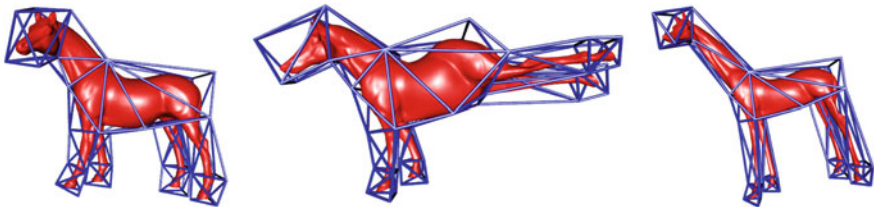


Fig. 2.5 Cage-based deformation: a low-dimensional deforming cage is used to drive natural deformations of a shape without relying on quasi-articulated priors (data courtesy of Ju et al. [32])

Automatic Cage Building. In cage-based approach, points controlling the deformations are the vertices of a specified polytope cage, bounding an enclosed template shape. A proper cage should envelop the original model without any self-intersection. The cage structure plays the role of a deformer. Thus, a cage is also considered better if it has significantly fewer vertices, while keeping global shapes similarities with the original model. Unfortunately, designing this coarse cage is the most tedious task for cage-based techniques. Solving the problem of automatic cage design can alleviate the user burden in setting up the cage. Relatively few cage-building techniques have investigated the complicated problem of generating a cage from a single model.

Even if few attention has been paid to the problem of automatic generation of coarse cage, promising solutions already exist. Importantly, a fully automatic method is presented by Xian et al. [33] to generate a coarse bounding cage that can keep the connectivity structure and significant features of the original mesh. In a first step, the bounding box is computed from the dense mesh by *Principal Analysis Component*. The box is slightly enlarged to ensure the cage to enclose the model. In a second step, the bounding box is discretized into voxels for finest shape enclosing, and the outer surface is triangulated. Finally, the output cage geometry is smoothed. They also proposed a formulation to compute the voxel resolution to have a desired sparse factor for the cage. Even if this technique provides a compelling direction to generate a cage, the resulting output geometry seems not a fairly abstracted coarse cage. In addition, Ben-Chen et al. [34] use an additional offsetting operation to enlarge the cage. Rather than a voxelization pass, the cage building technique of Deng et al. [35] propose an iterative simplification step using the well-known *Quadric Error Metric*. This procedure is followed by a self-intersection removal step. Topological intersections are detected using a marching squares method to avoid heavy topology repairing.

Most part of the time, the cage is built by hand or automatically obtained by adapting a bounding box with the condition that the cage should be coarse enough for easy manipulation. An automatic coarse cage-generation algorithm is also developed by Xian et al. [36], where the voxelization and principal analysis component are key ingredients to compute the oriented bounding box. Next, the OOB is sliced into multiple connected boxes. Each boxes are locally modified, reoriented, rescaled, and

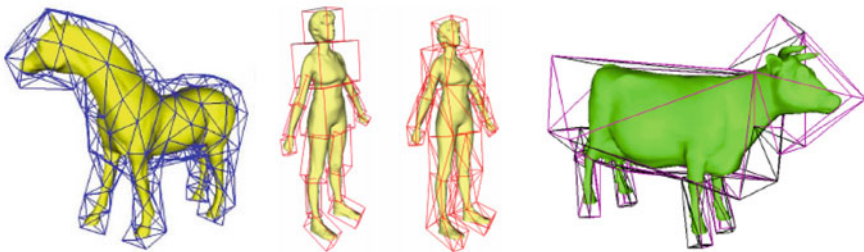


Fig. 2.6 Automatic cage generation: (from left to right) coarse bounding cage generation using voxelization (image courtesy of [33]), using oriented bounding box (images courtesy of [36])

merged together in order to generate the final cage mesh (see Fig. 2.6). This technique seems to provide meaningful cages as output for a given input model. In order to obtain a better abstraction of the envelop and a high degree of sparseness, Ju et al. [37] propose to reuse a library of pre-designed partial cage templates reassembled together with additional optimization to compose a complete cage. The resulting cage is well-adapted to the deformation behavior by ensuring tightness of the cage embedding. The technique of Kosinka et al. [38] relies on an injective shape deformation strategy using cube-like cages. Nevertheless, cube-like cages are not properly suitable for animated organic shape. Finally, the problem of rigidity for cage-based deformation is explored by Zhao et al. [39] with the incorporation of additional internal cage edges at the cage design step.

Coordinate Systems. There has been a great deal of work done in the past on developing techniques for deforming a mesh with generalized barycentric coordinates. Inspired from the pioneering work presented by Meyer et al. [40], cage-based methods are well-adapted for coherently deforming a surface by improving previous space-based deformation techniques. In particular, a cage parametrization allows model vertices to be expressed as a combination of cage-handles in conjunction with meaningful rigging functions. This family of methods tries to reach important properties: quasi-conformal mappings, shape preservation and smoothness. To animate the model, cage vertices are displaced and the vertices of the model move accordingly via a linear weighted combination of cage geometry parameters. By relying on a fixed cage connectivity as single geometric element, cage-based skinning brings more animator-friendly abstraction to manipulate a shape, other type of unstructured or degenerated geometries such as point clouds, polygon soup or particles. Recent years have seen an increasing interest for cage-based deformation: an emerging class of purely geometric space-based coordinates, widely used for controlling meshes enclosed in a flexible coarse bounding polytope. Then, we propose to review main coordinate systems (see Fig. 2.7).

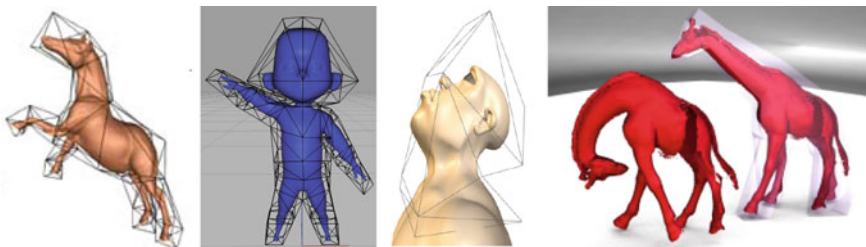


Fig. 2.7 Cage-based coordinates: (from *left to right*) mean value coordinates (image courtesy of [32]), harmonic coordinates (image courtesy of [41]), green coordinates (image courtesy of [42]), and variational harmonic mapping (image courtesy of [34])

Generalized Barycentricity. Various cage-vertex coordinate systems have been developed for space-based manipulations. Then, a rigging function expressed as generalized barycentric coordinates is associated with the cage-model paradigm to

describe realistic and plausible control behavior of the deformation. Motivated by the heavy use of interpolant function in surface parametrization, *Mean Value Coordinates* introduced by Ju et al. [32] is the first cage-vertex influence proposed as normalized generalization of homogeneous barycentric coordinates for cages. Unfortunately, negativity brought with *Mean Value Coordinates* leads to counter-intuitive deformation. To overcome this weakness, Lipman et al. [43] introduce the *Positive Mean Value Coordinates* by incorporating the visibility of the cage vertex from a point of the object in the coordinate system computation. More recently, *Poisson Mean Value Coordinates* developed by Li et al. [44] express *Mean Value Coordinates* with pseudo-harmonic properties. Finally, Langer et al. [45] also modify existing barycentric coordinates to *Higher-Order Barycentric Coordinates* by modifying *Mean Value Coordinates*. Generally, such coordinates have a lack of smoothness, and this drawback invited further exploration toward harmonicity.

Harmonicity. The importance of detail-preserving deformations and smoothness properties is the main issue for cage-based deformation. For this reason, harmonicity has attracted huge interest over the years to produce highly-detailed cage-based encoding. At first insight, a major research of DeRose et al. [46] explore *Harmonic Coordinates* for deforming characters for the movie Ratatouille at Pixar®. Then, the problem of designing and controlling volume deformations used to articulated characters are treated by Joshi et al. [41], with the introduction of *Harmonic Coordinates* significantly improves the deformation accuracy. In few words, *Harmonic Coordinates* are an elegant non-negative coordinates system for highly-detailed character animation, defined as the solution of Laplace’s equation in the form of volumetric heat diffusion in the cage interior subject to appropriately chosen boundary condition. Meanwhile, Raif Rustamov [47] proposes a very interesting *Boundary Element Method* formulation to speed-up and to reduce the storage of *Harmonic Coordinates*. Moreover, a variant for *Harmonic Coordinates* is proposed by Lin et al. [48] with smooth indirect binding using segmented thin-layers. Nevertheless, Ben Chen et al. [34] prefer to estimate harmonic maps using a set of harmonic basis functions computed from a collection of example poses. These *Variational Harmonic Maps* are insensitive to cage geometry and do not require a voxelization procedure (see Fig. 2.8).

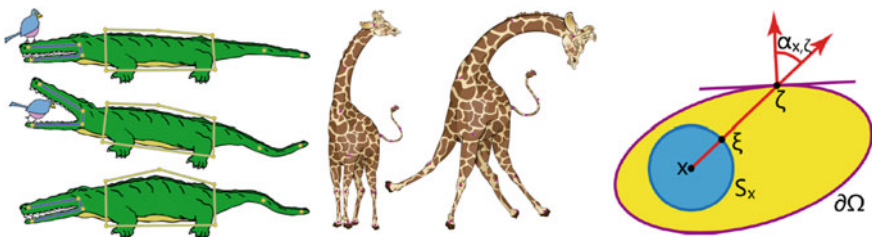


Fig. 2.8 Cage-based coordinates: (from left to right) bounded biharmonics weights (image courtesy of [49]), controllable conformal maps (image courtesy of [50]), and Poisson coordinates (image courtesy of [44])

Polyharmonicity. Furthermore, polyharmonicity has gained major attention recently and has demonstrated better shape-awareness and smoothness [51]. For instance, *Biharmonic Weights* described by Jacobson et al. [49] bring sparsity by restricting the weighting system to be *bounded biharmonic* with the use of a brush to define the region to be influenced. Nevertheless, automatic weight assignment method often generated weight matrices that required higher amount of storage space. Then, a Poisson-based weight reduction technique is proposed by Landerneau et al. [52] to produce geometric smooth weight reduction for cage-based deformation. Finally, late-breaking research of Jacobson et al. [53] demonstrate the usefulness of handle-based shape deformation using popular polyharmonic functions.

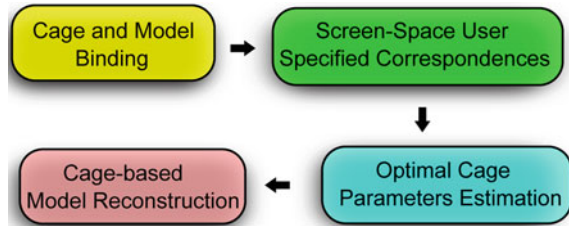
Quasi-conformality. Exact conformality is a very difficult property to obtain in conjunction with the cage-based strategy - and nearly impossible to formulate in 3D. Nevertheless, many improvements have been done toward better conformality with non-linear coordinates system. Outstandingly, Lipman et al. [42] introduce *Green Coordinates* leading to non-linear space deformations with shape-preserving properties. Then, *Green Coordinates* enforce quasi-conformal scale-preserving deformation and allow the use of partial cage. The use of partial cages are also investigated by Li et al. [54] with the introduction of cage-free local deformations using *Green Coordinates*. More importantly, we would like to point out that *Green Coordinates* ensure quasi-conformal mapping by integrating cage-face orientation, and overcome the lack of closed-form solution of *Harmonic Coordinates*. Even if the method of Weber et al. [50] produces pure conformal maps in 2D, unfortunately there is a scientific consensus that conformal maps in 3D nearly never exist.

Cage Reusability. The problem of reusing static cage templates from a small library is addressed by Ju et al. [37]. This technique proposes to rearrange pre-designed partial cage elements in order to form a desired final closed cage. Furthermore, learning deformation for a new shape from an existing pose is developed by Ben Chen et al. [55], demonstrating a cage-based method to transfer animation by employing spatial deformation transfer. Spatial deformation is also a useful tool for Chen et al. [56] for transferring deformation from one shape to another one using cage-based paradigm. Finally, the reuse of fairly coarse cages seems to be limited to deformation transfer only, and never associated before to the problem of capturing shapes.

2.3 Our Approach: Cage-based Shape Inverse kinematics

In this section, we propose the first formulation of *Inverse Cage-based Modeling* called *CageIK* where the problem of scalable mesh deformation is cast as an inverse variational problem ables to estimate cage parameters. In particular, we extend the *Inverse Kinematics* paradigm to classical cage-based deformation by recovering cage parameters from user-specified screen-space constraints. Simply put, we introduce the mathematical foundation of the core tool declined all along this book.

Fig. 2.9 Cage-based modeling: we introduce a novel hybrid method mixing surface-based and space-based deformation tools to perform cage-based inverse kinematics process



Boneless Mesh Deformation. *Boneless mesh deformation* techniques are well-known in the context of non-articulated surfaces deformation. This class of techniques is widely used to evolve a deformable triangulate mesh without any piecewise rigidity prior. Then, we conceive a novel *Dual-Laplacian* method for *Cage-Based Deformation Inversion* able to achieve large mesh deformation. We examine the feasibility of an *inverse strategy* using a detached cage control, without explicit rigid parts. For this purpose, our method combines cages and Laplacian techniques to interactively fit the bounding cage expressing the desired and unknown deformed model. Then, we exhibit a new estimation technique, involving an optimal reduced number of parameters suitable for animation re-use (see Fig. 2.9 for the pipeline).

Novel Hybrid Mesh Deformation. The idea of combining space-based deformation techniques with surface-based techniques was firstly enunciated by Daniel Cohen-Or [57]. Meanwhile, the lack of reusable parameterization for non-rigid surface invited us to abandon underlying articulated skeletons and to propose a novel approach based on a dual formulation of the Laplacian expressed on the cage structure itself. In addition, the main challenge is to deal with the high number of degrees of freedom provided by the coarse cage. Then, we aim to estimate cage parameters expressing a desired mesh from an input configuration. To realize this *Cage-based Inverse Kinematics* process, we cast the problem as a minimization problem for cage retrieval. This optimization process retrieves the pose in a least-squares sense from arbitrary sparse features constraints expressed directly on the enclosed mesh. Thus, specified constraints are transferred to the cage domain using the rig. In our system, we employ a Laplacian regularization on the cage to surfacically constraint the spatial deformation.

Cage Optimality. In contrast to previous approaches, we propose an easy-to-implement approach but yet efficient to estimate optimal cage parameters through minimization process. In our system, we employ cage-Laplacian allowing us to perform a coherent cage estimation under sparse constraints expressed on the enclosed mesh. The main benefit of our method lies in the fact that the parameterization is fully geometry-independent to the enclosed mesh resolution. Even if our work is closed to the one of Borosan et al. [58] around the idea of producing an hybrid mesh deformation, our work is still novel because we rely on a linear optimization problem. The described optimization algorithm minimizing the deformation energy is also closely related to the work of Chen et al. [56] for integrating the cage into a

minimization framework. However, the key contribution is to solve a sparse linear system that estimates the optimal cage parameters reproducing the desired deformation of the enclosed model. Besides, such constraints features are defined over the model and directly transferred into the subspace domain using the indirection of a coarse control meshes.

2.3.1 Laplacian-based Mean-Value Cage

Mean-Value Subspace Deformation. A cage is described as a continuous bounding surface enveloping a given volume where an enclosed model is immersed. Then, model vertices are expressed as a weighted linear combination of cage vertices. The weights are given by a set of generalized barycentric coordinates stored in a $m \times n$ deformation weights matrix denoted by $\mathcal{H}$, where m is the number of vertices for the model and n for the cage. We also denote by $g_k(i) \in \mathbb{R}$ the normalized blend weights representing the deforming influence of the k th cage vertex on the i th model vertex. Furthermore, it is also possible to deform an arbitrary point on the enclosed mesh written as a linear combination of the coarse mesh vertex position via a constant weight deformation. Then, a point inside the cage is represented by an affine sum of cage-handles multiplied by a specific weight function called the *cage coordinates*. A deformed point $v'_i \in \mathbb{R}^3$ is then obtained by the following equation.

$$v'_i = \sum_{k=1}^n g_k(i) \cdot c'_k$$

where $v'_i \in \mathbb{R}^3$ is the deformed Cartesian coordinates according to a vector of cage geometry $\{c'_1, \dots, c'_n\}$. In order to produce as-local-as possible deformation mapping on the enclosed surface. In this chapter, both coordinate systems (*Harmonic Coordinates* and *Mean Value Coordinates*) are mathematically suitable for an effective inverse modeling system. *Mean Value Coordinates* are a simple and powerful closed-form method, with linear properties for creating functions that interpolate values assigned to the vertices of a closed mesh. For more technical details, we refer the reader to the paper of Ju et al. [32] offering a pseudo-code algorithm as well as the mathematical formulation for this coordinate system.

Laplacian Cage. In our work, we rely on a Laplacian-type regularization applied on the cage structure instead of its traditional expression on the enclosed mesh itself (as seen in *Laplacian Mesh Editing* techniques). Then, we call *Laplacian Cage*, a coarse triangular bounding polytope enhanced with a *Discrete Laplacian Operator*. There is a number of approximations for the Laplacian operator, and each has their own advantages and disadvantages. At first glance, the three-dimensional Laplace's equation of an input function $f(x, y, z)$ is a partial differential equation enforcing the neighborhood curvature coherence by setting the second derivative of equal to zero.

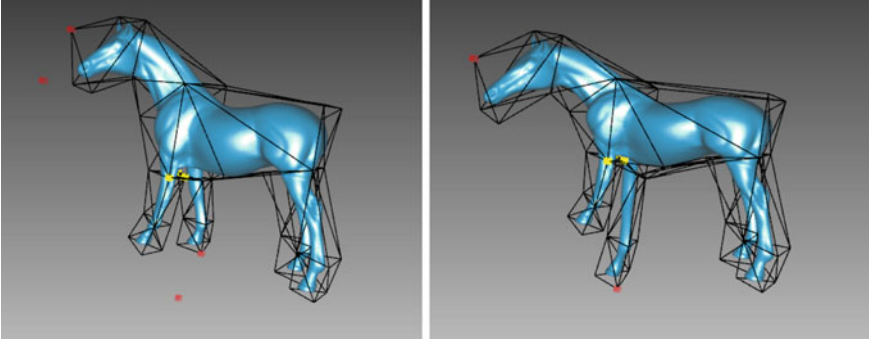


Fig. 2.10 Laplacian cage-based shape editing. The enclosed model can be directly edited by using traditional Laplacian mesh editing on the cage, followed by traditional cage-based deformation with the deformed cage (mean value coordinates are used in the example)

$$\Delta f = \nabla^2 f = \frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2} + \frac{\partial^2 f}{\partial z^2} = 0$$

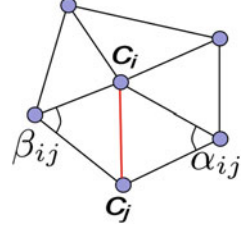
where (x, y, z) is a triplet of Cartesian coordinates, the symbol ∇^2 denotes the *Laplace Operator* and ∂^2 is the second partial derivatives, and Δ is called the *Laplacian Operator* or simply the *Laplacian*. An example of *Laplacian Cage-based Shape Editing* is proposed in Fig. 2.10, where the enclosed model is edited via a Laplacian-based deformation of the cage.

Cotangent Weighting System. To ensure better shape-awareness, we inject a weighting system inside the Laplacian encoding, similarly to previous techniques. Efficient vertex-based Laplacian deformation techniques always recommend cotangent weighting system for a better approximation of the local mean curvature. To define an accurate discretization, various researchers proposed to weight the Laplacian with a scale-dependence weighting system. Then, the tangential component of the Laplacian is eliminated. For instance, Desbrun et al. [59] describe three discrete approximations. Each of them which boil down to a weighted sum of the one-ring neighbors of a vertex. An interesting discretization of per-edge cotangent weight w_{ij} associated to the oriented edge (i, j) linking the given cage-handle c_i to c_j is defined as follows (see Fig. 2.11):

$$w_{ij} = \frac{1}{2\text{Area}(c_i)} (\cot(\alpha_{ij}) + \cot(\beta_{ij}))$$

where $\text{Area}(c_i)$ denotes the neighboring area of the adjacent faces of the cage-handle c_i . Moreover α_{ij} and β_{ij} denote the angles opposite to edge (i, j) . This weighting system takes advantage of the curvature flow. Given the fact that a coarse cage connectivity has often irregular triangles with degenerated aspect ratios, the use of the cotangent weights compensates the irregular triangulation. More generally, this is

Fig. 2.11 Cotangent weight:
We use a non-uniform weighting system for the Laplace Beltrami operator acting as cage-Laplacian



the better discretization of the *Laplace-Beltrami Operator* to functions on manifolds. Naturally, this discretization gives better results than the uniform weighting scheme by non-uniformly distributed mean curvature. The discrete version of the *Laplacian Operator* is also known as the discrete mean curvature normal operator because it evaluates the mean curvature normal.

Differential Cage Coordinates. In our work, we conservatively employ differential coordinates ($\hat{\delta}$ -coordinates) to encode the local surface properties of the cage. Then, the differential coordinates $\hat{\delta}_i$ of c_i are obtained by linear transformation of the cage-handles. In other words, differential coordinates of a given cage-handle are computed as the default difference between its absolute Cartesian coordinates and the center of mass of its immediate neighbors in the cage mesh:

$$\hat{\delta}_i = (\hat{\delta}_i^x, \hat{\delta}_i^y, \hat{\delta}_i^z) = \sum_{j \in N(i)} w_{ij} (c_i - c_j)$$

where $\hat{\delta}_i$ is a 3-vector packing the differential coordinates for the i th cage handle and w_{ij} is the per-edge cotangent weight. Instead of using absolute coordinates, we describe the cage mesh geometry using a set of differential $\hat{\Delta} = \{\hat{\delta}_i\}$.

Discrete Cage Laplacian. A discrete Laplacian operator is defined at each cage-handle by the weighted sum of the difference vectors between the given cage-handle and its adjacent neighbors. A discrete Laplacian operator $\mathcal{L}_C(\cdot)$ is defined by its linear action on handle-based functions. The Laplacian operator can be linearly approximated at each vertex by the Laplace operator given a simple local parametrization. The properties of $\mathcal{L}_C$ is encoded by the coefficients matrix w_{ij} . When the weights in the sum are represented using the Voronoi area and cotangents, the Laplacian vector approximates the mean curvature at the cage-handle.

$$\mathcal{L}_C(c_i) = c_i - \frac{1}{d_i} \sum_{j \in N(i)} c_j = c_i - \frac{1}{d_i} \sum_{k=1}^{d_i} c_{ik}$$

where d_i is the number of immediate neighbors $N(i)$ of i th cage handle (the degree or valence of i) and c_k is the k th direct cage-handle neighbor for a given cage-handle c_i .

Matrix-Form Cage-Laplacian. Thus, we propose a matrix-form expression discretizing this operator on the cage structure leading to the *cage-Laplacian* $\mathcal{L}_C$. First, we assume that i is the index of the inner cage-handle (also known as cage vertex) and j the index of a cage-handle belonging in the 1-neighborhood of i . Let A be the *Adjacency Matrix* of the cage connectivity, where its enteries A_{ij} are defined as follows:

$$A_{ij} = \begin{cases} 1 & (i, j) \in E \\ 0 & \text{otherwise} \end{cases}$$

where $(i, j) \in E$ means that cage-handles i and j are adjacent. We denote by E the set of cage edges. Let $D = \text{diag}(d_1, \dots, d_n)$ be the *Diagonal Degree Matrix*, where its enteries D_{ij} are defined as follows:

$$D_{ij} = \begin{cases} d_i & i = j, (i, j) \in E \\ 0 & \text{otherwise} \end{cases}$$

Given its discretized form, the cage is easily processed by differential geometry algorithm. So, the *Laplacian* of the cage, measuring the deviation of a cage-handle from the centroid of its neighbors, is represented by a $n \times n$ matrix called the *Laplacian matrix* L transforming the absolute cage-handle coordinates to relative coordinates. The matrix is obtained by subtracting the *Adjacency Matrix* A to the *Diagonal Degree Matrix* D , then its enteries L_{ij} are defined as follows:

$$L_{ij} = D_{ij} - A_{ij} = \begin{cases} -1 & i = j \\ d_i & (i, j) \in E \\ 0 & \text{otherwise} \end{cases}$$

The Laplace operator is also known as the divergence of the gradient at a scalar. Nonetheless, an important problem with the umbrella operator and a simple Laplacian discretization is they can provoke significant sliding and shape distortion when its applied to a coarse cage triangulation. Therefore, expressing the Laplacian on the cage can be seen as establishing a well-formed *Dual Laplacian* of the enclosed model.

2.3.2 Dual-Laplacian Surface Regularization

Regularization Energy Term. We propose to define a regularization term introducing additional information in order to solve an ill-posed problem or to prevent over-fitting. We determine the internal energy functional $E_{int}(c')$ that measures how smooth the cage is and how similar the deformed cage c' is to the original shape in term of local detail as follows:

$$E_{int}(c') = \left\| \mathcal{L}_C(c') - \hat{\delta}' \right\|_2^2$$

This functional guarantees smoothness on large deformation in order to preserve the subspace boundary properties without rigidity assumption. Ensuring such a property leads to guarantee global characteristic of the model linearly.

2.3.3 Screen-space Surface Constraints

Barycentric Anchors. Contrary to existing frameworks where positional constraints enforce vertices to move to a specific target 3D position, we prefer to enforce surface features that are not limited to the set of enclosed mesh vertices. In other to deform the bounding cage, positional constraints are defined on the model using barycentric anchor points (Fig. 2.12). A barycentric anchor q_l on a piecewise linear surface can be evaluated and described using a linear combination of the barycentric coordinates $\{\gamma_1, \gamma_2, \gamma_3\}$ associated to three vertices $\{v_1, v_2, v_3\}$ of their surrounding triangle T_l that contains this anchor point as follows:

$$q_l = \sum_{v_i \in T_l} \gamma_i \cdot v_i$$

In the scenario where the user directly specifies source and target positions over the enclosed mesh surface in screen space, dragged-and-dropped barycentric anchors always offer suitable and precise sparse positional deformation constraints. To estimate the target point in world-space coordinates, we compute the intersection point between the ray passing through the target screen point and the parallel plane to the screen plane defined by the source point.

GPU Barycentric Map. Barycentric information are collected in a floating map. Then, this barycentric map is used to establish 2D-to-3D correspondences, directly

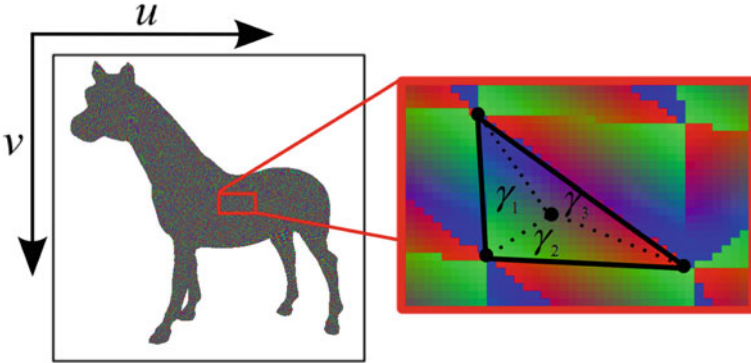


Fig. 2.12 GPU barycentric map: deformation handles are defined over the surface using barycentric coordinates expressed in image-space

by exploiting the GPU through a render-to-texture technique. More precisely, we store in a floating RGBA texture the index of the mesh face that reprojects into each screen-space pixel, and the associated barycentric coordinates corresponding to each pixel-based surface feature, according a given viewpoint (see Fig. 2.12). In other to deform the enclosed model via a space warping of the bounding cage, the user can drag-and-drop a collection of handles points over the enclosed mesh surface, directly by sketching a displacement vector on the screen for each handle. The origin point of this vector is called the source point and the final destination is called the target point. Finally, to estimate world space coordinates corresponding to the target point, we compute the intersection point between the ray passing through the target screen point and the parallel plane defined at the source point surfel. Thus, we can easily define a 2D-to-3D correspondence resulting in the association of a 3D target location coupled to a selected arbitrary point belonging to the surface space (see Fig. 2.13).

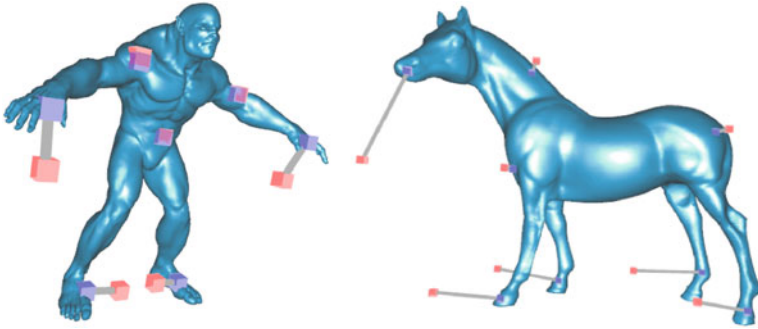


Fig. 2.13 Sparse deformation constraints: a small collection of user-specified geometric correspondences are defined in screen-space and backprojected over the surface in 3D. Such surface-based constraints serve to estimate a cage-based deformation field

Sparse Positional Constraints. Now, we provide a formulation expressing the Cartesian coordinates of a point q_l over the model only in term of the cage parameters:

$$q_l = \sum_{v_i \in T_l} \sum_{k=1}^n \gamma_i \cdot c_k \cdot g_k(i)$$

Then, we denote by q'_l the Cartesian coordinates position of the user-specified target point, associated to q_l to form a positional constraint. The last equation is a key component of the proposed method for the handling interaction. The transfer of surface constraints into the volumetric domain exploiting the cage indirection is expressed by this function. In other words, the last formulation permits to express surface constraints directly in terms of cage parameters linearly using an inverse near-conformal mapping, motivating the idea of boneless inverse kinematics.

Data Energy Term. We determine the external energy functional $E_{ext}(c')$ that measures the smoothness the cage under l positional constraints as follows:

$$E_{ext}(\mathbf{c}') = \sum_{j=1}^l \left\| \mathbf{q}'_j - \mathbf{q}_j \right\|_2^2$$

2.3.4 Indirect Dual-Laplacian Cage-based Fitting

Least-Squares Cage. After the introduction of the key components of our method, in this section we propose to develop the core of our approach including the linear minimization process. During the minimization process, the cage is seen as a connectivity mesh and feature constraints are seen as external deformation features. Our surface-and-space based deformation technique preserves the mesh spatial coherence. The geometry of the cage can be efficiently reconstructed indirectly from their harmonic coordinates and relative coordinates by solving a system of linear equations. We cast the problem of deformation as least-squares Laplacian cage reconstruction process using a consistent minimization approach of an objective function requiring linear constraints such as the positional edited constraints. By transposing ideas presented by Sorkine et al. [60] to the cage domain, our cage parameters recover the sparse pose feature by minimizing an objective function in a least-squares sense in order to fit a continuous volume. Then, the geometry of the desired model is simply obtained by generating its position vertex according to the reconstructed cage parameters obtained on the concept of *Least-Squares Cage* (see Figs. 2.14, 2.15).

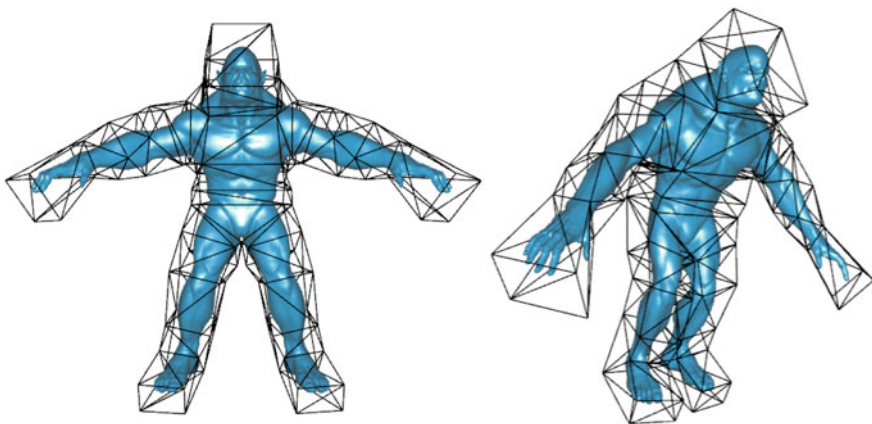


Fig. 2.14 Inverse cage-based modeling: subspace-regularized and space-based deformation under user-specified sparse constraints. A scanned model of an horse is enclosed in a coarse cage (*left*), the user defined a set of 3D correspondences in screen-space. (*middle*), then our cage fitting procedure retrieves the cage expressing the desired space-based deformation (*left*). We use Harmonic Coordinates for this example

Given differential coordinates, the Laplacian operator of the default cage, the generalized barycentric weights $g_k(i)$ according the cage and the model at the default

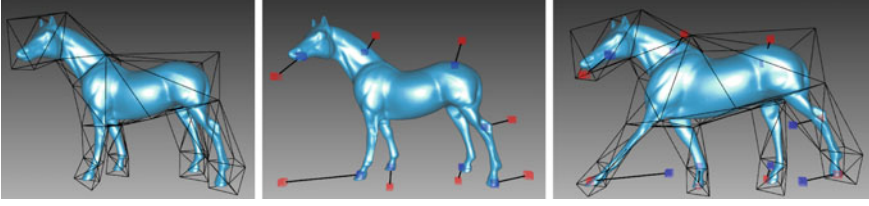


Fig. 2.15 Constraint-based scalable deformation: given a default cage-based shape (*left*), and user-specified constraints, our system retrieves a least-squares cage expressing the desired deformation in a scalable manner with shape resolution-independence (*right*)

pose, and a several 2D sparse surface constraints, the absolute coordinates of the model geometry can be reconstructed by estimating the absolute coordinates of the cage geometry. The combination of the differential coordinates and harmonic coordinates allows the reconstruction the edited surface by solving a linear system that satisfies the reconstruction of the local detail in least-squares sense.

Least-Squares Minimization. Since no exact solution generally exist, our linear least-squares system reconstructs the geometry of the coarse mesh, allowing to express the enclosed mesh using a linear cage-based deformation process. The key component of our inverse deformation algorithm is a least-squares minimization. We can reformulate the overall energy, leading to an overdetermined linear system to extract the cage parameters as follows:

$$\min_{\forall k, c'_k} \left(\alpha \sum_{k=1}^n \left\| \mathcal{L}_C(c'_k) - \hat{\delta}'_k \right\|_2^2 + \beta \sum_{j=1}^l \left\| q'_j - \sum_{v_i \in T^j} \sum_{k=1}^n \gamma_i \cdot c'_k \cdot g_k(i) \right\|_2^2 \right)$$

We denote by $\|\cdot\|_2$ the L_2 norm. Note that the first term of the energy (called the *smoothness term*) preserves global details of the cage and ensure a pleasant deformation under sparse constraints. The second term (called the *data term*) of the energy enforces the position of vertices to fit the desired model defined by positional constraints. Independent linear positional constraints are easily represented by separate equations for each three dimensions. The system can be weighted by both scalar $\alpha > 0$ and $\beta > 0$ to penalize or advantage each term with respect to the other one. In our experiments, we use $\alpha = \beta = 1$. Then, this linear least-squares system can be rewritten in the following matrix form:

$$\|\mathbf{A} \cdot \mathbf{X} - \mathbf{B}\|_2^2 = (\mathbf{A} \cdot \mathbf{X} - \mathbf{B})^T (\mathbf{A} \cdot \mathbf{X} - \mathbf{B}) = \mathbf{X}^T \mathbf{A}^T \mathbf{A} \cdot \mathbf{X} - 2\mathbf{B}^T \mathbf{A} \cdot \mathbf{X} + \mathbf{B}^T \mathbf{B}$$

where $\mathbf{A}$ stacks the cage-Laplacian, $\mathbf{B}$ stores the differential coordinates and target locations and $\mathbf{X}$ is the unknown location of cage vertices. To formulate a closed-form solution, we take partial derivatives with respect to the unknown vector and set them to zero. We then obtain the normal equation as follows:

$$2\mathbf{A}^T \cdot \mathbf{A} \cdot \mathbf{X} - 2\mathbf{A}^T \cdot \mathbf{B} = 0$$

that we simplify in that manner:

$$\mathbf{A}^T \cdot \mathbf{A} \cdot \mathbf{X} = \mathbf{A}^T \cdot \mathbf{B}$$

Our desired solution $\mathbf{X}$ can be written:

$$\mathbf{X} = \left(\mathbf{A}^T \cdot \mathbf{A}\right)^{-1} \cdot \mathbf{A}^T \cdot \mathbf{B}$$

Desired Model Reconstruction. To our best knowledge, the simple global optimization component of our framework does not already exist in the literature with such constraints to minimize. The overall energy-based warping performed by our technique recovers space deformation during the editing phase. After the cage retrieval process, the geometry of the desired enclosed model is reconstructed with linear combination function of cage geometry parameters related to the new estimation, preserving the fix connectivity of the cage. The system easily represented as a separate least squares system for each dimensions (x , y , z). The availability of iterative sparse linear solvers or direct method for solving the normal equation makes the computational kernel of our retrieval algorithm very efficient and simple to implement [61].

2.3.5 Experimental Results and Evaluation

Cage-based Inverse Modeling. This section describes our experiments using the proposed *Cage-based Inverse Modeling* system. Built upon the well-known least-squares formulation, our framework proposes a numerically stable and interactive mechanism to estimate new cage configuration. Our effective framework could serve for various geometric applications. Thus, we demonstrate the feasibility, stability and validity in practice with an experimental application as illustrated in Fig. 2.16. The cage is encoded with a winged-mesh structure to access faster to the one-ring-neighborhood of cage-handles and to efficiently compute the Laplacian coordinates. In the context of user-driven approach, we apply our algorithm by specifying sparse screen-space positional constraints over the enclosed surface. The indirect cage estimation improves the computation of the modeling thanks to the small system size evolving the cage indirection. The solving process of examples shown in Fig. 2.16 involving dense polygon meshes was generated under 100 μ s. Alternatively, even if we think that better control of the object is possible if the cage is subdivided, cage coarsity must remain a requirement to pretend being *cage-based*. Finally, the main computational cost of our method is dedicated to the cage-model rigging and the solving process for the objective functional using a direct linear least-squares system. The running time is therefore most strongly dependent on the number of cage vertices.

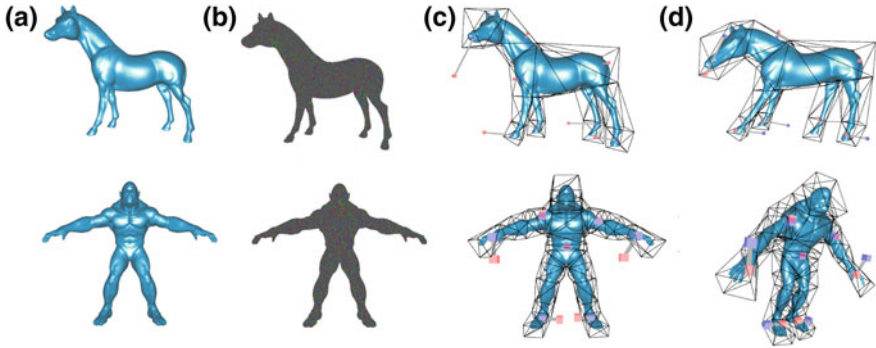


Fig. 2.16 Cage-based inverse modeling: the default enclosed model (a), its corresponding GPU barycentric map (b), user-specified constraints (c), the resulting cage-based deformation (d)

Sketching Positional Constraints. Additionally, we have developed an intuitive user interface that allows the user to model a new sophisticated model by sketching sparse positional constraints in the screen-space over the enclosed surface. This interaction technique provides a natural way to interact with basic input deformation features. Then, the indirect cage estimation improves the computation of the modeling because of the small system size involving the cage indirection. Our barycentric anchor system and surface-to-cage constraints transfer make our technique potentially interoperable with the emergent multi-touch surface interaction technology.

Benefits. Since global modeling operations naturally require high-level surface representations, our method improves the effectiveness of deformable surface by offering the possibility of encoding global shape deformation with respect of local influence. Then, our framework allows artists to re-use the estimated parameters during the modeling process with high flexibility. By solving the presented global linear least-squares system, this constraint-based deformation provides an efficient and robust computational framework. Moreover, it seems easy to integrate other type of linear deforming constraints. The linearity of the proposed objective functional makes the processing highly efficient. Nevertheless, our framework is not limited to *Mean Value Coordinates* or *Harmonic Coordinates* and could be adapted with other type of linear coordinate systems, such as *Biharmonic Coordinates*. Strong interest for cage-based deformation invites us to improve our framework in the future. We remark that the cage topology and geometry encapsulate a soft deformation resistance that allows us to produce a natural deformation for large deformation. In addition, the global explicit parameterization offers multiresolution editing.

Effectiveness. There are some limitations with our method that are theoretically derived from Laplacian and cage-based properties. The main drawback of our proposed technique arises from combining linear cage-based coordinates with Laplacian regularization. We remark that it is hard to obtain volume preserving with generalized barycentric coordinates. The major difficulty for these methods is to find efficient mappings which do not induce large distortion such as volume shrinking to the model

without enforcing additional constraints. Unfortunately, our system is not capable of ensuring isometry over the surface. Accordingly, volume constraints should be imposed to reduce such undesirable effects. Another notable limitation is the cage design. It is exceedingly tedious and time-consuming to extract automatically a reasonable default coarse bounding cage. An arbitrary cage may fail to express every desired pose correctly according to a given binding process.

2.3.6 A Mathematical Study

Cage-based Linear Algebra. In our technique, we do not apply the Laplacian directly on the enclosed mesh itself in order to reduce the computation cost and memory space, but more importantly to turn the shape modeling process into a scalable task. Notwithstanding, the deformation seems to preserve the local properties in the same visual fashion than if the *Laplacian Operator* will perform was defined over the enclosed model. Notwithstanding, we would like to develop a mathematical study to demonstrate that applying the Laplacian Operator on the cage structures is not mathematically equivalent (even with approximated derivation) to applying the *Laplacian Operator* on the enclosed model. In complement to this demonstration, we also propose some additional mathematical assertions concerning the cage-based linear algebra.

Subspace Decomposition. A vertex $p \in \mathbb{R}^3$ of the enclosed model can be decomposed into a cage-based subspace linear basis (only with near-conformal mapping properties), according to its associated to a set of rigging functions $h_k(p) : \Omega \mapsto \mathbb{R}^+$ (the rigging function can be defined using *Mean Value Coordinates* or *Harmonic Coordinates*) as follows:

$$p \approx \sum_{k=1}^n c_k \cdot h_k(p)$$

The well-known non-exact conformality of the cage-based rigging in three dimensional leads to an approximated projection of a given 3D point expressed in global euclidean into the cage-based subspace basis defined as the vector of n cage-handles $\{c_1, \dots, c_n\} \in \mathbb{R}^{3 \times n}$.

Subspace Extrapolation. As shown by Rustamov [47], it is possible to define the extrapolation of cage-based coordinates (only demonstrated for *Harmonic Coordinates*). For any default point p and its deformed corresponding point p' , we can write:

$$p' - p = \sum_{k=1}^n h_k(p) \cdot c'_k - \sum_{k=1}^n h_k(p) \cdot c_k = \sum_{k=1}^n h_k(p) \cdot (c'_k - c_k)$$

Blending into the Subspace. Given $\alpha \in \mathbb{R}^+$ and $\beta \in \mathbb{R}^+$ two factors and two points $v_a \in \mathbb{R}^3$ and $v_b \in \mathbb{R}^3$, we can write:

$$(\alpha \cdot v_a + \beta \cdot v_b) = \left(\alpha \cdot \sum_{k=1}^n c_k \cdot h_k(a) + \beta \cdot \sum_{k=1}^n c_k \cdot h_k(b) \right)$$

The last formulation leads to the following expressions:

$$\Leftrightarrow (\alpha \cdot v_a + \beta \cdot v_b) = \left(\sum_{k=1}^n \alpha \cdot c_k \cdot h_k(a) + \sum_{k=1}^n \beta \cdot c_k \cdot h_k(b) \right)$$

$$\Leftrightarrow (\alpha \cdot v_a + \beta \cdot v_b) = \sum_{k=1}^n c_k \cdot (\alpha \cdot h_k(a) + \beta \cdot h_k(b))$$

Enclosed Mesh Laplacian into the Subspace. Now, let's write the Laplacian of the enclosed mesh in term of the cage parameters. If we denote by $v_i \in \mathbb{R}^3$ and $v_j \in \mathbb{R}^3$ two different enclosed mesh vertices, $N(i)$ the one-ring neighborhood of v_i , and given the fact the *Laplacian Operator* of the enclosed mesh is written as follows:

$$\mathcal{L}(v_i) = v_i - \frac{1}{d_i} \sum_{j \in N(i)} v_j$$

where d_i is the number of direct vertices neighbors $N(i)$ for v_i . Then, we can rewrite this operator exclusively in term of the cage parameters, as follows:

$$\Leftrightarrow \mathcal{L}(v_i) = \left(\sum_{k=1}^n h_k(i) \cdot c_k \right) - \left(\frac{1}{d_i} \sum_{j \in N(i)} \left(\sum_{k=1}^n h_k(j) \cdot c_k \right) \right)$$

In addition, we can also rewrite this closed-form expression:

$$\Leftrightarrow \mathcal{L}(v_i) = \left(\sum_{k=1}^n h_k(i) \cdot c_k \right) - \left(\sum_{j \in N(i)} \left(\sum_{k=1}^n \frac{1}{d_i} \cdot h_k(j) \cdot c_k \right) \right)$$

Using previous assertions, we can simplify the last expression as follows:

$$\mathcal{L}(v_i) = v_i - \frac{1}{d_i} \sum_{j \in N(i)} v_j = \sum_{k=1}^n c_k \cdot \left(h_k(i) - \frac{1}{d_i} \sum_{j \in N(i)} h_k(j) \right)$$

Then, we can conclude that the cage-Laplacian has a multi-ring influence over the enclosed shape, and not exactly *model-Laplacian* equivalent because of its strong

dependence to cage-handles and associated rigging function. In the general case:

$$\mathcal{L}(\cdot) \neq \mathcal{L}_C(\cdot)$$

The resulting expression projects the mesh-Laplacian into the cage-based subspace with a minimal number of separable terms. This demonstration clearly shows that ensuring the Laplacian of the cage is not mathematically equivalent to ensure the Laplacian on the enclosed mesh.

Constraints Projected into the Subspace. Now, let's write the enclosed mesh-Laplacian term only in term of cage parameters.

$$\arg \min_{\forall i, v_i} \left(\sum_{i=0}^m \|\mathcal{L}(v_i) - \delta_i\|_2^2 \right) \Rightarrow$$

$$\arg \min_{\forall k, c_k} \left(\sum_{i=0}^m \left\| \sum_{k=1}^n c_k \cdot \left(h_k(i) - \frac{1}{d_i} \sum_{j \in N(i)} h_k(j) \right) - \delta_i \right\|_2^2 \right)$$

2.4 Beyond the Cage: A Generalization to Skeletal Structures

Generalized Laplacian Subspace. In this section, we show that our *Laplacian Subspace* technique can be generalized to other types of underlying animatable subspaces that are not purely geometric and two-manifold, such as a skeletal structure. By generalizing our spatial relationship principle on skeletal joint-handles, we show that the *Laplacian Regularization* can modify the rigidity of an input rigid skeleton. We also demonstrate a useful interactive application for subspace reediting.

Skeletal Structures. Then, we describe a new and simple approach to edit traditional skeleton of animation with joint-based Laplacian-type regularization, in the context of exaggerated skeleton-based character animation. Despite decades of research, interactive character animations still have a lack of flexibility and editability in order to re-use real vertebral motion. A major problem for artists in production is to enhance the expressiveness of classical motion clips by direct manipulation of the underlying skeletal structure. Relatively small number of researchers present an approach for processing non-rigid effects on motion data in [62–64]. However existing techniques often avoid dealing with the potential of skeletal-based optimization, while preserving the joint coherence and connectivity. Besides, range of living creature in real-world have the flexibility to stretch to extreme positions and squash to astounding shapes. It can also be noticed that *squash-and-stretch* effects are easier to realize in traditional animation rather than mocap-based computer generated animation. For this reason, *Ratatouille* a Pixar® movie, did not use a rigid skeleton, while abandoning motion capture to reach such essential non-ultra realistic appeal.

2.4.1 Differential Skeletal Editing

Differential Joint Coordinates. The idea of the motion capture is to use sensor placed on the subject, and to collect data that describe their motion while they are performing motions. The pose of an articulated figure can be specified by its joint configuration in addition to the position and root orientation segment. Skeleton of animation $\mathcal{S} = (\mathcal{J}, \mathcal{B}, \mathcal{M})$ is composed of a hierarchy of joints enriched with motion data, organized in frames. $\mathcal{S}$ is made of a set $\mathcal{J}$ of n joints, a set $\mathcal{B}$ of bones connecting joints, and a set $\mathcal{M}$ of k motion frames. Two joints i and j are connected into a unique bone only if $(i, j) \in \mathcal{B}$. Motion data consist of bundle of signal defined as a continuous function $\mathbf{f}(t)$. Compiled global rigid transformation matrix from captured motion data associated to the i th joint at the frame t is denoted by $\mathbf{M}_i^t \in \mathbb{R}^{4 \times 4}$. The global location $\mathbf{p}_i^t$ of the i th joint at the frame t is then the homogeneous zero transformed by the sequence of transformation and can be written as: $\mathbf{p}_i^t = \mathbf{M}_i^t \cdot [\mathbf{0} \mid 1]^T$. We define the *Differential Joint Coordinates* δ_i^t of joint i at frame t as follows.

$$\delta_i^t = \mathbf{p}_i^t - \sum_{j \in N(i)} \frac{1}{d_i} \left((\mathbf{M}_i^t - \mathbf{M}_j^t) \cdot [\mathbf{0} \mid 1]^T \right)$$

Skeletal Graph Laplacian. The degree of the i th joint denoted by d_i is equal to the number of joints linked to i . The set of immediate adjacent joints to i is denoted by $N(i) = \{j \mid (i, j) \in \mathcal{B}\}$. In addition, the skeleton topology is represented as an open directed acyclic graph. So, we introduce the *Skeletal Graph Laplacian* L_S by $L_S = D - A$ where D is the diagonal matrix of joint degrees and A is the adjacency joint matrix. We denote by $\mathcal{L}_S(\cdot)$ the per-joint *Uniform Laplacian Operator* applied on the skeleton graph structure. The entries of the corresponding square symmetric $n \times n$ matrix L_S are setup as follows:

$$L_S[i, j] = \begin{cases} 1 & \text{if } i = j \\ -1/d_i & \text{if } (i, j) \in \mathcal{B} \\ 0 & \text{otherwise} \end{cases}$$

We focus on adding non-rigid effects to an existing captured skeletal structure, while preserving its consistency and original connectivity. In our approach, we are referring to the well-known Laplacian shrinking effects (i.e. shearing and stretching distortion) in order to apply non-rigid warps over the rigid skeleton topology.

2.4.2 Rigidity-Breaking Skeletal Optimization

Differential-Aware Stretching. Our algorithm takes an arbitrary input articulated motion signal and perturbs it in other to make it more stretch-like. In order to hack the rigidity, we prefer to deal with the skeletal structure as Euclidean joint coordinates.

Our technique has single key component: the global joint location optimization. At the beginning of pre-optimization, we reformulate the *Squash-and-Stretch* problem as *skeletal adaptation optimization*. Given the fact it is nearly impossible to get a plausible squash-and-stretch by working exclusively in Euler space with *inverse kinematics*, we prefer to optimize the whole skeletal structure in term of global joint location of a pre-animated pose satisfying stretching constraints. The core algorithm of our technique relies on the *Skeletal Graph Laplacian*, with the aim to ensure spatial relationship of joints under sparse differential-aware stretching features over the whole joint hierarchy. As a result, the driving idea is to employ fast and accurate skeleton fitting process, guided by kinematic-free constraints.

Least-Squares Skeleton. At each frame, the initial solution is the Euclidean parameters of the animated pose, generated by forward kinematics. Thus, *pose estimation* recovers the pose by minimizing an objective function that is a combination of penalty and data terms. Smoothness term is required for making the warp field regular. To control a desired pose, the user inputs the targeted global joint positions $\mathbf{q}_i^t$ for a collection $\mathcal{C}$ of edited joints. We conserve the Laplacian coordinates δ_i^t for each joint i in the skeleton hierarchy. The reconstructed positions $\hat{\mathbf{p}}$ of the skeleton joints in world space coordinates are obtained by solving the following quadratic minimization problem separately from each Euclidean dimension:

$$\underset{\hat{\mathbf{p}}^t}{\operatorname{argmin}} \left(\sum_{l \in \mathcal{C}} \|\mathbf{q}_l^t - \hat{\mathbf{p}}_l^t\|^2 + \sum_{i=1}^n \|\mathcal{L}_{\mathcal{S}}(\hat{\mathbf{p}}_i^t) - \delta_i^t\|^2 \right)$$

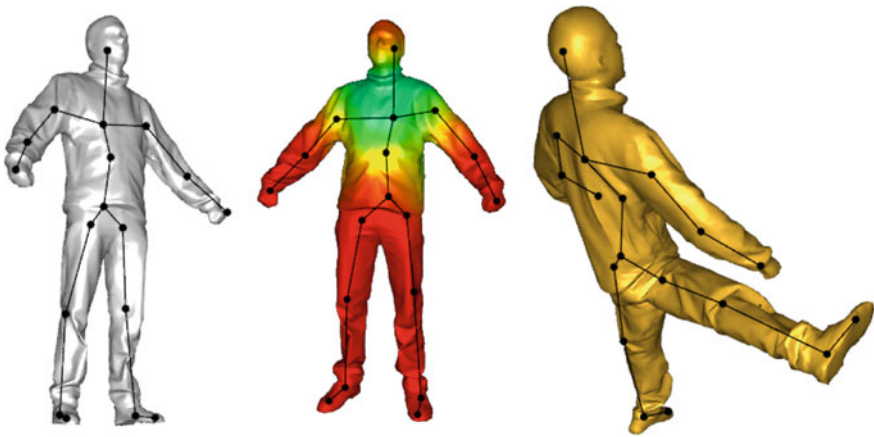


Fig. 2.17 Laplacian skeletal editing: optimizing traditional skeletal animated structure with differential-aware bone stretching effects allows animators to efficiently re-use previously captured MoCap data in the context of animation productions. We increase Squash-and-Stretch effects on existing realistic Basketball Dunk, Dancing, Horse Jump, Karate Kick-Off and motion clips (from left to right hand side). Original and intermediate edited poses are displayed in transparency and resulting edited poses in superposition

Automatic Stretching Constraints. Global joint location is estimated by minimizing the sum of squared difference between the data-driven pre-animated pose and input features cues. In order to avoid purely translation effects provoked by a single dragged-and-dropped joint, we fix by default the root joint, acting as skeleton gravity center. The root joint is generally located in a place where many bones come together. In our framework, bone elongation at frame t can also be automatically established along the bone direction for a selected joint l having the k th joint as parent with a time-dependent scaling factor α_t :

$$\mathbf{q}_l^t = \mathbf{p}_l^t \pm \alpha_t \cdot \frac{(\mathbf{p}_l^t - \mathbf{p}_k^t)}{\|\mathbf{p}_l^t - \mathbf{p}_k^t\|}$$

Poisson Skeletal Solver. Energy terms can be equilibrated by a weighting system in future work. Using such refined constraint formulation in conjunction with the Laplacian formulation on the bone structure is motivated by the fact that *Squash-and-Stretch* can be accomplished by differential scaling in Euclidean coordinates system. Hence, this minimization problem can be solved efficiently in least-squares sense based on the succeeding expression:

$$\mathbf{A} \cdot \mathbf{X}' = \mathbf{B}$$

where the matrix $\mathbf{A}$ stacks the skeletal-Laplacian and the matrix $\mathbf{B}$ stores the differential joint coordinates. This sparse editing technique performs spatio-temporal numerical optimization on MoCap data. Moreover, the global location of joints stored in the column matrix $\mathbf{X}'$ can be therefore found by solving at real-time rate the following small linear system that does not need to:

$$\mathbf{X}' = (\mathbf{A}^T \mathbf{A})^{-1} \mathbf{A}^T \mathbf{B}$$

2.4.3 Experimental Results

Laplacian Skeletal Subspace. To demonstrate the stability and the usefulness of our *Laplacian Skeletal Subspace* techniques, we have implemented an advanced and reproducible pipeline using only C++ and OpenGL. In all of our experiment, just a single editing feature is sufficient to adapt the whole skeletal pose in coherent manner, as shown in Fig. 2.18. Moreover, results have shown that the proposed technique is stable numerically enough to probably be used in low-budget productions, especially for processing sporty motion clips, as illustrated in Fig. 2.17. We have tested the robustness of our method on a corpus of motion clips with success. Few examples are shown in Fig. 2.18. In addition, we developed a simple interaction technique coupled with depth first search to handle the closest joint.

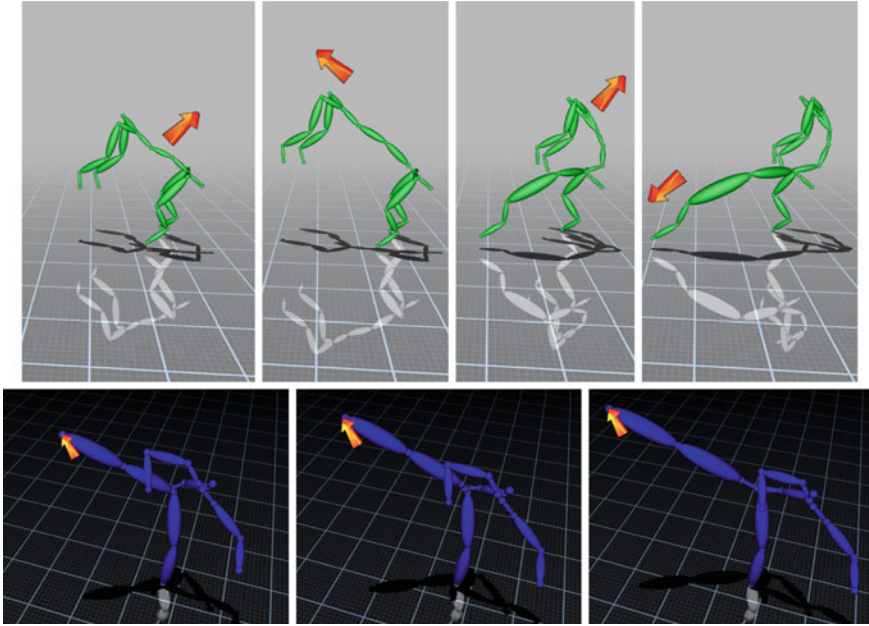


Fig. 2.18 Interactive skeletal editing: a single editing feature is sufficient to produce pleasant non-rigid gait style over the whole skeletal pose in coherent manner for a horse jump (*top row*) and a karate kick-off (*bottom row*)

Skeletal Shape Control. Even if human perception system tends to focus on kinematic parameters rather than on local features cues, real-life creatures actually elongate and stretch due to the elasticity of tendons and muscles. Consequently, breaking the rigidity of the underlying armature adds a pleasant realism with rubber-like effects. The resulting animated structure looks more pleasant while remains rigid in motion. Our technique could be potentially used in commercial 3D animation systems

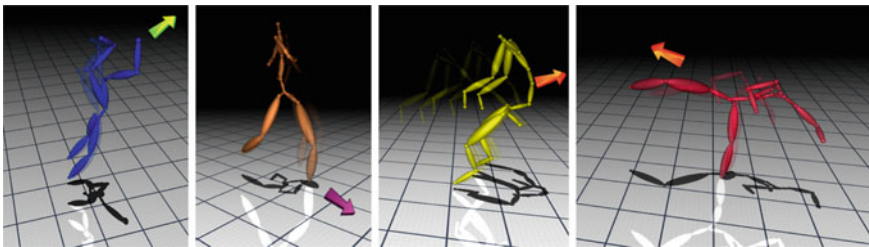


Fig. 2.19 Skeletal shape editing: our least-squares Skeleton can be used to drive the deformation of a static template mesh (*left*) having the same default topology characteristics. According to a given rigging function (*middle*), the deformed shape is obtained by a modified skeletal-based skinning operator (*right*)

to easily achieve non-rigid, rigged-and-skinned animation, skeletal shape control and elastic skin deformation from MoCap data. Our method yet offers the potential ability to control stretchable effect while allowing smooth behavior of skeletally-guided skin deformation, as far as the rigging function is smooth (see Fig. 2.19).

2.5 Discussion and Conclusions

Low-dimensional Parametrizations. In this chapter, we have presented a concise overview of various low-dimensional parametrizations useful to process sparse deformation constraints on a given static surface. Next, we have introduced an inverse modeling framework based on the estimation of a surfacically-constrained space-deformation mapping through the prism of numerical optimization. To the best of our knowledge, our well-designed technique does not previously exist in the literature. Then, we exhibit an external parameterization involving a reduced number of parameters suitable for maintaining one-to-one correspondence during the deformation phase. The dimension reduction offered by the cage connectivity is a strength for surface encoding and for further network transmission of deformation parameters.

CageIK. In the first contribution, a mix of generalized barycentric coordinates and Laplacian coordinates are injected inside a linear minimization framework to reconstruct a desired global pose from a default enclosed mesh. This *Dual Laplacian Cage-based Mesh Editing* technique allows users to produce visually pleasing deformations. Furthermore, the linearity of the core objective functional makes the processing highly efficient and improves the effectiveness of deformable surface computation. Obviously, cage connectivity and its location encode non-trivial soft kinematic constraints, because the cage topology drastically restricts the deformation under constraints. This could be particularly attractive for producing pseudo-kinematic space deformation over non-rigid surfaces without explicit skeletal constraints to be solved numerically.

Our method offers the possibility to encode global deformation of the shape with respect of local shape influences. Furthermore, we demonstrate the strength of near-conformal mapping and differential geometry to reconstruct an enclosed mesh by estimating its corresponding and desired cage locations. In addition, this cage-based approach is a turn-key tool for artist in order to generate, reuse, modify the resulting set of parameters, interact with the resulting data easily or potential deformation transfer. That is worth mentioning that our framework is not mathematically restricted to a specific coordinate system as-far-as the selected cage-based system implies linear constraints. The algorithm is effective, conceptually simple and comprehensible enough to be widely implemented and tested with previous mesh-based frameworks.

Laplacian Skeletal Editing. We demonstrate the strength of Laplacian and differential editing on underlying animatable structures not limited to bounding cages. So, we have also introduced a novel, accurate, robust, easy to use, skeletal editing and interactive techniques allowing unskilled animators to manipulate arbitrary skeletal

8. M. Wardetzky, S. Mathur, F. Kälberer, E. Grinspun, Discrete laplace operators: no free lunch, in SGP'07: Proceedings of the fifth Eurographics Symposium on Geometry Processing, 2007, pp. 33–37.
9. Y. Lipman, O. Sorkine, D.C. Or, D. Levin, C. Rössl, H-P. Seidel, Differential coordinates for interactive mesh editing, in Proceedings of International Shape Modeling **2004**, 181–190 (2004)
10. O. Sorkine, Differential representations for mesh processing. Comput. Graph. Forum 25, 789–807 (2006).
11. Y. Lipman, O. Sorkine, M. Alexa, D.C. Or, D. Levin, C. Rössl, H-P. Seidel, Laplacian framework for interactive mesh editing. Int. J. Shape Model. textbf11(1), 43–61 (2005).
12. A. Nealen, T. Igarashi, O. Sorkine, M. Alexa, *FiberMesh: designing freeform surfaces with 3D curves*, in *Proceedings of ACM SIGGRAPH* (ACM Trans. Graph, 2007)
13. R. Gal, O. Sorkine, N.J. Mitra, D.C. Or, iwires: an analyze-and-edit approach to shape manipulation. ACM Trans. Graph. Siggraph **28**(3), 1–10 (2009)
14. O. Sorkine and M. Alexa. As-rigid-as-possible surface modeling, in Symposium on Geometry Processing, 2007, pp. 109–116.
15. O.K-C. Au, C-L. Tai, L. Liu, H. Fu, Dual laplacian editing for meshes. IEEE Trans. Visual Comput. Graphics textbf12(3), 386–395 (2006).
16. Q. Luo, B. Liu, Z. Ma, H. Zhang, Mesh editing in roi with dual laplacian, in CGIV '07: Proceedings of the Computer Graphics, Imaging and Visualisation, 2007, pp. 195–199.
17. N. Thalmann, R. Laperrière, D. Thalmann, Joint-dependent local deformations for hand animation and object grasping, in Proceedings on Graphics, Interface '88, 1988, pp. 26–33.
18. I. Baran, J. Popović, Automatic rigging and animation of 3d characters, in ACM SIGGRAPH 2007 papers, 2007.
19. C. Miller, O. Arikan, D. Fussell, Frankenrigs: building character rigs from multiple sources, in Proceedings of I3D'10, 2010, pp. 31–38.
20. O. Weber, O. Sorkine, Y. Lipman, C. Gotsman, Context-aware skeletal shape deformation, in Proceedings of Eurographics, Compute. Graph. Forum (2007).
21. A. Jacobson, O. Sorkine, Stretchable and twistable bones for skeletal shape deformation, in Proceedings of ACM SIGGRAPH ASIA, ACM Trans. Graph. 30(6), (2011).
22. L. Kavan, S. Collins, Jiří Žára, C. O'Sullivan, Geometric skinning with approximate dual quaternion blending. ACM Trans. Graph. **27**(4), 105 (2008)
23. A.M. Adams, Y. Zhu, A. Selle, M. Empey, R. Tamstorf, J. Teran, E. Sifakis, Efficient elasticity for character skinning with contact and collisions. ACM Trans. Graph. 30(4), (2011).
24. T.W. Sederberg, S.R. Parry, Free-form deformation of solid geometric models, in SIGGRAPH '86, 1986, pp. 151–160.
25. S. Coquillart, P. Jancéne, *Animated free-form deformation: an interactive animation technique*, in *SIGGRAPH, 1991* (Comput. Graph, 1991)
26. R. MacCracken K.I. Joy, Free-form deformations with lattices of arbitrary topology, in SIGGRAPH '96, 1996, pp. 181–188.
27. K.G. Kobayashi, K. Ootsubo, t-ffd: free-form deformation by using triangular mesh, in Proceedings of the eighth ACM Symposium on Solid Modeling and Applications, 2003, pp. 226–234.
28. R.W. Sumner, M. Zwicker, C. Gotsman, J. Popović, Mesh-based inverse kinematics. ACM Trans. Graph. **24**(3), 488–495 (2005)
29. J. Huang, X. Shi, X. Liu, K. Zhou, L-Y. Wei, S-H. Teng, H. Bao, B. Guo, H-Y. Shum, Subspace gradient domain mesh deformation. ACM Trans. Graph. **25**(3), 1126–1134 (2006)
30. K. Zhou, J. Huang, J. Snyder, X. Liu, H. Bao, B. Guo, H-Y. Shum, Large mesh deformation using the volumetric graph laplacian. ACM Trans. Graph. **24**, 496–503 (2005)
31. K.G. Der, R.W. Sumner, J. Popović, Inverse kinematics for reduced deformable models, in SIGGRAPH '06: ACM SIGGRAPH. Papers **2006**, 1174–1179 (2006)
32. T. Ju, S. Schaefer, J. Warren, Mean value coordinates for closed triangular meshes. ACM Trans. Graph. **24**(3), 561–566 (2005)
33. C. Xian, H. Lin, S. Gao, Automatic generation of coarse bounding cages from dense meshes, in Shape Modeling, International, 2009, 2009.

34. M. B-Chen, O. Weber, C. Gotsman, *Variational harmonic maps for space deformation* (ACM Trans. Graph, 2009)
35. Z.-J. Deng, X.-N. Luo, X-P. Miao. Automatic cage building with quadric error metrics. *J. Comput. Sci. Technol.* **26**(3), 538–547 (2011)
36. C. Xian, H. Lin, S. Gao, Automatic cage generation by improved obbs for mesh deformation. *Vis. Comput.* **28**(1), 21–33 (2012)
37. T. Ju, Q.-Y. Zhou, M. Panne, D.C. Or, U. Neumann, Reusable skinning templates using cage-based deformations. *ACM Trans. Graph.* **27**(4), 1–23 (2008)
38. J. Kosinka, M. Barton, Injective shape deformations using cube-like cages. *Comput. Aided. Des. Appl.* **7**(3), 309–318 (2010)
39. Y. Zhao, X-G. Liu, Q-S. Peng, H-J. Bao, Rigidity constraints for large mesh deformation. *J. Comput. Sci. Technol.* **24**, (2009).
40. M. Meyer, H. Lee, A. Barr, M. Desbrun, Generalized barycentric coordinates on irregular polygons. *J. Graph. Tools.* **7**(1), 13–22 (2002)
41. P. Joshi, M. Meyer, T.D. Rose, B. Green, T. Sanocki, Harmonic coordinates for character articulation. *ACM Trans. Graph.* **26**(3) (2007).
42. Y. Lipman, D. Levin, D.C. Or, Green coordinates. *ACM Trans. Graph.* **27**(3) (2008).
43. Y. Lipman, J. Kopf, D. C. Or, D. Levin, Gpu-assisted positive mean value coordinates for mesh deformations, in *SGP*, 2007, pp. 117–123.
44. X.-Y. Li, S.-M. Hu, Poisson coordinates. *IEEE Trans. Visual Comput. Graphics* **19**(2), 344–352 (2012)
45. T. Langer, H.-P. Seidel, Higher order barycentric coordinates. *Comput. Graph. Forum* **27**(2), 459–466 (2008)
46. T.D. Rose, M. Meyer, *Harmonic coordinates* (Pixar Technical Memo, Pixar Animation Studios, 2006)
47. M.R. Rustamov, Boundary element formulation of harmonic coordinates, Technical report, (2008).
48. C-A. Lin, E. Miller, G.S. Lee, ibind: smooth indirect binding using segmented thin-layers, in *SIGGRAPH Talks*, 2009.
49. A. Jacobson, I. Baran, J. Popović, O. Sorkine, Bounded biharmonic weights for real-time deformation. *ACM Trans. Graph.* **30** (2011).
50. O. Weber, C. Gotsman, Controllable conformal maps for shape deformation and interpolation, in *ACM SIGGRAPH 2010 papers*, SIGGRAPH '10 vol 29 (2010).
51. O. Weber, R. Poranne, C. Gotsman, Biharmonic coordinates. *Comput. Graph. Forum.* **31**(8), 2409–2422 (2012)
52. E. Landreneau, S. Schaefer, Poisson-based weight reduction of animated meshes. *Comput. Graph. Forum.* **29**, 1945–1954 (2010)
53. A. Jacobson, T. Weinkauff, O. Sorkine, Smooth shape-aware functions with controlled extrema. *Proceedings of Eurographics/ACM SIGGRAPH Symposium on Geometry Processing.* *Comput. Graph. Forum.* **31**(5), 1577–1586 (2012)
54. Z. Li, D. Levin, Z. Deng, D. Liu, X. Luo, Cage-free local deformations using green coordinates. *Vis. Comput.* **26**, 1027–1036 (2010)
55. M.B. Chen, O. Weber, C. Gotsman. Spatial deformation transfer, in *Proceedings of the SCA'09*, 2009 pp. 67–74.
56. L. Chen, J. Huang, H. Sun, H. Bao, Cage-based deformation transfer. *Comput. Graph.* **34**, 107–118 (2010)
57. D.C. Or, Space deformations, surface deformations and the opportunities in-between. *J. Comput. Sci. Technol.* **24**(1), 2–5 (2009)
58. S. Zhang, P. Borosan, R. Howard, A. Nealen. Hybrid mesh editing, in *Proceedings of Eurographics 2010 (short papers)*, Sweden, (2010).
59. M. Desbrun, M. Meyer, P. Schröder, A.H. Barr. Implicit fairing of irregular meshes using diffusion and curvature flow, in *SIGGRAPH*, 1999 pp. 317–324.
60. O. Sorkine, D.C. Or, Least-squares meshes, in *Shape Modeling. International* **2004**, 191–199 (2004)

61. M. Botsch, D. Bommes, L. Kobbelt, Efficient linear system solvers for mesh processing, in IMA Conference on the Mathematics of Surfaces, 2005 pp. 62–83.
62. J. Kwon, I.-K. Lee, Rubber-like exaggeration for character animation, in Proceedings of the 15th Pacific Conference on Computer Graphics and Applications, 2007 pp. 18–26.
63. J.W. Davis, V.S. Kannappan, Expressive features for movement exaggeration, in SIGGRAPH '02, 2002 pp. 182–182.
64. C. Bregler, L. Loeb, E. Chuang, H. Deshpande, Turning to the masters: motion capturing cartoons, in SIGGRAPH '02, 2002 pp 399–407.

Cage-based Performance Capture

Savoye, Y.

2014, X, 141 p. 86 illus., 85 illus. in color., Hardcover

ISBN: 978-3-319-01537-8