Online Research @ Cardiff

This is an Open Access document downloaded from ORCA, Cardiff University's institutional repository: https://orca.cardiff.ac.uk/id/eprint/142001/

This is the author's version of a work that was submitted to / accepted for publication.

Citation for final published version:

Yuan, Yu-Jie, Lai, Yu-Kun ORCID: https://orcid.org/0000-0002-2094-5680, Wu, Tong, Gao, Lin and Liu, Ligang 2021. A revisit of shape editing techniques: from the geometric to the neural viewpoint. Journal of Computer Science and Technology 36 (3) , pp. 520-554. 10.1007/s11390-021-1414-9 file

Publishers page: http://dx.doi.org/10.1007/s11390-021-1414-9 <http://dx.doi.org/10.1007/s11390-021-1414-9>

Please note:
Changes made as a result of publishing processes such as copy-editing, formatting and page numbers may not be reflected in this version. For the definitive version of this publication, please refer to the published source. You are advised to consult the publisher's version if you wish to cite this paper.

This version is being made available in accordance with publisher policies. See http://orca.cf.ac.uk/policies.html for usage policies. Copyright and moral rights for publications made available in ORCA are retained by the copyright holders.

information services
gwasanaethau gwybodaeth

A Revisit of Shape Editing Techniques: From the Geometric to the Neural Viewpoint

Yu-Jie Yuan[1,2], Yu-Kun Lai[3], Tong Wu[1,2], Lin Gao[1,2,*], and Ligang Liu[4]

[1] *Beijing Key Laboratory of Mobile Computing and Pervasive Device, Institute of Computing Technology, Chinese Academy of Sciences, Beijing 100190, China*

[2] *University of Chinese Academy of Sciences, Beijing 100049, China*

[3] *School of Computer Science and Informatics, Cardiff University, Cardiff CF24 3AA, U.K.*

[4] *University of Science and Technology of China, Hefei 230026, China*

E-mail: yuanyujie@ict.ac.cn; LaiY4@cardiff.ac.uk; wutong19s@ict.ac.cn; gaolin@ict.ac.cn; lgliu@ustc.edu.cn

Abstract 3D shape editing is widely used in a range of applications such as movie production, computer games and computer aided design. It is also a popular research topic in computer graphics and computer vision. In past decades, researchers have developed a series of editing methods to make the editing process faster, more robust, and more reliable. Traditionally, the deformed shape is determined by the optimal transformation and weights for an energy formulation. With increasing availability of 3D shapes on the Internet, data-driven methods were proposed to improve the editing results. More recently as the deep neural networks became popular, many deep learning based editing methods have been developed in this field, which are naturally data-driven. We mainly survey recent research studies from the geometric viewpoint to those emerging neural deformation techniques and categorize them into organic shape editing methods and man-made model editing methods. Both traditional methods and recent neural network based methods are reviewed.

Keywords mesh deformation, man-made model editing, deformation representation, optimization, deep learning

1 Introduction

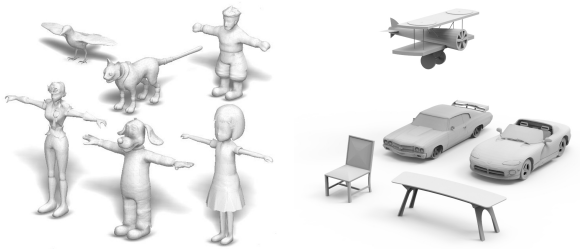
3D shapes are one of the most important types of objects in computer graphics and computer vision research. Editing or interactive deformation of 3D shapes provides an intuitive way to produce new shapes based on existing ones, which is fundamental for many applications. Methods for 3D shape editing are therefore one of the research hot spots. In recent years, deep learning has been widely used, and many research fields have developed new solutions based on deep learning, such as deep generation of 3D models [1, 2], 3D deep reconstruction [3, 4], deep neural network based 3D shape analysis methods [5, 6], 3D shape retrieval [7] and so on. 3D models can be generally divided into two types, namely organic shapes and man-made models. Fig. 1 shows some examples of these two types. Organic shapes such as human bodies, animals, and so on. are often deformable, whereas man-made objects tend to comprise of a larger number of (near-)rigid components. Different techniques are therefore needed to cope with these two types of shapes. Neural network based editing methods based on deep learning are also emerging, although they are still at relatively early stage and many open areas remain, which we will discuss later in the survey.

Survey

by National Natural Science Foundation of China under Grant Nos. 62061136007 and 61872440, Royal Society Newton Advanced Fellowship under Grant No. NAF\R2\192151, Youth Innovation Promotion Association CAS, Science and Technology Service Network initiative, Chinese Academy of Sciences under Grant No. KFJ-STS-QYZD-2021-11-001.

*Corresponding Author

Postprinted submitted to JCST



(a) (b)

Fig.1. Some examples of organic shapes (in [8]) and man-made models (in [9]). (a) Organic Shapes. (b) Man-made Models.

Early 3D model editing methods analyze the characteristics of the model itself, and strive to keep these characteristics unchanged during the deformation. For organic shapes, common examples include human bodies and animal shapes, which are near articulated as shown in Fig. 1(a). It is possible to bind a skeleton inside the model. On the one hand, the editing of these models typically defines deformation energies to impose constraints on the deformation, such as volume-preserving deformation. On the other hand, by binding skeletons for these models, the user can manipulate the skeleton to drive the deformation of the shape. Skeleton-based deformation is often convenient and leads to good results. However, the binding of the skeleton is not only time-consuming, but also requires professional software and expertise. For man-made models, the main purpose of editing is to modify the appearance, or geometric features of the models. For this purpose the topological structure of the model is usually a feature that needs to be maintained. Such kind of methods is referred to as structure-aware editing [10]. The editing of man-made models is more complicated than the deformation of organic shapes, because organic shapes are typically manifold meshes, while man-made models are often non-manifold with more complex structures.

Surveys on other aspects of 3D models have recently been published, such as 3D deep generative models [11], 3D deep reconstruction [12, 13] and 3D deep representation [14]. However, for 3D shape editing/deformation, existing surveys [15, 16] were published over a decade ago, only covering deformation methods of 3D organic shapes. Methods for editing of man-made models are not reviewed in specialized surveys, and only discussed in loosely related courses [10, 17]. The rapid development of deep learning in recent years has also led to the emergence and the growth of neural network based deformation and editing methods. It is necessary to have an extensive review to summarize the related research and discuss future directions. To this end, we present this survey, reviewing both traditional methods and methods based on deep neural networks, as well as methods applied to both organic shapes and man-made models.

This survey is structured as follows. We divide the editing methods according to different analysis perspectives, namely attribute-based (Section 2) or traditional data-driven based (Section 4) methods. Although neural editing methods also learn from datasets, because this kind of methods is a new direction that is currently being actively explored and often requires and benefits from a larger amount of data, we will introduce it in Section 5, separately from the traditional data-driven methods. For these three types, because organic shapes and man-made models have certain differences in representation, and their editing methods also have differences, we summarize methods for organic shapes and man-made models separately. Skeleton-based and cage-based deformation methods rely on handles, often referred to as proxies different from the model itself and usually require weighted interpolation of deformation on the skeleton or cage to obtain the transformation of the shape. They will be discussed in Section 3, and these methods can also exploit the information from datasets. Finally, we will conclude with existing prob-

lems and discuss interesting future research directions (Section 6). Fig. 2 provides a timeline of representative shape editing methods for organic shapes and man-made models.

2 Attribute-Based Model Editing

In this section, we discuss those methods that focus on analyzing various attributes of the surface model, including geometry characteristics and semantic attributes, to define different constraints which are then used to guide the model editing.

Organic shapes, a.k.a. deformable models, usually refer to models that are non-rigidly deformable. Human bodies and animal shapes are common examples. The editing of 3D organic shapes mostly uses interactive deformation. In these methods, organic shapes are often represented as triangular meshes. The deformation of organic shapes mainly strives to maintain the geometric details of the original shape and produce natural and reasonable results. The early deformation methods mainly analyze the geometry of the shape and define the constraints accordingly for the deformation. We summarize those methods as geometry-based mesh deformation methods.

The editing of man-made models will be relatively more complicated and difficult, compared to the deformation of organic shapes. On the one hand, the man-made models have different shapes and complex topological structures. On the other hand, the meshes of man-made models are generally not regular and consistent. This has led to certain obstacles to the direct application of some deformation methods of organic shapes. To achieve the purpose of editing, one should pose some constraints on 3D man-made models to ensure plausible results. One way to obtain those constraints is maintaining the structural relations between different components of the model, which can be seen as a kind of semantic knowledge. We summarize these semantic constraints for man-made models in Subsection 2.2.

2.1 Geometry-Based Mesh Deformation

Geometry-based deformation methods typically define energy functions which transform the deformation problem into a constrained optimization problem. The constraints are generally provided by the user by specifying control handles and their positions. Early research studies were all around simulating the elastic deformation of objects. Terzopoulos *et al.* [18] proposed the classical elastic energy or the so-called shell energy which measures stretching and bending by the change of the first and the second fundamental forms and optimizes the energy to obtain deformation results. Two follow-up studies [19, 20] propose to simplify the energy by replacing the first and the second fundamental forms by the first and the second order partial derivatives of displacement function. In order to solve the problems of computational complexity and distortion of geometric details, many studies [21, 22, 23, 24, 25, 26] based on multigrid solvers or multi-resolution deformation strategies have been proposed in succession. Integrated into a multi-resolution framework [24], Bostch and Kobbelt [27] proposed an intuitive framework for the user to define custom tailored basis functions, which are precomputed to achieve real-time editing. For the above works, please refer to [15] for a thorough introduction. Follow-up works change the form of energy formulation to facilitate the solution and achieve better results. The most widely used is Laplacian-based mesh editing. The well-known As-Rigid-As-Possible (ARAP) energy is also Laplacian-based, and has been applied to deformation with extensive research including recent development [28]. We will begin with Laplacian based methods, including ARAP and follow-ups, followed by

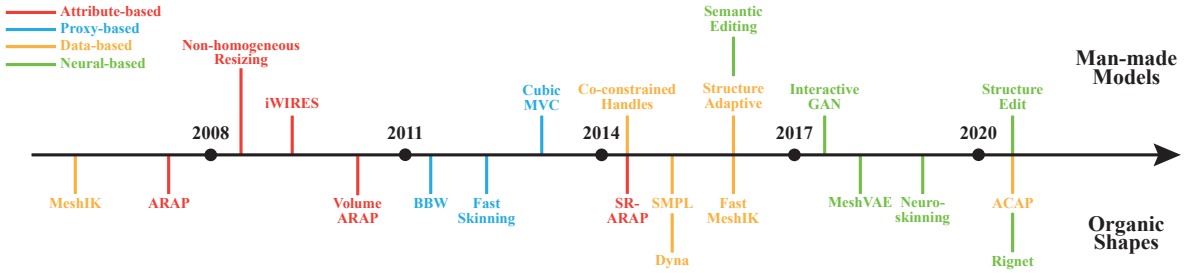


Fig.2. Timeline of representative 3D shape editing methods for two types of 3D models.

methods using other formulations.

2.1.1 *Laplacian-Based Mesh Deformation*

Kobbelt *et al.* [24] were the first to propose a multi-resolution Laplacian-based deformation method, which is able to approximately solve constrained mesh optimization in real time. The readers may refer to [29, 30] which give an early summary of Laplacian-based mesh processing. Differential coordinates can capture local properties [31] used in free-form deformation [32], and allow a direct detail-preserving reconstruction of the edited mesh by solving a linear least-squares system [33]. However, the differential coordinates are defined in a global coordinate system, and thus are not rotation-invariant, thereby it is necessary to introduce approximated local frames to compensate for distortions caused by orientation [33]. Laplacian coordinates, as pointed out by [33], are the simplest form of the differential coordinates. Given a triangular mesh model, we denote each vertex of the mesh as $\boldsymbol{v}_i, i = 1, \cdots, n$, where n is the number of vertices. The 1-ring neighborhood of $\boldsymbol{v}_i$ is denoted as $N(i)$. Then we can define Laplacian coordinates of the vertex $\boldsymbol{v}_i$ as Eq. 1

$$\boldsymbol{l}_i = \sum_{j \in N(i)} w_{ij}(\boldsymbol{v}_j - \boldsymbol{v}_i), \tag{1}$$

where w_{ij} is the weight of the edge $\boldsymbol{e}_{ij} = \boldsymbol{v}_j - \boldsymbol{v}_i$. It can be seen that $\boldsymbol{l}_i$ is a weighted average of position differences between the vertex $\boldsymbol{v}_i$ and its adjacent vertices, thereby it describes the local geometry at $\boldsymbol{v}_i$. By collecting all Laplacian coordinates $\boldsymbol{l}_i$ and presenting them in the matrix form, it can be written as $\boldsymbol{l} = \boldsymbol{L}\boldsymbol{V}$, where $\boldsymbol{L}$ is a $3n \times 3n$ matrix with elements composed of weights w_{ij}. $\boldsymbol{L}$ is referred to the Laplacian operator and its elements as the Laplacian coefficients. Sorkine *et al.* [34] proposed to minimize the differences between Laplacian coordinates before and after deformation to deform the surface models, which can form a sparse linear system and be solved in the least squares sense. Lipman *et al.* [35] reviewed the above two Laplacian based methods [33, 34] which both preserve shape details when editing mesh models.

Many works have improved Laplacian coordinates or proposed other forms of differential coordinates. For example, Yu *et al.* [36] proposed a gradient domain mesh editing method which deforms meshes by interpolating gradient fields derived from user constraints. Combined with [36], Zayer *et al.* [37] also proposed a gradient-based shape editing method, which propagates local deformations over the surface via harmonic scalar fields. Moreover, by specifying several corresponding markers on two models, the manipulations on one model can be transferred to the other model. However, the proposed method is translation-insensitive [38], which may cause unsatisfactory results when the constraints contain translation. Zhou *et al.* [39] proposed a Laplacian coordinate method based on a volumetric graph to better preserve the model volume during deformation. The above two methods re-

quire the orientation and local frames of the handles as input. Therefore if the users only move the handles, the orientation and the local frames are not changed accordingly, leading to shearing and stretching distortions caused by incompatible handle positions and orientations. Pyramid coordinates [40] and iterative dual Laplacian [41] are proposed to solve this problem , preserving rotation invariance and can avoid the distortion caused by incompatible rotation and translation of handles. However, such methods cannot handle large-scale rotations. To deal with the problem that the previous methods either cannot get the rotation information when the handles are only translated, or cannot handle large-scale rotations, Fu *et al.* [42] proposed to use an affine matrix at each vertex which is linear w.r.t. the vertex position. They further decomposed the affine matrix by polar decomposition to extract only rotation and uniform scaling to offset the impact of shearing distortion.

Most of the previous gradient based editing methods turn the problem into solving a linear system, but not all the constraints can be formulated linearly and also, non-linear constraints are more flexible. Huang *et al.* [43] proposed a solution framework that can effectively solve the deformation containing nonlinear constraints, such as skeleton constraints for skeleton-based deformation and volume constraints for volume preservation. Those constraints can be transformed to a nonlinear energy minimization problem, but the minimization faces the problems of slow convergence and numerical instability. So Huang *et al.* built a coarse cage mesh enclosing the original mesh shape, and used mean value coordinates [44] to transfer the energy and constraints to the cage.

Vallet *et al.* [45] proposed an efficient method to compute eigenfunctions of the Laplacian even for meshes with a million vertices. Based on that, they also proposed a filtering algorithm to filter specific frequency bands, which can be used to interactively edit mesh surfaces, such as exaggerating geometric details.

As-rigid-as-possible (ARAP) deformation is an important part of Laplacian-based methods. The principle of as-rigid-as-possible (ARAP) was first applied to shape interpolation [46] and the deformation of two-dimensional shapes [47]. Sorkine *et al.* [48] further proposed a 3D surface model deformation method that maintains local rigidity. The method is based on minimizing an ARAP energy, which measures non-rigid distortions in local 1-ring neighborhoods of all vertices.

We denote the triangle mesh as S, and $N(i)$ is the index set of vertices adjacent to vertex i. We denote $\boldsymbol{v}_i \in \mathbb{R}^3$ as the position of the vertex i on the mesh S. Also assume that S is to be deformed to S' with the same connectivity and different vertex positions $\boldsymbol{v}_i'$. As shown in Eq. 2, the overall deformation energy to measure the rigidity of the entire mesh is the sum of the distortion energies of each deformation cell C_i (including vertex i and its 1-ring neighbors)

$$\begin{aligned} E(S') &= \sum_{i=1}^{n} \bar{w}_i E(C_i, C_i') \\ &= \sum_{i=1}^{n} \bar{w}_i \sum_{j \in N(i)} w_{ij} \|(\boldsymbol{v}_i' - \boldsymbol{v}_i') - \boldsymbol{R}_i(\boldsymbol{v}_i - \boldsymbol{v}_j)\|^2. \end{aligned} \tag{2}$$

Here, C_i' denotes the deformed cell of C_i, $w_{ij} = \frac{1}{2}(\cot \alpha_{ij} + \cot \beta_{ij})$ is the cotangent weight, and α_{ij}, β_{ij} are the angles opposite of the mesh edge (i, j), $\bar{w}_i$ is the cell weight that needs to be pre-determined, which is set to 1 in [48]. We can notice that $E(S')$ depends only on the geometry of S and S', the positions of the vertices $\boldsymbol{v}_i$ and $\boldsymbol{v}_i'$. In particular, when the source model S is determined, the only variables in $E(S')$ are the deformed vertex coordinates $\boldsymbol{v}_i^{'}$. This is because the optimal rotation matrix $\boldsymbol{R}_i$ is a function of $\boldsymbol{v}_i^{'}$.

[48] takes vertex i and its 1-ring neighbors as a cell,

and each cell seeks the best rotation matrix $\boldsymbol{R}_i$ that satisfies the condition as much as possible. The overlaps between the cells ensure continuous deformation. They further formulate an iterative optimization framework that is easy to implement, which readers can refer to [48] for details. The deformation shows the advantages on detail preservation and elastic effect. Given only position constraints, reasonable deformation results can be obtained, as shown in Fig. 3.

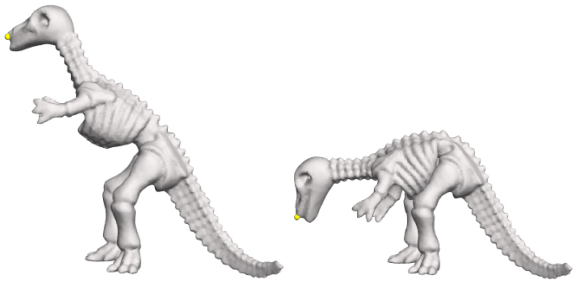


Fig.3. Moving a single position constraint, the result with large deformations can be obtained [48]. (a) The original shape. (b) The deformed shape.

Following [46, 47, 48], many ARAP extensions have been developed. Applied to volume deformation, Chao *et al.* [49] derived another discretization of ARAP energy from the continuous form. The ARAP energy can be further enhanced by smooth rotations [50], which can achieve comparable results on surface mesh models to volumetric ARAP [49] on tetrahedral meshes, as shown in Fig. 4. Cuno *et al.* [51] formulated the ARAP deformation with a Moving Least Squares (MLS) approach. Liu *et al.* [52] extended [46] and proposed a new morphing method for surface triangular meshes based on ARAP. Compared with [46], their method does not need tetrahedral meshes to represent the shapes, which reduces computation time, and by integrating the translation vector into the energy formulation, eliminates the need for users to specify the fixed vertices when solving the equation.

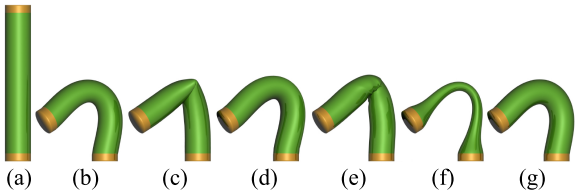


Fig.4. Comparison of SR-ARAP (smooth rotation ARAP) with some other deformation methods [50]. (a) Source shape. (b) PriMo [38]. (c) ARAP surface [48]. (d) ARAP volume [49]. (e) ARAP volume applied to a tetrahedral stratum. (f) ARAP surface with an additional term for a smooth map differential. (g) SR-ARAP [50].

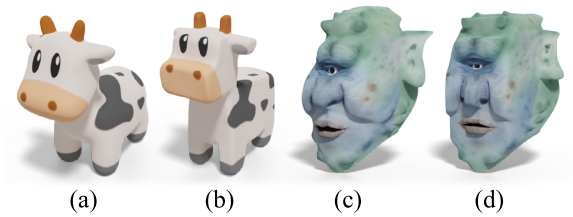


Fig.5. ARAP has been applied to deformation, with extensive research including some developments recently. Cubic stylization [28] minimizes the ARAP formulation with l_1 regularization to achieve locally isometric deformations while preserving texture attributes. (a) (c) are the source shapes. (b) (d) are the deformed shapes.

ARAP deformation has also been extended to make the stiffness of deformation controllable. Instead of using 1-ring neighborhoods, Chen *et al.* [53] specified larger neighborhood sizes to better preserve geometric details, and also offer a parameter to adjust physical stiffness. Qin *et al.* [54] replaced 1-ring neighborhoods with face-based local cells and specified a stiffness parameter for each local cell to simulate deformation of different materials. Le *et al.* [55] extended ARAP deformation to editing man-made models. They improved stiffness of ARAP deformation by introducing an anisotropic material and a membrane model. However, the deformation focuses on local stretching. Another approach to extending ARAP to anisotropic ARAP was proposed by Colaianni *et al.* [56, 57] , introducing an affine matrix $\boldsymbol{T}_i$ into the ARAP formulation

to enable directional editing:

$$E(S') = \sum_{i=1}^{n} \bar{w}_i \sum_{j \in N(i)} w_{ij} \| (\boldsymbol{v}_i' - \boldsymbol{v}_i') - \boldsymbol{T}_i \boldsymbol{R}_i (\boldsymbol{v}_i - \boldsymbol{v}_j) \|^2.$$

Different forms of matrix $\boldsymbol{T}_i$ can realize anisotropic scaling, anisotropic shearing or anisotropic rotation.

ARAP deformation methods are useful. However, they can only achieve interactive rates on coarse meshes [50]. Some research works [58, 59, 60] investigate acceleration techniques of ARAP. Borosan *et al.* [58] combined surface-based deformation with cage-based deformation to perform hybrid mesh editing. The user deforms the simplified version of the input shape using ARAP surface modeling [48], and the deformation is then propagated to the original shape by precomputed Mean Value Coordinates [44]. Manson *et al.* [61] also performed ARAP on simplified meshes and proposed a prototype of hierarchical ARAP. They built coarse meshes using edge contraction, and reversed the edge-collapse process to add details back after deformation on the simplified mesh. Following this acceleration strategy, Liu *et al.* [28] achieved cubic stylization of models by minimizing the ARAP energy with an l_1 regularization, as shown in Fig. 5. Sun *et al.* [59] also achieved hierarchical ARAP by constructing a biharmonic surface to decompose the mesh. Zollhofer *et al.* [60] proposed a GPU-based multi-resolution ARAP implementation, which accelerates the computation of ARAP and allows to pose even high-quality meshes consisting of millions of triangles in real time. Accelerating the optimization of ARAP has also been addressed in various recent works [62, 63, 64, 65, 66].

ARAP formulation has also been combined with other deformation methods. Zhang *et al.* [67, 68] integrated a skeleton into ARAP surface modeling, effectively extending it to volume modeling. They evenly sampled points on the skeleton and connected surface vertices with the sampled points to form skeleton edges, which are also considered in an ARAP energy together with the surface edges. In this way, the method can avoid volume loss, which is a common issue for surface-based deformation. Jacobson *et al.* [69] introduced the ARAP energy into the LBS (Linear Blend Skinning) deformation method to reduce the degrees of freedom that require user specification. They further clustered the vertices based on their Euclidean distances in the skinning weight space, and used the same rotation matrix for all the vertices in the same cluster. Yang *et al.* [70] proposed to combine the ARAP energy with a data-driven energy in deformation transfer. Their method also clusters vertices. However, their clustering is based on the rotation-augmented weight matrix, which is composed of the weight matrix and the ACAP (as-consistent-as-possible) deformation feature [71] (see Subsection 4.1 for more details). The resulting clusters are more reasonable than using the weight matrix alone. In addition to the above combinations, the ARAP energy has also been extended for use in other applications such as parametrization [72], data-driven interpolation [73], shape optimization [74], shape decomposition [75], mass-spring simulation [76], image registration [77, 78], image warping [79], and video stabilization [80].

2.1.2 Other Surface Geometry Properties

In addition to the Laplacian-based methods for analyzing the local characteristics of the mesh, there are many other geometry-based deformation methods that analyze surface mesh characteristics. For example, curvature is an important attribute of the surface. Crane *et al.* [81] edited shapes by manipulating the mean curvature and boundary data. The deformation is conformal and has less distortion. Fang *et al.* [82] utilized not only the mean curvature but also Gaussian curvature to perform editing. An example is shown in Fig. 6.

The method also performs conformal surface deformation which preserves local texture features.

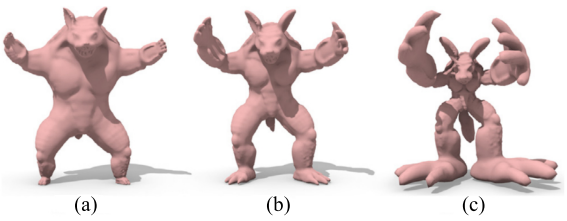


Fig.6. Shape editing by keeping the mean curvature while changing the Gaussian curvature [82]. As the control parameter λ increases, the details are preserved and the main structure is exaggerated. (a) $\lambda = 0.8$. (b) Input shape. (c) $\lambda = 1.2$.

Lai *et al.* [83] proposed a robust feature region extraction and classification method, which can identify features such as ridges, valleys and prongs on mesh surfaces. These feature regions are further used for feature editing of meshes, such as dilation and erosion of prong features and sharpening ridges. Lipman *et al.* [84] performed isometric deformations which preserve the first fundamental form of the surface and minimize the deviation of the second fundamental form. The proposed method preserves shape and volume even under a large rotation (2π radians).

Stretching a mesh may destroy its geometric details. Alhashim *et al.* [85] proposed a shape editing method for stretching, which replicates the geometric details along the stretching direction such that the geometric details are not distorted. They first used a base mesh to represent the general shape of the input, and then used the curve skeleton extracted by [86] to create a curvilinear grid on the desired stretching region, and project the region onto the grid to form a 2D texture. The user specifies the stretching direction by drawing a 3D curve, and the new geometric details will be synthesized according to the 2D texture.

Liu *et al.* [87] presented a set of scale-invariant measures consisting of triangle angles and edge angles (dihedral angles of the edges) to represent 3D shapes. The representation is unique for a given shape. Moreover, given one edge and the orientation of one of the triangles containing this edge, the mesh shape can be reconstructed by this representation uniquely. The reconstruction is through an iterative process that alternately solves the face normals and vertex coordinates. An ARAP-like formulation is introduced when updating the normals, and when solving for the vertex coordinates, the constraints obtained from the user's edited handles are added. The editing process preserves the local details at different scales.

Sparsity has also been widely used in geometry-based mesh deformation. Xu *et al.* [88] reviewed these methods in geometric modeling and processing that use sparsity, with one section discussing shape deformation based on sparsity. Gao *et al.* [89] introduced general l_p norms to shape deformation, and showed that different p values influence the distribution of unavoidable distortions. Deng *et al.* [90] explored local modifications of the shape, and proposed to use a mixed l_2/l_1 norm regularization which provides more local editing. Different from [89] that applies sparsity penalty on the error function, Deng *et al.* imposed it on the displacement vectors.

In addition to the explicit mesh representation, implicit representations such as distance fields or level sets also provide an efficient representation for some editing operations. Museth *et al.* [91] proposed a level set method for surface editing. They defined the speed function which describes the velocity at each surface vertex along the surface normal. Different speed functions develop different surface editing operators, such as the cut-and-paste operator for copying, removing and merging level set models, and smoothing operator for smoothing the enclosed surface to a predefined curvature value. The method enables easy blending and topological changes of models thanks to the flexibility of

implicit representation. Eyiyurekli and Breen [92] also operated on level set representations and aimed to prevent loss of surface details caused by movements in the direction of the surface normal. Inspired by the idea of multi-resolution deformation, they extracted geometric details in advance and stored them in the particles on the surface, and then combined the details back when the deformation is completed.

2.2 Semantic Constraints for Man-Made Models

Assuming that the input models are all meshes, the simplest way to edit a 3D model is to change the coordinates of the mesh vertices, but this way lacks the necessary constraints and is difficult to produce reasonable results. Therefore, we prefer to use high-level editing methods to edit multiple vertices at the same time, such as Free Form Deformation (FFD) [32, 93], which we will discuss with cage-based deformation in Subsection 3.2. Although this method is simple and straightforward, the users are required to adjust all parameters manually. Structure is only implicitly imposed by using only a few, low-frequency basis functions. It should be noted that structure is an important indicator in the editing of man-made models. [10] summarizes the method of structure-aware shape processing.

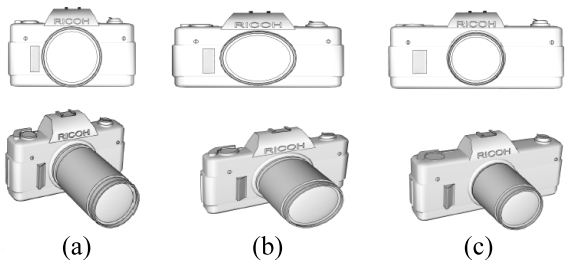


Fig.7. Non-homogeneous resizing results of stretching a camera which preserve structural features [94]. (a) Original. (b) Scaled. (c) Non-homogeneous.

2.2.1 *Local Adaptivity*

Early work on 3D man-made model editing efforts sought to maintain the reasonableness of the 3D shape when scaling 3D models. For example, Kraevoy *et al.* [94] proposed to estimate the "vulnerability" of local areas of the shape and adaptively deform the shape. This method prefers axis-aligned stretch when editing, as shown in Fig. 7. Xu *et al.* [95] proposed a joint-based shape deformation method, which uses joints to segment a 3D shape into parts, while constraining the relative spatial configuration of adjacent parts. The proposed deformation system edits the models under those joint constraints.

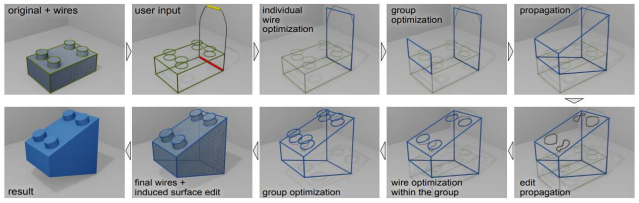


Fig.8. Pipeline of iWIRES [96].

2.2.2 *Global Relations*

It is not enough to only consider local adaptivity, so some methods explore the relationship between different parts or features of the whole model, and use this as a constraint to edit the model. Gal *et al.* proposed iWIRES [96], which forms an analyze-and-edit paradigm, as shown in Fig. 8. Based on the observation that man-made shapes can be abstracted by some special 1D line segments and their relationships, they abstracted the 3D shape into a set of curves and adopted simple methods to edit shapes while retaining geometry features. Utilizing 1D curves to represent the model structure, Li *et al.* [97] extracted a curve network and additional attributes as prior information to reconstruct the 3D model with detailed and interleaving structures from the scanned point cloud. Those extracted high-level curves can be used as han-

dles to edit the reconstructed model. Following a similar analyze-and-edit concept, Zheng *et al.* [98] decomposed the model into several meaningful parts, and abstracted the parts to simple geometric parametric primitives, named as component-wise controllers. During editing, the user manipulates one of the controllers and the applied change is automatically propagated to other controllers to maintain the structural relations among them, such as symmetry, coplanarity and parallelism. The final model is reconstructed with respect to the modified controllers. Those controllers also serve as deformation handles for image guided shape editing [99]. Zhang *et al.* [100] segmented a complex input mesh into several different primitives by clustering, which are depicted by a set of shape parameters and vertex coordinates. During the editing procedure, they added several different constraints on these parameters to minimize the target energy function. Optimized parameters are then applied to the corresponding primitives to change the shape of input mesh.

Architectural models such as buildings are also important editing targets. These models are highly structured and often have many repetitive patterns, such as windows. Based on this observation, Bokeloh *et al.* [101] first deformed the model under user constraints by as-rigid-as-possible deformation method [48] while maintaining continuous patterns. They found the repeated patterns in advance by sliding dockers and measured the stretch to determine insertion or deletion of those discrete repeated patterns after the elastic deformation. After finding those discrete or continuous regular patterns, the authors [102] further built a novel algebraic model of shape regularity and characterized the shape as a collection of those linked translational patterns. For those irregular architecture models, Lin *et al.* [103] proposed an editing method for resizing them. The users are required to specify the box hierarchy and corresponding attributes, such as replicated, scaled and fixed. Those irregular bounding boxes are then transformed into a set of disjoint sequences automatically. These sequences will be processed in turn. During processing, those user-specified operations are performed on corresponding boxes and their enclosed parts, while the remaining sequences are constrained. Milliez *et al.* [104] decomposed the model into different parts, each of which undergoes elastic deformation. They used several alternative rest states for each elastic part, so the deformation energy is computed by considering a set of those alternative rest shapes. The method further performs the corresponding model editing based on the jigsaw-puzzle-type local replacement mechanism on the user's interactive operations, such as replacement, stretching and shrinking, merging and cutting. Habbecke and Kobbelt [105] linearized the constraints that ensure regional and intuitive control in editing process, making real-time or interactive editing possible. Texture is an important attribute to show the appearance of the model, but it is not considered in the above methods. Cabral *et al.* [106] proposed an editing method for textured models which update the texture to maintain the texture features while editing the geometry of the model. They used directional autosimilarity, which measures the ability of a texture region to maintain similarity with itself under slight translation.

3 Proxy-Based Deformation

We focus on smoothly interpolating deformation along the surface of 3D models in this section, where proxies are used to drive the deformation of the models. An organic shape has a hinge structure. In addition to directly editing the vertices on the mesh, binding a skeleton to the shape, and driving the surface deformation through the skeleton are also a popular research direction. We summarize these as skeleton-based mesh

deformation methods. There is also extensive research of cage-based deformation methods that utilize a set of enclosing cages as proxies, which are not only suitable for organic shapes but also man-made models. We summarize these cage-based deformation methods in Subsection 3.2.

3.1 Skeleton-Based Mesh Deformation

Skeleton is one of the shape representations that can describe both the topology and the geometry of the shape [107]. There are various types of 3D skeletons, and we refer the readers to [107] for a thorough survey of the state-of-the-art of various 3D skeletons, while we mainly focus on bone-skeleton used for editing and deformation.

3.1.1 Skeleton-Based Skinning

Skeleton-based deformation is most commonly used for the deformation of realistic animated characters. It needs users to bind a skeleton to the shape first, which is termed as the bind time. The user then manipulates the skeleton to deform the shape accordingly, which is the pose time. Most methods propagate the handle transformations to the deformation of each surface vertex through a weighted blending of handle transformations. One of the classical methods that uses skeletons to drive the deformation of mesh surfaces is linear blend skinning (LBS), also known as skeleton subspace deformation (SSD) [108]. Let $\Omega \subset \mathbb{R}^2$ or $\mathbb{R}^3$ denote the volumetric domain enclosed by the given shape S. We denote the handles by $H_j \subset \Omega, j = 1, ..., n_h$. In fact, LBS is not limited to skeleton-based deformation. A handle can be a single point, a region, a skeleton bone or a vertex of a cage. Here, we focus on the skeleton bone, and others are easy to generalize. A transformation matrix $\boldsymbol{T}_j$ requires the user's specification for each handle H_j. Then all vertices $\boldsymbol{v} \in \Omega$ are deformed by their weighted blends:

$$\boldsymbol{v}_i^{'} = \sum_{j=1}^{n_h} \boldsymbol{W}_{ij} \boldsymbol{T}_j \boldsymbol{v}_i, \tag{3}$$

where $\boldsymbol{v}_i^{'}$ is the vertex coordinates after deformation, $\boldsymbol{v}_i$ is the vertex coordinates before deformation, and $\boldsymbol{W}_{ij}$ is the skinning weight of handle H_j on vertex i.

The linear blend weights $\boldsymbol{W}$ in Eq. 3 are crucial to the deformation. Usually, the LBS weights are determined by manual assignment or coming from dataset analysis, which not only takes lots of time and effort, but also produces unnatural deformation results due to the lack of smoothness. To address this, Bang *et al.* [109] proposed a spline interface for users to edit skinning weights interactively. Some early works use bone heat [110] or an improved version, bone glow [111], to assign the skinning weights. Jacobson *et al.* [112] proposed bounded biharmonic weights (BBWs), aiming at enabling users to work freely with the most convenient combination of handle types, and making deformation design and control easier. The BBWs produce smooth and intuitive deformation results for any topology of control points, skeletons, and cages. They define the weight vector $\boldsymbol{W}_j$ of the j-th handle (consisting of the control point weights at all vertices) as minimizers of a higher-order shape-aware smoothness functional, namely, the Laplacian energy:

$$\underset{\boldsymbol{W}_j, j=1,...,n_h}{\arg\min} \frac{1}{2} \int_{\Omega} \|\Delta \boldsymbol{W}_j\|^2 dV, \tag{4}$$

subject to: $\boldsymbol{W}_j|_{H_k} = \delta_{jk}$, $\sum_{j=1}^{n_h} \boldsymbol{W}_j(\boldsymbol{v}) = 1$ and $0 \leq \boldsymbol{W}_j(\boldsymbol{v}) \leq 1, j = 1, ..., n_h, \forall \boldsymbol{v} \in \Omega$,
where δ_{jk} is the Kronecker function, and $\boldsymbol{v}$ is the mesh vertices. It is natural that different control handles do not affect each other. The constraints also guarantee that the deformed shape will not scale and the handles all have positive contributions to the deformation.

For conveniently solving the Laplacian energy Eq. 4, [112] discretizes the equation using the standard linear

FEM Laplacian $\boldsymbol{M}^{-1}\boldsymbol{L}$, where $\boldsymbol{M}$ is the lumped mass matrix and $\boldsymbol{L}$ is the symmetric stiffness matrix. After discretizing the continuous integral, we can get Eq. 5

$$\sum_{j=1}^{n_h} \frac{1}{2} \int_{\Omega} ||\Delta \boldsymbol{W}_j||^2 dV \approx \frac{1}{2} \sum_{j=1}^{n_h} {\boldsymbol{W}_j}^{\mathrm{T}} (\boldsymbol{L}\boldsymbol{M}^{-1}\boldsymbol{L}) \boldsymbol{W}_j. \quad (5)$$

Through discretization, the minimization of an integral is converted into a quadratic optimization which is easy to compute. The above constraints are all linear equations or inequalities about $\boldsymbol{W}_j$. Once we know the matrices $\boldsymbol{M}$ and $\boldsymbol{L}$ of the given shape, the only thing left is solving a quadratic optimization problem under linear constraints. We can observe in Fig. 9 that the BBWs are smooth and local.

Directly adding constant bounds to high-order energy leads to more and more oscillation [113]. So Jacobson *et al.* [113] minimized quadratic energies while avoiding spurious local extrema to wrangle the oscillations. By exploiting datasets to strengthen the BBWs, Yuan *et al.* [114] used data-driven, ARAP and sparsity terms to optimize the BBWs. The deformation results using optimized weights can better reflect the deformation behavior of the example shapes in the dataset.

The above methods are suitable for manifold meshes. For non-manifold meshes, such as models obtained from 3D modeling software, they are often not watertight or have multiple components. One way of computing skinning weights is to voxelize the model [115, 116]. The weights are then calculated based on the geodesic distance between each voxel lying on a skeleton "bone" and all non-exterior voxels. [115, 116] also allow the user to modify weights interactively when deforming the model to test the effect of the modification.

The calculated weights always have an inapplicable area. Eliminating the trouble of assigning weights, Yan *et al.* [117, 118] proposed to use skeleton to drive the transformation of mesh simplices (triangles in 2D and tetrahedra in 3D) instead of vertices without the need of specification of skinning weights. The vertex connectivity information was directly exploited in their method since simplices include mesh connectivity information.

Although LBS is straightforward, easy to implement and has real-time performance, it can lead to well-known artifacts such as "collapsing elbow" and "candy wrapper". Some methods [119, 120, 121, 122] have been proposed to address these problems. Rumman and Fratarcangeli [123] first transformed the surface mesh to a tetrahedral mesh where LBS is performed, then added stretch constraint, tetrahedral volume constraint and bound constraint to eliminate the artifacts caused by LBS. The constraints are solved by a parallel Position-Based Dynamics schema. Performing contextual deformation, Weber *et al.* [124] separated surface detail information from skeleton driven pose changes and learned the deformation of skin details from the example characteristic shapes. The editing results can avoid the artifacts of LBS at body elbows. Shi *et al.* [125] also considered detailed motions (or secondary deformations formally) in skeleton-based deformation. They utilized LBS to generate primary deformations and learned those physical behaviors from the example sequences.

In addition to LBS, there are other alternative skinning methods, such as linear combinations of dual quaternions or dual quaternion skinning (DQS) [126, 127]. However, it suffers from more complex vertex processing [128]. So the work [128] makes improvements and only uses a few samples of nonlinear functions (virtual bones) in some key locations, such as joint areas. Other non-linear techniques, such as log-matrix skinning (LMS) [129, 130] and spherical blend skinning (SBS) [131] also perform volume-preserving deformation, but suffer from bulges near bent joints [132]. Kim and Han [133] proposed some post-processing opera-

Fig.9. The bounded biharmonic weights [112] are smooth and local.

tions such as modifying vertex coordinates and normals to solve the bulge and distortion problems faced by DQS.

Another choice is spline skeletons [134, 135, 136]. They viewed the bone as a spline and introduced spline deformation to skinning animation, replacing the previous transformation matrix guidance. These methods can produce better results but are nonlinear and often fail when large rotation deformations are encountered. The differential blending method proposed by Öztireli *et al.* [137] can solve this problem. The work uses sketch as the interaction tool, and the selected bones will deform to match the strokes drawn by the user.

Jocobson *et al.* [69] combined ARAP [48] with the original LBS formulation, different from some other methods [138, 139, 140] which change the LBS formulation to other forms. All computations of [69] except for SVD decomposition are linear. When the number of vertex clusters is reasonably selected, real-time deformation can be guaranteed. Also incorporating ARAP energy into LBS deformation, Thiery and Eisemann [141] proposed a method to generate skinning weights given a 3D mesh and corresponding skeleton. They used a variant of bone heat weights [110] to initialize the weights and optimize both weights and skeleton joints according to the deformation quality to example shapes. Li *et al.* [142] proposed an automatic implicit skinning method which bounds the surface onto the skeleton implicitly. The local surface surrounding the joint is used to parameterize the joint position. The deformation is achieved by Laplacian deformation energy with volumetric constraints which prevent those unnatural collapsing at the joints. Kavan and Sorkine [143] aimed to produce visually similar results to physical elastic simulations through a skeleton-based skinning method. So they proposed not only a new way to calculate the skinning weights but also a new skinning method based on a specific deformer, which they called joint-based deformers. Le *et al.* [132] proposed to impose orthogonal constraints to prevent those artifacts near the joints suffered by LBS, DQS, LMS, and SBS and guarantee real-time performance. However, they needed rest pose with skinning weights and bone transformations as inputs.

Artifacts at the joints are often caused by surface self-contact. Physically-based methods can solve the problem of skin collision well and produce visually plausible deformations, but even after significant optimization [144], they can only be close to real-time and cannot achieve complete real-time interactive posing. Vaillant *et al.* [145] segmented the mesh according to the skeleton bones by [110], and then approximated each part with an implicit surface utilizing Hermite Radial Basis Functions (HRBF) [146, 147], and at last merged different parts by union or other methods that perform better. They proposed to edit the shape through these field functions and geometric skinning methods. The rigid transformations are also applied to the field functions during deformation. The mesh vertices move along the gradient of field function and stop when they

reach the original field value or the point where the gradient is discontinuous so that the surface contacts can be handled well without collision detection. Based on [145], Vaillant *et al.* [148] further proposed a new family of gradient-based composition operators for combining those implicit surfaces which can deal with surface contacts better. They also derived a tangential relaxation scheme from ARAP [48] to track the iso-surface. The deformation results are better than these of [145], especially on extreme character movements. Teng *et al.* [149] applied the subspace simulation of articulated deformable shapes to deal with self-contact situation. They proposed a pose-space cubature scheme to resolve the collision without detecting all collision points.

Without the need to input the skeleton or predict the hierarchical structure of the bones, James and Twigg [150] used non-parametric mean shift clustering and least squares method to establish proxy bone transformations and vertex weights to edit and animate the shape. Yoshizawa *et al.* [151] proposed to extract a skeletal mesh from the dense mesh model. The skeletal mesh is deformed by FFD [32] and the deformation is back-propagated to the dense model using differential coordinates. A hierarchical framework is used to speed up the process. Xie *et al.* [152] proposed a shape editing method for personal fabrication applications where the user edits the shape through the constructed skeleton. They observed that most of the editing made by users are local. Based on this fact, they introduced a domain decomposition method that allows the FEM system to re-assemble the sub-matrices only for the local part modified by the user, while the rest remains unchanged, which can avoid unnecessary calculations for fast convergence. Following [152], Xu *et al.* [153] also used the skeleton to drive the deformation of the model, and locally updated the FEM system. Furthermore, they introduced multi-grid solvers into the analysis of the stress distribution. For man-made models, they introduced iWIRES [96] to preserve the characteristic structure of the model.

3.1.2 Automatic Rigging

In addition to studying how to use skeletons to drive shape deformation, another research direction is how to bind a skeleton to the shape. This problem is called rigging. In the traditional workflow, this process often needs manual specification with the help of professional 3D modeling software. This process usually consists of two steps. The first one is to specify the joint positions and their connections, and the other is to determine the skinning weights which we have mentioned some methods above. There are some works [86, 154, 155, 156, 157, 158, 159, 160] that extract skeletons aiming to discover the shape topology, typically called curve-skeletons, while we focus on another type of skeletons, called bone-skeletons, which can be directly used for editing. For the early work, Baran *et al.* [110] proposed an automatic method, called Pinocchio, to generate skeleton and the skinning weights from a single shape. They fit a pre-defined skeleton template to the input shape so the method may fail when the shape structure is different from that of the skeleton. Feng *et al.* [161] transferred high quality rigging to input body scan with the help of the SCAPE model [162]. However, they only dealt with human body shapes. For multi-component characters which are easily accessible on the Internet, Bharaj *et al.* [163] proposed a method to automatically bind skeletons to the character models. The method builds a contact graph for the components of the input model and exploits graph clustering to obtain the target skeleton with corresponding skinning weights from the input animation skeleton. The mapping from the input skeleton to the target skeleton of the input model is achieved by a novel mapping

scheme based on dynamic programming.

Also, the quality of the skeleton extracted using the information from a dataset is better than that extracted from a single shape. Most works use a set of example poses to extract a hierarchical, rigid skeleton. Schaefer *et al.* [164] used clustering to find the rigid bones of the skeleton, and then solved for the vertex skinning weights which are further used to determine the joint positions and their connections. Aguiar *et al.* [165] bridged the gap between mesh animation and skeleton-based animation. They also first performed clustering to extract rigid bone transformations and then estimate joint motion parameters and appropriate surface skinning weights. Different from former methods that extract a skeleton from the examples of the same subject, Hasler er al. [166] estimated a rigid skeleton including skinning weights from examples of different subjects. The skeleton extracted by their method represents either shape variations or pose variations. With the combination of pose skeleton and shape skeleton, the user can control them independently. However, Le *et al.* [167] pointed out that these data-driven methods, on the one hand, use motion driven clustering which does not model the skeleton structure well, so some specific parameter settings are required. On the other hand, the step-by-step process will cause error accumulations. So they adapted skinning decomposition [168] and added soft constraints converting unorganized bone transformations to hierarchical skeleton structure. They overestimated the number of the skeleton bones during initialization, and exploited an iterative framework to automatically prune the redundant bones and update the skinning weights, joint location and bone transformation. The rigging results are shown in Fig. 10.

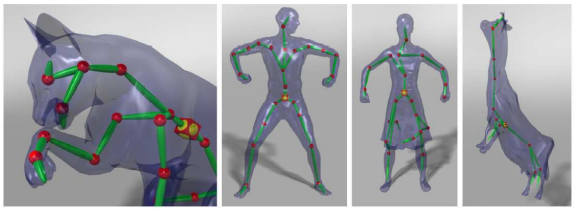


Fig.10. Example-based rigging results of [167].

3.2 Cage-Based Deformation

The cage-based deformation method is very similar to the skeleton-based deformation, but the difference is that the skeleton is generally inside the model, while the cage is generally wrapped outside the model. In essence, they both simplify the structure of the model and provide users with the handle to edit models. Free Form Deformation (FFD) [32] is first proposed to produce digital animation. This technique makes it possible to deform 3D shapes smoothly. Given the lattice vertices $\boldsymbol{v}_i, i = 1, \cdots, n$, we denote the new position of a point inside lattice as $\boldsymbol{p}'$, and low-frequency basis functions as ϕ_i, and then we can obtain the formulation as follow:

$$\boldsymbol{p}' = \sum_{i=1}^{n} \phi_i(\boldsymbol{p}) \boldsymbol{v}_i. \tag{6}$$

However, limited by the 3D control lattices, FFD is hard to realize complicated deformations like limb movements so that it is difficult to depict articulated shapes. Cage based deformation (CBD) is an extension of FFD. The control lattice is replaced by a polyhedral mesh which can better approximate the 3D shape and the deformation formulation is the same as Eq. 6.

3.2.1 Cage Prediction

The first thing of CBD is cage generation which can be divided into two kinds, automatic and user interactive. Automatic methods are typically completely geometric including mesh simplification [169, 170, 171, 172] and voxelization [173, 174]. But these methods tend to produce imperfect cages or sometimes fail. The inter-

active methods [175] allow users to add cage vertices to produce better cages for deformation but are more time-consuming. Ju *et al.* [176] proposed a data-driven method to exploit the cage template dataset created by artists for better cage selection in animation. Savoye *et al.* [177] proposed a linear cage estimation method for the target shape given the source shape and corresponding cage, which facilitates cage extraction and animation re-editing work.

3.2.2 Blending Weights Generation

The next step of cage based deformation is to establish the relationship between the cage and the interior shape. For this purpose, Mean Value Coordinates (MVC) are first introduced in [178, 179] and applied to the deformation for triangular meshes [44]. Hormann *et al.* [180] extended MVC to arbitrary polygon meshes. But these coordinates have a main drawback that they could be negative, which will produce unsatisfactory results. To avoid the negativeness, Joshi *et al.* [181] proposed Harmonic Coordinates which ensure positive values and produce more local deformations, but the computation is time-consuming. Lipman [182] improved MVC to avoid negative values, utilizing GPU visibility rendering. Langer *et al.* [183] generalized MVC and vector coordinates [184] to spherical barycentric coordinates which are defined for arbitrary polygonal meshes on a sphere. Those coordinates can also be integrated into existing space-based deformation frameworks. Later on, Lipman [185] found that the details of mesh surfaces are not retained when confronting large-scale deformations. In previous methods like MVC and Harmonic Coordinates, only cage vertex positions are considered. Therefore, he suggested to relate the cage's face normals to the interior vertices and proposed new coordinates called Green Coordinates. The Green Coordinates are further extended to complex domains, making the deformation better fit the user's input [186]. Unlike the original Green Coordinates, which associate the face normals with the vertex positions, the function of the face normals and the function of the vertices in [187] are independent, providing a higher degrees of freedom and a larger deformation space.

Yang *et al.* [188] added global and local stiffness control to the lattice-driven shape deformation. The global stiffness is provided by the width of overlapping lattice cells, and the local stiffness is controlled by the stiffness coefficient. The deformation of the lattice is transferred to the embedded shape by bilinear or trilinear interpolation. Manson and Schaefer [189] proposed moving least squares coordinates which suffer from the same problem on boundary edges as MVC and Hermite MVC [190] when used for deforming concave shapes. Weber *et al.* [191] further proposed biharmonic coordinates derived from the solutions to the biharmonic equation. They also presented a thickness-preserving deformation method which is better than As-Similar-As-Possible (ASAP) and ARAP methods [47]. In the context of transfinite interpolation, Li *et al.* [192] proposed Cubic Mean Value Coordinates (CMV). Cage-based deformation is essentially a series of interpolation approaches, which interpolate the control vertices of the cage, so CMV can also be used for cage-based shape deformation, as shown in Fig. 11. They showed shape deformations under the control of cage networks consisting of straight and curved edges.

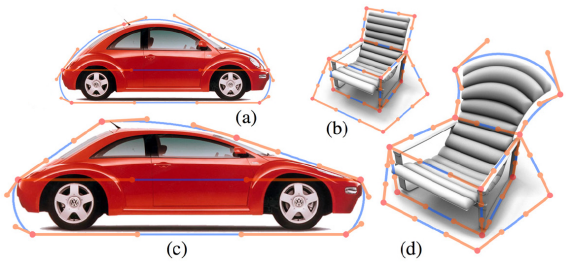


Fig.11. Deformation results using curved edge networks with cubic mean value coordinates [192]. (a) (b) are source models and (c) (d) are edited results.

Most of barycentric coordinates are global, that is, the vertices on the deformed model are determined by the weighted sum of *all* vertices on the cage, which will cause some counter-intuitive deformations, losing good controls for local variations. On the one hand, even for not too many vertices (50-100 vertices) on the cage, the calculation process is time-consuming and may not achieve real-time. On the other hand, since the coordinates are decreasing functions of distance, such as Euclidean distance [44] or geodesic distance [181], then there are some vertices on the cage that may have little influence on a single vertex of the deformed mesh. So reducing the number of weights is necessary and feasible. Based on these observations, Landreneau *et al.* [193] proposed a Poisson-based weight reduction method which can reduce the number of weights (control points) that affect a single vertex to a user-specified number, while preserving the deformation results. The method requires a certain number (typically 4-6) of example poses in the optimization to achieve better results, and the minimization energy is obtained from Poisson equation solved for the weights by Lagrange multipliers. Their method is also applicable to other deformation methods that require weights, such as skeleton-based deformation methods. However, imposing the sparsity constraint may obtain suboptimal solutions, which will lead to non-smooth results or even poor approximation results; and sometimes there are exceptional vertices, which are affected by more bones or control points than the preset threshold. Therefore, Le *et al.* [194] proposed a two-layer blend skinning model that performs lossy weight matrix compression to avoid imposing sparsity constraints. They added virtual bones as an intermediary between the original bones and vertices. They first blended the transformations of the original bones to obtain the transformations of the virtual bones, and then blended up to two virtual bones to obtain the transformation of each vertex. Although mainly dealing with skeleton-based deformation in their paper, their method could also be used in cage-based deformation after combining with an objective function similar to the one in [193]. Similar to enhancing the locality of the deformation, Zhang *et al.* [195] proposed Local Barycentric Coordinates (LBC) for better local deformation. They introduced total variation (TV) originally used in image smoothing and reconstruction [196], minimizing which under a couple of constraints of partition of unity, reproduction, and non-negativity. The deformation using LBC can realize multi-scale high-quality editing without any other manual specification. However, LBC has no closed-form expression and must solve a time-consuming optimization problem when dealing with dense mesh models, as pointed out in [197]. So Tao *et al.* [197] proposed a new efficient solver for the optimization of LBC.

Some works exceed the limits of single cage and lattice. Instead of using a polyhedral mesh cage or control lattice, Botsch *et al.* [198] proposed to use small voxels to enclose the model for space-based deformation. They defined a nonlinear elastic energy which supports large-scale deformations. However, the discretization may cause some aliasing problems. Li *et al.* [199] proposed a method to directly interpolate points on the mesh without constructing a whole cage for the mesh; instead, they only built an umbrella-like cell interac-

tively on the partial mesh where users are interested. Considering that Green Coordinates cannot ensure conformal deformations with an open cage or umbrella-like cell, they also took the local deformation differences of the cage into account. Replacing the cages with the Interior Radial Basis Functions (IRBFs) center points, Levi *et al.* [200] improved the cage-based deformation methods based on VHM (Variational Harmonic Maps) method [187]. The harmonic basis functions are replaced by IRBFs which are defined with respect to centers on the surface of the model. They also placed a set of spheres inside the model to minimize local distortions by preserving the shapes of the spheres. Aiming for multi-level detail and high-quality deformations, Garcia *et al.* [201] proposed a cage-based deformation method based on star-cages instead of a single cage as traditional methods do. The star-cage consists of multiple cages that offer easier interaction compared with the single cage. Based on a new representation, sphere-meshes, which can approximate shapes, Thiery *et al.* [202] used the sphere-mesh hierarchy as a deformation handle to deform shapes well. They [203] further apply this representation to the approximation of animated mesh sequences and the skinning weights obtained by skinning decomposition can guide pose editing well.

In the traditional FFD or cage-based deformation, after determining the lattice or cage, without re-parameterization, the user can only use the existing handle to deform the shape. Zhang *et al.* [204] proposed a control lattice with adjustable topology, which does not need to re-parameterize the relation between lattice and enclosed shape again after changing the lattice. This method uses a tailored T-spline volume to create the lattice and further uses a refinement algorithm to obtain a proxy, which is a simplified version of the lattice and fits the enclosed shape better. The user manipulates the proxy, driving the deformation of the lattice and the deformation is then transferred to the shape. There are fewer vertices on the proxy than the lattice, which is more convenient to manipulate. However, the method is essentially based on volumetric lattice, which is not as flexible as cage-based deformation in large-scale deformation.

4 Data-Based Deformation with Numerical Models

With the development of 3D scanning and registration techniques, geometric shape datasets [162, 205] are becoming increasingly available on the Internet. Analyzing the existing shapes from the shape dataset to provide prior information for deformation becomes an attractive direction. We summarize these as data-driven mesh deformation methods. The structural and semantic knowledge of the man-made model can also be obtained by analyzing multiple models. We summarize these as data-driven analysis for man-made models methods.

4.1 Data-Driven Mesh Deformation

Aforementioned geometry-based methods have some essential weaknesses. On the one hand, they are prone to producing unreasonable deformation results when the user's interaction is insufficient. On the other hand, they have high requirements for meshes and different models tend to require different parameter settings. To address these limitations, data-driven methods exploit plausible deformations from shape datasets and can produce more natural deformation results without manual selection of parameters or a large amount of user constraints. An important pioneering work is Mesh based Inverse Kinematics (MeshIK) [206], based on which a series of works has been proposed to improve or extend the method. We will group these meth-

ods according to different deformation representations of shapes. Another active research area is not limited to editing model pose, but also considering shape.

4.1.1 Blend Mesh Representation

In data-driven deformation of mesh models, the deformation representation is important for representing the mesh model. Euclidean coordinates are the most straightforward way to represent the model, but there are obvious limitations on rotations.

Gradient-Based representation. Deformation gradient is a straightforward gradient-based representation, which is defined as the affine transformation that optimally describes the mapping of local neighborhoods (triangles or one-ring neighbors) from the source mesh to the target mesh. Sumner and Popović [207] used deformation gradients to transfer the deformations between two mesh shapes. Further, Sumner *et al.* [206] proposed MeshIK, a method based on principal component analysis (PCA) to analyze the shape dataset and use the weighted combination of deformation gradients to edit shapes. MeshIK is used to produce stylized surface deformation and in analogy to traditional skeleton-based inverse kinematics for posing skeletons, and hence the name of MeshIK. Each example shape is represented using a feature vector, containing deformation gradients of triangles describing deformation from a reference model. The deformation gradient has a good property that it is a linear function of the mesh vertices. They further decompose the deformation gradient $\boldsymbol{T}_{ij}$ for the j-th triangle in the i-th shape into rotation and scaling/shear components using polar factorization $\boldsymbol{T}_{ij} = \boldsymbol{R}_{ij}\boldsymbol{S}_{ij}$. The rotation is not linear so if it needs to be interpolated linearly, one can map the rotation from 3D rotations $\mathbf{SO}(3)$ to $so(3)$ of skew symmetric 3×3 matrices [208]. The mapping uses the matrix logarithm and can be reversed by the matrix exponential [208]. Then the nonlinear span of the deformation gradient for the j-th triangle given m example meshes has the formulation as Eq. 7:

$$\boldsymbol{T}_j(\boldsymbol{w}) = \exp(\sum_{i=1}^{m} w_i \log(\boldsymbol{R}_{ij})) \sum_{i=1}^{m} w_i \boldsymbol{S}_{ij}. \quad (7)$$

This constitutes the nonlinear feature space, where w_i is the combination weight for the deformation gradient from the i-th shape. As shown in Fig. 12, given different example models, the editing will produce different results.

Der *et al.* [209] proposed a reduced model for inverse kinematics which is faster than MeshIK [206]. They clustered the vertices according to the influence of the control parameters, and replaced the same cluster of vertices with a proxy vertex located at the weighted centroid of the cluster. The method takes advantage of the reduced complexity of deformation proxies, not relying on the geometric complexity of the original mesh to interactively edit even extremely detailed geometry models. Wampler [210] exploited the ARAP energy [48] for interpolation between a set of example shapes. The method allows spatially localized interpolation which has more natural transitions. However, it also suffers from the problem of potential non-local editing and requiring to solve a complicated system of equations when a large number of examples are given.

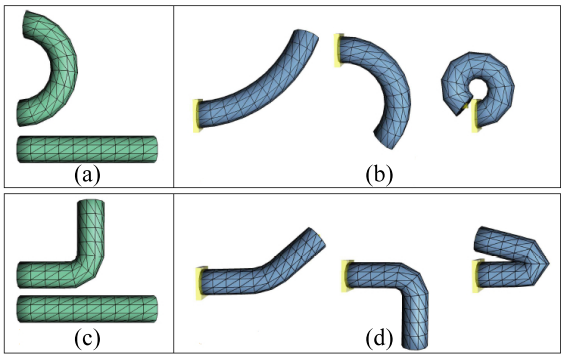


Fig.12. Given different examples, MeshIK [206] will produce different deformation results. (a) (c) are example bends and (b) (d) are outputs.

MeshIK cannot deal with large-scale deformations where rotations are larger than 180°. Gao *et al.* [71] proposed a shape editing method based on ACAP (as-consistent-as-possible) deformation features to address this problem. The rotation at each vertex can be represented using an axis-angle representation. However, the direction of the axis (one of the two opposite directions) and the rotation angle (with multiples of 2π added) are ambiguous. They proposed an integer optimization strategy to eliminate the ambiguities, so the proposed feature can express rotations greater than 180°. The method further introduces sparsity constraints into model editing that utilizes the prior information from the model dataset to automatically select a smaller number of basis deformations. It also supports multi-scale editing with high efficiency, as shown in Fig. 13.

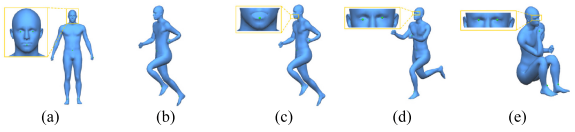


Fig.13. Using ACAP features [71] along with sparsity constraints enables multi-scale editing. (a) is the reference model. (b) is the deformation result with the simplified mesh. (c)-(e) are the deformed results on the high resolution mesh with both facial and body deformations. Their method automatically selects suitable basis modes for both small-scale facial expression editing and large-scale pose editing.

Rotation-Invariant representations. Another direction of research to tackle rotation ambiguities is to develop rotation-invariant representations. Lipman *et al.* [211] locally defined linear rotation-invariant (LRI) coordinates at each vertex which consist of two discrete forms. The discrete form coefficients w.r.t. orientation can be used to represent the mesh that facilitates detail-preserving surface editing and shape interpolation. Changing the definition domain from one-ring neighborhoods of the vertices to mesh patches, Baran *et al.* [212] proposed patch-based rotation-invariant coordinates, which solve the noise sensitivity problem of the original LRI [211] and accelerate the shape reconstruction. They used patch-based LRI coordinates to project the shape into the shape space and transfer semantic deformations to the target shape. The patch-based LRI representation is further used in data-driven shape interpolation and morphing [213] which provides an interface for users to intuitively edit the morphing results. Kircher and Garland [214] proposed a differential rotation-invariant surface representation for surface editing. The second-order differences are both rotation-invariant and translation-invariant. The editing can be operated both in time and space. Winkler *et al.* [215] used the edge lengths and dihedral angles as a representation for multi-scale shape interpolation. Their method supports input settings for more than two shapes. Further, Fröhlich and Botsch [216] proposed to use edge lengths and dihedral angles to represent shape deformation. However, since edge lengths cannot be negative, the method cannot handle extrapolation deformation well. Gao *et al.* [217] proposed a data-driven shape editing method based on a novel rotation-invariant representation named RIMD (Rotation Invariant Mesh Difference). They decomposed the deformation gradient into rotation and scaling/shear matrices, and combined the logarithm of the rotation difference of each edge and the scaling/shear matrix of each vertex to represent the shape. As shown in Fig. 14, the rotation difference cancels out global rotations, making it a rotation invariant representation and thus it can handle large-scale deformations. However, when applied for data-driven deformation, it uses global principal components extracted from example shapes, so it is difficult to perform local editing. Besides, the derivatives are calculated in a numerical way, which restricts the editing efficiency.

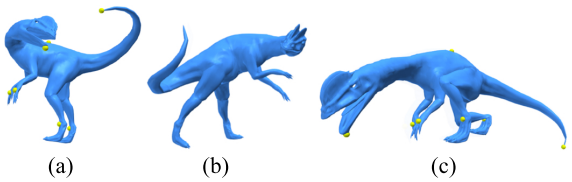


Fig.14. RIMD features can handle large-scale deformations [217]. (a) (b) are the input shapes and (c) is the result.

Generalized to deforming mesh sequences, Xu *et al.* [218] proposed a keyframe based mesh editing method. Once the constraints are specified by users or induced from the environment, the frames with those constraints become keyframes. And the constraints and deformations will be propagated to the whole mesh sequence. Instead of directly editing the input representation of the model, Sumner *et al.* [219] proposed to embed the model into a deformation graph which is built by uniformly sampling on the model surface. The graph node j is associated with an affine transformation $\boldsymbol{R}_j$ and a translation vector $\boldsymbol{t}_j$ which can map the point $\boldsymbol{p}$ to a new position by $\boldsymbol{p}' = \boldsymbol{R}_j(\boldsymbol{p} - \boldsymbol{g}_j) + \boldsymbol{g}_j + \boldsymbol{t}_j$, where $\boldsymbol{g}_j$ is the position of the graph node. Assuming there are m graph nodes, then the final deformed position $\boldsymbol{v}'$ of the model vertex $\boldsymbol{v}$ will be determined by the weighted sum of all influences, as shown in Eq. 8

$$\boldsymbol{v}' = \sum_{j=1}^{m} w_j(\boldsymbol{v}_i)[\boldsymbol{R}_j(\boldsymbol{v}_i - \boldsymbol{g}_j) + \boldsymbol{g}_j + \boldsymbol{t}_j]. \qquad (8)$$

In addition to editing the mesh models, this method can also perform particle simulation, but the disadvantage is that the local details cannot be edited.

Deformation components. Given a dataset, we can extract the deformation components of the shape and manipulate the basis to achieve the purpose of editing the shape. Early work [220] employed principal component analysis (PCA) to extract the deformation components, but the extracted components are global, which are not convenient for users to manipulate directly. Therefore, in combination with sparsity, a series of works propose the extraction of sparse deformation components. For the first work, Neumann *et al.* [221] proposed to decompose the animated mesh sequences into sparse localized deformation components (SPLOCS). Those components form a spatially localized basis which captures semantic deformations. The user can edit the shape by manipulating those components. However, the method operates on vertex coordinates, which are translation and rotation sensitive and thus cannot handle large rotations. Huang *et al.* [222] used the deformation gradient to represent the shape, and decomposed the deformation gradient into rotation and scale by polar decomposition, and finally used SPLOCS on those vector representations. But this method still cannot handle rotations larger than 180°. Bernard *et al.* [223] also aimed to find local support deformation components from the example shapes in the dataset. They used matrix factorization with sparsity and graph-based regularization terms, accounting for smoothness to automatically select the position and size of the local support component. Adopting a rotation-invariant representation, Wang *et al.* [224] extended [221] using the shape representation of edge lengths and dihedral angles. However, the problem that the extrapolation may fail due to the fact that the edge length cannot be negative still exists, and the insensitivity to scale will lead to lack of robustness to noise. The edge length and dihedral angle representation is also used in [225], which analyzes the edge length vectors and the dihedral angle vectors respectively to extract the adaptive sparse deformation components. Then, by adapting [216], the method allows users to directly edit vertices and produces deformation results under the guidance of components. Based on Nonlinear Rotation-Invariant Coordinates (NRIC) [226, 227], Sassen *et al.* [228] combined the advantages of principal geodesic analysis [229] and SPLOCS [221] and proposed

Sparse Principal Geodesic Analysis (SPGA) on the Riemannian manifold of discrete shells.

4.1.2 Blend Shape and Pose

This series of methods model the human body through several parameters (often related to shape and pose of the body), and the editing of the human body can be achieved by different parameter inputs. As one of the pioneering works and also one of the most successful works, SCAPE [162] uses the deformations of the triangular faces to represent the body shape and pose, separately. The follow-up work, Skinned Multi-Person Linear model (SMPL) [230], decomposes the body shape into identity-dependent shape and non-rigid pose-dependent shape with a vertex-based skinning approach, such as LBS and DQBS. Given shape parameters $\boldsymbol{\beta} \in \mathbb{R}^{\|\boldsymbol{\beta}\|}$ and pose parameters $\boldsymbol{\theta} \in \mathbb{R}^{\|\boldsymbol{\theta}\|}$, they propose to represent the neutral mesh $T(\beta, \theta)$ by adding a blend shape function, $B_S(\boldsymbol{\beta})$, which sculpts the subject identity, and a pose-dependent blend shape function, $B_P(\boldsymbol{\theta})$ to a mean mesh template $\bar{\boldsymbol{T}}$,

$$T(\beta, \theta) = \bar{\boldsymbol{T}} + B_S(\boldsymbol{\beta}) + B_P(\boldsymbol{\theta}). \tag{9}$$

The neutral pose is then deformed by some blend skinning methods,

$$M(\beta, \theta) = W(T(\beta, \theta), J(\beta), \theta, \boldsymbol{W}),$$

where $W(\cdot)$ represents a standard blend skinning function, and $\boldsymbol{W}$ is the skinning weights. $J(\beta)$ is a function that determines the joint locations, which transforms rest vertices into rest joint locations.

Although SMPL can model human bodies well, it lacks modeling of non-rigid dynamic deformations caused by body motions. To model them, the Dyna model [231] proposes to use a second-order auto-regressive model which predicts soft-tissue deformations. Specifically, it represents non-rigid deformation of a body, $\hat{\boldsymbol{T}}(\beta, \delta)$, by the combination of identity and soft-tissue deformations,

$$\hat{\boldsymbol{T}}(\beta, \delta) = \boldsymbol{S}(\beta) + \boldsymbol{D}(\delta).$$

Further, different from SMPL, Dyna follows the similar idea of SCAPE [162], which describes different human bodies by triangular deformations. Given the edge $\hat{\boldsymbol{e}}_i$ of triangle i in the template mesh, the edge $\boldsymbol{e}_i$ of triangle i belonging to the mesh at time t can be represent as,

$$\begin{aligned} \boldsymbol{e}_i(\beta, \theta_t, \delta_t) &= \boldsymbol{R}_i(\theta_t)\hat{\boldsymbol{T}}_i(\beta, \delta)\boldsymbol{Q}_i(\theta_t)\hat{\boldsymbol{e}}_i \\ &= \boldsymbol{R}_i(\theta_t)(\boldsymbol{S}_i(\beta) + \boldsymbol{D}_i(\delta_t))\boldsymbol{Q}_i(\theta_t)\hat{\boldsymbol{e}}_i, \end{aligned}$$

where β and θ are still body shape coefficients and body pose parameters, respectively. $\boldsymbol{Q}_i(\theta_t)$ represents pose dependent deformations which are a linear function of θ, $\boldsymbol{S}_i(\beta)$ represents identity-dependent transformations which are a linear function of β, $\boldsymbol{R}_i(\theta_t)$ represents absolute rigid rotations, and $\boldsymbol{D}_i(\delta_t)$ represents dynamics-dependent deformations which is a linear function of coefficients δ_t. Dynamics deformations are related to body motion; thus, velocities and accelerations. So, the angular velocity and acceleration $(\dot{\theta}_t, \ddot{\theta}_t)$ of body joints and the velocity and acceleration (v_t, a_t) of the root of the body at time t are also the inputs of the model. Let $\hat{\delta}_{t-1}$ and $\hat{\delta}_{t-2}$ be the coefficients representing the history of estimated low-dimensional dynamic deformation, and the dynamic control vector of the Dyna model is $\boldsymbol{x}_t = \{\dot{\theta}_t, \ddot{\theta}_t, v_t, a_t, \hat{\delta}_{t-1}, \hat{\delta}_{t-2}\}$ in total. Dynamics deformations also depend on the body shape, which is specified by the shape identity coefficients β. The dynamics-dependent deformations $\boldsymbol{D}_i(\delta_t)$ can be further specified as $\boldsymbol{D}_i(f(\boldsymbol{x}_t, \beta))$, where f is a function to be learned that maps dynamic control vector $\boldsymbol{x}_t$ and shape coefficients β to the low dimensional representation δ_t of the dynamics.

SMPL can also be extended to model those dynamic deformations by adding dynamic blend shape function,

$B_D(\boldsymbol{x}_t, \beta)$ to Eq. 9,

$$T(\beta, \theta_t, \boldsymbol{x}_t) = \bar{\boldsymbol{T}} + B_S(\boldsymbol{\beta}) + B_P(\boldsymbol{\theta_t}) + B_D(\boldsymbol{x}_t, \beta),$$

where $B_D(\boldsymbol{x}_t, \beta)$ also predicts vertex offsets. This model is named as Dynamic SMPL, abbreviated as DMPL.

4.2 Data-Driven Analysis for Man-Made Models

Data-driven analysis for man-made model editing is to learn some prior information from a model dataset that contains closely related models, such as those belonging to the same category or having the same style. The prior information provides plausible variations of the models and adds constraints to user editing which ensures reasonable results. [17] reviews the methods of data-driven analysis and processing.

4.2.1 Interactive Editing

Fish *et al.* [232] proposed meta-representation to represent the essence of 3D man-made model dataset. The representation is formulated from the correspondence between model segmented parts, which encodes the arrangement rules of the parts. So it can be viewed as a constraint guiding user editing, where models can maintain their familial traits and performing coupled editing, where several shapes can be collectively deformed by directly manipulating the distributions in the meta-representation. Yumer *et al.* [233] abstracted co-constrained handles for model editing. The handles are obtained from the different segmented parts through a co-abstraction method [234]. The co-constraints are generated by clustering the different planes of the abstracted parts. This method supports interactive editing by not only abstract handles but also sketches, as shown in Fig. 15. Based on this work, Yumer *et al.* [235] further proposed a semantic editing method for 3D models, where users can edit 3D models through semantic attributes. This method establishes a continuous mapping between semantic attributes and model geometry through the relative scores of attributes and geometric differences between models. Although the deformation is continuous, this method cannot add or remove certain parts of the model. The above methods all use the dataset of some categories to learn deformation constraints to edit shapes. These methods have been able to take advantage of the information in the shape dataset, and those pairwise parameter constraints work well during shape editing. However, their parameter pairs are of the same kind, and the constraints on the parameter pairs that may be formed by different kinds of parameters are not considered. To address this, [236] uses multivariate regression methods to learn the correlation between parameters. The proposed method can perform both structure-preserving and structure-varying shape editing. Laga *et al.* [237] analyzed the pairwise co-variation of the geometry and structure of the part. After the user edits a part of the model, the method can automatically find a suitable configuration for the entire model to ensure the plausibility of the edited model.

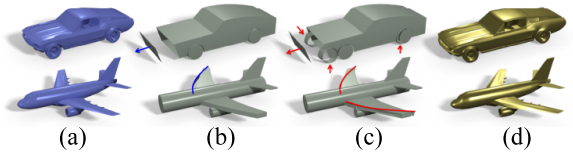


Fig. 15. [233] supports interactive editing through not only abstract handles but also sketches. (a) Input models. (b) User prescribed deformation (top: translation, bottom: silhouette sketching). (c) Constraints resolved by the system. (d) Final models.

4.2.2 Editing for Other Purposes

From a relatively novel perspective, Zheng *et al.* [238] wanted to change the model to suit for the input human body. As shown in Fig. 16, the input is a model with semantic labels, and a spatial relationship graph is used to represent the model, where the

graph nodes represent the model components, and the edges of the graph represent the spatial relationships of the components. They first established the contact constraints between the body skeleton and the model (such as buttocks and chair seats). Then the deformation is an optimization process, and an edit propagation algorithm is designed to deform the model according to these constraints while maintaining the model structure.

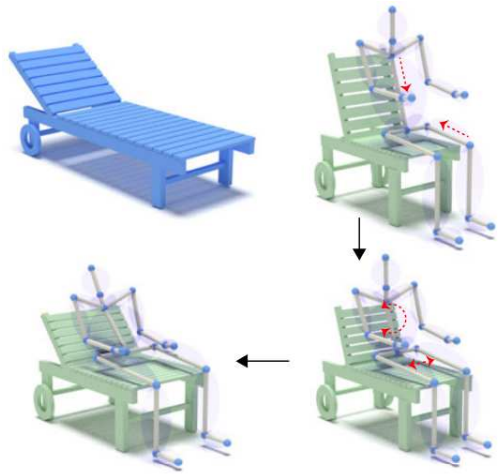


Fig.16. The model changes according to the change of the skeleton [238].

Model editing can also be used for other applications. For example, Ovsjanikov *et al.* [239] explored a shape dataset through the deformations of a template shape which abstracts the shape structure using several boxes. Ishimtsev *et al.* [240] proposed a data-driven mesh deformation method, named CAD-Deform, to fit the retrieved synthetic CAD models to the real 3D scan. The deformation energy ensures smooth deformation and keeps sharp features, which also includes a part-to-part mapping term, and nearest-neighbor mapping term. The former matches the deformed mesh and the target scan globally, while the latter makes them match more accurately when they get close enough.

5 Neural Shape Editing

In this section, we review attempts at the deformation methods based on deep learning in recent years. Combining with deep learning brings new opportunities and challenges to both organic and man-made shape editing methods.

5.1 Organic Shape Editing

With the availability of large human body datasets [231, 241], deep neural networks have also been introduced into the editing of organic shapes.

5.1.1 Editing via Learning Mesh Deformation

Tan *et al.* [242] first proposed to use a variational autoencoder (VAE) to encode shape deformations. They used the RIMD deformation feature [217] as input and can generate different poses after learning from the existing deformations in the dataset. The latent space can be used for shape exploration, which guides the user to find specific shapes they want. However, the network is entirely composed of fully connected layers, which has high memory consumption and thus, cannot handle dense mesh models. To solve this, graph-based convolutions [243] and mesh pooling [244] have been introduced. At the same time, they [245] proposed a convolutional mesh autoencoder utilizing the locality of the convolution operator and sparsity constraints to extract the local deformation components of the deformable shapes. The extracted deformation components can be used to synthesize new shapes. Also for extracting deformation components, Yang *et al.* [246] proposed to use multi-level VAEs, which can extract multi-scale deformation components, achieving better results. Qiao *et al.* [247] proposed bidirectional LSTM with graph convolutions to generate mesh sequences. As an application of shape deformation and editing, deformation transfer can transfer the user's editing of one

shape to another shape. Traditional deformation transfer [207, 212, 170] requires manually specifying several correspondences between source and target shapes. Although Yang *et al.* [70] proposed a method of automatically selecting appropriate key points to transfer the deformation, some candidate points still need to be manually specified. So Gao *et al.* [248] first proposed a fully automatic shape deformation transfer method. They trained SimNet, a neural network that predicts the similarity of two poses of source and target shapes. The proposed VC-GAN combines MeshVAE and CycleGAN [249] to transfer the latent vectors of two input shapes, enabling unpaired deformation transfer. Using shape deformation representations and neural networks, some works [250, 251] achieve super-resolution on cloth simulation. More shape analysis and shape generation works based on shape deformation are illustrated in Fig. 17. Wang *et al.* [252] represented 3D human meshes by a series of parameters including shape, pose, and vertex order, to perform deformation transfer. The method first encodes these parameters of the source shape by a permutation invariant encoder to extract pose feature and then uses a style transfer decoder together with the target identity mesh as condition to generate the target shape with the source pose.

5.1.2 Performing Mesh Deformation

Bailey *et al.* [253] proposed a convolutional neural network for approximating facial deformations which can handle high-frequency deformations. The method separates the process into three parts: a coarse approximation, a refined approximation and an approximation for rigid components of the mesh. The coarse and refined approximations are comprised of two independent convolutional networks by inputting rig parameters. For those segments that only undergo rigid rotations and translations, they are approximated by a faster rigid approximation rather than convolutional networks to improve efficiency. The method also proposes a feed-forward neural network to output rig parameters given user-specified control points for inverse kinematics.

The skinning technique also benefits from neural network based methods in deformation, skeleton and weights prediction. As the first work to introduce the neural network in character deformation, Bailey *et al.* [254] split skinning deformation into linear and nonlinear portions. The linear portion is approximated by a linear skinning method while the nonlinear portion is approximated by a neural network consisting of two fully connected layers. Based on the similar idea that decomposing the deformation into linear and nonlinear parts, Li *et al.* [255] proposed a graph attention based network to predict the nonlinear effects by inputting mesh graphs and linear deformations, while the linear part is computed with LBS. Liu *et al.* [256] also proposed a neural skinning method which utilizes graph convolutions. They first constructed a graph using the input 3D mesh with its associate skeleton hierarchy. Each graph node encodes the mesh-skeleton attributes. The graph and node attributes are fed into their graph convolution network to predict the skinning weights.

Almost at the same time, Xu *et al.* [257] proposed to convert an input 3D shape into a set of geometric representations expressed in a volumetric grid. The input representation is processed through a stack of 3D hourglass modules. Each module outputs joint and bone probabilities in the volumetric grid, which are progressively refined by the following module. The final joint and bone probabilities are processed through a Minimum Spanning Tree (MST) algorithm to extract the final skeleton. They further proposed RigNet [8], which can predict a skeleton with the skinning weights for the input model with the network shown in Fig. 18.

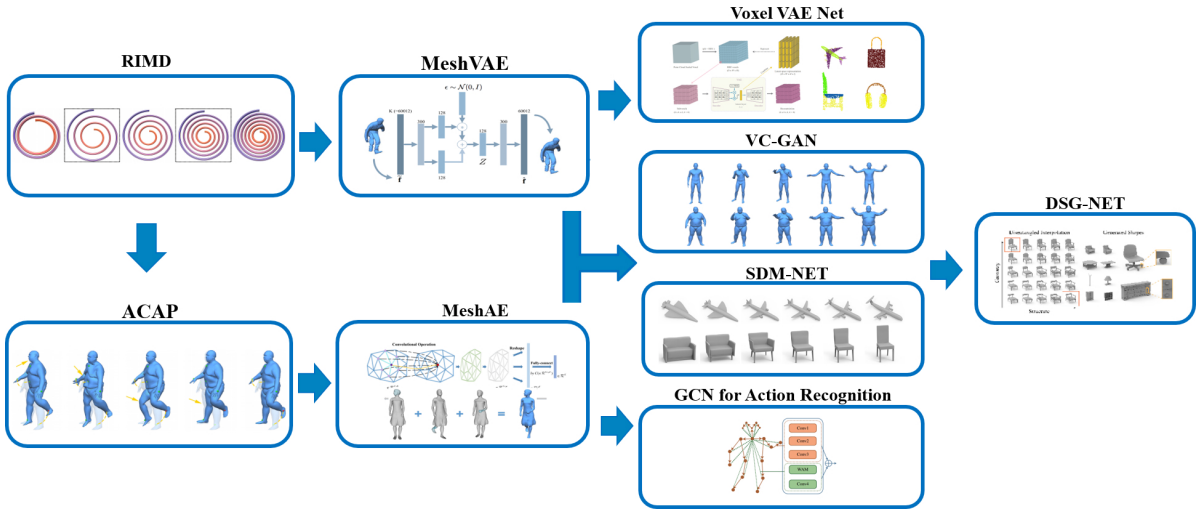


Fig.17. Shape editing, analysis and representation based on shape deformation feature, proposed by the geometry learning group, ICT, CAS.

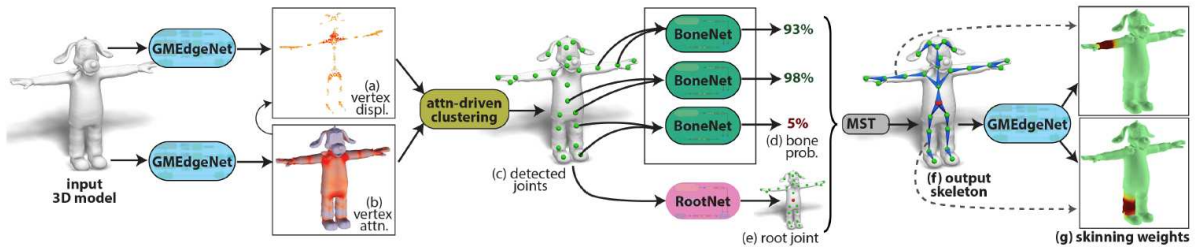


Fig.18. The network architecture of RigNet [8].

The method first extracts the geometric features from the input mesh and predicts candidate joint locations and an attention map indicating the confidence of each candidate joint. After the joints are detected, another network learns to determine the root joint and predict whether there is an edge connecting two joints. Finally, a Minimum Spanning Tree algorithm is performed to generate the final skeleton which is sent to another network to predict skinning weights. The method also considers user inputs like how many joints are wanted. The predicted skeleton and skinning weights can be directly used for editing and modeling. Vesdapunt *et al.* [258] proposed a joint-based representation for 3D face models which rigs semantic joints to the face model. The specified joints add prior information which reduces the demand for large amounts of training data. They also proposed an autoencoder network to predict the skinning weights which not only enhances the modeling capacity, but also supports users to edit the model.

NNWarp [259] designs a heuristic deformation feature vector including geodesic, potential and digression, and warps linear elastic simulations into nonlinear elastic simulations via a DNN prediction to handle a wide range of geometrically complex bodies, which is faster than existing nonlinear methods. Fulton *et al.* [260] compressed the solid dynamics deformation space into a nonlinear latent space with fewer degrees of freedom through a neural network, while achieving equivalent or even greater simulation fidelity, speed and robustness compared with other model reduction methods. They used the autoencoder architecture and initialize the outer layer with the basis computed by PCA.

Also based on an autoencoder, Santesteban *et al.* [261] combined the non-linear deformation subspace with a regressor composed of a recurrent architecture GRU (Gated Recurrent Unit) which regresses soft-tissue deformations. They propose that the soft-tissue deformations are not only related to shape and pose, but also motion, so the regressor also uses the motion descriptor as input. NASA [262] and NiLBS [263] condition the implicit field of articulated shapes on the skinning weights, enabling fast shape query without extra acceleration data structures.

5.2 Man-Made Models Editing

Some large man-made model datasets [264, 9, 265, 266] are also available on the Internet, which are the basis for some works that try to combine 3D shape editing with neural networks to realize the intelligent editing of 3D shapes.

5.2.1 Appearance Editing

Some methods are based on volumetric representations. For example, Yumer *et al.* [267] realized the semantic deformation of 3D shapes by a 3D volumetric convolutional network, predicting deformation flow from semantic attributes. However, each semantic attribute is only described by three numbers (0, 0.5, 1.0, indicating decreasing, keeping the same and increasing respectively), which lacks the degree of freedom and controllability of user editing. Liu *et al.* [268] realized interactive 3D modeling using adversarial generative networks, as shown in Fig. 19. But the edited object is a voxel-based shape, lacking geometric details, and the resulting shape may not be in line with the user's intentions. As for mesh representation, mesh models generally have inconsistent connectivity. Umetani *et al.* [269] presented a parameterization method for efficiently converting an unstructured mesh into a manifold mesh with consistent connectivity using depth information. The parameterization is then fed into an autoencoder, and the plausible deformation space is represented by the latent space of the autoencoder. The users can explore shape variations by directly manipulating the mesh vertices through an interactive interface. Also using autoencoder to optimize on the manifold, DiscoNet [270] believes that even if the 3D models belong to the same category, they generally do not lie on a connected manifold. So the authors proposed to use multiple autoencoders (two in their paper) to learn different connected components of the disconnected manifold, without any supervision. Extending the traditional cage based deformation, Wang *et al.* [271] proposed a neural architecture that predicts source cage and cage offset. The mean value coordinates are computed by a novel MVC layer, and a cage-based deformation layer produces the deformed result from the cage offset and mean value coordinates. Also inspired by traditional deformation methods, Liu *et al.* [272] proposed to use meta-handles, the combinations of control points, as deformation handles and biharmonic coordinates [273] to edit the 3D models. The control points are sampled by farthest point sampling, and meta-handles are predicted by MetaHandleNet. The meta-handles reflect the correlation between the control points. For example, the control points on the two chair armrests should maintain the symmetry of the chair armrests during deformation. At the same time, the plausible deformation range is predicted, and the specific deformation parameters are predicted by DeformNet to deform the source shape to match the target shape. They also proposed to use a soft rasterizer [274] and a 2D discriminator network to ensure reasonable and realistic deformation.

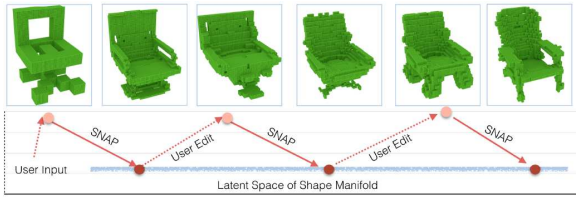


Fig. 19. An example of interactive neural editing of man-made 3D models. The user edits the model, and the network maps it to a latent space, and a new model is generated. The result is a voxel-based model with less geometric details [268].

DualSDF [275] uses a two-level representation to perform interactive shape editing and learn a tightly coupled latent space for the two-level representations by a variational autodecoder (VAD) framework [276]. The editing operations are performed on the coarse primitive-based representation, and the deformation results are presented as signed distance fields. Deng *et al.* [277] proposed a deformed implicit field network (DIF-Net) to represent 3D shapes and perform editing. The user freely selects one or more 3D points among the surface and specifies their new positions. The edited shape is obtained from the latent optimization. The editing also supports adding new structures to the given shape. Also utilizing deformation from a template SDF to represent 3D shapes, Zheng *et al.* [278] are able to manipulate shapes, but this is limited to mesh stretching.

Wei *et al.* [279] proposed an encoder-decoder network to edit shapes by editing semantic parameters like the height, depth, and width of each semantic part of man-made objects. Their method can be divided into two stages: semantic parameter encoding and deformation transfer. To provide semantic parameter supervision for the encoder, they first generated ground truth semantic parameters for shapes synthesized by bounding boxes of segmented shapes in the real dataset. They also edited these corresponding synthetic shapes and get corresponding semantic parameters. After encoding the original synthetic shapes and deformed synthetic shapes into the semantic latent space, a decoder uses shape-level Chamfer distance supervision to reconstruct both original shapes and deformed shapes. At the inference time, the network encodes a realistic shape into the parameter space, and after editing the shape parameters in the parameter space decodes the reconstructed synthetic shape and edited synthetic shape. As for deformation transfer, by defining a deformation field as the vertex displacements on the decoded synthetic shape, where each vertex on the real shape finds k nearest points on the synthetic shape and regards the weighted sum of the displacements of these nearest points as the vertex displacement of the realistic shape. In this way, deformation is transferred from the synthetic shape to the realistic shape. In addition, this method can be easily applied to non-rigid models by changing the definition of the semantic parameters, e.g., pose and shape for human bodies.

Sung *et al.* [280] embedded shapes into an idealized latent space where points represent shapes and vectors between points represent shape deformations. The deformation vector can be decoded into a deformation action which can be applied to new shapes directly.

5.2.2 *Other Forms*

In addition to geometry, structure is also editable. Mo *et al.* [281] developed a deep neural network based on structural shape representation StructureNet [282] to embed shape differences or deltas into the latent space of VAE, enabling multiple kinds of edits with both geometric and structural modifications. Representing 3D man-made models as a set of handles, Gadelha *et al.* [283] adopted a two-branch network architecture to generate shape handles. After training the network, users can edit any handle of the handle set, and the back propagation is used to optimize the latent vector to obtain a result that preserves the overall structure.

Reinforcement learning can also be integrated in model editing. For example, Lin *et al.* [284] proposed a reinforcement learning based method to edit mesh models. First, the Prim-Agent predicts a sequence of actions to operate the primitives to approximate the target shape given a shape reference and pre-defined primitives. Then the edge loops are added to the output primitives. Second, the Mesh-Agent takes as input the shape reference and the primitive-based representation, and predicts actions to edit the meshes to produce shapes with detailed geometry.

Some methods input some guidance to deform the models. Kurenkov *et al.* [285] took an image as input and retrieve a template mesh from the repository, and then they deformed the template 3D shape to match the input image while preserving the topology of the template shape using Free-Form Deformation. Also retrieving from the given dataset at first, Uy *et al.* [286] deformed the retrieved set of source models to the target image or scan. The retrieval and deformation modules are trained jointly, in an alternating way. The deformation is part-based and structure-aware, predicted by a general MLP which takes encoded target, global and per-part source codes as inputs. Wang *et al.* [287] extracted global features from both the source shape and the target input or point cloud. These features are then inputted to an offset decoder which predicts per-vertex offsets to deform the source to produce a deformed shape similar to the target. Groueix *et al.* [288] also performed per-vertex deformation, leveraging not only reconstruction loss, but also cycle-consistency loss.

6 Conclusions

In this survey, we reviewed the history of 3D model editing and the exploration of deep learning based editing methods in recent years. We divided the editing methods into four subjects based on the data sources. In each subject, we further discussed the respective editing methods around organic shapes and man-made models. The former is generally manifold, and the latter is generally designed by an artist and is non-manifold. We showed the whole map with some representative works in Fig. 20.

For organic shapes or deformable models, we first discussed classical Laplacian-based methods, especially ARAP [48] and its large amounts of derivative works. In addition to these surface-based deformation methods, there are also deformation methods based on skeletons and cages. Editing methods learning from a dataset take into account the deformation principles of existing models in the dataset, and the deformation results will be more natural. The neural network based methods are also mainly divided into two parts for exploration. On the one hand, they consider the surface meshes and use various deformation representations as input, like traditional data-driven methods. On the other hand, they consider the skinning deformation based on the skeleton to provide intelligent solution strategies for skeleton rigging and weight assignment.

For man-made models, keeping the structure of the model from drastic changes, or maintaining the salient features of man-made models is most important. Therefore, editing methods of man-made models will maintain the invariance of local areas, and at the same time analyze the correlation between different parts of the model to limit the influence of editing. This invariance can be obtained by analyzing a single model or a large number of models in the dataset.

Neural editing is a promising direction. Although some works have explored neural editing methods for 3D models, there are still many directions that can be improved.

Organic shapes. At present, most of the editing methods of organic shapes based on neural networks

Fig.20. The structure of the survey and some representative works in each subjects.

still use traditional skeleton-based or cage-based skinning methods, while neural networks are used for skeleton binding [8], cage prediction [271], and weight assignment [256]. Although there are some methods [259] that explore the direct use of neural networks to predict the displacements of nonlinear deformation, experiments have only been carried out on isotropic materials, and the anisotropic materials need to redesign the framework. Therefore, on the one hand, we still need to design an end-to-end framework which inputs user constraints, such as editing handles and handle displacements, and outputs shape transformation matrix or vertex displacements; on the other hand, we need to study how to relate the selection of the deformation handle with the deformation result, such as optimizing the selection of control points, character rigging, and the prediction of weights according to the deformation results. In the latter, reinforcement learning may be a possible solution where possible control points are selected by the agent and rewards are given based on the deformation results.

Man-made models. The neural editing of man-made models still requires to satisfy both easy-to-manipulate deformation handles, and representations that can fully show the details of the model. The existing neural editing methods either use implicit surfaces [275] or manifold surfaces [269] to approximate the non-manifold model, which will lose part of the model details; or directly use FFD or cage-based editing methods on the original non-manifold model, but the handles are limited, such as semantic vector [235] and global deformation through cage [271]. A good form of handles can be edge loop [284] or coarse primitive-based representation [275]. But how to relate these handles with the non-manifold models still needs lots of work.

As a recently widely studied representation, implicit surfaces can achieve arbitrary resolution theoretically, which is a potential representation in various areas. In addition to further explorations in 3D model editing, neural morphing, i.e., morphing two shapes using neural networks, and neural modeling, i.e., modeling 3D models using neural networks, can also be regarded as

possible research directions.

Acknowledgement This work was supported by National Natural Science Foundation of China under Grant Nos. 62061136007 and 61872440, Royal Society Newton Advanced Fellowship under Grant No. NAF\R2\192151, Youth Innovation Promotion Association CAS, Science and Technology Service Network initiative, Chinese Academy of Sciences under Grant No. KFJ-STS-QYZD-2021-11-001.

References

[1] Gao L, Yang J, Wu T, Yuan Y J, Fu H, Lai Y K, Zhang H. SDM-NET: Deep generative network for structured deformable mesh. *ACM Trans. Graph.*, 2019, 38(6):1–15. DOI: 10.1145/3355089.3356488

[2] Yang J, Mo K, Lai Y K, Guibas L J, Gao L. DSM-Net: Disentangled structured mesh net for controllable generation of fine geometry. *arXiv preprint arXiv:2008.05440*, 2020.

[3] Chen Z, Zhang H. Learning implicit fields for generative shape modeling. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019, pp. 5939–5948. DOI: 10.1109/cvpr.2019.00609

[4] Xu Q, Wang W, Ceylan D, Mech R, Neumann U. DISN: Deep implicit surface network for high-quality single-view 3D reconstruction. In *Advances in Neural Information Processing Systems*, 2019, pp. 492–502. DOI: 10.1109/cvpr.2019.00113

[5] Gao L, Zhang L X, Meng H Y, Ren Y H, Lai Y K, Kobbelt L. PRS-Net: Planar reflective symmetry detection net for 3D models. *IEEE Transactions on Visualization and Computer Graphics*, 2020, pp. 1–1. DOI: 10.1109/tvcg.2020.3003823

[6] Shi Y, Huang J, Zhang H, Xu X, Rusinkiewicz S, Xu K. SymmetryNet: Learning to predict reflectional and rotational symmetries of 3D shapes from single-view RGB-D images. *ACM Trans. Graph.*, 2020, 39(6). DOI: 10.1145/3414685.3417775

[7] Fu R, Yang J, Sun J, Zhang F L, Lai Y K, Gao L. RISA-Net: Rotation-invariant structure-aware network for fine-grained 3D shape retrieval, 2020.

[8] Xu Z, Zhou Y, Kalogerakis E, Landreth C, Singh K. RigNet: Neural rigging for articulated characters. *ACM Trans. Graph.*, 2020, 39(4). DOI: 10.1145/3386569.3392379

[9] Chang A X, Funkhouser T, Guibas L, Hanrahan P, Huang Q, Li Z, Savarese S, Savva M, Song S, Su H, Xiao J, Yi L, Yu F. ShapeNet: An Information-Rich 3D Model Repository. Technical Report arXiv:1512.03012, 2015.

[10] Mitra N J, Wand M, Zhang H, Cohen-Or D, Kim V, Huang Q X. Structure-aware shape processing. In *ACM SIGGRAPH 2014 Courses*, pp. 1–21. 2014.

[11] Chaudhuri S, Ritchie D, Wu J, Xu K, Zhang H. Learning generative models of 3D structures. *Computer Graphics Forum*, 2020, 39(2):643–666. DOI: 10.1111/cgf.14020

[12] Han X, Laga H, Bennamoun M. Image-based 3D object reconstruction: State-of-the-art and trends in the deep learning era. *IEEE transactions on pattern analysis and machine intelligence*, 2019, 43(5):1578–1604. DOI: 10.1109/tpami.2019.2954885

[13] Jin Y, Jiang D, Cai M. 3D reconstruction using deep learning: a survey. *Communications in Information and Systems*, 2020, 20(4):389–413. DOI: 10.4310/cis.2020.v20.n4.a1

[14] Xiao Y P, Lai Y K, Zhang F L, Li C, Gao L. A survey on deep geometry learning: From a representation perspective. *Computational Visual Media*, 2020, 6(2):113–133. DOI: 10.1007/s41095-020-0174-8

[15] Botsch M, Sorkine O. On linear variational surface deformation methods. *IEEE Transactions on Visualization and Computer Graphics*, 2008, 14(1):213–230. DOI: 10.1145/3414685.3417775

[16] Gain J, Bechmann D. A survey of spatial deformation from a user-centered perspective. *ACM Trans. Graph.*, 2008, 27(4):1–21. DOI: 10.1145/1409625.1409629

[17] Xu K, Kim V G, Huang Q, Mitra N, Kalogerakis E. Data-driven shape analysis and processing. In *SIGGRAPH ASIA 2016 Courses*, pp. 1–38. 2016.

[18] Terzopoulos D, Platt J, Barr A, Fleischer K. Elastically deformable models. In *Proceedings of the 14th annual conference on Computer graphics and interactive techniques*, 1987, pp. 205–214. DOI: 10.1145/37401.37427

[19] Celniker G, Gossard D. Deformable curve and surface finite-elements for free-form shape design. In *Proceedings of the 18th annual conference on Computer graphics and interactive techniques*, 1991, pp. 257–266. DOI: 10.1145/122718.122746

[20] Welch W, Witkin A. Variational surface modeling. *SIGGRAPH Comput. Graph.*, 1992, 26(2):157–166. DOI: 10.1145/142920.134033

[21] Botsch M, Bommes D, Kobbelt L. Efficient linear system solvers for mesh processing. In *Mathematics of Surfaces XI*, pp. 62–83. Springer, 2005. DOI: 10.1007/11537908_5

[22] Shi L, Yu Y, Bell N, Feng W W. A fast multigrid algorithm for mesh deformation. *ACM Trans. Graph.*, 2006, 25(3):1108–1117. DOI: 10.1145/1141911.1142001

[23] Zorin D, Schröder P, Sweldens W. Interactive multiresolution mesh editing. In *Proceedings of the 24th annual conference on Computer graphics and interactive techniques*, 1997, pp. 259–268. DOI: 10.1145/258734.258863

[24] Kobbelt L, Campagna S, Vorsatz J, Seidel H P. Interactive multi-resolution modeling on arbitrary meshes. In *Proceedings of the 25th annual conference on Computer graphics and interactive techniques*, 1998, pp. 105–114. DOI: 10.1145/280814.280831

[25] Guskov I, Sweldens W, Schröder P. Multiresolution signal processing for meshes. In *Proceedings of the 26th annual conference on Computer graphics and interactive techniques*, 1999, pp. 325–334. DOI: 10.1145/311535.311577

[26] Botsch M, Sumner R, Pauly M, Gross M. Deformation transfer for detail-preserving surface editing. In *Vision, Modeling & Visualization*, 2006, pp. 357–364.

[27] Botsch M, Kobbelt L. An intuitive framework for real-time freeform modeling. *ACM Trans. Graph.*, 2004, 23(3):630–634. DOI: 10.1145/1015706.1015772

[28] liu H T D, Jacobson A. Cubic stylization. *ACM Trans. Graph.*, November 2019, 38(6). DOI: 10.1145/3355089.3356495

[29] Sorkine O. Laplacian mesh processing. *Eurographics (STARs)*, 2005, 29.

[30] Sorkine O. Differential representations for mesh processing. *Computer Graphics Forum*, 2006, 25(4):789–807. DOI: 10.1111/j.1467-8659.2006.00999.x

[31] Alexa M. Differential coordinates for local mesh morphing and deformation. *The Visual Computer*, 2003, 19(2-3):105–114. DOI: 10.1007/s00371-002-0180-0

[32] Sederberg T W, Parry S R. Free-form deformation of solid geometric models. *SIGGRAPH Comput. Graph.*, 1986, 20(4):151–160. DOI: 10.1145/15886.15903

[33] Lipman Y, Sorkine O, Cohen-Or D, Levin D, Rossi C, Seidel H P. Differential coordinates for interactive mesh editing. In *Proceedings Shape Modeling Applications, 2004.*, 2004, pp. 181–190. DOI: 10.1109/smi.2004.1314505

[34] Sorkine O, Cohen-Or D, Lipman Y, Alexa M, Rössl C, Seidel H P. Laplacian surface editing. In *Proceedings of the 2004 Eurographics/ACM SIGGRAPH symposium on Geometry processing*, 2004, pp. 175–184. DOI: 10.1145/1057432.1057456

[35] Lipman Y, Sorkine O, Alexa M, Cohen-Or D, Levin D, Rössl C, Seidel H P. Laplacian framework for interactive mesh editing. *International Journal of Shape Modeling*, 2005, 11(01):43–61. DOI: 10.1142/s0218654305000724

[36] Yu Y, Zhou K, Xu D, Shi X, Bao H, Guo B, Shum H Y. Mesh editing with Poisson-based gradient field manipulation. In *ACM SIGGRAPH 2004 Papers*, pp. 644–651. 2004.

[37] Zayer R, Rössl C, Karni Z, Seidel H P. Harmonic guidance for surface deformation. *Computer Graphics Forum*, 2005, 24(3):601–609. DOI: 10.1111/j.1467-8659.2005.00885.x

[38] Botsch M, Pauly M, Gross M H, Kobbelt L. PriMo: coupled prisms for intuitive surface modeling. In *Symposium on Geometry Processing*, number CONF, 2006, pp. 11–20.

[39] Zhou K, Huang J, Snyder J, Liu X, Bao H, Guo B, Shum H Y. Large mesh deformation using the volumetric graph Laplacian. In *ACM SIGGRAPH 2005 papers*, pp. 496–503. 2005.

[40] Sheffer A, Kraevoy V. Pyramid coordinates for morphing and deformation. In *Proceedings. 2nd International Symposium on 3D Data Processing, Visualization and Transmission, 2004. 3DPVT 2004.*, 2004, pp. 68–75. DOI: 10.1109/tdpvt.2004.1335149

[41] Au O C, Tai C L, Liu L, Fu H. Dual Laplacian editing for meshes. *IEEE Transactions on Visualization and Computer Graphics*, 2006, 12(3):386–395.

[42] Fu H, Kin-Chung Au O, Tai C L. Effective derivation of similarity transformations for implicit Laplacian mesh editing. *Computer Graphics Forum*, 2007, 26(1):34–45. DOI: 10.1111/j.1467-8659.2007.00940.x

[43] Huang J, Shi X, Liu X, Zhou K, Wei L Y, Teng S H, Bao H, Guo B, Shum H Y. Subspace gradient domain mesh deformation. In *ACM SIGGRAPH 2006 Papers*, pp. 1126–1134. 2006.

[44] Ju T, Schaefer S, Warren J. Mean value coordinates for closed triangular meshes. In *ACM SIGGRAPH 2005 Papers*, pp. 561–566. 2005.

[45] Vallet B, Lévy B. Spectral geometry processing with manifold harmonics. *Computer Graphics Forum*, 2008, 27(2):251–260. DOI: 10.1111/j.1467-8659.2008.01122.x

[46] Alexa M, Cohen-Or D, Levin D. As-rigid-as-possible shape interpolation. In *Proceedings of the 27th annual conference on Computer graphics and interactive techniques*, 2000, pp. 157–164. DOI: 10.1145/344779.344859

[47] Igarashi T, Moscovich T, Hughes J F. As-rigid-as-possible shape manipulation. *ACM Trans. Graph.*, 2005, 24(3):1134–1141. DOI: 10.1145/1073204.1073323

[48] Sorkine O, Alexa M. As-rigid-as-possible surface modeling. In *Symposium on Geometry processing*, volume 4, 2007, pp. 109–116.

[49] Chao I, Pinkall U, Sanan P, Schröder P. A simple geometric model for elastic deformations. *ACM Trans. Graph.*, 2010, 29(4):1–6. DOI: 10.1145/1778765.1778775

[50] Levi Z, Gotsman C. Smooth rotation enhanced as-rigid-as-possible mesh animation. *IEEE Transactions on Visualization and Computer Graphics*, 2014, 21(2):264–277. DOI: 10.1109/tvcg.2014.2359463

[51] Cuno A, Esperança C, Oliveira A, Cavalcanti P R. 3D as-rigid-as-possible deformations using MLS. In *Proceedings of the 27th computer graphics international conference*, 2007, pp. 115–122.

[52] Liu Y S, Yan H B, Martin R R. As-rigid-as-possible surface morphing. *Journal of computer science and technology*, 2011, 26(3):548–557. DOI: 10.1007/s11390-011-1154-3

[53] Chen S Y, Gao L, Lai Y K, Xia S. Rigidity controllable as-rigid-as-possible shape deformation. *Graphical Models*, 2017, 91:13–21. DOI: 10.1016/j.gmod.2017.02.005

[54] Qin X, Wu T, Liu Y. A surface deformation method based on stiffness control. *Journal of Advanced Mechanical Design, Systems, and Manufacturing*, 2020, 14(1):JAMDSM0010–JAMDSM0010. DOI: 10.1299/jamdsm.2020jamdsm0010

[55] Le Vaou Y, Léon J C, Hahmann S, Masfrand S, Mika M. As-stiff-as-needed surface deformation combining ARAP energy with an anisotropic material. *Computer-Aided Design*, 2020, 121:102803. DOI: 10.1016/j.cad.2019.102803

[56] Colaianni M, Siegl C, Süßmuth J, Bauer F, Greiner G. Anisotropic surface based deformation. In *SIGGRAPH ASIA 2016 Technical Briefs*, pp. 1–4. 2016.

[57] Colaianni M, Siegl C, Süßmuth J, Bauer F, Greiner G. Anisotropic deformation for local shape control. *Computational Visual Media*, 2017, 3(4):305–313. DOI: 10.1007/s41095-017-0092-6

[58] Borosán P, Howard R, Zhang S, Nealen A. Hybrid mesh editing. In *Eurographics (short papers)*, 2010, pp. 41–44.

[59] Sun Q, Wan W, Feng X, Chen G, Rizwan M, Sánchez J A. Bi-harmonic surface based as-rigid-as-possible mesh deformation. *Journal of Computers*, 2018, 29(4):161–175.

[60] Zollhöfer M, Sert E, Greiner G, Süßmuth J. GPU based ARAP deformation using volumetric lattices. In *Eurographics (Short Papers)*, 2012, pp. 85–88.

[61] Manson J, Schaefer S. Hierarchical deformation of locally rigid meshes. *Computer Graphics Forum*, 2011, 30(8):2387–2396. DOI: 10.1111/j.1467-8659.2011.02074.x

[62] Kovalsky S Z, Galun M, Lipman Y. Accelerated quadratic proxy for geometric optimization. *ACM Trans. Graph.*, 2016, 35(4):1–11. DOI: 10.1145/2897824.2925920

[63] Peng Y, Deng B, Zhang J, Geng F, Qin W, Liu L. Anderson acceleration for geometry optimization and physics simulation. *ACM Trans. Graph.*, 2018, 37(4):1–14. DOI: 10.1145/3197517.3201290

[64] Rabinovich M, Poranne R, Panozzo D, Sorkine-Hornung O. Scalable locally injective mappings. *ACM Trans. Graph.*, 2017, 36(4):1. DOI: 10.1145/3072959.3126782

[65] Shtengel A, Poranne R, Sorkine-Hornung O, Kovalsky S Z, Lipman Y. Geometric optimization via composite majorization. *ACM Trans. Graph.*, 2017, 36(4):38–1. DOI: 10.1145/3072959.3073618

[66] Zhu Y, Bridson R, Kaufman D M. Blended cured quasi-Newton for distortion optimization. *ACM Trans. Graph.*, 2018, 37(4):1–14. DOI: 10.1145/3197517.3201359

[67] Zhang S, Nealen A, Metaxas D N. Skeleton based as-rigid-as-possible volume modeling. In *Eurographics (Short Papers)*, 2010, pp. 21–24.

[68] Zhang S, Huang J, Metaxas D N. Robust mesh editing using Laplacian coordinates. *Graphical Models*, 2011, 73(1):10–19. DOI: 10.1016/j.gmod.2010.10.003

[69] Jacobson A, Baran I, Kavan L, Popović J, Sorkine O. Fast automatic skinning transformations. *ACM Trans. Graph.*, 2012, 31(4):1–10. DOI: 10.1145/2185520.2185573

[70] Yang J, Gao L, Lai Y K, Rosin P L, Xia S. Biharmonic deformation transfer with automatic key point selection. *Graphical Models*, 2018, 98:1–13. DOI: 10.1016/j.gmod.2018.05.003

[71] Gao L, Lai Y K, Yang J, Ling-Xiao Z, Xia S, Kobbelt L. Sparse data driven mesh deformation. *IEEE Transactions on Visualization and Computer Graphics*, 2019, 27(3):2085–2100. DOI: 10.1109/tvcg.2019.2941200

[72] Liu L, Zhang L, Xu Y, Gotsman C, Gortler S J. A local/global approach to mesh parameterization. *Computer Graphics Forum*, 2008, 27(5):1495–1504. DOI: 10.1111/j.1467-8659.2008.01290.x

[73] Gao L, Lai Y K, Huang Q X, Hu S M. A data-driven approach to realistic shape morphing. *Computer graphics forum*, 2013, 32(2pt4):449–457. DOI: 10.1111/cgf.12065

[74] Bouaziz S, Deuss M, Schwartzburg Y, Weise T, Pauly M. Shape-up: Shaping discrete geometry with projections. *Computer Graphics Forum*, 2012, 31(5):1657–1667. DOI: 10.1111/j.1467-8659.2012.03171.x

[75] Huang Q X, Wicke M, Adams B, Guibas L. Shape decomposition using modal analysis. *Computer Graphics Forum*, 2009, 28(2):407–416.

[76] Liu T, Bargteil A W, O'Brien J F, Kavan L. Fast simulation of mass-spring systems. *ACM Trans. Graph.*, 2013, 32(6):1–7. DOI: 10.1145/2508363.2508406

[77] Levi Z, Gotsman C. D-snake: image registration by as-similar-as-possible template deformation. *IEEE Transactions on Visualization and*

Computer Graphics, 2012, 19(2):331–343. DOI: 10.1109/tvcg.2012.134

[78] Sỳkora D, Dingliana J, Collins S. As-rigid-as-possible image registration for hand-drawn cartoon animations. In *Proceedings of the 7th International Symposium on Non-Photorealistic Animation and Rendering*, 2009, pp. 25–33. DOI: 10 .1145/1572614.1572619

[79] Fadaifard H, Wolberg G. Image warping for retargeting garments among arbitrary poses. *The Visual Computer*, 2013, 29(6-8):525–534. DOI: 10 .1007/s00371-013-0816-2

[80] Wang Y S, Liu F, Hsu P S, Lee T Y. Spatially and temporally optimized video stabilization. *IEEE Transactions on Visualization and Computer Graphics*, 2013, 19(8):1354–1361. DOI: 10.1109/tvcg.2013.11

[81] Crane K, Pinkall U, Schröder P. Spin transformations of discrete surfaces. In *ACM SIGGRAPH 2011 papers*, pp. 1–10. 2011.

[82] Fang Q, Zhao Z Y, Liu Z Y, Liu L, Fu X M. Metric first reconstruction for interactive curvature-aware modeling. *Computer-Aided Design*, 2020, p. 102863. DOI: 10.1016/j.cad.2020.102863

[83] Lai Y K, Zhou Q Y, Hu S M, Wallner J, Pottmann H. Robust feature classification and editing. *IEEE Transactions on Visualization and Computer Graphics*, 2006, 13(1):34–45. DOI: 10.1109/ tvcg.2007.19

[84] Lipman Y, Cohen-Or D, Gal R, Levin D. Volume and shape preservation via moving frame manipulation. *ACM Trans. Graph.*, 2007, 26(1):5–es. DOI: 10.1145/1189762.1189767

[85] Alhashim I, Zhang H, Liu L. Detail-replicating shape stretching. *The Visual Computer*, 2012, 28(12):1153–1166. DOI: 10.1007/s00371 -011-0665-9

[86] Au O K C, Tai C L, Chu H K, Cohen-Or D, Lee T Y. Skeleton extraction by mesh contraction. *ACM Trans. Graph.*, 2008, 27(3):1–10. DOI: 10. 1145/1360612.1360643

[87] Liu Z, Wang W m, Liu X p, Liu L g. Scale-aware shape manipulation. *Journal of Zhejiang University SCIENCE C*, 2014, 15(9):764–775. DOI: 10. 1631/jzus.c1400122

[88] Xu L, Wang R, Zhang J, Yang Z, Deng J, Chen F, Liu L. Survey on sparsity in geometric modeling and processing. *Graphical Models*, 2015, 82:160–180. DOI: 10.1016/j.gmod.2015.06.012

[89] Gao L, Zhang G, Lai Y. l_p shape deformation. *Science China Information Sciences*, 2012, 55(5):983–993. DOI: 10.1007/s11432-012-4574-y

[90] Deng B, Bouaziz S, Deuss M, Zhang J, Schwartzburg Y, Pauly M. Exploring local modifications for constrained meshes. *Computer Graphics Forum*, 2013, 32(2pt1):11–20. DOI: 10. 1111/cgf.12021

[91] Museth K, Breen D E, Whitaker R T, Barr A H. Level set surface editing operators. In *Proceedings of the 29th annual conference on Computer graphics and interactive techniques*, 2002, pp. 330–338. DOI: 10.1145/566570.566585

[92] Eyiyurekli M, Breen D E. Detail-preserving level set surface editing and geometric texture transfer. *Graphical Models*, 2017, 93:39–52. DOI: 10. 1016/j.gmod.2017.08.002

[93] Coquillart S. Extended free-form deformation: a sculpturing tool for 3D geometric modeling. In *Proceedings of the 17th annual conference on Computer graphics and interactive techniques*, 1990, pp. 187–196. DOI: 10.1145/97879.97900

[94] Kraevoy V, Sheffer A, Shamir A, Cohen-Or D. Non-homogeneous resizing of complex models. *ACM Trans. Graph.*, 2008, 27(5):1–9. DOI: 10.1145/1409060.1409064

[95] Xu W, Wang J, Yin K, Zhou K, Van De Panne M, Chen F, Guo B. Joint-aware manipulation of deformable models. *ACM Trans. Graph.*, 2009, 28(3):1–9. DOI: 10.1145/1531326.1531341

[96] Gal R, Sorkine O, Mitra N J, Cohen-Or D. iWIRES: an analyze-and-edit approach to shape manipulation. In *ACM SIGGRAPH 2009 papers*, pp. 1–10. 2009.

[97] Li G, Liu L, Zheng H, Mitra N J. Analysis, reconstruction and manipulation using arterial snakes. *ACM Trans. Graph.*, 2010, 29(6):152. DOI: 10.1145/1882261.1866178

[98] Zheng Y, Fu H, Cohen-Or D, Au O K C, Tai C L. Component-wise controllers for structure-preserving shape manipulation. *Computer Graphics Forum*, 2011, 30(2):563–572. DOI: 10.1111/j.1467-8659.2011.01880.x

[99] Xu K, Zheng H, Zhang H, Cohen-Or D, Liu L, Xiong Y. Photo-inspired model-driven 3D object modeling. *ACM Trans. Graph.*, 2011, 30(4):1–10. DOI: 10.1145/2010324.1964975

[100] Zhang C, Yang L, Xu L, Wang G, Wang W. Real-time editing of man-made mesh models under geometric constraints. *Computers & Graphics*, 2019, 82:174–182. DOI: 10.1016/j.cag.2019.05.028

[101] Bokeloh M, Wand M, Koltun V, Seidel H P. Pattern-aware shape deformation using sliding dockers. In *Proceedings of the 2011 SIGGRAPH Asia Conference*, 2011, pp. 1–10. DOI: 10.1145/2024156.2024157

[102] Bokeloh M, Wand M, Seidel H P, Koltun V. An algebraic model for parameterized shape editing. *ACM Trans. Graph.*, 2012, 31(4):1–10. DOI: 10.1145/2185520.2185574

[103] Lin J, Cohen-Or D, Zhang H, Liang C, Sharf A, Deussen O, Chen B. Structure-preserving retargeting of irregular 3D architecture. *ACM Trans. Graph.*, 2011, 30(6):1–10.

[104] Milliez A, Wand M, Cani M P, Seidel H P. Mutable elastic models for sculpting structured shapes. *Computer Graphics Forum*, 2013, 32(2pt1):21–30. DOI: 10.1111/cgf.12022

[105] Habbecke M, Kobbelt L. Linear analysis of nonlinear constraints for interactive geometric modeling. *Computer Graphics Forum*, 2012, 31(2pt3):641–650. DOI: 10.1111/j.1467-8659.2012.03043.x

[106] Cabral M, Lefebvre S, Dachsbacher C, Drettakis G. Structure-preserving reshape for textured architectural scenes. *Computer Graphics Forum*, 2009, 28(2):469–480. DOI: 10.1111/j.1467-8659.2009.01386.x

[107] Tagliasacchi A, Delame T, Spagnuolo M, Amenta N, Telea A. 3D skeletons: A state-of-the-art report. *Computer Graphics Forum*, 2016, 35(2):573–597. DOI: 10.1111/cgf.12865

[108] Magnenat-Thalmann N, Laperrire R, Thalmann D. Joint-dependent local deformations for hand animation and object grasping. In *In Proceedings on Graphics interface'88*, 1988.

[109] Bang S, Lee S H. Spline interface for intuitive skinning weight editing. *ACM Trans. Graph.*, 2018, 37(5):1–14. DOI: 10.1145/3186565

[110] Baran I, Popović J. Automatic rigging and animation of 3D characters. *ACM Trans. Graph.*, 2007, 26(3):72–es. DOI: 10.1145/1276377 .1276467

[111] Wareham R, Lasenby J. Bone glow: An improved method for the assignment of weights for mesh deformation. In *International Conference on Articulated Motion and Deformable Objects*, 2008, pp. 63–71.

[112] Jacobson A, Baran I, Popovic J, Sorkine O. Bounded biharmonic weights for real-time deformation. *ACM Trans. Graph.*, 2011, 30(4):78–1. DOI: 10.1145/2578850

[113] Jacobson A, Weinkauf T, Sorkine O. Smooth shape-aware functions with controlled extrema. *Computer Graphics Forum*, 2012, 31(5):1577–1586. DOI: 10.1111/j.1467-8659.2012.03163.x

[114] Yuan Y J, Lai Y K, Wu T, Xia S, Gao L. Data-driven weight optimization for real-time mesh deformation. *Graphical Models*, 2019, 104:101037. DOI: 10.1016/j.gmod.2019.101037

[115] Dionne O, Lasa M. Geodesic voxel binding for production character meshes. In *Proceedings of the 12th ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, 2013, pp. 173–180. DOI: 10.1145/2485895.2485919

[116] Dionne O, De Lasa M. Geodesic binding for degenerate character geometry using sparse voxelization. *IEEE Transactions on Visualization and Computer Graphics*, 2014, 20(10):1367–1378. DOI: 10.1109/tvcg.2014.2321563

[117] Yan H B, Hu S M, Martin R. Skeleton-based shape deformation using simplex transformations. In *Computer Graphics International Conference*, 2006, pp. 66–77.

[118] Yan H B, Hu S, Martin R R, Yang Y L. Shape deformation using a skeleton to drive simplex transformations. *IEEE Transactions on Visualization and Computer Graphics*, 2008, 14(3):693–706. DOI: 10.1109/tvcg.2008.28

[119] Lewis J P, Cordner M, Fong N. Pose space deformation: a unified approach to shape interpolation and skeleton-driven deformation. In *Proceedings of the 27th annual conference on Computer graphics and interactive techniques*, 2000, pp. 165–172. DOI: 10.1145/344779.344862

[120] Mohr A, Gleicher M. Building efficient, accurate character skins from examples. *ACM Trans. Graph.*, 2003, 22(3):562–568. DOI: 10. 1145/882262.882308

[121] Wang R Y, Pulli K, Popović J. Real-time enveloping with rotational regression. In *ACM SIGGRAPH 2007 papers*, pp. 73–es. 2007.

[122] Chen C H, Lin I C, Tsai M H, Lu P H. Lattice-based skinning and deformation for real-time skeleton-driven animation. In *2011 12th International Conference on Computer-Aided Design and Computer Graphics*, 2011, pp. 306–312. DOI: 10.1109/cad/graphics.2011.41

[123] Abu Rumman N, Fratarcangeli M. Position-based skinning for soft articulated characters. *Computer Graphics Forum*, 2015, 34(6):240–250. DOI: 10. 1111/cgf.12533

[124] Weber O, Sorkine O, Lipman Y, Gotsman C. Context-aware skeletal shape deformation. *Com-*

puter Graphics Forum, 2007, 26(3):265–274. DOI: 10.1111/j.1467-8659.2007.01048.x

[125] Shi X, Zhou K, Tong Y, Desbrun M, Bao H, Guo B. Example-based dynamic skinning in real time. *ACM Trans. Graph.*, 2008, 27(3):1–8. DOI: 10.1145/1360612.1360628

[126] Kavan L, Collins S, Žára J, O'Sullivan C. Skinning with dual quaternions. In *Proceedings of the 2007 symposium on Interactive 3D graphics and games*, 2007, pp. 39–46. DOI: 10.1145/1230100.1230107

[127] Kavan L, Collins S, Žára J, O'Sullivan C. Geometric skinning with approximate dual quaternion blending. *ACM Trans. Graph.*, 2008, 27(4):1–23. DOI: 10.1145/1409625.1409627

[128] Kavan L, Collins S, O'Sullivan C. Automatic linearization of nonlinear skinning. In *Proceedings of the 2009 symposium on Interactive 3D graphics and games*, 2009, pp. 49–56. DOI: 10.1145/1507149.1507157

[129] Alexa M. Linear combination of transformations. *ACM Trans. Graph.*, 2002, 21(3):380–387. DOI: 10.1145/566654.566592

[130] Magnenat-Thalmann N, Cordier F, Seo H, Papagianakis G. Modeling of bodies and clothes for virtual environments. In *2004 International Conference on Cyberworlds*, 2004, pp. 201–208. DOI: 10.1109/cw.2004.47

[131] Kavan L, Žára J. Spherical blend skinning: a real-time deformation of articulated models. In *Proceedings of the 2005 symposium on Interactive 3D graphics and games*, 2005, pp. 9–16. DOI: 10.1145/1053427.1053429

[132] Le B H, Hodgins J K. Real-time skeletal skinning with optimized centers of rotation. *ACM Trans. Graph.*, 2016, 35(4):1–10. DOI: 10.1145/2897824.2925959

[133] Kim Y, Han J. Bulging-free dual quaternion skinning. *Computer Animation and Virtual Worlds*, 2014, 25(3-4):321–329. DOI: 10.1002/cav.1604

[134] Forstmann S, Ohya J. Fast skeletal animation by skinned arc-spline based deformation. In *Eurographics (Short Presentations)*, 2006, pp. 1–4.

[135] Forstmann S, Ohya J, Krohn-Grimberghe A, McDougall R. Deformation styles for spline-based skeletal animation. In *Proceedings of the 2007 ACM SIGGRAPH/Eurographics symposium on Computer animation*, 2007, pp. 141–150.

[136] Yang X, Somasekharan A, Zhang J J. Curve skeleton skinning for human and creature characters. *Computer Animation and Virtual Worlds*, 2006, 17(3-4):281–292. DOI: 10.1002/cav.132

[137] Öztireli A C, Baran I, Popa T, Dalstein B, Sumner R W, Gross M. Differential blending for expressive sketch-based posing. In *Proceedings of the 12th ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, 2013, pp. 155–164. DOI: 10.1145/2485895.2485916

[138] Wang X C, Phillips C. Multi-weight enveloping: least-squares approximation techniques for skin animation. In *Proceedings of the 2002 ACM SIGGRAPH/Eurographics symposium on Computer animation*, 2002, pp. 129–138. DOI: 10.1145/545261.545283

[139] Merry B, Marais P, Gain J. Animation space: A truly linear framework for character animation. *ACM Trans. Graph.*, 2006, 25(4):1400–1423. DOI: 10.1145/1183287.1183294

[140] Jacobson A, Sorkine O. Stretchable and twistable bones for skeletal shape deformation. In *Proceedings of the 2011 SIGGRAPH Asia Conference*, 2011, pp. 1–8. DOI: 10.1145/2024156.2024199

[141] Thiery J M, Eisemann E. ARAPLBS: Robust and efficient elasticity-based optimization of weights and skeleton joints for linear blend skinning with parametrized bones. *Computer Graphics Forum*, 2018, 37(1):32–44. DOI: 10.1111/cgf.13161

[142] Li J, Lu G. Skeleton driven animation based on implicit skinning. *Computers & Graphics*, 2011, 35(5):945–954. DOI: 10.1016/j.cag.2011.07.005

[143] Kavan L, Sorkine O. Elasticity-inspired deformers for character articulation. *ACM Trans. Graph.*, 2012, 31(6):1–8. DOI: 10.1145/2366145.2366215

[144] McAdams A, Zhu Y, Selle A, Empey M, Tamstorf R, Teran J, Sifakis E. Efficient elasticity for character skinning with contact and collisions. In *ACM SIGGRAPH 2011 papers*, pp. 1–12. 2011.

[145] Vaillant R, Barthe L, Guennebaud G, Cani M P, Rohmer D, Wyvill B, Gourmel O, Paulin M. Implicit skinning: real-time skin deformation with contact modeling. *ACM Trans. Graph.*, 2013, 32(4):1–12. DOI: 10.1145/2461912.2461960

[146] Wendland H. *Scattered data approximation*, volume 17. Cambridge university press, 2004. DOI: 10.1017/cbo9780511617539

[147] Macedo I, Gois J P, Velho L. Hermite radial basis functions implicits. *Computer Graphics Forum*, 2011, 30(1):27–42. DOI: 10.1111/j.1467-8659.2010.01785.x

[148] Vaillant R, Guennebaud G, Barthe L, Wyvill B, Cani M P. Robust iso-surface tracking for interactive character skinning. *ACM Trans. Graph.*, 2014, 33(6):1–11. DOI: 10.1145/2661229.2661264

[149] Teng Y, Otaduy M A, Kim T. Simulating articulated subspace self-contact. *ACM Trans. Graph.*, 2014, 33(4):1–9. DOI: 10.1145/2601097.2601181

[150] James D L, Twigg C D. Skinning mesh animations. *ACM Trans. Graph.*, 2005, 24(3):399–407. DOI: 10.1145/1073204.1073206

[151] Yoshizawa S, Belyaev A, Seidel H P. Skeleton-based variational mesh deformations. *Computer Graphics Forum*, 2007, 26(3):255–264. DOI: 10.1111/j.1467-8659.2007.01047.x

[152] Xie Y, Xu W, Yang Y, Guo X, Zhou K. Agile structural analysis for fabrication-aware shape editing. *Computer Aided Geometric Design*, 2015, 35:163–179. DOI: 10.1016/j.cagd.2015.03.019

[153] Xu W, Yang H, Yang Y, Wang Y, Zhou K. Stress-aware large-scale mesh editing using a domain-decomposed multigrid solver. *Computer Aided Geometric Design*, 2018, 63:17–30. DOI: 10.1016/j.cagd.2018.03.013

[154] Jiang W, Xu K, Cheng Z Q, Martin R R, Dang G. Curve skeleton extraction by coupled graph contraction and surface clustering. *Graphical Models*, 2013, 75(3):137–148. DOI: 10.1016/j.gmod.2012.10.005

[155] Wang S, Wu J, Wei M, Ma X. Robust curve skeleton extraction for vascular structures. *Graphical Models*, 2012, 74(4):109–120. DOI: 10.1016/j.gmod.2012.03.008

[156] Tagliasacchi A, Alhashim I, Olson M, Zhang H. Mean curvature skeletons. *Computer Graphics Forum*, 2012, 31(5):1735–1744. DOI: 10.1111/j.1467-8659.2012.03178.x

[157] Tagliasacchi A, Zhang H, Cohen-Or D. Curve skeleton extraction from incomplete point cloud. In *ACM SIGGRAPH 2009 papers*, pp. 1–9. 2009.

[158] Qin H, Han J, Li N, Huang H, Chen B. Mass-driven topology-aware curve skeleton extraction from incomplete point clouds. *IEEE Transactions on Visualization and Computer Graphics*, 2019, 26(9):2805–2817. DOI: 10.1109/tvcg.2019.2903805

[159] Livny Y, Yan F, Olson M, Chen B, Zhang H, El-Sana J. Automatic reconstruction of tree skeletal structures from point clouds. In *ACM SIGGRAPH Asia 2010 papers*, pp. 1–8. 2010.

[160] Huang H, Wu S, Cohen-Or D, Gong M, Zhang H, Li G, Chen B. L1-medial skeleton of point cloud. *ACM Trans. Graph.*, 2013, 32(4):65–1. DOI: 10.1145/2461912.2461913

[161] Feng A, Casas D, Shapiro A. Avatar reshaping and automatic rigging using a deformable model. In *Proceedings of the 8th ACM SIGGRAPH Conference on Motion in Games*, 2015, pp. 57–64. DOI: 10.1145/2822013.2822017

[162] Anguelov D, Srinivasan P, Koller D, Thrun S, Rodgers J, Davis J. SCAPE: shape completion and animation of people. *ACM Trans. Graph.*, 2005, 24(3):408–416. DOI: 10.1145/1073204.1073207

[163] Bharaj G, Thormählen T, Seidel H P, Theobalt C. Automatically rigging multi-component characters. *Computer Graphics Forum*, 2012, 31(2pt4):755–764. DOI: 10.1111/j.1467-8659.2012.03034.x

[164] Schaefer S, Yuksel C. Example-based skeleton extraction. In *Symposium on Geometry Processing*, 2007, pp. 153–162.

[165] De Aguiar E, Theobalt C, Thrun S, Seidel H P. Automatic conversion of mesh animations into skeleton-based animations. *Computer Graphics Forum*, 2008, 27(2):389–397. DOI: 10.1111/j.1467-8659.2008.01136.x

[166] Hasler N, Thormählen T, Rosenhahn B, Seidel H P. Learning skeletons for shape and pose. In *Proceedings of the 2010 ACM SIGGRAPH symposium on Interactive 3D Graphics and Games*, 2010, pp. 23–30. DOI: 10.1145/1730804.1730809

[167] Le B H, Deng Z. Robust and accurate skeletal rigging from mesh sequences. *ACM Trans. Graph.*, 2014, 33(4):1–10. DOI: 10.1145/2601097.2601161

[168] Le B H, Deng Z. Smooth skinning decomposition with rigid bones. *ACM Trans. Graph.*, 2012, 31(6):1–10. DOI: 10.1145/2366145.2366218

[169] Cohen J, Varshney A, Manocha D, Turk G, Weber H, Agarwal P, Brooks F, Wright W. Simplification envelopes. In *Proceedings of the 23rd Annual Conference on Computer Graphics and Interactive Techniques*, 1996, p. 119–128. DOI: 10.1145/237170.237220

[170] Ben-Chen M, Weber O, Gotsman C. Spatial deformation transfer. In *Proceedings of the 2009 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, SCA '09, 2009, p. 67–74. DOI: 10.1145/1599470.1599479

[171] Deng Z J, Luo X N, Miao X P. Automatic cage building with quadric error metrics. *J. Comput. Sci. Technol.*, 05 2011, 26(3):538–547. DOI: 10.1007/s11390-011-1153-4

[172] Sacht L, Vouga E, Jacobson A. Nested cages. *ACM Trans. Graph.*, October 2015, 34(6):1–14. DOI: 10.1145/2816795.2818093

[173] Chuhua Xian, Hongwei Lin, Shuming Gao. Automatic generation of coarse bounding cages from dense meshes. In *2009 IEEE International Conference on Shape Modeling and Applications*, 2009, pp. 21–27. DOI: 10.1109/smi.2009.5170159

[174] Xian C, Li G, Xiong Y. Efficient and effective cage generation by region decomposition. *Comput. Animat. Virtual Worlds*, March 2015, 26(2):173–184. DOI: 10.1002/cav.1571

[175] Le B H, Deng Z. Interactive cage generation for mesh deformation. In *Proceedings of the 21st ACM SIGGRAPH Symposium on Interactive 3D Graphics and Games*, 2017. DOI: 10.1145/3023368.3023369

[176] Ju T, Zhou Q Y, Panne M, Cohen-Or D, Neumann U. Reusable skinning templates using cage-based deformations. *ACM Trans. Graph.*, 2008, 27(5):1–10. DOI: 10.1145/1409060.1409075

[177] Savoye Y, Franco J S. CageIK: dual-laplacian cage-based inverse kinematics. In *International Conference on Articulated Motion and Deformable Objects*, 2010, pp. 280–289.

[178] Floater M S. Mean value coordinates. *Computer aided geometric design*, 2003, 20(1):19–27. DOI: 10.1016/s0167-8396(03)00002-5

[179] Floater M S, Kós G, Reimers M. Mean value coordinates in 3D. *Computer Aided Geometric Design*, 2005, 22(7):623–631. DOI: 10.1016/s0167-8396(03)00002-5

[180] Hormann K, Floater M S. Mean value coordinates for arbitrary planar polygons. *ACM Trans. Graph.*, 2006, 25(4):1424–1441. DOI: 10.1145/1183287.1183295

[181] Joshi P, Meyer M, DeRose T, Green B, Sanocki T. Harmonic coordinates for character articulation. *ACM Trans. Graph.*, 2007, 26(3):71–es. DOI: 10.1145/1276377.1276466

[182] Lipman Y, Kopf J, Cohen-Or D, Levin D. GPU-assisted positive mean value coordinates for mesh deformations. In *Symposium on geometry processing*, 2007.

[183] Langer T, Belyaev A, Seidel H P. Spherical barycentric coordinates. In *Symposium on Geometry Processing*, 2006, pp. 81–88.

[184] Ju T, Schaefer S, Warren J D, Desbrun M. A geometric construction of coordinates for convex polyhedra using polar duals. In *Symposium on Geometry Processing*, 2005, pp. 181–186. DOI: 10.1145/1186822.1073229

[185] Lipman Y, Levin D, Cohen-Or D. Green coordinates. *ACM Trans. Graph.*, 2008, 27(3):1–10. DOI: 10.1145/1360612.1360677

[186] Weber O, Ben-Chen M, Gotsman C et al. Complex barycentric coordinates with applications to planar shape deformation. In *Computer Graphics Forum*, volume 28, 2009, p. 587.

[187] Ben-Chen M, Weber O, Gotsman C. Variational harmonic maps for space deformation. *ACM Trans. Graph.*, 2009, 28(3):1–11. DOI: 10.1145/1531326.1531340

[188] Yang W, Feng J, Jin X. Shape deformation with tunable stiffness. *The Visual Computer*, 2008, 24(7-9):495. DOI: 10.1007/s00371-008-0230-3

[189] Manson J, Schaefer S. Moving least squares coordinates. *Computer Graphics Forum*, 2010, 29(5):1517–1524. DOI: 10.1111/j.1467-8659.2010.01760.x

[190] Dyken C, Floater M S. Transfinite mean value interpolation. *Computer Aided Geometric Design*, 2009, 26(1):117–134. DOI: 10.1016/j.cagd.2007.12.003

[191] Weber O, Poranne R, Gotsman C. Biharmonic coordinates. *Computer Graphics Forum*, 2012, 31(8):2409–2422. DOI: 10.1111/j.1467-8659.2012.03130.x

[192] Li X Y, Ju T, Hu S M. Cubic mean value coordinates. *ACM Trans. Graph.*, 2013, 32(4):126–1. DOI: 10.1145/2461912.2461917

[193] Landreneau E, Schaefer S. Poisson-based weight reduction of animated meshes. *Computer Graphics Forum*, 2010, 29(6):1945–1954. DOI: 10.1111/j.1467-8659.2010.01661.x

[194] Le B H, Deng Z. Two-layer sparse compression of dense-weight blend skinning. *ACM Trans. Graph.*, 2013, 32(4):1–10. DOI: 10.1145/2461912.2461949

[195] Zhang J, Deng B, Liu Z, Patanè G, Bouaziz S, Hormann K, Liu L. Local barycentric coordinates. *ACM Trans. Graph.*, 2014, 33(6):1–12. DOI: 10.1145/2661229.2661255

[196] Chambolle A, Caselles V, Cremers D, Novaga M, Pock T. An introduction to total variation for image analysis. *Theoretical foundations and numerical methods for sparse recovery*, 2010, 9(263-340):227.

[197] Tao J, Deng B, Zhang J. A fast numerical solver for local barycentric coordinates. *Computer Aided Geometric Design*, 2019, 70:46–58. DOI: 10.1016/j.cagd.2019.04.006

[198] Botsch M, Pauly M, Wicke M, Gross M. Adaptive space deformations based on rigid cells. *Computer Graphics Forum*, 2007, 26(3):339–347. DOI: 10.1111/j.1467-8659.2007.01056.x

[199] Li Z, Levin D, Deng Z, Liu D, Luo X. Cage-free local deformations using green coordinates. *The Visual Computer*, 2010, 26(6-8):1027–1036. DOI: 10.1007/s00371-010-0438-x

[200] Levi Z, Levin D. Shape deformation via interior RBF. *IEEE Transactions on Visualization and Computer Graphics*, 2013, 20(7):1062–1075. DOI: 10.1109/tvcg.2013.255

[201] García F G, Paradinas T, Coll N, Patow G. * cages: a multilevel, multi-cage-based system for mesh deformation. *ACM Trans. Graph.*, 2013, 32(3):1–13. DOI: 10.1145/2487228.2487232

[202] Thiery J M, Guy E, Boubekeur T. Sphere-meshes: Shape approximation using spherical quadric error metrics. *ACM Trans. Graph.*, 2013, 32(6):1–12. DOI: 10.1145/2508363.2508384

[203] Thiery J M, Guy E, Boubekeur T, Eisemann E. Animated mesh approximation with sphere-meshes. *ACM Trans. Graph.*, May 2016, 35(3):30:1–30:13. DOI: 10.1145/2898350

[204] Zhang Y, Zheng J, Cai Y. Proxy-driven free-form deformation by topology-adjustable control lattice. *Computers & Graphics*, 2020, 89:167–177. DOI: 10.1016/j.cag.2020.05.013

[205] Bogo F, Romero J, Loper M, Black M J. FAUST: Dataset and evaluation for 3D mesh registration. In *Proceedings IEEE Conference on Computer Vision and Pattern Recognition*, 2014. DOI: 10.1109/cvpr.2014.491

[206] Sumner R W, Zwicker M, Gotsman C, Popović J. Mesh-based inverse kinematics. *ACM*

Trans. Graph., 2005, 24(3):488–495. DOI: 10.1145/1073204.1073218

[207] Sumner R W, Popović J. Deformation transfer for triangle meshes. *ACM Trans. Graph.*, 2004, 23(3):399–405. DOI: 10.1145/1015706.1015736

[208] Murray R M, Li Z, Sastry S S, Sastry S S. *A mathematical introduction to robotic manipulation*. CRC press, 1994. DOI: 10.1201/9781315136370

[209] Der K G, Sumner R W, Popović J. Inverse kinematics for reduced deformable models. *ACM Trans. Graph.*, 2006, 25(3):1174–1179. DOI: 10.1145/1141911.1142011

[210] Wampler K. Fast and reliable example-based mesh IK for stylized deformations. *ACM Trans. Graph.*, November 2016, 35(6):1–12. DOI: 10.1145/2980179.2982433

[211] Lipman Y, Sorkine O, Levin D, Cohen-Or D. Linear rotation-invariant coordinates for meshes. *ACM Trans. Graph.*, 2005, 24(3):479–487. DOI: 10.1145/1073204.1073217

[212] Baran I, Vlasic D, Grinspun E, Popović J. Semantic deformation transfer. In *ACM SIGGRAPH 2009 papers*, pp. 1–6. 2009.

[213] Gao L, Chen S Y, Lai Y K, Xia S. Data-driven shape interpolation and morphing editing. *Computer Graphics Forum*, 2017, 36(8):19–31. DOI: 10.1111/cgf.12991

[214] Kircher S, Garland M. Free-form motion processing. *ACM Trans. Graph.*, 2008, 27(2):1–13. DOI: 10.1145/1356682.1356685

[215] Winkler T, Drieseberg J, Alexa M, Hormann K. Multi-scale geometry interpolation. *Computer graphics forum*, 2010, 29(2):309–318. DOI: 10.1111/j.1467-8659.2009.01600.x

[216] Fröhlich S, Botsch M. Example-driven deformations based on discrete shells. *Computer graphics forum*, 2011, 30(8):2246–2257. DOI: 10.1111/j.1467-8659.2011.01974.x

[217] Gao L, Lai Y K, Liang D, Chen S Y, Xia S. Efficient and flexible deformation representation for data-driven surface modeling. *ACM Trans. Graph.*, 2016, 35(5):158:1–158:17. DOI: 10.1145/2908736

[218] Xu W, Zhou K, Yu Y, Tan Q, Peng Q, Guo B. Gradient domain editing of deforming mesh sequences. *ACM Trans. Graph.*, 2007, 26(3):84–es. DOI: 10.1145/1276377.1276482

[219] Sumner R W, Schmid J, Pauly M. Embedded deformation for shape manipulation. In *ACM SIGGRAPH 2007 papers*, pp. 80–es. 2007.

[220] Alexa M, Müller W. Representing animations by principal components. *Computer Graphics Forum*, 2000, 19(3):411–418. DOI: 10.1111/1467-8659.00433

[221] Neumann T, Varanasi K, Wenger S, Wacker M, Magnor M, Theobalt C. Sparse localized deformation components. *ACM Trans. Graph.*, 2013, 32(6):1–10. DOI: 10.1145/2508363.2508417

[222] Huang Z, Yao J, Zhong Z, Liu Y, Guo X. Sparse localized decomposition of deformation gradients. *Computer Graphics Forum*, 2014, 33(7):239–248. DOI: 10.1111/cgf.12492

[223] Bernard F, Gemmar P, Hertel F, Goncalves J, Thunberg J. Linear shape deformation models with local support using graph-based structured matrix factorisation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 5629–5638. DOI: 10.1109/cvpr.2016.607

[224] Wang Y, Li G, Zeng Z, He H. Articulated-motion-aware sparse localized decomposition. *Computer Graphics Forum*, 2017, 36(8):247–259. DOI: 10.1111/cgf.13076

[225] Liu Y, Li G, Wang Y, Nie Y, Mao A. Discrete shell deformation driven by adaptive sparse localized components. *Computers & Graphics*, 2019, 78:76–86. DOI: 10.1016/j.cag.2018.11.005

[226] Wang Y, Liu B, Tong Y. Linear surface reconstruction from discrete fundamental forms on triangle meshes. *Computer Graphics Forum*, 2012, 31(8):2277–2287. DOI: 10.1111/j.1467-8659.2012.03153.x

[227] Sassen J, Heeren B, Hildebrandt K, Rumpf M. Geometric optimization using nonlinear rotation-invariant coordinates. *Computer Aided Geometric Design*, 2020, p. 101829. DOI: 10.1016/j.cagd.2020.101829

[228] Sassen J, Hildebrandt K, Rumpf M. Nonlinear deformation synthesis via sparse principal geodesic analysis. *Computer Graphics Forum*, 2020, 39(5):119–132. DOI: 10.1111/cgf.14073

[229] Heeren B, Zhang C, Rumpf M, Smith W. Principal geodesic analysis in the space of discrete shells. *Computer Graphics Forum*, 2018, 37(5):173–184. DOI: 10.1111/cgf.13500

[230] Loper M, Mahmood N, Romero J, Pons-Moll G, Black M J. SMPL: A skinned multi-person linear model. *ACM Trans. Graph.*, 2015, 34(6):1–16. DOI: 10.1145/2816795.2818013

[231] Pons-Moll G, Romero J, Mahmood N, Black M J. Dyna: A model of dynamic human shape in motion. *ACM Trans. Graph.*, 2015, 34(4):1–14. DOI: 10.1145/2766993

[232] Fish N, Averkiou M, Van Kaick O, Sorkine-Hornung O, Cohen-Or D, Mitra N J. Meta-representation of shape families. *ACM Trans. Graph.*, 2014, 33(4):1–11. DOI: 10.1145/2601097.2601185

[233] Yumer M E, Kara L B. Co-constrained handles for deformation in shape collections. *ACM Trans. Graph.*, 2014, 33(6):1–11. DOI: 10.1145/2661229.2661234

[234] Yumer M E, Kara L B. Co-abstraction of shape collections. *ACM Trans. Graph.*, 2012, 31(6):1–11. DOI: 10.1145/2366145.2366185

[235] Yumer M E, Chaudhuri S, Hodgins J K, Kara L B. Semantic shape editing using deformation handles. *ACM Trans. Graph.*, 2015, 34(4):1–12. DOI: 10.1145/2766908

[236] Fu Q, Chen X, Su X, Li J, Fu H. Structure-adaptive shape editing for man-made objects. *Computer Graphics Forum*, 2016, 35(2):27–36. DOI: 10.1111/cgf.12808

[237] Laga H, Tabia H. Modeling and exploring co-variations in the geometry and configuration of man-made 3D shape families. *Computer Graphics Forum*, 2017, 36(5):13–25. DOI: 10.1111/cgf.13241

[238] Zheng Y, Liu H, Dorsey J, Mitra N J. Ergonomics-inspired reshaping and exploration of collections of models. *IEEE Transactions on Visualization and Computer Graphics*, 2015, 22(6):1732–1744. DOI: 10.1109/tvcg.2015.2448084

[239] Ovsjanikov M, Li W, Guibas L, Mitra N J. Exploration of continuous variability in collections of 3D shapes. *ACM Trans. Graph.*, 2011, 30(4):1–10.

[240] Ishimtsev V, Bokhovkin A, Artemov A, Ignatyev S, Niessner M, Zorin D, Burnaev E. CAD-Deform: Deformable fitting of CAD models to 3D scans. *arXiv preprint arXiv:2007.11965*, 2020.

[241] Mahmood N, Ghorbani N, Troje N F, Pons-Moll G, Black M J. AMASS: Archive of motion capture as surface shapes. In *The IEEE International Conference on Computer Vision (ICCV)*, October 2019, pp. 5442–5451. DOI: 10.1109/iccv.2019.00554

[242] Tan Q, Gao L, Lai Y K, Xia S. Variational autoencoders for deforming 3D mesh models. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018, pp. 5841–5850. DOI: 10.1109/cvpr.2018.00612

[243] Defferrard M, Bresson X, Vandergheynst P. Convolutional neural networks on graphs with fast localized spectral filtering. In *Advances in neural information processing systems*, 2016, pp. 3844–3852.

[244] Yuan Y J, Lai Y K, Yang J, Duan Q, Fu H, Gao L. Mesh variational autoencoders with edge contraction pooling. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, 2020, pp. 274–275. DOI: 10.1109/cvprw50498.2020.00145

[245] Tan Q, Gao L, Lai Y K, Yang J, Xia S. Mesh-based autoencoders for localized deformation component analysis. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018. DOI: 10.1109/cvpr.2018.00612

[246] Yang J, Gao L, Tan Q, Huang Y, Xia S, Lai Y K. Multiscale mesh deformation component analysis with attention-based autoencoders, 2020.

[247] Qiao Y L, Gao L, Lai Y K, Xia S. Learning bidirectional LSTM networks for synthesizing 3D mesh animation sequences. *arXiv preprint arXiv:1810.02042*, 2018.

[248] Gao L, Yang J, Qiao Y L, Lai Y K, Rosin P L, Xu W, Xia S. Automatic unpaired shape deformation transfer. *ACM Trans. Graph.*, 2018, 37(6):1–15. DOI: 10.1145/3272127.3275028

[249] Zhu J Y, Park T, Isola P, Efros A A. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 2223–2232. DOI: 10.1109/iccv.2017.244

[250] Chen L, Zhang X, Ye J. Multi-feature super-resolution network for cloth wrinkle synthesis, 2020.

[251] Chen L, Gao L, Yang J, Xu S, Ye J, Zhang X, Lai Y K. Deep deformation detail synthesis for thin shell models, 2021.

[252] Wang J, Wen C, Fu Y, Lin H, Zou T, Xue X, Zhang Y. Neural pose transfer by spatially adaptive instance normalization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020, pp. 5831–5839. DOI: 10.1109/cvpr42600.2020.00587

[253] Bailey S W, Omens D, Dilorenzo P, O'Brien J F. Fast and deep facial deformations. *ACM Trans. Graph.*, 2020, 39(4):94–1. DOI: 10.1145/3386569.3392397

[254] Bailey S W, Otte D, Dilorenzo P, O'Brien J F. Fast and deep deformation approximations. *ACM Trans. Graph.*, 2018, 37(4):1–12. DOI: 10.1145/3197517.3201300

[255] Li T, Shi R, Kanai T. DenseGATs: A graph-attention-based network for nonlinear character deformation. In *Symposium on Interactive 3D Graphics and Games*, 2020, pp. 1–9.

[256] Liu L, Zheng Y, Tang D, Yuan Y, Fan C, Zhou K. NeuroSkinning: Automatic skin binding for production characters with deep graph networks. *ACM Trans. Graph.*, 2019, 38(4):to appear. DOI: 10.1145/3306346.3322969

[257] Xu Z, Zhou Y, Kalogerakis E, Singh K. Predicting animation skeletons for 3D articulated models via volumetric nets. In *2019 International Conference on 3D Vision (3DV)*, 2019. DOI: 10.1109/3dv.2019.00041

[258] Vesdapunt N, Rundle M, Wu H, Wang B. JNR: Joint-based neural rig representation for compact 3D face modeling. *arXiv preprint arXiv:2007.06755*, 2020. DOI: 10.1145/3384382.3384525

[259] Luo R, Shao T, Wang H, Xu W, Chen X, Zhou K, Yang Y. NNWarp: Neural network-based nonlinear deformation. *IEEE Transactions on Visualization and Computer Graphics*, 2018, pp. 1–1. DOI: 10.1109/tvcg.2018.2881451

[260] Fulton L, Modi V, Duvenaud D, Levin D I, Jacobson A. Latent-space dynamics for reduced deformable simulation. *Computer graphics forum*, 2019, 38(2):379–391. DOI: 10.1111/cgf.13645

[261] Santesteban I, Garces E, Otaduy M A, Casas D. SoftSMPL: Data-driven modeling of nonlinear soft-tissue dynamics for parametric humans. *Computer Graphics Forum*, 2020, 39(2):65–75. DOI: 10.1111/cgf.13912

[262] Deng B, Lewis J, Jeruzalski T, Pons-Moll G, Hinton G, Norouzi M, Tagliasacchi A. Neural articulated shape approximation. In *The European Conference on Computer Vision (ECCV)*, August 2020. DOI: 10.1109/cvpr42600.2020.00011

[263] Jeruzalski T, Levin D I, Jacobson A, Lalonde P, Norouzi M, Tagliasacchi A. NiLBS: Neural inverse linear blend skinning. *arXiv preprint arXiv:2004.05980*, 2020.

[264] Wu Z, Song S, Khosla A, Yu F, Zhang L, Tang X, Xiao J. 3D ShapeNets: A deep representation for volumetric shapes. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015. DOI: 10.1109/cvpr.2015.7298801

[265] Zhou Q, Jacobson A. Thingi10K: A dataset of 10,000 3d-printing models. *arXiv preprint arXiv:1605.04797*, 2016.

[266] Koch S, Matveev A, Jiang Z, Williams F, Artemov A, Burnaev E, Alexa M, Zorin D, Panozzo D. ABC: A big CAD model dataset for geometric deep learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019. DOI: 10.1109/cvpr.2019.00983

[267] Yumer M E, Mitra N J. Learning semantic deformation flows with 3D convolutional networks. In *The European Conference on Computer Vision (ECCV)*, 2016, pp. 294–311.

[268] Liu J, Yu F, Funkhouser T. Interactive 3D modeling with a generative adversarial network. In *2017 International Conference on 3D Vision (3DV)*, 2017, pp. 126–134. DOI: 10.1109/3dv.2017.00024

[269] Umetani N. Exploring generative 3D shapes using autoencoder networks. In *SIGGRAPH Asia 2017 Technical Briefs*, pp. 1–4. 2017.

[270] Mehr E, Jourdan A, Thome N, Cord M, Guitteny V. DiscoNet: Shapes learning on disconnected manifolds for 3D editing. In *Proceedings of the IEEE International Conference on Computer Vision*, 2019, pp. 3474–3483. DOI: 10.1109/iccv.2019.00357

[271] Yifan W, Aigerman N, Kim V G, Chaudhuri S, Sorkine-Hornung O. Neural cages for detail-preserving 3D deformations. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020, pp. 75–83. DOI: 10.1109/cvpr42600.2020.00015

[272] Liu M, Sung M, Mech R, Su H. DeepMetaHandles: Learning deformation meta-handles of 3D meshes with biharmonic coordinates. *arXiv preprint arXiv:2102.09105*, 2021.

[273] Wang Y, Jacobson A, Barbič J, Kavan L. Linear subspace design for real-time shape deformation. *ACM Trans. Graph.*, 2015, 34(4):1–11. DOI: 10.1145/2766952

[274] Liu S, Li T, Chen W, Li H. Soft rasterizer: A differentiable renderer for image-based 3D reasoning. In *The IEEE International Conference on Computer Vision (ICCV)*, Oct 2019. DOI: 10.1109/iccv.2019.00780

[275] Hao Z, Averbuch-Elor H, Snavely N, Belongie S. DualSDF: Semantic shape manipulation using a two-level representation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020, pp. 7631–7641. DOI: 10.1109/cvpr42600.2020.00765

[276] Zadeh A, Lim Y C, Liang P P, Morency L P. Variational auto-decoder. *arXiv preprint arXiv:1903.00840*, 2019.

[277] Deng Y, Yang J, Tong X. Deformed implicit field: Modeling 3D shapes with learned dense correspondence, 2020.

[278] Zheng Z, Yu T, Dai Q, Liu Y. Deep implicit templates for 3D shape representation, 2020.

[279] Wei F, Sizikova E, Sud A, Rusinkiewicz S, Funkhouser T. Learning to infer semantic parameters for 3D shape editing. In *2020 International Conference on 3D Vision (3DV)*, 2020, pp. 434–442. DOI: 10.1109/3dv50981.2020.00053

[280] Sung M, Jiang Z, Achlioptas P, Mitra N J, Guibas L J. DeformSyncNet: Deformation transfer via synchronized shape deformation spaces. *ACM Trans. Graph.*, 2020, 39(6):1–16. DOI: 10.1145/3414685.3417783

[281] Mo K, Guerrero P, Yi L, Su H, Wonka P, Mitra N J, Guibas L J. StructEdit: Learning structural shape variations. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020, pp. 8859–8868. DOI: 10.1109/cvpr42600.2020.00888

[282] Mo K, Guerrero P, Yi L, Su H, Wonka P, Mitra N, Guibas L J. StructureNet: Hierarchical graph networks for 3D shape generation. *ACM Trans. Graph.*, 2019, 38(6):1–19. DOI: 10.1145/3355089.3356527

[283] Gadelha M, Gori G, Ceylan D, Mech R, Carr N, Boubekeur T, Wang R, Maji S. Learning generative models of shape handles. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020, pp. 402–411. DOI: 10.1109/cvpr42600.2020.00048

[284] Lin C, Fan T, Wang W, Nießner M. Modeling 3d shapes by reinforcement learning. In *European Conference on Computer Vision*, 2020, pp. 545–561.

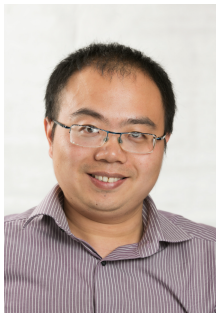
[285] Kurenkov A, Ji J, Garg A, Mehta V, Gwak J, Choy C B, Savarese S. DeformNet: Free-form deformation network for 3D shape reconstruction from a single image. In *2018 IEEE Winter Conference on Applications of Computer Vision*, 2018, pp. 858–866. DOI: 10.1109/wacv.2018.00099

[286] Uy M A, Kim V G, Sung M, Aigerman N, Chaudhuri S, Guibas L. Joint learning of 3D shape retrieval and deformation, 2021.

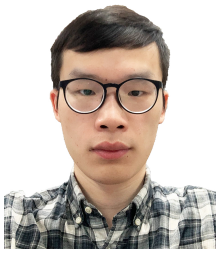
[287] Wang W, Ceylan D, Mech R, Neumann U. 3DN: 3D deformation network. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. DOI: 10.1109/cvpr.2019.00113

[288] Groueix T, Fisher M, Kim V G, Russell B C, Aubry M. Unsupervised cycle-consistent deformation for shape matching. *Computer Graphics Forum*, 2019, 38(5):123–133. DOI: 10.1111/cgf.13794

Yu-Jie Yuan received the bachelor's degree in mathematics from Xi'an Jiaotong University in 2018. He is currently a Ph.D. candidate in the Institute of Computing Technology, Chinese Academy of Sciences. His research interests include computer graphics and geometric processing.



Yu-Kun Lai received his bachelor's degree and PhD degree in computer science from Tsinghua University in 2003 and 2008, respectively. He is currently a Professor in the School of Computer Science & Informatics, Cardiff University. His research interests include computer graphics, geometry processing, image processing and computer vision. He is on the editorial boards of Computer Graphics Forum and The Visual Computer.



Tong Wu received the bachelor's degree in computer science from Huazhong University of Science and Technology in 2019. He is currently a Master student in the Institute of Computing Technology, Chinese Academy of Sciences. His research interests include computer graphics and 3D vision.

Lin Gao received the bachelor's degree in mathematics from Sichuan University and the PhD degree in computer science from Tsinghua University. He is currently an Associate Professor at the Institute of Computing Technology, Chinese Academy of Sciences. He has been awarded the Newton Advanced Fellowship from the Royal Society and the AG young researcher award. His research interests include computer graphics and geometric processing.

Ligang Liu is a professor at the University of Science and Technology of China. He received his PhD in mathematics from Zhejiang University in 2001. He once worked at Microsoft Research Asia, Zhejiang University, and visited Harvard University. His research interests include computer graphics and geometry processing. He serves as the associated editors for journals including IEEE TVCG, IEEE CG&A, CAGD, C&G, The Visual Computer, etc. He served as the conference co-chair of GMP 2017 and the program co-chairs of conferences including Chinagraph 2020, SIAM GD 2019, GMP 2018, CAD/Graphics 2017, CVM 2016, SGP 2015, and SPM 2014. He serves as the steering committee member of GMP and the secretary of Asiagraphics Association.