Template-based 3D Model Fitting Using Dual-domain Relaxation

I-Cheng Yeh* Chao-Hung Lin* Olga Sorkine† Tong-Yee Lee*
*National Cheng Kung University †New York University

***Abstract*—We introduce a template fitting method for 3D surface meshes. A given template mesh is deformed to closely approximate the input 3D geometry. The connectivity of the deformed template model is automatically adjusted to facilitate the geometric fitting and to ascertain high quality of the mesh elements. The template fitting process utilizes a specially tailored Laplacian processing framework, where in the first, coarse fitting stage we approximate the input geometry with a linearized biharmonic surface (a variant of LS-mesh [36]), and then the fine geometric detail is fitted further using iterative Laplacian editing with reliable correspondence constraints and a local surface flattening mechanism to avoid foldovers. The latter step is performed in the dual mesh domain, which is shown to encourage near-equilateral mesh elements and significantly reduces the occurrence of triangle fold-overs, a well known problem in mesh fitting. To experimentally evaluate our approach, we compare our method with relevant state-of-the-art techniques and confirm significant improvements of results. In addition, we demonstrate the usefulness of our approach to the application of consistent surface parameterization (also known as cross-parameterization).**

***Index Terms*—Template-based fitting, dual mesh, local surface flattening, consistent parameterization, Laplacian coordinates,cross-parameterization,inter-surface mapping**

I. Introduction

TEmplate fitting has been used in many geometry processing applications, such as surface reconstruction from point clouds and geometry completion [1]–[3], [8], [17], [30], [38], [41], surface cross-parameterization and correspondence [39], [43], remeshing and subdivision [4], [23]. Template meshes essentially provide a good “initial guess” for the surface fitting or reconstruction process since they have valid connectivity and are assumed to be of the appropriate topological type. In some cases, the geometry of the template mesh is known to resemble the target 3D data, which further facilitates the reconstruction process. Moreover, a template mesh can be conveniently used as the base domain for a sequence of shapes, for instance, when processing time series of moving geometry: having a single template adapted to multiple shapes allows to establish and maintain cross-correspondence between them.

The quality of the template-based fitting result depends on two factors: geometric approximation and mesh quality. Deforming an arbitrary mesh to fit irregularly sampled and possibly partially missing geometric data is clearly a difficult problem. In some settings, simplifying assumptions can be made, such as the template and the target geometry belonging to the same restricted class of shapes (e.g., humans) or having the same pose [17], [30], [38], [39]. In the absence of these assumptions it is challenging to establish dense correspondence between the template and the target shape; spurious and conflicting point correspondences may easily lead to self-intersections as the template deforms. In the general case, however, a suitable template mesh that fits the above criteria may not be available, and the goal is to reduce the restrictions on the fitting as much as possible. Mesh quality, i.e., proper sampling rate, the shape of the elements and valid manifoldness, is of equal importance since the template fitting process is typically only an intermediate stage followed by further processing. Most digital geometry processing applications perform better when acting on well-constructed meshes with near-quadrilateral elements [5] (for example, lower parametric distortion and better approximation quality is achieved in global parameterization and multiresolution analysis [14]).

In this paper, we propose an iterative template-based fitting scheme that is capable of deforming an arbitrary template mesh to approximate the given geometric data, which comes in the form of a polygonal mesh. In contrast to previous work, the major contribution of this paper is to design a specially tailored Laplacian processing framework for the template fitting process. Specifically, our method consists of two stages: coarse and fine fitting, as shown in Fig. 1. The coarse fitting step relies on several point correspondences between the template mesh and the target geometry and creates a rough approximation by converting the template model into a least-squares mesh [36], which is a linearized bi-harmonic surface. At this stage, we additionally design a WYSIWYG (What You See Is What You Get) kneading system to help the users interactively specify the correspondences between the template and target shapes in an intuitive manner. In the fine fitting stage, additional reliable correspondences are determined automatically and the template mesh is further iteratively adjusted to the target geometry. The fine fitting is performed using the dual domain mesh, which leads not only to high approximation quality but also encourages inner fairness of the mesh in the primal domain. While fitting the target, many methods such as [38], [43] automatically find extra correspondences based on the shortest Euclidean distance via normal projection or nearest neighbor search, and filter out unreliable correspondences via additional rules. However, these fine correspondences are still fragile and easily lead to many triangle foldovers, especially if the meshes have drastically different geometries. To better solve this problem, we propose a local surface flattening mechanism for unreliable correspondences after the above filtering step. This novel method is particularly useful for extremely concave or protruding regions where triangle foldovers are always generated. In addition, we allow subdivision of the template model to adjust the sampling rate and decrease the approximation error in cases where the template mesh is coarser than the target shape.

Thanks to the above design, our framework enables deforming 2-manifold templates with same genus, and less restrictions on their shape or pose. In practice, this allows establishing dense correspondence between different shapes, which is useful for such applications as consistent surface parameterization. Additionally, since the fine fitting step maintains well-formed mesh elements and reliable point correspondences, the occurrence of self-intersections and fold-overs is significantly reduced. We demonstrate the effectiveness of our method with experimental results on cross-parameterization and show that it compares favorably with other related works [39], [43].

II. Related Work

We classify the previous work about template-based fitting for 3D models into four categories: *surface completion*, *non-rigid shape registration*, *cross-parameterization* and *surface subdivision* based on which applications the fitting approaches are applied.

Surface completion: Many approaches have been proposed to fill the missing parts of the incomplete point cloud model by utilizing the geometry information of the template model [1]–[3], [8], [17], [30], [38], [41]. Theses approaches establish a mapping between the point and template models to appropriately fill the holes. Most of these methods assume that the template is very similar to the point model in terms of shape [1], [3], [17], [38] or even pose [2], [8], [30], [41]. Pauly et al. [30] present an example-based completion framework that treats surface completion as a case of mesh merging. Several templates are warped to align with the point model, and then blended together to fill the holes. Similarly, Kraevoy and Sheffer [17] select a template model to merge an incomplete target model. To establish the correspondence between the template and the target models, they embed them onto a base mesh. This approach is very robust even for an incomplete model containing large complex holes or multiple components. However, the completion results may sometimes be unnatural, even if the shapes of the input models are very similar, due to a large difference in geometry resolution between the input models. On the other hand, Stoll et al. [38] directly deform the template model to approximate the incomplete point model. However, the selected template model must be similar to the point model in shape and even geometry details. In addition, this approach requires users to align the local frames of the corresponding feature points, which might be a tedious and unintuitive process. Sharf et al. [35] present a coarse-to-fine deformable model for reconstructing surfaces from point clouds with arbitrary genus. This interesting work can automatically adjust mesh resolution for surface reconstruction. However, it is not suitable for establishing consistent correspondence between multiple targets.

Non-rigid registration: Automatic non-rigid registration is a recent popular research topic, especially for real-time range scan data. Without user intervention, most of the existing algorithms [7], [13], [20], [21], [40] depend on good initial match or similarity of the initial shape and the local details between source and target models. Chang et al. [7] propose an unsupervised algorithm for registering articulated objects without prior correspondence required. On modern hardware, their system still requires over one hour to obtain the final result. Similar to [7], Li et al. [21] design a novel registration system for scanned shapes based on the fact that both the source and target surfaces are already parameterized on the depth image, i.e., without extra cost for parameterization. Recently, Li et al. [20] further propose a template-based dynamic registration algorithm based on few assumptions on the geometry and motion of the scanned objects. Furthermore, the authors provide a detail synthesis method that employs a spatio-temporal analysis of detail vectors to propagate detail into occluded regions and remove high-frequency acquisition noise. However, these two methods [20], [21] require the input data to be sampled very densely in both time and space. Also, both of them are specified design for single view reconstruction. Huang et al. [13] present an efficient algorithm to align point clouds by enforcing preservation of geodesic distances between sets of corresponding points. This algorithm works well only when both source and target models are similar in geometric appearance. Similarly, Winter et al. [40] present a reconstruction system for continuous and time-varying data. By sliding the template mesh along the space-time surface in an as-rigid-as-possible manner, this algorithm can retrieve an identical mesh with a rigid motion for all time frames. However, it requires the input data to be sampled very densely in both time and space. Recently, Zhang et al. [42] developed a deformation-driven algorithm that automatically finds corresponding feature pairs between two models with large, non-rigid shape variations. The system obtains good correspondence for the extremities of the models, though at a significant computational cost. This approach may potentially fail to align detailed features and incorrectly align two models with different number of protrusions. Steinbrecher et al. [37] present an iterative and partial fitting algorithm for dental inlay construction. This system is particularly useful for spherical objects and uses significant domain knowledge for the specific dental application.

Cross-parameterization: In this category, the general approach is to establish the correspondence by parameterizing the closed input meshes on an intermediate common domain such as a sphere or another canonical base mesh domain [16], [17], [19], [32]–[34]. Sumner et al. [39] propose an alternative solution which deforms one input model to fit the other model and therefore establishes their correspondence. This method requires the source and target models to be similar in shape and pose. Zhang et al. [43] used least-squares meshes (LS-meshes) introduced by Sorkine and Cohen-Or [36] to compute cross-parameterization between models directly on the primal domain. However, in [43], both source and target meshes are required to be similar in shape and pose as well. We perform thorough experiments in Sec. V to compare our work with the above two most related techniques [39], [43].

Without careful treatment, poor mesh quality and mesh fold-overs can easily occur in cross-parameterization. Therefore, in the work of Kraevoy et al. [16], the authors guarantee each correspondent patch is in triangular shape to make sure there are no folds in parameterization. Also, Schreiner et al. [34] propose a vertex optimization to improve quality and solve this problem simultaneously. Our approach avoids this problem by encouraging equilateral triangles throughout the fitting process, and is capable of establishing cross-correspondence between very different shapes. We adapt a dual mesh Laplacian formulation to enhance mesh quality and eliminate artifacts in the iterative mesh-fitting process. Dual Laplacians have been introduced in recent works on mesh editing and deformation [12], [15], [25], where they are exploited for surface detail representation and preservation.

Surface subdivision: In this category, most of the approaches begin with an initial base mesh (or call control mesh) which is generated from the input point data [8], [11], [22], [26], [41]. These approaches utilize surface subdivision techniques such as Loop subdivision [24] and Catmull-Clark subdivision [6] to subdivide and fit the base mesh to approximate the point data. Generally, if a base model with narrow faces is used, a subdivision surface with a lot of narrow faces will be generated. In contrast, the model faces generated from the proposed method is near-equilateral even a template model with narrow faces is used. Recently, Martin et al. [26] present a method to generate a novel subdivision surface to approximate scattered data. Based on the procedure of unconditionally convergent parameter correction, this approach may iteratively optimize the control mesh with respect to L2 metric by adding control vertices. More recently, Li et al. [22] introduce a new surface reconstruction technique to automatically generate an anisotropy-adapted T-Spline surface to mimic the mesh that a designer would create manually. In the meantime, an interactive system can help the user to refine the T-Spline surface directly. In these above two parametric surface fitting techniques [22], [26], we can also regard the parameter domain of a parametric surface as its topology and regard the positions of its control points as its geometry. Therefore, both methods are related to our method in some sense, although our method is not built on parametric surface domain.

III. System Overview

Fig. 1 schematically illustrates the proposed template fitting scheme (see our accompanying video). Our fitting scheme consists of two major steps: (1) coarse fitting using an LS-mesh [36] and (2) fine fitting that employs iterative estimation of reliable point correspondences and operates in the dual mesh domain to improve mesh element quality. After the fitting procedure is completed, an optional subdivision step can be performed to improve the approximation quality in cases where the resolution of the template is initially lower than that of the target model.

For the coarse fitting step, we require a few point correspondences between the template and the target model in order to compute the LS-mesh that roughly approximates the target. These sparse point correspondences may be supplied by the user or automatically computed by one of the recent techniques (e.g., [7], [42], when the conditions permit that). Note that our method poses no restraints on the shape of the template mesh: it can be completely different from the target shape, provided that several correspondence pairs can be reasonably specified The subsequently constructed LS-mesh provides an initial guess to the automatic fine fitting procedure, as it passes close to the selected points on the target shape and constitutes a smooth (linearized bi-harmonic) surface.

When using LS-meshes [36], all the information about the template except for connectivity is discarded. The motivation behind this is avoiding transferring the geometric details of the input template to the target fitted model such as in the work of Stoll et al. [38]. A comparison example in Fig. 2 clearly shows the advantage. Furthermore, the proposed technique is specially tailored for the application of consistent surface parameterization, where multiple mesh models may be configured with very different geometric details and connectivity or even poses. In such scenario, it is not necessary to maintain the original geometric details of the template mesh; retaining the original connectivity of template mesh is useful, on the other hand, to build a compatible model for these input models. In addition, throwing away geometric information of the template mesh offers us more flexibility in the proposed fitting method to facilitate the adjustment of triangle shapes for better quality in the final fitting result. In other words, the fitted mesh quality is less dependent on the quality of the input template mesh in our case. For example, in Fig. 10, a bad-quality *Cat* mesh (i.e., with foldovers) used as a template, can still lead to a good result when fitting the *Bunny* model using the proposed method.

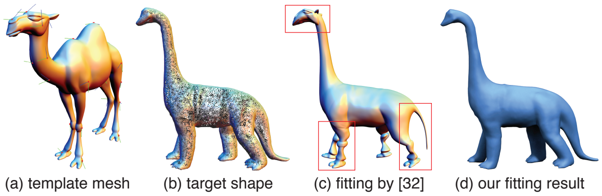

Fig. 2. Comparison of our method with Stoll et al. [38]. The images in (a), (b) and (c) are cited from [38]).

IV. Template-based Fitting

We begin by introducing the notation to be used in the remainder of the paper. We denote the template mesh by $\mathcal{S} = (\mathcal{V}, \mathcal{K})$, where $\mathcal{V} \subset \mathbb{R}^3$ is the set of vertex positions and $\mathcal{K}$ represents the connectivity. Similarly, $\mathcal{T}$ is the target model. We denote the individual vertex locations by $\mathbf{v}_i^s$, $\mathbf{v}_i^t$, where the superscripts s and t indicate template and target geometry, respectively. The total number of vertices and faces is denoted by n_v and n_f, respectively (again, with the appropriate superscripts).

A. Coarse Fitting

To obtain a rough approximation of the target shape, we compute a least-squares mesh $\mathcal{M}_\mathcal{C} = (\mathcal{V}_\mathcal{C}, \mathcal{K})$ from the template $\mathcal{S} = (\mathcal{V}, \mathcal{K})$ and the specified point correspondences. We essentially discard the geometry of the template model, only using its connectivity as the base domain for the bi-harmonic surface, such that a crude approximation of the pose of the target model is obtained, without the fine geometric

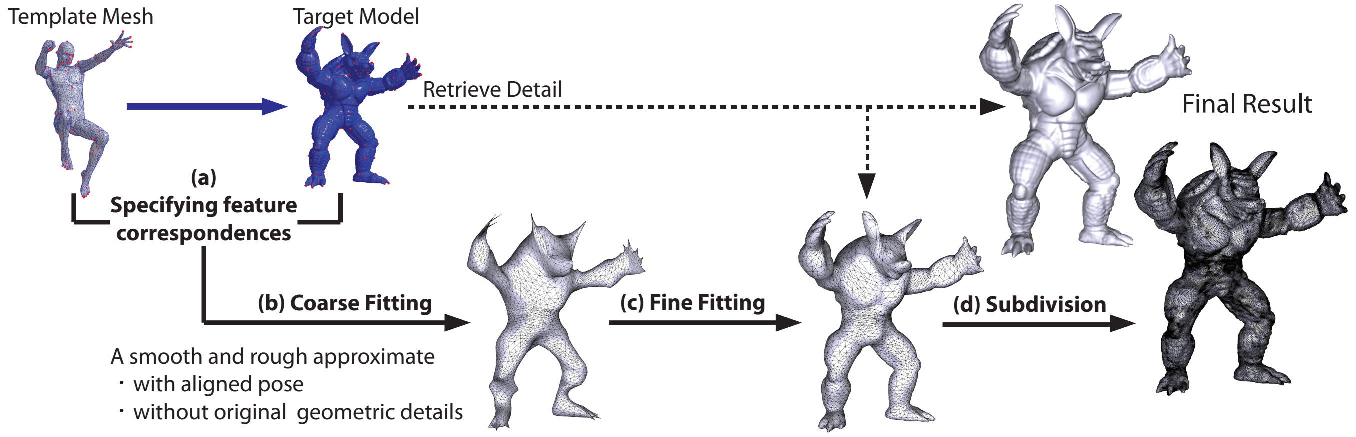

Fig. 1. System overview. (a) The input data with several user-specified feature correspondences (*Male* is the template mesh model and *Armadillo* is the target model); (b) coarse fitting step; (c) automatic iterative refinement adds geometric details to the coarse approximation (d) finally, executing mesh subdivision if necessary.

details. Without loss of generality, let us assume that the first k vertices of $\mathcal{S}$ are the ones marked as corresponding to the points $\{\mathbf{v}_1^t, \mathbf{v}_2^t, \ldots, \mathbf{v}_k^t\}$ on $\mathcal{T}$. The geometry of the LS-mesh is obtained by minimizing the bi-Laplacian energy:

$$\mathcal{V}_\mathcal{C} = \operatorname*{argmin}_{\mathbf{v}} \|L\mathbf{v}\|_F^2 \tag{1}$$

subject to constraints $\mathbf{v}_i = \mathbf{v}_i^t\,,\ i = 1, 2, \ldots, k$. Here, $\|\cdot\|_F$ is the Frobenius norm and L is the uniform $n_v^s \times n_v^s$ Laplacian matrix corresponding to the connectivity $\mathcal{K}$, i.e.,

$$(L)_{ij} = \begin{cases} -1/d_i, & \text{vertices } i \text{ and } j \text{ are neighbors} \\ 1, & i = j \\ 0, & \text{otherwise} \end{cases} \tag{2}$$

The d_i refers to the valence of vertex i. Choosing the uniform Laplacian over a geometric discretization (e.g., the cotan Laplacian [31]) results in a smooth mesh that also optimizes inner fairness: each vertex strives to lie in the centroid of its 1-ring neighbors and thus the edge lengths are strive to be locally equalized [27], [28].

The minimization in Eq. (1) amounts to solving the sparse bi-Laplacian equation

$$L^T L\mathbf{v} = 0, \quad \text{subject to positional constraints} \quad \mathbf{v}_i = \mathbf{v}_i^t\,,$$

which can be done efficiently using modern linear solvers (for details, see [36]).

From our practical experience, our guidance on feature specification is described as follows. First, we will initially place some corresponding landmark points on somewhere with clear semantic correspondence between template and target models such as legs and heads of animal models. These landmark points are used to coarsely approximate/align the target shape and pose. Subsequently, if the above coarse approximation is good enough, the user can stop specifying more landmark points. Otherwise, we will tend to place more landmark points on some locations with high-curvature and the largest distance between the template and target models. These landmark points are used to approximate the shape of target model and make the convergence of surface fitting more efficient. The user can continuously place landmark points until the result of coarse fitting is good enough, i.e., under some error threshold. Generally, the user should specify points with similar semantic features as the corresponding landmark points using above two simple rules. However, in some case such as sphere and head models, we can not find clear semantic feature correspondences between them. In our design, we provide a WYSIWYG kneading system. The proposed interactive system can help users freely add, remove and modify correspondences while manipulating the template fitting. Fig. 1(b) shows an example of the coarse fitting by using the *Male* connectivity and several feature pairs to obtain a rough approximation of the *Armadillo* model. Fig. 3(c) shows the kneading process of the template mesh in this example. For a better impression we recommend the reader to examine the supplementary video of this work.

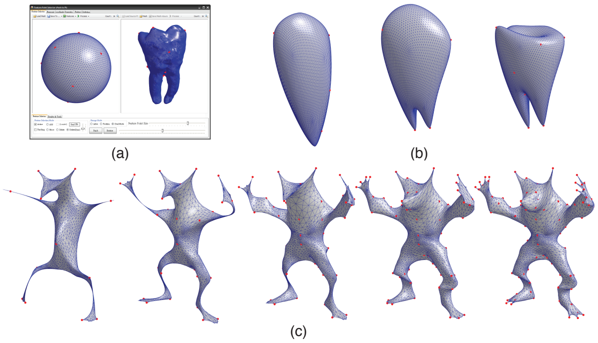

Fig. 3. A demonstration of the kneading process operated by the user: (a) a snapshot of the user interface in our kneading system; (b) and (c) show two examples of iterative kneading sequences for specifying correspondences.

Furthermore, our approach is generally not very sensitive to the accuracy of the correspondence. While allowing the user to roughly (not accurately) specify the feature pairs with similar semantics, our approach still can generate good/acceptable fitting results. For example, in Fig. 4, we used old man head as a template with two different sets of feature pairs to fit a Venus head model. Even though the feature positions in these two sets are not the same, the proposed method still can achieve similar fitting results.

In the fine fitting step described next, more corresponding feature pairs are automatically computed and the rough approximation $\mathcal{M}_{\mathcal{C}}$ is iteratively deformed to approximate $\mathcal{T}$ more closely (see Fig. 1(c)).

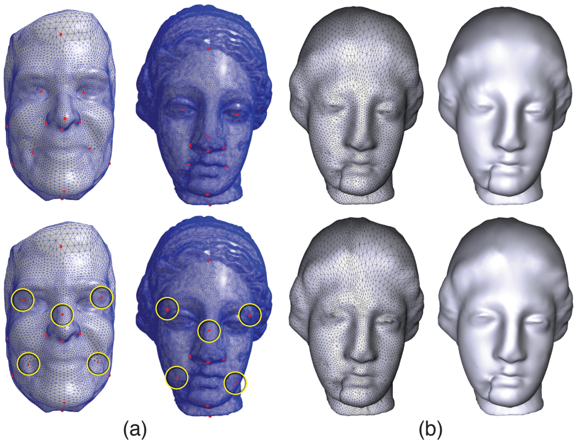

Fig. 4. An experiment shows the comparison of the results with two different sets of feature pairs (a) template and target model with feature points. The yellow circles highlight the different features. (b) the results of deformed templates

B. Fine Fitting

Let us denote the geometry of the mesh that will be iteratively deforming in this stage by $\mathcal{V}_{\mathcal{F}}$, such that the fine-fitted mesh is $\mathcal{M}_{\mathcal{F}} = (\mathcal{V}_{\mathcal{F}}, \mathcal{K})$. We initialize $\mathcal{M}_{\mathcal{F}}$ with the geometry of the coarsely-fitting LS-mesh computed in the first step, $\mathcal{V}_{\mathcal{F}} := \mathcal{V}_{\mathcal{C}}$. In the fine fitting stage, we iteratively compute reliable correspondence points between $\mathcal{M}_{\mathcal{F}}$ and $\mathcal{T}$ and update $\mathcal{V}_{\mathcal{F}}$ accordingly, essentially adding more and more point constraints to the LS-mesh computation. In addition, to ensure triangle shape regularity, we alternate between relaxation steps in the dual and the primal domains, as explained below.

1) Dual domain mesh relaxation: The dual mesh $\mathcal{M}^*_{\mathcal{F}} = (\mathcal{V}^*_{\mathcal{F}}, \mathcal{K}^*)$ has n^s_f vertices, where each dual vertex $\mathbf{v}^*_i$ corresponds to the face t^s_i in the primal domain mesh (see Fig. 5(a)). Two dual vertices are connected by an edge whenever the corresponding primal faces are neighbors (share an edge in the primal mesh). If the primal mesh is a closed manifold triangle mesh, the dual mesh will have only regular vertices of degree 3. Therefore, the dual mesh has an ideal regular connectivity, and constructing an LS-mesh based on this connectivity facilitates near equilateral elements [28]. Moreover, pinching and fold-over artifacts can be thus largely avoided or eliminated, as illustrated in Fig. 5(b-c). To take advantage of these properties, we alternate between template fitting in the dual domain and in the primal domain. The dual-domain fitting takes care of inner fairness and the primal domain enables better fitting of features (as described later in Sec. IV-B2). It is theoretically possible to achieve the fitting result by integrating the concept of dual Laplacian into primal domain and solving everything in one step. However, due to large constraint sets and complex weighting mechanisms (described latter), the error of this integrated system may potentially not be propagated smoothly in practice. Therefore, we solve the problem in two separate stages.

Determining reliable correspondences: Given the current fitted mesh $\mathcal{M}_{\mathcal{F}} = (\mathcal{V}_{\mathcal{F}}, K)$, we wish to compute reliable point correspondences between $\mathcal{M}_{\mathcal{F}}$ and the target shape $\mathcal{T}$ (as a reminder, our initial guess for $\mathcal{M}_{\mathcal{F}}$ is $\mathcal{M}_{\mathcal{C}}$). We project each vertex (or point) $\mathbf{p}$ of $\mathcal{T}$ onto $\mathcal{M}_{\mathcal{F}}$ along $\mathbf{p}$'s normal direction; this results in hitting a point on a triangle t^s_i on $\mathcal{M}_{\mathcal{F}}$. The correspondence pair $(\mathbf{p}, t^s_i)$ is *reliable* if both following conditions hold:

1) The normal of $\mathbf{p}$ is similar to that of the triangle t^s_i (the angle between the normals is less than $90°$).
2) The k-nearest neighbors of $\mathbf{p}$ project onto faces neighboring to t^s_i.

Next, we compute the geometry of the dual mesh $\mathcal{M}^*_{\mathcal{F}}$ using the reliable correspondences above. For each primal triangle t^s_i that received some reliable correspondence points in the projection procedure, we compute the centroid $\mathbf{c}^t_i$ of those points and set it as the positional constraint for $\mathbf{v}^*_i$ in the LS-mesh setup. We call such dual vertices "matched". In the current stage, not all primal triangles will have reliable correspondences (especially in concave regions, Fig. 6(a)), so that some dual vertices remain unmatched. The naive approach would be to leave the unmatched dual vertices as free variables and solve for an LS-mesh using only the matched vertices as constraints, just like we did earlier in the coarse fitting step (Eq. 1). However, this approach leads to artifacts and may cause the resulting mesh to self-intersect, because the unmatched areas tend to "shrink". Note that several previous methods, such as [38], [43], compute reliable correspondences in the primal domain by projecting or nearest neighbor searching, using similar conditional rules, and consequently suffer these artifacts as well.

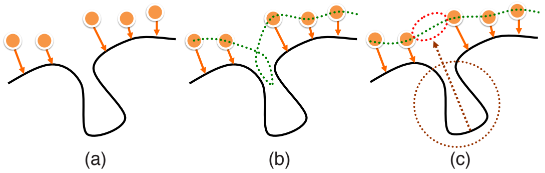

Fig. 6. (a) Vertex projection. The orange points represent the vertices of the target model, and the black curve represents template mesh. (b) The fitting of the template (the green dots) may cause fold-overs in concave regions. (c) Flattening the unmatched vertices (within the brown circle) into the regions near the corresponding positions of the nearby matched vertices (red ellipse).

To remove the above artifacts, we introduce a novel local surface flattening mechanism and we assign a rough corresponding position constraint for each unmatched dual vertex as well, by placing each unmatched vertex in the centroid of its neighbors. This results in a simple linear system of the form

$$L^* \mathbf{v}^* = 0,$$

with hard constraints posed on the positions of the matched dual vertices ($\mathbf{v}^*_i = \mathbf{c}^t_i$). L^* is the uniform Laplacian of the dual mesh; all the matched vertices are thus eliminated from the system and we solve for the position estimates of the unmatched ones. In other words, the unmatched vertices are

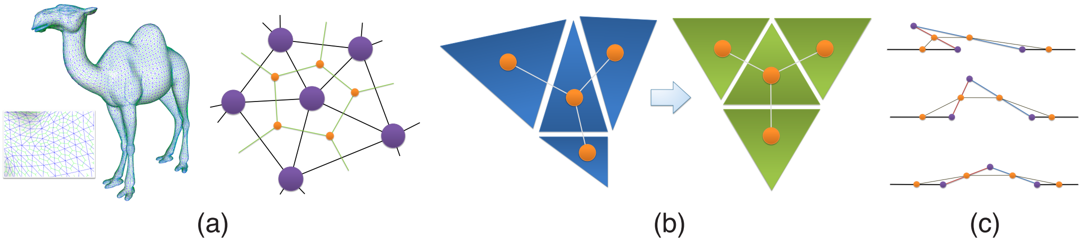

Fig. 5. Fitting in the dual domain: (a) a dual mesh with regular vertices of degree 3 (the orange dots); (b) regularization of the dual mesh of template leads to near equilateral faces; (c) fold-overs are avoided or relaxed by fitting in dual domain.

uniformly flattened in the regions that are surrounded by the nearby matched vertices, i.e., with reliable correspondences. In our experience, this process aids dual-domain relaxation to resolve artifacts which are generated in the fitting process, and it greatly improves the convergence speed as well. We demonstrate this local flattening technique to avoid triangle foldovers in Fig. 10.

Once the corresponding position of each vertex v_i^* in dual mesh is obtained, we use these corresponding positions as constraints in the LS-mesh setup to re-fit the dual mesh $\mathcal{M}_{\mathcal{F}}^*$:

$$\mathcal{V}_{\mathcal{F}}^* = \arg\min_{\mathbf{v}^*} \left(\omega \left\| L^* \mathbf{v}^* \right\|^2 + \sum_{i=1}^{n_f^s} w_i \|\mathbf{v}_i^* - \mathbf{c}_i^*\|^2 \right) , \qquad (3)$$

where the smoothness term is controlled by the weight ω (we used $\omega = 7.0$ for enforcing the smoothness term) and the individual vertex position constraints are controlled by the weights w_i.

The weight w_i of vertex v_i^* in the above minimization is set to the average surface variation of vertex v_i^*'s corresponding position c_i^*, measured on $\mathcal{T}$. This weighting scheme allows for a tighter approximation in high curvature areas of target model (where the weights are higher) and good triangle quality in flat areas (where the weights are lower). Fold-overs and incorrect correspondences may still occur when the regions of unmatched vertices are extremely concave, but such situations are ultimately eliminated by the iterative relaxation and re-fitting process.

2) Surface detail fitting: After obtaining the dual mesh geometry in Eq. 3, we need to transform the fitting result back to the primal domain. We minimize the following energy in the primal domain:

$$\mathcal{V}_{\mathcal{F}} = \arg\min_{\mathbf{v}} \left(\left\| L\mathbf{v} \right\|^2 + \alpha C_1(\mathbf{v}) + \beta C_2(\mathbf{v}) + \gamma C_3(\mathbf{v}) \right) . \qquad (4)$$

Here, apart from the bi-Laplacian term that again serves for regularization, we use three additional energy terms C_1, C_2, C_3 to improve the fitting of the vertex positions, according to the geometry of the dual mesh $\mathcal{V}_{\mathcal{F}}^*$ and the target $\mathcal{T}$.

The first energy term C_1 is the fitting term to the dual mesh. Since each dual vertex corresponds to a primal face, we simply ask the centroid of the vertices of each primal face to fit the dual vertex:

$$C_1(\mathbf{v}) = \sum_{i=1}^{n_f^s} w_i \left\| \frac{1}{3} \cdot (\mathbf{v}_{i1} + \mathbf{v}_{i2} + \mathbf{v}_{i3}) - \mathbf{v}_i^* \right\|^2 \qquad (5)$$

where $\{i_1, i_2, i_3\}$ are the indices of the vertices participating in triangle t_i^s and $\mathbf{v}_i^*$ is the position of the corresponding dual vertex obtained earlier in Eq. 3. The weights w_i are again computed by the surface variation measure [29].

At a first glance, it may seem that the above energy is enough to transform the template mesh back into primal domain; however, the high-curvature areas may receive insufficient fitting quality if no reliable correspondences were found there earlier. To improve the fitting of surface features, we thus add two additional terms C_2 and C_3. As before, to detect high-curvature areas on $\mathcal{T}$ we rely on the surface variation measure [29] with a user-defined threshold.

For each high curvature point $\mathbf{v}_i^t$ on $\mathcal{T}$, we use the reliable projection criteria described in the previous section to find a corresponding position in the current template geometry $\mathcal{M}_{\mathcal{F}}$; denoted that projected position by $\mathbf{p}_i$. Assume that $\mathbf{p}_i$ is on face t_j^s; we can then write it as a linear combination of t_j^s's vertices using barycentric coordinates: $\mathbf{p}_i = b_1\mathbf{v}_{j1}^s + b_2\mathbf{v}_{j2}^s + b_3\ \mathbf{v}_{j3}^s$. Just as in the dual mesh fitting, some high-curvature points will not have a corresponding projected points because no reliable match could be found, and thus they remain unmatched. We define C_2 as the target-to-template fitting terms for the matched high-curvature points and C_3 the corresponding term for the unmatched points:

$$C_2(\mathbf{v}) = \sum_{i \in \mathcal{B}} w_i \left| (b_1\mathbf{v}_{j1} + b_2\mathbf{v}_{j2} + b_3\mathbf{v}_{j3}) - \mathbf{v}_i^t \right|^2 \qquad (6)$$

where $\mathcal{B}$ represents the set of matched high-curvature points on $\mathcal{T}$. Similarly, the target-to-template fitting term C_3 is formulated by the geometry constraints of the unmatched points, where we denote by $\mathcal{U}$ the set of unmatched high-curvature points and $\mathbf{v}_{k(i)}$ the vertex of the current $\mathcal{M}_{\mathcal{F}}$ nearest to an unmatched point $\mathbf{v}_i^t$ (if a vertex of the template ends up corresponding to several unmatched points, we only keep the farthest of those to participate in C_3 to avoid getting stuck at a local minimum):

$$C_3(\mathbf{v}) = \sum_{i \in \mathcal{U}} w_i \left\| \mathbf{v}_{k(i)} - \mathbf{v}_i^t \right\|^2 . \qquad (7)$$

The importance of the C_3 energy term will decrease throughout the iterative relaxation, because we fit a reasonable guess to each unmatched point and thus in the course of the iterations the number of the unmatched points will decrease.

There is a trade-off between the mesh quality and feature correspondence in the optimization of the surface feature fitting. Enforcing the term C_1 can obtain a better mesh quality, and on the contrary, enforcing the terms C_2 and C_3 can obtain a better correspondence on high-curvature area. In all our experiments, to take mesh quality more consideration, we give a larger value to the weighting factor (i.e., 3.0) for the term C_1 and give a smaller value to the weighting factors and (i.e., 1.0) for the terms C_2 and C_3. Similar to Eq. 1, both Eqs. 3 and 4 can be efficiently solved by a direct sparse linear solver.

Iterative fitting: To summarize, we perform the fine fitting in the dual domain (Eq. 3) and the primal domain (Eq. 4) in alternating steps, updating the positional correspondences each time. In each iteration, we keep monitoring the decreasing of Hausdorff distance [9] between template and target surface. When the decreasing is less than 1% of current distance, we stop the optimization; typically no more than 30 iterations were required for convergence.

C. Mesh Subdivision

If the number of vertices in the template mesh is insufficient to tightly approximate the target model model, we subdivide the deformed template mesh. Any uniform or adaptive subdivision approach can be adopted here; we use simple 1-to-4 uniform subdivision [18]. The positions of the newly introduced vertices are determined by re-solving Eq. 4, i.e., re-fitting using the new connectivity. To solve Eq. 4, the corresponding position of a new vertex is determined by a linear interpolation of the three points in the target model whose projection positions are the nearest to the new vertex (see Fig. 7). After a few subdivision iterations (typically one or two in all experiments), the target model can be approximated very well and the shapes of the triangles remain near equilateral. We show an example of this the subdivision post-processing step in Fig. 7.

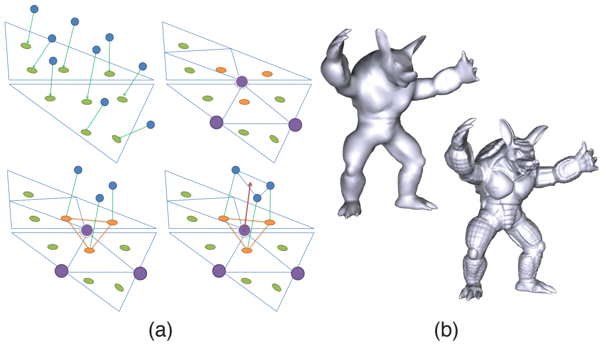

Fig. 7. Subdivision post-processing. (a) Determination of the corresponding positions for the new vertices. The blue and the green dots represent the points in the target model and their projected positions in the template model, respectively. The purple points represent the new vertices and the red arrows indicate their corresponding positions; (b) the subdivided result.

V. Experimental Results and Comparisons

A. Dual-domain Relaxation

Since the proposed template-based fitting scheme is based on dual-domain relaxation, we start this section with the comparisons of fitting with and without this step, i.e., we compare to using primal-domain relaxation only (see Fig. 8). As expected, the fitting results obtained by dual-domain relaxation are better in terms of the triangle quality, as many sliver triangles occur when performing solely primal-domain relaxation. In the meantime, the Fig. 9 shows statistical information for ratio between the inner angle of triangle over the ideal angle (i.e, 360 degree /vertex valence in a 1-ring neighborhood). The x-axis indicates possible ratio ranges in Fig. 9. The y-axis indicates the ratio of the histogram over the total integral of the histogram for each possible ratio range in the x-axis. Fig. 9(a) and (b) show that our method with dual domain relaxation (blue) yields better angle ratio than one without it (red) and even better than original mesh (green). In these results, the triangles quality can be improved by the proposed method. In addition, we also measure the number of fold-over triangles with/without dual domain relaxation for Fig. 8. With dual domain relaxation, the proposed method does not generate any triangle foldover. However, without dual domain relaxation, there are 408 and 359 fold-over triangles for both upper and bottom examples in Fig. 8.

According to this experiment, we believe the same phenomenon could also occur in the results of Zhang et al. [43]. Later in Sec. V-C, we will experimentally compare the proposed method with [43] in numerous examples.

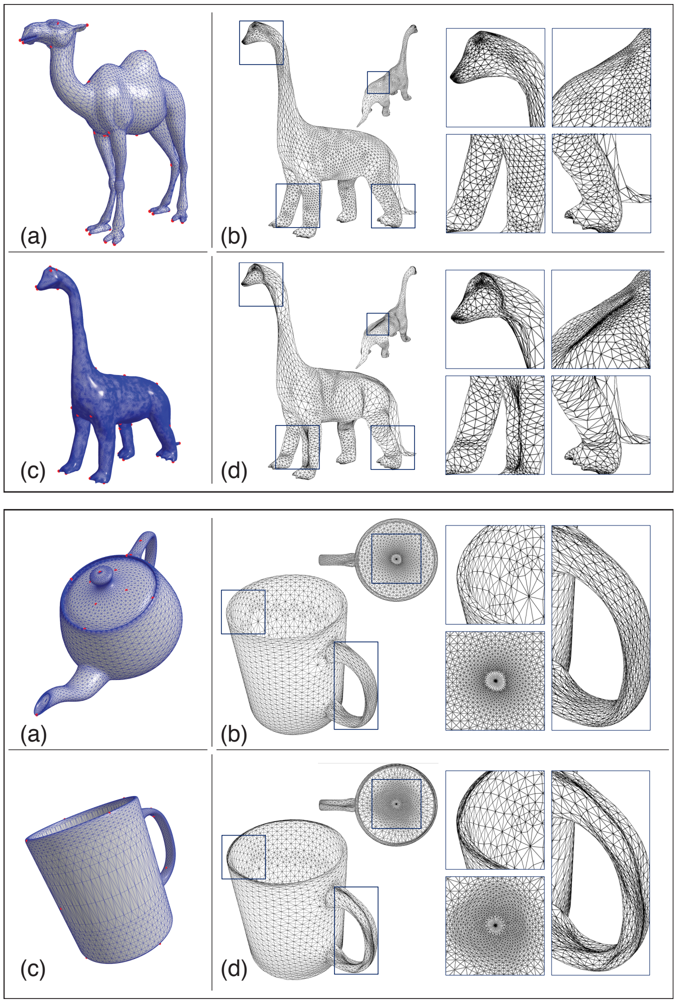

Fig. 8. Comparisons of fitting with and without dual-domain relaxation. (a) Template mesh; (b) with our dual-domain relaxation; (c) Target point model; (d) without dual-domain relaxation.

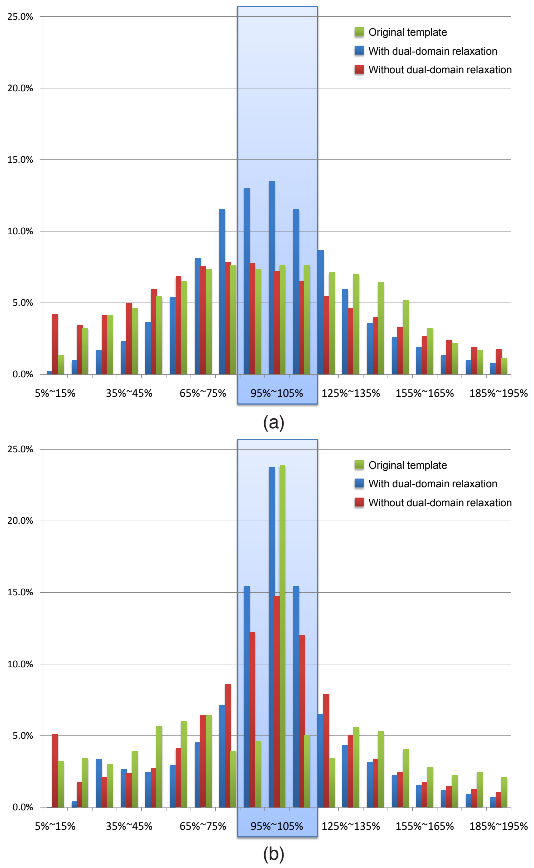

Fig. 9. A plot to compare the triangle quality in terms of angle ratio for original template (green), and results with (blue) and without (red) the dual domain relaxation. Both (a) and (b) are statistics information for the upper and bottom cases in Fig. 8, respectively.

B. *Local Surface Flattening*

Fig. 10 shows comparison results with and without the proposed local flattening method. In this example, we use the *Cat* model (Fig. 10(b)) as a template to fit the *Bunny* model (Fig. 10(a)). Fig. 10(c) is the coarse fitting result of the first stage. Fig. 10(d) is the final fitting result using our method. In Fig. 10(d), our approach requires only 8 iterations to resolve the foldovers in the marked area, and Fig. 10(e) shows a closeup on the region at the 4th, 8th, 12th and 16th iteration. However, without surface flattening, the method requires 40 iterations to resolve foldovers in the same area (Fig. 10(f) shows corresponding intermediate results). This experiment shows that the flattening process can aid dual-domain relaxation to resolve triangles foldovers and greatly improve the convergence speed as well.

C. *Experimental Comparison*

To evaluate the proposed approach, we compare our results with the two most related recent works: [39] and [43]. In particular, Zhang et al. [43] also use LS-meshes. Fig. 11 shows several examples using the sphere to fit target models. In this experiment, our results are better than [39], [43] in terms of the fitting results and triangle quality. Fig. 12 shows additional comparison examples. Some of these examples have very different shapes of target and template models. Consequently, both [39], [43] have significant difficulties handling these inputs, in particular, for the first three examples in Fig. 12; in contrast, our method can handle these models well. Furthermore, Fig. 13 compares our technique with the approaches of [39], [43] in the cases where the template models differ from the target models in their poses. Our method performs more robustly than the previous approaches in this situation as well. Finally, we use the same set of examples as in [43] for further evaluation of our method in Fig. 14. Observing the shape quality of the triangles in both fitting results, ours is better than that of [43]. Even though [43] discard original geometry like our system, their results still suffer from numerous sliver triangles clumped together in several places. Additionally, we also show the fitting error measured using Hausdorff distance [9] in this figure.

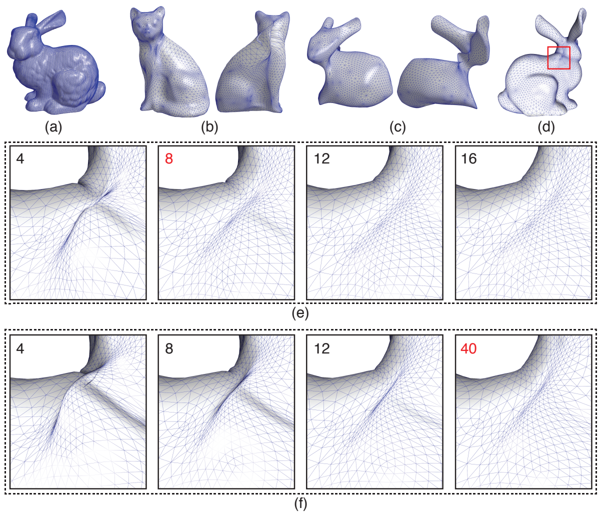

Fig. 10. (a) target model; (b) a bad-quality mesh can be used as a template in our method, and it still leads to a reasonable result; (c) coarse fitting result at the first stage; (d) final fitting result using the proposed method; (e) intermediate results at the 4th, 8th, 12th and 16th iteration *with* surface flattening; (f) fitting result obtained *without* surface flattening, requiring 40 iterations to resolve foldovers.

D. *Performance*

Generally, the approximation of fitting results to the original model is related to the resolution of template mesh. The high resolution can lead to a better approximation but take longer timing to finish fitting. Usually, from our experience, after some resolution, the approximation will not be increased significantly. So, it is not necessary to use too fine mesh as the template. Fig. 15. shows an example of fitting results using different resolution meshes as the template. In addition, we also like to verify how robust our method can perform against different triangle configurations. Fig. 16. shows such an example of using a cat model as the template to fit a bunny model. Fig. 16(a) is the result using the original cat model, (b) is the result by re-meshing the cat model that becomes an irregular mesh with many thin triangles and (c) is the result by subdividing the mesh and thus the template mesh with very high valence vertices. As shown in this figure, our

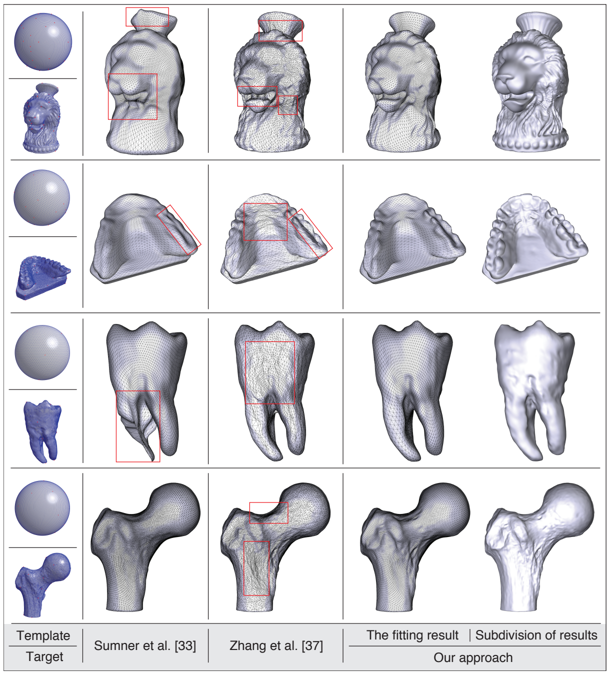

Fig. 11. Result comparison for different methods.

fitting results are very insensitive to the triangle quality of the template mesh and our method still generates good/acceptable quality triangle fitting results under (b) and (c) conditions.

E. Statistics

Various template and target models were tested in our experiments, as summarized in Table I. The experiments were performed on a PC with an Intel Core 2 Duo E6420 CPU and 4GB memory. The fitting error of final result was measured using the METRO tool [9] in this table.

Also we provide an additional plot to show how approximation error distance was decreased during the optimization process in Fig. 17. As shown in this figure, the approximated results will become stable after some fixed iterations in the optimization process. In the examples, we have target and template models with both similar and different shapes (the images of the fitting results can be found in the figures of this paper). In addition, Table II shows the fitting error statistics of our comparison study with the related works.

F. Applications

Many approaches have been proposed for consistent parameterization of 3D models [16], [19], [32]–[34]. In the recent state-of-the-art techniques [16], [34], the input 3D models are embedded onto a constructed base mesh first, and then are compatibly resampled and remeshed using the common parameterization. This process proves difficult when the input models have high genus and/or complex shape. In contrast, we directly deform the template mesh to fit the target model. Our generated consistent parameterization is suited for further geometry processing applications, such as morphing and motion retargeting, since the elements of the compatible remeshed/subdivided meshes are of high quality. We demonstrate several morphing results in Fig. 18. In these examples, all meshes in the morphing sequence use the same connectivity of the template mesh, shown on the left. A template mesh can be conveniently used as the base domain for a sequence of shapes, for instance, when processing time series of moving hands with different geometry in Fig. 19: having a single template adapted to multiple shapes allows to establish and maintain cross-correspondence between them.

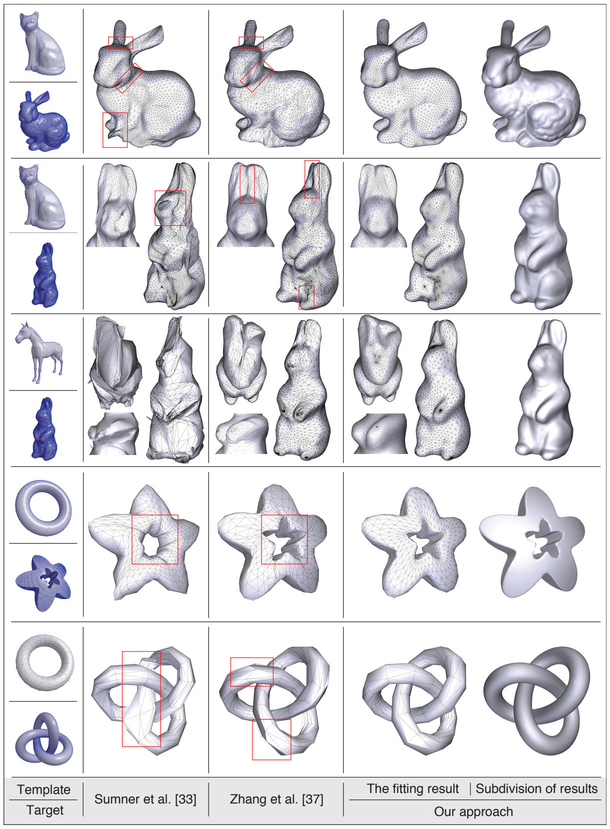

Fig. 12. Another side-by-side result comparison for different methods.

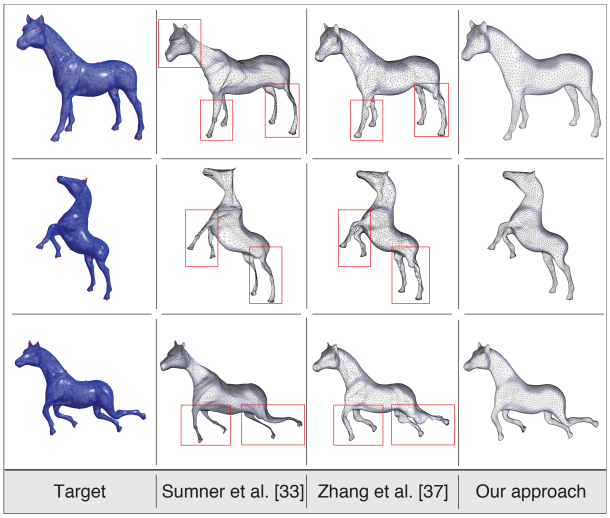

Fig. 13. Result comparison for template/target pairs with very different pose. In this study, the template model is a camel model in Figure 8.

TABLE I

PERFORMANCE SUMMARY. 2^{nd} COL.: THE NUMBER OF VERTICES (#V.) AND FACES (#F.) IN THE TEMPLATE MODEL; 3^{rd} COL.: THE NUMBER OF VERTICES IN THE TARGET MODEL; 4^{th} COL.: THE NUMBER OF USER-SPECIFIED VERTEX CORRESPONDENCE PAIRS (#F.V.); 5^{th} COL.: THE NUMBER OF VERTICES AND FACES IN THE SUBDIVIDED MODELS; 6^{th} COL.: THE TOTAL COMPUTATION TIME. 7^{th} COL.: HAUSDORFF DISTANCE W.R.T. THE BOUNDING BOX DIAGONAL [9].

Model		**Template Model**		**Target Model**	**Feature Vertices**	**Subdivided Model**		**Time**	**Hausdorff distance**
Template	Target	#V	#F	#V	$\#FV$	#V	#F	(sec)	Subdivided Model
Camel	Horse	6k	12k	80.6k	34	18k	36k	78.98	0.0027
Camel	Dinosaur	6k	12k	87.5k	37	54k	108k	81.77	0.0064
Cat	Bunny	2.6k	11k	104k	26	50.5k	101k	97.88	0.0077
Head	Venus head	7.7k	15.4k	150k	21	69k	138.6k	73.27	0.0064
Head	Shaven	7.7k	15.4k	134.9k	24	69k	138.6k	118.25	0.0211
Horse	Rabbit	3k	6k	90k	40	80.5k	160.9k	29.65	0.0030
Sphere	BallJoint	18.8k	37.5k	51k	25	56k	112k	210.179	0.0030
Sphere	Lion vase	18.8k	37.5k	23.9k	38	56.2k	112.5k	297.25	0.0963
Sphere	Gums	6.3k	12.5k	87.4k	30	56.2k	112.5k	54.62	0.0088
Sphere	Tooth	6.3k	12.5k	65.7k	16	56.2k	112.5k	47.324	0.0280
Teapot	Mug	9.2k	18.3k	98.8k	22	27.5k	55k	70.61	0.0333
Torus	Torus-knot	0.3k	0.6k	23k	23	7.8k	15.6k	9.22	0.0045
Torus	Star	1k	2k	83k	25	27k	54k	10.89	0.0053

TABLE II

APPROXIMATION QUALITY SUMMARY. 1^{st} ROW: TEMPLATE MODEL; 2^{nd} ROW: TARGET MODEL; $3^{rd} - 5^{th}$ ROW: THE HAUSDORFF DISTANCE W.R.T. THE BOUNDING BOX DIAGONAL [9]. IN THIS TABLE, WE DO NOT PERFORM SUBDIVISION FOR OUR APPROACH TO CONDUCT A FAIR COMPARISON.

Template	**Cat**		**Horse**	**Torus**		**Sphere**			
Target	**Bunny**	**Rabbit**	**Rabbit**	**Star**	**Knot**	**Lion Vase**	**Gums**	**Tooth**	**Ball Joint**
Sumner et al. [39]	0.0730	0.0509	0.0842	0.0573	0.0864	0.1840	0.0251	0.1569	0.1648
Zhang et al. [43]	0.0363	0.0092	0.0304	0.0217	0.0611	0.0164	0.0138	0.0251	0.0104
Our approach	0.0118	0.0046	0.0112	0.0343	0.0172	0.0096	0.0109	0.0210	0.0035

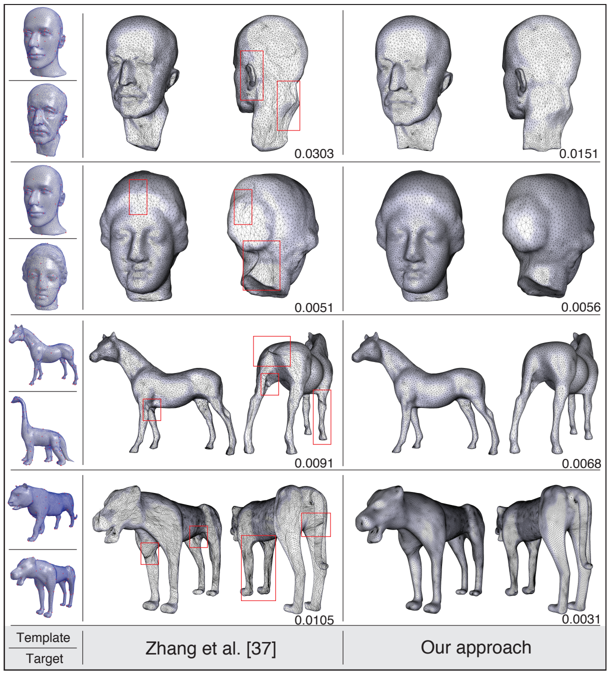

Fig. 14. Another series of comparison with Zhang et al. [43]. Their fitting results in sliver triangles clumped together or even foldovers in several regions. In contrast, our approach generates better triangle shapes and avoids foldovers. In addition, our fitting approach achieves similar or better shape approximation than [43]; the fitting error was measured by the Hausdorff distance [9] and is indicated in each example.

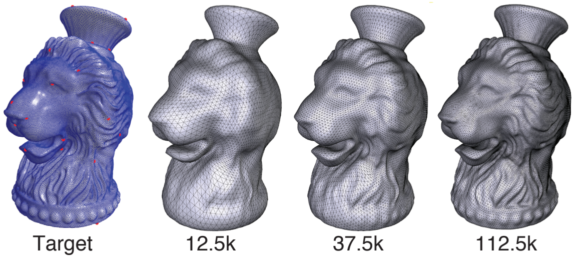

Fig. 15. An example of fitting results with different resolution template meshes. The computing time to converge and the distance error between template and target are (31.1 sec., 0.018), (100.1 sec., 0.0084) and (823.2 sec., 0.0045) for these there resolutions from left to right.

G. Discussion

To the best of knowledge, Sumner et al. [39] and Zhang et al. [43] are the most related works in literature to our approach, since their approaches gradually deform a template model to fit a target model. We share the same spirit as theirs. On the other hand, both Kraevoy et al. [16] and Schreiner et al. [34] did not employ template fitting for their cross-parameterization. Both approaches heavily depend on surface parameterization, compatible surface partitions etc. Our approach did not perform surface parameterization (embedding) and surface partitions. Therefore, their approaches are quite different from ours. Note that their main purpose is to achieve cross-parameterization that is one of applications provided by our paper.

Template fitting has been used in many geometry processing applications such as surface reconstruction from point clouds, cross-parameterization, remeshing, and subdivision. Therefore, comparing our approach with [16], [34], in terms of other applications (including surface reconstruction, remeshing, and subdivision), is not fair for these two approaches. Similarity, it is not fair for our approach to compare them in terms of solely the application of cross-parameterization. For cross-parameterization, in term of cross-parameterization quality, [34] achieved very good quality and much higher

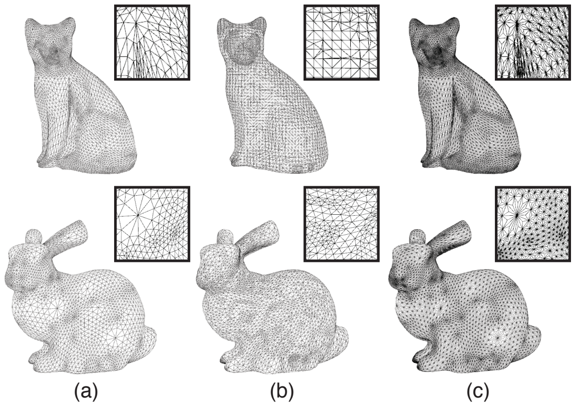

Fig. 16. The bunny fitting results obtained from the cat templates with varied triangle connectivity. On the top row: (a) original cat template, (b) cat template with irregular connectivity and thin triangles and (c) cat template with high valence triangle vertices. As a result, on the bottom row, our method can still generate good triangle quality fitting results regardless of (a), (b) and (c) triangle connectivity.

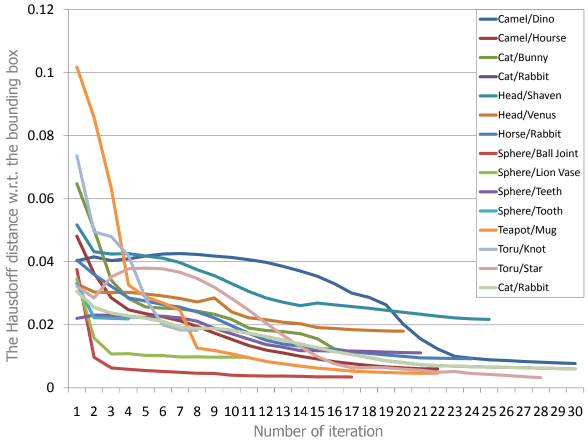

Fig. 17. A plot to show how approximation error distance is decreased v.s. the number of iterations in the optimization process. The vertical axis is the distance error measured by Hausdorff distance between template and target surfaces and the horizontal axis shows the number of iterations required.

than [16]. Ours is comparable to [16]. However, as addressed by [34] , to generate high quality result, their approach is very computationally intensive (couple hours) and is appropriate only for two input meshes rather than an army of input models. On the other hand, in terms of mesh quality, our approach is better than these two approaches since we enforce the mesh to be near-equilateral in the optimization of surface fitting. Furthermore, [34] indicated that the system propose by [16] is only robust for genus-zero model. To build a consistent surface partition for high genus model, their algorithm may not succeed with arbitrary feature sets. In contrast, with our proposed method, the template model can be arbitrary genus and the fitting result can be generated successfully with arbitrary feature sets.

VI. Conclusion and Future Work

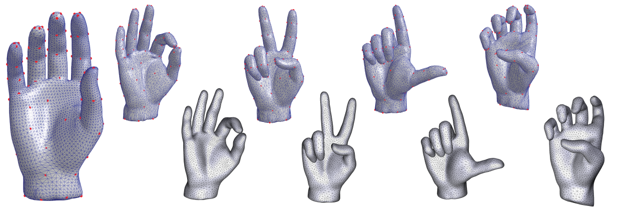

Fig. 19. A single hand mesh template (Left) is adapted to fit multiple hand-shaped models with different connectivity and pose (Right). Our approach can establish and maintain cross-correspondence between the models to facilitate hand animation with different poses. The Hausdorff distance, i.e., fitting error, of all fitting results is below 0.015.

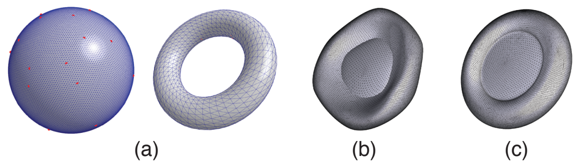

Fig. 20. Using a genus-0 sphere to fit a genus-1 torus model can lead to an incorrect result.

We presented a novel approach for template-based 3D model fitting. Our method largely avoids artifacts such as fold-overs and poor mesh element quality by performing iterative relaxations in the dual and the primal domains. We have demonstrated the robustness of our approach when handling different shape and pose situations, and we have demonstrated the utility of the template-based fitting for consistent parameterization. Our approach has some limitations which we would like to eliminate in future work. Firstly, we currently cannot handle input models with different genus; if both tempalte and target models are not with the same genus, our method can not generate correct results as shown in Fig. 20. It would be interesting to develop a method that refines the topology of the template in addition to the geometry. Another challenging direction is to fit a template to the target mesh without any a priori knowledge about corresponding landmark points.

Acknowledgements

We are grateful to authors of [43], in particular, Ligang Liu and Lei Zhang, for sharing the software and examples of their paper.

References

[1] B. Allen, B. Curless, and Z. Popović, “Articulated body deformation from range scan data,” *ACM Trans. Graph.*, vol. 21, no. 3, pp. 612–619, 2002.

[2] B. Allen, B. Curless, and Z. Popović, “The space of human body shapes: reconstruction and parameterization from range scans,” *ACM Trans. Graph.*, vol. 22, no. 3, pp. 587–594, 2003.

[3] D. Anguelov, P. Srinivasan, D. Koller, S. Thrun, J. Rodgers, and J. Davis, “SCAPE: shape completion and animation of people,” *ACM Trans. Graph.*, vol. 24, no. 3, pp. 408–416, 2005.

[4] H. Biermann, A. Levin, and D. Zorin, “Piecewise smooth subdivision surfaces with normal control,” in *Proceedings of ACM SIGGRAPH*, 2000, pp. 113–120.

[5] M. Botsch, M. Pauly, L. Kobbelt, P. Alliez, B. Lévy, S. Bischoff, and C. Rössl, “Geometric modeling based on polygonal meshes,” in *SIGGRAPH ’07: ACM SIGGRAPH 2007 courses*. New York, NY, USA: ACM, 2007, Course Notes (revised), p. 1.

[6] E. Catmull and J. Clark, “Recursively generated b-spline surfaces on arbitrary topological meshes,” pp. 183–188, 1998.

[7] W. Chang and M. Zwicker, “Automatic registration for articulated shapes,” *Comput. Graph. Forum*, vol. 27, no. 5, pp. 1459–1468, 2008.

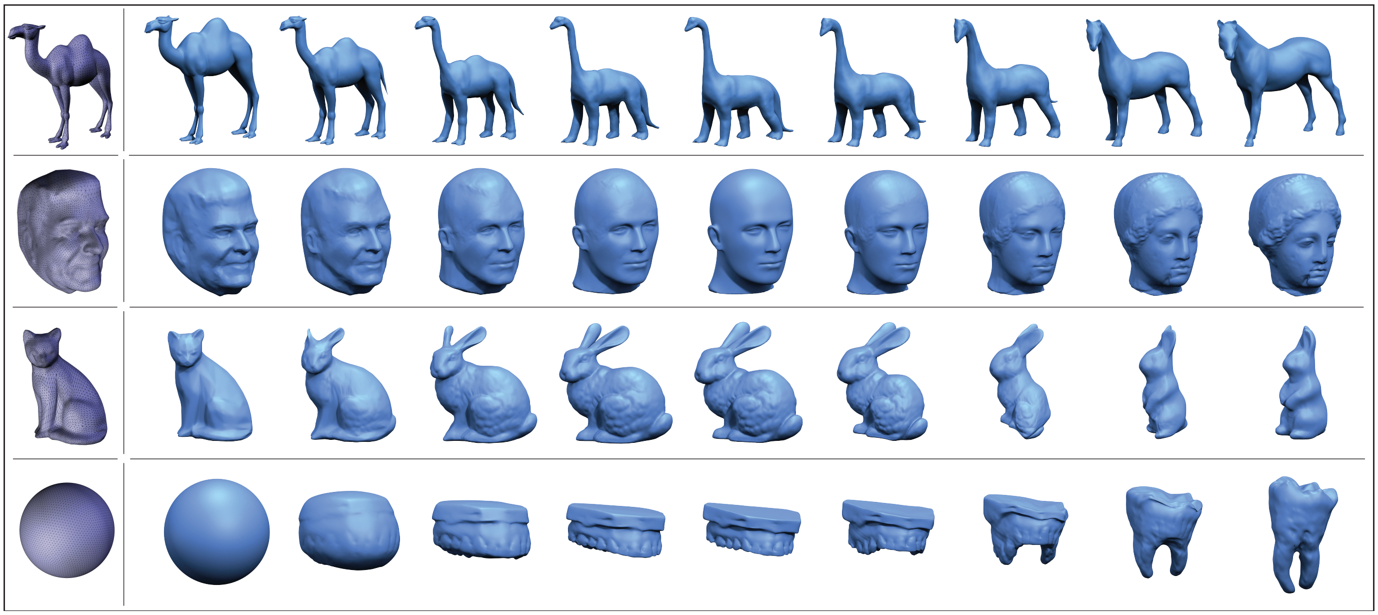

Fig. 18. (Left) Template mesh; (Right) morphing sequence.

[8] K.-S. D. Cheng, W. Wang, H. Qin, K.-Y. K. Wong, H. Yang, and Y. Liu, “Fitting subdivision surfaces to unorganized point data using sdm,” in *Pacific Graphics*. IEEE Computer Society, 2004, pp. 16–24.

[9] P. Cignoni, C. Rocchini, and R. Scopigno, “Metro: Measuring error on simplified surfaces,” *Comput. Graph. Forum*, vol. 17, no. 2, pp. 167–174, 1998.

[10] T. K. Dey, G. Li, and J. Sun, “Normal estimation for point clouds: a comparison study for a Voronoi based method,” in *Eurographics Symposium on Point-Based Graphics*, 2005, pp. 39–46.

[11] H. Hoppe, T. DeRose, T. Duchamp, M. Halstead, H. Jin, J. McDonald, J. Schweitzer, and W. Stuetzle, “Piecewise smooth surface reconstruction,” in *SIGGRAPH ’94: Proceedings of the 21st annual conference on Computer graphics and interactive techniques*. New York, NY, USA: ACM, 1994, pp. 295–302.

[12] J. Hu, L. Liu, and G. Wang, “Dual Laplacian morphing for triangular meshes,” *Comput. Animat. Virtual Worlds*, vol. 18, no. 4-5, pp. 271–277, 2007.

[13] Q.-X. Huang, B. Adams, M. Wicke, and L. J. Guibas, “Non-rigid registration under isometric deformations,” *Comput. Graph. Forum*, vol. 27, no. 5, pp. 1449–1457, 2008.

[14] A. Khodakovsky, N. Litke, and P. Schröder, “Globally smooth parameterizations with low distortion,” *ACM Trans. Graph.*, vol. 22, no. 3, pp. 350–357, 2003.

[15] O. Kin-Chung Au, C.-L. Tai, L. Liu, and H. Fu, “Dual laplacian editing for meshes,” *IEEE Transactions on Visualization and Computer Graphics*, vol. 12, no. 3, pp. 386–395, 2006.

[16] V. Kraevoy and A. Sheffer, “Cross-parameterization and compatible remeshing of 3D models,” *ACM Trans. Graph.*, vol. 23, no. 3, pp. 861–869, 2004.

[17] V. Kraevoy and A. Sheffer, “Template-based mesh completion,” in *Eurographics Symposium on Geometry Processing*, 2005, pp. 13–22.

[18] A. Lee, H. Moreton, and H. Hoppe, “Displaced subdivision surfaces,” in *Proceedings of ACM SIGGRAPH*, 2000, pp. 85–94.

[19] A. W. F. Lee, D. Dobkin, W. Sweldens, and P. Schröder, “Multiresolution mesh morphing,” in *Proceedings of ACM SIGGRAPH*, 1999, pp. 343–350.

[20] H. Li, B. Adams, L. J. Guibas, and M. Pauly, “Robust single-view geometry and motion reconstruction,” *ACM Transactions on Graphics (Proceedings SIGGRAPH Asia 2009)*, vol. 28, no. 5, December 2009.

[21] H. Li, R. W. Sumner, and M. Pauly, “Global correspondence optimization for non-rigid registration of depth scans,” *Comput. Graph. Forum*, vol. 27, no. 5, pp. 1421–1430, 2008.

[22] W.-C. Li, N. Ray, and B. Lévy, “Automatic and interactive mesh to t-spline conversion,” in *SGP ’06: Proceedings of the fourth Eurographics symposium on Geometry processing*. Aire-la-Ville, Switzerland, Switzerland: Eurographics Association, 2006, pp. 191–200.

[23] N. Litke, A. Levin, and P. Schröder, “Fitting subdivision surfaces,” in *Proceedings of IEEE Visualization*. IEEE Computer Society, 2001, pp. 319–324.

[24] C. Loop, “Smooth subdivision surfaces based on triangles,” *Master’s thesis, University of Utah, Department of Mathematics*, 1987.

[25] Q. Luo, B. Liu, Z.-g. Ma, and H.-b. Zhang, “Mesh editing in roi with dual laplacian,” in *CGIV ’07: Proceedings of the Computer Graphics, Imaging and Visualisation*. Washington, DC, USA: IEEE Computer Society, 2007, pp. 195–199.

[26] M. Marinov and L. Kobbelt, “Optimization methods for scattered data approximation with subdivision surfaces,” *Graphical Models*, vol. 67, no. 5, pp. 452–473, 2005.

[27] A. Nealen, T. Igarashi, O. Sorkine, and M. Alexa, “Laplacian mesh optimization,” in *Proceedings of ACM GRAPHITE*, 2006, pp. 381–389.

[28] A. Nealen, T. Igarashi, O. Sorkine, and M. Alexa, “FiberMesh: Designing freeform surfaces with 3D curves,” *ACM Transactions on Graphics (Proceedings of ACM SIGGRAPH)*, vol. 26, no. 3, p. article no. 41, 2007.

[29] M. Pauly, M. Gross, and L. Kobbelt, “Efficient simplification of point-sampled surfaces,” in *Proceedings of IEEE Visualization*. IEEE Computer Society, 2002, pp. 163–170.

[30] M. Pauly, N. J. Mitra, J. Giesen, M. Gross, and L. J. Guibas, “Example-based 3D scan completion,” in *Eurographics Symposium on Geometry Processing*, 2005, p. 23.

[31] U. Pinkall and K. Polthier, “Computing discrete minimal surfaces and their conjugates,” *Experiment. Math.*, vol. 2, no. 1, pp. 15–36, 1993.

[32] E. Praun and H. Hoppe, “Spherical parametrization and remeshing,” *ACM Trans. Graph.*, vol. 22, no. 3, pp. 340–349, 2003.

[33] E. Praun, W. Sweldens, and P. Schröder, “Consistent mesh parameterizations,” in *Proceedings of ACM SIGGRAPH*. ACM, 2001, pp. 179–184.

[34] J. Schreiner, A. Asirvatham, E. Praun, and H. Hoppe, “Inter-surface mapping,” *ACM Trans. Graph.*, vol. 23, no. 3, pp. 870–877, 2004.

[35] A. Sharf, T. Lewiner, A. Shamir, L. Kobbelt, and D. Cohen-Or, “Competing fronts for coarse-to-fine surface reconstruction,” *Comput. Graph. Forum*, vol. 25, no. 3, pp. 389–398, 2006.

[36] O. Sorkine and D. Cohen-Or, “Least-squares meshes,” in *Proceedings of Shape Modeling International*. IEEE Computer Society, 2004, pp. 191–199.

[37] T. Steinbrecher and M. Gerth, “Dental inlay and onlay construction by iterative laplacian surface editing,” *Comput. Graph. Forum*, vol. 27, no. 5, pp. 1441–1447, 2008.

[38] C. Stoll, Z. Karni, C. Rössl, H. Yamauchi, and H.-P. Seidel, “Template deformation for point cloud fitting,” in *Eurographics Symposium on Point-Based Graphics*, 2006, pp. 27–35. [Online]. Available: http://cgal.inria.fr/Publications/2006/SKRYS06

[39] R. W. Sumner and J. Popović, “Deformation transfer for triangle meshes,” *ACM Trans. Graph.*, vol. 23, no. 3, pp. 399–405, 2004.

[40] J. Süßmuth, M. Winter, and G. Greiner, "Reconstructing animated meshes from time-varying point clouds," *Comput. Graph. Forum*, vol. 27, no. 5, pp. 1469–1476, 2008.

[41] H. Suzuki, S. Takeuchi, F. Kimura, and T. Kanai, "Subdivision surface fitting to a range of points," in *Pacific Graphics*. IEEE Computer Society, 1999, pp. 158–167.

[42] H. Zhang, A. Sheffer, D. Cohen-Or, Q. Zhou, O. van Kaick, and A. Tagliasacchi, "Deformation-driven shape correspondence," *Comput. Graph. Forum*, vol. 27, no. 5, pp. 1431–1439, 2008.

[43] L. Zhang, L. Liu, Z. Ji, and G. Wang, "Manifold parameterization," in *Proceedings of Computer Graphics International*, 2006, pp. 160–171.