ELASTIFACE: Matching and Blending Textured Faces

Eduard Zell, Mario Botsch
Computer Graphics & Geometry Processing, Bielefeld University

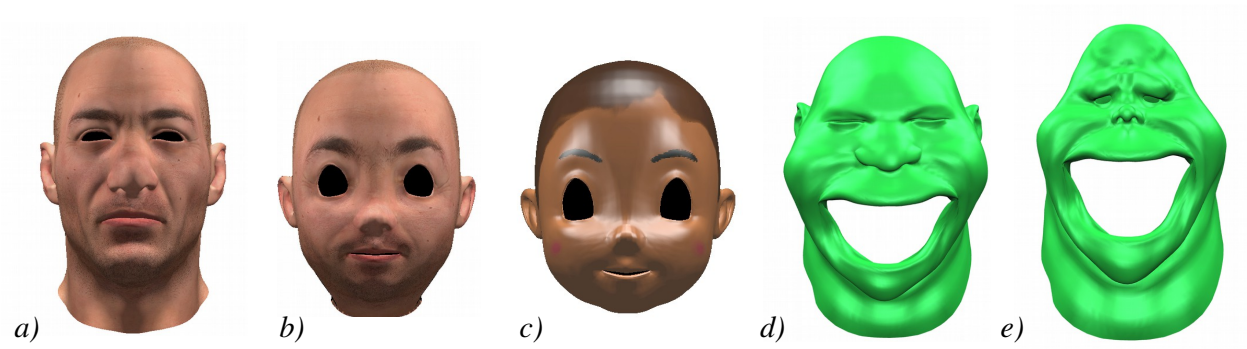

Figure 1: *Our* ElastiFace *framework matches faces with strongly different shapes and textures: Face (a) has been matched to (c) and (e), with face (b) being a 50% morph of (a) and (c), and face (d) being a part-based morph with mouth and neck of (e) and the rest of (a).*

Abstract

In this paper we present ELASTIFACE, a simple and versatile method for establishing correspondence between textured face models, either for the construction of a blend-shape facial rig or for the exploration of new characters by morphing between a set of input models. While there exists a wide variety of approaches for inter-surface mapping and mesh morphing, most techniques are not suitable for our application: They either require the insertion of additional vertices, are limited to topological planes or spheres, are restricted to near-isometric input meshes, and/or are algorithmically and computationally involved. In contrast, our method extends linear non-rigid registration techniques to allow for strongly varying input geometries. It is geometrically intuitive, simple to implement, computationally efficient, and robustly handles highly non-isometric input models. In order to match the requirements of other applications, such as recent perception studies, we further extend our geometric matching to the matching of input textures and morphing of geometries and rendering styles.

CR Categories: I.3.5 [Computer Graphics]: Computational Geometry and Object Modeling—Hierarchy and geometric transformations

Keywords: nonrigid registration, face morphing

1 Introduction

In human perception and communication faces are of significant importance, because they give people a distinguishable identity and reflect their mimics, emotions, and moods. It is therefore not surprising that digital characters are ubiquitous in computer games, special effects, and animation movies. But digital faces are also of increasing interest for perception studies [McDonnell et al. 2012], virtual prototyping for humanoid robots, or even for traditional stop-motion animation.

With ELASTIFACE we present a simple and effective framework for establishing correspondence between a given set of face models, both in terms of their geometries and textures, using a novel non-rigid registration technique. In contrast to most previous work, our method can handle input models of extremely different geometries. Nevertheless, its algorithmic core is based on solving simple Laplacian linear systems and therefore is easy to implement.

The main motivation of our approach is to *incrementally* build a database of face models with identical connectivity, i.e., with one-to-one vertex correspondence. Applications are, for instance, to add new expressions to a blend-shape rig at a late production stage or a face to a database during an ongoing study. In the former case, changing the connectivity during face matching would require to recreate a significant amount of the facial rig. Similar constraints apply for ongoing psychological studies where input data is prohibited to change in order to be comparable. In our target application we plan to analyze how likable or trustworthy a virtual character is depending on age, gender, ethnicity, and degree of realism, where the origin and look of these models can be arbitrary, meaning that a hand-modeled cartoon character from the Internet should be equally well handled as a carefully sculpted or scanned model.

Although there exists a wide variety of approaches for mapping one mesh onto another, most of them disqualify for our face morphing application due to their inherent limitations. Methods based on mapping the input models to a common simple parameter domain typically require the input models to be homeomorphic to a plane or a sphere. More general cross-parameterization techniques have to adjust the mesh connectivity, thereby breaking the (required) one-to-one correspondence when matching to several target models. Non-rigid registration approaches avoid these topological limitations by deforming a given template model to different target shapes, but most of them are restricted to near-isometric input models. For the strongly varying face models we are interested in, their closest point correspondences fail to give valid results.

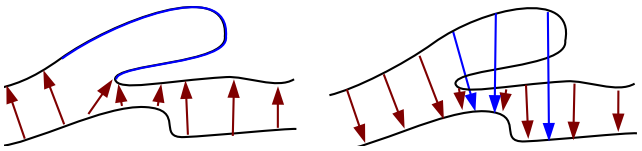

Figure 2: *Difficult case for ICP-based non-rigid registration. Left: The lower surface mainly aligns to the black part of the surface and discard the details marked blue. Right: The blue correspondences cause self-intersections when aligning the upper surface to the lower one.*

By combining the concepts of (i) deformation-based registration and (ii) transformation of models into a simpler domain we overcome the individual limitations of these approaches. A novel simultaneous fairing technique transforms source and target meshes into a simple, feature-less, and geometrically very similar state, from which accurate correspondences for a non-rigid registration of the original meshes can be robustly determined. This extension to closest point correspondences is the key contribution of our work. Our fairing and fitting techniques are both simple to implement and computationally efficient, since both are based on the minimization of a geometrically intuitive quadratic energy. In addition to matching the shape of the given input models, we further extend our framework by blending between facial parts, input textures, and several rendering styles, as shown in Figure 1.

2 Related Work

Establishing a mapping from one model to another has been investigated in several fields and under different names, and therefore is referred to as mesh morphing, cross-parameterization, non-rigid registration, or correspondence estimation. We can only discuss the most relevant methods here, and refer the reader to the book of Bronstein et al. [2008], the survey of van Kaick et al. [2011], or the course of Chang et al. [2010] for more details.

An overview of early mesh morphing methods is given in [Alexa 2002]. The described methods typically parametrize the input models into a simple domain, such as a disk or a sphere. In case of the former, a constrained planar parameterization [Lévy 2001; Kraevoy et al. 2003] can then be used to establish correspondence. However, in both cases the input models are restricted to topological planes or spheres. Similar topological restrictions apply to the approaches of Blanz and Vetter [1999] or of Wang et al. [2008], who fit a model to a scan via cylindrical or disk parameterization.

In contrast, inter-surface mapping approaches [Kraevoy and Sheffer 2004; Schreiner et al. 2004] avoid the common parameterization domain by constructing a direct mapping between the two 3D models. Although these methods can match arbitrary non-isometric objects, they have to insert additional vertices to the resulting mesh, which breaks the requirements of our application of incrementally building a face database. Bronstein et al. [2006; 2008] also compute a direct mapping between two surfaces. They minimize parametric distortion through generalized multidimensional scaling (GMDS). While their method does not introduce new vertices, it is designed particularly for isometric or close-to-isometric models—a prerequisite not met by our strongly varying face models.

Other approaches embed the input models into spaces where correspondences are easier to detect. For instance, Ovsjanikov et al. [2010] match objects based on a single correspondence by embedding the models using the Heat Kernel Map. Lipman and colleagues [2009] map models to the complex plane using Möbius

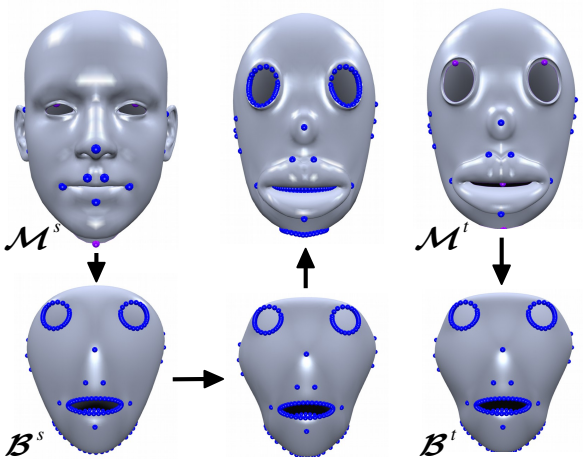

Figure 3: *Overview of the geometric matching: After selecting correspondence constraints (blue and pink dots), the source mesh $\mathcal{M}^s$ and target mesh $\mathcal{M}^t$ are smoothed into base meshes $\mathcal{B}^s$ and $\mathcal{B}^t$. Then $\mathcal{B}^s$ is fitted to $\mathcal{B}^t$ (bottom center), and—based on the resulting correspondences—$\mathcal{M}^s$ is fitted to $\mathcal{M}^t$ (top center).*

transformations and find correspondences by a voting scheme that evaluates candidate three-point correspondences. Both methods, however, are again designed for near-isometric input models. Kim et al. [2011] overcome this restriction by blending several of these intrinsic maps. However, as shown in Section 6 their method yields non-plausible mappings for our examples.

Non-rigid registration approaches overcome the limitation to certain topological types by deforming a template model until it matches the given target model. The approach of Lee et al. [2000] employs free-form deformation with manually specified curve constraints to fit a generic head model to photographs. Similarly, Bui et al. [2003] use an RBF space warp with feature points determined by a genetic algorithm. Both approaches have the drawback that their low-DoF space warps are not capable of mapping strongly differing geometries, e.g., to unfold a "source ear" to a "target non-ear" (see Figure 3). Similar restrictions apply to the correspondence estimation of [Noh and Neumann 2001], which employs an RBF warp followed by a cylindrical projection.

More recent non-rigid registration approaches fit several scans of deforming objects [Li et al. 2008; Huang et al. 2008; Tevs et al. 2009] or fit a template model to 3D scans [Allen et al. 2003; Amberg et al. 2007; Weise et al. 2009; Weise et al. 2011]. But most of these methods assume near-isometric deformations. For models of very different shape, e.g., because one model has ears while the other has not, these methods fail to find valid correspondences (Figure 2), which in turn leads to fold-overs and inter-penetrations in the mapped model, as we demonstrate in Section 6.

Our method can be considered as a combination of the ideas presented above: Similar to non-rigid registration, it deforms a source/template model into given target shapes. But it avoids the problem of invalid closest point correspondences by first mapping source and target models into a simpler space and computing correspondences there. The simpler space, however, is not a planar or spherical parameterization, but a smoothed, feature-less version of the input models—computed by the joint fairing technique proposed in the next section.

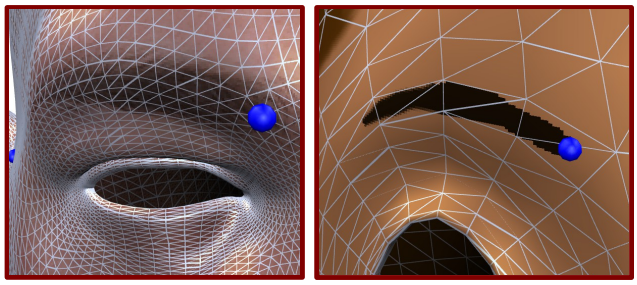

Figure 4: *In order to accurately specify corresponding points at the eyebrow, we have to use interior triangle points instead of simple vertex-to-vertex correspondences.*

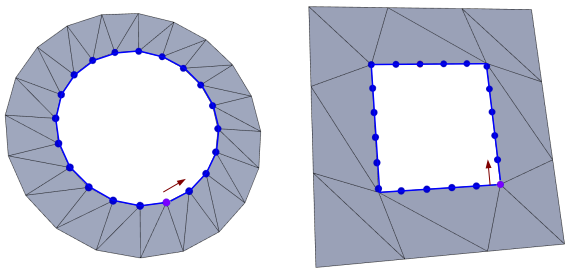

Figure 5: *Starting from one or more correspondence constraints on a boundary loop (purple) we automatically assign correspondences for all other boundary loop vertices (blue) based on relative distance along the boundary loop.*

3 Geometry Matching

The first step of our face matching is to geometrically map the source face mesh $\mathcal{M}^{\mathrm{s}}$ onto the target face model $\mathcal{M}^{\mathrm{t}}$. To this end we adjust the source model's vertex positions only, and keep its connectivity fixed. After the user has manually marked a few correspondences (Section 3.1), we first transform both the source and the target model $\mathcal{M}^{\mathrm{s}}$ and $\mathcal{M}^{\mathrm{t}}$ into smoothed *base meshes* $\mathcal{B}^{\mathrm{s}}$ and $\mathcal{B}^{\mathrm{t}}$ (Section 3.2), which afterwards are brought into correspondence by a non-rigid registration approach (Section 3.3). The correspondences derived from the smoothed models $\mathcal{B}^{\mathrm{s}}$ and $\mathcal{B}^{\mathrm{t}}$ are then used as initial guess for the registration of the original models $\mathcal{M}^{\mathrm{s}}$ and $\mathcal{M}^{\mathrm{t}}$. This process is depicted in Figure 3 and explained below.

In the following we denote the vertices of the source and target mesh by $\mathbf{x}_i^{\mathrm{s}}$, $i = 1, \ldots, n$, and $\mathbf{x}_j^{\mathrm{t}}$, $j = 1, \ldots, m$, respectively. If we particularly emphasize properties of the original, undeformed meshes or of the smoothed meshes, we denote this by a bar ($\bar{}$) or a hat ($\hat{}$), respectively.

3.1 Manual Correspondence Specification

As in most morphing or cross-parameterization approaches, the user initially specifies a few correspondences by selecting the respective feature points on both the source model $\mathcal{M}^{\mathrm{s}}$ and the target model $\mathcal{M}^{\mathrm{t}}$. In our application, the user typically marks about 15–20 correspondences for nose, eyes, mouth, and ears, as shown in Figure 3, left column.

In contrast to most other approaches, however, a simple vertex-to-vertex correspondence ($\mathbf{x}_i^{\mathrm{s}} \mapsto \mathbf{x}_j^{\mathrm{t}}$) is often not accurate enough in our context, since our face models are equipped with high-resolution textures. For instance, accurately selecting correspondences for the eyebrows in Figure 4 requires to specify reference points within triangles. Each of these reference points is represented by a barycentric combination of its triangle vertices. Our set of correspondence constraints therefore consists of tuples of reference points $(\mathbf{r}_k^{\mathrm{s}}, \mathbf{r}_k^{\mathrm{t}})$, $k = 1, \ldots, K$, represented as

$$\alpha_k^{\mathrm{s}}\mathbf{a}_k^{\mathrm{s}} + \beta_k^{\mathrm{s}}\mathbf{b}_k^{\mathrm{s}} + \gamma_k^{\mathrm{s}}\mathbf{c}_k^{\mathrm{s}} \quad \text{and} \quad \alpha_k^{\mathrm{t}}\mathbf{a}_k^{\mathrm{t}} + \beta_k^{\mathrm{t}}\mathbf{b}_k^{\mathrm{t}} + \gamma_k^{\mathrm{t}}\mathbf{c}_k^{\mathrm{t}}. \quad (1)$$

In the above equation, $\alpha_k, \beta_k, \gamma_k$ denote the barycentric coordinates of $\mathbf{r}_k$ with respect to the containing triangle $(\mathbf{a}_k, \mathbf{b}_k, \mathbf{c}_k)$, both on $\mathcal{M}^{\mathrm{s}}$ and $\mathcal{M}^{\mathrm{t}}$, respectively.

In order to accurately map boundaries (e.g., eyes, mouth, neck), the user can manually specify one or more points on corresponding boundary loops. The system then automatically determines target positions for all other vertices on the source boundary loop by preserving the relative edge lengths on the target boundary loop (see Figure 5). Note that the target positions will lie on boundary edges, and can therefore also be represented as barycentric combination of boundary vertices in the form of (1).

Based on these correspondences we initialize the registration by aligning the two models using the best-matching similarity transform between source and target. This amounts to minimizing the squared distances of transformed source points to their corresponding target points:

$$\min_{\mathbf{R},\,\mathbf{t},\,s} \sum_{k=1}^{K} \left\| s\,\mathbf{R}\,\mathbf{r}_k^{\mathrm{s}} + \mathbf{t} - \mathbf{r}_k^{\mathrm{t}} \right\|^2.$$

The optimal rotation $\mathbf{R}$, translation $\mathbf{t}$, and uniform scaling s can be computed in closed form [Horn 1987; Umeyama 1991].

3.2 Joint Fairing

By obtaining the reference points $(\mathbf{r}_k^{\mathrm{s}}, \mathbf{r}_k^{\mathrm{t}})$ for models with similar proportions and a similar level of geometric detail, it would be possible to establish a mapping between $\mathcal{M}^{\mathrm{s}}$ and $\mathcal{M}^{\mathrm{t}}$ using a deformation-based registration with ICP-like closest point constraints, such as the method of Weise et al. [2009]. However, if the models differ significantly, e.g., one model has ears while the other one has not, the closest point constraints fail to give reasonable results (see Figure 2). This typically leads to fold-overs and self-intersections in the deformed source mesh. To overcome the limitations of closest point correspondences in 3D, other approaches first transform both models into a simpler space and find correspondences there. For instance, Lipman et al. [2009] map models into the complex plane using a Möbius transform, which relaxes isometry to conformal equivalence. However, these methods typically require both models to differ only by a near-isometry, which is not the case for our application.

In contrast, we propose to transform both models into a simple, similar, and feature-less shape, on which we then compute robust correspondences. This transformation will be non-isometric such that the initial source and target models can differ significantly. The main idea is to (i) apply aggressive fairing to remove geometric details, (ii) force corresponding points to coincide to achieve a sufficient geometric similarity, and (iii) allow correspondences to move to a certain degree in order to unfold geometrically complex regions like mouth and nose (see Figure 3, bottom).

We perform this transformation by a simultaneous optimization of the vertex positions of both $\mathcal{M}^{\mathrm{s}}$ and $\mathcal{M}^{\mathrm{t}}$, where we minimize the following energy function:

$$E_{\mathrm{fair}}\big(\mathbf{x}_1^{\mathrm{s}},\dots,\mathbf{x}_n^{\mathrm{s}},\mathbf{x}_1^{\mathrm{t}},\dots,\mathbf{x}_m^{\mathrm{t}}\big) \;= \tag{2}$$

$$\frac{\lambda_1}{\sum_i A_i^{\mathrm{s}} + \sum_j A_j^{\mathrm{t}}}\left[\sum_{i=1}^{n} A_i^{\mathrm{s}}\left\|\Delta\mathbf{x}_i^{\mathrm{s}}\right\|^2 \;+\; \sum_{i=1}^{m} A_j^{\mathrm{t}}\left\|\Delta\mathbf{x}_j^{\mathrm{t}}\right\|^2\right] \tag{3}$$

$$+\;\frac{\lambda_2}{K}\sum_{k=1}^{K}\left\|\mathbf{r}_k^{\mathrm{s}}-\mathbf{r}_k^{\mathrm{t}}\right\|^2 \tag{4}$$

$$+\;\frac{\lambda_3}{K}\sum_{k=1}^{K}\left\|\frac{1}{2}(\mathbf{r}_k^{\mathrm{s}}+\mathbf{r}_k^{\mathrm{t}})-\frac{1}{2}(\bar{\mathbf{r}}_k^{\mathrm{s}}+\bar{\mathbf{r}}_k^{\mathrm{t}})\right\|^2. \tag{5}$$

The first term (3) penalizes the squared norms of per-vertex Laplacians $\Delta\mathbf{x}_i$, weighted by their Voronoi areas A_i. Minimizing this discrete version of the continuous thin plate energy removes geometric details and leads to as smooth as possible surfaces. For discretizing Laplacians and Voronoi areas we use the cotangent weights [Pinkall and Polthier 1993; Meyer et al. 2003].

The second term (4) penalizes the deviation of corresponding reference points, hence is responsible for making them coincident. Note that $\mathbf{r}_k^{\mathrm{s}}$ and $\mathbf{r}_k^{\mathrm{t}}$ are barycentric combinations of vertices $\mathbf{x}_i^{\mathrm{s}}$ and $\mathbf{x}_j^{\mathrm{t}}$ (see (1)), such that this objective can be formulated in terms of the latter. The last term (5) is required to avoid the trivial solution. It basically prescribes a target position for the two corresponding points $\mathbf{r}_k^{\mathrm{s}}$ and $\mathbf{r}_k^{\mathrm{t}}$, which can be considered to be coincident due to the second term. The target position is chosen as the point where $\mathbf{r}_k^{\mathrm{s}}$ and $\mathbf{r}_k^{\mathrm{t}}$ can meet with least movement, which is the average $\frac{1}{2}(\bar{\mathbf{r}}_k^{\mathrm{s}}+\bar{\mathbf{r}}_k^{\mathrm{t}})$ of their original positions (before the optimization).

Note that we do not combine the terms (4) and (5) since we want to enforce *strongly* that corresponding points become coincident, while we only enforce *weakly* a specific target position. This allows the reference points to move to a certain degree in order to further decrease the curvature term, which effectively leads to an unfolding of geometrically difficult parts, such as nose, mouth, and ears (Figure 3). If we strictly enforced $\mathbf{r}_k^{\mathrm{s}} = \mathbf{r}_k^{\mathrm{t}} = \frac{1}{2}(\bar{\mathbf{r}}_k^{\mathrm{s}}+\bar{\mathbf{r}}_k^{\mathrm{t}})$ the surface would not be able to unfold to a state without self-intersections. This behavior was achieved by $\lambda_1 = 0.1$, $\lambda_2 = 100$, $\lambda_3 = 1$ in almost all examples; only the Slimer model (Figure 12) and the clay faces (Figure 10) required a higher smoothing weight ($\lambda_1 = 10$) to fully unfold.

If we keep the cotangent weights and Voronoi areas fixed, then minimizing the quadratic objective function (2) amounts to solving three $(n+m)\times(n+m)$ linear systems of normal equations for the x, y, and z coordinates of the vertex positions of $\mathcal{M}^{\mathrm{s}}$ and $\mathcal{M}^{\mathrm{t}}$. This system is highly sparse, symmetric, and positive definite and we solve it using a sparse Cholesky factorization [Chen et al. 2008].

The results of this energy minimization noticably depend on the underlying triangulation, since the cotangent weights and Voronoi areas in fact depend nonlinearly on the vertex positions. Hence, the simple linear solve might not be a sufficiently accurate approximation to the true nonlinear solution. To take this mesh-dependence into account we iteratively update the weights and re-solve the linear system. To avoid numerical problems due to degenerated cotangent weights, we follow [Kazhdan et al. 2012] and update the Voronoi areas only, while keeping the cotangent weights fixed. Although this process might not converge in a theoretical sense, in practice there are no noticeable changes after 5–10 iterations.

The joint fairing approach turned out to be numerically very robust, and it yields two highly smooth and geometrically very similar *base meshes* $\mathcal{B}^{\mathrm{s}}$ and $\mathcal{B}^{\mathrm{t}}$—even for highly different initial geometries and tessellations—as shown in the second column of Figure 3 and the accompanying video.

3.3 Non-Rigid Registration

The two base meshes resulting from the joint fairing process are void of any geometric details due to smoothing and are geometrically very similar, since they correspond to discrete curvature-minimizing thin plate surfaces with identical Dirichlet boundary constraints. As such, they are an easy task for a deformation-based non-rigid registration approach. We therefore first deform the smooth source mesh $\mathcal{B}^{\mathrm{s}}$ onto the smoothed target model $\mathcal{B}^{\mathrm{t}}$, and then use their resulting vertex correspondences as initial guess for the registration of the original models $\mathcal{M}^{\mathrm{s}}$ and $\mathcal{M}^{\mathrm{t}}$.

For matching $\mathcal{B}^{\mathrm{s}}$ to $\mathcal{B}^{\mathrm{t}}$, we adjust the vertex positions of the former. To this end we iteratively minimize an energy consisting of a fitting term and a smoothness term, as it is done by most non-rigid registration approaches. The smoothness term (6) penalizes bending (i.e., change of curvature) of the source model, measured by the Laplacians of vertex displacements $\Delta(\mathbf{x}_i^{\mathrm{s}} - \hat{\mathbf{x}}_i^{\mathrm{s}})$, where $\hat{\mathbf{x}}_i^{\mathrm{s}}$ denotes the vertex position on mesh $\mathcal{B}^{\mathrm{s}}$. The fitting term (7) tries to minimize the distance of each source vertex $\mathbf{x}_i^{\mathrm{s}}$ to its closest point $\hat{\mathbf{c}}_i^{\mathrm{t}}$ on the smoothed target model $\mathcal{B}^{\mathrm{t}}$. The third term (8) ensures that the coincident reference points $\mathbf{r}_k$ remain at their position:

$$E_{\mathrm{fit}}(\mathbf{x}_1^{\mathrm{s}},\dots,\mathbf{x}_n^{\mathrm{s}}) \;= \frac{\mu_1}{\sum_i \hat{A}_i^{\mathrm{s}}}\sum_{i=1}^{n}\hat{A}_i^{\mathrm{s}}\left\|\Delta(\mathbf{x}_i^{\mathrm{s}}-\hat{\mathbf{x}}_i^{\mathrm{s}})\right\|^2 \tag{6}$$

$$+\;\frac{\mu_2}{n}\sum_{i=1}^{n}\left\|\mathbf{x}_i^{\mathrm{s}}-\hat{\mathbf{c}}_i^{\mathrm{t}}\right\|^2 \tag{7}$$

$$+\;\frac{\mu_3}{K}\sum_{k=1}^{K}\left\|\mathbf{r}_k^{\mathrm{s}}-\hat{\mathbf{r}}_k^{\mathrm{t}}\right\|^2. \tag{8}$$

While we use the Voronoi area $\hat{A}_i^{\mathrm{s}}$ of the smoothed base mesh $\mathcal{B}^{\mathrm{s}}$, we keep the cotangent weights that have been computed on the initial mesh $\mathcal{M}^{\mathrm{s}}$. Note that in contrast to most other non-rigid registration approaches we can use a simple quadratic energy, since both meshes are already very similar, such that the closest point constraints $(\mathbf{x}_i^{\mathrm{s}}, \hat{\mathbf{c}}_i^{\mathrm{t}})$ are meaningful and a linear deformation model is sufficient. The energy is again minimized by solving a sparse linear system, similar to the one of the previous section.

In an ICP-like manner [Besl and McKay 1992] we iteratively update closest point correspondences $(\mathbf{x}_i^{\mathrm{s}}, \hat{\mathbf{c}}_i^{\mathrm{t}})$ and re-solve the linear system, which typically converges after 3–4 iterations. Similar to other methods, we start with a rather stiff surface ($\mu_1 = 10$), which is then made softer ($\mu_1 = 1$ and $\mu_1 = 0.1$) in order to allow for a more precise fit. The other weights are set to $\mu_2 = 1$ and $\mu_3 = 10$. In total, the registration takes about 10 iterations, and accurately maps the smoothed source model $\mathcal{B}^{\mathrm{s}}$ onto the smoothed target model $\mathcal{B}^{\mathrm{t}}$ (see Figure 3, bottom center).

As the last step of our registration pipeline, we take the final closest point correspondences $(\mathbf{x}_i^{\mathrm{s}}, \hat{\mathbf{c}}_i^{\mathrm{t}})$ computed on the smooth meshes, and use them for matching the original source model $\mathcal{M}^{\mathrm{s}}$ to the original target model $\mathcal{M}^{\mathrm{t}}$. To this end we simply replace $\hat{\mathbf{c}}_i^{\mathrm{t}}$, which is represented by triangle index and barycentric coordinates on $\mathcal{B}^{\mathrm{t}}$, by the equivalent point $\bar{\mathbf{c}}_i^{\mathrm{t}}$ on the original target model $\mathcal{M}^{\mathrm{t}}$ in the registration energy (7). Similarly, we also replace all reference points and weights of the smoothed meshes ($\hat{A}_i^{\mathrm{s}}$, $\hat{\mathbf{x}}_i^{\mathrm{s}}$, $\hat{\mathbf{r}}_k^{\mathrm{t}}$) by their counterparts of the original meshes and set the fitting weight very high (typically 300), while keeping the other weights at 0.1. Since

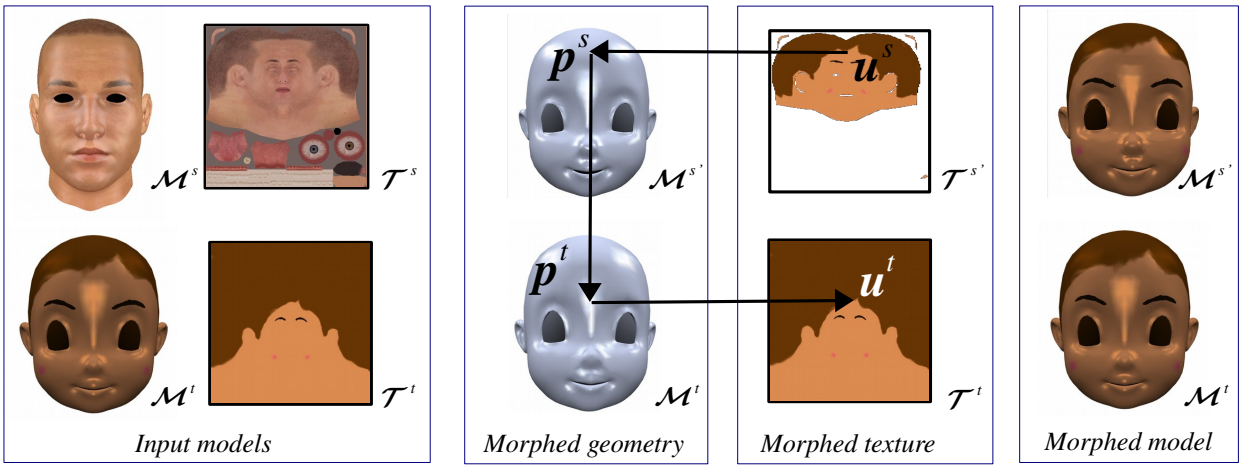

Figure 6: *From left to right: Original textured meshes as input models, morphed and target geometry, morphed and target texture, and the final morphed model.*

the correspondences computed on the smoothed meshes are very good, we only need 1 or 2 iterations on the original models to achieve the results shown in Figure 3 or Section 6.

4 Texture Matching

Once the source mesh $\mathcal{M}^{\mathrm{s}}$ has been deformed to geometrically match the target mesh $\mathcal{M}^{\mathrm{t}}$, the source texture has to be adjusted, such that the textured versions of both meshes look identical as well. In the following we denote by $\mathcal{T}^{\mathrm{s}}$ and $\mathcal{T}^{\mathrm{t}}$ the texture images of the source and target mesh, and by $\mathbf{u}_i^{\mathrm{s}}$ and $\mathbf{u}_j^{\mathrm{t}}$ their texture coordinates or uv-coordinates. The planar triangle mesh with uv-coordinates assigned as vertex positions is referred to as the uv-layout. Moreover, we now denote by source mesh $\mathcal{M}^{\mathrm{s}}$ the *deformed* source mesh after the geometric matching unless stated otherwise.

Note that there are two options for mapping the target texture onto the source mesh. We can either adjust the texture coordinates of the source mesh in order to properly access the target texture, or we can transform the target texture in order to match the uv-layout of the source mesh. However, since we later want to morph the geometry and appearance of *several* models by blending either their vertex positions or texture images, all meshes must have the same mesh connectivity and uv-layout. Consequently, we cannot adjust the texture coordinates $\mathbf{u}_i^{\mathrm{s}}$, but instead have to replace the source texture $\mathcal{T}^{\mathrm{s}}$ by a transformed version of the target texture $\mathcal{T}^{\mathrm{t}}$.

Another problem is that the uv-layouts of source and target might have incompatible seams or even consist of a different number of connected components. As a consequence, a smooth (or even continuous) 2D warp $\mathbf{f}\colon \mathcal{T}^{\mathrm{t}} \rightarrow \mathcal{T}^{\mathrm{s}}$ between source and target uv-layouts does not exist in general. We therefore perform the inverse transformation $\mathbf{f}^{-1}$ in a pixel-by-pixel manner: For each pixel $\mathbf{u}^{\mathrm{s}} \in \mathcal{T}^{\mathrm{s}}$ we find its pre-image $\mathbf{u}^{\mathrm{t}} = \mathbf{f}^{-1}(\mathbf{u}^{\mathrm{s}}) \in \mathcal{T}^{\mathrm{t}}$ and copy its color value to the source texture.

As illustrated in Figure 6, this per-pixel mapping is computed through 3D closest point correspondence of $\mathcal{M}^{\mathrm{s}}$ and $\mathcal{M}^{\mathrm{t}}$. For each pixel $\mathbf{u}^{\mathrm{s}} \in \mathcal{T}^{\mathrm{s}}$ we first find the 2D triangle covering it in the texture layout. Using barycentric coordinates with respect to this triangle, the pixel $\mathbf{u}^{\mathrm{s}}$ can be mapped to its corresponding 3D point $\mathbf{p}^{\mathrm{s}}$ on the source mesh. A BSP-accelerated closest point query reveals the corresponding point $\mathbf{p}^{\mathrm{t}}$ on the target mesh, which is finally mapped to $\mathbf{u}^{\mathrm{t}}$ using its (interpolated) texture coordinates. Because $\mathbf{u}^{\mathrm{t}}$ is a non-integer texture coordinate in general, its color value is obtained using bilinear texture interpolation and assigned to the pixel $\mathbf{u}^{\mathrm{s}}$. Although this interpolation inevitably leads to a slight texture blurring, it did not yield noticable artifacts in our examples.

This algorithm successfully transfers colors of the target texture to all pixels covered by a uv-triangle in the source texture, and it leaves blank all pixels not covered by a uv-triangle. However, round-off errors during the mapping might cause artifacts at texture boundaries, where a few pixels might be missing. We address this issue by performing a simple one-pixel dilation, i.e., by filling all transparent pixels with the average of their opaque neighbors' colors, which effectively eliminates this problem.

5 Face Morphing

The geometry and texture matching introduced in the previous sections allows us to map a source model to one or more target models. This precomputation results in a set of meshes with identical mesh connectivity and uv-layout (that of the source model), but with different shapes and texture images (that of the target models). Due to their identical connectivity and texture layout, these models are all in one-to-one vertex and pixel correspondence, which enables us to easily morph their geometries and appearances by blending their vertex positions and texture images (Figure 7, left).

For the blending between entire faces we employ simple linear interpolation of vertex positions, which we prefer over more sophisticated techniques due to its simplicity, efficiency, and the fact that for faces there are almost no visible differences between linear and nonlinear geometry interpolation. In order to blend only parts of the input models (as shown in Figure 1) we closely follow [Alexa 2003]: Instead of vertex positions we interpolate per-vertex Laplacians and solve a Poisson system for the desired vertex positions. The only difference is that instead of the uniform graph Laplacian used by Alexa, we employ the cotangent discretization [Pinkall and Polthier 1993; Meyer et al. 2003], which avoids distortion in the case of irregular meshes.

The recent perceptional study of McDonnell and colleagues [2012] demonstrates that besides shape, also the texture, material appearance, or rendering style have a significant impact on how a virtual character is perceived. For full texture blending we use simple linear interpolation, but for local blending a gradient-based technique

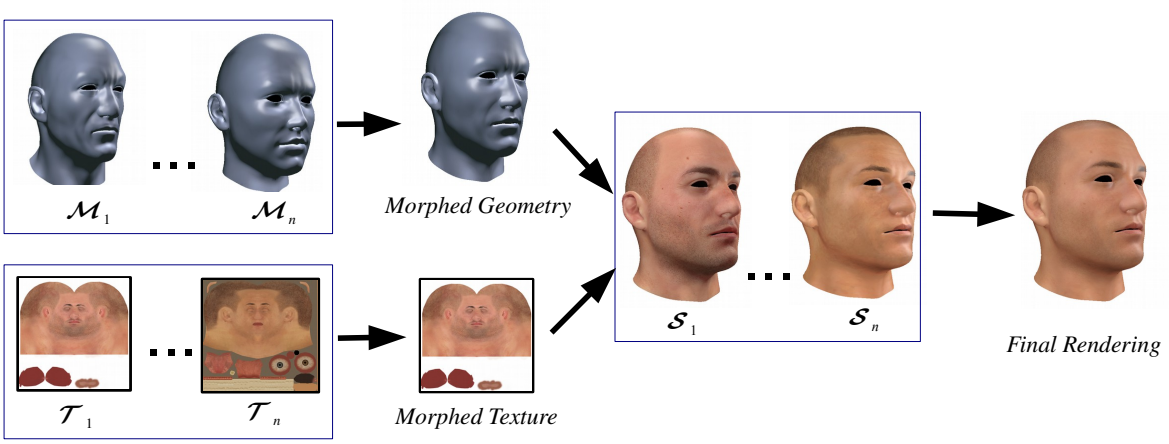

Figure 7: *Overview of the face morphing algorithm: In a first step, the face meshes ($\mathcal{M}_1, \ldots, \mathcal{M}_n$) and textures ($\mathcal{T}_1, \ldots, \mathcal{T}_n$) are interpolated. The resulting interpolated mesh is rendered with the interpolated texture into several off-screen buffers using different rendering styles ($\mathcal{S}_1, \ldots, \mathcal{S}_m$). At the end the final rendering is obtained by blending between the off-screen buffers.*

should be employed [Pérez et al. 2003]. Additionally, we incorporate several real-time rendering styles into our face morphing application, which model a wide range of effects, ranging from realistic skin [d'Eon et al. 2007], over illustrative game characters [Mitchell et al. 2007], to NPR shading [Barla et al. 2006], as shown in Figure 9 and the accompanying video.

In our face morphing application the user can interactively adjust the blending weights, and the face rendering is adjusted in real time. Our pipeline for blending geometry, texture, and rendering style is illustrated in Figure 7. After interpolating vertex positions, normal vectors, and texture images, the resulting morphed model is rendered into a set of off-screen buffers using all active rendering styles, and the final image is a simple linear interpolation of these off-screen buffers. Since most computations are performed on the GPU using a combination of OpenCL and GLSL, the morphing can be performed in real-time even for complex models (see the accompanying video).

6 Results & Discussion

In order to evaluate the capabilities of our face matching framework, we first present qualitative registration and morphing results for several characters, then provide a quantitative analysis of fitting accuracy and computational time, and finally discuss and compare to related work.

Figure 8 demonstrates that our system is capable of robustly mapping a source face model to a set of target meshes ranging from realistic to highly abstract characters. These results have been obtained by five iterations of joint fairing and five iterations of the geometric registration. Besides accurately mapping the face geometries, it also successfully transfers the textures from target to source. After mapping the source model to these four target models, we can blend between the shapes and appearances of the deformed source models, which now have identical mesh connectivity and texture layouts. Figure 9 shows the results of 50%-morphs between these four characters.

In order to push our system to its limits we sculpted several clay faces, with strong deformation as they might appear in cartoon animation. These clay figures have been 3D-scanned and converted to a blend-shape model by matching the neutral model (source) to the other expressions (targets). Photographs of the clay models and

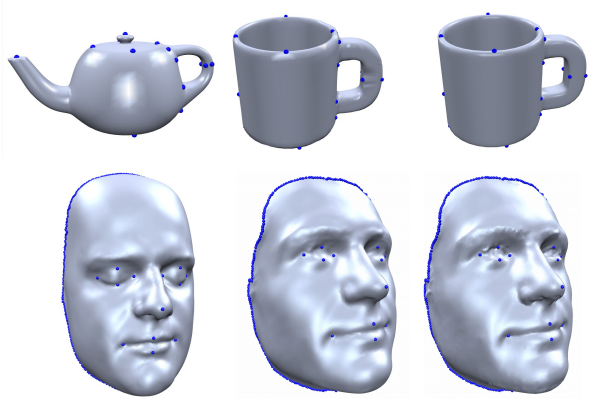

Figure 11: *Fitting results for an object of genus 1 (top) and for face scans with disk topology (bottom). From left to right: source model, fitting result, target model.*

rendering of the blend shapes are shown in Figure 10. As mentioned in Section 3.2, for these extreme examples the joint fairing did not unfold sufficiently with the default weights. However, increasing the smoothing weight allowed for the successful matching shown in Figure 10 and the accompanying video.

Although our method has been developed primarily for faces, which typically are genus 0 objects with holes and boundaries, Figure 11 demonstrates on a teapot-to-cup morph that our method also works for higher genus models. Of course, the method also works for objects of disk-topology, such as the face scans shown in Figure 11.

Fitting	n	m	Time	Error
Viktor to Dave	19k	10k	7s	0.5%
Viktor to Loki	19k	6.7k	5s	0.4%
Viktor to Girl	19k	2.8k	5s	0.3%
Viktor to Kissmouth	19k	7.8k	7s	0.3%

Table 1: *Statistics for different mapping examples, listing number of source and target vertices n and m, total time for fitting, and relative Hausdorff distance.*

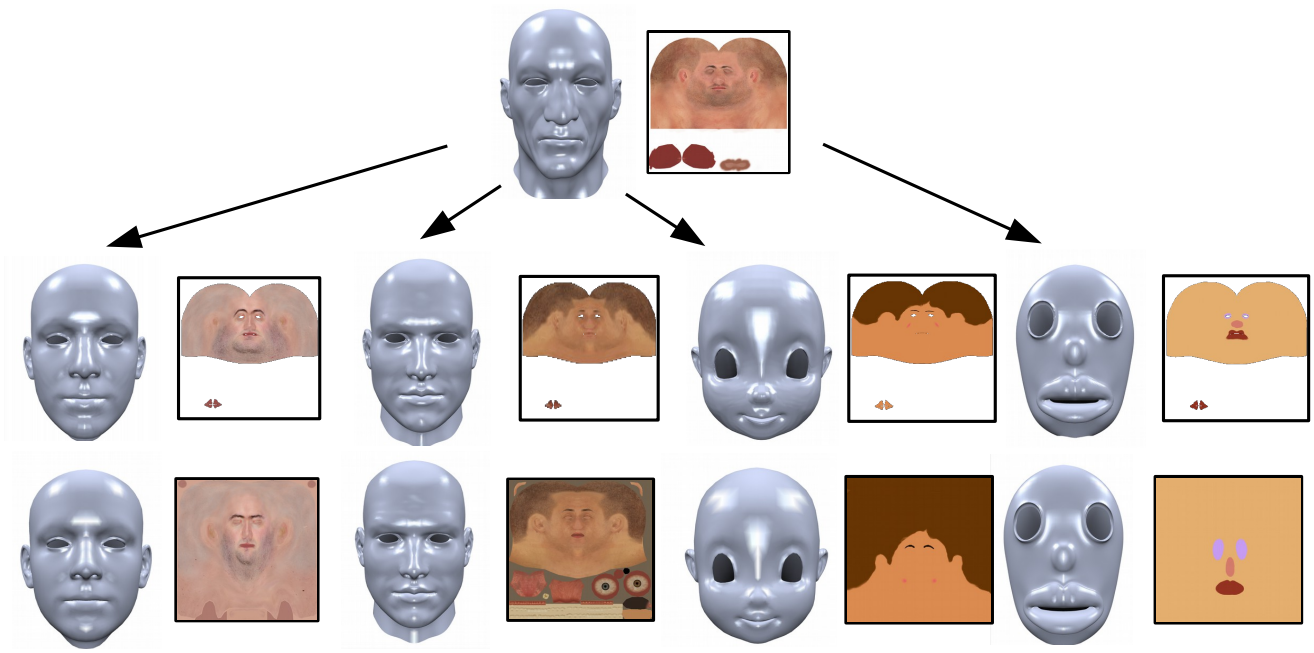

Figure 8: *Mapping geometry and texture of the source model Viktor (top) to four other faces, ranging from realistic to highly abstract (from left to right: Loki, Dave, Girl, Kissmouth). The target models are shown in the bottom row, the morphed source models in the middle row.*

Figure 9: *Different models and rendering styles (odd columns) and 50%-blends between them (even columns).*

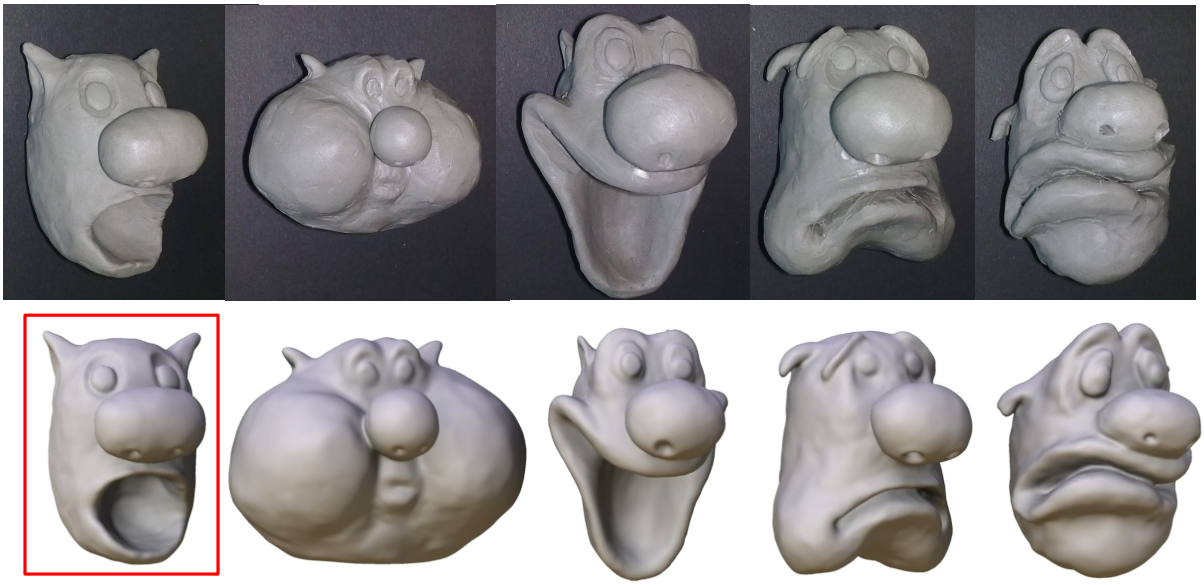

Figure 10: *An application example to demonstrate the possibilities of our face matching technique. Clay faces with extreme deformations have been sculpted, scanned, and transformed into a blend-shape model by matching the neutral model (red frame) to the other expressions. The top row shows photographs of the clay models, the bottom row shows the resulting blend shapes.*

In order to quantitatively evaluate our method, we give performance numbers and fitting accuracies in Table 1. The timings of our non-optimized single-threaded implementation were measured on a MacPro with Intel Xeon 2.67GHz. For all synthetic face models the fitting took just a few seconds, and even for the high resolution scanned models (Figures 10,11) it only took about 30 seconds. The computational cost is dominated by the computation of closest point correspondences, which due to the hierarchical kD-tree is $O(n \log m)$. Solving the linear systems using sparse Cholesky factorization is close to $O(n)$. Note that our simple and efficient method allows to optimize the position of each source vertex, which leads to highly accurate fits with errors (in terms of Hausdorff distance) well below 1% of the bounding box diagonal.

As mentioned in Section 2 there are many approaches for mapping one model to another. In Figure 12 we compare our method to the non-rigid registration of Weise [2011] and blended intrinsic maps [Kim et al. 2011]. Blended intrinsic maps compute correspondences in a fully automatic manner, but fail for all our examples, presumably because face models are highly symmetric and the geometric features are less prominent than the protruding arms and legs shown in the original paper [Kim et al. 2011].

To allow for a fair comparison to [Weise et al. 2011], we manually specified more feature correspondences at the boundaries for Weise's method in order to compensate for our automatic boundary correspondences (cf. Figure 5). While their results are similar to ours for near-isometric models (Figure 12, bottom row), self-intersections in the nose or mouth area can be observed for the the first and second row. For the Slimer example (third row) their method yields wrong alignments near the self-intersecting regions of the chin and neck of the target model.

We would have also liked to compare to [Kraevoy and Sheffer 2004], but even with strong involvement of the authors we could not restore a completely working version of their software from original code fragments. Our final implementation is able to produce the initial cross-parametrization, but cannot perform the subsequent smoothing of the parameterization. As a consequence, we cannot produce full cross-parameterizations similar to the examples shown in Figure 12. However, we can evaluate how many additional vertices have to be inserted during their initial path creation step. When testing the top row example of Figure 12, the numbers of vertices for the source and target meshes increase from 7224 and 2911 to 8611 and 4071, respectively. This massive change in connectivity strongly contradicts our goals described in Section 1.

Hence, compared to other approaches for establishing correspondence our method has a few important advantages. First, in contrast to cross-parametrization techniques [Kraevoy and Sheffer 2004; Schreiner et al. 2004] it does not require inserting new vertices or edges, which would prevent the compatible fitting to multiple target models (Figure 8). Second, in contrast to parametrization-based methods [Blanz and Vetter 1999; Lipman and Funkhouser 2009] it can handle models of higher genus (Figure 11). Third, in contrast to most non-rigid registration techniques [Huang et al. 2008; Weise et al. 2011] it can handle highly non-isometric input models, where existing methods often produce self-intersections (Figure 12). Finally, our method is easier to implement, very robust, and more efficient than most other approaches.

A limitation of our technique is that the aggressive fairing might collapse protruding extremities, such as arms, fingers, or legs. Because of this the subsequent registration is not capable to determine correct correspondences, which results in strongly distorted triangles (see Figure 13). Although this collapsing could be avoided by manually specifying additional reference points, this would require considerably more work for the user.

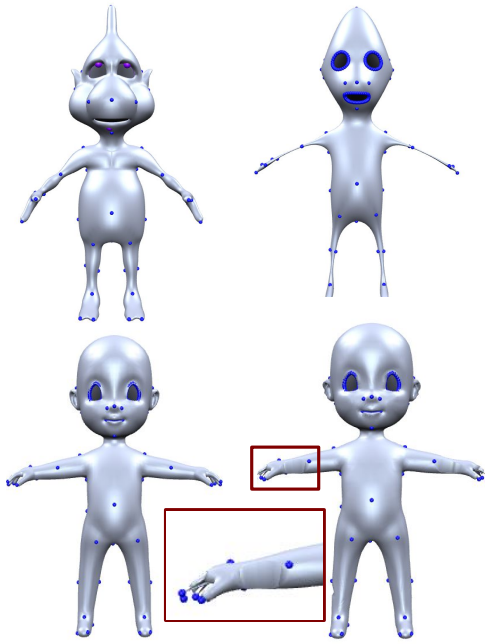

Figure 13: *When fitting the source character (top left) to the target character (bottom left), our method fails at extremities. These tend to collapse during the joint fairing (top right), thus causing wrong correspondences for the subsequent fitting and leading to distorted triangles in the fitting result (bottom right).*

7 Conclusion

With ELASTIFACE we presented a novel method for establishing correspondences between textured face models. The strength of our approach is its simplicity, robustness, and performance. We have shown that our method is more suitable for fitting of non-isometric objects than recent non-rigid registration techniques. Additionally, we have presented a robust extension for matching arbitrary texture layouts. In the future we would like to extend our framework by adding eyes, teeth, and tongue to the face models. Another promising direction for future work is the extension from static face models to dynamic facial animations.

Acknowledgments

We are grateful to the authors of [Weise et al. 2011] for providing an executable of their method, the authors of [Kim et al. 2011] for publishing full source code, and the authors of [Kraevoy and Sheffer 2004] for providing source code and helping with debugging. The Loki and Slimer models are courtesy of Weise et al. [2011], the Viktor model is courtesy of Faceware, the left scan of Figure 11 is courtesy of Infinite Realities. Characters of Figure 13 and the Girl model are from Freebie 3D and Creative Crash.

References

ALEXA, M. 2002. Recent advances in mesh morphing. *Computer Graphics Forum 21*, 2, 173–198.

ALEXA, M. 2003. Differential coordinates for mesh morphing and deformation. *The Visual Computer 19*, 2, 105–114.

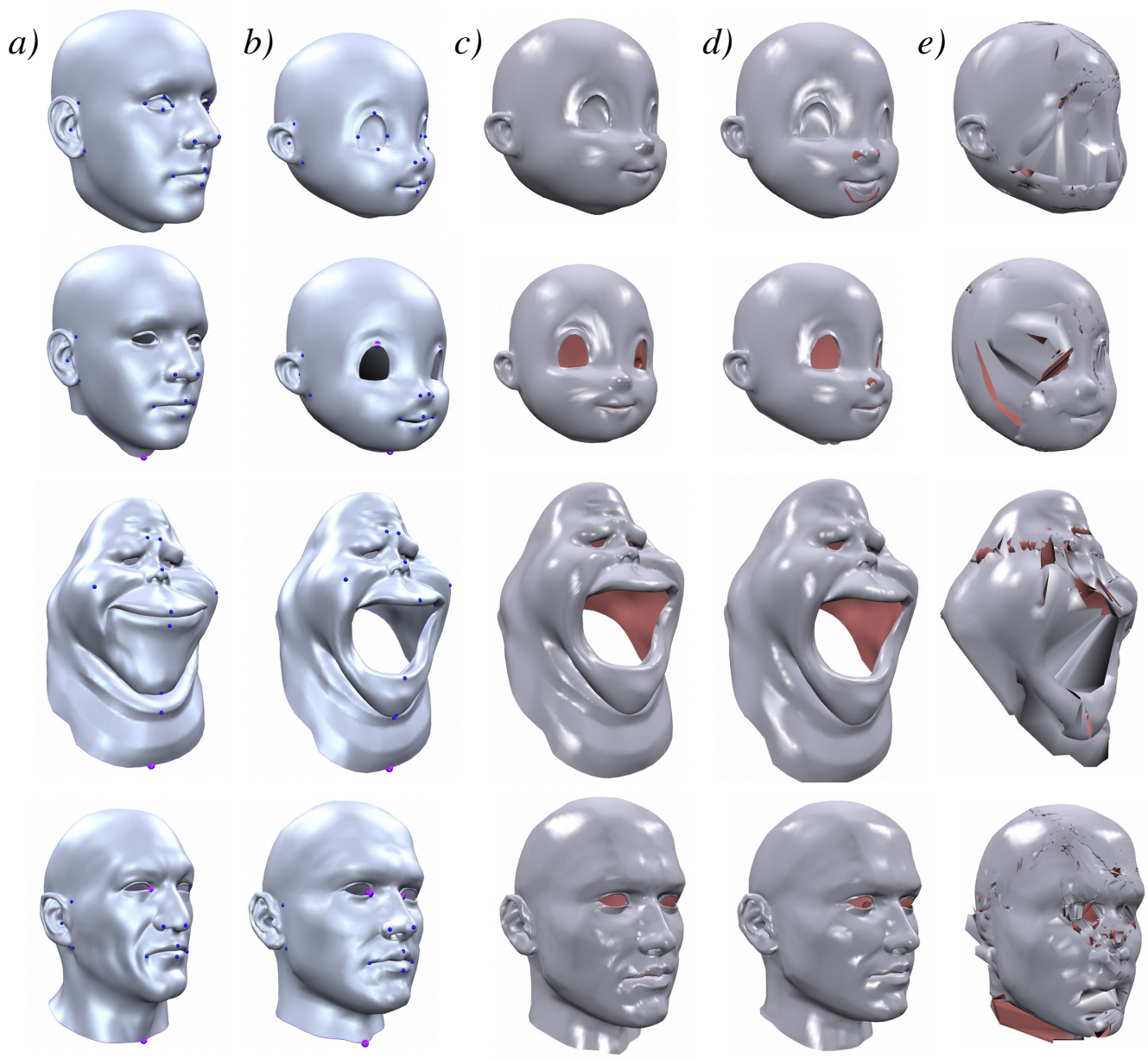

Figure 12: *Results of fitting source models (column a) to target models (column b). While our method (column c) handles all examples without problems, the method of Weise et al. [2011] (column d) causes self-intersections around the nose for the upper two examples and around the mouth for the third example. The blended intrinsic maps [Kim et al. 2011] (column e) are not suitable for face matching.*

ALLEN, B., CURLESS, B., AND POPOVIĆ, Z. 2003. The space of human body shapes: reconstruction and parameterization from range scans. *ACM Transactions on Graphics (Proc. SIGGRAPH) 22*, 3, 587–594.

AMBERG, B., ROMDHANI, S., AND VETTER, T. 2007. Optimal step nonrigid ICP algorithms for surface registration. In *Proceesings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 1–8.

BARLA, P., THOLLOT, J., AND MARKOSIAN, L. 2006. X-toon: an extended toon shader. In *Proceedings of Symposium on Non-photorealistic animation and rendering (NPAR)*, 127–132.

BESL, P. J., AND MCKAY, N. D. 1992. A method for registration of 3-d shapes. *IEEE Transactions on Pattern Analysis and Machine Intelligence 14*, 2, 239–256.

BLANZ, V., AND VETTER, T. 1999. A morphable model for the synthesis of 3d faces. In *Proceedings of ACM SIGGRAPH*, 187–194.

BRONSTEIN, A., BRONSTEIN, M., AND KIMMEL, R. 2006. Generalized multidimensional scaling: A framework for isometry-invariant partial surface matching. *Proceedings of the National Academy of Sciences (PNAS) 103*, 5, 1168–1172.

BRONSTEIN, A., BRONSTEIN, M., AND KIMMEL, R. 2008. *Numerical Geometry of Non-Rigid Shapes*. Springer.

BUI, T. D., POEL, M., HEYLEN, D., AND NIJHOLT, A. 2003. Automatic face morphing for transferring facial animation. In *Proc. of IASTED International Conference on Computers, Graphics, and Imaging*, 19–24.

CHANG, W., LI, H., MITRA, N., PAULY, M., AND WAND, M. 2010. Geometric registration for deformable shapes. In *Eurographics 2010 course*.

CHEN, Y., DAVIS, T. A., HAGER, W. W., AND RAJAMANICKAM, S. 2008. Algorithm 887: Cholmod, supernodal sparse cholesky factorization and update/downdate. *ACM Transactions on Mathematical Software 35*, 3, 1–14.

D'EON, E., LUEBKE, D., AND ENDERTON, E. 2007. Efficient rendering of human skin. In *Eurographics Workshop on Rendering*, 147–158.

HORN, B. K. P. 1987. Closed-form solution of absolute orientation using unit quaternions. *Journal of the Optical Society of America 4*, 629–642.

HUANG, Q.-X., ADAMS, B., WICKE, M., AND GUIBAS, L. J. 2008. Non-rigid registration under isometric deformations. *Computer Graphics Forum (Proc. SGP) 27*, 5, 1449–1457.

KAZHDAN, M., SOLOMON, J., AND BEN-CHEN, M. 2012. Can mean-curvature flow be modified to be non-singular? *Computer Graphics Forum (Proc. SGP) 31*, 5, 1745–1754.

KIM, V. G., LIPMAN, Y., AND FUNKHOUSER, T. 2011. Blended intrinsic maps. *ACM Transactions on Graphics (Proc. SIGGRAPH) 30*, 4.

KRAEVOY, V., AND SHEFFER, A. 2004. Cross-parameterization and compatible remeshing of 3d models. *ACM Transactions on Graphics (Proc. SIGGRAPH) 23*, 3, 861–869.

KRAEVOY, V., SHEFFER, A., AND GOTSMAN, C. 2003. Matchmaker: constructing constrained texture maps. *ACM Transactions on Graphics (Proc. SIGGRAPH) 22*, 3, 326–333.

LEE, W.-S., AND MAGNENAT-THALMANN, N. 2000. Fast head modeling for animation. *Journal of Image and Vision Computing 18*, 4, 355–364.

LÉVY, B. 2001. Constrained texture mapping for polygonal meshes. In *Proceedings of ACM SIGGRAPH*, 417–424.

LI, H., SUMNER, R. W., AND PAULY, M. 2008. Global correspondence optimization for non-rigid registration of depth scans. *Computer Graphics Forum (Proc. SGP) 27*, 5, 1421–1430.

LIPMAN, Y., AND FUNKHOUSER, T. 2009. Möbius voting for surface correspondence. *ACM Transactions on Graphics (Proc. SIGGRAPH) 28*, 3, 72:1–72:12.

MCDONNELL, R., BREIDT, M., AND BÜLTHOFF, H. H. 2012. Render Me Real?: Investigating the Effect of Render Style on the Perception of Animated Virtual Humans. *ACM Transactions on Graphics (Proc. SIGGRAPH) 31*, 4, 91:1–91:11.

MEYER, M., DESBRUN, M., SCHRÖDER, P., AND BARR, A. H. 2003. Discrete differential-geometry operators for triangulated 2-manifolds. In *Visualization and Mathematics III*, H.-C. Hege and K. Polthier, Eds. Springer-Verlag, Heidelberg, 35–57.

MITCHELL, J. L., FRANCKE, M., AND ENG, D. 2007. Illustrative rendering in Team Fortress 2. In *ACM SIGGRAPH courses*.

NOH, J.-Y., AND NEUMANN, U. 2001. Expression cloning. In *Proceedings of ACM SIGGRAPH*, 277–288.

OVSJANIKOV, M., MERIGOT, Q., MEMOLI, F., AND GUIBAS, L. 2010. One point isometric matching with the heat kernel. *Computer Graphics Forum (Proc. SGP) 29*, 5, 1555–1564.

PÉREZ, P., GANGNET, M., AND BLAKE, A. 2003. Poisson image editing. *ACM Transactions on Graphics (Proc. SIGGRAPH) 22*, 3, 313–318.

PINKALL, U., AND POLTHIER, K. 1993. Computing discrete minimal surfaces and their conjugates. *Experiment. Math. 2*, 1, 15–36.

SCHREINER, J., ASIRVATHAM, A., PRAUN, E., AND HOPPE, H. 2004. Inter-surface mapping. *ACM Transactions on Graphics (Proc. SIGGRAPH) 23*, 3, 870–877.

TEVS, A., BOKELOH, M., WAND, M., SCHILLING, A., AND SEIDEL, H.-P. 2009. Isometric registration of ambiguous and partial data. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 1185–1192.

UMEYAMA, S. 1991. Least-squares estimation of transformation parameters between two point patterns. *IEEE Transactions on Pattern Analysis and Machine Intelligence 13*, 4, 376–380.

VAN KAICK, O., ZHANG, H., HAMARNEH, G., AND COHEN-OR, D. 2011. A survey on shape correspondence. *Computer Graphics Forum 30*, 6, 1681–1707.

WANG, Y., GUPTA, M., ZHANG, S., WANG, S., GU, X., SAMARAS, D., AND HUANG, P. 2008. High resolution tracking of non-rigid motion of densely sampled 3d data using harmonic maps. *International Journal of Computer Vision 76*, 3, 283–300.

WEISE, T., LI, H., VAN GOOL, L., AND PAULY, M. 2009. Face/Off: Live facial puppetry. In *Proceedings of ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, 7–16.

WEISE, T., BOUAZIZ, S., LI, H., AND PAULY, M. 2011. Real-time performance-based facial animation. *ACM Transactions on Graphics (Proc. SIGGRAPH) 30*, 4, 77:1–77:10.